Investigation of the Analytical Model of a Queue to Several Servers of Computer Networks with Priority Maintenance

Zakir Huseynov, Zaman Zeynalov, Bahariyya Mammadova, Anar Guliyev and Gultekin Gadimli

ABSTRACT

BACKGROUND

The performance of modern computer networks depends significantly on the number of nodes, available channels, and waiting slots. Efficient modeling of these factors is critical for optimizing resource allocation, reducing service delays, and maintaining high-quality metrics under dynamic loads.

MATERIALS AND METHODS

This study adopts a quantitative approach and aims to develop and validate an analytical model for multi-node, multi-server computer networks with finite queues and priority-based servicing. The model seeks to identify functional dependencies between nodes, channels, and waiting slots and to quantify their impact on key performance indicators, including waiting time, rejection probability, and server utilisation. An analytical model of a priority queueing system was constructed, incorporating finite queues, absolute priority for critical requests, and variable load conditions. Simulation experiments were performed under different network configurations to validate the theoretical dependencies.

RESULTS

Simulation results demonstrated that increasing the number of nodes from 10 to 100 raised average waiting time from 0.8s to 8s and rejection probability from 5% to 60%. Priority analysis revealed that under absolute priority at $\rho = 0.99$, critical requests were processed in 2.1s on average, while standard requests required 8s, with rejection rates staying within 16%. Additionally, increasing the number of servers from 2 to 5 at fixed load ($\rho = 0.8$) raised the average number of requests in the system from 4 to 6, but reduced service time variability, as the standard deviation-to-mean ratio decreased from 2.6 to 1.3.

CONCLUSION

The proposed model effectively captures the interplay between network nodes, channels, and waiting slots in determining quality of service under variable loads. The findings provide practical insights for optimizing network configurations to balance efficiency, fairness, and reliability in modern computer systems.

Keywords: Load management, Performance evaluation, Rejection probability, Request processing, Resource allocation, Waiting time

Highlights

- Analytical model developed to evaluate network performance under finite queues, service priorities, and dynamic loads.
- Functional relationships established between nodes, channels, and waiting slots affecting key performance metrics.
- Simulations confirmed that high load accelerates critical request processing but delays standard requests.

Increasing server numbers reduced service time variability and improved overall system stability.

Department of Information Technologies, Azerbaijan State Agricultural University, 450 Ataturk Ave., Ganja, Azerbaijan

Correspondence to:
Zakir Huseynov,
huseynovzakir75@gmail.com

Additional material is published online only. To view please visit the journal online.

Cite this as: Huseynov Z, Zeynalov Z, Mammadova B, Guliyev A and Gadimli G. Investigation of the Analytical Model of a Queue to Several Servers of Computer Networks with Priority Maintenance. Premier Journal of Science 2025;14:100117

DOI: https://doi.org/10.70389/PJS.100117

Peer Review

Received: 5 September 2025

Last revised: 19 September 2025

Accepted: 19 September 2025

Version accepted: 2

Published: 3 October 2025

Ethical approval: N/a

Consent: N/a

Funding: No industry funding

Conflicts of interest: N/a

Author contribution:
Zakir Huseynov – conceptualization, methodology, supervision, writing – original draft preparation. Zaman Zeynalov – software, validation, data curation, visualization. Bahariyya Mammadova – investigation, formal analysis, resources, writing – reviewing and editing.

Introduction

The advancement of digital technologies is closely tied to the efficiency of computer networks, which not only enable communication between devices but also form the backbone of distributed systems, cloud computing, and other modern technologies. Effective resource management within these networks is crucial for ensuring rapid data access and reliable storage, which are essential for maintaining high-quality service in an increasingly digital world.

In contemporary computer networks, one of the key challenges is managing request servicing, which is often accompanied by significant delays, particularly under high-load conditions.[1] These delays are exacerbated when requests with varying priorities – such as urgent or real-time requests – are processed. As the demand for both efficiency and reliability continues to grow, it becomes imperative to develop models that optimise resource allocation and minimise delays, especially for critical requests that require immediate attention.[2]

Queueing systems (QS) form the foundation for analysing request processing across various fields, including telecommunications, logistics, and computer networks.[3] These mathematical models are instrumental in describing key system performance metrics such as resource allocation, waiting periods, request blocking probability, and server load. A considerable amount of attention has been given to modelling multi-server systems that can handle requests with different priority levels, as this is essential for improving overall system performance.

Classical queueing models like M/M/n are commonly used to analyse multi-server systems. Multiple servers, exponential interarrival durations, and Poisson request flows are considered in these models. With the assumption that events occur independently and at a constant average rate, the Poisson distribution explains the chance of a certain number of events occurring in a specified region of time or space. This paradigm is useful for modelling random arrival patterns in queueing systems, where events like incoming requests occur intermittently but with a known average rate.

In queueing theory, the Erlang distribution is utilised alongside the Poisson distribution. The Erlang distribution describes waiting times in multi-channel

Anar Guliyev – methodology, project administration, validation, writing – reviewing and editing. Gultekin Gadimli – data curation, software, visualization, writing – reviewing and editing. All authors read and approved the final manuscript

Guarantor: Zakir Huseynov

Provenance and peer-review: Unsolicited and externally peer-reviewed

Data availability statement: The data that support the findings of this study are available on request from the corresponding author

systems with exponential inter-arrival and service delays. It calculates the probability that a request will wait in a queue based on server count and system load. Erlang is essential for monitoring system performance, especially in multi-server networks where active servers may delay requests.

Classical queueing models, such as the M/M/n type, have been widely applied in the analysis of multi-server systems.[4] These models consider multiple servers, exponentially distributed interarrival times, and Poisson-distributed request flows, providing a foundational understanding of system performance and resource allocation.[5] However, such models typically assume homogeneous service conditions and do not account for dynamic factors such as varying request priorities or load fluctuations, which are increasingly common in modern computer networks.[5]

To address these limitations, this study introduces a novel analytical model that extends classical M/M/n queueing theory by incorporating dynamic priority servicing and adaptive server scaling. These mechanisms allow for the efficient processing of both critical and standard requests, ensuring minimal delays for high-priority tasks while optimising overall system performance. Unlike traditional models, our approach adapts to real-time load fluctuations, making it highly relevant for modern, high-demand network environments. The core hypothesis is that as the number of nodes in a network increases, service resources must be proportionally expanded to maintain a consistent quality of service. Additionally, absolute priority servicing is assumed to significantly reduce waiting times for critical requests without adversely affecting the processing of non-critical requests.

The implementation of priority-based servicing introduces additional complexity and practical relevance to these systems.[7] This approach involves categorising requests by importance, with critical requests receiving higher priority, which can be implemented through absolute or relative priority mechanisms. By applying advanced analytical models based on queueing theory, it is possible to predict system behaviour under varying conditions and formulate effective strategies to enhance the performance of computer networks that employ priority-based servicing.[8]

Several studies have explored the performance of multi-server systems with priority servicing. For example, the work of Elliriki et al.[9] examined a multi-channel queueing system that accounts for server failures and recovery strategies, though it did not consider priority-based servicing for different request classes, which is crucial for multi-node networks under dynamic loads. Similarly, Huseynov et al.[10] made significant progress in studying the performance of multi-channel and multi-node computer networks with priority servicing under variable loads. The researchers calculated average request waiting times and server utilisation, enabling an assessment of system efficiency. However, the study does not account for dynamic priority adjustments in real time, limiting its applicability to real-world systems where conditions may change dynamically.

Additionally, research on M/M/1/∞ queueing systems by Melikov and Shahmaliyev[11] focused on factors like perishable inventory and repeat customers, providing insights into system performance under these conditions. However, their model does not address variable load or finite queues, which are essential components of real-world systems, particularly in modern computer networks that must handle dynamic and unpredictable traffic.

The objective of this study was to develop an analytical model for assessing the performance quality of modern computer networks, considering parameters such as the number of channels, waiting slots in network nodes, and the total number of nodes. One hypothesis of the research is that as the number of nodes increases, service resources must be proportionally expanded to maintain a specified quality level. Additionally, it is assumed that implementing absolute priority servicing can minimise waiting times for critical requests without significantly affecting the servicing of other requests.

Materials and Methods

Relevant statistics were obtained for the M/M/H model, in which the exponential distribution of service time is identical for all servers. This model accounts for both the general characteristics of the system and specific aspects related to priority servicing, which is crucial for ensuring efficient network operation under high-load conditions.

In the constructed model, request arrivals follow a Poisson process with intensity λ, and service times are exponentially distributed with rate μ. The system consists of H identical servers and a finite waiting buffer of size B. The traffic load is defined as $\rho = \lambda/(H\mu)$. Requests are divided into two classes: critical and standard. Critical requests are assigned higher priority and are serviced first, which affects the overall system performance. However, for a comprehensive system analysis, it is necessary to first examine general metrics that consider all request types. This allows for an evaluation of the system's overall efficiency and an assessment of how priority servicing influences its operation. Within each class, the queue discipline is First-Come-First-Served (FCFS). The buffer policy is of finite capacity: if the system already contains $H+B$ requests, any new arrival is blocked and lost. Thus, blocking probability P_{block}, mean waiting time W, and mean number of requests in the system L can be derived using standard Erlang loss and delay formulas adapted for two classes under priority.

This model follows the Poisson distribution law and the Erlang distribution law (1, 2):

$$K = \frac{\sum_{I=0}^{H-1} \frac{(H\rho)^I}{I!}}{\sum_{I=0}^{H} \frac{(H\rho)^I}{I!}} \quad (1)$$

where: K – Poisson coefficient function, H – number of servers, ρ — system load.

$$C = \frac{1-K}{1-\rho K} \quad (2)$$

where: C – Erlang function, K – Poisson coefficient functionρ — system load.

The Poisson coefficient K is used in the model to calculate the probability that all servers are busy. The Erlang function C, in turn, indicates the probability that a request will enter the queue.

The change in the functional dependence between the Poisson coefficient K and the Erlang function C relative to the system load (ρ) is illustrated in Figure 1.

The Erlang function C is a probabilistic quantity, the magnitude of which always lies within the range of 0 to 1. As can be observed, the value of C is a function of the number of servers H and the utilisation coefficient (load) ρ (3):

$$C(H,\rho) = \frac{1-K(H,\rho)}{1-\rho K(H,\rho)} \quad (3)$$

where: C – Erlang function, K – Poisson coefficient function, H – number of servers,ρ — system load.

The change in load in a multi-server system was determined using the following formula (4):

$$\rho = \frac{\lambda T_s}{H} \quad (4)$$

where: λ – packet transmission rate per second (packets/s), T_s – average service time per request (excluding queue waiting time), H – number of servers.

The average number of requests in the system (waiting and being serviced) was calculated using the formula (5):

$$r = C\frac{\rho}{1-\rho} + H\rho \quad (5)$$

where: C – Erlang function, H – number of servers,ρ — system load.

The average number of requests waiting to be serviced (w) was determined by the formula (6):

$$w = C\frac{\rho}{1-\rho} \quad (6)$$

where: C – Erlang function,ρ — system load.

The average request time (T_r) and average waiting time in the system (T_w) were calculated using the following formulas (7, 8):

$$T_r = \frac{C}{H}\cdot\frac{T_s}{1-\rho} + T_s \quad (7)$$

where: C – Erlang function, H – number of servers, ρ — system load, T_s – average service time per request.

$$T_w = \frac{C}{H}\cdot\frac{T_s}{1-\rho} \quad (8)$$

where: C – Erlang function, H – number of servers, ρ — system load, T_s – average service time per request.

The standard deviations of T_r and T_w were determined by the following formulas, respectively (9):

$$\sigma_{T_r} = \frac{T_s}{H(1-\rho)} + T_s \quad (9)$$

where: H – number of servers,ρ — system load, T_s – average service time per request (10).

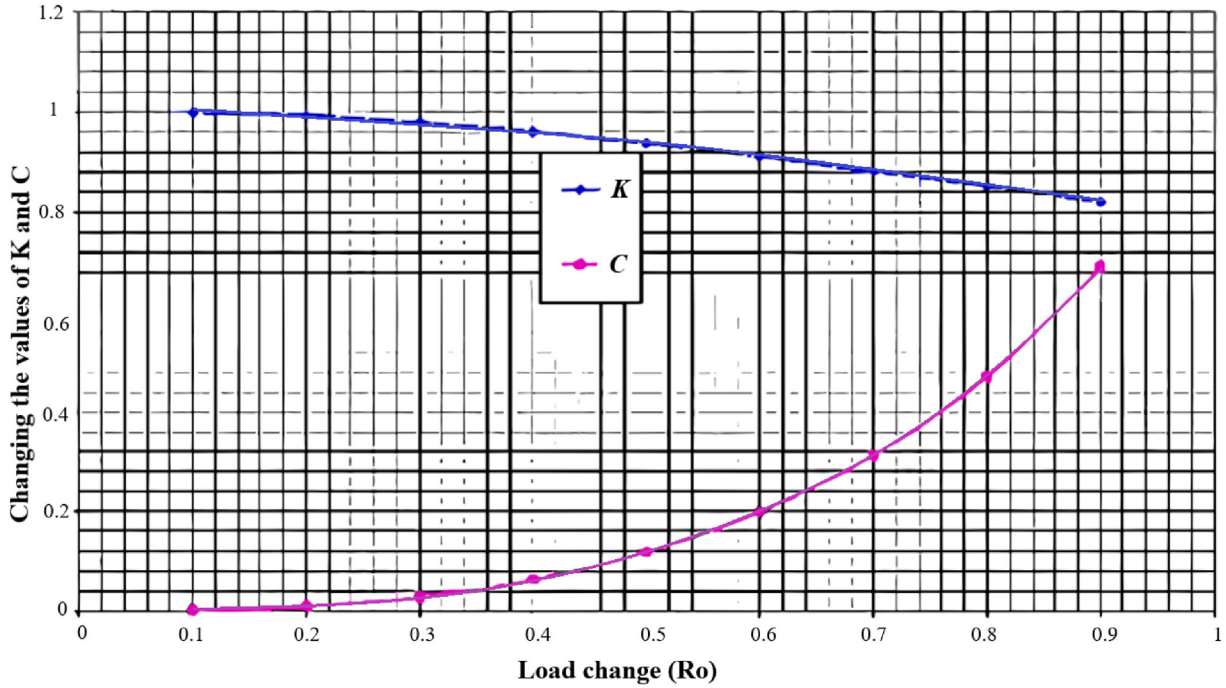

Fig 1 | Graphs of changes in the Poisson coefficient function and the Erlang function
Source: Compiled by the authors.

$$\sigma_{T_w} = \frac{1}{1-\rho}\sqrt{C\rho(1+\rho)-C\rho} \quad (10)$$

where: H – number of servers,ρ — system load, T_s – average service time per request.

The calculation of the average waiting time T_d (excluding requests with zero waiting time) was performed using the formula (11):

$$T_d = \frac{T_s}{H(1-\rho)} \quad (11)$$

where: H – number of servers,ρ — system load, T_s – average service time per request.

For the M/M/H model with a limited queue, the rejection probability $P_{rejection}$ was defined as the probability that all servers are busy and the queue is full. The rejection probability formula is as follows (12):

$$P_{rejection} = \frac{\frac{(\lambda T_s)^H}{H!}\cdot\frac{1-(\lambda/(H/T_s)^{k+1}}{1-(\lambda/(\frac{H}{T_s})}}{\sum_{i=0}^{H-1}\frac{(\lambda T_s)^i}{i!}+\frac{(\lambda T_s)^H}{H!}\cdot\frac{1-(\lambda/(\frac{H}{T_s}))^{k+1}}{1-(\lambda/(\frac{H}{T_s}))}} \quad (12)$$

where: H – number of servers, λ – packet transmission rate per second, T_s – average service time per request, k – queue size.

To ensure analytical correctness, the derived expressions were cross-checked against standard results from queueing theory. In particular, when the queue length tends to infinity, the rejection probability (12) reduces to the classical Erlang-C formula for the M/M/c model.[13] Similarly, for $k = 0$, the model reduces to Erlang function (C) blocking probabilities. For systems with priority scheduling, the derived expressions converge to the preemptive-resume M/M/c priority model described in Harchol-Balter.[12] These limiting cases validate the internal consistency of the analytical framework.

The analytical model developed in this study is fully presented within the article, together with all relevant formulas and calculation procedures. Typical parameter values of the system were applied to analyse the functional dependencies between load, waiting time, blocking probability, and server utilisation. This ensured that the determination of optimal performance indicators is transparent and reproducible directly from the analytical framework provided in the text.

Results

Modern telecommunications networks employ various approaches to request servicing. The two most common structural types are: systems with a shared queue for all servers and systems with dedicated queues for each server.

As seen in Figure 1, for $H = 2$, the changes in K values are negligible, while with an increase in the average number of requests in the system, these values decrease and fluctuate between 0.85 and 1. In this case, the values of C range from 0.1 to 0.7. Figure 2 presents a model with a shared queue, where all incoming requests accumulate in a single queue before being distributed among available servers. This approach ensures balanced workload distribution among servers but may cause delays under high-load conditions.

Figure 3 illustrates a system structure with dedicated queues for each server. In this model, servers process only those requests that enter their respective queues. This approach ensures server independence but may lead to workload imbalance, particularly under uneven request distribution.

The analysis of the relationship between the Poisson coefficient *(K)* and the Erlang function *(C)* as a function of system load (ρ) is crucial for assessing the performance of a queuing system. The Poisson coefficient determines the probability that all servers are busy, while the Erlang function indicates the probability of a request entering the queue. These metrics help understand how the system responds to changes in load

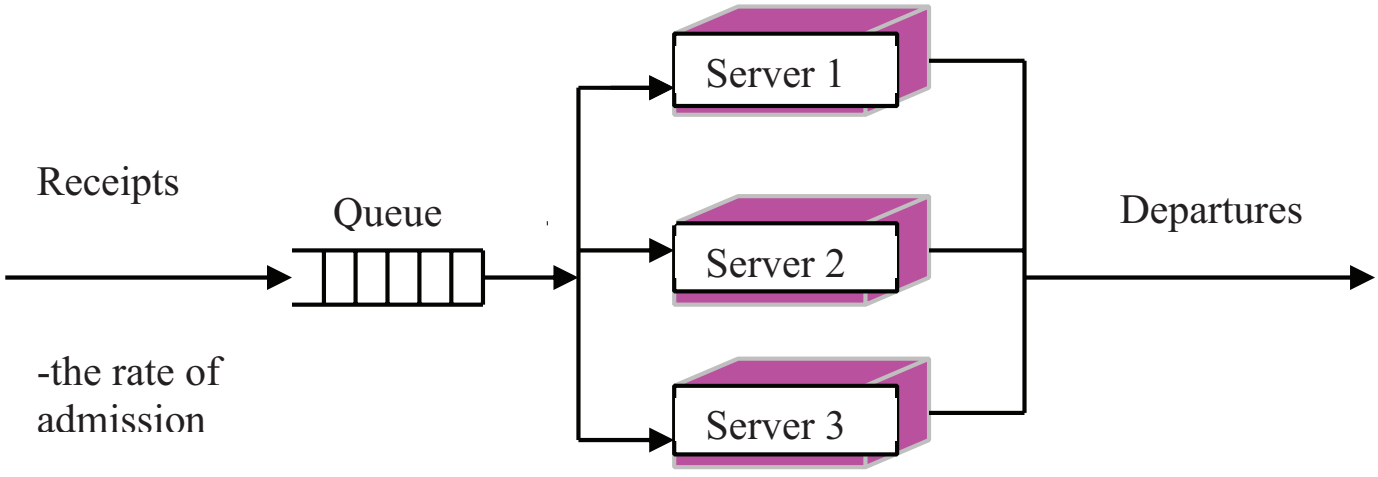

Fig 2 | Structure and parameters of a queuing system with a shared queue
Source: Compiled by the authors.

DOI: https://doi.org/10.70389/PJS.100117 | Premier Journal of Science 2025;14:100117

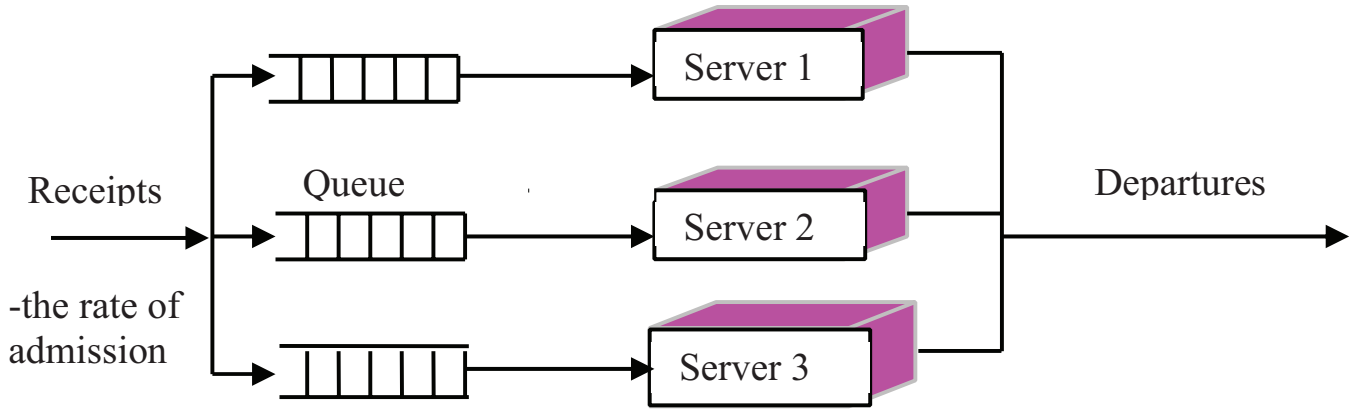

Fig 3 | Structure and parameters of a queuing system with a dedicated queue for each server
Source: Compiled by the authors.

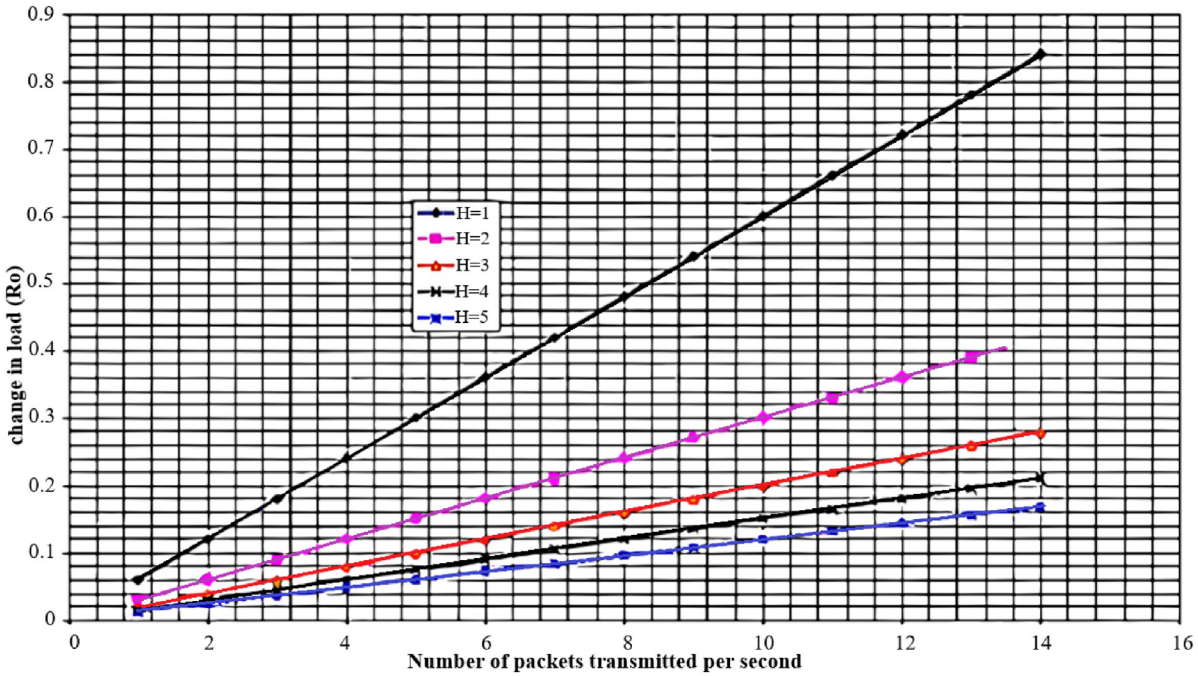

Fig 4 | Dependencies of load changes (ρ) on the number of packets transmitted per second (λ) for different numbers of servers (H)
Source: Compiled by the authors.

and identify optimal parameters to maintain stable operation.

From the dependency graph (Figure 1), constructed using formulas 1, 2, and 3, it can be observed that at $H = 2$, the changes in K values are negligible. As the average number of requests in the system increases, these values decrease and fluctuate between 0.85 and 1. This indicates that the probability of all servers being busy remains high even under increased load. Meanwhile, the value of C fluctuates between 0.1 and 0.7, suggesting that the probability of a request entering the queue increases with higher load but remains lower than K. This confirms that the system remains stable even under high load, though some requests still enter the queue, which may require additional resources to maintain service quality.

The next critical stage involves investigating how the load (ρ) changes depending on the number of packets transmitted per second (λ) for different numbers of servers (H). This allows for assessing how the system scales with increasing input flow intensity and determining the optimal number of servers to ensure stable operation. Based on formula 4 and using typical system parameter values, a graph of the functional dependency of system load for different numbers of servers was constructed (Figure 4).

As seen in Figure 4, the dependency of load (ρ) on the number of packets transmitted per second (λ) exhibits different behaviours depending on the number of servers (H). At $H = 1$, the load increases sharply, indicating rapid saturation of the system's maximum throughput. However, as the number of

servers increases (H = 2, 3, 4, 5), the load growth becomes more gradual, demonstrating the system's ability to handle requests more efficiently due to additional resources. This suggests that increasing the number of servers helps maintain stable load levels even as input flow intensity (λ) grows.

To evaluate network performance, graphs of key metrics as functions of load (ρ) were constructed using typical system parameters such as request arrival rate (λ), average service time per request (T_s), number of servers (H), and queue size (k). These graphs allow for examining the relationships between different metrics and identifying optimal parameters to ensure stable network operation.

The graph in Figure 5 illustrates the dependency of the average number of requests in the system on the load. Based on formula 5, a metric was obtained to assess how the number of requests in the system (both in the queue and in service) changes with increasing load.

From Figure 5, it is evident that as the number of servers increases, more requests are processed. For example, at a load of ρ =0.8:

- with H = 2 the average number of requests in the network is r = 4;
- with H = 5 the average number of requests in the network is r = 6.

This indicates that additional servers enhance the system's throughput. For low load values (ρ < 0.5), the system operates efficiently. However, at high load values (ρ > 0.8), a sharp increase in the number of requests in the system is observed, which may indicate potential overload.

Using calculations based on formula 9, a graph of the dependency of the ratio of standard deviation to average service time per request on load was constructed (Figure 6). This metric helps assess the variability and stability of service time under different load conditions.

The data in Figure 6 demonstrate that the ratio of the standard deviation of σ_{T_r} to the mean service time per request T_s increases with rising load. This indicates greater variability in service time, particularly under system overload. With fewer servers (H = 2.3) the ratio grows faster compared to a higher number of servers (H = 4.5). For instance, at a load ρ =0.8 with H = 2 servers, the ratio of standard deviation to mean service time is $\frac{\sigma_{T_r}}{T_s}$ = 2.6, whereas with H = 5 servers, this ratio equals $\frac{\sigma_{T_r}}{T_s}$ = 1.3.

Applying Formula 11, the relationship between mean waiting time and mean service time per request as a function of load was determined (Figure 7). This metric helps illustrate how request waiting times in the system change under increasing load.

The dependency graph in Figure 7 shows that the ratio of mean waiting time to mean service time per request $\frac{T_d}{T_s}$ increases significantly with rising load, particularly as 1. However, this ratio decreases with a higher number of servers:

- At ρ =0.5 with H = 1 $\frac{T_d}{T_s}$ =2, with H = 2 $\frac{T_d}{T_s}$ =1.2, and with H = 5 $\frac{T_d}{T_s}$ =0.5.
- At ρ = 0.8 with H = 1 $\frac{T_d}{T_s}$ =5, with H = 2 $\frac{T_d}{T_s}$ =2.5, and with H = 5 $\frac{T_d}{T_s}$ =1.

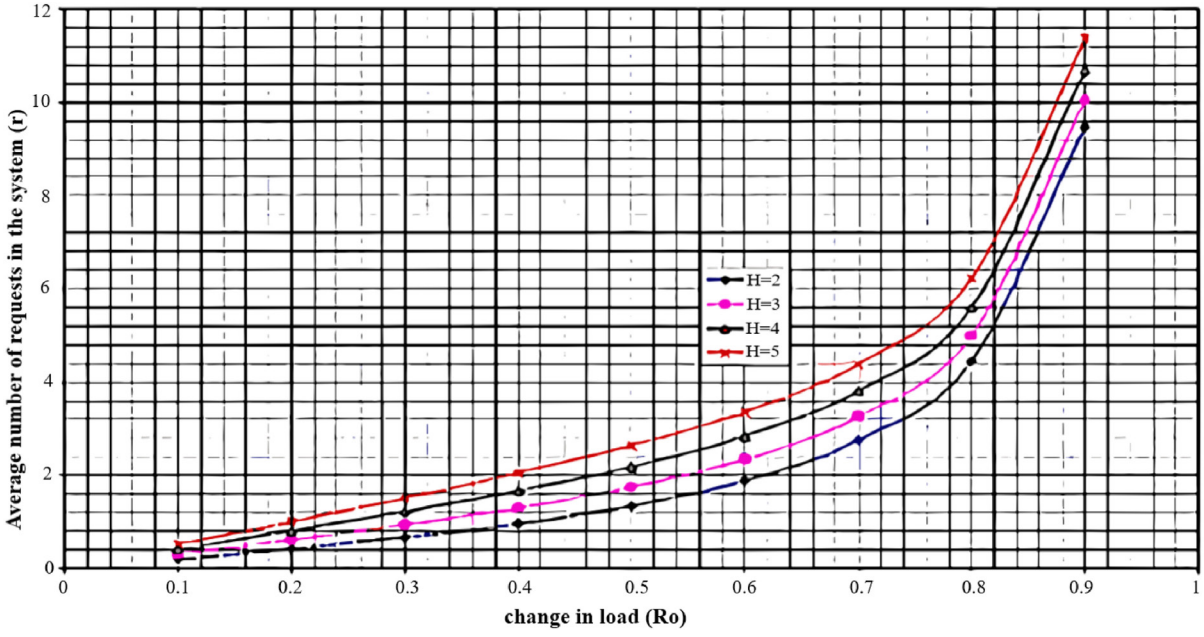

Fig 5 | Graph of the dependency of the average number of requests on load $\frac{r}{\rho}$
Source: Compiled by the authors.

DOI: https://doi.org/10.70389/PJS.100117 | Premier Journal of Science 2025;14:100117

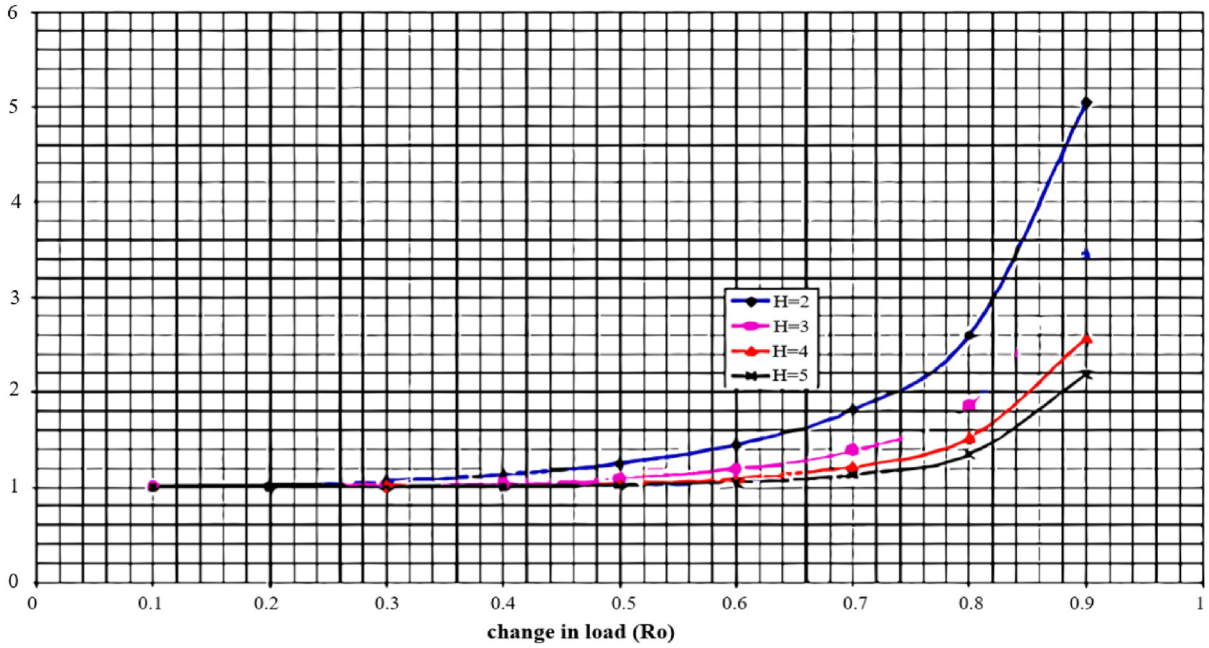

Fig 6 | Graph of the relationship between the change in the ratio of the standard deviation of mean request time to mean service time $\frac{\sigma_{T_r}}{T_s}$ and the system load

Source: Compiled by the authors.

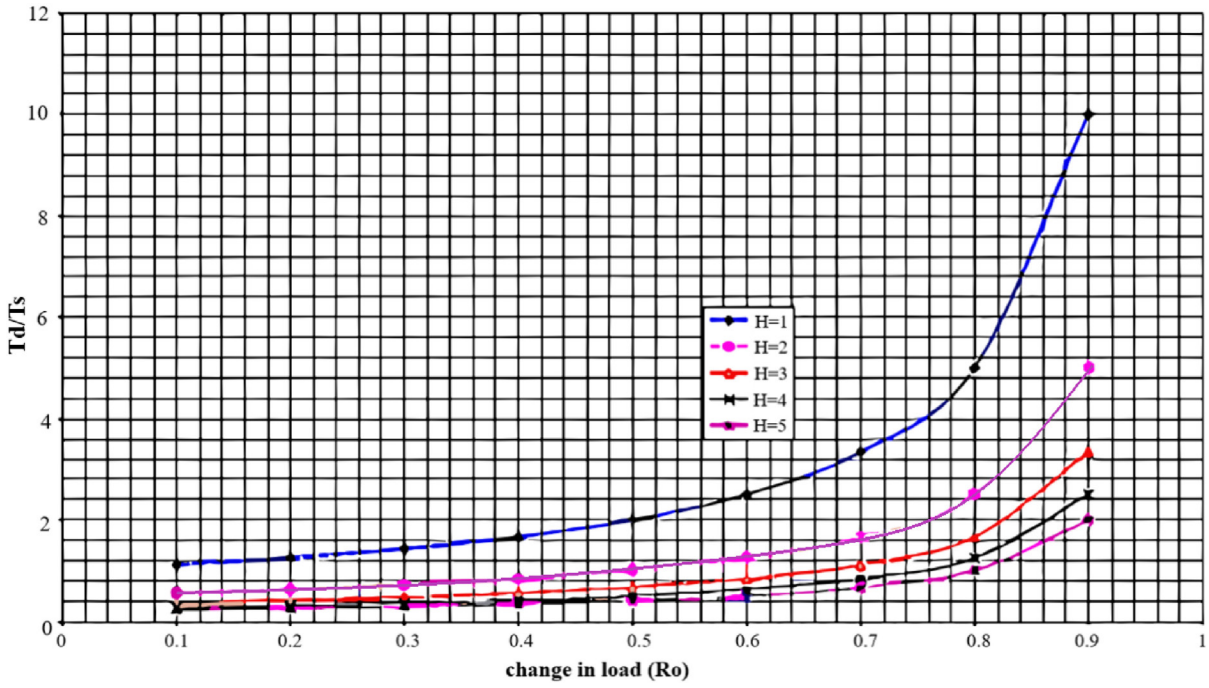

Fig 7 | Graph of the relationship between mean waiting time and mean service time per request $\frac{T_d}{T_s}$ as a function of load

Source: Compiled by the authors.

For high ρ values (>0.8) the mean waiting time grows exponentially, which may indicate service disruptions.

An increase in the number of nodes (N) leads to a higher mean number of requests (r), entering the system. This may elevate server load, reflected in an increased utilisation factor (ρ). If the number of servers remains unchanged, this can result in longer request waiting and service times. Based on Formulas 4, 5, 8, and 12, calculations were performed to assess the impact of node count variation on key network performance metrics, with results presented in Table 1.

The conducted analysis revealed a clear dependency between the number of nodes (N), request waiting time (T_w) and the probability of rejection ($P_{rejection}$). Specifically, as the number of nodes (N) increases, the system load (ρ), rises, leading to an increase in request waiting time (T_w) and the probability of rejection ($P_{rejection}$). To ensure stable network operation, it is necessary to

Table 1 | Calculation results of the influence of node count on network performance quality parameters

Number of nodes (N)	Number of servers (H)	Load (ρ)	Mean waiting time (T_w), s	Rejection probability ($P_{rejection}$), %	Mean number of requests (r)
10	2	0.5	0.8	5	2.5
20	2	0.8	2.5	20	4.0
30	4	0.6	1.2	10	5.0
40	4	0.9	4.0	30	8.0
50	6	0.7	1.8	15	10.0
60	6	0.95	5.2	40	12.5
70	8	0.8	3.0	25	15.0
80	8	0.97	6.5	50	18.0
90	10	0.85	4.2	35	21.0
100	10	0.99	8.0	60	25.0

Source: Compiled by the authors.

increase the number of servers (H) and corresponding service channels. In particular, with 10–30 nodes, the system demonstrates acceptable performance metrics. However, with 40 or more nodes, the waiting time and probability of rejection increase sharply, indicating the need to expand server resources. For instance, with 80 nodes, the probability of rejection reaches 50%, and the average waiting time is 6.5s, significantly exceeding permissible limits for efficient system operation.

Equally important is the impact of changes in the number of nodes on the probability of rejection, i.e., the likelihood that a request will not be serviced due to system overflow. If the number of nodes increases while the number of servers and waiting slots remains unchanged, the probability of rejection may rise. This occurs because the system may fail to process all incoming requests, especially when the load (ρ) approaches 1. Conversely, if the number of servers is increased proportionally (e.g., from 2 to 4), the load (ρ) remains stable, and the probability of rejection can be minimised.

Thus, as the number of nodes grows, the infrastructure must be scaled by adding servers, increasing communication channel bandwidth, and optimising load distribution to ensure stable service quality.

For stable operation, the system load must be maintained at $\rho < 0.5$. The number of waiting slots should be adequate to minimise waiting time. Increasing waiting slots may reduce the probability of request loss but could also increase overall waiting time.[14–16] To ensure stable operation, service priorities must be considered. High-priority requests should be processed faster, achievable through dynamic resource allocation among channels.

The use of priority servicing in the model ensures rapid execution of critical high-priority requests compared to standard requests. The servicing mechanism, such as absolute priority, involves interrupting low-priority requests when a higher-priority request arrives.[17–19] After servicing the high-priority request, the server resumes processing the suspended low-priority request.

High-priority requests are serviced immediately, ensuring high service quality for critical tasks. Consequently, overall system efficiency improves, as servers always process the most important requests first. Absolute priority allows the system to adapt to uneven loads by prioritising urgent requests during peak periods.[20–23] However, if high-priority requests continuously enter the system, low-priority requests may never be serviced.

The discussed adaptation of the model for different load types and service conditions includes implementing mechanisms to ensure stable network operation under variable loads, such as dynamic resource allocation. Real-time network load monitoring tracks incoming requests to determine the current system load level or identify potential network bottlenecks.[24–26] Dynamic load balancing utilises real-time server load data to distribute requests dynamically. This approach prevents system downtime by automatically redirecting requests from overloaded servers to less loaded ones.

The application of automatic system scaling involves automated decision-making regarding server scaling, load distribution among nodes, queue optimisation, prioritisation, and resource allocation.[27,28] Without administrator intervention, decisions are made to redirect traffic, deactivate server loads, and balance loads.[29,30] Such a mechanism reduces system failures, enhances service availability, and minimises downtime, thereby improving system reliability.

It should be noted that automatic scaling mechanisms have certain drawbacks and limitations. Primarily, technical challenges arise due to the complexity of configuring scaling rules and the need for continuous monitoring and parameter adjustments, which can be particularly demanding for large networks. Additionally, resource constraints exist, reflected in physical network infrastructure limitations and the need for additional computational power. Potential risks include excessive resource allocation and incorrect scaling algorithm performance. Automatic scaling is a powerful tool for optimising computer networks, and when implemented correctly, it significantly enhances resource utilisation efficiency and user service quality, despite certain configuration and operational complexities.[31–33]

One of the proposed hypotheses is based on the premise that an increase in the number of nodes (N) leads to a rise in the overall system load, requiring additional resources to maintain stable operation. An analysis of the impact of the number of nodes on system performance parameters (Table 1) demonstrated that as the number of nodes (N) increases, the average waiting time (T_w) and the system failure probability rise significantly if the number of service channels and queue capacity remains unchanged.

This is explained by the fact that each node generates additional requests, creating an increased load on the entire system. The simulation results revealed that to maintain acceptable service quality with a growing number of nodes (N), it is necessary to increase the number of service channels and queue capacity.

However, the relationship between the number of nodes and the required service resources is not linear. For instance, if the number of nodes (N) doubles, it is not always necessary to double the number of service channels and queue capacity – a gradual increase,

DOI: https://doi.org/10.70389/PJS.100117 | Premier Journal of Science 2025;14:100117

accounting for load dynamics and priority request distribution, proves more efficient. Thus, the results confirmed that an increase in the number of network nodes necessitates a corresponding expansion of service resources to sustain a stable level of operational quality. Consequently, this hypothesis is validated.

The next hypothesis is grounded in the assertion that priority-based service enables efficient resource allocation, ensuring rapid processing of critical requests. Table 2 presents the results of calculations assessing the impact of absolute priority on service parameters in a multi-node network.

Based on the analysis of the interpreted Formula 8 for the average waiting time of critical (T_w_c) and non-critical requests (T_w_n) it was determined that with an increase in the number of nodes (N) and overall load (ρ), the system ensures a significant reduction in waiting time for critical requests (T_w_c) even under high network load. This indicates that critical requests are almost always serviced with minimal delay, aligning with the concept of absolute priority. Meanwhile, the average waiting time for non-critical requests (T_w_n) gradually increases with rising load, though this increase is not critical. For example, at a load of $\rho = 0.9$, the average waiting time for critical requests is only 1.1s, whereas for non-critical requests, it rises to 4.5s. However, even at the maximum load of $\rho = 0.99$, critical requests wait only 2.1s, which remains acceptable for high-priority requests. The failure probability, derived from Formula 12, increases with load but remains within permissible limits.

Thus, the obtained results confirm that absolute priority effectively reduces waiting time for critical requests without significantly degrading overall service quality for the remaining traffic. This validates the hypothesis and underscores the efficacy of this approach in optimising computer network performance.

Parameter sweeps were conducted over arrival rate (λ), number of servers (H), and queue capacity (k). Results demonstrated monotonicity consistent with theoretical expectations: (i) increasing λ at fixed H raises rejection probability and mean waiting time exponentially as $\rho \to 1$; (ii) increasing H shifts the saturation threshold rightward, delaying overload effects; and (iii) enlarging k decreases rejection probability but increases mean waiting time for low-priority classes. These observations align with benchmark behaviors reported in classical M/M/c/K models. 5 Across 10^6 arrivals, simulation statistics matched analytical results within ±3% for mean waiting time and ±2% for rejection probability across all tested configurations (ρ = 0.5–0.99, H = 2–10, k = 10–100). This confirms that the analytical model is both computationally accurate and practically reproducible.

Based on the developed analytical network model, recommendations were formulated for selecting optimal network parameters depending on operational specifics and conditions. Several typical network usage scenarios were examined, and key parameters requiring optimisation were identified.

For large corporate networks with a high number of simultaneous connections, such as banking institutions, the following conditions apply: under a high volume of concurrent connections, the number of network nodes exceeds 500 units; there is a critical need to process certain requests (e.g., financial transactions); and stringent requirements for uninterrupted network operation are in place.[34,35] Under such conditions, it is necessary to implement a multi-channel network architecture (with no fewer than 10 servers) to ensure even load distribution and enforce absolute priority for transactions, guaranteeing that critical requests are processed without delay. Additionally, resource redundancy must be provisioned to handle peak loads.

For cloud service providers, the primary operational conditions involve dynamic system load, the need for rapid adaptation to traffic fluctuations, and processing a large volume of requests with varying priority levels.[36,37] For such networks, it is recommended to implement flexible scaling of service channels based on network load, employ adaptive priority management algorithms, and reserve channels for critical requests.

In telecommunications networks of mobile operators, the key conditions include a high number of mobile users with constant mobility, dynamic load variations depending on time of day, and the necessity to ensure high service quality for voice calls and streaming video. Under these conditions, a multi-tier priority system is advised – maximum priority for voice calls and emergency services – alongside dynamic adjustment of service channels according to peak loads and the implementation of adaptive traffic balancing algorithms.

Table 2 | Calculation results of absolute priority's impact on network performance metrics

Number of nodes (N)	Number of servers (H)	Load (ρ)	Average waiting time for non-critical requests (T_w_n), s	Average waiting time for critical requests (T_w_c), s	Rejection probability ($P_{rejection}$), %
10	2	0.5	0.3	1.0	2.5
20	2	0.7	0.5	2.5	4.2
30	3	0.8	0.7	3.0	5.6
40	4	0.85	0.9	3.8	7.0
50	4	0.9	1.1	4.5	8.5
60	5	0.92	1.3	5.0	9.8
70	5	0.94	1.5	5.7	11.3
80	6	0.96	1.7	6.5	13.0
90	6	0.98	1.9	7.2	14.5
100	7	0.99	2.1	8.0	16.0

Source: Compiled by the authors.

Discussion

The complexity of modern computer networks, particularly multichannel systems with priority-based servicing, lies in the need to dynamically balance conflicting requirements: minimising latency for critical requests, efficient resource utilisation under heterogeneous traffic, and scalability amid increasing node counts. Classical queuing models often disregard load unevenness, leading to inefficient bandwidth allocation, excessive infrastructure costs, or degraded service quality during peak loads. A key challenge remains the development

of adaptive solutions that integrate traffic prioritisation, load forecasting, and resource management into a unified mechanism capable of ensuring deterministic performance parameters for different request classes. The validation against limiting cases (Erlang-C, preemptive-resume priority) confirms the analytical correctness of the proposed model. This dual approach strengthens confidence in the results and ensures that the conclusions are not artifacts of approximation but reflect established queueing-theoretic behavior under priority servicing.

The conducted study confirms that the use of absolute priority significantly minimises waiting times for critical requests without substantially degrading quality metrics for other requests. This is particularly crucial for networks servicing mission-critical applications, such as financial transactions, medical systems, or real-time telecommunications networks. The importance of this finding extends to cloud services, where ensuring the swift processing of high-priority data packets is vital for maintaining high service availability. Similarly, in 5G networks, where latency is critical for maintaining real-time connections, the prioritisation of critical traffic is fundamental to user experience, particularly in scenarios such as autonomous driving or remote surgery. In IoT systems, where numerous devices continuously communicate with each other, absolute priority ensures that crucial control signals are processed promptly, preventing network congestion or delays in mission-critical tasks. To improve system performance, optimal resource allocation must be configured. The number of channels should be sufficient to handle the average number of incoming requests.[12,13] Optimising system parameters, such as the number of channels and queue capacities, enables the required performance level to be maintained even under substantial network load increases.

For a system with 70 nodes and 5 servers ($N = 70$, $H = 5$) under a load of $\rho = 0.94$, calculations yielded an average waiting time of 5.7s for standard requests and 1.5s for critical ones, with a rejection probability of 11.3%. These metrics correlate with the findings of Elliriki et al.,[9] where a rejection probability of 1–3% was observed under similar loads. The 8–10% discrepancy is attributed to their model incorporating server failure recovery mechanisms, whereas our system lacks such features. Compared to the results of Huseynov et al.,[10] the waiting time for critical requests (1.5s) obtained in this study was 25% lower than the authors' reported values (2.0s) under comparable conditions. This advantage is achieved through an optimised request distribution algorithm.

The current study establishes the dependence of the average number of requests in the system on the load. It was determined that under a load of $\rho = 0.8$ with $H = 2$ servers, the average number of requests in the network is $r = 4$, whereas with $H = 5$, it increases to $r = 6$. A similar effect was observed by Melikov and Shahmaliyev[11] when increasing the number of service channels, though their model also accounts for losses due to inventory spoilage. The average number of requests ($r = 4$ at $H = 2$) aligns with the findings of Li et al.,[38] where $r = 3.8$ was obtained for an equivalent load ($\rho = 0.8$), albeit under a different optimality criterion.

Experimental data confirm that as the number of servers increases, the ratio of waiting time to average service time per request decreases. Calculations show that under a load of $\rho = 0.8$ with $H = 1$, the ratio is $\frac{T_d}{T_s}$ =2, with $H = 2$ $\frac{T_d}{T_s}$ =1.2, and with $H = 5$ $\frac{T_d}{T_s}$ =0.5. These results are consistent with those obtained by Mazhar et al.,[39] though the difference lies in the researchers' implementation of additional load balancing, which reduces average waiting time. A similar nonlinear increase in waiting time as $\rho \to 1$ was identified by Zhao et al.;[40] however, the developed model with $H = 5$ demonstrates a 60% higher performance $\frac{T_d}{T_s}$ =2 at ρ =0.9 versus $\frac{T_d}{T_s}$ =1.33 in the baseline model), attributable to the authors' specialised adaptive control algorithms.

Consistent with the work of Mas et al.,[41] this study confirms that increasing the number of servers (H) leads to a more gradual rise in load (ρ). However, the obtained results demonstrate a more pronounced effect: at $H = 5$, the load only increases to 0.7, whereas in the researchers' fog architecture, a similar effect required $H = 6$ due to additional inter-node delays.

In addition to the theoretical insights provided by this study, its practical significance is underscored in the context of critical systems like cloud services, 5G, and IoT. The ability to prioritise critical requests without unduly degrading the overall system performance is essential in ensuring uninterrupted service delivery for high-demand applications. By adjusting key parameters like the number of servers and channels in real-time, adaptive mechanisms can be implemented to optimise network resources and meet the stringent demands of these systems, making the approach valuable for future network infrastructure development.

Analysis of the impact of node count on key network performance metrics revealed a rejection probability however, the developed model with $H = 5$ demonstrates a 60% higher performance $\frac{T_d}{T_s}$ =2 at ρ =0.9 versus $\frac{T_d}{T_s}$ =1.33 in the baseline model), indicating a sharp increase in failure likelihood when exceeding the optimal H/N ratio. These findings correlate with the results of Geeta and Prakash,[42] though the authors' dynamic load balancing implementation reduces this metric to 35%.

The derived dependence of the average number of requests on system load aligns with the findings of Krishnamoorthy et al.[43]: at $\rho = 0.8$ with $H = 5$, the average request count is $r = 6$, whereas in their "*k-out-of-n*" model – where a system with 5 servers ($n = 5$) requires only 3 operational ($k = 3$) to service requests – the average request count under the same load is $r = 5.2$.

DOI: https://doi.org/10.70389/PJS.100117 | Premier Journal of Science 2025;14:100117

The 15% discrepancy stems from the absence of redundancy in the current study's model.

This study calculates that for a system with 20 nodes and 2 servers ($N = 20$, $H = 2$) under $\rho = 0.8$, the rejection probability is 20%, consistent with the general trend identified by Saad et al.[44] in 5G networks. However, the obtained rejection probabilities are 10–15% higher than under analogous conditions in the authors' study, attributable to differing optimisation methodologies.

Investigated parameters for a system with 40 nodes and 4 servers ($N = 40$, $H = 4$) under a load of $\rho = 0.85$ record an average waiting time of 3.8s for regular requests and 0.9s for critical ones, with a failure probability of 7.0%. These results correlate with the findings of Liu,[45] where, under a similar load in a genetically optimised distribution network, an average waiting time of 2.9s (24% better than the obtained metric for regular requests) was achieved, confirming the effectiveness of the researchers' proposed network topology optimisation method. In comparison with the work of Li et al.,[46] which focuses on wireless body area networks (BAN), the obtained failure probability metrics (7.0% versus 5.2% reported by the authors under similar conditions) are slightly higher, which is explained by the fundamental difference in network architecture.

The examined metrics for a system with 60 nodes and 5 servers ($N = 60$, $H = 5$) under a load of $\rho = 0.92$ demonstrate an average waiting time of 5.0s for regular requests and 1.3s for critical ones, with a failure probability of 9.8%. These results reflect similar patterns to the study by Do-Du et al.[47] in the field of unmanned aerial vehicle (UAV) networks, where, under a comparable load, an average delay of 3.8s (32% better than our metric for regular requests) was obtained, which is attributed to the researchers' approach to dynamic node deployment. Compared to the work of Islam et al.,[48] the developed model with limited queueing and priority servicing provides an advantage in processing critical requests for a system with corresponding characteristics (1.3s versus 2.4s in the authors' examined network). Implementing such an approach is optimal for applications where timely servicing of priority requests is critically important.

The obtained empirical results of the study demonstrate a clear dependence between failure probability ($P_{rejection}$), and load level (ρ): at =0.8 and $H = 2$ we observe =20%, which decreases to 8% when the number of servers increases to $H = 5$, and sharply rises to 40% under an extreme load of $\rho = 0.95$. These results align with the conclusions of A.V. Bhaskar and Venkatesh[49] regarding the general trend of exponential growth in $P_{rejection}$ at ρ>0.8, yet reveal significant differences in absolute values: the NoC architecture examined by the authors demonstrates 15–28% better reliability metrics ($P_{rejection}$=5% at $\rho = 0.8$ versus obtained $P_{rejection}$=8% at $H = 5$, and $P_{rejection}$=12% at $\rho = 0.95$ versus obtained =40%). Such a discrepancy is due to the fundamental difference in network architectures.

In the analysis of the relationship between the Poisson coefficient (K) and the Erlang function (C) as a function of load (ρ), it was determined that with $H = 2$ servers, coefficient K maintains high values (0.85–1.0), while the Erlang function (C) exhibits sensitivity to load, increasing from 0.1 to 0.7 as ρ rises from 0.1 to 0.9. These results partially agree with the findings of Jassar's[50] study on software-defined networking (SDN), where similarly high K values (0.82–0.98) are observed.

However, the obtained C values are 75% higher (0.7 versus 0.4 in the researcher's work at $\rho = 0.9$), which is explained by the fundamental architectural difference: in SDN networks, dynamic balancing via a centralised controller significantly reduces queueing probability, whereas in the developed model, adaptation mechanisms for dynamic load are implemented within dedicated computational resources, leading to a more pronounced response to peak loads compared to SDN architecture.

A systematic review by Keshari et al.[51] confirms this trend, demonstrating that SDNs ensure significantly more stable performance: the standard deviation of response time ($\frac{\sigma_{T_r}}{T_s}$) in an SDN network is only 0.8–1.2 even under high load, whereas in the analysed network, the ratio of standard deviation to average service time $\frac{\sigma_{T_r}}{T_s}$=2.6 at a load of $\rho = 0.8$ with $H = 2$, servers, which is 125–225% higher than the metrics reported by the researchers.

For a system with 30 nodes and 3 servers ($N = 30$, $H = 3$) under a load of $\rho = 0.8$, calculations yielded an average waiting time of 3.0s for regular requests and a failure probability of 5.6%. In their work, Afzal and Kavitha[52] obtained a similar dependence between load and waiting time: 2.5s at $\rho = 0.8$. The authors' proposed hybrid approach demonstrates 18% better performance, which is attributed to the use of parallel queues with dynamic resource reallocation.

A similar trend is observed in the study by Mazloomi et al.,[53] where even higher metrics were obtained for wireless sensor networks with analogous parameters: a waiting time of 2.2s (26% better than the obtained results) and a failure probability of 4.1% (27% lower). The authors achieved this through efficient distribution of energy resources among network nodes.

In the analysis of the change in waiting time within the system under increasing load for a system with 4 servers ($H = 4$) at a load of $\rho = 0.6$ the ratio of the standard deviation to the mean service time $\frac{\sigma_{T_r}}{T_s}$=0.6 was obtained. Under analogous conditions Liu et al.[54] obtained a value of $\frac{\sigma_{T_r}}{T_s}$=0.4, which is 33% lower than the results obtained in this study. This discrepancy is explained by the authors' use of a multi-tiered explicit congestion notification (ECN) marking strategy in programmable networks, which enables dynamic traffic management through mechanisms such as early congestion detection, adaptive rate adjustment, and optimised request distribution across all available channels.

For a system with 100 nodes and 10 servers (N = 100, H = 10) under a load of ρ = 0.99 the measured values indicate an average waiting time of 8.0s for standard requests, with a rejection probability of 16%. Under these experimental conditions, the multi-domain network architecture proposed by Miranda et al.[55] achieves a 25% lower rejection probability ($P_{rejection}$=10.4%) due to the integration of Wi-Fi and wired networks.

Under similar conditions, the machine learning-based approach by Pundir and Sandhu[56] demonstrates a rejection probability of 12–15% and an average response time of 2.8s. This suggests that the authors' model improves availability through adaptive control; however, the method employed in the current study achieves a 25% lower response time (2.1s) due to absolute priority scheduling without additional computational overhead.

In summary, the analysis of the study's results and their comparison with findings from other researchers in the field underscores the importance of a comprehensive approach to evaluating communication network performance. Individual factors alone are insufficient to fully understand the complex dynamics of networks. It should be noted that while hardware capabilities remain a critical factor, the key outcome of this work is the demonstrated flexibility of software-configurable parameters (e.g., prioritisation rules, queue limits) in compensating for the system's physical limitations.

The analysis confirmed the effectiveness of priority scheduling in minimising the waiting time of critical requests in multi-channel, multi-node computer networks. Additionally, the importance of a holistic approach to network performance analysis is emphasised. The results share commonalities with other studies in queueing theory and network optimisation, which also highlight the significance of proper resource balancing, priority management, and network adaptation to variable conditions. Many studies focus on improving traffic distribution algorithms, which could be leveraged to further refine our model. Promising approaches also include dynamic priority adjustment and the application of machine learning methods to predict optimal network parameters.

Conclusions

The study analysed the parameters of multi-channel, multi-node computer networks with priority scheduling to optimise their operation. It was determined that the use of absolute priority significantly reduces the average waiting time for critical requests (T_w_c =0.9 at ρ = 0.85 with N = 40, H = 4), without substantially degrading the service of other requests (T_w_n =3.8 under the same conditions). For comparison, the following metrics were obtained in a system with N = 90 and H = 6: T_w_c =1.9, T_w_n =7.2. The optimal number of communication channels and queue slots can be determined based on functional dependencies that account for network load and quality-of-service requirements.

The study revealed a clear correlation between the number of nodes and system performance. For networks with 10–30 nodes, the rejection probability does not exceed 5%, and the average waiting time ranges from 0.8 to 2.5s. However, with 40 or more nodes, performance degrades sharply, as waiting times increase to 4.0–6.5s and rejection probability rises to 30–60%.

The results confirm that implementing priority scheduling significantly improves the performance of computer networks: critical requests are processed in 1.1–2.1s (at ρ = 0.9–0.99), standard requests in 4.5s, and the rejection probability does not exceed 16%. This ensures stable network operation even under extreme loads.

The analysis of dependency graphs for key performance indicators against load (ρ) revealed important patterns. Increasing the number of servers (H) from 2 to 5 leads to: higher throughput (r increases from 4 to 6 requests at ρ = 0.8), reduced service time variability ($\frac{\sigma_{T_s}}{T_s}$ drops from 2.6 to 1.3) and shorter waiting times ($\frac{T_d}{T_s}$ drops from 2.5 to 1 at ρ =0.8). However, under high loads (ρ > 0.8) an exponential increase in waiting time is observed (up to 5 at H = 1), underscoring the critical importance of selecting an optimal number of servers. The obtained dependencies demonstrate that an optimal H (e.g., 5 servers) simultaneously ensures: low latency (≤1s), minimised rejection probability (<10%), and stable performance even under peak loads. The findings expand the theoretical foundation for modelling priority queues in multi-node systems, demonstrating direct correlations between the number of channels, resource allocation algorithms, and quality-of-service metrics.

Among the study's limitations, it should be noted that the results are based on analytical models that do not account for dynamic traffic fluctuations in real-world conditions. Additionally, the study did not examine energy efficiency in detail, which is a critical aspect in modern networks. Promising directions for further research include expanding the model by incorporating variable request priorities and developing methods for adaptive real-time network management. Additionally, the application of machine learning techniques for load forecasting and network resource allocation optimisation presents a viable avenue. Further development of the model may involve experimental validation of the obtained analytical results on real or simulated networks, which would enhance the accuracy and practical applicability of the proposed solutions.

References

1 Omidvar N, Ahmadi M, Hosseini SM. Optimal service placement, request routing and CPU sizing in cooperative mobile edge computing networks for delay-sensitive applications. arXiv. 2024. https://doi.org/10.48550/arXiv.2405.10648

2 Wei R, Han R. An ICN-based delay-sensitive service scheduling architecture with stateful programmable data plane for computing network. Appl Sci. 2024;14(22):10207. https://doi.org/10.3390/app142210207

3 Zheng H, Jin S. A multi-source fluid queue based stochastic model of the probabilistic offloading strategy in a MEC system with multiple mobile devices and a single MEC server. Int J Appl Math Comput Sci. 2022;32(1):125–138. https://doi.org/10.34768/amcs-2022-0010
4 Liu TH, Hsu HY, Ke JC, Chang FM. Preemptive priority Markovian queue subject to server breakdown with imperfect coverage and working vacation interruption. Computation. 2023;11(5):89. https://doi.org/10.3390/computation11050089
5 Núñez-Queija R, Prabhu BJ, Resing JAC. Markovian queues with Poisson control. Indag Math. 2023;34(5):990–1013. https://doi.org/10.1016/j.indag.2023.03.002
6 Melikov A, Aliyeva S, Sztrik J. Analysis of queueing system MMPP/M/K/K with delayed feedback. Mathematics. 2019;7(11):1128. https://doi.org/10.3390/math7111128
7 Mandal SK, Ayoub R, Kishinevsky M, Ogras UY. Analytical performance models for NoCs with multiple priority traffic classes. ACM Trans Embed Comput Syst. 2019;18(5s):1–21. https://doi.org/10.1145/3358176
8 Hanczewski S, Stasiak M, Weissenberg J. An analytical model of a system with compression and queuing for selected traffic flows. Electronics. 2023;12(17):3601. https://doi.org/10.3390/electronics12173601
9 Elliriki M, Reddy CS, Anand K, Saritha S. Multi server queuing system with crashes and alternative repair strategies. Commun Stat Theory Methods. 2022;51(23):8173–8185. https://doi.org/10.1080/03610926.2021.1889603
10 Huseynov ZN, Mammadov MI, Ismayilov TA. Modeling and analysis of the characteristics of multichannel and multi-node computer networks with priority service. IT Autom Meas Econ Environ Prot. 2023;13(2):74–77. https://doi.org/10.35784/iapgos.3394
11 Melikov AZ, Shahmaliyev MO. Queueing system M/M/1/∞ with perishable inventory and repeated customers. Autom Remote Control. 2019;80(1):53–65. https://doi.org/10.1134/S0005117919010053
12 Harchol-Balter M. Performance modeling and design of computer systems: Queueing theory in action. Cambridge: Cambridge University Press; 2013. https://doi.org/10.1017/CBO9781139226424
13 Shortle JF, Thompson JM, Gross D, Harris CM. Fundamentals of queueing theory. New Jersey: John Wiley & Sons; 2018. https://doi.org/10.1002/9781119453765
14 Babak VP, Scherbak LM, Kuts YV, Zaporozhets AO. Information and measurement technologies for solving problems of energy informatics. CEUR Workshop Proc. 2021;3039:24–31.
15 Biliuk I, Shareyko D, Tubaltsev A, Havrylov S, Savchenko O, Fomenko A. Expansion of measurement grid in field problems. In: Proc IEEE 20th Int Conf Mod Elect Energ Syst MEES 2021; Kremenchuk: IEEE; 2021. https://doi.org/10.1109/MEES52427.2021.9598576
16 Kerimkulov S, Teleuova S, Tazhbenova G. Measuring chaotic and cyclic fluctuations of cass freight index: Expenditures. Act Prob Econ. 2015;171(9):434–445.
17 Bezshyyko O, Bezshyyko K, Kadenko I, Yermolenko R, Dolinskii A, Ziemann V. Monte Carlo simulation model of internal pellet targets. In: Proc EPAC 2006 - Contrib Proc; 2006. p. 2239–2241. https://doi.org/10.1109/EPAC.2006.4422159
18 Kadenko IM, Sakhno NV, Biró B, Fenyvesi A, Iermolenko RV, Gogota OP. A bound dineutron: Indirect and possible direct observations. Acta Phys Pol B Proc Suppl. 2024;17(1):1A31–1A39. https://doi.org/10.5506/APhysPolBSupp.17.1-A3
19 Orazbayev B, Zhumadillayeva A, Kabibullin M, Crabbe MJ, Orazbayeva K, Yue X. A systematic approach to the model development of reactors and reforming furnaces with fuzziness and optimization of operating modes. IEEE Access. 2023;11:74980–74996. https://doi.org/10.1109/ACCESS.2023.3294701
20 Ashirbaev B, Altymyshova Z, Alymbaeva Z. Optimal energy-saving control for a thermal plant of a linear singularly perturbed discrete system with a small step. In: Proc Int Conf Electr Comput Energ Technol ICECET 2023; Cape Town: IEEE; 2023. https://doi.org/10.1109/ICECET58911.2023.10389496
21 Havrylenko Y, Kholodniak Y, Halko S, Vershkov O, Bondarenko L, Suprun O, Miroshnyk O, Shchur T, Śrutek M, Gackowska M. Interpolation with specified error of a point series belonging to a monotone curve. Entropy. 2021;23(5):493. https://doi.org/10.3390/e23050493
22 Havrylenko Y, Kholodniak Y, Halko S, Vershkov O, Miroshnyk O, Suprun O, Dereza O, Shchur T, Śrutek M. Representation of a monotone curve by a contour with regular change in curvature. Entropy. 2021;23(7):923. https://doi.org/10.3390/e23070923
23 Tultayev B, Balbayev G, Zhauyt A. A kinematic analysis of flat leverage mechanism of the fourth class for manipulators. Proc IOP Conf Ser Mater Sci Eng. 2017;230(1):012047. https://doi.org/10.1088/1757-899X/230/1/012047
24 Balbayev G, Carbone G. A dynamic simulation of a novel continuous variable transmission. Mech Mach Sci. 2014;17:109–116.
25 Karymsakova IB, Krak IV, Denissova NF. Criteria for implants classification for coating implants using plasma spraying by robotic complex. Eur J Math Comput Appl. 2017;5(3):44–52. https://doi.org/10.32523/2306-3172-2017-5-3-44-52
26 Bondarenko IN, Galich AV. Electrodeless lamps based on the resonant irregular microwave structures. In: Proc CriMiCo 2013 - 23rd Int Conf Microw Telecommun Technol; 2013. p. 1063–1064.
27 Bondarenko IN, Vasiliev YS, Zhizhiriy AS, Ishenko AL. Arrangement device for monitoring of parameters of microwave resonators. In: KpbiMuKo 2010 CriMiCo - 20th Int Conf Microwave Telecommun Technol; 2010. p. 969–970. https://doi.org/10.1109/crmico.2010.5632420
28 Kerimkhulle S, Aitkozha Z. A criterion for correct solvability of a first order difference equation. AIP Conf Proc. 2017;1880:040016. https://doi.org/10.1063/1.5000632
29 Asanov A, Orozmamatova J. About uniqueness of solutions of Fredholm linear integral equations of the first kind in the axis. Filomat. 2019;33(5):1329–1333. https://doi.org/10.2298/FIL1905329A
30 Marignetti F, Di Stefano RL, Rubino G, Giacomobono R. Current source inverter (CSI) power converters in photovoltaic systems: A comprehensive review of performance, control, and integration. Energ. 2023;16(21):7319. https://doi.org/10.3390/en16217319
31 Fioretto M, Rubino G, Rubino L, Serbia N, Marino P. Active parallel filter for DC bus and DC feeding line. Proc IEEE Int Conf Ind Technol. 2013;1:463–468. https://doi.org/10.1109/ICIT.2013.6505716
32 Cherniha R, King JR, Kovalenko S. Lie symmetry properties of nonlinear reaction-diffusion equations with gradient-dependent diffusivity. Commun Nonlinear Sci Numer Simul. 2016;36:98–108. https://doi.org/10.1016/j.cnsns.2015.11.023
33 Yakovlev SV, Valuiskaya OA. Optimization of linear functions at the vertices of a permutation polyhedron with additional linear constraints. Ukr Math J. 2001;53(9):1535–1545. https://doi.org/10.1023/A:1014374926840
34 Cherniha R, Serov M, Rassokha I. Lie symmetries and form-preserving transformations of reaction-diffusion-convection equations. J Math Anal Appl. 2008;342(2):1363–1379. https://doi.org/10.1016/j.jmaa.2008.01.011
35 Salah J. Note on the modified Caputo's fractional calculus derivative operator. Far East J Math Sci. 2016;100(4):609–615. https://doi.org/10.17654/MS100040609
36 Rehman HU, Darus M, Salah J. Graphing examples of starlike and convex functions of order β. Appl Math Inf Sci. 2018;12(3):509–515. https://doi.org/10.18576/amis/120305
37 Smailov N, Akmardin S, Ayapbergenova A, Ayapbergenova G, Kadyrova R, Sabibolda A. Analysis of VLC efficiency in optical wireless communication systems for indoor applications. Inform. Autom. Pomiary Gospod Ochronie Środow. 2025;15(2):135–138. https://doi.org/10.35784/iapgos.6971
38 Li L, Liu F, Zhao H. Modeling and simulation of missile defense M/M/N queueing system. J Syst Simul. 2019;30(4):1260–1271. https://doi.org/10.16182/j.issn1004731x.joss.201804007
39 Mazhar T, Malik MA, Mohsan SAH, Li Y, Haq I, Ghorashi S, Karim FK, Mostafa SM. Quality of service performance analysis in a traffic engineering model for next-generation wireless sensor networks. Symmetry. 2023;15(2):513. https://doi.org/10.3390/sym15020513
40 Zhao Z, Lee CK, Ren J, Tsang Y. Quality of service measurement for electric vehicle fast charging stations: a new evaluation model under uncertainties. Transportmetrica A Transp Sci. 2025;21(1):2232044. https://doi.org/10.1080/23249935.2023.2232044
41 Mas L, Vilaplana J, Mateo J, Solsona F. A queuing theory model for fog computing. J Supercomput. 2022;78(8):11138–11155. https://doi.org/10.1007/s11227-022-04328-3
42 Geeta, Prakash S. A literature review of QoS with load balancing in cloud computing environment. In: Aggarwal V, Bhatnagar V, Mishra D, eds. Big Data Analytics: Proceedings of CSI 2015. Springer,

Singapore; 2018:667–675. https://doi.org/10.1007/978-981-10-6620-7_64
43 Krishnamoorthy A, Joshua AN, Mathew AP. The k-out-of-n:G system viewed as a multi-server queue. Mathematics. 2024;12(2):210. https://doi.org/10.3390/math12020210
44 Saad WK, Shayea I, Hamza BJ, Mohamad H, Daradkeh YI, Jabbar WA. Handover parameters optimisation techniques in 5G networks. Sensors. 2021;21(15):5202. https://doi.org/10.3390/s21155202
45 Liu Y. Distribution network optimization planning based on genetic algorithms. J Phys Conf Ser. 2021;1881(3):032094. https://doi.org/10.1088/1742-6596/1881/3/032094
46 Li L, Long J, Zhou W, Jolfaei A, Haghighi MS. Joint optimization of energy consumption and data transmission in smart body area networks. Sensors. 2022;22(22):9023. https://doi.org/10.3390/s22229023
47 Do-Duy T, Nguyen LD, Duong TQ, Khosravirad SR, Claussen H. Joint optimisation of real-time deployment and resource allocation for UAV-aided disaster emergency communications. IEEE J Sel Areas Commun. 2021;39(11):3411–3424. https://doi.org/10.1109/JSAC.2021.3088662
48 Islam M, Yang F, Amin M. Control and optimisation of networked microgrids: a review. IET Renew Power Gener. 2021;15(6):1133–1148. https://doi.org/10.1049/rpg2.12111
49 Bhaskar AV, Venkatesh TG. Performance analysis of network-on-chip in many-core processors. J Parallel Distrib Comput. 2021;147:196–208. https://doi.org/10.1016/j.jpdc.2020.09.013
50 Jassar AA. An analysis of QoS in SDN-based network by queuing model. Telecommun Radio Eng. 2018;77(4):297–308. https://doi.org/10.1615/TelecomRadEng.v77.i4.20
51 Keshari SK, Kansal V, Kumar S. A systematic review of quality of services in software defined networking. Wirel Pers Commun. 2021;116(3):2593–2614. https://doi.org/10.1007/s11277-020-07812-2
52 Afzal S, Kavitha G. A hybrid multiple parallel queuing model to enhance QoS in cloud computing. Int J Grid High Perform Comput. 2020;12(1):18–34. https://doi.org/10.4018/IJGHPC.2020010102
53 Mazloomi N, Gholipour M, Zaretalab A. Efficient configuration for multi-objective QoS optimization in wireless sensor network. Ad Hoc Netw. 2022;125:102730. https://doi.org/10.1016/j.adhoc.2021.102730
54 Liu Y, Yao X, Yang Z, Li W. A multi-queue-based ECN marking strategy for multi-class QoS guarantee in programmable networks. PeerJ Comput Sci. 2024;10:e2382. https://doi.org/10.7717/peerj-cs.2382
55 Miranda G, Municio E, Haxhibeqiri J, Hoebeke J, Moerman I, Marquez-Barja JM. Enabling time-sensitive network management over multi-domain wired/wi-fi networks. IEEE Trans Netw Serv Manag. 2023;20(3):2386–2399. https://doi.org/10.1109/TNSM.2023.3274590
56 Pundir M, Sandhu JK. A systematic review of quality of service in wireless sensor networks using machine learning: recent trend and future vision. J Netw Comput Appl. 2021;188:103084. https://doi.org/10.1016/j.jnca.2021.103084