Minimax Theory and its Applications
Volume 11 (2026), No. 3s, 969 to 979

# Intelligent Task Offloading for Latency-Sensitive Fog-IoT Architectures

**Pradeep M[1], Dr. M. Rajeshwari[2], Dr. Makineedi Raja Babu[3], Arivukkodi R[4], Dr. R. Balamurugan[5], Madhukar E[6]**

[1]Associate Professor, Department of Electronics and Communication Engineering, Shri Vishnu Engineering College for Women, Email: pradeepm999@gmail.com, ORCID ID: 0000-0002-1605-5589
[2]Assistant Professor, Department of Computer Applications, Sri Ramakrishna College of Arts & Science (Autonomous), Coimbatore – 6, Email: rajeshwari@srcas.ac.in, ORCID ID: 0009-0003-8448-2085
[3]Department of Information Technology, Aditya University, Surampalem, Andhra Pradesh – 533437 Email: rajababu.makineedi@adityauniversity.in, ORCID ID: 0009-0009-7364-8805
[4]Assistant Professor, Department of Computer Science, Meenakshi College of Arts and Science, Meenakshi Academy of Higher Education and Research, Email: arivukodir@maher.ac.in, ORCID ID: 0009-0006-0782-4987
[5]Associate Professor, Department of Computer Science and Engineering (Cyber Security), New Prince Shri Bhavani College of Engineering and Technology, Email: drrbalamurugan.cse@gmail.com, ORCID ID: 0000-0002-2424-3109
[6]Professor, Department of Computer Science and Engineering, Sreenidhi Institute of Science and Technology Email: emadhukar@gmail.com, ORCID ID: 0000-0002-2506-3759

**Abstract**
Latency-sensitive IoT applications need task execution in the proximity of data sources under resource constraints at devices, varying bandwidth in communication, and dynamic congestion at fog nodes. Typical offloading techniques usually analyze either the transmission, computation, queuing, or deadline individually, leading to the choice of a fog node that may look attractive in terms of computing power but turns out to be inappropriate as the queue at the fog node approaches saturation. In this paper, we propose an Intelligent Queue-aware Task Offloading (LRI-IQATO) approach to Latency-Sensitive Fog-IoT systems. The proposed approach includes the construction of a nonlinear function called Queue-Aware Latency Risk Index (LRI), where the risk of transmission, propagation, processing, and congestion is incorporated into the latency, which is then normalized by the deadline of the task. The degree of the risk penalty as a function of congestion sensitivity is included in the LRI formulation. From the LRI constraint, a closed-form task-specific maximum fog-utilization threshold is derived to rule out infeasible fog nodes before offloading the task.Three operating regime of three regimes are used to classify execution states as being low risk, transition risk, and high risk. Numerical analysis of the approach has been done to showcase the behaviour of the proposed model. The approach is considered to be a framework for decision-making that will support the intelligent offloading policy and not a new queuing model or machine learning technique.

*Keywords:* Fog computing; Internet of Things; task offloading; latency sensitivity; queue-aware optimization; deadline feasibility; congestion risk; resource allocation.

## 1. Introduction

### 1.1 Motivation and Research Problem

The emergence of IoT application systems has increased the need for computations that are not only nearby but also responsive to tight deadlines. The execution of computations on the cloud alone could lead to some delays on account of propagation and communication, while their execution locally could face limitations due to the computational power of the device and its available energy. This gap is filled by the advent of fog computing, where the computation resources lie between the edge devices and the distant cloud computing environment. However, the offloading of any computation from an edge device is not influenced by the CPU speed and proximity of fog nodes alone. If the processing capabilities of a fog node are high, then it is likely that the task queue of such a fog node will also be highly loaded.

It has also been shown in current fog-IoT literature that both time for task completion, energy usage, quality of service needs, and deadline limitations can be accounted for together when offloading (Shahryari et al., 2021). Recent intelligent methods for offloading have also employed reinforcement learning or deep reinforcement learning as a means to adapt their decision-making process to workload variations and heterogeneous fog nodes

(Alam et al., 2019; Tomar et al., 2025). This demonstrates the necessity of an adaptive offloading mechanism, but also points out a problem in modelling since the learning algorithm could recognize queue load or latency as state variables without any measure of deadline feasibility.

This paper solves this problem using a novel approach by introducing a Queue-Aware Latency Risk Index (LRI). This index is not intended to be an alternative to existing M/M/1 queueing theory. Rather, the usage of a queue is taken as the congestion state, whereas the nonlinear transformation offers a parameterized risk penalty as the utilization goes closer to saturation. This is a significant difference from other methods since queueing-based response time models and deadline-aware scheduling have already been developed. The novelty here is in the combination of these concepts mathematically.

### 1.2 Research Contributions

The major achievements consist of four points. Firstly, the nonlinear LRI is defined by integrating the latencies associated with transmission, propagation, processing, and queueing into a nondimensional form of a deadline-based criterion. Secondly, the deadline feasibility theorem is obtained, from which the threshold maximum value of the utilization for each task-fog node pair can be found explicitly. Thirdly, the sensitivity and convexity analysis is conducted in order to understand how the arrival rate, service rate, bandwidth, CPU capability, and the congestion factor influence the probability of risk. Finally, the theoretical findings are incorporated into the intelligent offloading scheme.

## 2. Related Work

### 2.1 Fog-IoT Task Offloading

Task offloading is an important means of enhancing the computational capacity of resource-limited IoT devices. Chang et al. (2021) developed dynamic resource allocation and computation offloading strategies for the IoT fog system, taking into account offloading, radio, and computation resources, latency, and energy. Shahryari et al. (2021) presented the selection of fog nodes and computation allocation as a mixed-integer nonlinear optimization problem where the goal is to minimize task completion time and energy. From these works, we can infer that fog offloading is a resource management problem that cannot be classified simply as one between local and cloud.

Further development of this aspect is seen in adaptive and predictive algorithms. Gao et al. (2020) presented a predictive offloading and resource allocation algorithm for dynamic fog computing. On the other hand, Tong et al. (2020) presented an adaptive computation offloading and resource allocation approach for mobile edge computing. Cooperative resource allocation is another area where research has been done to reduce latency in hierarchical and cooperative edge computing resources (Kuang et al., 2021). Recent Pareto optimization takes makespan and cost into account in Fog-Cloud environments (Bernard et al., 2024).

### 2.2 Latency- and Queue-Aware Offloading

Latency is especially relevant for IoT applications that are real-time, since it is possible that the process of offloading may become counterproductive if communication and waiting times outweigh the benefits of computation in the fog. Shahryari et al. (2021) specifically take task deadlines and QoS into account, while Panda et al. (2025) factor in energy, delay, priority, and load balancing in fog task offloading. Deng et al. (2024) moreover prove that uncertainty in processing cycles and communication makes the task offloading problem different. The abovementioned studies emphasize the importance of the consideration of the dynamic system state, but they do not provide the risk function presented herein.The current model includes the following components in fog latency: transmission time, processing time, propagation time, and queue time. The state of the queue is modelled using its utilization rho=lambda/mu. In order to differentiate a moderate queue from near saturation, the risk function increases non-linearly as rho increases to one.

### 2.3 Intelligent Offloading and Research Gap

Learning has become more prevalent in adapting offloading decision-making. Alam et al. (2019) employed Deep Reinforcement Q-learning for autonomic computation offloading. Tomar et al. (2025) defined fog-IoT task offloading as a multi-objective decision-making problem that includes priorities, energy, latency, and deadline and then analyzed DQN, DDPG, and SAC. Liakath et al. (2025) examined deadline-aware fog-cloud scheduling through multi-agent reinforcement learning, whereas recent studies have applied federated deep reinforcement learning in the Edge-Fog-Cloud hierarchy (Ambika et al., 2026).

Although all these approaches highlight the benefits of adaptive intelligence, their decision state interpretation may not be analytically interpretable. Hence, the research gap covered in this paper is not on the application of fog computing, queues, deadlines, or learning alone but on the lack of a concise nonlinear risk formulation that translates existing congestion, communication load, processing need, and deadline into a feasibility boundary. The LRI-IQATO covers this gap with analytical risk being an eligibility layer before intelligent selection.

## 3. System Model and Problem Formulation

### 3.1 Fog-IoT System Model

Let us consider the Fog-IoT architecture comprising a set of IoT devices, wireless network communication, K heterogeneous fog nodes, and a cloud network. The IoT device computes a task.$i, j$ whose input size is $D_{ij}$,CPU cycle needed $C_{ij}$and the maximum acceptable task completion time is $D_{ij}^{\max}$.The task may be computed locally, offloaded to an eligible fog node k, or sent to the cloud when there is no fog node that meets the demands of the task.

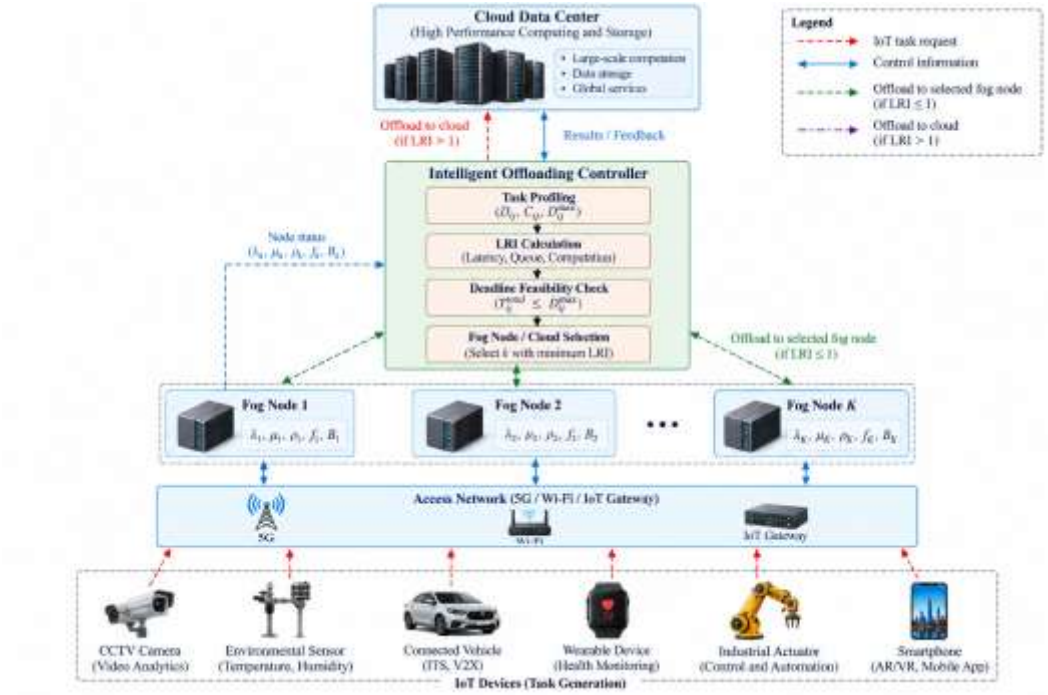



**Figure 1.** Intelligent Queue-Aware Task Offloading Architecture for Fog-IoT.

The available effective bandwidth for task $i,j$ is denoted by $B_{ik}$, processing capability $f_k$, task arrival rate $\lambda_k$, and service rate $\mu_k$. Utilization is defined as $\rho_k = \frac{\lambda_k}{\mu_k}$,and stability necessitates $0 \leq \rho_k < 1$. The cloud is considered an alternative source compared to the optimization of the cloud since the algorithm seeks to schedule feasible fog task execution.

### 3.2 Task and Latency Model

The transmission latency is represented by:

$$T_{\text{tx},ij}^{k} = \frac{D_{ij}}{B_{ik}}$$

while the processing latency is:

$$T_{\text{proc},ij}^{k} = \frac{C_{ij}}{f_k}$$

A propagation term $T_{\text{prop},ij}^{k}$ accounts for path-dependent propagation delay. The queue component is represented by:

$$Q_k = \frac{1}{\mu_k(1-\rho_k)^{\eta}}$$

where $\eta > 0$ is the congestion sensitivity parameter. For $\eta = 1$, it has a functional form similar to the M/M/1 response time relationship; for a generic value of $\eta$, the equation is deliberately employed as a nonlinear risk transformation of utilization and not as a queueing identity.

Accordingly, the proposed expected fog latency is:

$$L_{ij}^{k} = \frac{D_{ij}}{B_{ik}} + \frac{C_{ij}}{f_k} + T_{\text{prop},ij}^{k} + \frac{1}{\mu_k(1-\rho_k)^{\eta}}$$

The normalized Queue-Aware Latency Risk Index is:

$$\text{LRI}_{ij}^{k} = \frac{L_{ij}^{k}}{D_{ij}^{\max}}$$

A candidate fog node is considered latency-feasible when $\text{LRI}_{ij}^{k} \leq 1$.

### 3.3 Queue Model and Optimization Problem

The offloading decision variable $x_{ij}^{k}$ is binary and has a value of one if task $i,j$ is allocated to a fog node $k$. The objective function for selection takes into account risk factors related to latency as well as, optionally, the energy and computation cost components. The normalized formulation ensures that heterogeneous factors can be summed without having raw latency units automatically dominate the summation process. The mathematical optimization model is formulated as:

$$\min \quad J = \sum_{i}\sum_{k} x_{ij}^{k}\left[\alpha \text{LRI}_{ij}^{k} + \beta E_{ij}^{k} + \gamma C_{ij}^{k}\right]$$

$$\text{subject to} \quad \sum_{k} x_{ij}^{k} \leq 1,$$
$$\text{LRI}_{ij}^{k} \leq 1,$$
$$\rho_k \leq \rho_{k,ij}^{\max},$$
$$\rho_k < 1,$$
$$x_{ij}^{k} \in \{0,1\}$$

$\alpha, \beta$, and$\gamma$ are here non-negative weights on the policy. The LRI layer could thus be used in conjunction with an analytical selector or as a state feature with a learning controller.

## 4. Proposed Intelligent Task Offloading Framework

### 4.1 Queue-Aware Latency Risk Index

The main component of LRI-IQATO is the deadline normalized risk index. The first three terms represent the cost of deterministic communication and computations, while the last term represents the cost of congestion in the queue. By normalizing the risk index by the task deadline, LRI directly reflects whether the estimated fog latency exceeds the deadline budget; LRI < 1 means that the estimated fog latency is inside the deadline budget,

while LRI > 1 indicates deadline infeasibility under the given state. η parameter specifies the sensitivity of the risk function to utilization close to saturation. So, it can be treated as a policy parameter rather than a physical queue parameter. A conservative controller will set η higher, while a non-conservative controller will set it lower.

### 4.2 Deadline-Based Fog Eligibility

Let $\Delta_{ij}^{k} = D_{ij}^{\max} - \frac{D_{ij}}{B_{ik}} - \frac{C_{ij}}{f_k} - T_{\text{prop},ij}^{k}$ denote the residual deadline budget for queueing effects. Feasibility of deadlines needs $\Delta_{ij}^{k} > 0$. Substituting the proposed queue risk into $LRI \leq 1$ yields:

$$\frac{1}{\mu_k(1-\rho_k)^{\eta}} \leq \Delta_{ij}^{k}$$

Rearranging produces the task-specific threshold:

$$\rho_k^{\max} = 1 - \left(\frac{1}{\mu_k \Delta_{ij}^{k}}\right)^{\frac{1}{\eta}}$$

provided $\mu_k \Delta_{ij}^{k}$ is sufficiently large for a feasible operating region.

It is significant since the deadline condition gets converted into a clear congestion bound. This means that in case the value of $\rho_k$ is greater than $\rho_k^{\max}$, the fog node is removed even prior to the last selection phase. The threshold value is contingent upon all seven factors at the same time, including the task size, CPU requirement, communication bandwidth, propagation latency, serving capacity, deadline, and η. It is the main distinguishing feature of the suggested solution.

### 4.3 Intelligent Offloading Strategy

The decision procedure contains five stages. First, the device describes the task using data size, CPU demand, and deadline. Second, the controller estimates bandwidth, propagation delay, fog CPU capacity, arrival rate, and service rate. Third, LRI and the utilization threshold are calculated for each candidate fog node. Fourth, nodes violating either the threshold or the deadline are removed from the eligible set. Fifth, the remaining nodes are ranked using LRI and optional energy/resource costs, and the selected task is offloaded. After execution, observed queue state and deadline status are returned as feedback.

This design makes the framework compatible with intelligent control without requiring a black-box predictor for the basic feasibility decision. A reinforcement-learning agent, for example, can use LRI, residual deadline budget, and normalized utilization as state features. The analytical threshold acts as a safety filter that prevents the policy from selecting a node that is mathematically infeasible under the current model. This separation between feasibility and preference improves interpretability and provides a clear mechanism for incorporating learning later.

The complete decision process is therefore:

task generation → state estimation → LRI calculation → threshold filtering → candidate ranking → offloading → execution → feedback

The framework is deliberately designed so that the analytical layer remains understandable even when an intelligent selector is added.

## 5. Analytical Analysis

### 5.1 Deadline-Feasibility Analysis

**Theorem 1:** A task can satisfy its deadline at a fog node $k$ under the proposed model if the non-queue latency budget $\Delta_{ij}^{k}$ is positive and $\rho_k \leq \rho_k^{\max}$.

**Proof:**LRI ≤ 1 requires the queue term to be no greater than $\Delta_{ij}^{k}$. Hence:

$$(1-\rho_k)^{\eta} \geq \frac{1}{\mu_k \Delta_{ij}^{k}}$$

Since $\eta > 0$, raising both sides to the power $\frac{1}{\eta}$ and rearranging yields:

$$\rho_k \leq 1 - \left(\frac{1}{\mu_k \Delta_{ij}^{k}}\right)^{\frac{1}{\eta}}$$

It is necessary for $\Delta$ to be positive in order that the tasks of communication, computation and propagation should not take up the whole deadline period.

The theorem provides an operational interpretation directly. A node with high CPU capability may get rejected on account of being too highly utilized at present. However, a relatively provisioned node will continue to be qualified if its queue is small and the path of communication gives adequate residual deadline time.

**5.2 Parameter Sensitivity and Congestion Analysis**

For fixed service capacity, increasing arrival rate increases risk. With $\rho = \frac{\lambda}{\mu}$, the derivative of the queue component with respect to $\lambda$ is:

$$\frac{\partial Q_k}{\partial \lambda} = \frac{\eta}{\mu^2 \left(1 - \frac{\lambda}{\mu}\right)^{\eta+1}}$$

which is positive in the stable region. Higher service rates reduce queue risk. Similarly, the derivative of the transmission term with respect to bandwidth is:

$$\frac{\partial \mathrm{LRI}_{\mathrm{tx}}}{\partial B_{ik}} = -\frac{D_{ij}}{D_{ij}^{\max} B_{ik}^2}$$

showing that additional bandwidth monotonically reduces LRI. The derivative with respect to CPU capacity is:

$$\frac{\partial \mathrm{LRI}_{\mathrm{proc}}}{\partial f_k} = -\frac{C_{ij}}{D_{ij}^{\max} f_k^2}$$

showing that stronger fog processors reduce processing-related risk.

The congestion-sensitivity derivative is also positive:

$$\frac{\partial Q}{\partial \eta} = -\frac{\ln(1-\rho)}{\mu(1-\rho)^{\eta}} > 0 \quad \text{for } 0 < \rho < 1$$

Therefore, greater $\eta$ means a bigger risk penalty for the same utilization. The derivatives show the interpretable link between the controllable or observable variables of the system and risk.

**5.3 Convexity and Three-Regime Behaviour**

Let $Q(\lambda) = \frac{1}{\mu\left(1-\frac{\lambda}{\mu}\right)^{\eta}}$. Its second derivative is:

$$\frac{\partial^2 Q}{\partial \lambda^2} = \frac{\eta(\eta+1)}{\mu^3 \left(1 - \frac{\lambda}{\mu}\right)^{\eta+2}}$$

which is positive for the region in question. Thus, the function is strictly convex with respect to arrival rate. Moreover, Q tends to infinity when ρ tends to one from the left.($\rho \to 1^-$).This means that the impact of additional arrivals is increasing as a fog node approaches saturation.

The behaviour gives rise to three operating modes. Low-risk mode means that the utilization is sufficiently lower than the task-specific threshold and fog execution is comfortable. Transition-risk mode implies that utilization is approaching the threshold, and comparison of residual deadline budget and risk is required. Finally, critical-risk mode indicates that utilization is equal to or greater than the threshold and the node should be excluded from further consideration.

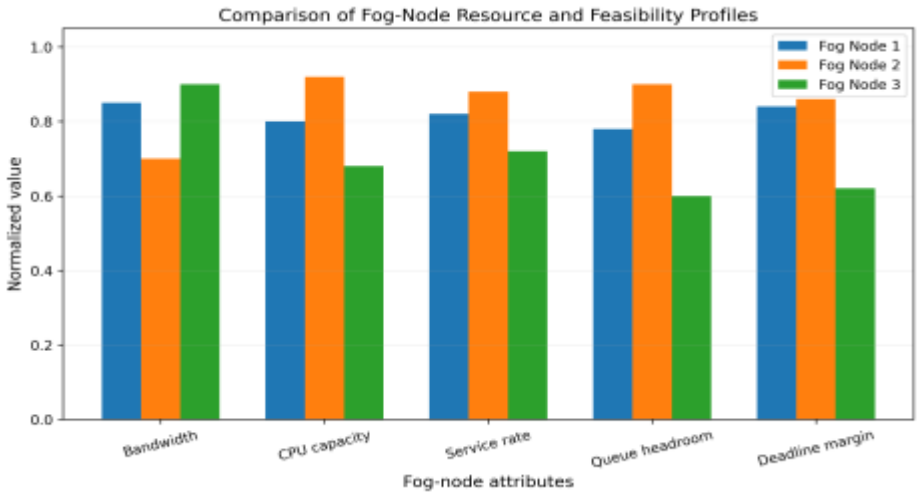



**Figure 2.** Nonlinear Queue-Aware Latency Risk Index versus fog utilization for representative congestion-sensitivity values.

Figure 2 shows the convex rise in LRI when utilization is close to being saturated. The plotted curves show representative values used only to illustrate the math concept and do not represent actual readings from any practical system.

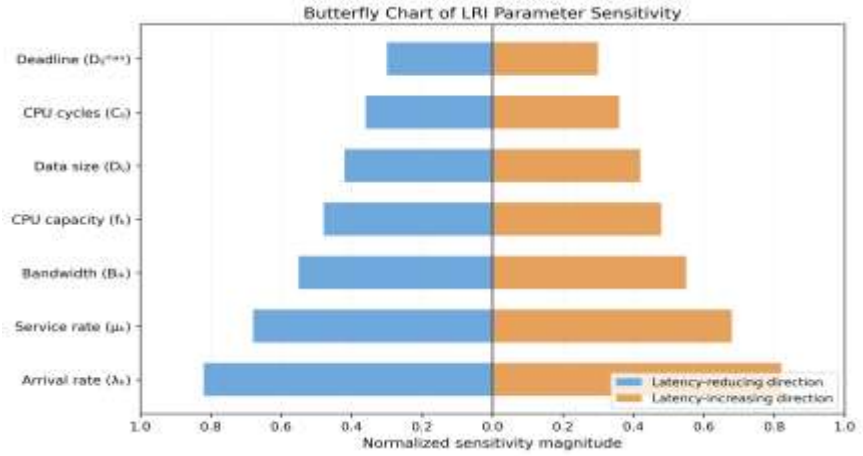



**Figure 3.** Maximum allowable fog utilization versus residual deadline budget.

From Figure 3, it is evident that the increase in residual deadline budget leads to an increase in the region of feasible utilization. How fast this region grows depends on the service rate and congestion sensitivity.

## 6. Performance Evaluation and Results

### 6.1 Evaluation Configuration

Evaluation of the equations is performed in an analytical and computational manner and not as a measurement exercise. Parameter values representative of typical ranges for the demonstration of the characteristics of the proposed equations are used. This is in keeping with the analytical nature of the research draft since numerical graphs are meant to show mathematical characteristics and should not be viewed as real-life deployment measurements. The key evaluation parameters are fog use, service rate, deadline budget remaining, congestion sensitivity, and normalized task latency, a summarizes the mathematical parameters Table 1

**Table 1.** summarizes the mathematical parameters used throughout the framework.

| **Symbol** | **Meaning** | **Unit / Domain** |
|---|---|---|
| D_ij | Task input data size | MB |
| C_ij | Required CPU cycles | cycles |

| | | |
|---|---|---|
| B_ik | Effective communication bandwidth | Mbps |
| f_k | Fog CPU capacity | GHz |
| lambda_k | Task arrival rate | tasks/s |
| mu_k | Fog service rate | tasks/s |
| rho_k | Fog utilization = lambda_k/mu_k | 0 to 1 |
| D_ij^max | Task deadline | ms |
| T_prop | Propagation delay | ms |
| eta | Congestion sensitivity | >0 |
| LRI_ij^k | Queue-Aware Latency Risk Index | dimensionless |

### 6.2 Analytical and Performance Results

Figure 2 demonstrates that LRI remains comparatively moderate at low utilization but rises rapidly as the fog node approaches saturation. The increase becomes sharper as eta increases, confirming the intended congestion-sensitive behaviour. The horizontal feasibility boundary at LRI=1 provides a direct visual criterion for deadline-aware node rejection. The result is consistent with the analytical limit established in Section 5.3.

Figure 3 shows the corresponding threshold behaviour. When the residual deadline budget is small, only lightly loaded nodes remain feasible. As the residual budget increases, the maximum admissible utilization rises. Increasing service rate also expands the feasible region because more tasks can be processed per unit time. These trends follow directly from the closed-form threshold and therefore do not depend on a fitted data model.

Figure 4 provides a representative comparative evaluation. At low load, the normalized latency of conventional, load-aware, and LRI-based policies remains relatively close. As load increases, the conventional policy becomes increasingly sensitive to queue growth because it does not explicitly impose a task-specific saturation boundary. The load-aware policy improves behaviour by incorporating current congestion, while the proposed LRI-IQATO policy maintains a lower normalized latency in the representative calculation because it rejects nodes once their task-specific risk becomes infeasible.

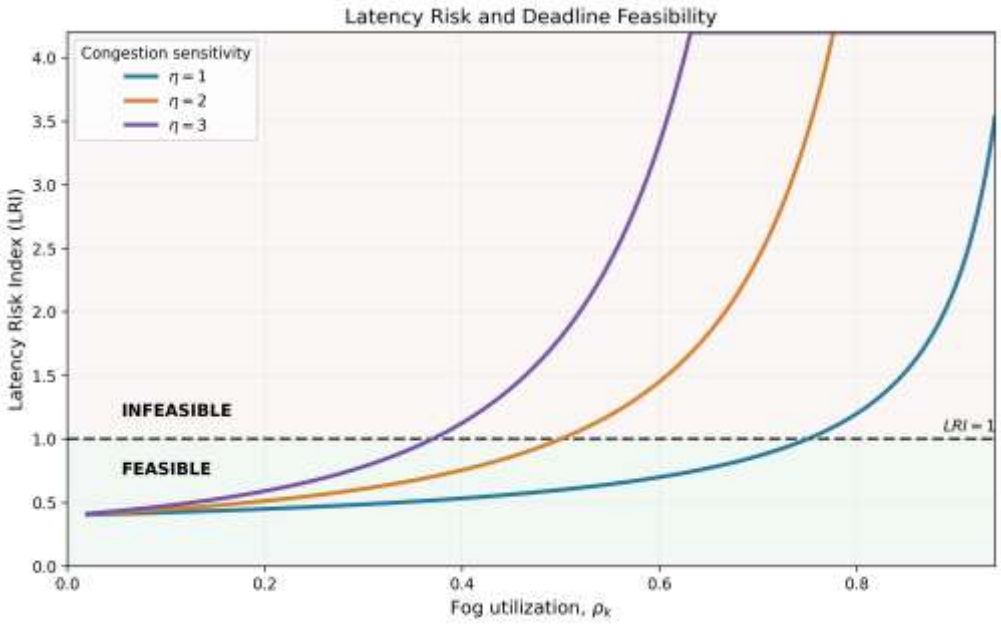



**Figure 4.** Representative normalized latency-risk performance under increasing fog/network load.

The values in Figure 4 are representative outputs generated from the analytical evaluation setup to compare the shape of the competing decision behaviours. They are not claimed as empirical measurements. The main

observation is the increasing separation between a policy without explicit nonlinear congestion protection and the proposed risk-filtered strategy at high load.

### 6.3 Comparative Discussion

Table II positions the proposed framework against representative journal-based approaches. Existing methods demonstrate dynamic resource allocation, deadline-aware optimization, load balancing, Pareto optimization, and reinforcement learning. As summarized in **Table II**, the proposed LRI-IQATO framework differs primarily through the explicit nonlinear risk transformation and the resulting closed-form, task-specific utilization threshold. The comparison therefore focuses on methodological characteristics rather than claiming universal superiority over other methods.

**Table 2.** Comparison of the Proposed LRI-IQATO Framework with Existing Fog Offloading Approaches

| Approach | Fog Offloading | Latency | Queue/Load | Deadline | Intelligent Decision | Nonlinear Risk | Task-Specific Threshold |
|---|---|---|---|---|---|---|---|
| Chang et al. (2021) | Yes | Yes | Resource state | QoS-related | Optimization | No | No |
| Shahryari et al. (2021) | Yes | Yes | Node resources | Yes | GA-PSO | No | No |
| Gao et al. (2020) | Yes | Yes | Dynamic state | Implicit | Predictive | No | No |
| Alam et al. (2019) | Yes | Yes | State-based | QoS | Deep Q-learning | No | No |
| Tomar et al. (2025) | Yes | Yes | Dynamic fog state | Yes | DRL | No | No |
| Panda et al. (2025) | Yes | Yes | Load balancing | Priority/QoS | Multi-objective | No | No |
| Bernard et al. (2024) | Yes | Makespan | Resource state | No | Pareto optimization | No | No |
| Liakath et al. (2025) | Yes | Yes | Dynamic state | Yes | Multi-agent RL | No | No |
| Proposed LRI-IQATO | Yes | Yes | Explicit utilization | Yes | Risk-guided selection | Yes | Yes |

## 7. Discussion

An interpretable link between analytical queue-aware modelling and intelligent task offloading is proposed. The main practical benefit is that it can transform a deadline into a congestion limit that can be specific to the task and the candidate fog node. This reduces the search space for an intelligent controller by eliminating infeasible nodes before ranking or learning-based selection. The formulation also gives explicit sensitivity relationships that can be checked by the system designers. There are some restrictions that should be noted. This is not a universal queueing law, and the queue-risk expression is a proposed nonlinear risk transformation. The basic model assumes an equilibrium effective service process and characterizes queue behaviour with the utilization. Fog environments can include correlated arrivals, multiple servers, non-exponential service times, mobility, packet loss, and task dependency. Future research can be expanded to an M/G/1 or multi-server queue and tested using actual traces or a trained reinforcement-learning controller in the analytical filter.

## 8. Conclusion

This paper presented an intelligent task-offloading framework for latency-sensitive Fog-IoT architectures based on a Queue-Aware Latency Risk Index. The proposed LRI aggregates the transmission, processing, propagation and nonlinear congestion risk and normalizes it by the task deadline. A maximum utilization threshold was derived for each task-node pair from the LRI feasibility condition, which is in closed form. The analytical results confirmed the positive sensitivity of the arrival rate, the negative sensitivity of the service rate, the bandwidth and the CPU capacity, and the strict convexity of the congestion risk with respect to the arrival rate. This leads to a three-regime interpretation that affords a practical way to separate safe, transition, and critical fog operating conditions. The expected non-linearity was validated using a representative computational evaluation, demonstrating the potential of using task-specific threshold filtering to facilitate latency-aware offloading decisions. The framework is designed to be an analysis prerequisite to be joined with an optimisation- or learning-based selector. Further research is needed to test the risk model using real fog traces, generalise the queue formulation to multi-server systems, and combine the analytical eligibility filter with adaptive reinforcement learning for online offloading.

## References

1. Alam, M. G. R., Hassan, M. M., Uddin, M. Z., Almogren, A., & Fortino, G. (2019). Autonomic computation offloading in mobile edge for IoT applications. Future Generation Computer Systems, 90, 149–157. https://doi.org/10.1016/j.future.2018.07.050

2. Ambika, B., Pandian, S. M. V., Sundaravadivel, P., & Vinothkumar, E. S. (2026). Federated deep reinforcement learning-enabled hierarchical Edge–Fog–Cloud architecture for intelligent task offloading in 6G networks. Scientific Reports. https://doi.org/10.1038/s41598-026-57660-6

3. Bernard, L., Yassa, S., Alouache, L., & Romain, O. (2024). Efficient Pareto-based approach for IoT task offloading on Fog–Cloud environments. Internet of Things, 27, 101311. https://doi.org/10.1016/j.iot.2024.101311

4. Chang, Z., Liu, L., Guo, X., & Sheng, Q. (2021). Dynamic resource allocation and computation offloading for IoT fog computing system. IEEE Transactions on Industrial Informatics, 17(5), 3348–3357. https://doi.org/10.1109/TII.2020.2978946

5. Deng, T., Yu, Z., & Yuan, D. (2024). Task offloading optimization in mobile edge computing under uncertain processing cycles and intermittent communications. Computer Networks, 245, 110359. https://doi.org/10.1016/j.comnet.2024.110359

6. Gao, X., Huang, X., Bian, S., Shao, Z., & Yang, Y. (2020). PORA: Predictive offloading and resource allocation in dynamic fog computing systems. IEEE Internet of Things Journal, 7(1), 72–87. https://doi.org/10.1109/JIOT.2019.2945066

7. Jain, V., & Kumar, B. (2022). Blockchain-enabled trusted task offloading scheme for fog computing: A deep reinforcement learning approach. Transactions on Emerging Telecommunications Technologies, 33(11), e4587. https://doi.org/10.1002/ett.4587

8. Kuang, Z., Ma, Z., Li, Z., & Deng, X. (2021). Cooperative computation offloading and resource allocation for delay minimization in mobile edge computing. Journal of Systems Architecture, 118, 102167. https://doi.org/10.1016/j.sysarc.2021.102167

9. Liakath, J. A., Gandhimaruthian, L., Nanajappan, M., &Jegatheeshan, R. (2025). Deadline-aware task scheduling in fog-cloud computing using multi-agent reinforcement learning and software-defined network

security. Concurrency and Computation: Practice and Experience, 37(25–26), e70258. https://doi.org/10.1002/cpe.70258

10. Mikavica, B., & Kostic-Ljubisavljevic, A. (2024). A truthful double auction framework for security-driven and deadline-aware task offloading in fog-cloud environment. Computer Communications, 217, 183–199. https://doi.org/10.1016/j.comcom.2024.01.033

11. Panda, S. K., Pounjula, T., Ravirala, B., &Taniar, D. (2025). An energy-, delay-, and priority-aware task offloading algorithm for fog computing incorporating load balancing. The Journal of Supercomputing, 81(1), 52. https://doi.org/10.1007/s11227-024-06557-0

12. Rajapackiyam, E., Devi, A., Reedy, M. S., Arumugam, U., Vairavasundaram, S., Vairavasundaram, I., & Suresh, V. (2024). An efficient computation offloading in edge environment using genetic algorithm with directed search techniques for IoT applications. Future Generation Computer Systems, 158, 378–390. https://doi.org/10.1016/j.future.2024.04.021

13. Shahryari, O.-K., Pedram, H., Khajehvand, V., & Dehghan TakhtFooladi, M. (2021). Energy and task completion time trade-off for task offloading in fog-enabled IoT networks. Pervasive and Mobile Computing, 74, 101395. https://doi.org/10.1016/j.pmcj.2021.101395

14. Sharma, M., Tomar, A., & Hazra, A. (2026). Delay optimized task offloading and performance evaluation in Fog-Enabled IoT networks. Pervasive and Mobile Computing, 117, 102177. https://doi.org/10.1016/j.pmcj.2026.102177

15. Shi, X. (2023). A task segmentation and computing offload algorithm for mobile edge computing. The Journal of Engineering, 2023(1), e12207. https://doi.org/10.1049/tje2.12207

16. Su, C., Wang, L., Liu, K., & Gao, J. (2024). Task offloading and resource allocation for fog computing in NG wireless networks: A federated deep reinforcement learning approach. IEEE Internet of Things Journal. https://doi.org/10.1109/JIOT.2023.3314496

17. Tomar, A., Sharma, M., Agarwal, A., Jha, A. N., & Jaiswal, J. (2025). Task offloading of IoT devices in a fog-enabled architecture using a deep reinforcement learning approach. Pervasive and Mobile Computing, 112, 102067. https://doi.org/10.1016/j.pmcj.2025.102067

18. Tong, Z., Deng, X., Ye, F., Basodi, S., Xiao, X., & Zhang, S. (2020). Adaptive computation offloading and resource allocation strategy in a mobile edge computing environment. Information Sciences, 537, 116–131. https://doi.org/10.1016/j.ins.2020.05.057

19. Allaoui, T., Gasmi, K., & Ezzedine, T. (2024). Reinforcement learning-basedtask offloading of IoT applications in fog computing: Algorithms and optimization techniques. Cluster Computing, 27(8), 10299–10324. https://doi.org/10.1007/s10586-024-04518-z

20. Pourashory, M., Oustad, E., & Ansari, M. (2026). TELECOM: Latency-aware load-balanced task offloading in vehicular fog computing. IEEE Internet of Things Journal. https://doi.org/10.1109/JIOT.2026.3724770