Federated Regression-Based Remaining Queue Time Estimation in Cloud-Edge Multiprocessor Systems with Fault Tolerance and Cost Optimization

Sarla More[1], Ajay Kumar[2]
[1, 2] Lincoln University College, Malaysia
pdf.sarla@lincoln.edu.my, pdfsv.ajaykumar@lincoln.edu.my

Abstract: Distributed multiprocessor systems operating across cloud-edge infrastructures face unpredictable node failures, network partitions, and communication overheads that complicate accurate estimation of the remaining ready-queue processing time. Classical regression-type estimators have shown strong efficiency gains over simple mean-based estimation in multiprocessor ready queue time estimation, but they assume a centralized, fully observable queue and a single point of computation, which makes them unsuitable for distributed, failure-prone cloud-edge environments. This paper proposes a Federated Regression-based Remaining Queue Time Estimation (FR-RQTE) framework that reformulates the classical auxiliary-variable regression estimator in a federated setting: each cloud-edge node computes a local regression-type estimate of its own remaining queue processing time using locally observed auxiliary information such as queue length, service rate and processor load, and shares only regression coefficients and summary statistics — never raw job-level data — with a federated aggregation server. To prevent from corrupted updated, stragglers and crashed nodes to not distort the global estimates, the fault tolerance layer is appended with the help of partial state replication checkpoints, redundant node grouping and the byzantine resilient trimmed weighted aggregation rule. A cost-optimization layer jointly minimizes communication cost, computation cost and expected recovery cost subject to a bounded estimation-error constraint, giving system administrators a tuneable trade-off between accuracy and operating expense. The proposed framework is expected to outperform both the classical centralized regression estimator and existing non-regression federated queue-estimation approaches in terms of estimation efficiency, fault resilience, and cost efficiency, particularly under node failure rates and network degradation typical of edge-cloud multiprocessor deployments.

Keywords: Federated regression estimator; Remaining queue processing time; Fault-tolerant federated aggregation; Cost optimization; Cloud-edge multiprocessor systems

Introduction

Estimation of the ready queue processing time in multiprocessor environments has long relied on classical sampling-based estimators, including mean, ratio and regression-type estimators [1,2,3] that exploit an auxiliary variable correlated with the processing time, such as queue length, arrival rate or processor utilization, to improve estimation efficiency over simple random sampling. Regression-type estimators [4], in particular, have been shown to substantially reduce estimation variance when the auxiliary variable is well correlated with the ready queue processing time, and this efficiency gain has been established for single-node and centrally observable multiprocessor settings. Modern computing infrastructures [5], however, are rarely centralized. Cloud-edge deployments, autonomous vehicle fleets, smart factories, and IoT scheduling systems [6] distribute ready queues and scheduling decisions across geographically dispersed nodes. In such settings, a centralized regression estimator cannot be applied directly for three reasons.

First, raw queue and job-level data at each node is often privacy-sensitive and cannot be pooled centrally. Second, individual edge and cloud nodes are prone to crashes, network partitions, and Byzantine (corrupted or adversarial) behavior, so any aggregation mechanism [7, 8] must tolerate missing or unreliable local estimates.

Third, every round of coordination between nodes and a central server incurs communication and computation cost [9], and in bandwidth-constrained or degraded-network conditions this cost must be actively managed rather than ignored. Federated Learning offers a template for privacy-preserving collaborative estimation without centralizing raw data, and it has already been combined with lottery scheduling for privacy-preserving remaining queue time prediction.

That framework, however, does not exploit the statistical efficiency of regression-type estimation, nor does it explicitly address fault tolerance against node failures or the operating cost of federated coordination [10]. This paper extends that line of work by embedding the classical regression-type estimator into a federated framework, and by adding explicit fault-tolerance and cost-optimization layers so the resulting estimator remains accurate, resilient and economical to run at scale.

Problem Definition

Consider a cloud-edge infrastructure with H distributed nodes, each maintaining a local ready queue, local processors, and a locally observed auxiliary variable correlated with the remaining processing time of jobs in that queue (for instance, current queue length, mean service rate, or processor utilization). At recovery time, following a partial system failure, jobs at each node fall into three categories: fully completed jobs whose results are ready, partially completed jobs in a recoverable intermediate state, and unexecuted jobs still waiting in the local queue. When the estimation is required, some of the nodes may crashed, unreachable or represents corrupted values. The identified problem is to estimate the total remaining ready queue processing time of all available H nodes with the assigned properties of estimator which includes (i) at each node for a regression type estimator achieves the variance reduction benefit (ii) it can tolerate the failure of nodes, the straggling and the adversarial corruption without global estimates biasedness. (iii) to leave the node of origin it never requires raw job-level. (iv) to track the computation cost and keep the communication, it is targeted for an estimation accuracy constraint.

Research Objectives

The first objective of this work is to reformulate the classical regression-type estimator for remaining ready-queue processing time as a per-node local estimator, using locally available auxiliary variables so that no cross-node data pooling is required. The second objective is to design a federated aggregation rule that combines these local regression estimates into a global estimate with a provably lower variance than a simple federated mean, similar in spirit to the efficiency gain the regression estimator achieves over the sample mean in the classical single-node case. The third objective is to embed fault tolerance directly into the aggregation rule, through redundant node grouping and a Byzantine-resilient trimmed weighting scheme, so that node crashes, stragglers, or corrupted local estimates are automatically down-weighted or redistributed without manual intervention. The fourth objective is to formulate and solve a joint cost-optimization problem that minimizes total communication, computation and expected recovery cost subject to a bound on estimation mean squared error, giving administrators a single tenable accuracy-cost trade-off parameter. For the federated regression estimate, to represent the statistical sampling variance and the uncertain induced node failure, the model needs to derive the adaptive confidence interval. So that the defensive uncertainty bound can be maintained to recover the planning decisions and this was the fifth objective of the proposed model.

Proposed System Architecture

The proposed architecture is described in figure 1: Each cloud-edge node has their own ready queue, a local an auxiliary-variable monitor, and a local regression estimator module. Nodes are grouped in a redundancy. If a peer node in a group fails, then the other nodes in the same group can replace it, as long as the total number of nodes in the group is ρ (redundancy factor). node is unable to respond in a certain round of coordination. At each node, only its regression coefficient is sent, and its local mean estimate to a federated, and an auxiliary-variable summary statistic (not raw per-job data) aggregation server. The aggregation server combines using a trimmed weighting rule that is Byzantine resilient. the local estimates send them to a cost controller which monitors communication volume, computation load and estimated recovery cost, each round, to adjust the redundancy factor and coordination frequency to ensure cost of operation is within budget and takes into account the accuracy constraint.

Node Layer (repeated at each of H nodes): Local ready queue → Auxiliary variable monitor (queue length, service rate, utilization) → Local regression estimator → Redundancy-group replication → Encrypted transmission of (coefficient, mean, variance) to aggregation server.

Aggregation Layer: Failure/straggler detection → Byzantine-resilient trimming → Reliability-weighted regression aggregation → Global estimate and confidence interval → Broadcast of aggregated model parameters back to nodes.

Cost Control Layer: Communication cost tracker, computation cost tracker, expected recovery cost estimator → Joint cost-accuracy optimizer → Adjusted redundancy factor and round frequency fed back to the node and aggregation layers.

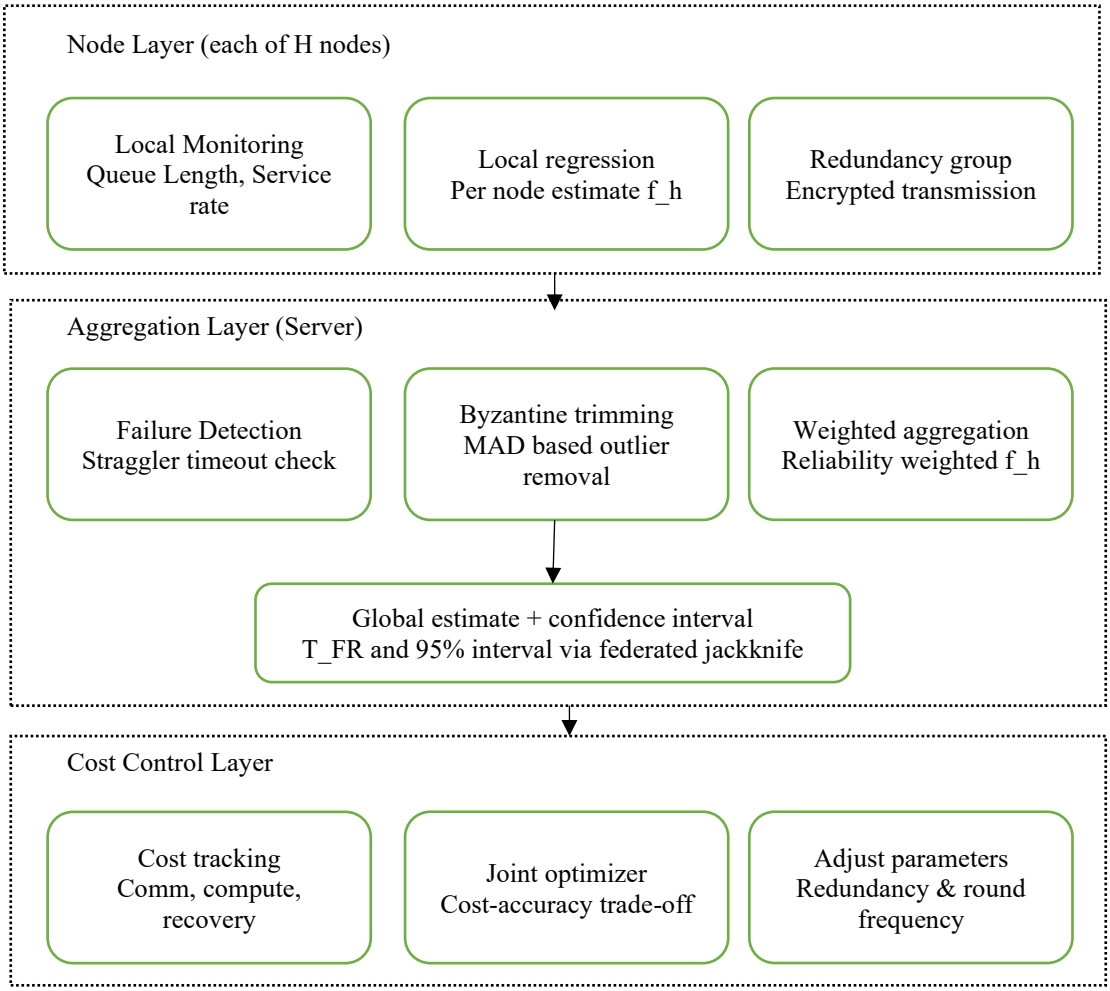

Figure 1. Federated Regression Estimation Architecture with Fault Tolerance and Cost Control

The FR-RQTE architecture consists of three "coordinated layers". The node layer operates independently at each of the H cloud-edge nodes, and a local monitoring keeps track of auxiliary variables such as the length of the queue and the rate of the service; the local regression estimator computes an estimate $\hat{t}_h$ for each node using the local information, and a redundancy group replicates and encrypts the estimate before it is sent—no raw data about the jobs leaves the node! These local estimates are computed on the aggregation layer that is located in the federated server where it detects unresponsive stragglers, trims updates that are corrupted and outlier using a MAD based rule, and then aggregates the remaining estimates to get a global estimate of T_{FR}, with the corresponding 95% confidence interval computed using a federated jackknife. Lastly, the cost-control layer updates the accuracy of the estimation and the cost (dashed feedback path) of communication, computation and recovery to the node layer and aggregation layer, respectively, and continuously optimizes the system to achieve the best accuracy at minimum cost.

Proposed Federated Regression Estimation Model

Classical Regression-Type Estimator (Baseline)

For a single centrally observable node, the classical regression-type estimator of the remaining ready-queue processing time uses an auxiliary variable X (such as queue length or service rate) correlated with the processing time t:

$$\hat{t}\_reg = \bar{t} + b\,(\bar{X} - \bar{x})$$

where $\bar{t}$ is the sample mean remaining time, $\bar{X}$ is the population mean of the auxiliary variable, $\bar{x}$ is its sample mean, and b is the sample regression coefficient of t on X. This estimator is more efficient than the simple sample mean whenever X is well correlated with t, but it requires all queue data to be centrally observable — an assumption that does not hold in a distributed cloud-edge setting.

Local Federated Regression Estimator

Each node h computes its own regression-type estimate using only its own locally observed data:

$$\hat{t}\_h = \bar{t}\_h + b\_h\,(\bar{X}\_h - \bar{x}\_h), \qquad b\_h = S\_tXh \,/\, S^2\_Xh$$

where $\bar{t}\_h$ and $\bar{x}\_h$ are the local sample means of remaining time and the auxiliary variable at node h, $\bar{X}\_h$ is the locally known population mean of the auxiliary variable, S_tXh is the local sample covariance of t and X, and S^2_Xh is the local sample variance of X. No raw job-level records are required beyond the node boundary — only the scalar quantities $\hat{t}\_h$, b_h and the associated variance estimate are transmitted.

Reliability-Weighted Federated Aggregation

The federated server combines the H local regression estimates into a global estimate using reliability-based weights:

$$T\_FR = \Sigma\,(h = 1 \text{ to } H)\ w\_h \cdot \hat{t}\_h\,, \qquad \Sigma\, w\_h = 1, \qquad w\_h = (n\_h\, r\_h) \,/\, \Sigma\_j\,(n\_j\, r\_j)$$

where n_h is the local sample size at node h and $r\_h \in [0,1]$ is a node reliability score (one minus the node's estimated failure probability over the reporting window). In the global estimate the more reliable local samples receives the greater weights in proportion to larger nodes.

Fault Tolerance Mechanism

There are two mechanisms which are complementary in nature that provide fault tolerance.

1). Redundancy-group failure redistribution: The nodes are pre-assigned to groups of size ρ, G_h, and if node h doesn't report within the coordination timeout, its weight is re-distributed among the other nodes in its group:

$$w\_h' = w\_h \,/\, |G\_h'| \quad \text{for each surviving peer in } G\_h' \subseteq G\_h$$

2). before the weighted aggregation is performed, any estimate $\hat{t}\_h$ not in the interval [median − k·MAD, median + k·MAD] of the estimates that were reported from the previous round is dropped from the aggregation, so that one corrupted or adversarial node does not have an unduly large influence on the global estimate.

Cost Optimization Function

Total operating cost is the sum of the communication, computation and expected recovery costs:

$$C\_total = C\_comm + C\_comp + C\_recovery$$

$$C\_comm = \Sigma^h\, \kappa\_h\, \psi\_h\,, \quad C\_comp = \Sigma^h\, \gamma\_h\, \tau\_h\,, \quad C\_recovery = \Sigma^h\, P\_fail,h\, \rho\_h\, \delta$$

In this case where the "ψ_h" is the amount of the message per round on node h, the "κ_h" is the cost of a unit of message, the "τ_h" is the local computation time on node h, the "P_fail,h" is an estimate of failure probability at node h, the "ρ_h" is the redundancy factor at node h, and the "δ" is per-unit recovery cost. The optimization problem is a joint minimization of total cost of the estimation problem with a bound on estimation error:

$$\min C\_total \quad \text{subject to} \quad E[(T\_FR - T\_true)^2] \le \varepsilon$$

The cost controller varies the number of rounds and the redundancy factor (ρ) to trace out an accuracy-cost frontier and provides a point on this frontier as services under the service-level agreement (SLA) are chosen by the administrator. The cost controller varies the number of rounds and the redundancy factor (ρ) to trace out an accuracy-cost frontier, and supplies a point on this frontier as services under the SLA are picked by the administrator.

Adaptive Federated Confidence Interval

The variance of the global federated regression estimate is obtained through a federated jackknife procedure across surviving nodes, giving an adaptive (1 − α) confidence interval for the recovery-time estimates:

$$P(T\_FR - 1.96\sqrt{Var(T\_FR)} \leq T\_true \leq T\_FR + 1.96\sqrt{Var(T\_FR)}) = 0.95$$

This interval widens automatically as more nodes are trimmed or fail to report, giving system administrators a realistic, failure-aware bound on expected recovery time rather than a false sense of precision.

Novel Contributions

- A federated reformulation of the classical regression-type estimator for remaining ready-queue processing time, retaining its variance-reduction property in a fully distributed, privacy-preserving setting.
- A fault-tolerant aggregation rule combining redundancy-group failure redistribution with Byzantine-resilient trimmed weighting, protecting the global estimate against node crashes, stragglers and corrupted updates.
- A joint cost-optimization layer that minimizes communication, computation and expected recovery cost subject to a bounded estimation-error constraint, exposing a single tunable accuracy-cost trade-off.
- An adaptive, failure-aware federated confidence interval derived through a federated jackknife procedure across surviving nodes.

Table 1. Expected Advantages over Classical and Existing Federated Approaches

Parameter	**Classical Regression Estimator**	**Existing Federated (Non-regression)**	**Proposed FR-RQTE**
Estimation efficiency	High (single node only)	Moderate	High (federated, variance-reduced)
Fault tolerance	None	Limited	Strong (redundancy + Byzantine-resilient)
Cost awareness	Not addressed	Partial	Explicit joint optimization
Scalability	Low	High	High
Privacy	Low (centralized)	High	High

Real-World Applications

- Cloud disaster recovery and service-level agreement planning
- Edge computing platforms with intermittent connectivity
- Autonomous vehicle fleets and UAV swarm scheduling
- Smart manufacturing and industrial IoT scheduling systems
- Healthcare distributed servers with strict data-privacy requirements
- Smart datacentre capacity and cost planning

Conclusion

1. Classical regression-type estimators offer strong efficiency gains for ready-queue processing time estimation but assume centralized, fully observable data, which is incompatible with privacy-sensitive, failure-prone cloud-edge multiprocessor systems.
2. The proposed FR-RQTE framework reformulates the regression-type estimator for a federated setting, combining it with redundancy-group and Byzantine-resilient fault tolerance and a joint cost-optimization layer.
3. The framework is expected to deliver lower estimation variance than non-regression federated approaches, stronger resilience to node failure than the classical centralized estimator, and a tenable cost-accuracy trade-off suited to real deployment budgets.
4. Open directions include empirical validation on real distributed schedulers (Linux CFS, YARN, Kubernetes), handling non-IID auxiliary-variable distributions across nodes, and hardening the trimming rule against coordinated adversarial collusion.

References

1. More, S., and Shukla, D. A generalized approach in multiprocessor environment using regression type estimator and cost analysis. Reliability: Theory and Applications (RT&A), Vol. 17, Issue 2, 2022. https://doi.org/10.24412/1932-2321-2022-268-217-234.
2. More, S., and Shukla, D. Sampled Ready Queue Processing Time Estimation Using Size Measure Information in Multiprocessor Environment. Reliability: Theory and Applications, Vol. 16 (3), 2021. https://doi.org/10.24412/1932-2321-2021-363-63-80.
3. Shukla, D., and More, S. Modified Group Lottery Scheduling Algorithm for Ready Queue Mean Time Estimation in Multiprocessor Environment. Reliability: Theory & Applications (RT&A), Vol. 15, Issue 4, 2020. https://doi.org/10.24411/1932-2321-2020-14008.
4. More, S., and Kumar, A. Privacy-Preserving Remaining Ready Queue Processing Time Prediction Using Federated Learning and Lottery Scheduling. SGS Initiative, Vol. 1, No. 7, 2026.
5. Mansouri, M., Önen, M., and Ben Jaballah, W. Learning from Failures: Secure and Fault-Tolerant Aggregation for Federated Learning. In Proc. 38th Annual Computer Security Applications Conference (ACSAC), pp. 146–158, 2022. DOI: 10.1145/3564625.3568135.
6. Liao, Y., Xu, Y., Xu, H., Chen, M., Wang, L., and Qiao, C. Asynchronous decentralized federated learning for heterogeneous devices. IEEE/ACM Transactions on Networking, vol. 32, no. 5, pp. 4535–4550, 2024. DOI: 10.1109/TNET.2024.3424444.
7. Feng, J., Liu, L., Pei, Q., and Li, K. Min-max cost optimization for efficient hierarchical federated learning in wireless edge networks. IEEE Transactions on Parallel and Distributed Systems, vol. 33, no. 11, pp. 2687–2700, 2021.
8. Luo, S., Chen, X., Wu, Q., Zhou, Z., and Yu, S. HFEL: Joint edge association and resource allocation for cost-efficient hierarchical federated edge learning. IEEE Transactions on Wireless Communications, vol. 19, no. 10, pp. 6535–6548, 2020.
9. Wang, C., Yang, Y., and Zhou, P. Towards efficient scheduling of federated mobile devices under computational and statistical heterogeneity. IEEE Transactions on Parallel and Distributed Systems, vol. 32, no. 2, pp. 394–410, Feb. 2021. DOI: 10.1109/TPDS.2020.3005079.
10. Sun, W., Liu, F., Zhang, W., and Wang, R. Research on computing task scheduling method for distributed heterogeneous parallel systems. Scientific Reports, vol. 15, Art. no. 9142, Mar. 2025. DOI: 10.1038/s41598-025-94068-0.