Computers, Materials & Continua

Tech Science Press

https://doi.org/10.32604/cmc.2026.088018

ARTICLE

5G-Aware Incremental Routing and Scheduling for Dynamic Time-Triggered Flow Admission in Time-Sensitive Networks

Zhihao Liu[1,2], **Yi Zhang**[3], **Wei Zhang**[1,2], **Jian Wang**[4], **Huiling Shi**[1,2] **and Xiaolong Wang**[1,2,*]

[1]Key Laboratory of Computing Power Network and Information Security, Ministry of Education, Shandong Computer Science Center (National Supercomputer Center in Jinan), Qilu University of Technology (Shandong Academy of Sciences), Jinan, China
[2]Shandong Provincial Key Laboratory of Computing Power Internet and Service Computing, Shandong Fundamental Research Center for Computer Science, Jinan, China
[3]School of Information and Communication Engineering, University of Electronic Science and Technology of China, Chengdu, China
[4]College of Science, China University of Petroleum (East China), Qingdao, China

*Corresponding Author: Xiaolong Wang. Email: wangxlong@sdas.org

Received: 26 June 2026; Accepted: 21 August 2026; Published: 15 September 2026

ABSTRACT: Mobile edge services require deterministic communication across Time-Sensitive Networking (TSN) and 5G access, where the standardized integration architecture exposes the 5G System (5GS) to the TSN controller as a logical bridge. We study dynamic admission of time-triggered (TT) flows using reported 5GS bridge delay and TSN-to-5GS Quality of Service (QoS) mapping in route selection and Gate Control List (GCL) scheduling. Arrivals and departures can split available transmission time into noncontiguous windows. Online insertion preserves admitted schedules but may reduce subsequent schedulability, whereas full recomputation can restore schedulability but changes many routes and GCL entries, complicating coordinated activation. Coupling routing and GCL scheduling under timing, bridge-delay, QoS-mapping, and a bound on changes to admitted schedules yields an NP-hard problem. To address it, we propose a two-timescale scheduling mechanism. The fast timescale uses current 5GS bridge information to place arrivals without modifying admitted flows. Fragmented windows, repeated insertion failures, or changes in reported 5GS state invoke the slower timescale, which sequentially reschedules a bounded subset of admitted flows and commits only feasible improvements. At 0.95 offered load across A380, CEV, and Ring6, admission improves by 14.8–18.5 percentage points over online-only scheduling and remains within 1.7–2.2 points of full recomputation, while per-event runtime falls by over one order of magnitude with limited GCL changes.

KEYWORDS: Mobile edge computing; 5G-aware scheduling; time-sensitive networking; time-triggered flow; gate control list; online admission; selective rescheduling

1 Introduction

Industrial and mobile applications increasingly run latency-sensitive control and data processing close to users, machines, and production systems. Mobile Edge Computing (MEC) supports this shift by moving computation, storage, and service control from remote cloud data centers toward nearby edge servers. MEC surveys and European Telecommunications Standards Institute (ETSI) specifications describe this shift as a way to support low-latency services through nearby computing and edge resource management [1,2]. For many applications, however, nearby computation creates value only when the communication path can deliver predictable service.

Proximity to computation is therefore not enough. A control task may run on an edge server, but its deadline still depends on the network path between the device, the 5G access segment, the Time-Sensitive Networking (TSN) domain, and the edge node. Harmatos and Maliosz studied 5G, TSN, and edge-computing integration for smart manufacturing [3], while Rost and Kolding analyzed performance issues in integrated 5G and TSN networks [4]. These studies reflect the same practical requirement: industrial MEC services need bounded latency, time synchronization, and traffic isolation on the communication path.

TSN supplies the scheduling machinery for this deterministic communication. The Institute of Electrical and Electronics Engineers (IEEE) 802.1AS standard supports time synchronization, IEEE 802.1Qbv defines scheduled traffic through the Time-Aware Shaper (TAS) and Gate Control List (GCL) configurations, and IEEE 802.1Qcc supports centralized configuration [5–7]. In 5G-TSN, the 5G System (5GS) can be exposed to the TSN control plane as a 5GS logical bridge through Device-side TSN Translator (DS-TT), Network-side TSN Translator (NW-TT), and TSN Application Function (TSN AF) entities [8,9]. The controller then schedules over bridge and port management information, bridge delay, Quality of Service (QoS) mapping configuration, and forwarding availability, rather than over 5G physical-layer details.

Dynamic time-triggered (TT) admission introduces two coupled problems. Runtime flow churn can leave enough total idle slots but no contiguous window for a new frame. Moreover, a route crossing the 5GS logical bridge must satisfy the reported bridge-delay, QoS-mapping, and forwarding-availability state. The problem is therefore 5G-aware at the TSN controller, without extending its decision space to radio scheduling or allocation.

Existing joint routing and scheduling methods are effective when the TT-flow set is known in advance or when the controller can afford batch recomputation. Prior studies considered traffic scheduling for integrated 5G-TSN systems and learning-assisted end-to-end optimization [10,11]. Online scheduling reduced current-flow decision latency [12], but a foreground-only inserter may gradually consume scarce contiguous windows. Full recomputation can recover schedulability, but it may change many routes and GCL entries. Deploying such a broad update requires coordinated activation across the affected bridges and end systems. If activation is non-atomic or mistimed, packets may encounter inconsistent old and new routes or gate states, causing transient queuing, loss, reordering, or deadline violations. Dynamic admission must therefore balance fast decisions and future schedulability against the scope and coordination cost of schedule transitions.

To balance these requirements, we develop a two-timescale, 5G-aware mechanism. Foreground bounded search filters routes using current 5GS information and places GCL windows without moving admitted flows. The background lane is invoked when a composite trigger identifies depleted placement options from fragmentation, reduced window flexibility, recent insertion failures, or adverse 5GS state. It assigns its bounded budget first to bottleneck-relevant flows and repairs them sequentially. Feasible coordinate updates are retained in a candidate state, and the aggregate incremental GCL delta is committed only after final feasibility and utility validation.

The main contributions are as follows.

- We formulate controller-side dynamic TT-flow admission as a 5G-aware incremental routing and scheduling problem that combines TAS/GCL constraints with runtime 5GS bridge-information inputs, and establish that its underlying joint feasibility problem is NP-hard.
- We design the 5G-Aware Foreground Online Insertion for TT-Flow Admission algorithm, which performs 5G-aware route filtering, flexibility-aware route ranking, and bounded feasible-window propagation without relocating admitted TT flows.

- We develop the Fragmentation-Aware Background Selective Rescheduling for GCL Repair algorithm, which assigns a bounded disturbance budget to bottleneck-relevant flows and sequentially re-evaluates their routes and windows, improving long-term admission while controlling background computation and GCL modifications.

The remainder of this paper is organized as follows. Section 2 reviews MEC resource management and 5G-TSN deterministic scheduling. Section 3 presents the system model and problem formulation. Section 4 describes the proposed two-timescale mechanism. Section 5 reports the performance evaluation. Section 6 concludes the paper.

2 Related Work

Prior work around this problem falls into two groups: MEC resource management and 5G-TSN deterministic scheduling. The distinction is useful because edge computing explains where services run, while TSN and 5G-TSN scheduling determine whether their traffic can meet deterministic timing requirements after runtime changes.

2.1 Mobile Edge Computing and Edge Resource Management

Broader MEC resource-management work mainly decided where computation should run and how scarce edge resources should be assigned. Zhang and Debroy surveyed MEC resource management across computation offloading, service placement, communication control, and mobility-aware allocation [1]. ETSI MEC 003 defined the architectural role of MEC platforms and services, providing the system-level reference model used by many later studies [2]. These works established the edge-computing context of this paper, but did not specify how deterministic transmission windows should be maintained after runtime traffic changes.

Industrial MEC services also required a deterministic network substrate. Harmatos and Maliosz connected 5G, TSN, and edge computing in a smart-manufacturing architecture [3]. Rost and Kolding evaluated performance aspects of integrated 5G and TSN networks [4]. Following this communication-oriented MEC line, our work studies how an edge-side TSN controller admits TT flows and updates GCL resources under runtime traffic changes; it does not design an offloading policy.

2.2 5G-TSN Integration and Dynamic Deterministic Scheduling

TSN standards provided the deterministic Ethernet basis: IEEE 802.1AS defined time synchronization, IEEE 802.1Qbv scheduled traffic control, and IEEE 802.1Qcc centralized network configuration [5–7]. Third Generation Partnership Project (3GPP) TS 23.501 further specified how the 5GS can appear to the TSN side as a logical or virtual bridge through translator and application-function entities [8]. Sasiain et al. surveyed this 5G-TSN convergence and identified industrial communication as a major use case [9].

Static and batch TSN schedulers usually assumed that the flow set was known before optimization. Wang et al. formulated joint routing and scheduling with cyclic queuing and forwarding for TSN [13]. Yang and Yu studied traffic scheduling for 5G-TSN integrated systems [10]. Yang et al. proposed a performance-balanced scheduling algorithm for diverse TSN scenarios [14]. These methods could produce high-quality schedules, but applying them after every arrival, departure, or 5GS information change can be too disruptive for online admission.

Several studies moved closer to hybrid or dynamic 5G-TSN operation. Wang et al. combined reinforcement learning and particle swarm optimization for end-to-end TSN-5G traffic scheduling [11]. Cheng et al. studied joint time-frequency resource scheduling over a cyclic-queuing-and-forwarding-based TSN-5G

system [15]. Cai et al. addressed dynamic QoS mapping and adaptive semi-persistent scheduling in 5G-TSN networks [16], while Satka et al. proposed QoS-MAN for TSN-5G flow mapping [17]. Larranaga et al. examined configured-grant scheduling for 5G-TSN support in Industry 4.0 [18], and Wang et al. analyzed time synchronization for integrated 5G and TSN networking [19].

Online and incremental TSN schedulers reduced response time for runtime traffic events. InNetSched-uler showed that time- and event-triggered critical traffic could be scheduled with in-network online decisions [12]. However, foreground insertion alone could not restore contiguous scheduling opportunities once successive events had fragmented idle windows or concentrated route bottlenecks. Our method retains online response while using the reported 5GS bridge-information snapshot as a feasibility input and selectively repairing bottleneck-relevant routes and windows instead of rebuilding the full schedule.

3 System Model and Problem Formulation

The system model separates controller-visible 5GS information from the TSN scheduling configuration, and then defines the dynamic TT-flow model, GCL idle-window metrics, and foreground and background optimization problems.

3.1 5GS Information Available to the TSN Controller

As shown in Fig. 1, the standardized integration architecture exposes the 5GS as a logical TSN-capable bridge between device-side and network-side TSN domains [8]. DS-TT, NW-TT, user equipment, gNodeB, and the User Plane Function jointly provide bridge-like forwarding behavior. The TSN AF supplies the controller-visible information used here: bridge and port management information, maximum bridge delay per port pair and traffic class, QoS mapping, and forwarding availability.

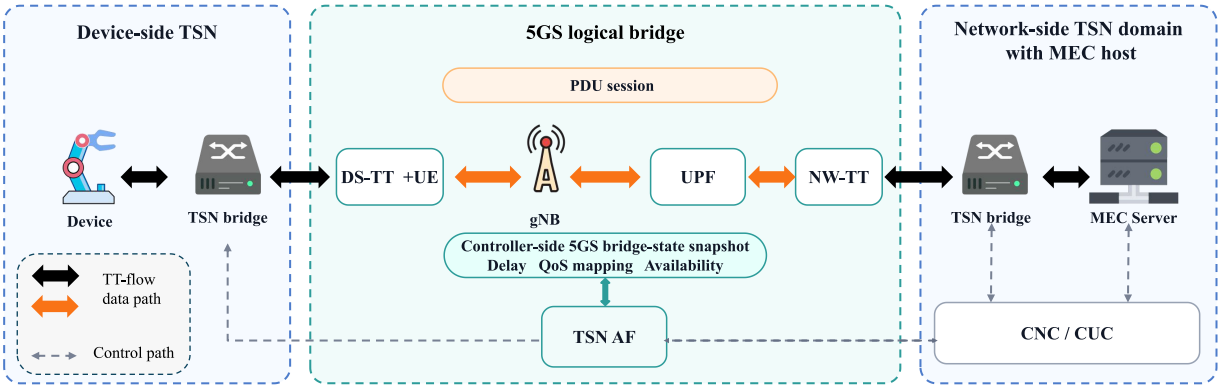

Figure 1: MEC-oriented 5G-TSN architecture and controller-side 5G-aware scheduling view. CNC: Centralized Network Configuration; CUC: Centralized User Configuration.

For each decision epoch, the scheduler reads the snapshot $\mathcal{E}_t = (I_t^{br}, \Delta_t^{pp}, M_t^{QoS}, A_t^{fwd})$. Here, I_t^{br} is bridge and port management information, Δ_t^{pp} is the configured or reported maximum bridge delay (seconds) per port pair and traffic class, M_t^{QoS} is the TSN-to-5GS QoS mapping, and A_t^{fwd} is logical-hop availability. Thus, $\mathcal{E}_t$ is not a new 3GPP entity or a radio-delay estimator; it is the controller's current scheduling input. The resulting admission guarantee is conditional on the validity of these reported bounds. Radio mechanisms that support the selected QoS profile, such as configured grant, semi-persistent scheduling, and HARQ policy, remain within the 5GS [16,18].

A logical hop e is termed a 5GS-related hop if it traverses the 5GS logical bridge; the set of such hops is denoted by $\mathcal{L}_{5GS}$. For scheduling decisions, let $q_t(f_i, e) = 1$ when the profile selected by M_t^{QoS} on logical hop e satisfies the packet-delay and packet-error-rate requirements associated with f_i, and let $a_t(e) = 1$ when e is currently available. A route R is 5GS-state-feasible only if $q_t(f_i, e) = a_t(e) = 1$ for every $e \in R \cap \mathcal{L}_{5GS}$ and its reported delay is included in the deadline check. These predicates prevent the TSN scheduler from ranking a route unsupported by the current 5GS state.

3.2 Network and Flow Model

The network is a directed graph $G = (V, \mathcal{L})$ containing end systems, TSN switches, and a 5GS logical bridge. $\mathcal{P}$ is the set of TAS/GCL egress ports and $\mathcal{P}(R)$ those on route R. Slots last τ seconds and the hyperperiod contains $\mathcal{H}$ slots, all referenced to IEEE 802.1AS time. The configured residual relative synchronization-error bound ϵ_{sync} (seconds), including the bridge boundary, is included in bridge-adjacent guard bands [5,19].

A TT flow is $f_i = (v_i^s, v_i^d, c_i, P_i, l_i, r_i, D_i, \varepsilon_i, t_i^{arr}, t_i^{dep})$. Here, c_i is a dimensionless class; $P_i, r_i, D_i, t_i^{arr}, t_i^{dep}$ are in seconds; l_i is in bytes; and ε_i is the maximum packet-error rate. Lifecycle arrival t_i^{arr} is an admission event, not a packet release. After activation at a hyperperiod boundary, instance q is released at $\rho_i + q\pi_i$ slots, where $\pi_i = P_i/\tau$, $\rho_i = \lceil r_i/\tau \rceil$, and $d_i = \lfloor D_i/\tau \rfloor$. We use $0 < D_i \le P_i - r_i$ and $\pi_i \mid \mathcal{H}$. The active admitted set is $\mathcal{F}^{act}(t) = \{f_i \in \mathcal{F}^{adm}(t) \mid t_i^{arr} \le t < t_i^{dep}\}$.

For each admitted flow, the controller stores $\Omega_i = (R_i, \Phi_i, W_i)$, where R_i is the selected route, Φ_i contains per-hop offsets, and W_i contains reserved transmission windows. When $t = t_i^{dep}$, f_i leaves $\mathcal{F}^{act}(t)$ and the controller releases Ω_i. For each reserved window $w_{i,h}^{(q)} \in W_i$, it appends the released-window record $(e_{i,h}, s_{i,h} + q\pi_i, n_{i,h})$ to $\mathcal{W}^{rel}$. A departure does not relocate other admitted flows; it only changes the idle-window structure observed by later arrivals and can contribute to the background trigger.

The scheduling configuration is $C_t = (\mathcal{F}^{act}(t), \mathcal{R}_t, \Phi_t, \mathcal{G}_t)$; $\mathcal{E}_t$ is an external input. On its update, affected active routes are revalidated before new admissions, and hard violations receive maximum 5GS-risk priority in $\mathcal{F}^{elig}$. If bounded repair finds no feasible remapping, the affected stream is suspended from deterministic service at the next common cycle boundary and enters 5GS reconfiguration or stream re-admission; no guarantee is asserted while its current reported bound is violated.

3.3 GCL Idle-Window Metrics

TAS controls each egress port by opening and closing queue gates according to a GCL. As illustrated in Fig. 2, TT flows obtain reserved transmission windows, while other intervals are available to lower-priority traffic or remain idle. In a static setting, the controller can compute a complete GCL for a known flow set. In a dynamic setting, the controller must insert and remove reservations while preserving existing deterministic transmissions.

The idle-window structure of each egress port is therefore a scheduling resource. A port may have sufficient unused time in aggregate but no contiguous interval long enough for the next TT frame. This distinction matters because each hop of a TT flow needs a feasible window, and adjacent windows must satisfy propagation, processing, and possible 5GS bridge-delay constraints. Fragmentation on one bottleneck port can invalidate an otherwise feasible multi-hop route.

For an egress port p, the GCL idle windows are $\mathcal{W}_p^{idle} = \{(g_{p,j}, \Delta_{p,j})\}_{j=1}^{m_p}$, where start $g_{p,j}$ and length $\Delta_{p,j}$ are in slots. The total idle time $I_p = \sum_{j=1}^{m_p} \Delta_{p,j}$ and maximum contiguous window $MCW_p = \max_{1 \le j \le m_p} \Delta_{p,j}$ are also measured in slots.

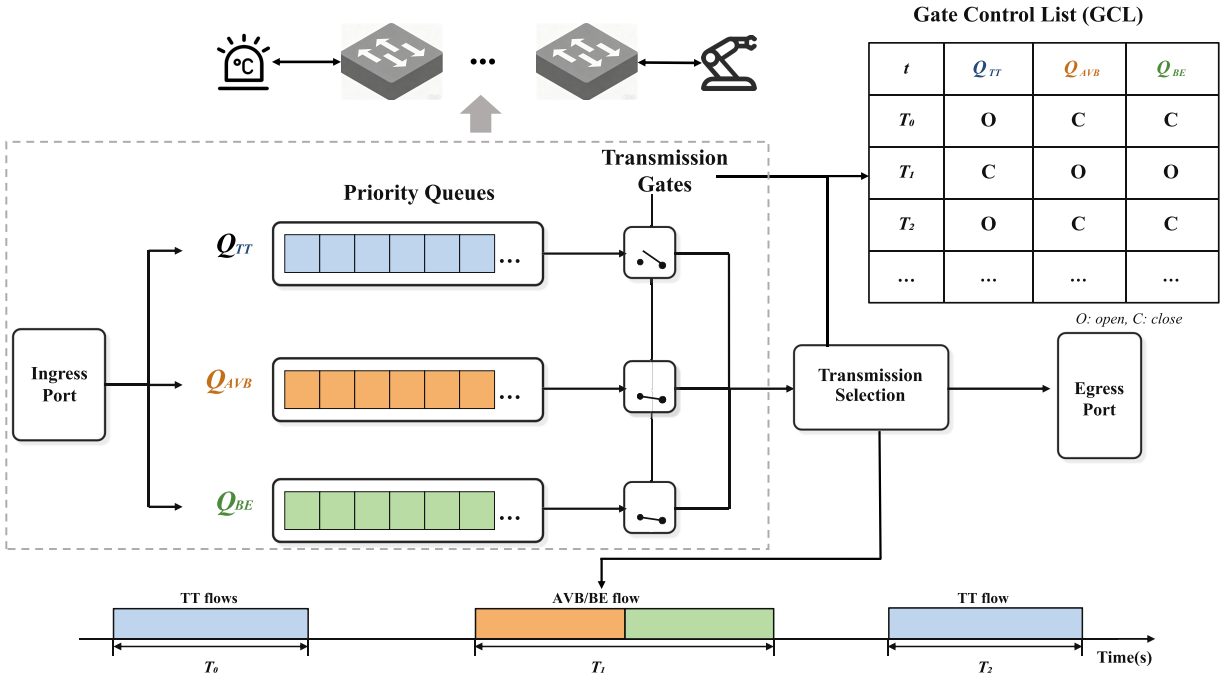

Figure 2: TAS/GCL gate states and reserved transmission windows.

The GCL Fragmentation Index (FI) is defined as

$$FI_p = \begin{cases} 0, & I_p = 0 \text{ or } m_p \le 1, \\ 1 - \frac{MCW_p}{I_p}, & \text{otherwise.} \end{cases} \tag{1}$$

The value FI_p measures fragmentation among existing idle windows. If $I_p = 0$, the port has a capacity shortage rather than a fragmentation problem.

For a transmission window of length n, the port-local flexibility and route-level bottleneck estimate are

$$\hat{h}_p(n) = \sum_{j=1}^{m_p} \max(\Delta_{p,j} - n + 1, 0). \quad h_R(n) = \min_{p \in \mathcal{P}(R)} \hat{h}_p(n), \tag{2}$$

where $\hat{h}_p(n)$ counts possible start slots at one egress port and is not an end-to-end feasible-placement count. Let $n_i^{max} = \max_h n_{i,h}$ and let $\mathcal{N}_t$ contain these lengths for active flows and the most recent L_f arrivals. For nonempty $\mathcal{N}_t$, $Flex_p(C_t) = |\mathcal{N}_t|^{-1} \sum_{n \in \mathcal{N}_t} \hat{h}_p(n) / \max(I_p - n + 1, 1)$ and $Flex(C_t) = |\mathcal{P}|^{-1} \sum_{p \in \mathcal{P}} Flex_p(C_t)$; both equal one when $\mathcal{N}_t$ is empty. Final placement feasibility is still verified by hop-by-hop propagation in Algorithm 1. A route is capacity-feasible only when $Cap_R(n) = \mathbf{1}[\min_{p \in \mathcal{P}(R)} \hat{h}_p(n) > 0] = 1$; otherwise, it is filtered before cost ranking.

3.4 Feasibility Constraints

Following the standard TAS/GCL non-overlap and precedence model [6,7,13], the payload serialization length is $n_{i,h}^{tx} = \lceil 8 l_i / (B_{e_{i,h}} \tau) \rceil$ slots and the reserved length is $n_{i,h} = n_{i,h}^{tx} + g_{i,h}$, where $B_{e_{i,h}}$ is in bit/s and $g_{i,h}$ is the configured protocol and guard-band allowance. On a bridge-adjacent hop, $g_{i,h}$ includes at least $g_{sync} = \lceil \epsilon_{sync} / \tau \rceil$ slots. For instance $q = 0, \ldots, \mathcal{H}/\pi_i - 1$, the hop-$h$ window is $w_{i,h}^{(q)} = [s_{i,h} + q\pi_i, s_{i,h} + q\pi_i + n_{i,h})$,

with $0 \le s_{i,h} < \pi_i$ and $s_{i,h} + q\pi_i + n_{i,h} \le \mathcal{H}$. Let $e_{i,h}$ denote the egress port used by flow f_i at hop h. Non-overlap requires that any two TT frame instances reserved on the same egress port do not intersect:

$$w_{i,h}^{(q)} \cap w_{j,h'}^{(q')} = \varnothing, \quad \forall (i,h,q) \neq (j,h',q') : e_{i,h} = e_{j,h'}. \tag{3}$$

Causality between adjacent hops is written as

$$s_{i,h+1} \ge s_{i,h} + n_{i,h} + \delta_{i,h}, \tag{4}$$

where $d_{i,h}^{link}$, $d_{i,h}^{proc}$, and $d_{i,h}^{5GS}$ are propagation, processing, and bridge delays in seconds, and $\delta_{i,h} = \lceil (d_{i,h}^{link} + d_{i,h}^{proc} + d_{i,h}^{5GS})/\tau \rceil$ is in slots. If hop h traverses the 5GS logical bridge, $\chi_{i,h} = (e_{i,h}^{in}, e_{i,h}^{out})$ denotes its ingress–egress pair and $d_{i,h}^{5GS} = \Delta_t^{pp}(\chi_{i,h}, c_i)$; otherwise, $d_{i,h}^{5GS} = 0$. The first hop cannot precede release, $s_{i,1} \ge \rho_i$. With H_i hops, completion before the instance-relative deadline requires

$$s_{i,H_i} + n_{i,H_i} \le \rho_i + d_i. \tag{5}$$

Candidate routes must also satisfy $q_t(f_i, e) = a_t(e) = 1$ on every 5GS-related hop.

3.5 Problem Definition

Given C_t, $\mathcal{E}_t$, and a newly arrived TT flow f_{new}, the foreground problem seeks $(R_{new}, \Phi_{new}, W_{new})$ and generates incremental GCL insertion entries $\Delta\mathcal{G}^{ins}$. Background repair is invoked by the composite rule in Section 4.3, which aggregates fragmentation, reduced window flexibility, recent insertion failures, and reported 5GS risk. The background problem then selects a bounded subset $\mathcal{F}_{sel} \subseteq \mathcal{F}^{act}(t)$ and computes $\widetilde{\Omega}_i = (\widetilde{R}_i, \widetilde{\Phi}_i, \widetilde{W}_i)$ for $f_i \in \mathcal{F}_{sel}$, which generates GCL modification entries $\Delta\mathcal{G}^{mod}$.

The main objective is to maximize long-term admission success rate:

$$\eta(T_{obs}) = \frac{|\mathcal{F}^{adm}(T_{obs})|}{|\mathcal{F}^{arr}(T_{obs})|}, \quad \max \eta(T_{obs}), \tag{6}$$

where $\mathcal{F}^{arr}(T_{obs})$ is the set of TT flows that arrived during $[0, T_{obs}]$, and $\mathcal{F}^{adm}(T_{obs}) \subseteq \mathcal{F}^{arr}(T_{obs})$ is the subset successfully admitted by time T_{obs}.

Proposition 1 (computational hardness). The joint route/window feasibility subproblem underlying dynamic admission is NP-hard even with one egress port, fixed one-hop routes, one instance per hyperperiod, and static 5GS state. *Proof sketch.* Reduce classical NP-complete non-preemptive single-machine feasibility with release times and deadlines [20]. Map the machine to the port and each job to a one-hop flow whose release offset, window length, and deadline encode the corresponding job data. GCL non-overlap, release, and deadline constraints exactly enforce machine capacity and timing. This polynomial mapping proves NP-hardness of the restricted, hence general, problem.

The objective is subject to non-overlap, causality, deadline, 5GS-state feasibility, and disturbance constraints. For a triggered background repair, let $\mathcal{F}^{elig}(t) \subseteq \mathcal{F}^{act}(t)$ denote admitted flows that intersect bottleneck resources or 5GS delay-risk records. The disturbance budget is defined as

$$|\mathcal{F}_{sel}| \le N_s(t),$$
$$N_s(t) = \begin{cases} 0, & \mathcal{F}^{elig}(t) = \varnothing, \\ \min\{|\mathcal{F}^{elig}(t)|, N_\Gamma(t)\}, & \text{otherwise}, \end{cases} \tag{7}$$

where $N_\Gamma(t) = \max(1, \lfloor \Gamma |\mathcal{F}^{act}(t)| \rfloor)$ and dimensionless $0 < \Gamma \leq 1$ controls the maximum fraction of active schedules that can be adjusted. Consequently, one repair invocation can alter at most $N_s(t)$ flow schedules; all others remain fixed. GCL insertion or modification entries are activated at a common cycle boundary after controller-side validation, following the configured deployment procedure.

For later use, the controller maintains a candidate-route pool $\mathcal{K}_{route}(s,d) = \{R_1, \ldots, R_K\}$ for each source-destination pair. The pool may be generated by a standard K-shortest loopless-path routine or maintained at runtime. Departures create released-window records $\mathcal{W}^{rel} = \{(p, g, \Delta)\}$, where (p, g, Δ) denotes an idle interval of length Δ released at egress port p from start slot g. Rejected insertions create failure records $\mathcal{R}^{fail} = \{(R, p_b, c_b)\}$, where R is the failed route, p_b is the bottleneck port, and c_b is the violated constraint type.

The configuration aggregates are $FI(C) = |\mathcal{P}|^{-1} \sum_p FI_p$ and $Load(C) = |\mathcal{P}|^{-1} \sum_p (1 - I_p/\mathcal{H})$. The recent-failure term is the rejected fraction among the latest L_f arrivals. For the 5GS-related hop set $\mathcal{L}_{5GS}$, $Risk_i^{5GS}$ is zero if $R_i \cap \mathcal{L}_{5GS} = \varnothing$; otherwise it is the maximum over those hops of $1 - q_t(f_i, e)$, $1 - a_t(e)$, and $\min\{1, \Delta_t^{pp}(\chi_{i,e}, c_i)/D_i\}$. Thus, $\overline{FI}_t = FI(C_t)$, $\overline{Flex}_t = Flex(C_t)$, $\overline{Fail}_t$ is the recent-failure fraction, and $\overline{Risk}_t^{5GS}$ is the active-flow mean. $Cost_i$ is the decision-normalized number of GCL entries affected by changing f_i; $Risk^{5GS}(X)$ is the corresponding mean in candidate X; and $Change(X)$ is the decision-normalized sum of modified-flow, route-change, hop-offset-change, and GCL-entry-change counts. Ratios already in $[0,1]$ are used directly. Any other scalar z is normalized over candidates in the same decision as $\tilde{z} = (z - z_{\min})/(z_{\max} - z_{\min})$, or zero if the range vanishes. All weights are dimensionless.

4 Two-Timescale Incremental Routing and Scheduling

The proposed mechanism has a fast foreground path for newly arrived flows and a bounded background path for repairing the scheduling state.

4.1 Overview

Fig. 3 shows the proposed two-timescale mechanism. The foreground lane receives a new TT-flow arrival, reads the current scheduling configuration and 5GS information snapshot, and returns insertion entries or a rejection record. The background lane uses fragmentation, flexibility, insertion-failure, and 5GS-risk diagnostics to identify bottleneck resources and repairs only a selected subset of admitted flows. Both lanes update controller-side state through incremental GCL entries.

The mechanism separates latency-critical admission from capacity restoration. Algorithm 1 does not relocate admitted flows. Algorithm 2 broadens search only within $\mathcal{F}_{sel}$, processing high-priority bottleneck, released-window, or 5GS-risk flows first. Sequential candidate-state updates avoid joint enumeration across all selected routes and windows, so Γ bounds both reconfiguration scope and background work.

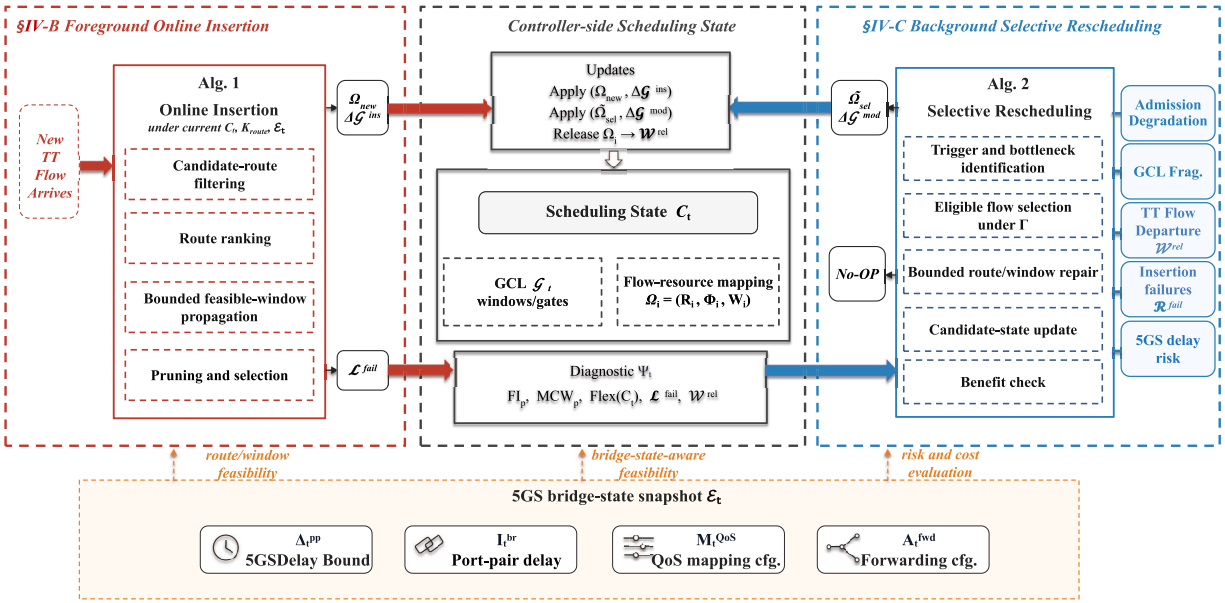

Figure 3: Two-timescale online admission and selective repair framework.

4.2 Algorithm 1: 5G-Aware Foreground Online Insertion for TT-Flow Admission

Algorithm 1 performs fast 5G-aware foreground insertion for a newly arrived TT flow. It first builds or reads a candidate-route pool, then filters routes that violate loop-freedom, forwarding availability, QoS mapping, capacity feasibility, or a coarse delay lower bound. For route ranking, n_{new}^{max} denotes the conservative maximum required hop-window length of the new flow on a candidate route. The remaining routes are ranked by a lightweight cost:

$$J_R(R) = \alpha_1 d_R^{link} + \alpha_2 d_R^{5GS} + \alpha_3 U_R + \alpha_4 FI_R - \alpha_5 h_R(n_{new}^{max}), \tag{8}$$

where d_R^{link} and d_R^{5GS} are wired and reported 5GS delay in seconds, U_R is the average occupied-slot ratio over $\mathcal{P}(R)$, FI_R is the average FI_p over nonempty route ports, and $h_R(n_{new}^{max})$ is the bottleneck flexibility estimate in possible start slots. The normalization rule defined above is applied before weighting. If no port on R has idle slots, the route is filtered before ranking.

For a partial placement π during hop-by-hop propagation, the foreground placement cost is

$$J_{fg}(\pi) = \rho_1 L(\pi) + \rho_2 FI(\pi) - \rho_3 S(\pi), \tag{9}$$

where $L(\pi)$ is accumulated latency in slots, $FI(\pi)$ is the dimensionless average fragmentation score of used ports, and $S(\pi)$ is residual deadline slack in slots. These terms use the same decision-local normalization.

The scheduling step uses bounded feasible-window propagation. At each hop, the algorithm enumerates feasible windows from the GCL idle-window set and propagates only the best partial placements. If a complete placement is found, the controller generates $\Delta\mathcal{G}^{ins}$; otherwise, it records the failure information for the background lane. With K candidate routes, at most H_{max} hops, propagation width M, and per-placement window cap W, the bounded search costs $O(KH_{max}MW\log(MW))$ when partial placements are ranked at each hop. With a linear top-M selection implementation, the ranking step can be reduced to $O(KH_{max}MW)$.

Algorithm 1: 5G-aware foreground online insertion for TT-Flow admission

Input: f_{new}, C_t, $\mathcal{E}_t$, $\mathcal{K}_{route}$, M, W;
Output: $\Delta\mathcal{G}^{ins}$ or REJECT;
Compute required hop-window lengths and n_{new}^{max};
Filter loop-free candidate routes by M_t^{QoS}, A_t^{fwd}, capacity feasibility, and delay lower bound;
Rank remaining routes by $J_R(R)$ in (8);
for each ranked route R **do**;
  Initialize partial placements at the first hop;
  for each next hop on R **do**;
    Enumerate the first W feasible windows sorted by earliest start time and local placement cost;
    Enforce non-overlap, causality, deadline, and 5GS-state feasibility;
    Keep at most M partial placements with the lowest J_{fg};
  end for;
  if a complete placement exists **then**;
    Generate $\Delta\mathcal{G}^{ins}$ and **return** success;
  end if;
end for;
Record failed routes, bottleneck ports, and failed constraints;
return REJECT;

4.3 Algorithm 2: Fragmentation-Aware Background Selective Rescheduling for GCL Repair

Algorithm 2 triggers when $\Psi_t = \omega_1\overline{FI}_t + \omega_2(1-\overline{Flex}_t) + \omega_3\overline{Fail}_t + \omega_4\overline{Risk}_t^{5GS}$ exceeds θ_H. The four terms measure fragmentation, flexibility, recent failures, and 5GS risk and are evaluated at the same decision epoch. Thus, simultaneous warnings accumulate in one score and cause at most one invocation per epoch; the disturbance budget $N_s(t)$ does not grow with the number of active warnings. After triggering, these signals guide bottleneck identification and Sel_i, with hard 5GS violations assigned the maximum $Risk_i^{5GS}$. $\mathcal{B}_t$ contains ports with $FI_p \geq \theta_{FI}$ or $Flex_p \leq \theta_F$, ports in $\mathcal{R}^{fail}$, and resources mapped from risky 5GS pairs. For an eligible flow, $B_i = |\mathcal{P}(R_i) \cap \mathcal{B}_t|/|\mathcal{P}(R_i)|$; Rel_i is the fraction of route ports with a released window long enough for n_i^{max}. $Risk_i^{5GS}$ includes hard violations, while $Cost_i$ estimates affected GCL entries.

$Sel_i = \beta_1 B_i + \beta_2 Rel_i + \beta_3 Risk_i^{5GS} - \beta_4 Cost_i$ ranks expected repair value per disturbance. In descending order, bounded coordinate repair temporarily removes each mapping, re-evaluates its route and windows using Algorithm 1's propagation, and restores the original if no feasible candidate exists. This rollback-safe sequence isolates each change and avoids simultaneous route/window enumeration across $\mathcal{F}_{sel}$.

With I_{bg} passes, K_{rep} routes, at most H_{max} hops, width M, and window cap W, background cost is $O(I_{bg}N_s(t)K_{rep}H_{max}MW\log(MW))$. Since $N_s(t)$ is at most $\Gamma|\mathcal{F}^{act}(t)|$ up to rounding, the disturbance budget also bounds controller computation and coordination. Background repair uses a versioned snapshot while foreground insertion continues; before activation, the controller checks the base version and revalidates the aggregate delta against the latest state. A stale or conflicting delta is discarded, whereas a valid delta is activated atomically at a common cycle boundary.

The background objective combines feasibility, flexibility, fragmentation, load balance, 5GS delay risk, and adjustment cost:

$$\begin{aligned} J_{bg}(X) = {} & \lambda_1 V(X) + \lambda_2 (1 - Flex(C_X)) \\ & + \lambda_3 FI(C_X) + \lambda_4 Load(C_X) \\ & + \lambda_5 Risk^{5GS}(X) + \lambda_6 Change(X), \end{aligned} \tag{10}$$

where $V(X) = 0$ for feasible candidates and $V(X) > 0$ otherwise. A result is accepted only when $V(X) = 0$ and $J_{bg}(X)$ improves by at least dimensionless threshold ϵ. $Change(X)$ includes modified flows, route changes, offset shifts, and changed GCL entries. Thus, the algorithm can adjust routes and windows inside $\mathcal{F}_{sel}$ while keeping the disturbance scale controlled by (7).

Proposition 2 (feasibility preservation and bounded disturbance). If C_t is feasible under the current $\mathcal{E}_t$ and Algorithm 2 returns $\Delta\mathcal{G}^{mod}$, the resulting state is feasible and differs from C_t in at most $N_s(t)$ admitted-flow schedules. *Proof.* Flows outside $\mathcal{F}_{sel}$ remain fixed. Each unsuccessful replacement restores Ω_i^{old}, each candidate is checked against all feasibility constraints, and the final update is accepted only when $V(X) = 0$. Eq. (7) gives $|\mathcal{F}_{sel}| \le N_s(t)$.

Algorithm 2: Fragmentation-aware background selective rescheduling for GCL repair

Input: C_t, $\mathcal{F}^{act}(t)$, $\mathcal{W}^{rel}$, $\mathcal{R}^{fail}$, $\mathcal{E}_t$, Γ;
Output: $\Delta\mathcal{G}^{mod}$ or No-Op;
1: Compute diagnostics and trigger score Ψ_t;
2: **if** $\Psi_t < \theta_H$ **then return** No-Op;
3: Build $\mathcal{B}_t$, $\mathcal{F}^{elig}$, and $Sel_i = \beta_1 B_i + \beta_2 Rel_i + \beta_3 Risk_i^{5GS} - \beta_4 Cost_i$;
4: Select $\mathcal{F}_{sel} = Top^{Sel}_{N_s(t)}(\mathcal{F}^{elig})$ using (7);
5: **if** $N_s(t) = 0$ **then return** No-Op;
6: Initialize candidate state $X \leftarrow C_t$ and mark $\mathcal{F}^{act}(t) \backslash \mathcal{F}_{sel}$ as fixed;
7: **for** at most I_{bg} repair passes **do**;
8: **for** each $f_i \in \mathcal{F}_{sel}$ in descending Sel_i **do**;
9: Store and temporarily remove Ω_i^{old};
10: Rerun bounded route/window propagation under fixed unselected flows;
11: Retain the lowest-cost feasible candidate in X, otherwise restore Ω_i^{old};
12: **end for**;
13: **end for**;
14: Revalidate X against the latest state version; **if** conflicting **then return** No-Op;
15: **if** X is feasible and $J_{bg}(X) < J_{bg}(C_t) - \epsilon$ **then**;
16: Generate $\Delta\mathcal{G}^{mod}$ from the difference between X and C_t;
17: **return** $\Delta\mathcal{G}^{mod}$;
18: **else**;
19: **return** No-Op;
20: **end if**;

5 Performance Evaluation

The evaluation covers admission, decision cost, configuration disturbance, exact-instance quality, arrival-order dependence, multi-slot fragmentation, and dynamic controller-visible 5GS input.

5.1 Experimental Setup

A380, CEV, and Ring6 contain $(9, 8)$, $(9, 9)$, and $(6, 6)$ bridges/end systems, with one 5GS logical bridge between TSN domains. The event-driven simulator enforces Section 3 constraints, departures, snapshot changes, and common-boundary GCL activation. Methods share C_t, lifecycle events, route pools, and ordered $\{\mathcal{E}_t\}$.

Source and destination nodes are sampled uniformly from distinct end systems. We set $r_i = 0$, $P_i \in \{2, 4, 8, 16\}$ ms, $D_i/P_i \in \{0.5, 0.75, 1\}$, and $l_i \in [64, 1500]$ bytes. Flow lifecycles follow Poisson arrivals and exponential holding times with mean 0.5 s. Offered load is the maximum aggregate-serialization-rate/link-rate ratio on shortest reference routes. It is computed before admission, so rejection cannot lower the assigned offered-load level.

The main setting uses $B_e = 1$ Gbit/s, $\tau = 5$ µs, and $\epsilon_{sync} = 1$ µs. Consecutive 10,000-sample blocks from public COTS-5G uplink/downlink one-way-delay traces [21] provide time-ordered empirical 99.9th-percentile bridge-delay envelopes. These envelopes are empirical rather than standardized worst-case guarantees. Controlled QoS/availability transitions supplement the traces, and every update revalidates affected admitted routes using the Section 3 recourse rule. Table 1 summarizes the experimental configuration, workload, algorithm parameters, exact-benchmark protocol, and execution platform.

Table 1: Experimental settings and reproducibility controls.

Item	Setting
Topology	A380, CEV, Ring6; one 5GS logical bridge
Time/link	$B_e = 1$ Gbit/s; $\tau = 5$ µs; $\mathcal{H} = 3200$ slots (16 ms); $\epsilon_{sync} = 1$ µs
Workload	$P_i \in \{2, 4, 8, 16\}$ ms; $D_i/P_i \in \{0.5, 0.75, 1\}$; $l_i = 64$–1500 bytes; offered load 0.35–0.95
Dynamic input	1000 events/trace; $\mathcal{E}_t$ every 25 events; ordered COTS-5G delay blocks and controlled QoS/availability changes
Replication	30 shared-seed traces/point; paired 95% bootstrap confidence intervals
Route/window search	$\lvert\mathcal{K}_{route}\rvert = 5$ Yen routes [22]; $M = 10$; candidate-window cap $W = 20$
Background lane	$\Gamma = 15\%$; $I_{bg} = 2$; $K_{rep} = 3$; $L_f = 20$; $\theta_H = 0.55$; $\theta_{FI} = 0.50$; $\theta_F = 0.30$; $\epsilon = 0.01$
Weights	$\alpha_{1:5} = 0.2$; $\rho_{1:3} = 1/3$; $\omega_{1:4} = 0.25$; $\beta_{1:4} = 0.25$; $\lambda_1 = 1$, $\lambda_{2:6} = 0.2$
Exact benchmark	Ring6; 10–40 active flows; all loop-free routes; same Γ and no-eviction rule; CP-SAT, 300 s limit, one worker
Platform	Python 3.12.13 on Windows 11; Intel Core i5-14600K, 64 GB RAM; one worker/thread for all methods

5.2 Methods and Protocol

SWOTS-ASAP-WS (SWOTS) is the incremental TSN baseline [23]; all implementations share route pools, replication, and checks. 5GA-SWOTS adds hard M_t^{QoS}, A_t^{fwd}, and Δ_t^{pp} filters. Online-only uses Algorithm 1, Proposed adds Algorithm 2, and Batch-Recompute reinserts all mandatory flows and the arrival, committing only if fully feasible.

For Ring6, OR-Tools CP-SAT [24] receives all loop-free routes, mandatory flows, and the same $N_s(t)$ limit. It lexicographically maximizes new-flow admission and minimizes changed GCL entries. Only OPTIMAL runs enter the empirical gap; timeout coverage is separate. This exact bounded-instance benchmark

is not an approximation bound. Metrics are $\eta(T_{obs})$, trace-level P95 event runtime, and changed routes/GCL entries/schedules. Paired bootstrap intervals use 10,000 resamples; disjoint pilot seeds serve tuning only.

5.3 Admission and Operational Cost

In Fig. 4a–c, methods remain close at light load, whereas repair recovers windows fragmented by departures and earlier insertions at high load. At load 0.95, Proposed exceeds Online-only by 14.8, 18.5, and 16.0 points on A380, CEV, and Ring6. The corresponding gains over 5GA-SWOTS are 19.0, 21.0, and 18.5 points, while the gaps to Batch-Recompute are only 2.0, 2.2, and 1.7 points. Thus, bounded repair recovers most of the broader-rescheduling admission without rebuilding every schedule.

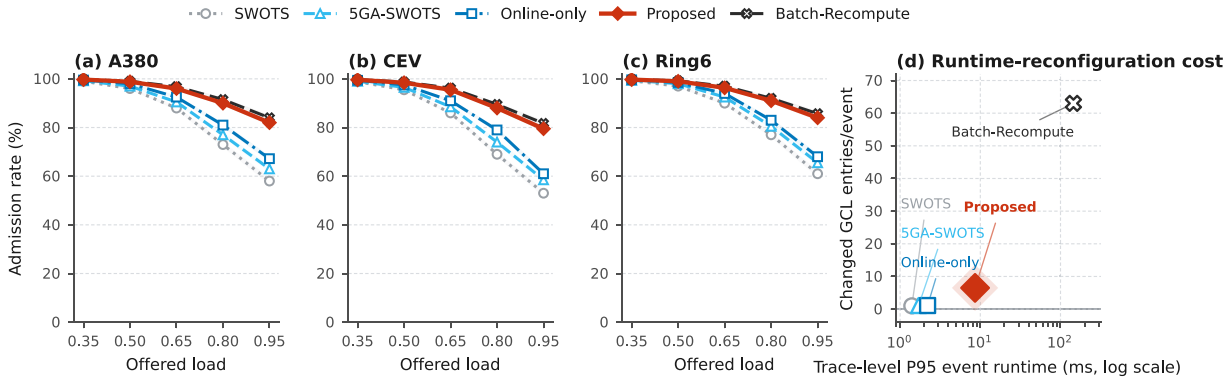

Figure 4: Overall admission and operational cost. (**a**) A380 admission vs. load; (**b**) CEV admission vs. load; (**c**) Ring6 admission vs. load; and (**d**) A380 P95 event runtime vs. changed GCL entries at load 0.95. Bands show paired 95% confidence intervals.

On A380, Fig. 4d gives Proposed a P95 runtime of 8.6 vs. 145 ms for Batch-Recompute, with 6.5 vs. 63.0 changed GCL entries per event. Faster foreground-only methods trade this cost for lower high-load admission. Proposed spends the bounded repair budget only when accumulated fragmentation warrants coordinated change.

5.4 Quality and Mechanism Validation

On Ring6, Fig. 5a gives Proposed an empirical optimality gap of 0.1%–1.8% against the exact benchmark at 10–40 active flows, vs. 9.0% for Online-only at 40; CP-SAT proves OPTIMAL for 100%, 100%, 96%, and 88% of cases, with timeouts excluded. Fig. 5b links this quality to restored feasible start positions after repair.

In Fig. 5c, the variant without 5GS guidance retains all hard delay, mapping, and availability constraints but removes $Risk^{5GS}$ from rankings and objectives. Random-selection retains $N_s(t)$ and the repair operator but randomizes the Sel_i order. Their admission changes are −4.7 and −7.2 points, while disabling repair changes it by −16.4 points. The ordering therefore directs a fixed disturbance budget toward higher-value repairs. Across ten fixed flow multisets and ten lifecycle orderings each, admission standard deviation is 1.6 points for Proposed vs. 3.9 for Online-only, showing reduced—not eliminated—arrival-order dependence.

Fig. 5d varies $\Gamma \in \{5, 10, 15, 20\}\%$ and $\theta_H \in \{0.45, 0.55, 0.65, 0.75\}$. Beyond the default $(15\%, 0.55)$, admission rises only 1.3 points while changes increase. Perturbing each normalized weight by ±25% changes admission by at most 2.3 points. Varying ϵ_{sync} from 0.5 to 2 µs changes admission by 1.8 points, with the bound always included in bridge guard bands.

Fig. 5e changes only τ among 50, 5, and 1 µs. The shares of hop payload transmissions spanning more than one slot are 0%, 66%, and 95%. This measure uses $n^{tx}_{i,h} > 1$ and therefore excludes guard-band slots.

Proposed admits 71%, 84%, and 85%, vs. Online-only's 68%, 68%, and 61%. The growing fine-grid separation confirms that repair is exercised by heterogeneous multi-slot windows. In Fig. 5f, each trigger forks identical C_t, $\mathcal{E}_t$, and next 20 arrivals into repair and no-op branches, with later repair disabled. All triggers remain in the analysis. The J_{bg} decrease has a Pearson correlation coefficient of 0.80 with additional admissions, experimentally supporting it as a near-future-capacity proxy.

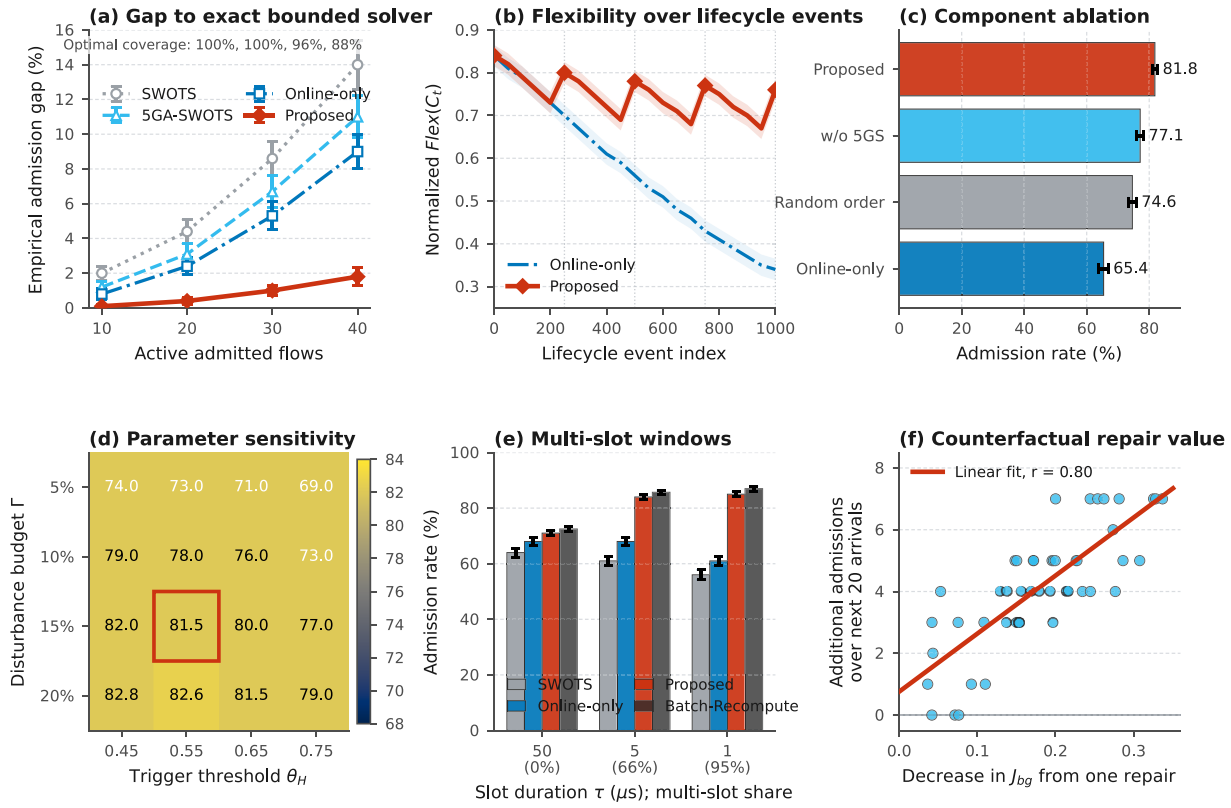

Figure 5: Solution quality and mechanism validation. (**a**) Gap to the exact Ring6 solver; (**b**) normalized feasible-start flexibility $Flex(C_t)$; (**c**) component ablations; (**d**) sensitivity to disturbance fraction Γ and trigger threshold θ_H; (**e**) slot duration τ (labels: share with $n^{tx}_{i,h} > 1$); and (**f**) next-20-arrival gain vs. the repair-induced decrease in objective J_{bg}. Error bars or bands show paired 95% confidence intervals.

Trace-driven Δ^{pp}_t increases also revalidate admitted flows. Algorithm 2 remaps 92.4% of affected schedules within budget; 0.8% of active flows require re-admission or 5GS reconfiguration, and no state violating current $\mathcal{E}_t$ is committed.

6 Conclusion

This work formulated controller-side dynamic TT-flow admission using reported 5GS bridge information and established NP-hardness of the underlying joint feasibility problem. Fast foreground insertion protects admitted schedules, while priority-ordered background repair concentrates a bounded disturbance budget on bottleneck-relevant flows. At 0.95 offered load, the mechanism improves admission by 14.8–18.5 points over online-only scheduling and remains within 1.7–2.2 points of full recomputation. On A380, its P95 runtime is 8.6 ms rather than 145 ms, with 6.5 rather than 63.0 changed GCL entries per event. These results show that controller-side 5G-aware admission can preserve most of the broader-rescheduling admission benefit with substantially lower decision cost and schedule disturbance.

Acknowledgement: None.

Funding Statement: This work was supported in part by the Shandong Provincial Natural Science Foundation under Grant Nos. ZR2023LZH011 and ZR2026QC0792, and the Taishan Scholar Program of Shandong Province in China under Grant No. TSON202312230.

Author Contributions: Zhihao Liu: conception, methods, software, experiments, analysis, drafting; Xiaolong Wang: design, supervision, revision; Yi Zhang: data; Wei Zhang: methods; Jian Wang and Huiling Shi: validation. All authors reviewed and approved the final version of the manuscript.

Availability of Data and Materials: The simulation data and source code supporting the findings of this study are available from the corresponding author upon reasonable request. The public COTS-5G delay dataset is available at https://doi.org/10.5281/zenodo.10390211.

Ethics Approval: Not applicable.

Conflicts of Interest: The authors declare no conflicts of interest.

References

1. Zhang X, Debroy S. Resource management in mobile edge computing: a comprehensive survey. ACM Comput Surv. 2023;55(13s):291.
2. ETSI GS MEC 003. Multi-access edge computing (MEC); framework and reference architecture. Sophia Antipolis, France: ETSI; 2022. [cited 2026 Aug 20]. Available from: https://www.etsi.org/deliver/etsi_gs/MEC/001_099/003/03.01.01_60/gs_MEC003v030101p.pdf.
3. Harmatos J, Maliosz M. Architecture integration of 5G networks and time-sensitive networking with edge computing for smart manufacturing. Electronics. 2021;10(24):3085. doi:10.3390/electronics10243085.
4. Rost PM, Kolding T. Performance of integrated 3GPP 5G and IEEE TSN networks. IEEE Commun Stand Mag. 2022;6(2):51–6. doi:10.1109/mcomstd.0001.2000013.
5. 802.1AS-2020. IEEE Standard for local and metropolitan area networks–timing and synchronization for time-sensitive applications. New York, NY, USA: IEEE; 2020. doi:10.1109/IEEESTD.2020.9121845.
6. 802.1Qbv-2015. IEEE standard for local and metropolitan area networks–bridges and bridged networks–amendment 25: enhancements for scheduled traffic. New York, NY, USA: IEEE; 2016. doi:10.1109/IEEESTD.2016.8613095.
7. 802.1Qcc-2018. IEEE standard for local and metropolitan area networks–bridges and bridged networks–amendment 31: stream reservation protocol enhancements and performance improvements. New York, NY, USA: IEEE; 2018. doi:10.1109/IEEESTD.2018.8514112.
8. ETSI TS 123501. 5G; System architecture for the 5G system (5GS) (3GPP TS 23.501 version 18.5.0 Release 18). Sophia Antipolis, France: ETSI; 2024 [cited 2026 Aug 20]. Available from: https://www.etsi.org/deliver/etsi_TS/123500_123599/123501/18.05.00_60/ts_123501v180500p.pdf.
9. Sasiain J, Franco D, Atutxa A, Astorga J, Jacob E. Toward the integration and convergence between 5G and TSN technologies and architectures for industrial communications: a survey. IEEE Commun Surv Tutor. 2025;27(1):259–314. doi:10.1109/comst.2024.3422613.
10. Yang J, Yu G. Traffic scheduling for 5G-TSN integrated systems. In: Proceedings of the 2022 International Symposium on Wireless Communication Systems (ISWCS); 2022 Oct 19–22; Hangzhou, China. p. 1–6.
11. Wang X, Yao H, Mai T, Guo S, Liu Y. Reinforcement learning-based particle swarm optimization for end-to-end traffic scheduling in TSN-5G networks. IEEE/ACM Trans Netw. 2023;31(6):3254–68. doi:10.1109/tnet.2023.3276363.
12. Zhuge X, Cai X, He X, Wang Z, Dang F, Xu W, et al. InNetScheduler: in-network scheduling for time- and event-triggered critical traffic in TSN. In: Proceedings of IEEE INFOCOM 2024–IEEE Conference on Computer Communications; 2024 May 20–23; Vancouver, BC, Canada. p. 421–30.

13. Wang X, Yao H, Mai T, Xiong Z, Wang F, Liu Y. Joint routing and scheduling with cyclic queuing and forwarding for time-sensitive networks. IEEE Trans Veh Technol. 2023;72(3):3793–804. doi:10.1109/tvt.2022.3216958.
14. Yang Q, Jiang X, Liu R, Li T, Yang H, Quan W, et al. A performance-balanced scheduling algorithm for diverse real-world TSN scenarios. IEEE Trans Parallel Distrib Syst. 2025;36(12):2469–81. doi:10.1109/tpds.2025.3580402.
15. Cheng Z, Yang D, Guo R, Zhang W. Joint time-frequency resource scheduling over CQF-based TSN-5G system. In: Proceedings of the 15th International Conference on Communication Software and Networks (ICCSN); 2023 Jul 21–23; Shenyang, China. p. 60–5.
16. Cai Y, Zhang X, Hu S, Wei X. Dynamic QoS mapping and adaptive semi-persistent scheduling in 5G-TSN integrated networks. China Commun. 2023;20(4):340–55. doi:10.23919/jcc.fa.2022-0548.202304.
17. Satka Z, Ashjaei M, Fotouhi H, Daneshtalab M, Sjodin M, Mubeen S. QoS-MAN: a novel QoS mapping algorithm for TSN-5G flows. In: Proceedings of the IEEE 28th International Conference on Embedded and Real-Time Computing Systems and Applications (RTCSA); 2022 Aug 23–25; Taipei, Taiwan. p. 220–7.
18. Larranaga A, Lucas-Estan MC, Martinez I, Gozalvez J. 5G configured grant scheduling for 5G-TSN integration for the support of industry 4.0. In: Proceedings of the 18th Wireless On-Demand Network Systems and Services Conference (WONS); 2023 Jan 30–Feb 1; Madonna di Campiglio, Italy. p. 72–9.
19. Wang Z, Li Z, Long C, Zheng Y, Ai B, Song X. Time synchronization for 5G and TSN integrated networking. IEEE J Sel Areas Commun. 2025;43(9):2969–80. doi:10.1109/jsac.2025.3574595.
20. Lenstra JK, Rinnooy Kan AHG, Brucker P. Complexity of machine scheduling problems. Ann Discrete Math. 1977;1:343–62. doi:10.1016/s0167-5060(08)70743-x.
21. Mostafavi SS, Sharma GP. DETERMINISTIC6G COTS 5G latency measurements (Version v1). Zenodo. 2023. doi:10.5281/zenodo.10390211.
22. Yen JY. Finding the K shortest loopless paths in a network. Manag Sci. 1971;17(11):712–6. doi:10.1287/mnsc.17.11.712.
23. Alnajim A, Salehi S, Shen CC. Incremental path-selection and scheduling for time-sensitive networks. In: Proceedings of the 2019 IEEE Global Communications Conference (GLOBECOM); 2019 Dec 9–13; Waikoloa, HI, USA. p. 1–6.
24. Perron L, Didier F, Gay S. The CP-SAT-LP solver. In: Proceedings of the 29th International Conference on Principles and Practice of Constraint Programming. Vol. 280 of Leibniz International Proceedings in Informatics; 2023 Aug 27–31; Toronto, ON, Canada. p. 3:1–2.