QUEUE-AWARE COMBINATORIAL BANDIT SCHEDULING FOR SPECULATIVE DECODING IN CLOUD–EDGE LLM SERVING

Anonymous authors
Paper under double-blind review

ABSTRACT

Speculative decoding accelerates autoregressive LLM inference by enabling a draft model to generate candidate token blocks, which the target model then verifies in parallel. Existing research largely targets draft-model quality and point-to-point cloud-edge efficiency, leaving unaddressed the contention among edge-generated verification demands for the central target model's resources under multi-user workloads. The scheduling problem that arises when candidate blocks generated asynchronously by multiple edge nodes compete for limited verification resources has become a major bottleneck in improving system efficiency. To address this problem, we formulate shared-verifier admission as a queue-aware combinatorial semi-bandit problem for speculative decoding and propose the Speculative Decoding Pipeline Scheduling (SDPS) algorithm. Each node places its locally generated draft blocks into a pending-verification queue, while the central verifier selects a set of nodes in each round based on queue pressure, verification yield, and latency constraints. Without prior knowledge of node-specific yields, SDPS provably reduces its utility loss as the scheduling horizon grows, while average pending backlog is bounded under a windowed load condition through a single control parameter. Experiments on a distributed LLM system demonstrate that SDPS effectively improves the utilization of centralized verification resources while controlling pending backlog and protecting latency SLOs under multi-user contention. Code is available at https://anonymous.4open.science/r/SDPS1-2E52/.

1 INTRODUCTION

Speculative decoding accelerates autoregressive large language model (LLM) inference by separating lightweight drafting from heavyweight verification: a small *draft* model proposes candidate tokens, and a stronger *target* model verifies them in parallel, committing the accepted prefix and a correction/bonus token while preserving the target distribution (Leviathan et al., 2023; Chen et al., 2023; Stern et al., 2018). This naturally fits cloud-edge serving, with local edge drafting and centralized verification (Kang et al., 2017; Zhou et al., 2019). However, independently generated blocks compete for limited verification capacity (Figure 1), making admission decisions central to committed-token throughput and exposure-to-completion latency.

Existing speculative-decoding studies primarily improve draft-model quality, speculative structures, token selection, batching, or draft-verify execution within a co-located or centrally managed serving engine (Miao et al., 2024; Cai et al., 2024; Li et al., 2024; Qian et al., 2024). Recent cloud-edge approaches additionally optimize pipeline organization, draft configuration, batch construction, or long-term capacity allocation (Han et al., 2026; Tran et al., 2026; Xu et al., 2026). These directions improve upstream generation efficiency or overall system configuration, but they do not directly address the joint problem of online admission among asynchronously arriving remote blocks whose committed-output yields are initially unknown and whose exposure-to-completion latency grows while they wait for limited verification channels.

Motivated by this gap, we study shared-verifier admission under asynchronous multi-user cloud-edge workloads. Given only the head-of-line blocks exposed at the central server, how should the verifier select at most $K_{\max}$ blocks at each dispatch epoch to maximize committed-token output

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

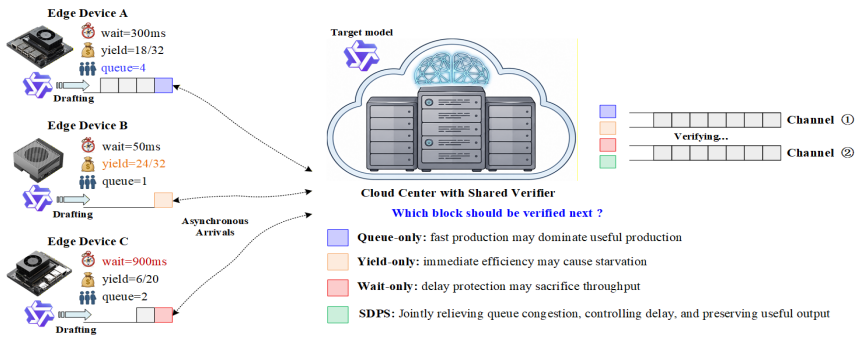


Figure 1: Shared cloud-edge verification.

while controlling pending backlog and exposure-to-completion latency? The scheduler should make this decision without predicting future uploads or request lengths, and it must learn the heterogeneous verification yields of edge nodes using feedback observed only from selected blocks.

Solving this problem presents three coupled challenges. First, *asynchronous block arrivals* from heterogeneous drafting times and network delays change the ready set and per-node backlogs; because requests resume drafting only after verification, arrivals also depend on past scheduling decisions. Second, *unknown verification yields* must be learned from selective feedback: committed-token counts are observed only for verified blocks. Third, *per-block latency constraints* require controlling exposure-to-completion delay while sustaining committed-token throughput. These challenges are interdependent: yield-only scheduling may undersample uncertain nodes and overlook backlogs; backlog-only scheduling ignores productive yield; and soft latency weights cannot protect individual blocks when outweighed by queue or yield terms.

To address these challenges, we develop **S**peculative **D**ecoding **P**ipeline **S**cheduling (SDPS), a queue-aware online policy for the $K_{\max}$ channels of a shared verifier. SDPS adopts a staged treatment of latency: waiting urgency influences the scheduling of ordinary blocks, while strict priority takes over once a block becomes latency-critical. At each dispatch epoch, SDPS first serves latency-critical blocks in earliest-deadline-first (EDF) order, then fills the remaining channels with ordinary blocks ranked by an optimistic score that combines queue backlog, waiting urgency, and a clipped upper confidence bound (UCB) on committed-token yield. Operating only on currently exposed blocks, SDPS admits as many blocks as the ready workload permits and remains non-idling at every dispatch epoch. Our contributions are summarized as follows:

- Shared-verifier admission formulation. We identify online admission to a shared verifier as a distinct systems problem in cloud-edge speculative decoding and formulate it as a queue-aware combinatorial semi-bandit over a dynamic ready set. The formulation explicitly captures asynchronous arrivals, node-level pending queues, selective committed-output feedback, and exposure-based latency requirements.
- Queue-aware online scheduling framework. We propose SDPS, a prediction-free and non-idling top-K policy that combines queue-aware optimistic scheduling for ordinary blocks with strict EDF protection for latency-critical blocks. Using a size-K_t heap, each dispatch decision can be computed in $O(|\mathcal{A}_t| \log K_t)$ time.
- Finite-horizon learning, backlog, and latency guarantees. Under conditionally unbiased bounded rewards and a windowed weighted-load condition, we establish an $O(1/V) + \widetilde{O}(\sqrt{NK_{\max}/H})$ utility-loss bound against a statewise true-mean reference and an $O(V)$ backlog bound. With a uniform load condition, horizon tuning yields a $\widetilde{O}(\sqrt{NK_{\max}H})$ cumulative statewise gap. Independently of reward statistics, bounded cycle durations yield a pathwise exposed-block lateness bound and a sufficient zero-violation condition.
- System and simulation evaluation. Testbed experiments demonstrate near-zero exposure-SLO violations (0.033%), consistent accepted-token yield gains over SLO-safe baselines, and service reallocation toward improved nodes after quality reversal. Additional experiments further validate critical-first safety, load robustness, and sliding-window adaptation.

2 Related Work

Speculative decoding and cloud–edge serving. Speculative decoding uses draft–verify execution (Leviathan et al., 2023; Chen et al., 2023; Stern et al., 2018), with extensions including tree speculation (Miao et al., 2024), multiple decoding heads (Cai et al., 2024), feature-level drafting (Li et al., 2024), optimal-transport verification (Sun et al., 2023), and drafter adaptation (Liu et al., 2024a; Zhou et al., 2024); see Xia et al. (2024) for a survey. Serving systems improve batching, memory management, and phase scheduling (Yu et al., 2022; Kwon et al., 2023; Agrawal et al., 2024; Zhong et al., 2024; Patel et al., 2024), while speculative serving further optimizes batched verification (Qian et al., 2024), speculation control (Liu et al., 2024b), acceptance-aware token selection (Wu et al., 2025), SLO-adaptive speculation (Huang et al., 2025; Li et al., 2026b), and draft–verify phase management (Chen et al., 2026). Distributed systems extend this setting to heterogeneous edge drafters sharing a central verifier (Li et al., 2025b; 2026a; Han et al., 2026; Tran et al., 2026; Xu et al., 2026). Within this setting, SLED studies edge drafting and shared verification, WISP combines speculation control with verification-time prediction and SLO-aware batching, and GoodSpeed optimizes long-term proportional-fair capacity allocation. LAPS-SD targets mean request latency (Li et al., 2025a). SDPS focuses on per-dispatch admission among asynchronously exposed FIFO heads: it learns unknown committed-token yields while combining queue-aware selection with strict EDF protection for latency-critical blocks.

Learning-augmented queueing and deadline scheduling. SDPS builds on combinatorial semi-bandits (Auer et al., 2002; Chen et al., 2013; Kveton et al., 2015; Gai et al., 2012), MaxWeight scheduling (Tassiulas & Ephremides, 1992), and Lyapunov drift-plus-penalty optimization (Neely, 2010; Georgiadis et al., 2006). Queueing-bandit methods already combine online estimation with MaxWeight-type scheduling under unknown service statistics (Krishnasamy et al., 2021; Neely et al., 2012; Stahlbuhk et al., 2021; Yang et al., 2023). Our focus is therefore not this combination alone, but its interaction with mandatory EDF service (Liu & Layland, 1973) in an availability-constrained top-K admission problem. SDPS and its statewise true-mean reference select identical critical heads, confining their learning comparison to the remaining ordinary slots. Persistent one-head exposure separately supports a pathwise exposure-to-completion bound. The resulting guarantees concern statewise learning loss, backlog under a weighted-drift condition, and exposed-block latency, rather than global scheduling optimality.

3 System Model and Problem Formulation

System Architecture and End-to-End Workflow. We consider the cloud-edge collaborative speculative decoding system depicted in Figures 1 and 2. The system consists of N edge nodes, indexed by $\mathcal{N} = \{1, \ldots, N\}$, and a central server hosting the target LLM. Each edge node runs a lightweight draft model and concurrently serves multiple interactive inference requests. For every request, the system repeatedly executes a closed-loop cycle of drafting, uploading, verification, and continued generation: a request regains drafting eligibility once the verification of its previous speculative block has completed, the result has been returned, and the request has not yet terminated. For such an eligible request, the hosting edge node first invokes its local draft model to produce a *speculative block* consisting of L draft tokens, and uploads this block, together with the request context required for target-model verification, to the central server. At each dispatch epoch, the central server admits at most $K_{\max}$ exposed blocks into the current verification group, where $K_{\max}$ represents the verifier capacity exposed to the scheduler. Because edge nodes differ in computational capability, draft-model quality, request context, and network conditions, the blocks they generate arrive at the central server asynchronously and carry heterogeneous, initially unknown verification rewards.

Once a request's speculative block has been uploaded, the hosting node need not remain idle while awaiting the verification result; it may instead draft for another eligible request on the same node. A given request, however, cannot generate its next speculative block until the current one has been verified and the result returned. Drafting is therefore serial within each request, yet pipelined across the multiple requests hosted on the same edge node.

At the central server, each edge node has a first-in-first-out (FIFO) pending-verification queue buffering blocks uploaded by its requests. In each dispatch round, only each queue's head is visible to the scheduler, which assigns at most $K_{\max}$ heads to verification channels; unselected blocks remain

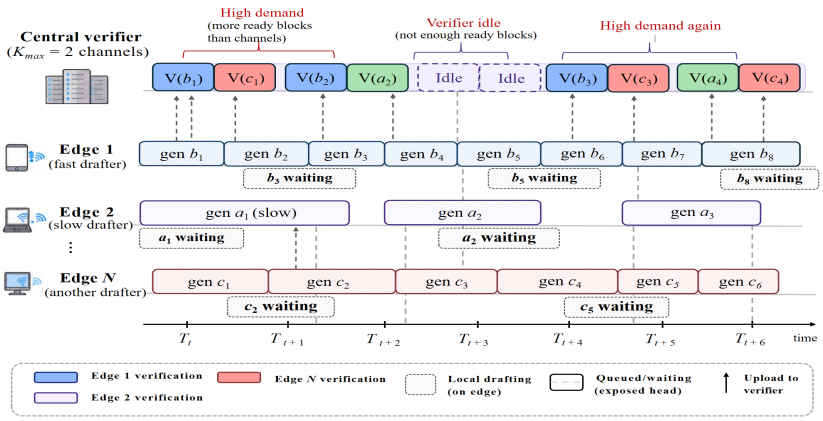


Figure 2: Asynchronous draft–queue–verify workflow.

queued. The selected blocks are verified in parallel by the target model. For each block, the target model accepts the longest valid draft prefix and commits one target-generated correction or bonus token, unless an accepted EOS terminates the request. The server returns the result to the corresponding edge node, where the request either terminates or resumes the closed-loop draft–upload–verify process. Verification scheduling determines not only the service order of pending blocks but also the availability of future drafts. Since a request resumes drafting only after its preceding block is verified and the result returned, subsequent uploads depend on past scheduling decisions. Block arrivals are therefore endogenous to the scheduling trajectory rather than exogenous.

Pending-Verification Queues and Head-of-Line Exposure. The server maintains one FIFO pending-verification queue for each edge node. Only the head of each nonempty queue is exposed to the scheduler; all other blocks remain hidden until their predecessors complete. Consequently, each node contributes at most one scheduling candidate and occupies at most one verification channel in a dispatch round. For the j-th block uploaded by node n, let $U_{n,j}$ and $C_{n,j}$ denote its upload and verification-completion times. Its exposure time is

$$R_{n,j} = \begin{cases} U_{n,j}, & \text{if it enters an empty queue,} \\ C_{n,j-1}, & \text{otherwise.} \end{cases} \tag{1}$$

An admitted head remains in its queue until verification completes. It is then removed, and its FIFO successor, if any, becomes exposed immediately.

Event-Driven Dispatch and Queue Dynamics. Time is indexed by event-driven dispatch rounds $t = 1, 2, \ldots$, where T_t denotes the starting time of round t. A new dispatch round is triggered either when the verification group launched in the preceding round has completed and exposed work remains, or when a new block arrives while the server is idle.

Let $Q_n(t)$ denote the number of blocks originating from node n that are either waiting for or currently under verification at time T_t. The set of nodes with an exposed block is therefore $\mathcal{A}_t = \{n \in \mathcal{N} : Q_n(t) > 0\}$. At round t, the scheduler selects distinct nodes $S(t) \subseteq \mathcal{A}_t$, where selecting node n dispatches its exposed head-of-line block; let $S_n(t) = \mathbf{1}\{n \in S(t)\}$. Because at most $K_{\max}$ blocks can be verified concurrently and each node exposes one block, a non-idling scheduler admits $|S(t)| = K_t \triangleq \min\{K_{\max}, |\mathcal{A}_t|\}$. The selected blocks form the current verification group, while unselected heads remain waiting and non-head blocks remain hidden in their FIFO queues.

Let $A_n(t)$ denote the number of blocks uploaded by node n during $(T_t, T_{t+1}]$, and assume the pathwise bound $0 \le A_n(t) \le A_{\max} < \infty$. This covers two regimes: in the *closed-loop* regime, node n serves at most $J_{\max}$ requests and each request has at most one outstanding block, so one may take $A_{\max} = J_{\max}$ and $Q_n(t) \le J_{\max}$; in the *open-arrival* regime, $J_{\max}$ is not enforced and $Q_n(t)$ is not structurally bounded. The analysis uses a generic $A_{\max}$ and applies to both. After the selected blocks complete verification, one queue unit is removed for each admitted node, while newly uploaded blocks are appended to the corresponding queues. The queue dynamics are therefore

$$Q_n(t+1) = [Q_n(t) - S_n(t)]^+ + A_n(t). \tag{2}$$

Exposure-Based Latency Accounting. The per-block latency requirement is measured from the time at which a block becomes exposed to the scheduler, rather than from its upload time. This distinction separates the delay directly controlled by the scheduling policy from the time a block spends hidden behind preceding blocks in the same node-level FIFO queue. Let R_n be the exposure time of node n's current head. Its waiting time at round t is $D_n(t) = T_t - R_n$, $n \in \mathcal{A}_t$. For $n \notin \mathcal{A}_t$, set $D_n(t) = 0$. For a completed block (n, j), its exposure-to-completion latency is $C_{n,j} - R_{n,j}$. Given a common per-block latency SLO D_{ref}, its lateness is $[C_{n,j} - R_{n,j} - D_{\mathrm{ref}}]^+$. Let $T_{\mathrm{cyc}}(t)$ denote the duration of a complete dispatch cycle, including scheduling, launch, and target-model verification. For the latency analysis, we assume the following uniform bounds on active dispatch cycles at fixed block length L:

$$0 < T_{\min} \le T_{\mathrm{ver}}(t) \le T_{\mathrm{cyc}}(t) \le T_{\mathrm{cyc}}^{\max} < D_{\mathrm{ref}}/2. \tag{3}$$

Accordingly, we define the critical waiting threshold as

$$D_{\mathrm{crit}} = D_{\mathrm{ref}} - 2T_{\mathrm{cyc}}^{\max} > 0. \tag{4}$$

The two-cycle margin accounts for one possibly ongoing verification cycle and one additional cycle required to verify the block itself. Therefore, once an exposed head-of-line block reaches D_{crit}, deferring it for another dispatch may cause it to violate the latency SLO. This observation motivates the strict priority assigned to latency-critical blocks in the proposed scheduling algorithm.

Reward and Scheduling Objective. An admitted block commits $Y_n(t) \in \{1, \ldots, L+1\}$ tokens: the accepted draft prefix and a target correction/bonus token, unless accepted EOS terminates the request. Define normalized feedback $X_n(t) = Y_n(t)/(L+1)$ and mean yield $\mu_n = \mathbb{E}[X_n(t)]$. This is a committed-output mean, not a per-token acceptance probability; independence across draft positions is not assumed. Round utility is $U(t) = \sum_{n \in S(t)} X_n(t)$. For a causal policy π, let $\overline{U}_H^{\pi} = H^{-1} \sum_{t=1}^{H} \mathbb{E}[U^{\pi}(t)]$. At common L, this measures expected committed output per dispatch, which we seek to improve while controlling backlog and exposure latency.

4 SDPS Algorithm

Overview and design rationale. SDPS separates hard latency protection from soft efficiency optimization. At each dispatch epoch, it first estimates the unknown verification yield of every ready node using an optimistic confidence index. Ordinary heads are then ranked by a score combining queue pressure, waiting urgency, and estimated yield. Once a head becomes latency-critical, however, it is promoted to a strict EDF class that cannot be displaced by any ordinary score. SDPS serves the critical class first, fills the remaining channels with the highest-scoring ordinary heads, and updates its yield estimates from the resulting semi-bandit feedback. This staged design prevents a large queue or yield term from overwhelming an urgent block while retaining yield-aware allocation whenever sufficient latency slack remains. Algorithm 1 gives the complete online procedure.

Optimistic Yield Estimation. Let $H_n(t) = \sum_{\tau < t} S_n(\tau)$ be the number of observations collected from node n before round t, and let $\widehat{\mu}_n(t)$ be their empirical mean. SDPS assigns an unseen node the optimistic value $w_n(t) = 1$; otherwise,

$$w_n(t) = \min\left\{\widehat{\mu}_n(t) + \sqrt{\beta \log(t+1)/(2H_n(t))}, 1\right\}, \ \beta > 3. \tag{5}$$

The confidence bonus encourages exploration, while clipping preserves optimism on the concentration event and keeps the quality term bounded.

Critical-First Admission. For each ready node, SDPS computes the ordinary-head score

$$p_n(t) = \alpha_Q Q_n(t)/Q_{\mathrm{ref}} + \alpha_W D_n(t)/D_{\mathrm{ref}} + V w_n(t), \tag{6}$$

where $\alpha_Q, \alpha_W, V > 0$ and $Q_{\mathrm{ref}} > 0$. The three terms represent queue pressure, waiting urgency, and optimistic verification yield; V controls the emphasis on yield.

Latency protection, by contrast, is enforced through a separate hard-priority mechanism rather than an arbitrarily large soft weight. Define the criticality indicator as $z_n(t) = \mathbf{1}\{D_n(t) \ge D_{\mathrm{crit}}\}$. Nodes are ranked lexicographically by $(z_n(t), z_n(t)D_n(t) + (1 - z_n(t))p_n(t))$. Critical heads precede

Algorithm 1: SDPS

Input: $K_{\max}$, D_{ref}, $T_{\text{cyc}}^{\max}$, α_Q, α_W, V, β, Q_{ref}
$D_{\text{crit}} \leftarrow D_{\text{ref}} - 2T_{\text{cyc}}^{\max}$
$H_n \leftarrow 0$, $\widehat{\mu}_n \leftarrow 0$, $\forall n \in \mathcal{N}$
for *dispatch round* $t = 1, 2, \ldots$ **do**
  Observe $\mathcal{A}_t$ and $\{Q_n(t), R_n\}_{n \in \mathcal{A}_t}$
  foreach $n \in \mathcal{A}_t$ **do**
    $D_n(t) \leftarrow T_t - R_n$
    $w_n(t) \leftarrow 1$ if $H_n = 0$; otherwise use (5)
    Compute $p_n(t)$ by (6)
  end
  Sort $\mathcal{C}_t = \{n \in \mathcal{A}_t : D_n(t) \geq D_{\text{crit}}\}$ by increasing (R_n, n)
  Sort $\mathcal{O}_t = \mathcal{A}_t \setminus \mathcal{C}_t$ by decreasing $p_n(t)$, ties by n
  Select $S(t)$ by (7) and dispatch its heads
  Upon completion, observe $\{X_n(t)\}_{n \in S(t)}$ and update $H_n, \widehat{\mu}_n$
  Update queues and exposure states using Eqs. (2) and (1)
end

ordinary heads, ordered by increasing (R_n, n) (EDF); ordinary heads follow decreasing $p_n(t)$, with node-index tie-breaking.

Top-K Dispatch and Online Update. Let $\mathcal{C}_t$ be the critical nodes sorted by increasing (R_n, n), and let $\mathcal{O}_t = \mathcal{A}_t \setminus \mathcal{C}_t$ be the ordinary nodes sorted by decreasing $p_n(t)$ with ascending node-index tie-breaking. SDPS selects

$$S(t) = [\mathcal{C}_t; \mathcal{O}_t]_{1:K_t},\ K_t = \min\{K_{\max}, |\mathcal{A}_t|\}, \tag{7}$$

Here $[\cdot]_{1:K_t}$ selects the first K_t nodes of the concatenated ordering, making admission non-idling.

Only admitted nodes update their reward statistics. Algorithm 1 presents the rule using full sorting, which costs $O(|\mathcal{A}_t| \log |\mathcal{A}_t|)$. An equivalent implementation maintains a size-K_t heap over all ready nodes using the lexicographic key, reducing selection to $O(|\mathcal{A}_t| \log(K_t + 1))$.

5 Theoretical Analysis

Statewise Reference and Analytical Conditions. The analysis assumes *(P1)* bounded uploads, $0 \leq A_n(t) \leq A_{\max}$; *(P2)* one queue unit removed per admission; and *(P3)* at most one exposed head per node, remaining exposed until served. Let $\mathcal{F}_t$ be the round-t pre-decision history. To isolate learning loss, the statewise reference $\pi_V^\star$ uses SDPS's states and non-idling critical-first rule, replacing $w_n(t)$ with the true mean μ_n. Its action is $S_V^\star(t)$, with finite-horizon utility

$$U_{H,V}^\star \triangleq (1/H) \sum_{t=1}^{H} \mathbb{E}\left[\sum\nolimits_{n \in S_V^\star(t)} \mu_n\right].$$

Because it is evaluated off-policy without evolving an independent queueing system, $\pi_V^\star$ is a statewise learning comparator, not an executable scheduler or a classical policy-regret benchmark.

Assumption 1 (Conditionally unbiased rewards)**.** *For every round t,*

$$\mathbb{E}[X_n(t) \mid \mathcal{F}_t] = \mu_n \text{ on } \{n \in S(t)\}.$$

Assumption 2 (Trajectory-wise weighted-drift certificate)**.** *For a given V, there exist $W \geq 1$, $\epsilon > 0$, and $B_{\text{win}} < \infty$ such that, with $Z_{n,t}^{(W)} = \sum_{\tau=t}^{t+W-1} (A_n(\tau) - S_{n,V}^\star(\tau))$, for every t, we have*

$$\mathbb{E}\left[\sum\nolimits_{n=1}^{N} Q_n(t) Z_{n,t}^{(W)} \mid \mathcal{F}_t\right] \leq B_{\text{win}} - W\epsilon \sum\nolimits_{n=1}^{N} Q_n(t).$$

Assumption 1 enables adaptive concentration, while Assumption 2 is a conditional windowed-load requirement along the observed trajectory. Its queue-weighted form allows service slack to be shared

across nodes and absorbs bounded transient imbalance through B_{win}; it does not assert that the reference is an independently stable policy.

Throughput Utility and Backlog Bounds. Let $c = \alpha_Q/Q_{\text{ref}}$, $B_0 = N(A_{\max}^2+1)/2$, $B_W = NW^2(A_{\max}+1)^2$, and

$$B \triangleq c\,(B_0 + (B_{\text{win}} + B_W)/W) + \alpha_W K_{\max}. \tag{8}$$

Also let $L(1) = \frac{c}{2}\sum_n Q_n^2(1)$ and $C_\beta = 3\sqrt{2\beta}$.

Theorem 1 (Finite-horizon throughput utility and backlog)**.** *Fix $V > 0$. Under Assumptions 1–2,* (P1)–(P3)*, and $\beta > 3$, for every horizon H,*

$$U^\star_{H,V} - \frac{1}{H}\sum_{t=1}^{H}\mathbb{E}[U(t)] \le \frac{B}{V} + C_\beta\sqrt{\frac{NK_{\max}\log(H+1)}{H}} + \frac{WK_{\max}}{H} + \frac{E_{\text{bad}}(V)+L(1)}{VH}. \tag{9}$$

where $E_{\text{bad}}(V) \le a_E + b_E V$ for constants a_E, b_E independent of V and H. For $H = mW$,

$$\begin{aligned}\tfrac{1}{H}\sum_{t=1}^{H}\sum_{n=1}^{N}\mathbb{E}[Q_n(t)] \le{}& \tfrac{B+VK_{\max}}{c\epsilon} + NWA_{\max} \\ &+ \tfrac{C_\beta V}{c\epsilon}\sqrt{\tfrac{NK_{\max}\log(H+1)}{H}} + \tfrac{E_{\text{bad}}(V)+L(1)}{c\epsilon H}.\end{aligned} \tag{10}$$

For a general H, the final incomplete window contributes at most $NWA_{\max} + NW\max_n Q_n(1)/H$.

For fixed system constants, the bounds yield an $O(1/V)$ queue-and-urgency term, vanishing learning loss, and an $O(1+V)$ average backlog; the latter primarily concerns open arrivals, since closed-loop queues are bounded by construction (Appendix D). Define the cumulative statewise learning gap as

$$\text{Reg}_H(V) \triangleq \sum_{t=1}^{H}\mathbb{E}\left[\sum_{n\in S^\star_V(t)}\mu_n - U(t)\right] = H\cdot U^\star_{H,V} - \sum_{t=1}^{H}\mathbb{E}[U(t)]. \tag{11}$$

Under the stronger requirement that the drift certificate holds uniformly over $V \in [1, \sqrt{H}]$, choosing $V = \sqrt{H}$ yields $\text{Reg}_H(\sqrt{H}) = \widetilde{O}(\sqrt{NK_{\max}H})$ and average backlog $O(\sqrt{H})$. The comparator is then horizon-dependent.

Pathwise SLO Protection. The following results require no reward or arrival distribution and hold on every path satisfying (3).

Proposition 1 (Deterministic exposure-to-completion bound)**.** *Under* (P3)*, every exposed head completes within*

$$B_{\text{head}} \triangleq D_{\text{ref}} + (\lceil N/(K_{\max})\rceil - 1)\,T^{\max}_{\text{cyc}} \tag{12}$$

of exposure; its lateness is at most $(\lceil N/K_{\max}\rceil - 1)T^{\max}_{\text{cyc}}$.

Once a head becomes critical, at most $N-1$ exposed heads can precede it in EDF order, and newly exposed successors cannot overtake it. Draining these heads yields (12).

Definition 1 (Critical-load condition)**.** *At every round, at most $K_{\max}$ exposed heads are critical.*

Corollary 1 (Zero violation)**.** *Under* (P3) *and (3), every exposed head meets D_{ref} whenever the critical-load condition (Definition 1) holds. The condition holds automatically when $N \le K_{\max}$.*

The condition is sufficient, not necessary; proofs appear in Appendix C.

6 Experimental Evaluation

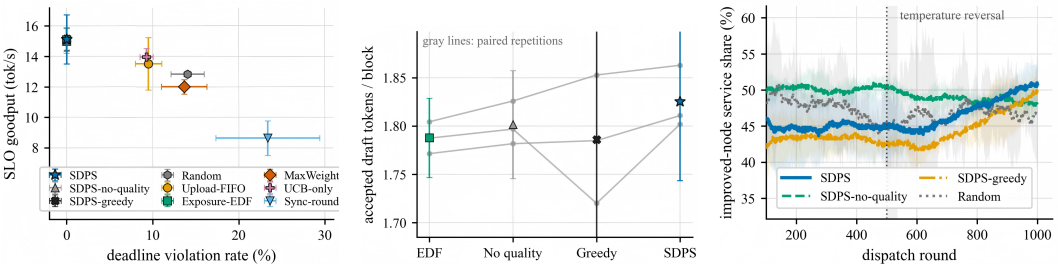
Setup. Our testbed uses an RTX 5090 Qwen3-8B verifier and six Qwen3-1.7B/0.6B drafters with dual-center verification. We add 300 ms per batch to induce contention; Figure 4(c) examines delay sensitivity. Three seeded runs use 1,000 dispatches per policy, with temperature-induced quality reversal at round 500. We compare Upload-FIFO, Exposure-EDF, MaxWeight, UCB-only, Random, SLED-inspired Sync-round (Li et al., 2025b), and two ablations: SDPS-greedy and SDPS-no-quality. Workload, calibration, baseline, and metric details appear in Appendix E.

Table 1: Three-run testbed results as mean (95% CI half-width), with P99 exposure-to-completion latency, accepted draft tokens (Acc.), and round-averaged total pending blocks (Backlog).

Policy	**Goodput** (tok/s)	**Viol.** (%)	**P99** (ms)	**Acc.** (tok/blk)	**Backlog** (blocks)
Upload-FIFO	13.51 (1.72)	9.5 (1.5)	2175 (438)	1.795 (0.045)	16.01 (0.17)
Exposure-EDF	15.10 (0.78)	0.0	1145 (76)	1.788 (0.041)	11.20 (0.11)
MaxWeight	12.00 (0.47)	13.7 (2.6)	2603 (449)	1.751 (0.063)	17.36 (0.25)
UCB-only	13.96 (0.56)	9.3 (0.8)	4341 (224)	1.875 (0.067)	10.66 (0.34)
Sync-round	8.63 (1.14)	23.4 (6.0)	2032 (135)	1.819 (0.122)	16.47 (0.41)
Random	12.83 (0.12)	14.0 (1.9)	3727 (354)	1.781 (0.044)	12.14 (0.26)
SDPS-no-quality	15.01 (0.81)	0.0	1459 (8)	1.801 (0.056)	11.89 (0.42)
SDPS-greedy	15.15 (0.73)	0.02 (0.07)	1480 (29)	1.786 (0.165)	11.95 (0.84)
SDPS	15.10 (1.61)	0.03 (0.14)	1474 (6)	1.825 (0.082)	11.11 (0.59)

Throughput and SLO performance. Table 1 and Figures 3(a)–3(b) show that SDPS combines 15.105 tok/s SLO goodput with a near-zero violation rate, while its 1474 ms P99 latency remains below the profiled D_{ref} range. Relative to Upload-FIFO, MaxWeight, UCB-only, Random, and Sync-round, SDPS improves SLO goodput by 11.8%, 25.9%, 8.2%, 17.7%, and 75.0%, respectively, while reducing their 9.29%–23.35% violation rates to near zero. These results show that upload order, queue pressure, yield-only selection, random admission, and synchronized batching cannot jointly maintain productive output and per-block latency.

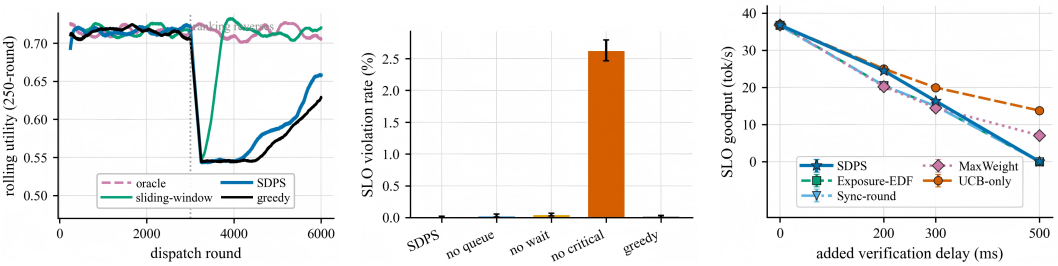
Compared with the stronger SLO-safe policies (Exposure-EDF, SDPS-greedy, and SDPS-no-quality), SDPS retains comparable SLO goodput while delivering consistent accepted-token gains. These policies provide strong deadline protection, limiting the room in which quality-aware scheduling can further improve goodput. The effect of quality awareness becomes more visible after the edge-drafter quality ranking is reversed. Verification productivity reveals this difference. SDPS accepts 1.825 draft tokens per verified block, improving on SDPS-no-quality, Exposure-EDF, and SDPS-greedy by 1.3%, 2.1%, and 2.2% on average, respectively. Goodput alone cannot separate these policies once deadline protection is in place; accepted-token yield measures how much each admitted verification returns, and the ordering is consistent. SDPS exceeds all three policies in every paired repetition, providing evidence that quality-aware admission improves the productive value of each verification. Although UCB-only obtains a higher yield, it also incurs substantial latency and SLO violations, showing that unconstrained yield maximization is not a viable shared-verifier policy. SDPS therefore improves the productive efficiency of the shared verifier while maintaining near-zero empirical exposure-SLO violations. The round-averaged total backlog is 11.11 blocks for SDPS, compared with 16.01 for Upload-FIFO and 17.36 for MaxWeight, reductions of 30.6% and 36.0%, respectively. Exposure-EDF has a comparable backlog of 11.20 blocks. SDPS and Exposure-EDF both use full two-request batches in more than 99.8% of dispatches, indicating that the accepted-token gain reflects verification yield rather than better batch filling.



(a) SLO Trade-off (b) Accepted-Token Yield (c) Real Quality Reversal

Figure 3: Real batch-2 performance and adaptation over three runs. Error bars in (a–b) show 95% CIs; (c) shows mean service-share trajectories.

Adaptation under reward drift. Figure 3(c) evaluates whether learned quality changes allocation in the physical system. Across three repetitions, SDPS increases the service share assigned



(a) Reward Reversal (b) Critical-First Ablation (c) Verifier-Delay Sensitivity

Figure 4: Controlled simulation over five seeds: reward adaptation (a), critical-first safety (b), and delay sensitivity (c). Curves in (a) show seed means; error bars in (b–c) show standard deviations.

to improved nodes from 45.13% before the reversal to 47.60% afterward, a mean shift of $+2.47$ percentage points that is positive in every repetition. In contrast, SDPS-no-quality, Exposure-EDF, and Random shift by -1.43, -1.28, and -0.35 points on average, respectively. SDPS-greedy also adapts ($+2.61$ points), showing that learned quality, rather than the stationary UCB bonus alone, drives the allocation response. Thus, the real testbed as a whole supports the effectiveness of the quality-aware mechanism. Figure 4(a) evaluates persistent nonstationarity under a controlled ranking reversal. Over the post-change period, sliding-window SDPS reaches normalized utility 0.693, compared with 0.582 for stationary UCB and 0.569 for greedy scheduling. It improves on greedy by 21.7% and remains within 3.0% of the dynamic oracle, showing that recency-aware estimation recovers productive output when full-history estimates become stale.

Mechanism and operating regime. Figure 4(b) identifies strict critical-first promotion as the main SLO-safety mechanism: removing it raises violations from 0.014% to 2.63%, increases P99 latency from 1419 to 1737 ms beyond the 1500 ms SLO, and lowers goodput by 1.44%. Complementary testbed ablations show that removing learned quality lowers accepted-token yield from 1.825 to 1.801 and reverses the post-change allocation shift from $+2.47$ to -1.43 points, whereas unconstrained UCB-only selection obtains higher yield at the cost of 9.29% violations and 4.34 s P99 latency. Figure 4(c) locates the operating regime in which quality-aware scheduling is useful. With no added verification delay, SDPS and Exposure-EDF achieve nearly identical SLO goodput. Under sustained but feasible contention, SDPS improves SLO goodput by 19.5% and 9.1% at 200 and 300 ms, respectively. At 500 ms, SDPS, Exposure-EDF, and Sync-round reach zero SLO goodput, marking an extreme regime beyond SDPS's effective range.

Theory consistency and overhead. At $V = 5$, the statewise utility gap is consistent with the predicted $\sqrt{\log H/H}$ shape ($R^2 = 0.998$), while the normalized score-learning diagnostic falls from 0.519 to 0.077. Under overload, SDPS remains 34% below B_{head} whereas UCB-only exceeds it by over $20\times$. In the physical traces, all $5{,}996$ SDPS blocks satisfy their profiled B_{head}; the only two stricter-SLO violations occur in the sole run whose sufficient-condition audit fails. Moreover, $26{,}962/27{,}000$ dispatches are full real batch-2 executions, and scheduling costs 0.062 ms on average (0.130 ms) versus 372.8 ms for batch verification, confirming negligible $O(|\mathcal{A}_t| \log K_t)$ overhead.

7 Conclusion

This paper presented SDPS, a queue-aware bandit scheduler for shared verification under unknown edge-node yields and per-block latency SLOs. SDPS combines quality-aware admission with strict EDF promotion, providing finite-horizon utility and backlog bounds under stated assumptions and deterministic exposure-to-completion protection under bounded cycle durations. On the testbed, SDPS achieves 15.105 tok/s SLO goodput with 0.033% violations, improving goodput by 11.8% over Upload-FIFO and 25.9% over MaxWeight. Against Exposure-EDF and the quality ablations, it maintains comparable goodput while increasing accepted-token yield by 1.3%–2.2%, with positive paired gains in all three runs. After quality reversal, its service share toward improved nodes rises by 2.47 percentage points. Controlled simulations confirm critical-first safety and show that sliding-window estimation improves post-reversal utility by 21.7% over greedy.

AI USE STATEMENT

We used generative AI tools to assist with experimental design, implementation and debugging, interpretation of experimental results, scientific figure preparation, translation, and manuscript editing. AI tools also assisted with discussion and revision of the theoretical exposition. The authors are responsible for verifying the mathematical claims, code, experimental results, and cited sources, and take responsibility for the final submission.

ETHICS STATEMENT

This work studies verification scheduling rather than the capabilities or safety of the underlying language models. Exposure-based latency protection does not imply end-to-end request guarantees or equal service shares across users. Any deployment must separately address data privacy, access control, and application-specific safety requirements.

REPRODUCIBILITY STATEMENT

Algorithm 1 specifies the admission rule. Appendix E details the workload, calibration, baselines, and metrics. Proofs and analytical assumptions are provided in Appendices A–D.

REFERENCES

Amey Agrawal, Nitin Kedia, Ashish Panwar, Jayashree Mohan, Nipun Kwatra, Bhargav S. Gulavani, Alexey Tumanov, and Ramachandran Ramjee. Taming throughput-latency tradeoff in LLM inference with Sarathi-Serve. In *USENIX Symposium on Operating Systems Design and Implementation (OSDI)*, 2024.

Peter Auer, Nicolò Cesa-Bianchi, and Paul Fischer. Finite-time analysis of the multiarmed bandit problem. *Machine Learning*, 47:235–256, 2002.

Tianle Cai, Yuhong Li, Zhengyang Geng, Hongwu Peng, Jason D. Lee, Deming Chen, and Tri Dao. Medusa: Simple LLM inference acceleration framework with multiple decoding heads. In *International Conference on Machine Learning (ICML)*, 2024.

Charlie Chen, Sebastian Borgeaud, Geoffrey Irving, Jean-Baptiste Lespiau, Laurent Sifre, and John Jumper. Accelerating large language model decoding with speculative sampling. *arXiv preprint arXiv:2302.01318*, 2023.

Wei Chen, Yajun Wang, and Yang Yuan. Combinatorial multi-armed bandit: General framework and applications. In *International Conference on Machine Learning (ICML)*, 2013.

Wenyan Chen, Chengzhi Lu, Yanying Lin, and Dmitrii Ustiugov. FASER: Fine-grained phase management for speculative decoding in dynamic LLM serving. *arXiv preprint arXiv:2604.20503*, 2026.

Yi Gai, Bhaskar Krishnamachari, and Rahul Jain. Combinatorial network optimization with unknown variables: Multi-armed bandits with linear rewards and individual observations. *IEEE/ACM Transactions on Networking*, 20(5):1466–1478, 2012.

Leonidas Georgiadis, Michael J. Neely, and Leandros Tassiulas. Resource allocation and cross-layer control in wireless networks. *Foundations and Trends in Networking*, 1(1):1–144, 2006.

Yunhe Han, Yunqi Gao, Bing Hu, Mahdi Boloursaz Mashhadi, Yitong Duan, Pei Xiao, and Yanfeng Zhang. PipeSD: An efficient cloud–edge collaborative pipeline inference framework with speculative decoding. In *Proceedings of the 43rd International Conference on Machine Learning*, volume 306 of *Proceedings of Machine Learning Research*. PMLR, 2026. URL https://arxiv.org/abs/2605.13319.

Kaiyu Huang, Hao Wu, Zhubo Shi, Han Zou, Minchen Yu, and Qingjiang Shi. AdaSpec: Adaptive speculative decoding for fast, SLO-aware large language model serving. In *ACM Symposium on Cloud Computing (SoCC)*, 2025. doi: 10.1145/3772052.3772239.

Yiping Kang, Johann Hauswald, Cao Gao, Austin Rovinski, Trevor Mudge, Jason Mars, and Lingjia Tang. Neurosurgeon: Collaborative intelligence between the cloud and mobile edge. In *International Conference on Architectural Support for Programming Languages and Operating Systems (ASPLOS)*, 2017.

Subhashini Krishnasamy, Rajat Sen, Ramesh Johari, and Sanjay Shakkottai. Learning unknown service rates in queues: A multiarmed bandit approach. *Operations Research*, 69(1):315–330, 2021.

Branislav Kveton, Zheng Wen, Azin Ashkan, and Csaba Szepesvári. Tight regret bounds for stochastic combinatorial semi-bandits. In *International Conference on Artificial Intelligence and Statistics (AISTATS)*, 2015.

Woosuk Kwon, Zhuohan Li, Siyuan Zhuang, Ying Sheng, Lianmin Zheng, Cody Hao Yu, Joseph E. Gonzalez, Hao Zhang, and Ion Stoica. Efficient memory management for large language model serving with PagedAttention. In *ACM Symposium on Operating Systems Principles (SOSP)*, 2023.

Yaniv Leviathan, Matan Kalman, and Yossi Matias. Fast inference from transformers via speculative decoding. In *International Conference on Machine Learning (ICML)*, 2023.

Ruixiao Li, Fahao Chen, and Peng Li. Semi-clairvoyant scheduling of speculative decoding requests to minimize LLM inference latency. In *Proceedings of the Thirty-Fourth International Joint Conference on Artificial Intelligence*, pp. 8554–8562, 2025a. doi: 10.24963/ijcai.2025/951.

Xiangchen Li, Dimitrios Spatharakis, Saeid Ghafouri, Jiakun Fan, Hans Vandierendonck, Deepu John, Bo Ji, and Dimitrios S. Nikolopoulos. SLED: A speculative LLM decoding framework for efficient edge serving. In *ACM/IEEE Symposium on Edge Computing (SEC)*, 2025b.

Xiangchen Li, Jiakun Fan, Qingyuan Wang, Dimitrios Spatharakis, Saeid Ghafouri, Hans Vandierendonck, Deepu John, Bo Ji, Ali R. Butt, and Dimitrios Nikolopoulos. WISP: Waste- and interference-suppressed distributed speculative LLM serving at the edge via dynamic drafting and SLO-aware batching. *Proceedings of the ACM on Measurement and Analysis of Computing Systems*, 10(2):Article 55, May 2026a. doi: 10.1145/3805653. URL `https://doi.org/10.1145/3805653`.

Yuhui Li, Fangyun Wei, Chao Zhang, and Hongyang Zhang. EAGLE: Speculative sampling requires rethinking feature uncertainty. In *International Conference on Machine Learning (ICML)*, 2024.

Zikun Li, Zhuofu Chen, Remi Delacourt, Gabriele Oliaro, Zeyu Wang, Qinghan Chen, Shuhuai Lin, April Yang, Zhihao Zhang, Zhuoming Chen, Yi-Hsiang Lai, Xinhao Cheng, Xupeng Miao, and Zhihao Jia. AdaServe: Accelerating multi-SLO LLM serving with SLO-customized speculative decoding. In *European Conference on Computer Systems (EuroSys)*, 2026b. doi: 10.1145/3767295.3769315.

C. L. Liu and James W. Layland. Scheduling algorithms for multiprogramming in a hard-real-time environment. *Journal of the ACM*, 20(1):46–61, 1973.

Xiaoxuan Liu, Lanxiang Hu, Peter Bailis, Alvin Cheung, Zhijie Deng, Ion Stoica, and Hao Zhang. Online speculative decoding. In *International Conference on Machine Learning (ICML)*, 2024a.

Xiaoxuan Liu, Jongseok Park, Langxiang Hu, Woosuk Kwon, Zhuohan Li, Chen Zhang, Kuntai Du, Xiangxi Mo, Kaichao You, Alvin Cheung, Zhijie Deng, Ion Stoica, and Hao Zhang. TurboSpec: Closed-loop speculation control system for optimizing LLM serving goodput. *arXiv preprint arXiv:2406.14066*, 2024b.

Xupeng Miao, Gabriele Oliaro, Zhihao Zhang, Xinhao Cheng, Zeyu Wang, Zhengxin Zhang, Rae Ying Yee Wong, Alan Zhu, Lijie Yang, Xiaoxiang Shi, Chunan Shi, Zhuoming Chen, Daiyaan Arfeen, Reyna Abhyankar, and Zhihao Jia. SpecInfer: Accelerating large language model serving with tree-based speculative inference and verification. In *International Conference on Architectural Support for Programming Languages and Operating Systems (ASPLOS)*, 2024.

Michael J. Neely. *Stochastic Network Optimization with Application to Communication and Queueing Systems*. Morgan & Claypool, 2010.

Michael J. Neely, Scott T. Rager, and Thomas F. La Porta. Max-weight learning algorithms for scheduling in unknown environments. *IEEE Transactions on Automatic Control*, 57(5):1179–1191, 2012.

Pratyush Patel, Esha Choukse, Chaojie Zhang, Aashaka Shah, Íñigo Goiri, Saeed Maleki, and Ricardo Bianchini. Splitwise: Efficient generative LLM inference using phase splitting. In *ACM/IEEE International Symposium on Computer Architecture (ISCA)*, 2024.

Haifeng Qian, Sujan Kumar Gonugondla, Sungsoo Ha, Mingyue Shang, Sanjay Krishna Gouda, Ramesh Nallapati, Sudipta Sengupta, Xiaofei Ma, and Anoop Deoras. BASS: Batched attention-optimized speculative sampling. In *Findings of the Association for Computational Linguistics (ACL)*, 2024.

Thomas Stahlbuhk, Brooke Shrader, and Eytan Modiano. Learning algorithms for minimizing queue length regret. *IEEE Transactions on Information Theory*, 67(3):1759–1781, 2021.

Mitchell Stern, Noam Shazeer, and Jakob Uszkoreit. Blockwise parallel decoding for deep autoregressive models. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2018.

Ziteng Sun, Ananda Theertha Suresh, Jae Hun Ro, Ahmad Beirami, Himanshu Jain, and Felix Yu. SpecTr: Fast speculative decoding via optimal transport. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2023.

Leandros Tassiulas and Anthony Ephremides. Stability properties of constrained queueing systems and scheduling policies for maximum throughput in multihop radio networks. *IEEE Transactions on Automatic Control*, 37(12):1936–1948, 1992.

Phuong Tran, Tzu-Hao Liu, Long Tan Le, Tung-Anh Nguyen, Van Quan La, Eason Yu, Han Shu, Choong Seon Hong, and Nguyen H. Tran. GoodSpeed: Optimizing fair goodput with adaptive speculative decoding in distributed edge inference. In *IEEE INFOCOM 2026 – IEEE Conference on Computer Communications*, pp. 1–10. IEEE, 2026. doi: 10.1109/INFOCOM59046.2026.11571335. URL https://doi.org/10.1109/INFOCOM59046.2026.11571335.

Zhaoxuan Wu, Zijian Zhou, Arun Verma, Alok Prakash, Daniela Rus, and Bryan Kian Hsiang Low. TETRIS: Optimal draft token selection for batch speculative decoding. In *Annual Meeting of the Association for Computational Linguistics (ACL)*, pp. 33329–33345, 2025.

Heming Xia, Zhe Yang, Qingxiu Dong, Peiyi Wang, Yongqi Li, Tao Ge, Tianyu Liu, Wenjie Li, and Zhifang Sui. Unlocking efficiency in large language model inference: A comprehensive survey of speculative decoding. In *Findings of the Association for Computational Linguistics (ACL)*, 2024.

Yaodan Xu, Sheng Zhou, and Zhisheng Niu. Dip-sd: Distributed pipelined speculative decoding for efficient LLM inference at the edge. *arXiv preprint arXiv:2604.20919*, 2026.

Zixian Yang, R. Srikant, and Lei Ying. Learning while scheduling in multi-server systems with unknown statistics: MaxWeight with discounted UCB. In *International Conference on Artificial Intelligence and Statistics (AISTATS)*, 2023.

Gyeong-In Yu, Joo Seong Jeong, Geon-Woo Kim, Soojeong Kim, and Byung-Gon Chun. Orca: A distributed serving system for transformer-based generative models. In *USENIX Symposium on Operating Systems Design and Implementation (OSDI)*, 2022.

Yinmin Zhong, Shengyu Liu, Junda Chen, Jianbo Hu, Yibo Zhu, Xuanzhe Liu, Xin Jin, and Hao Zhang. DistServe: Disaggregating prefill and decoding for goodput-optimized large language model serving. In *USENIX Symposium on Operating Systems Design and Implementation (OSDI)*, 2024.

Yongchao Zhou, Kaifeng Lyu, Ankit Singh Rawat, Aditya Krishna Menon, Afshin Rostamizadeh, Sanjiv Kumar, Jean-François Kagy, and Rishabh Agarwal. DistillSpec: Improving speculative decoding via knowledge distillation. In *International Conference on Learning Representations (ICLR)*, 2024.

Zhi Zhou, Xu Chen, En Li, Liekang Zeng, Ke Luo, and Junshan Zhang. Edge intelligence: Paving the last mile of artificial intelligence with edge computing. *Proceedings of the IEEE*, 107(8): 1738–1762, 2019.

APPENDIX

A DETAILED PROOF OF THEOREM 1

A.1 SETTING, NOTATION, AND THE REWARD FILTRATION

Throughout, $\mathcal{F}_t$ is the history up to the beginning of round t; $c = \alpha_Q/Q_{\mathrm{ref}}$; $S_n(t) \in \{0,1\}$ (per-node cap); $0 \le A_n(t) \le A_{\max}$, with no assumed rate and no independence from the round's decision; $X_n(t) \in (0,1]$; $|S(t)| \le K_{\max}$. Queues may start at any finite state, and grow at most linearly: $Q_n(t) \le Q_n(1) + A_{\max} t$ deterministically. The comparator $\pi^\star_V$ is the *per-round statewise reference* defined in the main paper—SDPS at the same V with $w_n(t)$ replaced by μ_n, evaluated along SDPS's trajectory—and not an executable scheduler. At each round it selects $S^\star_V(t) \subseteq \mathcal{A}_t$ with $|S^\star_V(t)| = K_t$, respecting the same critical-first protocol and (R_n, n) tie-break; its selections are $\mathcal{F}_t$-measurable and do not feed back into the system path. The empirical mean $\widehat{\mu}_n(t)$ is the average of the first $H_n(t)$ observations.

Restated analytical conditions. For self-containment, we restate the two analytical conditions used by the main theorem.

Assumption 3 (Conditionally unbiased rewards)**.** *For every round t,*

$$\mathbb{E}\left[X_n(t) \mid \mathcal{F}_t\right] = \mu_n \text{ on } \{n \in S(t)\}. \tag{A.1}$$

Assumption 4 (Trajectory-wise weighted-drift certificate)**.** *For a given V, there exist $W \ge 1$, $\epsilon > 0$, and $B_{\mathrm{win}} < \infty$ such that, with*

$$Z^{(W)}_{n,t} \triangleq \sum\nolimits_{\tau=t}^{t+W-1} \left(A_n(\tau) - S^\star_{n,V}(\tau)\right), \tag{A.2}$$

we have

$$\mathbb{E}\left[\sum\nolimits_n Q_n(t) Z^{(W)}_{n,t} \mid \mathcal{F}_t\right] \le B_{\mathrm{win}} - W\epsilon \sum\nolimits_n Q_n(t) \tag{A.3}$$

for every t.

Admission-indexed reward filtration. Assumption 3 is used in two equivalent forms. The round-indexed form is Eq. (A.1): $\mathbb{E}[X_n(t) \mid \mathcal{F}_t] = \mu_n$ on $\{n \in S(t)\}$. For the concentration argument we index by admissions: let $\tau_{n,h}$ be the round of node n's h-th admission, write $X_{n,h} = X_n(\tau_{n,h})$, and let

$$\mathcal{G}_{n,h-1} \triangleq \sigma\left(\mathcal{F}_{\tau_{n,h}},\, S(\tau_{n,h})\right) \tag{A.4}$$

be the information available after the round-$\tau_{n,h}$ admission decision and before its reward is observed. Then

$$\mathbb{E}\left[X_{n,h} \mid \mathcal{G}_{n,h-1}\right] = \mu_n \text{ for all } h \ge 1. \tag{A.5}$$

Since SDPS's decision is a deterministic function of $\mathcal{F}_{\tau_{n,h}}$, we have $\mathcal{G}_{n,h-1} = \sigma(\mathcal{F}_{\tau_{n,h}})$, so Eqs. (A.1) and (A.5) are the same condition. Moreover $X_{n,1}, \ldots, X_{n,h-1}$ are contained in $\mathcal{G}_{n,h-1}$, having been observed in earlier rounds, and each $X_{n,h}$ is $\mathcal{G}_{n,h}$-measurable, having been observed before round $\tau_{n,h+1}$; hence $\{\mathcal{G}_{n,h}\}_{h\ge 0}$ is a filtration adapted to node n's observation sequence.

Roadmap. The proof of Theorem 1 proceeds in six steps. Step 1 (A.2) establishes the one-step weighted Lyapunov drift and identifies the drift-plus-penalty objective. Step 2 (A.3) shows that SDPS's lexicographic rule maximizes an optimistic perturbation of that objective. Step 3 (A.4) proves the concentration lemma. Step 4 (A.5) compares SDPS with $\pi^\star_V$ round by round, on and off the good event, and quantifies the bad-event charge. Step 5 (A.10) telescopes over W-round windows using the certificate. Step 6 (A.11) bounds the accumulated confidence radii by a counting inequality and assembles the theorem. Two companion results follow: A.12 proves a square-root score-learning gap at every fixed V, and A.13 proves the horizon-tuned statewise-gap Corollary 2.

A.2 STEP 1: ONE-STEP WEIGHTED DRIFT

Define the weighted quadratic Lyapunov function and its conditional drift,

$$L(t) = \frac{c}{2} \sum\nolimits_{n=1}^{N} Q_n^2(t),\ \Delta(t) = \mathbb{E}[L(t+1) - L(t) \mid \mathcal{F}_t]. \tag{A.6}$$

Lemma 1 (One-step drift). *With* $B_0 = \frac{N}{2}(A_{\max}^2 + 1)$,

$$\Delta(t) \le cB_0 + c\sum\nolimits_{n=1}^{N} Q_n(t)\,\mathbb{E}\left[A_n(t) - S_n(t) \mid \mathcal{F}_t\right]. \tag{A.7}$$

Proof. Fix n and write $q = Q_n(t)$, $s = S_n(t) \in \{0,1\}$, $a = A_n(t) \ge 0$. The queue recursion Eq. (2) gives $Q_n(t+1) = [q-s]^+ + a$.

Case $q \ge s$. Then $[q-s]^+ = q - s$, and expanding the square,

$$Q_n^2(t+1) = (q - s + a)^2 = q^2 + s^2 + a^2 + 2q(a-s) - 2sa \le q^2 + s^2 + a^2 + 2q(a-s),$$

where the inequality drops $-2sa \le 0$.

Case $q < s$. Then $q = 0$ and $s = 1$, so

$$Q_n^2(t+1) = a^2 \le a^2 + 1 = q^2 + s^2 + a^2 + 2q(a-s).$$

In both cases, using $s^2 \le 1$ and $a^2 \le A_{\max}^2$ (property (P1)),

$$Q_n^2(t+1) - Q_n^2(t) \le s^2 + a^2 + 2q(a-s) \le A_{\max}^2 + 1 + 2\,Q_n(t)(A_n(t) - S_n(t)). \tag{A.8}$$

Sum Eq. (A.8) over $n = 1, \dots, N$, multiply by $c/2$, and take conditional expectations given $\mathcal{F}_t$, under which $Q_n(t)$ and $S_n(t)$ are measurable; Eq. (A.7) follows. Two features are deliberate. First, $\mathbb{E}[A_n(t) \mid \mathcal{F}_t]$ is left *as is*: no arrival rate is assumed and uploads may depend on past decisions, since the certificate Eq. (A.3) is stated in exactly this form and absorbs the term wholesale in Step 5. Second, service is deterministic in queue units—one admission provides exactly one unit of service—so no completion model appears. □

Step 1b: the drift-minus-reward inequality. Subtract $V\,\mathbb{E}[U(t) \mid \mathcal{F}_t]$ from both sides of Eq. (A.7). Since $S(t)$ is $\mathcal{F}_t$-measurable, Assumption 3 in the form Eq. (A.1) gives

$$\mathbb{E}[U(t) \mid \mathcal{F}_t] = \mathbb{E}\left[\sum\nolimits_{n \in S(t)} X_n(t) \middle| \mathcal{F}_t\right] = \sum\nolimits_{n \in S(t)} \mu_n.$$

Splitting $Q_n(t)\,\mathbb{E}[A_n(t) - S_n(t) \mid \mathcal{F}_t]$ into its upload and service parts and regrouping the service part with the reward,

$$\Delta(t) - V\mathbb{E}[U(t) \mid \mathcal{F}_t] \le cB_0 + c\sum\nolimits_{n=1}^{N} Q_n(t)\mathbb{E}[A_n(t) \mid \mathcal{F}_t] - \mathbb{E}\left[\sum\nolimits_{n=1}^{n \in S(t)} (c\,Q_n(t) + V\mu_n) \middle| \mathcal{F}_t\right]. \tag{A.9}$$

The bracketed sum is the *drift-plus-penalty objective*; its queue weights $c\,Q_n = \alpha_Q Q_n / Q_{\mathrm{ref}}$ coincide exactly with the queue term of the score Eq. (6)—the coincidence enforced by the weighted Lyapunov function, which enables the arg-max argument of Step 2 without rescaling.

A.3 Step 2: The Selection Rule as a Perturbed Optimistic Arg-Max

Fix t and condition on $\mathcal{F}_t$. Let $\mathcal{C}_t = \{n \in \mathcal{A}_t : z_n(t) = 1\}$ be the critical class and $m_t = \min\{|\mathcal{C}_t|, K_t\}$.

Step 2a: the critical positions coincide and cancel. Both SDPS and $\pi_V^\star$ are non-idling at dispatch epochs (Eq. (7)), obey the same lexicographic protocol, and order the critical class by the same key (R_n, n) (the SDPS algorithm in the main paper); neither key involves μ_n or $w_n(t)$, so the two rules fill the first m_t positions with the *identical* set of nodes, call it $\mathcal{C}_t^m$, which is $\mathcal{F}_t$-measurable. Hence, for any function f,

$$\sum\nolimits_{n \in S(t)} f(n) - \sum\nolimits_{n \in S_V^\star(t)} f(n) = \sum\nolimits_{n \in S^{\mathrm{res}}(t)} f(n) - \sum\nolimits_{n \in S_V^{\star,\mathrm{res}}(t)} f(n), \tag{A.10}$$

where $S^{\mathrm{res}}(t) = S(t) \setminus \mathcal{C}_t^m$ and $S_V^{\star,\mathrm{res}}(t) = S_V^\star(t) \setminus \mathcal{C}_t^m$ are the residual selections, both of size $K_t - m_t$, both drawn from $\mathcal{A}_t \setminus \mathcal{C}_t$.

Step 2b: the residual rule is an arg-max. On $\mathcal{A}_t \setminus \mathcal{C}_t$, SDPS ranks by $p_n(t)$ of Eq. (6) and takes the top $K_t - m_t$. Since $c = \alpha_Q / Q_{\mathrm{ref}}$ by definition, the ordinary-node score *is* $p_n(t) = c\,Q_n(t) +$

$\alpha_W D_n(t)/D_{\text{ref}} + V w_n(t)$ already—no rescaling of the key is performed or needed—so SDPS's residual selection satisfies

$$S^{\text{res}}(t) \in \arg \max_{S \subseteq \mathcal{A}_t \setminus \mathcal{C}_t, |S| = K_t - m_t} \sum\nolimits_{n \in S} \left(c Q_n(t) + \alpha_W \tfrac{D_n(t)}{D_{\text{ref}}} + V w_n(t) \right), \tag{A.11}$$

and $S_V^{\star,\text{res}}(t)$ satisfies the same with $w_n(t)$ replaced by μ_n. Crucially, the queue weight in Eq. (A.11) is $c\,Q_n(t)$—exactly the weight the drift Eq. (A.9) carries—because the Lyapunov function was scaled by $c = \alpha_Q/Q_{\text{ref}}$ for this purpose. In particular, V keeps its meaning throughout: it is neither rescaled nor renamed.

Step 2c: the waiting term is a bounded perturbation. Every noncritical node has $D_n(t) < D_{\text{crit}} < D_{\text{ref}}$, so for every candidate S with $|S| \le K_{\max}$,

$$0 \le \sum\nolimits_{n \in S} \alpha_W D_n(t)/D_{\text{ref}} \le \alpha_W K_{\max}. \tag{A.12}$$

It is present in the objective SDPS maximizes but absent from the drift-plus-penalty objective, and Eq. (A.12) is what lets us add and remove it at a cost of $\alpha_W K_{\max}$ per round.

A.4 Step 3: Concentration and Optimism

For $H_n(t) \ge 1$ let $r_n(t) = \sqrt{\beta \log(t+1)/(2H_n(t))}$, and $r_n(t) = 1$ when $H_n(t) = 0$. Define $\mathcal{G}_t = \{|\widehat{\mu}_n(t) - \mu_n| \le r_n(t)\ \forall n : H_n(t) \ge 1\}$.

Lemma 2. *For $\beta > 3$: (i) $\Pr[\mathcal{G}_t^c] \le 2Nt(t+1)^{-\beta}$, hence $\sum_{t \ge 1} t \Pr[\mathcal{G}_t^c] \le 2N \sum_{t \ge 1} t^{2-\beta} < \infty$; (ii) on $\mathcal{G}_t$, $\mu_n \le w_n(t) \le \mu_n + 2r_n(t)$ for all n.*

Proof. (i) A single deviation. Fix n and a candidate value $h \in \{1, \dots, t-1\}$ of the admission count. By Eq. (A.5) the centered observations $D_j \triangleq X_{n,j} - \mu_n$, $j = 1, \dots, h$, form a martingale-difference sequence with respect to the filtration $\{\mathcal{G}_{n,j}\}_{j \ge 0}$ of Eq. (A.4); since $X_{n,j} \in (0, 1]$, each increment takes values in the interval $(-\mu_n,\, 1 - \mu_n]$, of length one. The Azuma–Hoeffding inequality for martingale differences with range-one increments gives, for any $\lambda > 0$,

$$\Pr\left[\left| \sum\nolimits_{j=1}^{h} D_j \right| \ge \lambda \right] \le 2 \exp(-2\lambda^2/h).$$

Choose $\lambda = h\sqrt{\beta \log(t+1)/(2h)} = \sqrt{h\beta \log(t+1)/2}$; then $2\lambda^2/h = \beta \log(t+1)$, and writing $\widehat{\mu}_n^{(h)}$ for the empirical mean of the first h observations,

$$\Pr\left[\left| \widehat{\mu}_n^{(h)} - \mu_n \right| > \sqrt{\tfrac{\beta \log(t+1)}{2h}} \right] \le 2(t+1)^{-\beta}.$$

(ii) Union bound. At round t the count $H_n(t)$ takes one of the at most $t-1$ values $1, \dots, t-1$, and $\widehat{\mu}_n(t) = \widehat{\mu}_n^{(H_n(t))}$. A union bound over these values and over the N nodes gives

$$\Pr[\mathcal{G}_t^c] \le 2N(t-1)(t+1)^{-\beta} \le 2Nt(t+1)^{-\beta},$$

and since $t \cdot t(t+1)^{-\beta} \le t^{2-\beta}$, the weighted series $\sum_{t \ge 1} t \Pr[\mathcal{G}_t^c] \le 2N \sum_{t \ge 1} t^{2-\beta}$ converges for $\beta > 3$.

(iii) Optimism. If $H_n(t) = 0$: $w_n(t) = 1 \ge \mu_n$ and $w_n(t) \le \mu_n + 2$ with $r_n(t) = 1$. If $H_n(t) \ge 1$: on $\mathcal{G}_t$, $\widehat{\mu}_n(t) + r_n(t) \ge \mu_n$, and clipping at 1 preserves this lower bound since $\mu_n \le 1$; the upper bound $w_n(t) \le \mu_n + 2r_n(t)$ uses $\widehat{\mu}_n(t) \le \mu_n + r_n(t)$ and $\min\{x, 1\} \le x$. □

A.5 Step 4: Per-Round Comparison and the Bad-Event Term

Work conditionally on $\mathcal{F}_t$ and, to begin, on the good event $\mathcal{G}_t$. Let $S_V^{\star,\text{res}}(t)$ be the benchmark's residual selection; it is feasible for the arg-max problem Eq. (A.11) (if it is smaller, nonnegativity of all terms only helps). We compare the drift-plus-penalty value of the two residual selections in three sub-steps and then account for the bad event.

A.6 STEP 4A: OPTIMISM REPLACES μ BY w AT BOUNDED COST.

On $\mathcal{G}_t$, Lemma 2(ii) gives $\mu_n \geq w_n(t) - 2r_n(t)$ for every n, hence

$$\sum\nolimits_{n\in S^{\mathrm{res}}(t)}(cQ_n + V\mu_n) \geq \sum\nolimits_{n\in S^{\mathrm{res}}(t)}(cQ_n + Vw_n) - 2V\sum\nolimits_{n\in S^{\mathrm{res}}(t)} r_n. \tag{A.13}$$

A.7 STEP 4B: INSERT THE WAITING TERM, USE OPTIMALITY, REMOVE THE WAITING TERM.

By the upper end of Eq. (A.12) applied to $S^{\mathrm{res}}(t)$, by the optimality of $S^{\mathrm{res}}(t)$ in Eq. (A.11) (of which $S_V^{\star,\mathrm{res}}(t)$ is a feasible candidate), and by the lower end of Eq. (A.12) applied to $S_V^{\star,\mathrm{res}}(t)$, respectively,

$$\begin{aligned}
\sum\nolimits_{n\in S^{\mathrm{res}}(t)}(cQ_n + Vw_n) &\geq \sum\nolimits_{n\in S^{\mathrm{res}}(t)}\left(cQ_n + \alpha_W\tfrac{D_n(t)}{D_{\mathrm{ref}}} + Vw_n\right) - \alpha_W K_{\max} && \text{(A.14)}\\
&\geq \sum\nolimits_{n\in S_V^{\star,\mathrm{res}}(t)}\left(cQ_n + \alpha_W\tfrac{D_n(t)}{D_{\mathrm{ref}}} + Vw_n\right) - \alpha_W K_{\max} && \text{(A.15)}\\
&\geq \sum\nolimits_{n\in S_V^{\star,\mathrm{res}}(t)}(cQ_n + Vw_n) - \alpha_W K_{\max}. && \text{(A.16)}
\end{aligned}$$

A.8 STEP 4C: OPTIMISM LOWER-BOUNDS w BY μ.

On $\mathcal{G}_t$, $w_n(t) \geq \mu_n$ for every n (Lemma 2(ii)), hence

$$\sum\nolimits_{n\in S_V^{\star,\mathrm{res}}(t)}(cQ_n + Vw_n) \geq \sum\nolimits_{n\in S_V^{\star,\mathrm{res}}(t)}(cQ_n + V\mu_n). \tag{A.17}$$

Chaining Eqs. (A.13)–(A.17) gives, on $\mathcal{G}_t$,

$$\sum\nolimits_{n\in S^{\mathrm{res}}(t)}(cQ_n+V\mu_n) \geq \sum\nolimits_{n\in S_V^{\star,\mathrm{res}}(t)}(cQ_n+V\mu_n) - \alpha_W K_{\max} - 2V\sum\nolimits_{n\in S^{\mathrm{res}}(t)} r_n. \tag{A.18}$$

Adding the identical critical contributions (Step 2a) to both sides gives, on $\mathcal{G}_t$,

$$\begin{aligned}
\mathbb{E}\left[\sum\nolimits_{n\in S(t)}(cQ_n + V\mu_n)\middle|\mathcal{F}_t\right] \geq{}& \mathbb{E}\left[\sum\nolimits_{n\in S_V^\star(t)}(cQ_n + V\mu_n)\middle|\mathcal{F}_t\right]\\
&- \alpha_W K_{\max} - 2V\mathbb{E}\left[\sum\nolimits_{n\in S(t)} r_n(t)\middle|\mathcal{F}_t\right].
\end{aligned} \tag{A.19}$$

A.9 STEP 4D: THE BAD EVENT.

On $\mathcal{G}_t^c$ the chain may fail, and we charge the failure explicitly. Define

$$e_t \triangleq \mathbf{1}_{\mathcal{G}_t^c}\left[c\sum\nolimits_n Q_n(t) + 2VK_{\max}\right]. \tag{A.20}$$

On $\mathcal{G}_t^c$, the left side of Eq. (A.19) falls short of its right side by at most $\sum_{n\in S_V^\star(t)} cQ_n + V\sum_{S^\star}\mu_n + V\sum_S \mu_n \leq c\sum_n Q_n(t) + 2VK_{\max}$ (using $\mu_n \leq 1$ and dropping SDPS's nonnegative queue part). Hence Eq. (A.19) holds unconditionally once e_t is *subtracted* from its right-hand side; equivalently, $+e_t$ appears on the right of the master inequality Eq. (A.23), where the drift-plus-penalty objective enters with a negative sign. Since $Q_n(t) \leq Q_n(1) + A_{\max}t$ deterministically,

$$\begin{aligned}
\sum\nolimits_{t\geq 1}\mathbb{E}[e_t] &\leq \sum\nolimits_{t\geq 1}\left(c\sum\nolimits_n Q_n(1) + cNA_{\max}t + 2VK_{\max}\right)2Nt(t+1)^{-\beta}\\
&\triangleq E_{\mathrm{bad}} < \infty \qquad (\beta > 3).
\end{aligned} \tag{A.21}$$

$E_{\mathrm{bad}} = E_{\mathrm{bad}}(V; N, A_{\max}, K_{\max}, c, \beta, \mathbf{Q}(1))$ is finite and affine (not linear) in V. Splitting the series of Eq. (A.21) into its V-free and V-proportional parts makes the affine structure explicit:

$$\begin{aligned}
&E_{\mathrm{bad}}(V) \leq a_E + b_E V,\\
&a_E \triangleq 2Nc\sum\nolimits_{t\geq 1} t(t+1)^{-\beta}\left(\textstyle\sum_n Q_n(1) + NA_{\max}t\right),\\
&b_E \triangleq 4NK_{\max}\sum\nolimits_{t\geq 1} t(t+1)^{-\beta}.
\end{aligned} \tag{A.22}$$

both finite for $\beta > 3$; this split is consumed by Corollary 2 in A.13. This is also the step that fixes the exponent: the queue-weighted charge grows like t, the bad-event probability decays like $t^{1-\beta}$, and summability of $t^{2-\beta}$ requires $\beta > 3$.

A.10 Step 5: W-Step Telescoping with the Drift Certificate

Step 5a: the per-round master inequality. Combining Eqs. (A.9) and (A.19) (with e_t) gives the unconditional per-round master inequality

$$\Delta(t) - V\,\mathbb{E}[U(t) \mid \mathcal{F}_t] \le cB_0 + \alpha_W K_{\max} + c\sum\nolimits_n Q_n(t)\mathbb{E}\left[A_n(t) - S^\star_{n,V}(t)\middle|\mathcal{F}_t\right] - V\mathbb{E}\left[\sum\nolimits_{n\in S^\star_V(t)}\mu_n\middle|\mathcal{F}_t\right] + 2V\mathbb{E}\left[\sum\nolimits_{n\in S(t)} r_n\middle|\mathcal{F}_t\right] + e_t, \tag{A.23}$$

whose queue-weighted term is already in the exact form consumed by the certificate Eq. (A.3); no arrival rate was introduced.

Step 5b: windows and the leftover rounds. The certificate is a window condition, so we apply Eq. (A.23) over blocks of W rounds. Let $H' = W\lfloor H/W\rfloor$ and $r = H - H' < W$, and fix window starts $t_k = (k-1)W + 1$ for $k = 1, \dots, H'/W$. Leftover rounds are handled separately and cheaply: since $0 \le U(t) \le K_{\max}$ and $0 \le \sum_{n\in S^\star_V(t)}\mu_n \le K_{\max}$ pathwise,

$$\sum\nolimits_{t=1}^{H}\left(\sum\nolimits_{S^\star_V(t)}\mu_n - U(t)\right) \le \sum\nolimits_{t=1}^{H'}\left(\sum\nolimits_{S^\star_V(t)}\mu_n - U(t)\right) + rK_{\max}, \tag{A.24}$$

so it suffices to run the window argument on the first H' rounds and add $rK_{\max} \le WK_{\max}$, which after division by H is exactly the $WK_{\max}/H$ term of Theorem 1. (The drift constants are *not* claimed to absorb it.)

Step 5c: within-window queue movement. Within a window, queues move slowly:

$$|Q_n(\tau) - Q_n(t_k)| \le W(A_{\max}+1) \text{ for all } \tau \in [t_k,\, t_k + W). \tag{A.25}$$

Sum Eq. (A.23) in expectation over $\tau \in [t_k, t_k + W)$ and replace $Q_n(\tau)$ by $Q_n(t_k)$ in the queue-weighted term. The replacement error is bounded, using Eq. (A.25) and $|A_n(\tau) - S^\star_{n,V}(\tau)| \le A_{\max} + 1$, by

$$c\sum\nolimits_n\sum\nolimits_{\tau=t_k}^{t_k+W-1}|Q_n(\tau) - Q_n(t_k)||A_n(\tau) - S^\star_{n,V}(\tau)| \le cNW^2(A_{\max}+1)^2 = cB_W. \tag{A.26}$$

Step 5d: the certificate via the tower property. After the replacement, the queue-weighted contribution of the window is

$$c\sum\nolimits_n \mathbb{E}\left[Q_n(t_k)\sum\nolimits_{\tau=t_k}^{t_k+W-1}(A_n(\tau) - S^\star_{n,V}(\tau))\right],$$

and here the tower property does the work: $Q_n(t_k)$ is $\mathcal{F}_{t_k}$-measurable, so conditioning on $\mathcal{F}_{t_k}$ and applying Eq. (A.3) inside the conditional expectation gives

$$c\mathbb{E}\left[\mathbb{E}\left[\sum\nolimits_{n=1}^{N} Q_n(t_k)\sum\nolimits_\tau\left(A_n(\tau) - S^\star_{n,V}(\tau)\right)\middle|\mathcal{F}_{t_k}\right]\right] \le cB_{\text{win}} - cW\epsilon\sum\nolimits_{n=1}^{N}\mathbb{E}[Q_n(t_k)]. \tag{A.27}$$

This is the only place the certificate is used, and it is used exactly once per window—which is why Eq. (A.3) is stated in precisely this weighted, windowed, conditional form.

Step 5e: the window-summed inequality. Collecting Steps 5a–5d, the window-summed master inequality reads

$$\begin{aligned}&\mathbb{E}[L(t_k{+}W)] - \mathbb{E}[L(t_k)] - V\sum\nolimits_{\tau=t_k}^{t_k+W-1}\mathbb{E}[U(\tau)]\\ &\le W(cB_0 + \alpha_W K_{\max}) + c(B_{\text{win}}{+}B_W) - cW\epsilon\sum\nolimits_n\mathbb{E}[Q_n(t_k)]\\ &- V\sum\nolimits_{\tau=t_k}^{t_k+W-1}\mathbb{E}\left[\sum\nolimits_{n\in S^\star_V(\tau)}\mu_n\right] + 2V\sum\nolimits_{\tau=t_k}^{t_k+W-1}\mathbb{E}\left[\sum\nolimits_{n\in S(\tau)} r_n(\tau)\right] + \sum\nolimits_{\tau=t_k}^{t_k+W-1}\mathbb{E}[e_\tau].\end{aligned} \tag{A.28}$$

Step 5f: telescoping. Sum Eq. (A.28) over $k = 1, \dots, H'/W$. Because consecutive windows abut ($t_{k+1} = t_k + W$), the Lyapunov increments telescope:

$$\sum_{k=1}^{H'/W} (\mathbb{E}[L(t_k{+}W)] - \mathbb{E}[L(t_k)]) = \mathbb{E}[L(H'{+}1)] - L(1) \geq -L(1),$$

using $L \geq 0$. For the utility bound, drop the nonpositive queue term $-cW\epsilon \sum_n \mathbb{E}[Q_n(t_k)] \leq 0$ in every window; the per-window constant $W(cB_0 + \alpha_W K_{\max}) + c(B_{\text{win}}{+}B_W)$, accumulated over $H'/W \leq H/W$ windows, totals at most $H\,(cB_0 + \alpha_W K_{\max} + c(B_{\text{win}}{+}B_W)/W) = HB$ by Eq. (8). Adding the leftover term of Eq. (A.24) multiplied by V, and bounding $\sum_{t \leq H} \mathbb{E}[e_t] \leq E_{\text{bad}}$ by Eq. (A.21),

$$\begin{aligned} VHU^\star_{H,V} - V \sum_{t=1}^{H} \mathbb{E}[U(t)] \leq HB + 2V\mathbb{E}\left[\sum_{t=1}^{H} \sum_{n \in S(t)} r_n(t)\right] \\ + VWK_{\max} + E_{\text{bad}} + L(1). \end{aligned} \tag{A.29}$$

A.11 Step 6: Counting Inequality and Conclusion

Step 6a: per-node harmonic sums. Fix a node n and enumerate its admissions. Its first admission (if any) occurs at a round with $H_n(t) = 0$ and contributes $1/\sqrt{H_n(t) \vee 1} = 1$; its h-th admission for $h \geq 2$ occurs at a round with $H_n(t) = h - 1$ and contributes $1/\sqrt{h-1}$. Writing $H_n(H{+}1)$ for node n's total admission count over the horizon,

$$\sum_{t=1}^{H} \frac{\mathbf{1}\{n \in S(t)\}}{\sqrt{H_n(t) \vee 1}} \leq 1 + \sum_{j=1}^{H_n(H+1)-1} \frac{1}{\sqrt{j}} \leq 1 + 2\sqrt{H_n(H{+}1)},$$

using $\sum_{j=1}^{m} j^{-1/2} \leq \int_0^m x^{-1/2}\,dx = 2\sqrt{m}$.

Step 6b: aggregation over nodes. Let $M_H = |\{n : H_n(H{+}1) \geq 1\}| \leq \min\{N, K_{\max}H\}$ be the number of nodes ever admitted; only these contribute the leading "1" of the harmonic sum. Summing the display of Step 6a over admitted nodes and applying Cauchy–Schwarz,

$$\begin{aligned} \sum_{t=1}^{H} \sum_{n \in S(t)} \frac{1}{\sqrt{H_n(t) \vee 1}} &\leq M_H + 2\sqrt{N \sum_n H_n(H{+}1)} \\ &\leq \min\{N, K_{\max}H\} + 2\sqrt{NK_{\max}H} \\ &\leq 3\sqrt{NK_{\max}H}. \end{aligned} \tag{A.30}$$

where $\sum_n H_n(H{+}1) \leq K_{\max}H$ (at most $K_{\max}$ admissions per round) and $\min\{N, K_{\max}H\} \leq \sqrt{N \cdot K_{\max}H}$. This last step is what keeps the learning term a pure square root: no separate N/H term survives.

Step 6c: from counts to radii. For $t \leq H$, $\log(t+1) \leq \log(H+1)$, so $r_n(t) \leq \sqrt{(\beta/2)\log(H{+}1)} \cdot (H_n(t) \vee 1)^{-1/2}$ when $H_n(t) \geq 1$; when $H_n(t) = 0$, $r_n(t) = 1 \leq \sqrt{(\beta/2)\log(H{+}1)}$ as well, since $\beta > 3$ and $\log(H{+}1) \geq \log 2$ make the radicand exceed one. Hence, by Eq. (A.30),

$$\sum_{t=1}^{H} \sum_{n \in S(t)} r_n(t) \leq \sqrt{\tfrac{\beta}{2}\log(H{+}1)} \cdot 3\sqrt{NK_{\max}H} = \tfrac{3}{\sqrt{2}}\sqrt{\beta\, NK_{\max}H \log(H{+}1)}.$$

Step 6d: assembling Theorem 1. Substituting the display of Step 6c into Eq. (A.29) and dividing by VH yields Eq. (9) term by term: the HB term gives B/V; the radius sum gives the learning term with $C_\beta = 3\sqrt{2\beta}$, which depends only on β; the leftover of Eq. (A.24) gives $WK_{\max}/H$; and the remaining constants give $(E_{\text{bad}} + L(1))/(VH)$. ■

A.12 Score-Learning Gap at Fixed V

Theorem 1 bounds the *average* utility gap, and its bound contains a term, B/V, that does not vanish with H. The *cumulative* utility difference $H\left(U^\star_{H,V} - \frac{1}{H}\sum_{t \leq H} \mathbb{E}[U(t)]\right)$ is therefore bounded by

$HB/V+\widetilde{O}(\sqrt{NK_{\max}H})+\ldots$, whose leading term is *linear* in H at fixed V; nothing in our analysis predicts that this cumulative quantity grows like $\sqrt{H\log H}$ at fixed V, and we do not report it as if it did. (The bound is an upper bound, so a sublinear empirical curve is not a contradiction—it is simply not a prediction. Corollary 2 recovers a sublinear cumulative statewise-gap bound by letting V grow with the horizon.)

The quantity that has a square-root bound at every fixed V is the learning error of the ranking score itself. With $c=\alpha_Q/Q_{\mathrm{ref}}$, define the true-mean score of a feasible set and the induced score-learning gap

$$G_t^{\mu}(S)=\sum\nolimits_{n\in S}\left(cQ_n(t)+\alpha_W\tfrac{D_n(t)}{D_{\mathrm{ref}}}+V\mu_n\right),\ R_H^{\mathrm{score}}=\sum\nolimits_{t=1}^{H}\left[G_t^{\mu}(S_V^{\star}(t))-G_t^{\mu}(S(t))\right]. \tag{A.31}$$

The critical positions cancel *unconditionally* (Step 2a of A.3 uses neither μ_n nor $w_n(t)$, so the cancellation Eq. (A.10) holds on $\mathcal{G}_t^c$ as well), leaving two residual selections drawn from the noncritical class. On the good event $\mathcal{G}_t$, SDPS's residual set maximizes the same expression with $w_n(t)$ in place of μ_n (Eq. (A.11)), optimism gives $w_n(t)\ge\mu_n$, and $w_n(t)-\mu_n\le 2r_n(t)$; hence, pathwise on $\mathcal{G}_t$,

$$G_t^{\mu}(S_V^{\star}(t))-G_t^{\mu}(S(t))\le 2V\sum\nolimits_{n\in S(t)}r_n(t),$$

with no waiting-term surcharge, because the waiting term sits inside G_t^{μ}. On $\mathcal{G}_t^c$ the per-round gap is charged pathwise by e_t' below, so that pathwise $R_H^{\mathrm{score}}\le 2V\sum_{t\le H}\sum_{n\in S(t)}r_n(t)+\sum_{t\le H}e_t'$; applying the deterministic counting inequality Eq. (A.30) and taking expectations,

$$\mathbb{E}[R_H^{\mathrm{score}}]\le C_\beta V\sqrt{NK_{\max}H\log(H{+}1)}+E'_{\mathrm{bad}},\ C_\beta=3\sqrt{2\beta}, \tag{A.32}$$

where the bad-event charge is defined explicitly—the score contains the waiting term, which A.5's e_t does not carry—by

$$e_t'=\mathbf{1}_{\mathcal{G}_t^c}[c\textstyle\sum_n Q_n(t)+\alpha_W K_{\max}+2VK_{\max}],\ E'_{\mathrm{bad}}\triangleq\sum_{t\ge1}\mathbb{E}[e_t'].$$

On $\mathcal{G}_t^c$ the three terms bound, respectively, the queue part of G_t^{μ} (at most $c\sum_n Q_n(t)$ over any admissible set), the waiting part (at most $\alpha_W K_{\max}$, since $D_n(t)/D_{\mathrm{ref}}\le 1$ for a non-critical head), and the quality part (at most $2VK_{\max}$, since $\mu_n,w_n\in[0,1]$). As in A.5, $\sum_n Q_n(t)=O(t)$ deterministically and $\Pr(\mathcal{G}_t^c)=O(t^{1-\beta})$, so $E'_{\mathrm{bad}}<\infty$ for $\beta>3$. Thus $\mathbb{E}[R_H^{\mathrm{score}}]/V=\widetilde{O}(\sqrt{NK_{\max}H})$: this score-learning quantity, rather than the fixed-V cumulative utility difference, is normalized by $\sqrt{NK_{\max}H\log H}$ in Figure 5(b). ■

A.13 Horizon-Tuned Statewise Learning Gap

For completeness, we restate the cumulative statewise learning gap defined in the main paper:

$$\mathrm{Reg}_H(V)\triangleq H\left(U_{H,V}^{\star}-\frac{1}{H}\sum\nolimits_{t=1}^{H}\mathbb{E}[U(t)]\right). \tag{A.33}$$

This quantity compares two actions on the trajectory generated by SDPS; it is not policy regret against an independently evolving scheduler.

Corollary 2 (Horizon-tuned statewise learning gap)**.** *Suppose the weighted-drift certificate holds uniformly for $V\in\mathcal{V}\supseteq[1,\sqrt{H}]$ with common $(W,\epsilon,B_{\mathrm{win}})$. If SDPS runs with $V=\sqrt{H}$, then*

$$\mathrm{Reg}_H(\sqrt{H})\le\left(B+C_\beta\sqrt{NK_{\max}\log(H+1)}\right)\sqrt{H}+WK_{\max}+b_E+\frac{a_E+L(1)}{\sqrt{H}}. \tag{A.34}$$

Consequently,

$$\mathrm{Reg}_H(\sqrt{H})=\widetilde{O}(\sqrt{NK_{\max}H}),$$

and the corresponding average backlog is $O(\sqrt{H})$.

Proof. Multiplying the finite-horizon utility bound in the main paper by H gives

$$\mathrm{Reg}_H(V)\le\frac{HB}{V}+C_\beta\sqrt{NK_{\max}H\log(H+1)}+WK_{\max}+\frac{E_{\mathrm{bad}}(V)+L(1)}{V}. \tag{A.35}$$

Uniformity of the certificate makes $(W, \epsilon, B_{\text{win}})$, and hence B, independent of V over the tuning range. Moreover, $E_{\text{bad}}(V) \le a_E + b_E V$. Substituting $V = \sqrt{H}$ into Eq. (A.35) yields Eq. (A.34).

For the backlog, substituting $V = \sqrt{H}$ into the main-paper bound gives

$$\frac{1}{H}\sum_{t=1}^{H}\sum_{n=1}^{N}\mathbb{E}[Q_n(t)] \le \frac{B + \sqrt{H}\,K_{\max}}{c\epsilon} + NWA_{\max} + \frac{C_\beta}{c\epsilon}\sqrt{NK_{\max}\log(H+1)} + \frac{E_{\text{bad}}(\sqrt{H}) + L(1)}{c\epsilon H}.$$

The first term is $O(\sqrt{H})$, the second is independent of H, the learning term is $O(\sqrt{\log H})$, and the final term is $O(1/\sqrt{H})$. Hence, the average backlog is $O(\sqrt{H})$.

Finally, the comparator $\pi^\star_{\sqrt{H}}$ depends on the selected horizon through $V = \sqrt{H}$. The result is therefore a horizon-tuned state-wise comparison, not classical regret against a fixed executable policy. □

B Proof of the Backlog Bound (10) in Theorem 1

Take $H = mW$. Return to Eq. (A.28), retain the negative queue term, bound $V\sum_\tau \mathbb{E}[U(\tau)] \le VWK_{\max}$ per window, and drop the nonpositive benchmark-utility term. Summing over k, telescoping ($\mathbb{E}[L(H+1)] \ge 0$), and applying Eqs. (A.30) and (A.21):

$$cW\epsilon\sum_{k=1}^{m}\sum_{n}\mathbb{E}[Q_n(t_k)] \le H\left(cB_0 + \alpha_W K_{\max} + \tfrac{c(B_{\text{win}}+B_W)}{W} + VK_{\max}\right) + 2V\sqrt{\tfrac{\beta}{2}\log(H+1)}\cdot 3\sqrt{NK_{\max}H} + E_{\text{bad}} + L(1). \tag{B.1}$$

Window starts undersample the full time average only by the within-window movement Eq. (A.25): for $\tau \in [t_k, t_k + W)$ we have $Q_n(\tau) \le Q_n(t_k) + WA_{\max}$, and since $H = mW$, every round belongs to exactly one window,

$$\sum_{t=1}^{H}\sum_{n}\mathbb{E}[Q_n(t)] \le W\sum_{k=1}^{m}\sum_{n}\mathbb{E}[Q_n(t_k)] + NHWA_{\max}. \tag{B.2}$$

Combining Eqs. (B.1)–(B.2) and dividing by H gives Eq. (10) term by term: $H(B + VK_{\max})$ divided by $c\epsilon H$ gives the leading term; the undersampling correction of Eq. (B.2) gives $NWA_{\max}$; the radius sum gives $\frac{C_\beta V}{c\epsilon}\sqrt{NK_{\max}\log(H+1)/H}$ with $C_\beta = 3\sqrt{2\beta}$ as in Theorem 1 (since $6\sqrt{\beta/2} = 3\sqrt{2\beta}$); and the constants of Eq. (A.21) together with the initial Lyapunov value give $\frac{E_{\text{bad}}+L(1)}{c\epsilon H}$, which Theorem 1 states explicitly rather than hiding inside the $\widetilde{O}(\cdot)$, because E_{bad} is affine in V and $L(1)$ carries the initial backlog. For fixed V the time-averaged expected backlog is bounded uniformly in H: the pending-verification queues are strongly stable.

For general H, write $H = H' + r$ with $H' = W\lfloor H/W\rfloor$ and $r < W$. The leftover rounds cannot be charged to a completed window, so we bound them by the deterministic growth $Q_n(t) \le Q_n(1) + A_{\max}t \le Q_n(1) + A_{\max}H$, contributing at most $rN(\max_n Q_n(1) + A_{\max}H) \le WN(\max_n Q_n(1) + A_{\max}H)$ to $\sum_t\sum_n \mathbb{E}[Q_n(t)]$. After dividing by H, the right side of Eq. (10) therefore grows by at most an additional $NWA_{\max} + NW\max_n Q_n(1)/H$. The completed prefix keeps its functional form under the rescaling from H' to H, since

$$\frac{H'}{H}\sqrt{\frac{\log(H'+1)}{H'}} \le \sqrt{\frac{\log(H+1)}{H}},\ \frac{H'}{H}\cdot\frac{1}{H'} = \frac{1}{H},$$

using $H' \le H$; when $H < W$ no window completes and the deterministic leftover bound alone covers the horizon. This is an additive constant, not a vanishing term, which is why Theorem 1 states the clean bound for $H = mW$ and the surcharge explicitly for general H; the $O(V)$ scaling is unaffected. ■

Parameter scaling in V. The following corollary, read off Eqs. (9) and (10), records how the analytical upper bounds scale as V is swept. It is a bound-level statement rather than a regression model imposed on finite-horizon measurements.

Corollary 3 (Parameter scaling in V). *If Assumption 4 holds uniformly over $V \in \mathcal{V}$ with common $(W, \epsilon, B_{\mathrm{win}})$ and fixed $N, K_{\max}, A_{\max}$, sweeping V scales the drift-and-urgency term of Eq. (9) as $O(1/V)$ and the leading backlog terms of Eq. (10) as $(B+VK_{\max})/(c\epsilon)+NWA_{\max} = O(1+V)$; each run is compared with its corresponding reference $\pi^\star_V$.*

Proof. Uniformity keeps $(W, \epsilon, B_{\mathrm{win}})$—hence B, B_W, and $L(1)$—independent of V, while Eq. (A.22) makes $E_{\mathrm{bad}}(V)$ affine; the claim follows from Eqs. (9) and (10). □

Neither bound passes through the origin: the utility bound retains the V-independent learning term and the backlog bound the constant $B/(c\epsilon) + NWA_{\max}$. The empirical sweep in Appendix F.4 therefore reports measured goodput, latency, and backlog without fitting these finite-horizon observations to $a + b/V$ or $a + bV$. Because the reference moves with V, the corollary is a statement about how the *bounds* scale, not a trade-off curve against one fixed optimum (Appendix D.3). Without V-uniformity, Theorem 1 still holds for each fixed V, but their constants may differ across V.

C Proofs of Proposition 1 and Corollaries 1 and 4

Setup. Fix a tagged block j entering the ready set at wall-clock time R_j from node n_j, and fix a sample path on which Eq. (3) holds. Dispatch epochs are governed by the two triggering rules of the SLO subsection: a decision is made immediately when a channel group completes with ready work present, and immediately when the first block arrives at an idle verifier. Consequently, whenever at least one block is ready, consecutive dispatch epochs are at most $T^{\max}_{\mathrm{cyc}}$ apart, where the dispatch cycle covers scheduler decision time, channel launch, and verification; the result return is asynchronous and does not gate the next epoch. At every dispatch epoch, Eq. (7) admits every slot the node-distinct constraint allows, so no epoch leaves ready work unadmitted while a slot is free. By property (P3), j stays exposed until admitted.

Step 1: threshold crossing. Let t_0 be the first dispatch round with $D_j(t_0) \geq D_{\mathrm{crit}}$ (if j is admitted earlier, its exposure-to-completion latency is at most $D_{\mathrm{crit}} + T^{\max}_{\mathrm{cyc}}$, below the bound Eq. (12)). Two cases give Eq. (C.1). If j is exposed while the verifier is idle, it triggers a dispatch immediately, so $D_j = 0$ at that epoch and the claim is trivial. Otherwise j is exposed during an active cycle; at the dispatch round preceding t_0, either j was not yet exposed or $D_j < D_{\mathrm{crit}}$, and consecutive epochs *with work present* are at most $T^{\max}_{\mathrm{cyc}}$ apart, so

$$D_j(t_0) \leq D_{\mathrm{crit}} + T^{\max}_{\mathrm{cyc}}. \tag{C.1}$$

Step 2: EDF forbids critical overtaking. Two facts combine. First, *criticality is a cutoff on exposure time*: at round t,

$$z_n(t) = 1 \iff D_n(t) = T_t - R_n \geq D_{\mathrm{crit}} \iff R_n \leq T_t - D_{\mathrm{crit}},$$

so the critical class at round t is exactly the set of heads exposed at or before the moving cutoff $T_t - D_{\mathrm{crit}}$. Second, *the EDF key is exposure time reversed*: within the critical class SDPS ranks by decreasing $D_n(t) = T_t - R_n$, which for a common T_t is increasing R_n, with ties broken by node index. Hence the critical order is the total order induced by (R_n, n), and this order is *independent of* t: if $(R_m, m) > (R_j, j)$ then at every round at which both are critical, j ranks above m. The order is therefore stable in time, and no critical block can ever overtake one exposed before it.

Two consequences give the count. (i) Only heads with $(R, \cdot) \leq (R_j, j)$—i.e., those already critical, or tied, at t_0—can ever precede j. By property (P3) each node exposes at most one head, so there are at most $N - 1$ such blocks other than j itself. (ii) A node whose critical head is admitted promotes its successor, but by Eq. (1) the successor's exposure timestamp is set *at that departure instant*, which is at or after $t_0 > R_j$; the promoted block therefore ranks below j forever. It is exactly here that the two-case timestamp of Eq. (1) is indispensable: had R been stamped at upload, a node holding a long backlog of early uploads could promote successor after successor, each carrying a stale timestamp earlier than R_j, each legitimately outranking j—and the number of blocks able to precede j would be governed by backlog rather than by N, destroying the bound.

Step 3: draining the criticals ahead. Critical heads outrank all ordinary ones, so each round admits $\min\{K_t, |\mathcal{C}_t|\}$ criticals, in EDF order. Let $C_0 \le N-1$ count the criticals ranked ahead of j at t_0; by Step 2 this set never grows. Suppose at some round fewer than $K_{\max}$ of them remain. Then j is admitted in that round: if $|\mathcal{A}_t| \ge K_{\max}$, work conservation fills all $K_{\max}$ slots and j, ranking behind fewer than $K_{\max}$ criticals, occupies one of them; if $|\mathcal{A}_t| < K_{\max}$, every exposed head is admitted, including j. Since each round removes $K_{\max}$ of the blocks ahead of j while none is added,

$$j \text{ is admitted by round } t_0 + \left\lceil \frac{C_0+1}{K_{\max}} \right\rceil - 1 \le t_0 + \left\lceil \frac{N}{K_{\max}} \right\rceil - 1. \tag{C.2}$$

Step 4: assembling the bound. Each of the at most $\lceil N/K_{\max} \rceil - 1$ rounds between t_0 and admission lasts at most $T_{\text{cyc}}^{\max}$, and j's own verification adds one more. With Eq. (C.1), j completes within

$$D_j(t_0) + (\lceil N/K_{\max} \rceil - 1)\, T_{\text{cyc}}^{\max} + T_{\text{cyc}}^{\max} \le D_{\text{crit}} + (\lceil N/K_{\max} \rceil + 1)\, T_{\text{cyc}}^{\max}$$

of R_j; substituting Eq. (4) gives Eq. (12). No stochastic assumption was used. ■

Proof of Corollary 1. (i) Under the critical-load condition, at t_0 the criticals including j number at most $K_{\max}$, so $C_0 \le K_{\max} - 1$ and Step 3 admits j at t_0 itself; completion by $D_j(t_0) + T_{\text{cyc}}^{\max} \le D_{\text{crit}} + 2T_{\text{cyc}}^{\max} = D_{\text{ref}}$ follows from Eqs. (C.1) and (4). (ii) If $N \le K_{\max}$, then $|\mathcal{A}_t| \le K_{\max}$ always, so Eq. (7) admits every ready block at the first round after it becomes ready, within $T_{\text{cyc}}^{\max}$ of R_j; completion within $2T_{\text{cyc}}^{\max} \le D_{\text{ref}}$ follows from Eq. (3). ■

We additionally state an upload-to-completion bound conditioned on the number of FIFO predecessors.

Corollary 4 (Backlog-conditioned upload-to-completion bound)**.** *On any sample path respecting Eq. (3), a block uploaded behind q pending predecessors at its own node is exposed within $q\, B_{\text{head}}$ of upload and completes within $(q+1)B_{\text{head}}$.*

It covers the phase the exposure-anchored deadline does not: the bound is coarse—each predecessor is charged a full budget—but pathwise, and it needs no arrival rate, which is exactly what Theorem 1 cannot supply.

Proof of Corollary 4. Fix a block uploaded at time U with q FIFO predecessors at its own node, and a sample path respecting Eq. (3). By Eq. (1), the node's head at time U (if any) was exposed no later than U, so by Proposition 1 it completes—and hence departs—by $U + B_{\text{head}}$. When it departs, the next FIFO block is exposed at that instant, and Proposition 1 applies to it in turn, giving departure within another B_{head}. Iterating over the q predecessors, the tagged block is exposed by $U + q\, B_{\text{head}}$; one further application of Proposition 1 gives completion by $U + (q+1)B_{\text{head}}$. The argument uses only Eq. (1) (a successor is exposed exactly when its predecessor departs), FIFO order, and Proposition 1; no stochastic assumption enters. ■

Remark 1. *The $\lceil N/K_{\max} \rceil$ dependence is order-wise tight: if all N nodes' heads turn critical simultaneously, the EDF-last one waits $\lceil N/K_{\max} \rceil$ cycles, so no priority-compliant rule improves the order of the bound. The critical-load condition is sufficient but not necessary for zero violations: e.g., two blocks crossing the threshold simultaneously with full slack D_{crit} under $K_{\max} = 1$ both finish by D_{ref} despite momentarily exceeding the critical count. A tighter—though still only sufficient—runtime test is the per-round demand-bound check on the sorted critical waits, $D_{(i)}(t) + \lceil i/K_{\max} \rceil\, T_{\text{cyc}}^{\max} \le D_{\text{ref}}$ for all i, which accounts for slack but not for blocks that turn critical later in the horizon; we adopt the simpler count condition for its transparency.*

D Discussion of the Assumptions

D.1 Why conditional unbiasedness, and what it excludes.

Assumption 3, in the admission-indexed form of Appendix A.1, requires $\mathbb{E}[X_{n,h} \mid \mathcal{G}_{n,h-1}] = \mu_n$ with respect to the *complete* history up to and including the round-$\tau_{n,h}$ admission decision. Arm-wise stationarity—the statement that node n's rewards come from a fixed distribution—does not imply this under adaptive admission: SDPS chooses *when* to admit node n as a function of its

backlog, waiting time, and current estimate, so if a node's verification quality covaried with its own congestion (say, its drafts degrade precisely when its queue is long, because long queues coincide with long contexts), the reward of the next admitted block would not be unbiased given the state that triggered the admission, and the UCB index would be systematically skewed. What the assumption asserts is the absence of exactly this covariation. It is substantive, but finite traces cannot verify it; the experiments therefore provide only consistency checks. Relaxations—contextual indices keyed on context length or task class, or sliding-window estimators under drift—would require a different adaptive concentration argument and are outside the stationary theorem.

D.2 What the certificate is, and what it is not.

Assumption 4 is the exact quantity consumed in Appendix A.10 and nothing more. Three of its features are deliberate. It is *queue-weighted*: nodes with $Q_n(t) = 0$ contribute nothing, which is necessary, since an empty node cannot be served and no scheduler could manufacture service for it—an unweighted per-node slack condition would be unsatisfiable in this model. It is *windowed*: within a single round a node may lose every channel to critical-priority traffic or to contention, so the drain is required only across W rounds, and W is an analytical window absorbing bursty service, correlated uploads, and critical episodes. It is emphatically *not* the drafting cadence of the slowest node: in the pending-block model a node with $Q_n(t) > 0$ has a head exposed continuously, so mid-drafting unavailability—the phenomenon a cadence-based W would model—does not arise.

What the certificate does *not* license is equally important. It is a statement along SDPS's observed trajectory, so it is not a capacity region, and it does not assert that $\pi^\star_V$ would stabilize a system it drove itself. It is also not verifiable state by state from a finite trace, and $(\epsilon, B_{\mathrm{win}})$ are not jointly identifiable from one: enlarging B_{win} absorbs almost any candidate ϵ. The experiments therefore report only finite-trace scaling checks and treat fitted slopes as descriptive statistics, not estimates of the certificate. Finally, the certificate is regime-dependent: under the closed-loop bound $\sum_n Q_n(t) \le N J_{\max}$ it holds for any $\epsilon > 0$ once B_{win} is taken large enough, so it constrains the trajectory only when arrivals are open.

D.3 The reference is per-round, and it moves with V.

Two properties of $\pi^\star_V$ bound the reading of Theorem 1.

It is a per-round action reference, not an executable scheduler. Its selections are evaluated on SDPS's trajectory and never remove a block. If SDPS leaves some head exposed for several consecutive rounds, $\pi^\star_V$ may select that same head in each of them and receive credit for μ_n each time—something no executable policy could do, since its first service would dequeue the block. This does not invalidate the algebra: Appendix A.5 compares the two selections at a common state, round by round, and never requires the reference to be realizable. But it means Theorem 1 is a *statewise learning comparison*, not policy regret against a scheduler that could reshape the trajectory, and we take care never to describe $\pi^\star_V$ as a stable, causal, or capacity-achieving policy. Making it executable would require a shadow FIFO with coupled uploads, hence a different ready trajectory and a policy-regret or MDP analysis—a substantially different paper.

It is indexed by V. The reference ranks by the same score as SDPS, which contains V; changing V therefore changes the reference's own selections and its utility $U^\star_{H,V}$. Consequently, sweeping V does not trace SDPS's distance from a *fixed* optimum, and Corollary 3 is stated as parameter scaling of the two bounds—each against its own reference—rather than as the classical utility–delay trade-off, whose usual reading presupposes a stationary comparator. Recovering that reading would require a V-free benchmark satisfying the certificate; the optimistic arg-max of Appendix A.3 would still go through against any critical-first rule that is non-idling at dispatch epochs, but the clean interpretation of the learning term—the reference differs from SDPS in exactly one respect—would be lost. We prefer the honest weaker statement to the familiar stronger one. *It is also not the executable SDPS-greedy ablation.* SDPS-greedy removes admitted blocks and evolves its own queues, exposures, and completions, so it has latency and goodput outcomes that $\pi^\star_V$ does not. It replaces optimism by the empirical mean rather than by the true mean; it therefore measures the stationary cost of exploration, whereas $\pi^\star_V$ isolates the statewise cost of unknown means on SDPS's trajectory.

D.4 Timing and Utility-to-Throughput Bridge.

During a verification cycle, edge nodes may continue drafting for other eligible requests, and newly produced blocks may arrive while the currently admitted group remains under verification. Such arrivals are appended to their node-level FIFO queues and become schedulable according to the head-of-line exposure rule at the next applicable dispatch epoch.

Let $T_{\mathrm{ver}}(t)$ be the maximum target-model verification duration among the blocks admitted in round t. Because a verification channel cannot be reused before its assigned block completes, the group duration is governed by the slowest admitted block. Let $T_{\mathrm{cyc}}(t)$ additionally include scheduling and launch overhead. At fixed block length L, assume the active-cycle bounds used in the main paper:

$$0 < T_{\min} \le T_{\mathrm{ver}}(t) \le T_{\mathrm{cyc}}(t) \le T_{\mathrm{cyc}}^{\max} < \frac{1}{2} D_{\mathrm{ref}}. \tag{D.1}$$

For a scheduling policy π, define its verifier-busy-time effective-output rate as

$$\Theta^{\pi} = \liminf_{H\to\infty} \frac{(L+1)\,\mathbb{E}\left[\sum_{t=1}^{H} U^{\pi}(t)\right]}{\mathbb{E}\left[\sum_{t=1}^{H} T_{\mathrm{ver}}(t)\right]}. \tag{D.2}$$

Let

$$\overline{U}^{\pi} = \liminf_{H\to\infty} \frac{1}{H}\mathbb{E}\left[\sum_{t=1}^{H} U^{\pi}(t)\right].$$

Equation (D.1) implies

$$(L+1)\overline{U}^{\pi}/T_{\mathrm{cyc}}^{\max} \le \Theta^{\pi} \le (L+1)\overline{U}^{\pi}/T_{\min}. \tag{D.3}$$

Thus, average per-round utility tracks verifier-busy-time efficiency up to the measured factor $\kappa_T = T_{\mathrm{cyc}}^{\max}/T_{\min}$. The bridge is tightest when profiling shows that κ_T is close to one.

The quantity Θ^{π} excludes intervals in which no block is ready and is therefore not an end-to-end wall-clock objective. Moreover, because admissions influence when requests resume drafting, different policies may induce different future upload trajectories. We consequently make no claim of global wall-clock optimality from Eq. (D.3); wall-clock SLO goodput is evaluated empirically.

E Experimental Protocol and Calibration

Testbed hardware and workload. The central verifier runs Qwen3-8B in FP16 with HuggingFace Transformers on one RTX 5090 (32 GB). At each dispatch it admits at most $K_{\max} = 2$ speculative blocks, pads their heterogeneous contexts, and validates both sequences in one joint target-model forward pass. Thus, $K_{\max} = 2$ is implemented as a real two-sequence verifier batch, not as two model replicas or sequentially calibrated per-sequence execution. Across the three reported repetitions, 26,962 of 27,000 policy dispatches use batch size two; the remaining 38 occur during startup or transient ready-set shortages. Each edge node maintains up to four concurrent requests, all using a common fixed prompt and a 64-token generation budget.

Six commodity edge devices on the same LAN run three Qwen3-1.7B and three Qwen3-0.6B draft models over HTTP. Their mean $L = 8$ draft times range from approximately 0.18 to 1.36 s. Each node maintains $J_{\max} = 4$ active tasks, and requests are truncated at 64 committed tokens. Node-specific sampling temperatures range from 0.35 to 1.15 and reverse the quality ranking at round 500. All upload, exposure, dispatch, batch-completion, model, temperature, and device metadata are recorded and joined by block identifier. Pending blocks at a policy boundary are explicitly cancelled, the draft sampling RNG is reset, and no block is carried into the next policy.

Verification-bottlenecked operating point. Native target-model batching remains faster than the heterogeneous edge drafting pipeline, so the raw verifier is not persistently backlogged. We therefore add a controlled 300 ms delay once per target-model batch to emulate a heavier shared verifier while preserving the measured native batched-forward time. The delay is not multiplied by batch size. Across all reported policy rounds, the resulting mean real batch-verification time is 372.8 ms. The

controlled sensitivity sweep in Appendix F.6 separately identifies the raw-like, contention-limited, and infeasible operating regimes.

Cycle and SLO calibration. Each repetition begins with a 50-round work-conserving warmup that fixes a shared configuration for all policies. The calibrated $T_{\mathrm{cyc}}^{\max}$ ranges from 455.4 to 462.0 ms, D_{ref} from 1639.2 to 1660.3 ms, and B_{head} from 2550.0 to 2584.2 ms. We use $V = 5$, $\alpha_Q = \alpha_W = 1$, and $\beta = 4$, with Q_{ref} determined from the warmup backlog. Each policy runs for $1{,}000$ dispatch rounds, with node temperatures changing at round 500 to reverse the quality ranking. Across repetitions, 18.8%–56.3% of admitted SDPS heads are critical, exercising both ordinary yield-aware admission and strict critical-first protection.

Baselines and ablations. Upload-FIFO orders exposed heads by upload time; Exposure-EDF orders them by exposure time, equivalent to EDF under a common relative deadline. MaxWeight selects the largest pending queues, whereas UCB-only uses only the learned-yield UCB without critical-first protection. Random uniformly samples up to $K_{\max}$ nodes from the ready set. Sync-round is a SLED-inspired synchronized variant (Li et al., 2025b): it waits for one exposed head from every live node, then serves the cohort in $K_{\max}$-sized waves. SDPS-greedy replaces the UCB index with the empirical mean, using 1 for unseen nodes. SDPS-no-quality sets the entire quality term to zero. Both retain the queue, waiting, and critical-first components; acceptance feedback is still recorded for analysis.

Metrics and aggregation. The primary metric is exposure-anchored SLO goodput: committed tokens from blocks meeting the exposure deadline per elapsed second. We additionally report exposure-deadline violations, exposure-to-completion P99 latency, accepted draft tokens per block, total backlog, and post-reversal service-share changes. Accepted-token yield excludes target correction/bonus tokens, whereas goodput includes them when emitted. Backlog is the total pending-block count sampled at dispatch epochs and averaged over rounds, not a time-weighted average. Aggregate testbed results are arithmetic means of the three run-level measurements, with two-sided 95% Student-t confidence intervals.

Controlled simulation. The simulator invokes the production admission-rule implementation and replaces only the physical layer. Rewards follow truncated-geometric block acceptance with known mean, node drafting delays are configurable, and verification durations lie in a known interval. Unless otherwise noted, $N = 6$, $K_{\max} = 2$, and $L = 8$. The learning-horizon and reward-reversal experiments use five seeds; the component ablation uses five seeds and $10{,}000$ rounds per variant; the V-sweep uses ten seeds and $10{,}000$ rounds per setting; and the load sweep uses five seeds and $6{,}000$ rounds per policy–load pair. Operational summaries discard the first 10% of rounds. In the drift experiment, all node-quality rankings reverse at round $3{,}000$; the sliding-window estimator uses the most recent 300 observed rewards per node.

F Supplementary Experimental Results

F.1 Testbed Goodput and Delay Diagnostics

Across the three real batch-2 repetitions, SDPS obtains 15.105 ± 1.614 tok/s SLO goodput, a $0.033 \pm 0.143\%$ violation rate, and 1474 ± 6 ms exposure-to-completion P99 latency. Its goodput confidence intervals overlap substantially with those of SDPS-greedy, Exposure-EDF, and SDPS-no-quality. By contrast, UCB-only, Upload-FIFO, Random, MaxWeight, and Sync-round incur mean violation rates between 9.29% and 23.35%. All $5{,}996$ completed SDPS blocks remain below their repetition-specific B_{head} values. The only two empirical SLO violations occur in the one repetition whose complete sufficient-condition audit is false; the other two repetitions satisfy the audit and have no violations.

SDPS obtains 1.825 ± 0.082 accepted draft tokens per block, compared with 1.801 ± 0.056 for SDPS-no-quality, 1.788 ± 0.041 for Exposure-EDF, and 1.786 ± 0.165 for SDPS-greedy. The confidence intervals overlap because only three independent repetitions are available, but the paired direction is consistent: SDPS exceeds all three comparison policies in every repetition. This result supports a consistent productive-yield improvement rather than a claim of statistically significant raw-goodput superiority.

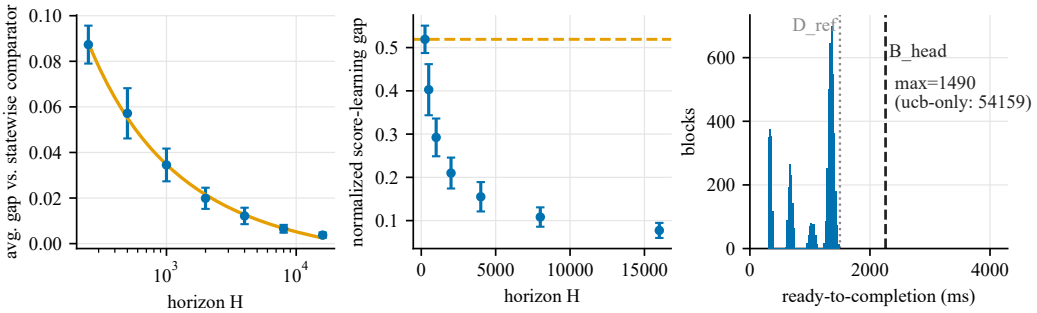


Figure 5: Controlled simulation (5 seeds, mean $\pm$ standard deviation). (a) The average statewise utility gap decays with horizon. (b) The correctly recomputed cumulative score-learning gap, divided by $\sqrt{NK_{\max}H\log H}$, remains bounded and decreases with horizon; the dashed line is the smallest constant envelope over the measured points. (c) Under deliberate overload, SDPS respects the pathwise head-delay bound, whereas UCB-only does not.

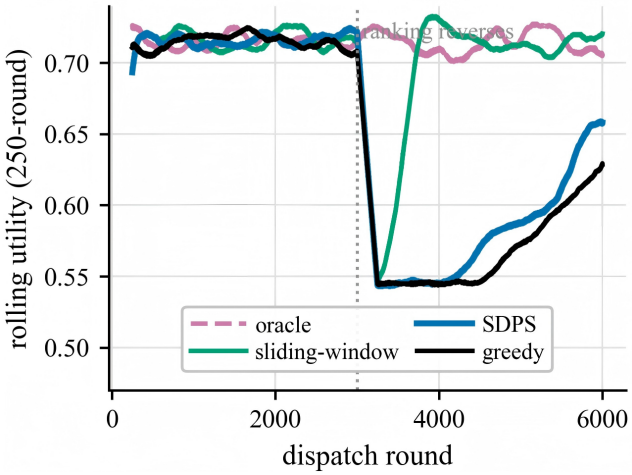


Figure 6: Controlled reward reversal over five seeds. The dotted line marks the round-3,000 reversal; curves show the five-seed mean utility smoothed over 250 rounds. The oracle knows the current μ_t, while sliding-window UCB uses the latest 300 observed rewards per node.

The no-quality ablation also clarifies the real drift mechanism. SDPS shifts $+2.47$ percentage points of service toward the improved nodes after the reversal, whereas SDPS-no-quality shifts -1.43 points. SDPS-greedy shifts $+2.61$ points, showing that the current testbed does not isolate an additional adaptation advantage from stationary UCB over the empirical mean. UCB-only shifts $+22.23$ points but incurs a 9.29% violation rate, demonstrating the SLO cost of unconstrained quality optimization.

F.2 Learning, Delay, and Nonstationary Adaptation

At fixed $V = 5$, the average statewise utility gap falls from 0.0873 at $H = 250$ to 0.00374 at $H = 16{,}000$, following the finite-horizon learning shape predicted by Theorem 1. The cumulative score-learning gap itself grows only from 66.8 to 105.4 over the same $64\times$ horizon increase; consequently, the normalized quantity in panel (b) decreases from 0.519 to 0.077. This is consistent with the derived sublinear upper order, but neither verifies the theorem's assumptions nor constitutes classical policy regret. For SLO protection, a light-load run satisfying the critical-load condition has zero violations over $3{,}000$ rounds. When the condition is first exceeded, the violation rate becomes 0.33%. Under deliberate overload, the unconditional pathwise bound continues to hold for SDPS, while UCB-only fails because it lacks the strict priority class.

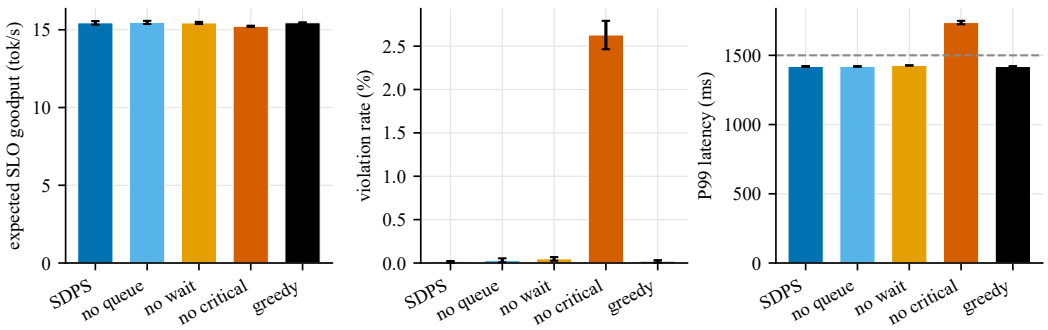


Figure 7: One-component-at-a-time ablations at offered load $\rho = 0.95$ (5 seeds, 10,000 rounds per variant, mean $\pm$ standard deviation). "No queue" sets $\alpha_Q = 0$; "no wait" sets $\alpha_W = 0$ for ordinary heads while retaining the critical class; "no critical" retains the complete ordinary score but removes strict critical-first promotion.

Figure 6 evaluates recovery when all node-quality rankings reverse midway through the run. Over the post-change half, the dynamic true-mean oracle obtains normalized utility 0.714 ± 0.004, while sliding-window UCB reaches 0.693 ± 0.005, compared with 0.582 ± 0.005 for stationary UCB and 0.569 ± 0.007 for greedy scheduling (mean $\pm$ standard deviation over five seeds). Sliding-window UCB therefore improves post-change utility by 19.0% over stationary UCB and 21.7% over greedy, while remaining within 3.0% of the oracle. The comparison shows that discarding stale observations is important after a persistent ranking change, whereas full-history UCB and greedy estimation recover substantially more slowly. This sliding-window variant is an empirical nonstationary extension; the guarantees in the main paper assume stationary reward means.

F.3 Component Ablations

Figure 7 isolates the mechanism responsible for SLO protection. Removing strict critical-first promotion raises the violation rate from 0.014% to 2.63%, increases P99 exposure-to-completion latency from 1419 to 1737 ms, and lowers expected SLO goodput by 1.44%. Removing only the ordinary queue or waiting term changes goodput by less than 0.2% in this stationary near-capacity regime, while the greedy variant remains statistically close to full SDPS. Thus strict priority drives the safety gain here; queue and ordinary-wait terms are robustness controls whose marginal value is workload-dependent rather than uniformly positive.

F.4 Parameter and Load Robustness

Across the $40\times$ range $V = 0.5$–20, expected SLO goodput stays within 15.35–15.41 tok/s; $V = 5$ has the highest mean, while mean P99 grows moderately from 1371 to 1426 ms and remains below $D_{\text{ref}} = 1500$ ms. This supports parameter robustness, not a claim that the empirical trajectory realizes the asymptotic $O(1/V)$–$O(V)$ bounds. The load sweep uses the open-arrival regime of the system model, in which $J_{\max}$ is not enforced and backlog is not structurally bounded. Under low load, all policies are nearly indistinguishable. At $\rho = 0.95$, SDPS achieves 15.38 tok/s with 0.025% violations, whereas MaxWeight and UCB-only incur 9.51% and 4.13% violations and P99 delays above 3.1 s. At $\rho = 1.05$, SDPS continues to protect exposed heads, but its mean backlog reaches 327 blocks; this super-capacity point is a finite-horizon stress test and does not imply queue stability.

F.5 Certificate Diagnostics

Assumption 4 is not identifiable from a finite trace: increasing B_{win} can absorb many candidate values of ϵ. Likewise, truncating requests and narrowing context variation make Assumption 3 more plausible but do not prove conditional unbiasedness. These experiments check consistency with the predicted scaling; the assumptions remain explicit analytical conditions, as discussed in Appendix D.

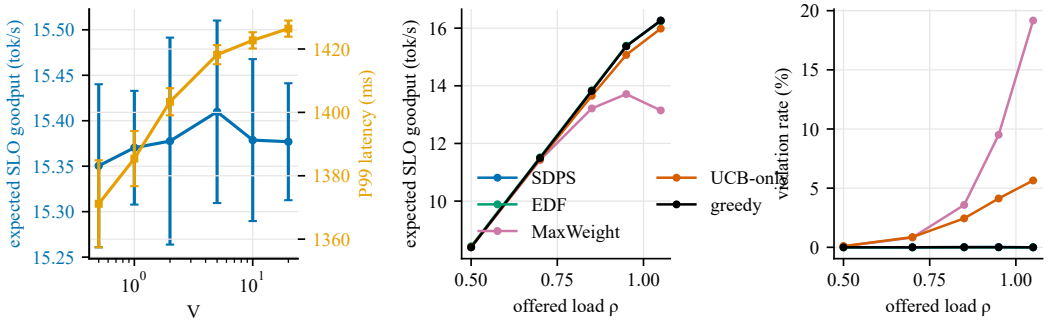


Figure 8: Parameter and offered-load robustness. (a) $V \in \{0.5, 1, 2, 5, 10, 20\}$ at $\rho = 0.95$ (10 seeds, 10,000 rounds per value). (b–c) Offered-load sweep (5 seeds, 6,000 rounds per policy–load pair). Error bars in (a) are standard deviations; the load-sweep curves show seed means. Goodput is simulator expected committed tokens completed within the exposure SLO per simulated second.

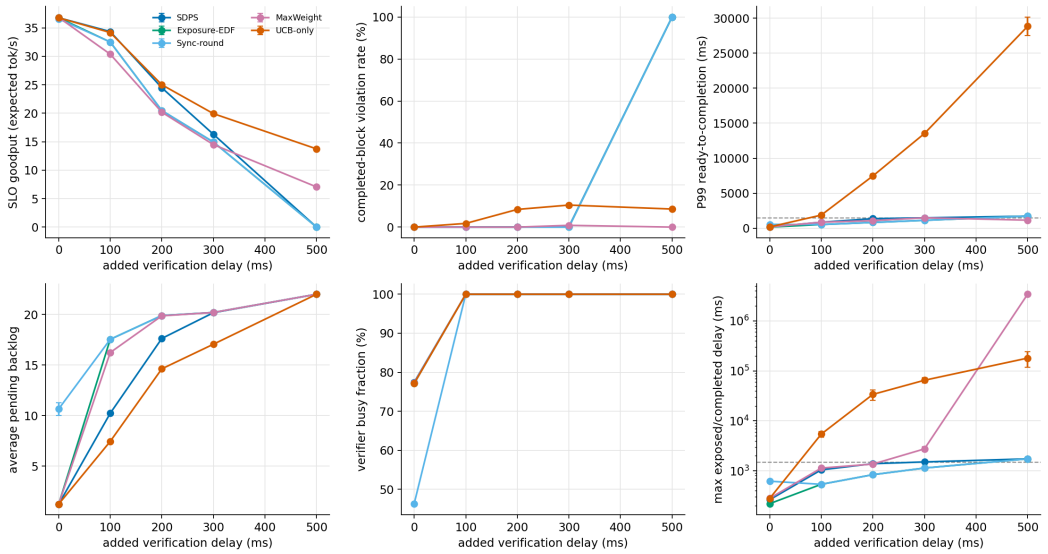


Figure 9: Controlled verification-delay sweep over five seeds, with 6,000 dispatch rounds per policy–delay pair. The added delay ranges from a simulated raw-like point to an intentionally infeasible stress point. Completed-block violation and P99 metrics are reported together with backlog, verifier utilization, and the starvation-aware maximum delay, preventing unfinished old heads from appearing artificially safe. Error bars show one standard deviation across seeds.

F.6 Verification-Bottleneck Sensitivity

In controlled simulation with five seeds and 6,000 dispatch rounds per policy–delay pair, the simulated raw-like point exhibits limited verification contention. At $d_{\text{add}} = 0$, SDPS and Exposure-EDF use approximately 77% of the verifier capacity and achieve nearly identical SLO goodput of 36.787 and 36.777 tok/s, respectively. Sync-round achieves 36.584 tok/s but uses only 46.3% of the verifier capacity because it waits for the readiness barrier. Thus, sophisticated scheduling has little room to improve productive output when verification is not the persistent bottleneck.

At $d_{\text{add}} = 200$ ms, the verifier becomes persistently saturated. SDPS achieves 24.474 tok/s, compared with 20.475 tok/s for Exposure-EDF and 20.477 tok/s for Sync-round, corresponding to SLO-goodput improvements of 19.5% over both baselines. All three policies incur zero violations at this point. SDPS uses more of the available SLO slack, with a P99 exposure-to-completion latency of 1373 ms, while still remaining below the fixed $D_{\text{ref}} = 1500$ ms deadline.

At $d_{\mathrm{add}} = 300\,\mathrm{ms}$, the system operates closer to the SLO boundary. SDPS achieves 16.303 tok/s, compared with 14.941 tok/s for Exposure-EDF and 14.942 tok/s for Sync-round, retaining a goodput advantage of approximately 9.1%. Its P99 latency reaches $1494\,\mathrm{ms}$ and its violation rate increases slightly to 0.211%, showing that nearly all of the available SLO slack is being used. Exposure-EDF and Sync-round remain violation-free but obtain lower productive output.

The $d_{\mathrm{add}} = 500\,\mathrm{ms}$ setting is an intentionally infeasible stress point rather than a normal operating regime. Under persistent six-node contention, SDPS, Exposure-EDF, and Sync-round all have exposure-to-completion latency of approximately $1732\,\mathrm{ms}$, exceeding the fixed $1500\,\mathrm{ms}$ SLO and yielding zero SLO goodput. This point does not indicate a failure of critical-first scheduling; instead, it exposes the feasibility boundary at which the service interval imposed by six persistently backlogged nodes and two verification slots exceeds the deadline. Across the sweep, SDPS is therefore most useful in the intermediate regime where verification contention is sustained but sufficient latency slack remains for yield-aware ordinary scheduling.