SLOPE: DECISION-ALIGNED MARGINAL RESOURCE ALLOCATION FOR CO-LOCATED LLM SERVING

Anonymous authors
Paper under double-blind review

ABSTRACT

Joint TTFT/E2E violations are terminal outcomes, but an online LLM scheduler must choose variable-duration actions before future arrivals and generation demand are known. Starting from the same latency allowances as the terminal objective, SLOPE jointly designs service valuation and execution-cost learning for online resource allocation. Its central idea is to translate terminal SLO risk into a computable local service comparison: an SLO-normalized potential unifies input progress, phase completion, and decode service, while analysis of the variable-duration service process yields potential removal per unit time as the allocation criterion. SLOPE then identifies marginal resource effects under a common request plan, making candidate comparisons determine both how service value is assessed and which execution-cost differences must be learned. Theoretical analysis establishes a pathwise upper bound on joint violations and a selection-quality guarantee under relative-cost error. Across three model/workload pairs and a $10\times$–$30\times$ sweep of system-normalized TTFT/E2E thresholds, SLOPE improves joint-SLO performance across the evaluated SLO regimes. At the common $20\times$ reference point, it reduces load-averaged joint violations by **15.5%–30.1% relatively** against the best evaluated baseline at each load, with scheduling-call p95 below **2 ms** on Llama 70B. Real-GPU trace analysis and controlled replay further show that planned service can be realized during execution and that cost-aware local choices can yield more jointly successful requests under the same continuation policy, providing empirical mechanism evidence for the allocation criterion.

1 INTRODUCTION

Co-located LLM serving interleaves input processing with token-by-token generation (Yu et al., 2022). Larger prefill chunks improve input-processing efficiency but lengthen shared iterations and delay ongoing decode requests (Zhong et al., 2024); smaller chunks preserve decode cadence but increase chunking and waiting overhead (Agrawal et al., 2024). The scheduler therefore has to allocate prefill resources while the evaluation objective is terminal: the same request must satisfy both time-to-first-token (TTFT) and end-to-end (E2E) deadlines after an entire serving trajectory. At a current boundary, future arrivals, final output lengths, later actions, and even action duration are not fully known. The central question is thus not merely how to write a throughput–latency trade-off, but how to construct a computable local comparison that remains tied to terminal joint-SLO risk.

SLOPE derives that comparison rather than postulating a generic utility-per-time heuristic. It starts from the weight-free occupancy $R_{\mathrm{slo}}(u) = n^P(u)/d^T + n^D(u)/d^E$, whose area upper-bounds the fraction of requests violating either deadline under explicit conditions. Because this binary signal is too sparse for token-level control, SLOPE upper-envelopes it with a dense SLO-normalized potential and, for variable-duration frames, uses the rate at which an action removes that potential. The familiar $N/\widehat{T}$ rule appears only at the end of this chain. Section 3.1 makes the derivation explicit.

Two additional design choices make this rate meaningful. First, SLOPE separates request planning from resource investment. If changing a budget also changes request eligibility or order, an endpoint difference confounds the direct resource effect with plan substitution; Section 3.2 shows that the former is not identified from such coupled endpoints. SLOPE therefore freezes the plan and varies only the prefill investment.

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.

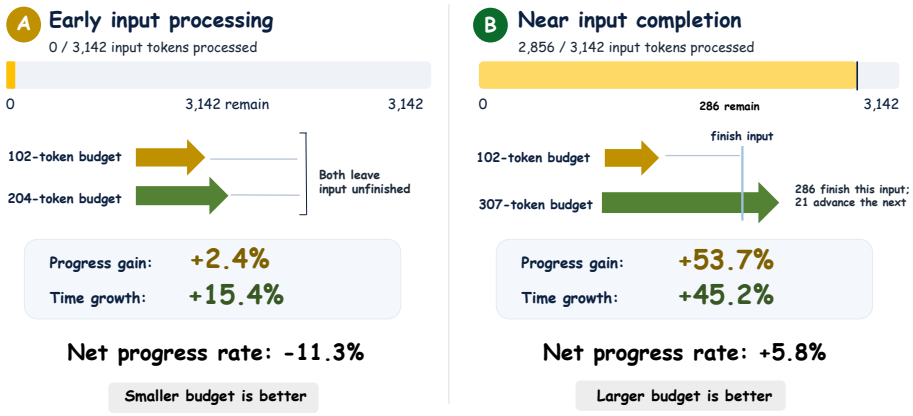


Figure 1: **Stage-dependent value of additional prefill.** For the same request, larger budgets lower the progress rate early (A) but raise it near input completion (B). Rates use Eq. (12), default weights (Appendix C.4), and the mean measured time over three repeats.

Fig. 1 illustrates why the service side of this control problem is hybrid. Early in input processing, additional budget mainly removes continuous unfinished work. Near input completion, a similar increment crosses a phase boundary and removes an entire unfinished input milestone, while also changing how much decode service fits in the shared iteration. A token-count objective cannot distinguish these state transitions.

Second, the numerator is not a post hoc feature sum. Under an additive, symmetric hybrid-state model, Section 3.3 shows that the primitive local coordinates are exactly decode coverage, continuous prefill work, and a discrete prefill-completion jump. Their positive coefficients are not terminal-success probabilities: after removing common scale,

$$\lambda = \frac{w_P}{w_F}, \qquad d_0 = \frac{w_D}{w_F}\frac{d^E}{d^T}, \tag{1}$$

where λ is certificate-preserving dense-prefill shaping and d_0 is a reference decode concurrency. The cost model follows the same identified intervention: ELP is trained on matched same-state log-time contrasts in addition to absolute latency, because those contrasts directly determine the allocation margin.

This positioning differs from prior chunked-execution and SLO-aware schedulers, which adapt batching, request priority, or prefill chunks using throughput, demand, payoff, or deadline information (Agrawal et al., 2024; Hong et al., 2025; Chen et al., 2025; Zhang et al., 2026; Goel et al., 2026). It also differs from generic predict-then-optimize objectives (Elmachtoub & Grigas, 2022; Wilder et al., 2019): here the common-plan construction first identifies the resource contrast, which then determines both the service comparison and the cost supervision. Appendix A gives the fuller comparison.

The empirical evaluation tests both the complete policy and the observable bridge supporting its interpretation. Across Hunyuan-A13B/SCROLLS, Llama-3.3-70B/RAGBench, and Qwen3-14B/LMSYS, SLOPE reduces load-averaged $20\times$ joint violations by **30.1%, 15.5%, and 23.3% relatively**, respectively, against the best evaluated baseline at each load. More importantly for the local-to-terminal argument, three real execution traces containing 329 mixed decisions have an equal-seed backlog-weighted planned-to-realized residual of $3.34\% \pm 2.64\%$; all 258 audited decisions with $n_t^D \geq d_0$ have zero sample residual. In six pre-specified saved states where SLOPE and progress-only select different first actions, the SLOPE-selected action yields higher/tied terminal joint success in **5/1** states under the same subsequent SLOPE policy. These audits are mechanism evidence, not estimates of Bellman regret; the stronger conditional candidate-class result is confined to Appendix B.10.

In summary, SLOPE contributes an SLO-calibrated derivation of local potential-removal rate; a common-plan construction that makes marginal resource effects identifiable; a structural characterization of the hybrid service numerator together with the (λ, d_0) weight interpretation; decision-aligned relative-cost learning for the matching duration contrast; and real-GPU audits connecting planned service, realized service, and terminal consequences of local allocation disagreements.

2 PROBLEM SETUP

We consider co-located serving in which prefill and decode share GPU execution and KV-cache capacity. Prefill can advance a request through successive input chunks, while autoregressive decode advances each served request by at most one new output token per iteration. A mixed iteration can serve both phases, coupling their progress through its execution time and resource occupancy.

Request i arrives at a_i with input length p_i and unknown final output length L_i. Let F_i and C_i be first-token delivery and completion times. TTFT and E2E are both measured from arrival, so E2E includes queueing, prefill, and generation. Input-phase completion is an engine event; F_i records client delivery and may occur later. With TTFT/E2E allowances $d_i^T, d_i^E > 0$, define request-level joint attainment

$$J_i(\pi) = \mathbf{1}\left[F_i(\pi) - a_i \le d_i^T \ \wedge \ C_i(\pi) - a_i \le d_i^E\right], \tag{2}$$

with failed or rejected requests assigned $J_i = 0$ and retained in the denominator.

At iteration t, observable history $\mathcal{H}_t$ contains arrived requests, computed input and generated-output progress, queues, resource state, and execution feedback; future arrivals and final output lengths are unavailable. Let s_t denote the current request and resource state available through $\mathcal{H}_t$. Predictions may use observed inputs and offline-trained models. An action $z_t \in \mathcal{A}(s_t)$ chooses requests and integer computation quotas subject to engine token, sequence, physical KV-block, and phase-dependency constraints (Appendix C.5). Action duration depends on state and action, so a resource choice changes both current service and the wall-clock time before the next decision.

For an externally specified request set $\mathcal{R}$ of size $n > 0$, the evaluation objective is

$$\max_{\pi \in \Pi_{\text{online}}} \ \mathbb{E}_{\omega \sim \mathcal{W}}\left[\frac{1}{n}\sum_{i \in \mathcal{R}} J_i(\pi;\omega)\right], \qquad z_t = \pi(\mathcal{H}_t) \in \mathcal{A}(s_t). \tag{3}$$

Here Π_{online} contains policies using only observable history, and $\mathcal{W}$ describes workload and execution randomness under fixed model and generation settings. The evaluation set is independent of admission decisions and need not be known to the scheduler in advance. The objective's current-action continuation value is not observable online because it depends on hidden generation demand, future arrivals, execution randomness, and subsequent scheduling. SLOPE therefore evaluates complete policies by Eq. (3) while using the derived local controller below; it does not assume one-step progress equals terminal reward.

3 SLOPE

Fig. 2 summarizes the design. SLOPE first derives the quantity that a variable-duration local action should improve from terminal joint-SLO risk, then constructs matched-plan budget interventions so that this resource effect is identified. The service numerator and the execution-time learner are designed around the two contrasts exposed by that construction.

3.1 FROM JOINT-SLO RISK TO LOCAL REMOVAL RATE

For clarity, first consider common TTFT/E2E allowances d^T, d^E; Appendix C.5 discusses the general evaluation setting. Let $I_i^P(u)$ and $I_i^D(u)$ indicate unfinished input and output phases. The weight-free SLO-risk occupancy is

$$R_{\text{slo}}(u) = \sum_{i:a_i \le u}\left[\frac{I_i^P(u)}{d^T} + \frac{I_i^D(u)}{d^E}\right] = \frac{n^P(u)}{d^T} + \frac{n^D(u)}{d^E}. \tag{4}$$

For a finite request set completing under Appendix B.6's conditions, Theorem B.10 gives the pathwise certificate

$$v_n \le \min\left\{1, \frac{1}{n}\int_0^H R_{\text{slo}}(u)\,du + \overline{h}_n\right\}, \tag{5}$$

where v_n is the fraction violating either deadline and $\overline{h}_n$ accounts for engine-to-client boundary delays. No progress-score weights enter Eq. (5).

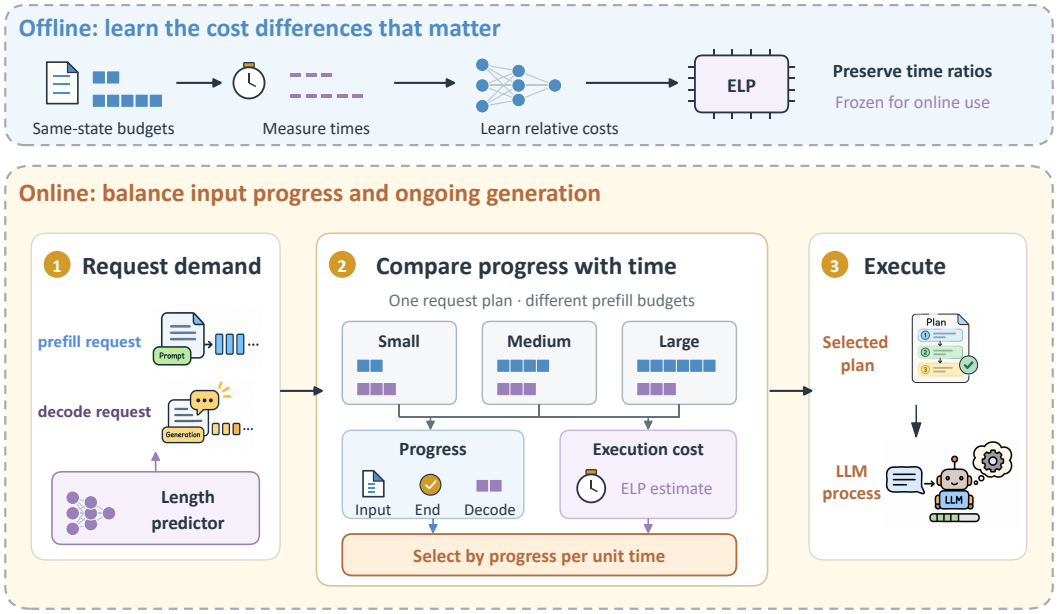


Figure 2: **SLOPE framework.** A request plan is frozen before candidate budgets are compared. Online selection uses the characterized two-phase service decrement per predicted frame time; offline ELP learns the same-state duration contrasts induced by those budget interventions.

This occupancy changes only at phase/request completion, so SLOPE densifies it without invalidating the certificate. With remaining first-time input work r_i^P and remaining output work r_i^D,

$$\overline{Q}_\lambda(u) = \sum_{i:a_i \le u} \left[\frac{I_i^P(u)}{d^T} + \frac{r_i^D(u)}{d^E} + \lambda \frac{r_i^P(u)}{Bd^T} \right], \qquad \lambda \ge 0. \tag{6}$$

Because $r_i^D \ge I_i^D$ and the shaping term is nonnegative, $\overline{Q}_\lambda \ge R_{\text{slo}}$ pointwise; its area therefore inherits Eq. (5)'s upper-bound role. In a variable-duration frame, actual decode, useful prefill, and phase-completion service (D_k, P_k, F_k) remove $\overline{S}_k = D_k/d^E + \lambda P_k/(Bd^T) + F_k/d^T$. The action-dependent term in normalized quadratic drift is $-\overline{Q}_k\, \overline{\mu}_k(a)$ with

$$\overline{\mu}_k(a) = \frac{\mathbb{E}_k^a[\overline{S}_k]}{T_k(a)}. \tag{7}$$

Since $\overline{Q}_k$ is fixed at the boundary, faster potential removal improves the controllable service term; the division by wall-clock time follows from variable frame duration, not from assuming a generic utility/time objective. Appendix B.7 gives the arrival and second-order terms.

Mapping the runtime weights to SLO units. The online numerator uses positive (w_D, w_P, w_F). Removing common scale leaves

$$\lambda = \frac{w_P}{w_F}, \qquad d_0 = \frac{w_D}{w_F} \frac{d^E}{d^T}. \tag{8}$$

With $d_t^D(\rho)$ planned decode advances, n_t^D the common waiting-decode count, useful prefill work $W_t(\rho)$, and phase-completion marker $f_t(\rho)$, the SLO-normalized planned decrement is

$$\overline{N}_t(\rho) = \frac{N_t(\rho)}{w_F d^T} = \frac{d_0}{n_t^D} \frac{d_t^D(\rho)}{d^E} + \lambda \frac{W_t(\rho)}{Bd^T} + \frac{f_t(\rho)}{d^T}. \tag{9}$$

Thus λ is exactly certificate-preserving dense-prefill shaping, while d_0 is the reference decode concurrency at which planned coverage is expressed in the same E2E-risk units as generation work. They are not terminal-success probabilities. The planned rate approximates realized SLO-normalized removal with an explicit residual:

$$\overline{\mu}_t(\rho) \ge \frac{\overline{N}_t(\rho)}{T_t(\rho)} - \overline{\xi}_t(\rho). \tag{10}$$

Appendix B.5 decomposes $\bar{\xi}_t$ into reference-concurrency normalization and plan-to-execution discrepancies. In particular, if planned decode coverage executes, normalization can overstate E2E-calibrated decode service only when $n_t^D < d_0$. Section 4.5 audits these terms directly.

3.2 Plan-Conditioned Marginal Identification

The rate derivation specifies which resource effect matters, but an arbitrary candidate comparison need not identify it. Let P collect request eligibility, ordering, aging state, and resource context, and let $Y(P, \rho)$ be a candidate outcome such as planned service or log duration. If budgets ρ_a, ρ_b also induce different plans P_a, P_b,

$$Y(P_b, \rho_b) - Y(P_a, \rho_a) = \underbrace{Y(P_a, \rho_b) - Y(P_a, \rho_a)}_{\text{resource effect at fixed } P_a} + \underbrace{Y(P_b, \rho_b) - Y(P_a, \rho_b)}_{\text{plan-substitution effect}}. \tag{11}$$

Only the first term asks whether additional resources are worthwhile for the same service obligations. From the two coupled endpoints, it is not identified: Proposition B.1 constructs arbitrary direct resource contrasts consistent with the same endpoints.

SLOPE therefore freezes P_t after native admission/ordering and varies only prefill investment $b(\rho) = \lfloor B\rho \rfloor$; identical ordered quota plans are deduplicated and empty plans removed, yielding Ω_t. Each candidate is thus a matched intervention (P_t, ρ). For planned decrement $N_t(\rho) > 0$ and predicted frame time $\widehat{T}_t(\rho) > 0$, SLOPE selects by

$$U_t(\rho) = \frac{N_t(\rho)}{\widehat{T}_t(\rho)}, \tag{12}$$

and the pairwise boundary is

$$m_t(\rho_a, \rho_b) = \log \frac{N_t(\rho_b)}{N_t(\rho_a)} - \log \frac{\widehat{T}_t(\rho_b)}{\widehat{T}_t(\rho_a)}. \tag{13}$$

The ratio is therefore the scalar form of an identified service/duration resource contrast, not the primitive assumption.

3.3 Characterizing the Hybrid Service Numerator

At the frozen boundary, a budget intervention can change three local service coordinates: pending next-step decode obligations, continuous unfinished prefill work, and a persistent resident prefill milestone whose completion changes phase. We represent the outstanding service in these coordinates by a local measure Φ_t. If this measure is separable across the three components, symmetric and additive over decode obligations, additive and regular over continuous prefill work, and assigns a fixed contribution to the unfinished resident prefill milestone, Proposition B.2 implies

$$\Phi_t(d, \mathbf{w}, I) = \alpha_t d + \beta \sum_i w_i + \gamma I. \tag{14}$$

For candidate ρ, define decode coverage $\eta_t^D(\rho) = d_t^D(\rho)/n_t^D$, useful first-time prefill work $W_t(\rho) = \sum_i \min\{q_{i,t}(\rho), w_{i,t}\}$, and $f_t(\rho) \in \{0, 1\}$ indicating completion of the fixed highest-ranked resident prefill milestone. Choosing $\alpha_t = w_D/n_t^D$, $\beta = w_P/B$, and $\gamma = w_F$, the planned reduction in this measure is exactly

$$N_t(\rho) = w_D \eta_t^D(\rho) + w_P \frac{W_t(\rho)}{B} + w_F f_t(\rho). \tag{15}$$

The first two terms quantify decode coverage and continuous prefill progress; f_t captures the discrete phase-completion jump. Appendix B.3 gives the complete definitions, construction of the measure before and after the planned action, and proof. Output-length prediction affects the frozen request plan, not the identity in Eq. (15); implementation and feasibility checks are in Appendix C.2.

3.4 Contrast-Aligned Execution-Time Learning

The common-plan construction also identifies the cost information needed by the decision. For a fixed state/plan, define the resource-induced log-time contrast $D_t^T(\rho_a, \rho_b) = \log T_t(P_t, \rho_b) -$

$\log T_t(P_t, \rho_a)$. Let $e_t(\rho) = \log \widehat{T}_t(\rho) - \log T_t(\rho)$. Replacing predicted times in Eq. (13) by conditional means gives the true margin $m_t^\star$, and

$$m_t(\rho_a, \rho_b) - m_t^\star(\rho_a, \rho_b) = -[e_t(\rho_b) - e_t(\rho_a)]. \tag{16}$$

Thus, allocation-margin error is exactly the error in the matched same-state duration contrast; a state-common log-time offset cancels, whereas relative response distortion can change the selected budget. ELP therefore combines absolute log-time supervision with direct supervision of these same-state contrasts; its full loss, architecture, pair construction, and training protocol are in Appendix C.3. Theorem B.3 further shows that if the within-state log-time error oscillation is at most ε_T, the selected true planned service rate is at least $e^{-\varepsilon_T}$ times the best candidate rate.

Appendix B.10 gives a stronger Bellman-style candidate-class consistency characterization. Instead, Section 4.5 tests the observable planned-to-realized residual in Eq. (10) and the terminal consequence of local allocation disagreements under a common continuation.

4 Experiments

We evaluate complete-policy joint-SLO performance, isolate the allocation and learning mechanisms, audit the observable local-to-terminal bridge, and measure robustness and online cost.

4.1 Experimental Setup

SLOPE is implemented on vLLM 0.11.1 (Kwon et al., 2023). We evaluate Hunyuan-A13B/SCROLLS, Llama-3.3-70B/RAGBench, and Qwen3-14B/LMSYS, covering long-context, RAG, and real-world conversation workloads (Team et al., 2026; Shaham et al., 2022; Meta AI, 2024; Friel et al., 2024; Yang et al., 2025; Zheng et al., 2024). Hunyuan and Qwen run on $2\times$A40/TP2; Llama runs on $8\times$RTX 4090 D/TP8. Each method/load uses three request-stream seeds with open-loop arrivals followed by drain; loads span $1.0\times$–$1.5\times$ (actual QPS in Appendix D.4). We report run-level means and sample SD and do not treat frames within a run as independent replicates.

We compare with JITServe (Zhang et al., 2026), QoServe (Goel et al., 2026), native vLLM FCFS, Global-SJF (Fu et al., 2024), Global-SRPT, FastServe-SJMLFQ (Wu et al., 2026), and Dual-stage EDF; mechanism coverage is summarized in Appendix C.6.

Normalized SLO calibration. Absolute latency scales differ substantially across model–hardware configurations, so a single wall-clock deadline would confound scheduler quality with the intrinsic latency scale of the serving system. We therefore define TTFT and E2E allowances relative to each configuration's isolated-service reference, using the medians over 300 individually served requests. We evaluate $10\times$, $20\times$, and $30\times$ these references to sweep SLO strictness from tight to loose operating regimes. The $20\times$ setting is used only as a common reporting point; the main evaluation covers the full three-threshold sweep. This normalization defines the evaluation protocol rather than a scheduler parameter. Appendix D.4 gives capacity, aggregation, and load-average definitions.

4.2 End-to-End Joint-SLO Performance

The purpose of the multiplier sweep is not to identify a favorable SLO threshold, but to test whether the allocation advantage persists as the latency budget changes. At $20\times$, SLOPE reduces load-averaged joint violations relative to the best evaluated baseline at each load by **30.1%, 15.5%, and 23.3%** for Hunyuan, Llama, and Qwen, respectively (24.10, 6.86, and 3.27 percentage points), and has lower mean joint violation at 6/6, 5/6, and 5/6 displayed loads. At $30\times$, the corresponding average reductions are **13.92, 7.10, and 1.69 percentage points**. Joint attainment is a request-level conjunction rather than a sum of marginal metrics, so component margins can trade off: for Hunyuan at $20\times$, TTFT is 23.35 points higher while E2E is 28.31 points lower than independently chosen component-best baselines. Appendix D.1 reports the matching TTFT and E2E curves.

4.3 Ablation and Robustness

Ablating allocation decisions. On Hunyuan-A13B/SCROLLS, we compare full SLOPE (A0), three fixed prefill-budget upper-limit ratios (A1–A3), fixed output demand (A4), and selection by progress alone (A5). These interventions examine dynamic investment, request-specific generation demand, and time normalization. Among the tested fixed configurations, ratio 1.0 has the lowest violation

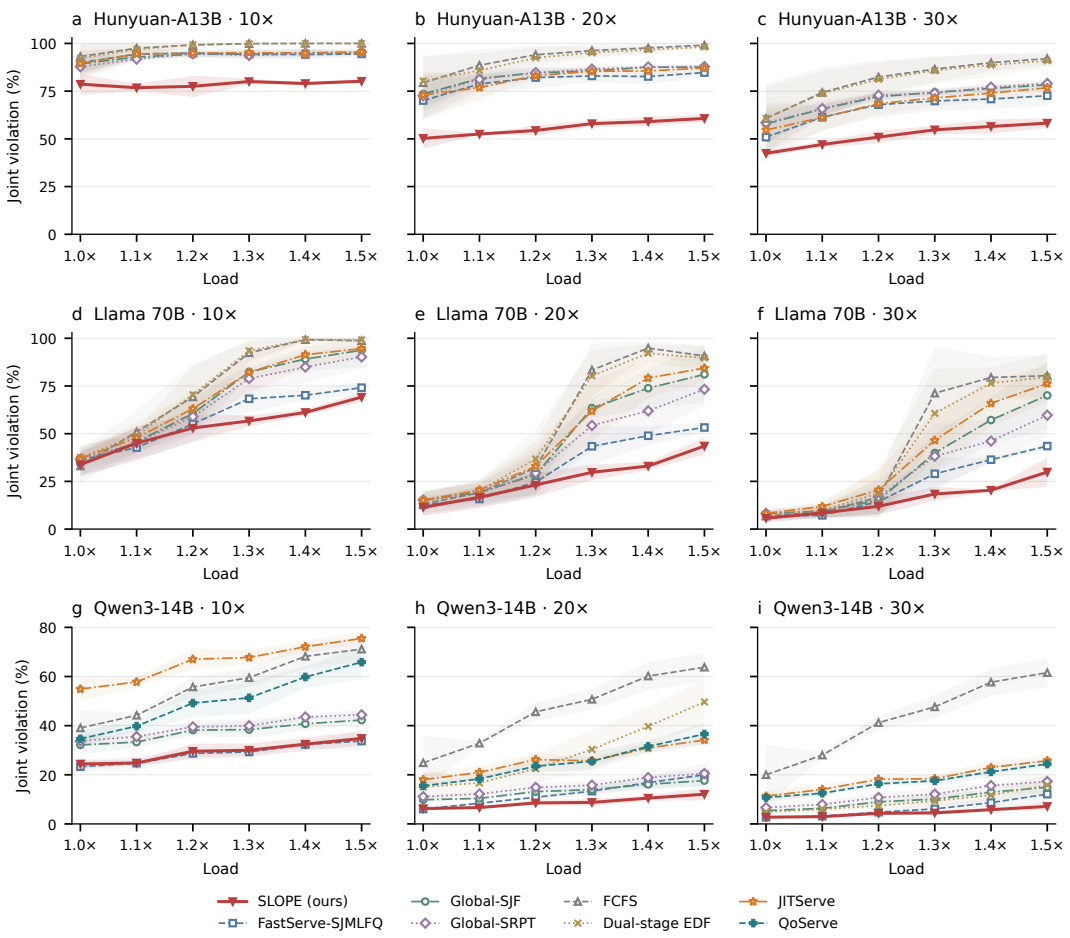


Figure 3: End-to-end joint violation rates across loads and SLO thresholds. Rows correspond to Hunyuan-A13B/SCROLLS, Llama 70B/RAGBench, and Qwen3-14B/LMSYS; columns correspond to 10×, 20×, and 30× SLOs. Points and lines show means over three request-stream seeds; bands show one sample standard deviation, clipped to $[0, 100]\%$.

rate, but remains **17.28 and 30.42 percentage points** above SLOPE at the two loads. Fixing output demand increases violations by **9.59 and 31.84 points**; selecting by progress alone increases them by **18.07 and 29.16 points**. These ablations show that the full gains require dynamic prefill investment, request-specific generation-demand estimation, and execution-time normalization.

Table 1: Allocation ablations on Hunyuan-A13B/SCROLLS. Mean joint violation rates (%) over three seeds; lower is better. Columns indicate load. A1–A3 fix budget upper-limit ratios, not realized token fractions.

Condition	1.0×	1.5×
A0 SLOPE (ours)	**50.25**	**60.71**
A1 Fixed prefill ratio 0.2	83.19	98.19
A2 Fixed prefill ratio 0.5	73.29	93.24
A3 Fixed prefill ratio 1.0	67.53	91.13
A4 Fixed output demand	59.84	92.55
A5 Progress-only selection	68.32	89.87

Sensitivity to shaping and reference-concurrency parameters. Across five weight configurations, two models, and two loads (60 runs), mean joint violation rates differ from the default by at most 0.63 percentage points for Hunyuan and 1.34 for Qwen. Under Eq. (8), these settings vary the dense-prefill shaping strength λ and/or reference decode concurrency d_0; Appendix D.6 gives the full sweep.

Robustness to bursty traffic and heterogeneous SLOs. We evaluate five Llama 70B/RAGBench arrival profiles, b0–b4, covering steady arrivals, periodic bursts of varying intensity, and flash

crowds. Each profile retains the same request-set size and primarily changes the arrival pattern of the batch tenant; requests use their own tenants' SLOs. SLOPE achieves a lower mean joint violation rate than the best baseline in each profile, improving by **0.72–6.06 percentage points**. The gap is smaller at b2, while the gain remains 6.06 points under b4. Appendix D.2 reports the corresponding TTFT/E2E components and Appendix D.3 gives the tenant composition and burst protocol.

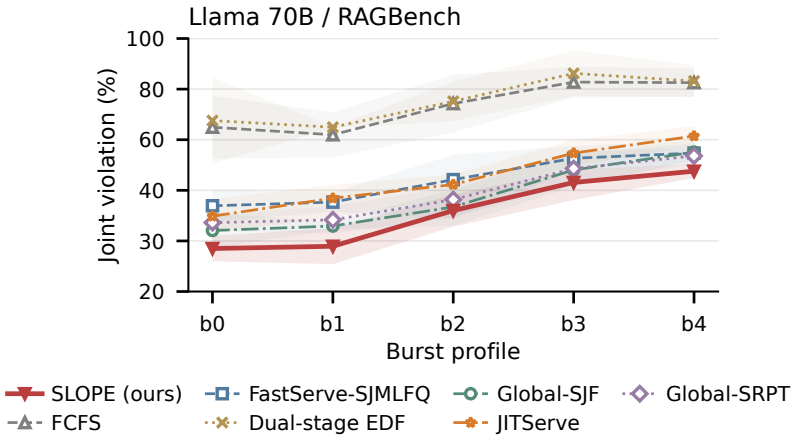


Figure 4: Bursty mixed-tenant traffic on Llama 70B/RAGBench. Means and sample-SD bands over three seeds use tenant-specific SLOs; profiles b0–b4 are defined in Appendix D.3.

4.4 Relative-Cost Learning and Candidate Selection

Table 2: Candidate selection at $1.0\times$ capacity. Mean normalized progress-rate loss; lower is better.

Selection rule	Loss (%)
Full ELP	**13.28**
Matched absolute-only	18.06
Progress only (N)	66.35

We compare full ELP, a matched absolute-only control, and progress-only selection on the *same request plans and candidate sets*. The learned predictors share 5,305 absolute labels, architecture, initialization, and absolute-supervision settings; full ELP additionally uses 168 same-state relative pairs from 30 training states. Using the common measured-cost statistic C_t, the normalized local selection loss is $1 - [N_t(\widehat{\rho}_t)/C_t(\widehat{\rho}_t)]/\max_{\rho\in\Omega_t}[N_t(\rho)/C_t(\rho)]$, which is distinct from terminal regret. Full ELP reduces this loss from 18.06% to **13.28%** (26.5% relative); progress-only reaches 66.35%. Appendix D.5 gives the timing scope and matched-control details.

4.5 Service Progress and Joint-SLO Gains

We examine how SLOPE's allocation criterion translates into service outcomes: whether planned progress is realized during execution, and whether the current resource choice improves terminal joint attainment. We use Hunyuan-A13B/SCROLLS at 1.5 × capacity with three independently generated request streams.

Planned and realized service. Across all 329 normal mixed decisions, planned prefill work is fully realized, with no service shortfall in planned decode coverage or designated phase completion. To quantify the discrepancy in SLO-normalized service, let $\widehat{\Delta}_k$ denote the sum of the positive planned-to-realized discrepancies in decode service, useful prefill work, and phase completion. Its componentwise definition follows the decomposition underlying Eq. (10) and is given in Appendix B.11. We report the backlog-weighted relative discrepancy

$$R_\Delta = \frac{\sum_k \overline{Q}_k \widehat{\Delta}_k}{\sum_k \overline{Q}_k \overline{N}_k}. \tag{17}$$

This weighting emphasizes discrepancies at larger service backlogs. Across the three request streams, R_Δ is $3.34\%\pm2.64\%$ (mean $\pm$ sample SD). All positive discrepancies arise from reference-concurrency normalization when $n_k^D < d_0$; all **258 decisions** with $n_k^D \geq d_0$ have zero sample

discrepancy. This agrees with the analytical characterization of the normalization term and locates the observed gap in that term rather than execution-side service loss. These statistics are computed offline from complete execution traces; final output lengths are used only for analysis.

Terminal gains from a single allocation choice. To isolate the effect of the current action, we use two pre-specified states per request stream at which SLOPE and progress-only select different actions: one during arrivals and one during drain. Starting from the same saved GPU state and future input trace, the two branches execute the first action selected by $N/\widehat{T}$ and N, respectively. Both then resume the same SLOPE policy until all requests drain.

Each action is executed twice. Averaging the two executions, SLOPE achieves a higher terminal joint-success count in **five of the six states** and ties in the remaining state. Averaging state differences within each stream and then weighting streams equally gives 0.83 **additional jointly successful requests per replayed state**. These results show that, in the tested disagreement states, accounting for execution cost in a single allocation choice can improve terminal joint attainment even under a shared continuation policy.

Table 3 summarizes these results. Appendix B.11 provides the complete measurement definitions, per-stream and concurrency-regime statistics, conservation checks, state-restoration protocol, execution order, and per-state replay outcomes.

Table 3: Service realization and terminal outcomes of local allocation choices. Trace metrics are computed within each request stream before cross-stream aggregation. Replay executions are technical repeats with a common SLOPE continuation.

Experiment	Evaluation scope	Main result
Planned and realized service	3 streams, 329 mixed decisions	$R_\Delta = 3.34\% \pm 2.64\%$; all 258 decisions with $n_k^D \geq d_0$ have zero sample discrepancy
Terminal gains from a single allocation choice	6 pre-specified disagreement states; 2 executions per action	Higher joint-success count in 5 states and a tie in 1; equal-stream mean gain of 0.83 successful requests per replayed state

4.6 Scheduling Overhead

Llama 70B scheduling calls remain at the millisecond scale. Table 4 compares FCFS and SLOPE call p95 and cumulative scheduling time as a fraction of the recorded run span. Across $1.0\times$–$1.5\times$ load, SLOPE's p95 is **1.06–1.86 ms** and its scheduling fraction is **0.53%–0.59%**. SLOPE adds online decision time relative to FCFS while delivering the joint-service gains reported above. Statistics cover recorded scheduling calls, and the run span includes drain. Appendix D.5 describes the measurement scope and aggregation.

Table 4: Scheduling overhead for Llama 70B/RAGBench, TP8. Entries are the mean $\pm$ sample SD over three seeds, after computing p95 and the scheduling fraction F separately within each run. F is not end-to-end slowdown.

Load	FCFS p95 (ms)	SLOPE p95 (ms)	FCFS F (%)	SLOPE F (%)
$1.0\times$	$0.455 \pm .093$	$1.059 \pm .198$	$0.226 \pm .011$	$0.529 \pm .035$
$1.3\times$	$0.708 \pm .005$	$1.602 \pm .047$	$0.244 \pm .004$	$0.560 \pm .023$
$1.5\times$	$0.753 \pm .029$	$1.857 \pm .312$	$0.252 \pm .003$	$0.591 \pm .040$

5 Conclusion

SLOPE connects terminal joint-SLO risk to online resource allocation in co-located LLM serving through an SLO-normalized potential-removal rate. The common-plan construction isolates marginal resource effects, bringing phase-aware service valuation and ELP's relative execution-cost learning into the same allocation comparison. Across three model/workload pairs, SLOPE reduces load-averaged joint violations at the $20\times$ SLO threshold by **15.5%–30.1% relatively** against the best evaluated baseline at each load, with scheduling-call p95 below **2 ms** on Llama 70B. Trace and controlled replay audits further connect planned service to realized execution and the terminal consequences of individual allocation choices. Together, these results support the joint design of service valuation and cost learning for effective, low-overhead resource allocation between input processing and ongoing generation.

AI USE STATEMENT

Generative AI tools were used to assist with language editing, rewriting and translation, figure and table presentation, and consistency checks. All such assistance was carried out under the authors' direction and control, and the resulting materials were reviewed and validated by the authors. The research questions, method design, experimental plans, interpretation of results, and scientific conclusions were ultimately determined by the authors.

REPRODUCIBILITY STATEMENT

Section 4.1 specifies the models, datasets, hardware, comparison methods, and evaluation metrics used in the experiments. Appendix B provides the assumptions, formal derivations, and proofs underlying the theoretical results. Appendix C describes the output-length predictor, candidate-budget construction and execution, the architecture and training settings of the execution latency predictor, and the default shaping/reference-concurrency parameterization. Appendix D reports supplementary TTFT and E2E component results and details the mixed-tenant and burst-arrival settings, load calibration, statistical aggregation, ablations, candidate evaluation, and scheduling overhead. Appendix B.11 gives the measurement definitions, aggregation rules, and replay protocol for the trace residual audit and controlled first-action intervention. These details support reproduction of the theoretical and empirical results reported in the paper.

REFERENCES

Amey Agrawal, Nitin Kedia, Ashish Panwar, Jayashree Mohan, Nipun Kwatra, Bhargav Gulavani, Alexey Tumanov, and Ramachandran Ramjee. Taming Throughput-Latency tradeoff in LLM inference with Sarathi-Serve. In *18th USENIX symposium on operating systems design and implementation (OSDI 24)*, pp. 117–134, 2024.

Siyuan Chen, Zhipeng Jia, Samira Khan, Arvind Krishnamurthy, and Phillip B Gibbons. Slos-serve: Optimized serving of multi-slo llms. *arXiv preprint arXiv:2504.08784*, 2025.

Adam N Elmachtoub and Paul Grigas. Smart "predict, then optimize". *Management science*, 68(1): 9–26, 2022.

Robert Friel, Masha Belyi, and Atindriyo Sanyal. Ragbench: Explainable benchmark for retrieval-augmented generation systems. *arXiv preprint arXiv:2407.11005*, 2024.

Yichao Fu, Siqi Zhu, Runlong Su, Aurick Qiao, Ion Stoica, and Hao Zhang. Efficient llm scheduling by learning to rank. *Advances in Neural Information Processing Systems*, 37:59006–59029, 2024.

Kanishk Goel, Jayashree Mohan, Nipun Kwatra, Ravi Shreyas Anupindi, and Ramachandran Ramjee. Qoserve: Breaking the silos of llm inference serving. In *Proceedings of the 31st ACM International Conference on Architectural Support for Programming Languages and Operating Systems, Volume 2*, pp. 1492–1507, 2026.

Ke Hong, Xiuhong Li, Lufang Chen, Qiuli Mao, Guohao Dai, Xuefei Ning, Shengen Yan, Yun Liang, and Yu Wang. Sola: Optimizing slo attainment for large language model serving with state-aware scheduling. *Proceedings of Machine Learning and Systems*, 7, 2025.

Woosuk Kwon, Zhuohan Li, Siyuan Zhuang, Ying Sheng, Lianmin Zheng, Cody Hao Yu, Joseph Gonzalez, Hao Zhang, and Ion Stoica. Efficient memory management for large language model serving with pagedattention. In *Proceedings of the 29th symposium on operating systems principles*, pp. 611–626, 2023.

Meta AI. Llama-3.3-70B-Instruct: Instruction-tuned multilingual llm. Hugging Face Model Hub, 2024. URL `https://huggingface.co/meta-llama/Llama-3.3-70B-Instruct`.

John Schulman, Sergey Levine, Pieter Abbeel, Michael Jordan, and Philipp Moritz. Trust region policy optimization. In *International conference on machine learning*, pp. 1889–1897. Pmlr, 2015.

Uri Shaham, Elad Segal, Maor Ivgi, Avia Efrat, Ori Yoran, Adi Haviv, Ankit Gupta, Wenhan Xiong, Mor Geva, Jonathan Berant, et al. Scrolls: Standardized comparison over long language sequences. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pp. 12007–12021, 2022.

Siyu Song, Qi Bai, Jinbo Hao, Kai Li, Chenchen Wang, and Jiayu Sun. Deadline-aware adaptive prefill chunking for efficient large language model serving. *arXiv preprint arXiv:2609.07883*, 2026.

Tencent Hunyuan Team, Ao Liu, Botong Zhou, Can Xu, Chayse Zhou, ChenChen Zhang, Chengcheng Xu, Chenhao Wang, Decheng Wu, Dengpeng Wu, Dian Jiao, Dong Du, Dong Wang, Feng Zhang, Fengzong Lian, Guanghui Xu, Guanwei Zhang, Hai Wang, Haipeng Luo, Han Hu, Huilin Xu, Jiajia Wu, Jianchen Zhu, Jianfeng Yan, Jiaqi Zhu, Jihong Zhang, Jinbao Xue, Jun Xia, Junqiang Zheng, Kai Liu, Kai Zhang, Kai Zheng, Kejiao Li, Keyao Wang, Lan Jiang, Lixin Liu, Lulu Wu, Mengyuan Huang, Peijie Yu, Peiqi Wang, Qian Wang, Qianbiao Xiang, Qibin Liu, Qingfeng Sun, Richard Guo, Ruobing Xie, Saiyong Yang, Shaohua Chen, Shihui Hu, Shuai Li, Shuaipeng Li, Shuang Chen, Suncong Zheng, Tao Yang, Tian Zhang, Tinghao Yu, Weidong Han, Weijie Liu, Weijin Zhou, Weikang Wang, Wesleye Chen, Xiao Feng, Xiaoqin Ren, Xingwu Sun, Xiong Kuang, Xuemeng Huang, Xun Cao, Yanfeng Chen, Yang Du, Zhen Yang, Yangyu Tao, Yaping Deng, Yi Shen, Yigeng Hong, and Yiqi Chen. Hunyuan-a13b technical report, 2026. URL https://arxiv.org/abs/2609.27284.

Bryan Wilder, Bistra Dilkina, and Milind Tambe. Melding the data-decisions pipeline: Decision-focused learning for combinatorial optimization. In *Proceedings of the AAAI conference on artificial intelligence*, volume 33, pp. 1658–1665, 2019.

Bingyang Wu, Yinmin Zhong, Zili Zhang, Shengyu Liu, Fangyue Liu, Yuanhang Sun, Gang Huang, Xuanzhe Liu, and Xin Jin. FastServe: Iteration-Level preemptive scheduling for large language model inference. In *23rd USENIX Symposium on Networked Systems Design and Implementation (NSDI 26)*, pp. 57–74, 2026.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, Chujie Zheng, Dayiheng Liu, Fan Zhou, Fei Huang, Feng Hu, Hao Ge, Haoran Wei, Huan Lin, Jialong Tang, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxi Yang, Jing Zhou, Jingren Zhou, Junyang Lin, Kai Dang, Keqin Bao, Kexin Yang, Le Yu, Lianghao Deng, Mei Li, Mingfeng Xue, Mingze Li, Pei Zhang, Peng Wang, Qin Zhu, Rui Men, Ruize Gao, Shixuan Liu, Shuang Luo, Tianhao Li, Tianyi Tang, Wenbiao Yin, Xingzhang Ren, Xinyu Wang, Xinyu Zhang, Xuancheng Ren, Yang Fan, Yang Su, Yichang Zhang, Yinger Zhang, Yu Wan, Yuqiong Liu, Zekun Wang, Zeyu Cui, Zhenru Zhang, Zhipeng Zhou, and Zihan Qiu. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*, 2025.

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 *16th USENIX symposium on operating systems design and implementation (OSDI 22)*, pp. 521–538, 2022.

Wei Zhang, Zhiyu Wu, Yi Mu, Rui Ning, Banruo Liu, Nikhil Sarda, Myungjin Lee, and Fan Lai. JITServe: SLO-aware LLM serving with imprecise request information. In *23rd USENIX Symposium on Networked Systems Design and Implementation (NSDI 26)*, pp. 825–848, 2026.

Lianmin Zheng, Wei-Lin Chiang, Ying Sheng, Tianle Li, Siyuan Zhuang, Zhanghao Wu, Yonghao Zhuang, Zhuohan Li, Zi Lin, Eric Xing, et al. Lmsys-chat-1m: A large-scale real-world llm conversation dataset. In *International Conference on Learning Representations*, volume 2024, pp. 22225–22257, 2024.

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 *18th USENIX symposium on operating systems design and implementation (OSDI 24)*, pp. 193–210, 2024.

APPENDIX CONTENTS

A Related Work **13**
A.1 SLO-Aware Scheduling and Resource Allocation . . . 13
A.2 Chunked Execution and Phase-Aware Allocation . . . 13
A.3 Prediction-Guided Scheduling and Decision-Induced Cost Learning . . . 13

B Theoretical Analysis **14**
B.1 Analytical Model and Notation . . . 14
B.2 Identification of the Marginal Resource Contrast . . . 15
B.3 Characterization of the Local Two-Phase Debt . . . 16
B.4 Execution-Response Distortion and Candidate Selection . . . 16
B.5 From Local Phase-Service Debt to an SLO-Normalized Trajectory Potential . . . 18
B.6 Potential Area and Joint Deadlines . . . 20
B.7 Online Service Rates and Cumulative Risk . . . 21
B.8 Auxiliary Moment and Cost-Statistic Results . . . 23
B.9 Output-Length Prediction and Demand Representation . . . 24
B.10 Conditional Candidate-Class Terminal Consistency . . . 24
B.11 Empirical Audit of the Local-to-Terminal Bridge . . . 28

C Implementation and Training Details **30**
C.1 Output-Length Predictor . . . 30
C.2 Candidate Construction and Execution . . . 30
C.3 Execution Latency Predictor and Cost Learning . . . 31
C.4 Runtime Shaping and Reference-Concurrency Parameters . . . 32
C.5 Online Information and Resource Feasibility . . . 33
C.6 Mechanism Comparison and Baseline Coverage . . . 33

D Supplementary Experiments and Load Definitions **35**
D.1 TTFT and E2E Components of the End-to-End Experiments . . . 35
D.2 Component Metrics under Bursty Mixed-Tenant Traffic . . . 36
D.3 Mixed-Tenant Composition and Burst Profiles . . . 36
D.4 Load Calibration and Statistical Aggregation . . . 37
D.5 Supplementary Details for Ablations and Candidate Evaluation . . . 38
D.6 Robustness to Shaping and Reference-Concurrency Parameters . . . 38

A Related Work

A.1 SLO-Aware Scheduling and Resource Allocation

LLM serving schedulers coordinate resource allocation using request demand, deadlines, and system state to meet service-level objectives. Prior work explores state-aware request ordering, workload adaptation, and token allocation for multistage requests (Hong et al., 2025; Chen et al., 2025). JIT-Serve (Zhang et al., 2026) combines request goodput with serving-bandwidth demand to organize request selection and batch execution, representing scheduling approaches that balance service benefits against resource requirements. QoServe (Goel et al., 2026) dynamically allocates prefill work according to deadlines, representing deadline-aware resource coordination across prefill and decode. SLOPE couples phase-aware service-value modeling with execution-cost learning tailored to allocation decisions. Under a common request plan, it represents candidate service progress through input advancement, prefill-completion events, and decode coverage, and evaluates resource investments by comparing relative progress gains with relative increases in shared execution time. This allocation criterion further identifies the candidate-relative cost information that prediction must preserve, bringing service-value modeling, online resource allocation, and the prediction objective into a unified method design.

A.2 Chunked Execution and Phase-Aware Allocation

Sarathi-Serve (Agrawal et al., 2024) coordinates input processing and ongoing decoding through chunked prefill and stall-free batching, supporting throughput–latency optimization in shared execution. Subsequent work dynamically adjusts prefill chunks using deadlines and available latency slack to adapt input progress to online service constraints (Goel et al., 2026; Song et al., 2026). SLOPE develops an allocation criterion that captures both continuous work advancement and discrete phase transitions. Effective input advancement, prefill completion of a designated resident request, and decode coverage jointly define candidate service progress; full-iteration execution time measures the shared cost. This representation connects chunk size to service state, allowing the same additional token budget to have different value depending on phase position and decode demand. Resource investment is then guided by the relative progress gain compared with the relative execution-time increase.

A.3 Prediction-Guided Scheduling and Decision-Induced Cost Learning

Predictive models provide future-demand information for online scheduling. LTR (Fu et al., 2024) learns relative output-length ordering to support approximate shortest-job-first scheduling. More broadly, predict-then-optimize and decision-focused learning design predictive objectives around downstream optimization tasks (Elmachtoub & Grigas, 2022; Wilder et al., 2019). SLOPE jointly designs budget allocation and execution-cost learning. Its progress-rate comparison identifies pairwise relative log-costs as the decision-relevant quantity, which determines the supervision of execution costs across candidates under the same state. SLOPE combines this relative-cost supervision with absolute log-latency supervision and establishes a bound linking relative-cost error to the progress-rate quality of candidate selection. The prediction objective, online selection rule, and error analysis therefore address the same allocation problem, connecting two-phase service-value modeling with decision-induced cost learning.

B THEORETICAL ANALYSIS

This appendix develops the derivation used by SLOPE rather than treating the online SLO violation ratio as a primitive objective. Throughout this appendix, we refer to the local measure of outstanding service introduced in Section 3.3 as *service debt*. At a frozen planning boundary, it accounts for pending next-step decode obligations, unfinished prefill work, and the designated unfinished resident prefill milestone. With SLOPE's coefficients, the planned reduction in this measure under a candidate action is exactly the service numerator $N_k(a)$.

Section B.1 defines the variable-duration service process. Section B.2 shows that a resource-only candidate contrast is not identifiable when the request plan is allowed to change with the budget, which motivates SLOPE's common-plan construction. Section B.3 characterizes the local two-phase debt within the additive, symmetric hybrid-state model used online. Section B.4 then bounds the effect of error in the identified execution-time contrast. Section B.5 maps the runtime weights to an SLO-normalized trajectory potential and makes explicit the residual between planned local decrement and realized potential removal. Sections B.6–B.8 show that the area of a weight-free TTFT/E2E risk occupancy—and therefore the SLO-normalized dense potential that upper-envelopes it—controls joint violations under explicit workload and service conditions. Section B.9 separately characterizes output-length error in request planning. Finally, Section B.10 gives an additional stronger Bellman-style candidate-class consistency result. Its Bellman margins are not used as numerical certificates for the reported experiments; the main terminal bridge used by the method is the SLO-normalized pathwise certificate and the explicit local residual developed here. Section B.11 then audits the sample counterparts of that residual on complete execution traces and uses controlled first-action replay to test whether local allocation disagreements can change terminal joint service under a common SLOPE continuation.

B.1 ANALYTICAL MODEL AND NOTATION

Decisions and execution. Let s_k be the time when iteration k's request-planning context is frozen, before budget selection, with $s_0 = 0$. In a finite, nonempty, deduplicated candidate set Ω_k, a denotes a prefill-budget candidate, corresponding to ρ in the main text. It produces initial plan $\widetilde{z}_k(a)$ and final action $z_k(a) = \mathcal{R}_k(\widetilde{z}_k(a))$. All candidates in an iteration share the same initial state, planning context, and resource-check rules.

Let $\mathcal{H}_k$ denote observable history and $\mathcal{F}_k$ an analytical history that additionally includes latent output lengths of arrived requests. The policy is measurable with respect to $\mathcal{H}_k$; latent lengths are used only for analysis. Write $\mathbb{E}_k^a[\cdot] = \mathbb{E}[\cdot \mid \mathcal{F}_k, a]$ and abbreviate this to $\mathbb{E}_k[\cdot]$ for the selected action. Candidate quantities condition on the same current state.

Local debt decrement and timing. The main text derives the progress numerator as the planned decrement of the local phase-service debt. The resulting numerator and selection rule are

$$\begin{aligned} N_k(a) &= w_D\, \eta_k^D(a) + w_P\, \frac{W_k(a)}{B} + w_F\, f_k(a), \\ \widehat{a}_k &\in \arg\max_{a \in \Omega_k} \frac{N_k(a)}{\widehat{T}_k(a)}, \qquad \widehat{T}_k(a) = \exp \ell_\theta(x_k, a). \end{aligned} \tag{B.1}$$

The positive weights w_D, w_P, w_F are fixed across states and candidates, as in Eq. (15). Here n_k^D is the number of all decode requests awaiting service, fixed before comparison, and $d_k^D(a)$ counts those advanced by a candidate, with $\eta_k^D(a) = d_k^D(a)/n_k^D$. Each request advances at most one new output token in the iteration; the denominator is not the total future output length. Mixed iterations have $n_k^D > 0$. For unified accounting, set $\eta_k^D = 0$ when no decode request awaits service; the actual action still follows that branch's established control. $W_k(a)$ is effective input quota and $f_k(a) \in \{0, 1\}$ marks the designated resident prefill request's phase completion. With $B = 16384$ and $W_k(a) \le 4096$, $0 \le N_k(a) \le w_D + w_P/4 + w_F$. Retained nonempty input candidates have $W_k(a) > 0$ and hence positive $N_k(a)$. Full decode coverage gives $N_k(a) \ge w_D$. For the default weights in Appendix C.4, these bounds specialize to $0 \le N_k(a) \le 13/16$, or $1/2 \le N_k(a) \le 13/16$ under full coverage.

Table B.1: Principal notation for the service analysis.

Symbol	Definition
$r_i^P(u), r_i^D(u)$	Remaining logical input and output work
$I_i^P(u)$	Indicator of an unfinished input milestone
$Q(u), Q_k$	Two-phase potential and boundary value $Q(s_k)$
$n_k^D, d_k^D(a), \eta_k^D(a)$	Fixed waiting-decode count, planned coverage count, and fraction
D_k, P_k, F_k	Actual first-time output/input progress and all input completions
S_k, A_k	Potential removed by service and added by arrivals
Λ, ν, h	Conditional bounds on arrival-work rate, net-work second moment, and within-frame area
q_0, κ	Backlog threshold and positive service margin
d^T, d^E	TTFT and E2E latency allowances
v_n, p_{joint}	Finite-set violation fraction and stationary typical-request violation probability

Define the full scheduling frame, its conditional mean cost, and the true score as

$$\tau_k = s_{k+1} - s_k > 0, \qquad T_k(a) = \mathbb{E}_k^a[\tau_k] \in (0, \infty), \qquad V_k(a) = \frac{N_k(a)}{T_k(a)}. \tag{B.2}$$

A full frame includes candidate comparison, final checks, execution, feedback, and common planning up to the next frozen boundary. Non-ELP branches use corresponding boundaries so that frames cover the entire time axis. Because all candidates at a boundary share the same frozen planning context, $V_k(a)$ compares alternative resource investments rather than alternative request-selection rules. The ratio uses mean time and generally differs from the expectation of an inverse random time. Appendix B.8.2 corrects for shorter measurement intervals.

Request model. Request i arrives at a_i with input length p_i and latent output length $L_i \geq 1$. Length may be random and input-dependent but is held consistent across compared scheduling policies. Service-progress analysis considers normal completion without dropping or cancellation. Logical tokens count as progress only at their first processing or generation. Prefix reuse may count as completed logical work; recomputation adds time without adding progress. There are no work jumps other than arrivals and accounted service. Candidate index a is distinguished from request arrival time a_i by its subscript, as is frame completion count F_k from client first-response time F_i.

B.2 Identification of the Marginal Resource Contrast

The main text factors a scheduling action into a request plan P and a resource investment ρ. This section formalizes why holding P fixed is necessary if the object of interest is the direct effect of changing ρ.

Let $Y(P, \rho)$ be any real-valued candidate outcome at a fixed serving state. For execution cost, one may take $Y = \log T$; for service, Y may be a planned local-debt decrement. If two observed candidates use plans P_a and P_b, then

$$Y(P_b, \rho_b) - Y(P_a, \rho_a) = [Y(P_a, \rho_b) - Y(P_a, \rho_a)] + [Y(P_b, \rho_b) - Y(P_a, \rho_b)]. \tag{B.3}$$

The first bracket is the resource contrast at fixed plan P_a; the second is a plan-substitution contrast evaluated at the same resource level.

Proposition B.1 (Plan coupling does not identify the resource contrast)**.** *Suppose only the two endpoint outcomes $y_a = Y(P_a, \rho_a)$ and $y_b = Y(P_b, \rho_b)$ are available, with $P_a \neq P_b$. Then the direct resource contrast*

$$\Delta_\rho Y(P_a; \rho_a, \rho_b) = Y(P_a, \rho_b) - Y(P_a, \rho_a)$$

is not identified by (y_a, y_b). In particular, for any $c \in \mathbb{R}$ there exists a real-valued outcome function $\widetilde{Y}$ that matches both observed endpoints and has direct contrast c at P_a.

Proof. Set $\widetilde{Y}(P_a, \rho_a) = y_a$, $\widetilde{Y}(P_b, \rho_b) = y_b$, and $\widetilde{Y}(P_a, \rho_b) = y_a + c$; define $\widetilde{Y}$ arbitrarily on all other plan–budget pairs. The two observed endpoints are unchanged, while the direct resource contrast equals c. Thus no endpoint-only rule can separate the first and second brackets in Eq. (B.3) without an additional structural assumption or a matched-plan evaluation. □

SLOPE supplies that matched evaluation by fixing P_k before candidate construction and evaluating the family $\{(P_k, a) : a \in \Omega_k\}$. Within this family the plan-substitution term in Eq. (B.3) is identically zero, so both the service contrast and the log-time contrast are resource-only quantities by construction. This proposition is an identifiability statement about the comparison object; it does not require a causal model for future workload arrivals.

B.3 Characterization of the Local Two-Phase Debt

We next formalize the local measure of outstanding service introduced in Section 3.3. Fix one frozen planning boundary. Let $d \in \{0, 1, \dots, n^D\}$ be the number of pending next-step decode obligations, let $\mathbf{w} = (w_1, \dots, w_m) \in \mathbb{R}_+^m$ be effective remaining prefill work for the requests exposed by the plan, and let $I \in \{0, 1\}$ indicate the designated resident prefill milestone.

Proposition B.2 (Additive representation of local two-phase debt)**.** *Let $\Phi(d, \mathbf{w}, I) \geq 0$ satisfy the following conditions at the frozen boundary:*

1. Debt-type separability: $\Phi(d, \mathbf{w}, I) = \Phi_D(d) + \Phi_P(\mathbf{w}) + \Phi_F(I)$ *and each component vanishes at zero.*
2. Decode symmetry and additivity: *the decode component depends only on the number of pending obligations and $\Phi_D(d_1 + d_2) = \Phi_D(d_1) + \Phi_D(d_2)$ whenever $d_1 + d_2 \leq n^D$.*
3. Continuous-work additivity and regularity: $\Phi_P(\mathbf{w}) = \sum_i \psi(w_i)$ *for a common continuous function ψ, and $\psi(x + y) = \psi(x) + \psi(y)$ for $x, y \geq 0$.*
4. Phase-boundary locality: *the binary resident milestone contributes a fixed amount when unfinished, i.e. $\Phi_F(1) = \gamma$ for some $\gamma \geq 0$.*

Then there exist $\alpha, \beta, \gamma \geq 0$ such that

$$\Phi(d, \mathbf{w}, I) = \alpha d + \beta \sum_i w_i + \gamma I. \tag{B.4}$$

Conversely, every functional of this form satisfies the four conditions.

Proof. Decode additivity on the nonnegative integers gives $\Phi_D(d) = d\Phi_D(1) = \alpha d$. For prefill work, the additive continuous function ψ on $\mathbb{R}_+$ is linear, so $\psi(w) = \beta w$ with $\beta = \psi(1) \geq 0$. Since I is binary and $\Phi_F(0) = 0$, $\Phi_F(I) = \gamma I$. Summing the three components proves Eq. (B.4); the converse is immediate. □

At state k, choose $\alpha = w_D/n_k^D$, $\beta = w_P/B$, and $\gamma = w_F$. The planned action removes $d_k^D(a)$ decode obligations, $W_k(a)$ units of continuous prefill work, and $f_k(a)$ units of the binary resident milestone. The decrement of Eq. (B.4) is therefore exactly

$$w_D \eta_k^D(a) + w_P \frac{W_k(a)}{B} + w_F f_k(a) = N_k(a).$$

The representation theorem determines the local state coordinates but does not by itself calibrate their coefficients. That calibration is supplied by the SLO-normalized trajectory interpretation in Section B.5. After removing a common positive scale, w_P/w_F becomes a certificate-preserving dense-prefill shaping coefficient and w_D/w_F determines the reference decode concurrency at which the local coverage term matches E2E-normalized generation service. Thus the weights are not interpreted as terminal-success probabilities.

B.4 Execution-Response Distortion and Candidate Selection

For a real function on a finite set, write $\mathrm{osc}_\Omega(g) = \max_{a \in \Omega} g_a - \min_{a \in \Omega} g_a$. Fix the current state and omit the frame index.

Within-state resource response. Fix any reference candidate $a_0 \in \Omega$ and write

$$R^T(a) = \log T_a - \log T_{a_0}, \qquad \widehat{R}^T(a) = \log \widehat{T}_a - \log \widehat{T}_{a_0}.$$

These functions describe how true and predicted execution time change as the resource investment moves away from a common reference while the serving state and request plan remain fixed. Candidate selection depends on the shape of this response, not on the arbitrary choice of reference.

If $e_a = \log \widehat{T}_a - \log T_a$, then for every candidate pair

$$(\widehat{R}^T(b) - \widehat{R}^T(a)) - (R^T(b) - R^T(a)) = e_b - e_a.$$

Hence $\mathrm{osc}_\Omega(e) = \max_a e_a - \min_a e_a$ is exactly the largest pairwise distortion of the learned log-time response, up to orientation. A state-wise common prediction offset cancels as a special case. The following theorem bounds allocation quality directly in terms of this response distortion.

Theorem B.3 (Relative-error guarantee for candidate selection)**.** *Suppose every candidate has $N_a, T_a, \widehat{T}_a > 0$, and let $e_a = \log \widehat{T}_a - \log T_a$. If*

$$\mathrm{osc}_\Omega(e) \le \varepsilon_T, \tag{B.5}$$

then the candidate selected by Eq. (B.1) *satisfies*

$$\frac{N_{\widehat{a}}}{T_{\widehat{a}}} \ge e^{-\varepsilon_T} \max_{a \in \Omega} \frac{N_a}{T_a}. \tag{B.6}$$

Proof. Take $a^\star \in \arg\max_a V_a$. Predicted-score maximization gives

$$\log V_{\widehat{a}} - e_{\widehat{a}} \ge \log V_{a^\star} - e_{a^\star}.$$

Hence $0 \le \log V_{a^\star} - \log V_{\widehat{a}} \le e_{a^\star} - e_{\widehat{a}} \le \varepsilon_T$. Exponentiation proves the claim. □

Corollary B.4 (Score loss and selection consistency)**.** *Under Theorem B.3, relative true-score loss is at most $1 - e^{-\varepsilon_T}$. If $|e_a| \le \epsilon$ for every candidate, take $\varepsilon_T = 2\epsilon$. If the optimum is unique and*

$$\min_{a \ne a^\star} \log \frac{V_{a^\star}}{V_a} > \varepsilon_T,$$

then $\widehat{a} = a^\star$.

Proof. The first two statements follow from Eq. (B.6) and $\max e - \min e \le 2\epsilon$. For any other candidate, the predicted log-score gap is at least $\log(V_{a^\star}/V_a) - \varepsilon_T > 0$, preserving the unique choice. □

Eq. (B.5) is equivalent to bounding every pairwise error in the learned log-time change by ε_T. Theorem B.3 is therefore a stability result for the state-conditioned execution response: a common offset has zero decision effect, while distortion of how latency changes across budget interventions controls the loss in the selected true debt-discharge rate. ELP's within-state relative-cost supervision targets these response differences directly; the absolute term separately calibrates the latency level.

Proposition B.5 (Correction for executed progress)**.** *Let $M_a = \mathbb{E}^a[S_a] > 0$ be the mean progress realized by the final action. If Theorem B.3 applies and*

$$\mathrm{osc}_\Omega\left(\log \frac{N_a}{M_a}\right) \le \varepsilon_N,$$

then

$$\frac{M_{\widehat{a}}}{T_{\widehat{a}}} \ge e^{-(\varepsilon_T + \varepsilon_N)} \max_{a \in \Omega} \frac{M_a}{T_a}. \tag{B.7}$$

Proof. For $R_a = M_a/T_a$, the log discrepancy between predicted score and actual progress rate is

$$\log \frac{N_a}{\widehat{T}_a} - \log R_a = \log \frac{N_a}{M_a} - e_a.$$

Its oscillation is at most $\varepsilon_N + \varepsilon_T$. Apply the maximization inequality from Theorem B.3. □

This result requires positive mean progress for all candidates. The next section supplies an additive correction allowing zero-progress candidates and specifies the two-phase meaning of S_k.

B.5 From Local Phase-Service Debt to an SLO-Normalized Trajectory Potential

Section 3.3 establishes an exact local statement: for the frozen request plan and candidate's planned quotas, $N_k(a)$ is the decrement of the local phase-service debt. We now place that decrement in the same TTFT/E2E units as the empirical objective.

Let $y_i(u)$ count output tokens first generated by time u, let $r_i^D(u) = (L_i - y_i(u))_+$, let $r_i^P(u)$ be remaining first-time logical input work, and let $I_i^P(u)$ indicate unfinished input processing. Also define $I_i^D(u) = \mathbf{1}[r_i^D(u) > 0]$. All four quantities are zero for completed requests. The weight-free risk occupancy is

$$R_{\mathrm{slo}}(u) = \sum_{i:a_i \le u} \left[\frac{I_i^P(u)}{d^T} + \frac{I_i^D(u)}{d^E} \right]. \tag{B.8}$$

Its area is the direct pathwise bridge to joint violations in Theorem B.10.

For the general service analysis, fix a reference decode concurrency $d_0 > 0$ and define

$$\alpha = \frac{w_D}{d_0}, \qquad \beta = \frac{w_P}{B}, \qquad \gamma = w_F, \qquad Q(u) = \sum_{i:a_i \le u} \left[\alpha r_i^D(u) + \beta r_i^P(u) + \gamma I_i^P(u) \right]. \tag{B.9}$$

The constant d_0 is fixed over the analytical interval and does not change the algorithm's state-dependent denominator n_k^D. The general form is convenient for the drift analysis below. To connect it to the actual SLO allowances, choose

$$d_0^{\mathrm{slo}} = \frac{w_D}{w_F} \frac{d^E}{d^T}, \qquad \lambda = \frac{w_P}{w_F}, \qquad c_{\mathrm{slo}} = \frac{1}{w_F d^T}. \tag{B.10}$$

Then

$$\overline{Q}_\lambda(u) := c_{\mathrm{slo}} Q(u) = \sum_{i:a_i \le u} \left[\frac{r_i^D(u)}{d^E} + \lambda \frac{r_i^P(u)}{B d^T} + \frac{I_i^P(u)}{d^T} \right]. \tag{B.11}$$

Because $r_i^D \ge I_i^D$ for every unfinished output and the shaping term is nonnegative,

$$R_{\mathrm{slo}}(u) \le \overline{Q}_\lambda(u) \qquad \text{for every } \lambda \ge 0. \tag{B.12}$$

This gives the runtime weights a specific interpretation. Common weight scaling is irrelevant. The ratio w_P/w_F selects how much nonnegative continuous prefill shaping is added to a valid SLO-risk certificate, while w_D/w_F fixes the reference concurrency d_0^{slo} at which one-step decode coverage is calibrated to $1/d^E$ generation-risk units. Neither ratio is interpreted as a terminal-success probability.

Actual service removes potential according to

$$S_k = \alpha D_k + \beta P_k + \gamma F_k, \qquad Q_{k+1} = Q_k + A_k - S_k \ge 0. \tag{B.13}$$

D_k includes first tokens generated at prefill completion; F_k counts all actual input completions and need not equal the designated-target marker f_k. The recursion follows directly from logical per-request accounting.

Lemma B.6 (Service-progress decomposition)**.** *For every state and candidate,*

$$\begin{aligned} N_k(a) - \mathbb{E}_k^a S_k = \alpha\big(d_0 \eta_k^D(a) - \mathbb{E}_k^a D_k\big) + \beta\big(W_k(a) - \mathbb{E}_k^a P_k\big) \\ + \gamma\big(f_k(a) - \mathbb{E}_k^a F_k\big). \end{aligned} \tag{B.14}$$

Define the progress overcount per unit time as

$$\xi_k(a) = \frac{\alpha(d_0 \eta_k^D(a) - \mathbb{E}_k^a D_k)_+ + \beta(W_k(a) - \mathbb{E}_k^a P_k)_+ + \gamma(f_k(a) - \mathbb{E}_k^a F_k)_+}{T_k(a)}. \tag{B.15}$$

Then the actual rate of service-potential reduction satisfies

$$\mu_k(a) := \frac{\mathbb{E}_k^a S_k}{T_k(a)} \ge V_k(a) - \xi_k(a). \tag{B.16}$$

Proof. Since $\alpha d_0 \eta_k^D(a) = w_D \eta_k^D(a)$, subtracting Eq. (B.13) from Eq. (B.1) yields Eq. (B.14). Upper-bound each discrepancy by its positive part and divide by $T_k(a) > 0$. □

Corollary B.7 (SLO-normalized local residual)**.** *Use the SLO-calibrated reference in Eq.* (B.10) *and define*

$$\overline{N}_k(a) = \frac{N_k(a)}{w_F d^T}, \qquad \overline{S}_k = \frac{S_k}{w_F d^T}, \qquad \overline{\mu}_k(a) = \frac{\mathbb{E}_k^a \overline{S}_k}{T_k(a)}.$$

Then

$$\overline{N}_k(a) - \mathbb{E}_k^a \overline{S}_k = \frac{(d_0^{\mathrm{slo}}/n_k^D) d_k^D(a) - \mathbb{E}_k^a D_k}{d^E} + \lambda \frac{W_k(a) - \mathbb{E}_k^a P_k}{B d^T} + \frac{f_k(a) - \mathbb{E}_k^a F_k}{d^T}. \tag{B.17}$$

Let $\overline{\xi}_k(a)$ be the sum of the positive parts of these three terms, divided by $T_k(a)$. Then

$$\overline{\mu}_k(a) \geq \frac{\overline{N}_k(a)}{T_k(a)} - \overline{\xi}_k(a). \tag{B.18}$$

Moreover, the positive decode-normalization residual satisfies

$$\left(\frac{d_0^{\mathrm{slo}}}{n_k^D} d_k^D(a) - \mathbb{E}_k^a D_k \right)_+ \leq \left(\frac{d_0^{\mathrm{slo}}}{n_k^D} - 1 \right)_+ d_k^D(a) + \left(d_k^D(a) - \mathbb{E}_k^a D_k \right)_+. \tag{B.19}$$

Thus the local-to-trajectory gap separates into an observable concurrency normalization term and plan-to-execution discrepancies. In particular, when $n_k^D \geq d_0^{\mathrm{slo}}$, the normalization itself cannot overstate E2E-calibrated decode service.

Proof. Multiply Eq. (B.14) by $1/(w_F d^T)$ and substitute Eq. (B.10); this gives Eq. (B.17). Taking positive parts gives Eq. (B.18). Finally write $ad - D = (a-1)d + (d - D)$ with $a = d_0^{\mathrm{slo}}/n_k^D$ and use $(x+y)_+ \leq x_+ + y_+$. □

Corollary B.8 (Service rate under predicted selection)**.** *If Theorem B.3 holds at the current state and $\sup_{a \in \Omega_k} \xi_k(a) \leq \xi$, then*

$$\mu_k(\widehat{a}_k) \geq e^{-\varepsilon_T} \max_{a \in \Omega_k} V_k(a) - \xi. \tag{B.20}$$

Proof. Apply Eq. (B.16) to the selected candidate and substitute Eq. (B.6). □

Corollary B.9 (SLO-normalized service rate under predicted selection)**.** *Use the SLO-calibrated normalization in Corollary B.7. If Theorem B.3 holds at the current state and $\sup_{a \in \Omega_k} \overline{\xi}_k(a) \leq \overline{\xi}$, then*

$$\overline{\mu}_k(\widehat{a}_k) \geq e^{-\varepsilon_T} \max_{a \in \Omega_k} \frac{\overline{N}_k(a)}{T_k(a)} - \overline{\xi}.$$

Proof. Theorem B.3 is unchanged by the common positive scaling $\overline{N}_k = N_k/(w_F d^T)$. Apply Eq. (B.18) to the selected candidate and substitute the scaled form of Eq. (B.6). □

The general decomposition covers all score components: current decode coverage, continuous input progress, and the designated input milestone. For $n_k^D > 0$, the unnormalized decode discrepancy also decomposes as

$$w_D \eta_k^D(a) - \alpha \mathbb{E}_k^a D_k = \frac{w_D}{n_k^D} \left(d_k^D(a) - \mathbb{E}_k^a D_k \right) + w_D \left(\frac{1}{n_k^D} - \frac{1}{d_0} \right) \mathbb{E}_k^a D_k.$$

Corollary B.7 is the form used for interpretation because it expresses the same discrepancy directly in TTFT/E2E units. The first term in Eq. (B.17) separates a reference-concurrency mismatch from planned-versus-realized generation; the other two terms isolate useful prefill-work and milestone mismatches. Extra realized service only strengthens the lower bound.

This reformulation also makes the local approximation empirically auditable. The analytical residual $\overline{\xi}_k(a)$ contains conditional expectations, so a single realized frame is not an exact observation of that quantity. Appendix B.11.1 instead measures the corresponding realized positive-part decomposition

on three complete request streams. Across 329 normal mixed decisions, the equal-seed backlog-weighted sample residual is $3.34\% \pm 2.64\%$; no positive execution-side discrepancy is observed, and all positive residual is confined to the $n_k^D < d_0$ reference-concurrency normalization regime. All 258 audited decisions with $n_k^D \geq d_0$ have zero sample residual, matching Eq. (B.19). The low-concurrency regime is not uniformly tight, so this audit does not establish a statewise bound on $\overline{\xi}_k$ or a numerical upper bound on terminal violations. Rather, it tests the observable planned-to-realized link in the analytical bridge. Appendix B.11.2 separately tests the terminal consequence of local allocation disagreements under a fixed SLOPE continuation, while Section 4.2 measures complete-policy joint-SLO performance directly.

B.6 Potential Area and Joint Deadlines

Theorem B.10 (Pathwise SLO-risk occupancy bound)**.** *Consider $n \geq 1$ requests starting from an empty system, all completing normally before H, with common allowances $d^T, d^E > 0$. Let u_i^P, u_i^D be the times of first input-phase completion and first generation of the final output token, and F_i, C_i the client first-response and completion times. Define*

$$\begin{gathered} h_i^T = (F_i - u_i^P)_+, \qquad h_i^E = (C_i - u_i^D)_+, \\ J_i = \mathbf{1}\{F_i - a_i \leq d^T,\ C_i - a_i \leq d^E\}, \qquad v_n = \frac{1}{n}\sum_{i=1}^{n}(1 - J_i), \\ \overline{h}_n = \frac{1}{n}\sum_{i=1}^{n}\left(\frac{h_i^T}{d^T} + \frac{h_i^E}{d^E}\right). \end{gathered} \tag{B.21}$$

Let

$$R_{\mathrm{slo}}(u) = \frac{n_P(u)}{d^T} + \frac{n_D(u)}{d^E},$$

where $n_P(u)$ and $n_D(u)$ count requests with unfinished input and unfinished output, respectively. Every service trajectory satisfies

$$\begin{aligned} v_n &\leq \min\left\{1, \frac{1}{n}\int_0^H R_{\mathrm{slo}}(u)\,\mathrm{d}u + \overline{h}_n\right\} \\ &\leq \min\left\{1, \frac{1}{n}\int_0^H \overline{Q}_\lambda(u)\,\mathrm{d}u + \overline{h}_n\right\}, \qquad \lambda \geq 0. \end{aligned} \tag{B.22}$$

For the general potential Q in Eq. (B.9)*, the second line may be replaced by*

$$v_n \leq \min\left\{1, \frac{c_Q}{n}\int_0^H Q(u)\,\mathrm{d}u + \overline{h}_n\right\}, \qquad c_Q = \max\left\{\frac{1}{\gamma d^T}, \frac{1}{\alpha d^E}\right\}. \tag{B.23}$$

Under the SLO-calibrated choice in Eq. (B.10)*, $c_Q = 1/(w_F d^T)$ and $c_Q Q = \overline{Q}_\lambda$.*

Proof. Integrating unfinished-request indicators gives the occupancy identities

$$\sum_i (u_i^P - a_i) = \int_0^H n_P(u)\,\mathrm{d}u, \qquad \sum_i (u_i^D - a_i) = \int_0^H n_D(u)\,\mathrm{d}u.$$

With $\mathrm{TTFT}_i = F_i - a_i$ and $\mathrm{E2E}_i = C_i - a_i$, the union bound and nonnegative delays yield

$$1 - J_i \leq \mathbf{1}\{\mathrm{TTFT}_i > d^T\} + \mathbf{1}\{\mathrm{E2E}_i > d^E\} \leq \frac{\mathrm{TTFT}_i}{d^T} + \frac{\mathrm{E2E}_i}{d^E}.$$

Using $\mathrm{TTFT}_i \leq u_i^P - a_i + h_i^T$ and $\mathrm{E2E}_i \leq u_i^D - a_i + h_i^E$, then summing over requests, gives the first line of Eq. (B.22). Eq. (B.12) gives $R_{\mathrm{slo}} \leq \overline{Q}_\lambda$, proving the second line. For the general potential, every unfinished output has $r_i^D \geq 1$, so $Q(u) \geq \gamma n_P(u) + \alpha n_D(u)$ and therefore $R_{\mathrm{slo}}(u) \leq c_Q Q(u)$. This proves Eq. (B.23). Finally, substituting Eq. (B.10) gives $\alpha d^E = \gamma d^T = w_F d^T$, hence $c_Q = 1/(w_F d^T)$ and Eq. (B.11) gives $c_Q Q = \overline{Q}_\lambda$. □

The boundary delays preserve differences between engine milestones and client delivery. Heterogeneous allowances admit a conservative version using $d^T = \min_i d_i^T$ and $d^E = \min_i d_i^E$. The first line of Eq. (B.22) is weight-free and does not require stationary arrivals or independent lengths; the dense potential is an upper envelope introduced only to obtain a useful local control signal.

B.7 Online Service Rates and Cumulative Risk

For the joint-SLO interpretation in this section, take the SLO-calibrated reference $d_0 = d_0^{\mathrm{slo}}$ from Eq. (B.10). Then $c_{\mathrm{slo}} = 1/(w_F d^T)$ and $c_{\mathrm{slo}} Q = \overline{Q}_\lambda$. We bound potential area using conditional frame moments, allowing new requests to arrive during execution. Assumptions cover the complete scheduler, including ELP mixed iterations and all other branches.

Assumption B.1 (Conditional arrival and net-change moments)**.** Finite constants $\Lambda, \nu \geq 0$ satisfy, at every frame,

$$\mathbb{E}_k A_k \leq \Lambda T_k, \qquad \mathbb{E}_k[(A_k - S_k)^2] \leq \nu T_k, \qquad T_k = \mathbb{E}_k \tau_k. \tag{B.24}$$

Λ has units of potential work per second, not requests per second.

Assumption B.2 (Service margin at high backlog)**.** There are $q_0 > 0$ and $\kappa > 0$ such that whenever $Q_k \geq q_0$,

$$\mathbb{E}_k S_k \geq (\Lambda + \kappa) T_k. \tag{B.25}$$

Assumption B.3 (Within-frame area)**.** A finite $h \geq 0$ satisfies

$$\mathbb{E}_k \left[\int_{s_k}^{s_{k+1}} Q(u)\, \mathrm{d}u \right] \leq Q_k T_k + h T_k. \tag{B.26}$$

Corollary B.8 provides a sufficient condition linking selection quality to Assumption B.2. Suppose every eligible high-backlog ELP state has $\max_a V_k(a) \geq v_*$, $\varepsilon_T \leq \varepsilon$, and $\sup_a \xi_k(a) \leq \xi$, while other branches provide actual service rate at least μ_{fb}. Then one may take

$$\kappa = \min\{e^{-\varepsilon} v_* - \xi,\ \mu_{\mathrm{fb}}\} - \Lambda, \qquad \kappa > 0. \tag{B.27}$$

Here v_* represents the capability of the request plan and valid candidates, and μ_{fb} covers pure-prefill, pure-decode, pressure, and fallback branches. Candidate capability, prediction error, and execution discrepancy enter the same margin.

Theorem B.11 (Cumulative potential bound for variable-length frames)**.** *Suppose Eq.* (B.13)*, Assumptions B.1–B.3, and $\mathbb{E}Q_0^2 < \infty$ hold. For a deterministic frame count K with $\mathbb{E}s_K < \infty$,*

$$\mathbb{E} \int_0^{s_K} Q(u)\, \mathrm{d}u \leq b_\kappa \mathbb{E}s_K + \frac{\mathbb{E}Q_0^2}{2\kappa}, \qquad b_\kappa = \frac{\nu}{2\kappa} + q_0 \left(1 + \frac{\Lambda}{\kappa}\right) + h. \tag{B.28}$$

The bound also holds for a stopping frame count K relative to the frame history, provided $\mathbb{E}s_K < \infty$.

Proof. Set $\mathcal{L}(Q) = Q^2/2$. The recursion gives

$$\Delta\mathcal{L}_k = Q_k(A_k - S_k) + \tfrac{1}{2}(A_k - S_k)^2.$$

For $Q_k \geq q_0$, the moment and margin assumptions give $\mathbb{E}_k \Delta\mathcal{L}_k \leq (\nu/2 - \kappa Q_k) T_k$. For $Q_k < q_0$, nonnegative service gives $\mathbb{E}_k \Delta\mathcal{L}_k \leq (\nu/2 + \Lambda Q_k) T_k$. With $c_0 = \nu/2 + (\Lambda + \kappa) q_0$, both cases imply

$$\mathbb{E}_k \Delta\mathcal{L}_k \leq (c_0 - \kappa Q_k) T_k. \tag{B.29}$$

For deterministic K, sum and take expectations, discarding the nonnegative terminal squared potential:

$$\kappa \sum_{k<K} \mathbb{E}[Q_k T_k] \leq c_0 \sum_{k<K} \mathbb{E}T_k + \tfrac{1}{2}\mathbb{E}Q_0^2.$$

Conditional expectation gives $\mathbb{E}[Q_k T_k] = \mathbb{E}[Q_k \tau_k]$ and $\sum_{k<K} \mathbb{E}T_k = \mathbb{E}s_K$. Summing Assumption B.3 proves the bound. Finite-step second-moment integrability follows recursively from the initial moment, Assumption B.1, and $\mathbb{E}s_K < \infty$.

For a stopping frame count, first sum to $K \wedge m$. Since $\mathbf{1}\{k < K\}$ is $\mathcal{F}_k$-measurable, the same conditional-expectation argument applies. Discard the nonnegative terminal term, let $m \to \infty$, and use monotone convergence of nonnegative area together with $\mathbb{E}s_K < \infty$. □

Corollary B.12 (Joint violations for a finite request set)**.** *In Theorem B.10's model, let $H = s_K$ be an ending boundary after all client completions, with stopping frame count K and $\mathbb{E}H < \infty$. If Assumptions B.1–B.3 hold throughout arrivals and drain, then*

$$\mathbb{E}v_n \le \min\left\{1, \frac{c_{\mathrm{slo}}}{n} b_\kappa \mathbb{E}H + \mathbb{E}\overline{h}_n\right\}. \tag{B.30}$$

Proof. The initially empty system has $Q_0 = 0$. Take expectations in the untruncated pathwise bound, substitute the stopping-time version of Theorem B.11, and use $\mathbb{E}v_n \le 1$. □

Finite expected completion time is an independent condition, particularly in low-backlog states unconstrained by Assumption B.2. A nonempty initial system retains the initial-value term in Eq. (B.28).

Corollary B.13 (Stationary joint violation probability)**.** *In addition to Assumptions B.1–B.3, suppose the boundary process is stationary and ergodic, $0 < \mathbb{E}\tau_0 < \infty$, $\mathbb{E}Q_0^2 < \infty$, and $s_K \to \infty$. Suppose arrival rate $\lambda > 0$, relevant first delay moments are finite, and occupancy relations between time averages and typical arriving requests hold. Then*

$$p_{\mathrm{joint}} \le \min\left\{1, \frac{c_{\mathrm{slo}}}{\lambda} b_\kappa + \overline{h}\right\}, \qquad \overline{h} = \mathbb{E}^{\mathrm{arr}}\left[\frac{h^T}{d^T} + \frac{h^E}{d^E}\right], \tag{B.31}$$

where $\mathbb{E}^{\mathrm{arr}}$ denotes expectation for a typical arriving request.

Proof. Stationarity gives $\mathbb{E}\Delta\mathcal{L}_0 = 0$. Taking expectations in Eq. (B.29) and applying Assumption B.3,

$$\mathbb{E}\int_{s_0}^{s_1} Q(u)\mathrm{d}u \le \left(\frac{c_0}{\kappa} + h\right)\mathbb{E}\tau_0 = b_\kappa \mathbb{E}\tau_0.$$

The ratio of accumulated area to elapsed time under stationary ergodic frames yields $\mathbb{E}^{\mathrm{time}}Q \le b_\kappa$. Occupancy relations give $\mathbb{E}^{\mathrm{time}}n_P = \lambda\mathbb{E}^{\mathrm{arr}}(u^P - a)$ and $\mathbb{E}^{\mathrm{time}}n_D = \lambda\mathbb{E}^{\mathrm{arr}}(u^D - a)$. The indicator inequality from Theorem B.10 and $Q \ge \gamma n_P + \alpha n_D$ therefore imply

$$p_{\mathrm{joint}} \le \frac{1}{\lambda}\mathbb{E}^{\mathrm{time}}\left(\frac{n_P}{d^T} + \frac{n_D}{d^E}\right) + \overline{h} \le \frac{c_{\mathrm{slo}}}{\lambda} b_\kappa + \overline{h}.$$

Taking the minimum with one proves the claim. □

Optimization interpretation. For a fixed state, the service term in normalized quadratic drift is $-Q_k\mu_k(a)$. Eq. (B.16) upper-bounds it by $-Q_kV_k(a) + Q_k\xi_k(a)$. Since Q_k is common across candidates, increasing $V_k(a)$ improves this service term. Under the SLO calibration in Eq. (B.10), multiplying by c_{slo} rewrites the same statement in the d^T, d^E units of Corollary B.9; no additional progress–deadline calibration coefficient is introduced. The common ν controls second-order changes, while duration-contrast error and the explicit local residual quantify the proxy's approximation to the drift term.

Holding Λ, ν, q_0, h fixed,

$$\frac{\mathrm{d}b_\kappa}{\mathrm{d}\kappa} = -\frac{\nu + 2q_0\Lambda}{2\kappa^2} \le 0. \tag{B.32}$$

Together with Eq. (B.27), improved candidate capability, smaller relative latency errors, or less progress overcount tighten the conditional risk bound until another branch becomes limiting. Finite-set comparisons also involve $\mathbb{E}H$, while stationary comparisons involve λ and delivery delays. This is a direction for a bound under a common workload envelope, not per-trajectory monotonicity of violations.

Proposition B.14 (Area bound with temporary service deficits)**.** *Retain Eq.* (B.13) *and Assumptions B.1, B.3. For fixed $\kappa > 0$, define*

$$Z_k^\kappa = [(\Lambda + \kappa)\tau_k - S_k]_+. \tag{B.33}$$

If $\mathbb{E}Q_0^2 < \infty$ and $H = s_K$ satisfies Theorem B.11's deterministic or stopping-frame conditions, then without Assumption B.2,

$$\mathbb{E}\int_0^H Q(u)\mathrm{d}u \le \left(\frac{\nu}{2\kappa} + h\right)\mathbb{E}H + \frac{\mathbb{E}Q_0^2}{2\kappa} + \frac{1}{\kappa}\mathbb{E}\sum_{k<K} Q_k Z_k^\kappa. \tag{B.34}$$

Proof. The moment assumption and quadratic expansion give

$$\begin{aligned}\mathbb{E}_k \Delta\mathcal{L}_k &\le \tfrac{\nu}{2} T_k - \kappa Q_k T_k + Q_k \mathbb{E}_k[(\Lambda+\kappa)\tau_k - S_k] \\ &\le \tfrac{\nu}{2} T_k - \kappa Q_k T_k + Q_k \mathbb{E}_k Z_k^{\kappa}.\end{aligned}$$

Sum as in Theorem B.11, discard the nonnegative terminal potential, and apply Assumption B.3. For stopping times, localize at $K \wedge m$ and use monotone convergence of nonnegative terms. An infinite deficit expectation leaves the inequality valid but uninformative as a finite bound. □

The additional term permits temporary service shortfalls caused by phase transitions, fallback, or recomputation, explicitly charging backlog-weighted deficits. Substitution into Theorem B.10 gives the corresponding finite-set violation bound. Average service rate alone does not determine positive-part deficit fluctuations.

B.8 Auxiliary Moment and Cost-Statistic Results

B.8.1 Sufficient conditions for the conditional moments

Proposition B.15 (Bounded busy frames and marked Poisson arrivals)**.** *Suppose busy frames satisfy $0 < \tau_{\min} \le \tau_k \le \tau_{\max} < \infty$ and, conditional on state and action, their durations are independent of future within-frame arrivals. Requests arrive as a rate-λ Poisson process with independent identically distributed nonnegative work marks Y satisfying $\mathbb{E}Y^2 < \infty$, and busy-frame service obeys $S_k \le S_{\max}$. Assumptions B.1, B.3 hold with*

$$\begin{aligned}&\Lambda = \lambda \mathbb{E}Y, \\ &\nu = 2\left(\lambda \mathbb{E}Y^2 + \Lambda^2 \tau_{\max} + \frac{S_{\max}^2}{\tau_{\min}}\right), \qquad h = \frac{\Lambda \tau_{\max}}{2}.\end{aligned} \tag{B.35}$$

Proof. Conditional on a busy-frame duration τ, compound Poisson arrivals satisfy

$$\mathbb{E}[A_k \mid \tau] = \Lambda\tau, \qquad \mathbb{E}[A_k^2 \mid \tau] = \lambda \mathbb{E}Y^2 \tau + \Lambda^2 \tau^2.$$

Apply $(A_k - S_k)^2 \le 2A_k^2 + 2S_k^2$, $\tau^2 \le \tau_{\max}\tau$, and $S_{\max}^2 \le (S_{\max}^2/\tau_{\min})\tau$, then average over duration. Within-frame service only decreases the potential, while new arrivals contribute mean area $\Lambda\tau^2/2$, proving the area condition.

For an empty system, waiting until the next arrival can be an idle frame. Then $S_k = 0$, within-frame area is zero, $\mathbb{E}_k \tau_k = 1/\lambda$, $\mathbb{E}_k A_k = \mathbb{E}Y$, and $\mathbb{E}_k A_k^2 = \mathbb{E}Y^2$; the same constants apply. □

One may take $Y = \alpha L + \beta p + \gamma \mathbf{1}\{p > 0\}$. If reusable prefixes are credited at arrival, p denotes the corresponding residual logical input. These are sufficient conditions; Theorem B.11 does not require Poisson arrivals or universal duration/arrival independence.

B.8.2 Timing scope and log regression

If labels represent mean execution interval C_a but the full mean frame is $T_a = C_a + O_a$, $O_a \ge 0$, then

$$\log \widehat{T}_a - \log T_a = (\log \widehat{T}_a - \log C_a) - \log(1 + O_a/C_a).$$

For $r_{\min} \le O_a/C_a \le r_{\max}$, subadditivity of oscillation gives

$$\operatorname{osc}\left(\log \frac{\widehat{T}_a}{T_a}\right) \le \operatorname{osc}\left(\log \frac{\widehat{T}_a}{C_a}\right) + \log \frac{1 + r_{\max}}{1 + r_{\min}}. \tag{B.36}$$

Thus both shorter-interval prediction error and candidate-dependent omitted overhead determine the full-frame error needed by Theorem B.3.

For log-duration regression, let $m_a = \mathbb{E}^a[\log \tau_a]$ and $j_a = \log \mathbb{E}^a[\tau_a] - m_a \ge 0$, with finite relevant expectations. Then

$$\ell_\theta(x, a) - \log T_a = \big(\ell_\theta(x, a) - m_a\big) - j_a. \tag{B.37}$$

Error oscillation also includes candidate-dependent Jensen gaps; a common gap does not change selection.

B.8.3 MEASUREMENT UNCERTAINTY

On an event simultaneously covering all mean costs, suppose $T_a \in [L_a, U_a]$ with $L_a > 0$. Then $\log \widehat{T}_a - \log U_a \le e_a \le \log \widehat{T}_a - \log L_a$, so

$$\operatorname{osc}(e) \le \max_a (\log \widehat{T}_a - \log L_a) - \min_a (\log \widehat{T}_a - \log U_a). \tag{B.38}$$

If simultaneous coverage has probability at least $1 - \delta$, Theorem B.3 with this upper bound on ε_T also holds with at least that probability. Interval construction depends on repetition tails and dependence; candidates must start from the same initial state.

B.9 OUTPUT-LENGTH PREDICTION AND DEMAND REPRESENTATION

Output-length prediction acts on request planning before candidate construction and is therefore analytically distinct from ELP's within-set cost prediction. The following result describes remaining-demand representation, length ordering, and anticipated KV demand.

Proposition B.16 (Propagation of length error)**.** *For unfinished generation, set $\widehat{r}_i^D = \max(1, \widehat{L}_i - y_i)$ and replace r_i^D in Eq.* (B.9) *by this proxy to obtain $\widehat{Q}$, keeping completed requests at zero. Then*

$$|\widehat{Q} - Q| \le \alpha \sum_{i \text{ still generating}} |\widehat{L}_i - L_i|. \tag{B.39}$$

If two requests each have length error at most δ and their true total lengths differ by more than 2δ, predicted-total-length ordering preserves their true order. For anticipated KV demand

$$K_i(L) = \left\lceil \frac{\min(L_{\max}, p_i + L)}{b_{\mathrm{KV}}} \right\rceil,$$

we have

$$|K_i(\widehat{L}_i) - K_i(L_i)| \le \left\lceil \frac{|\widehat{L}_i - L_i|}{b_{\mathrm{KV}}} \right\rceil. \tag{B.40}$$

Proof. An unfinished request satisfies $L_i - y_i \ge 1$. Since $x \mapsto \max(1, x)$ is 1-Lipschitz, $|\widehat{r}_i^D - r_i^D| \le |\widehat{L}_i - L_i|$; weighted summation proves the potential bound. If $L_j - L_i > 2\delta$, then $\widehat{L}_j - \widehat{L}_i \ge L_j - L_i - 2\delta > 0$. Finally, truncation is nonexpansive, and $|\lceil x \rceil - \lceil y \rceil| \le \lceil |x - y| \rceil$ proves the KV bound. □

$\widehat{Q}$ explains demand error and is not an additional online step. Length predictions can change order, admission, and candidate sets, affecting v_* in Eq. (B.27). Theorem B.3 instead describes ELP selection within a fixed set. Full planning consistency additionally depends on ordering gaps and resource thresholds.

B.10 CONDITIONAL CANDIDATE-CLASS TERMINAL CONSISTENCY

This section asks a stronger question than the observable bridge audited in Appendix B.11: under what additional conditions does repeated local selection attain the terminal optimum within SLOPE's planning and candidate-policy class? We establish an exact identity expressing terminal regret as accumulated Bellman action loss, derive a transfer bound that separates relative-cost error from progress–deadline calibration, and give a decision-margin condition under which the implemented selector attains the class optimum. The continuation values and margins here are analytical quantities; they are not estimated by the trace audit or first-action replay.

B.10.1 CAUSAL OBJECTIVE AND EVALUATION EPISODE

Fix the externally specified set of $n > 0$ requests and the workload law in Equation (3). Write

$$\mathcal{J}(\pi) = \frac{1}{n} \mathbb{E}_\pi \sum_{i=1}^{n} J_i.$$

Let h_k be the full observable history at a frozen decision boundary, including wall-clock time, request progress, available delivery feedback, and physical resource state. Using histories avoids assuming that ELP's reduced feature vector is a sufficient Markov state. Conditional expectations integrate over unobserved lengths, future arrivals, and execution and delivery randomness.

Let $\Omega(h)$ be the finite action set induced by the prescribed request plan and final resource-check mapping. The class Π_Ω consists of causal policies selecting among these candidates while retaining the same planning and branch rules. Pure-phase, pressure, and fallback branches are represented by singleton sets when their action is prescribed. Changing earlier budget choices can change later histories and plans; the class does not freeze an entire realized trajectory.

Let R_k count requests whose successful client completion occurs in frame k and whose actual TTFT and E2E both meet their allowances. Each request is counted at most once. This is an analytical reward and need not be immediately observable. Rejected requests and requests with an irreversible deadline failure earn no reward and remain in the denominator. A failed request can still affect other requests through its physical resource occupancy, which remains part of the transition.

Choose an observable stopping boundary K after which no further joint-success reward is possible. A boundary beyond a known finite evaluation-arrival window and all associated deadline allowances is one sufficient construction. This verdict horizon need not stop the serving engine. Pathwise,

$$\sum_{k<K} R_k = \sum_{i=1}^{n} J_i. \tag{B.41}$$

The equality accommodates heterogeneous allowances, failures, and delivery delays without a normal-completion assumption or a union bound.

Assumption B.4 (Finite causal evaluation)**.** The episode begins before any evaluation reward is earned. Conditional transition kernels and measurable maximizing actions exist. For all policies in Π_Ω, the verdict boundary is almost surely finite and $\sup_{\pi\in\Pi_\Omega} \mathbb{E}_\pi K < \infty$. The full-frame mean

$$T(h,a) = \mathbb{E}[\tau_k \mid h_k = h, a_k = a]$$

is finite and strictly positive. These conditions also hold after reachable histories. When a uniform time bound is used, assume $\sup_{\pi\in\Pi_\Omega} \mathbb{E}_\pi \sum_{k<K} \tau_k \le \overline{H} < \infty$.

These are episode and integrability conditions, not a positive service-margin assumption. The mean is conditioned on observable history here; the algebra of Theorem B.3 applies to this choice of conditional costs as well. One must not identify it without justification with either a shorter measured interval or a mean conditioned on the latent-length history in Appendix B.1. Appendix B.8.2 explains the timing and log-statistic distinctions.

B.10.2 An exact local-to-terminal regret identity

Define the optimal remaining-success value and its one-action continuation by

$$\begin{aligned} \mathcal{V}^*(h) &= \sup_{\pi\in\Pi_\Omega} \mathbb{E}_\pi \left[\sum_{j=k}^{K-1} R_j \,\middle|\, h_k = h \right], \\ \mathcal{Q}^*(h,a) &= \mathbb{E}[R_k + \mathcal{V}^*(h_{k+1}) \mid h, a]. \end{aligned} \tag{B.42}$$

Both are between zero and n, and $\mathcal{V}^*(h_K) = 0$. They are joint-system values: separately optimizing each request and summing would ignore shared-resource competition. The Bellman equation and local loss are

$$\mathcal{V}^*(h) = \max_{a\in\Omega(h)} \mathcal{Q}^*(h,a), \qquad d(h,a) = \mathcal{V}^*(h) - \mathcal{Q}^*(h,a) \ge 0. \tag{B.43}$$

For comparison with a progress rate, define

$$G^*(h,a) = \frac{\mathcal{Q}^*(h,a) - \mathcal{V}^*(h)}{T(h,a)} = -\frac{d(h,a)}{T(h,a)}. \tag{B.44}$$

Thus $\max_a G^*(h,a) = 0$. Subtracting the current value is essential: normalizing an uncentered positive success value by action-dependent time does not generally preserve a finite-set attainment

objective. The zero maximum makes division by any positive duration preserve Bellman-optimal actions for Equation (B.44). The transfer analysis below relates this centered rate to the implemented progress score.

Theorem B.17 (Terminal joint-attainment regret)**.** *Under Assumption B.4, let $\mathcal{J}^*_\Omega = \mathcal{V}^*(h_0)/n$. For every $\pi \in \Pi_\Omega$,*

$$\begin{aligned} n\big(\mathcal{J}^*_\Omega - \mathcal{J}(\pi)\big) &= \mathbb{E}_\pi \sum_{k<K} d(h_k, a_k) \\ &= \mathbb{E}_\pi \sum_{k<K} T(h_k, a_k)[-G^*(h_k, a_k)]. \end{aligned} \tag{B.45}$$

Consequently, selecting an action with $d(h, a) = 0$ at every reached history attains $\mathcal{J}^*_\Omega$.

Proof. Sum Equation (B.43)'s definition of d to $K \wedge m$ and take expectations. Since $\mathbf{1}\{k < K\}$ is measurable at the current boundary, conditional expectations telescope:

$$\mathbb{E}_\pi \sum_{k<K\wedge m} d(h_k, a_k) = \mathcal{V}^*(h_0) - \mathbb{E}_\pi \sum_{k<K\wedge m} R_k - \mathbb{E}_\pi \mathcal{V}^*(h_{K\wedge m}).$$

The nonnegative left side converges monotonically. Cumulative reward is bounded by n, and the bounded terminal-value term tends to zero almost surely because $K < \infty$. Dominated convergence and Equation (B.41) give the first identity; Equation (B.44) gives the second. □

This is the finite-episode, variable-frame form of the performance-difference argument used in policy improvement (Schulman et al., 2015). Expectations follow the evaluated policy's own histories, including its effects on future queues. A telescoping identity for an arbitrary potential alone would not justify greedy selection; the Bellman property supplies that step here.

B.10.3 Transfer from the implemented progress score

To avoid overloading the main text's predicted score U_t, write

$$V(h, a) = \frac{N(h, a)}{T(h, a)}, \qquad V_{\max}(h) = \max_{a\in\Omega(h)} V(h, a)$$

for the true progress rate, consistently with Equation (B.2). At a score-selected branch, suppose all retained numerators are positive and

$$\operatorname{osc}_a\big[\log \widehat{T}(h, a) - \log T(h, a)\big] \le \epsilon_T(h). \tag{B.46}$$

For any positive action-independent analytical scale $c(h)$, define

$$\delta(h; c) = \operatorname{osc}_{a\in\Omega(h)}[G^*(h, a) - c(h)V(h, a)]. \tag{B.47}$$

This oscillation measures candidate-dependent calibration between progress and deadline value, allowing a common additive offset. The transfer bound retains this calibration term explicitly. The analytical scale c leaves the scheduler's weights unchanged.

Theorem B.18 (Progress-score to attainment transfer)**.** *Under Assumption B.4 and Equation* (B.46)*, the implemented selector satisfies*

$$\mathcal{J}^*_\Omega - \mathcal{J}(\pi_{\mathrm{SLOPE}}) \le \frac{1}{n}\mathbb{E}_{\pi_{\mathrm{SLOPE}}} \sum_{k<K} T_k \left[c_k(1 - e^{-\epsilon_{T,k}})V_{\max,k} + \delta_k\right]. \tag{B.48}$$

Here $T_k = T(h_k, a_k)$, and the other quantities are evaluated at h_k. The bracket is set to zero on singleton branches, which require no positive-numerator assumption. If $cV_{\max} \le M$, $\epsilon_T(h) \le \epsilon$, and $\delta(h; c) \le \delta_0$ uniformly on the relevant histories, the right side is at most

$$\frac{\overline{H}}{n}\left[M(1 - e^{-\epsilon}) + \delta_0\right].$$

Proof. Theorem B.3 gives $V(h,\widehat{a}) \geq e^{-\epsilon_T(h)} V_{\max}(h)$. Write $e_a = G^*(h,a) - c(h)V(h,a)$ and choose a^* with $G^*(h,a^*) = 0$. Then

$$\begin{aligned} -G^*(h,\widehat{a}) &= c(h)\big[V(h,a^*) - V(h,\widehat{a})\big] + e_{a^*} - e_{\widehat{a}} \\ &\leq c(h)(1 - e^{-\epsilon_T(h)})V_{\max}(h) + \delta(h;c). \end{aligned}$$

Multiply by $T(h,\widehat{a})$ and apply Theorem B.17. At a singleton, the only available action has zero local loss. The uniform result uses $\mathbb{E}\sum_{k<K} T_k = \mathbb{E}\sum_{k<K} \tau_k \leq \overline{H}$. □

For the unrestricted objective in Equation (3), additionally retain

$$\Delta_{\text{plan}} = \sup_{\pi \in \Pi_{\text{online}}} \mathcal{J}(\pi) - \mathcal{J}_\Omega^* \geq 0.$$

Together, these terms distinguish candidate-class coverage, relative-cost accuracy, and progress–deadline calibration. The bound is informative for an attainment fraction when the combined upper bound is below one.

B.10.4 Decision margins and terminal consistency

Equation (B.47) is convenient for a numerical regret bound, but zero affine calibration is stronger than necessary for selecting the correct action. Define the analytically optimal action set

$$\mathcal{A}^*(h) = \{a \in \Omega(h) : d(h,a) = 0\},$$

and the separation between the best progress rate and all deadline-suboptimal candidates,

$$\Gamma(h) = \begin{cases} \log \dfrac{V_{\max}(h)}{\max_{a \notin \mathcal{A}^*(h)} V(h,a)}, & \Omega(h) \setminus \mathcal{A}^*(h) \neq \varnothing, \\ +\infty, & \Omega(h) = \mathcal{A}^*(h). \end{cases} \tag{B.49}$$

This is a log-score separation, not a request's temporal deadline slack. For a finite positive-score set, $\Gamma(h) > 0$ exactly when every true progress-rate maximizer is Bellman-optimal. It requires only a separation of decision classes, not agreement in the magnitudes of progress and deadline values.

For $\epsilon \geq 0$, define the possible-selection set and its loss envelope,

$$\begin{aligned} \mathcal{S}_\epsilon(h) &= \{a \in \Omega(h) : V(h,a) \geq e^{-\epsilon} V_{\max}(h)\}, \\ L(h,\epsilon) &= \max_{a \in \mathcal{S}_\epsilon(h)} d(h,a), \qquad d_{\max}(h) = \max_{a \in \Omega(h)} d(h,a). \end{aligned} \tag{B.50}$$

These sets are used only in the analysis. They do not filter runtime candidates.

The following theorem turns this decision separation into a quantitative tolerance to relative-cost error. Its premise is a statewise alignment condition involving analytical Bellman values.

Theorem B.19 (Margin-based consistency and localized regret)**.** *Under the conditions of Theorem B.18, if $\epsilon_T(h) < \Gamma(h)$ at every reached nonsingleton history, then $\mathcal{J}(\pi_{\text{SLOPE}}) = \mathcal{J}_\Omega^*$. More generally,*

$$\begin{aligned} n\big(\mathcal{J}_\Omega^* - \mathcal{J}(\pi_{\text{SLOPE}})\big) &\leq \mathbb{E}_{\pi_{\text{SLOPE}}} \sum_{k<K} L(h_k, \epsilon_T(h_k)) \\ &\leq \mathbb{E}_{\pi_{\text{SLOPE}}} \sum_{k<K} d_{\max}(h_k) \mathbf{1}\{\Gamma(h_k) \leq \epsilon_T(h_k)\}. \end{aligned} \tag{B.51}$$

Use $L = 0$ and $\Gamma = +\infty$ on singleton branches.

Proof. Relative-cost robustness implies $\widehat{a} \in \mathcal{S}_{\epsilon_T}(h)$. If a deadline-suboptimal candidate exists, its rate is at most $e^{-\Gamma(h)} V_{\max}(h)$. When $\epsilon_T(h) < \Gamma(h)$, this is strictly below the minimum rate allowed for the selected action. Hence $\widehat{a} \in \mathcal{A}^*(h)$ and its local loss is zero. If every action is optimal, this holds immediately. In general,

$$d(h,\widehat{a}) \leq L(h, \epsilon_T(h)) \leq d_{\max}(h) \mathbf{1}\{\Gamma(h) \leq \epsilon_T(h)\}.$$

Sum and apply Theorem B.17. □

The strict margin handles ties explicitly. Unlike a requirement that $\delta = 0$, this theorem permits arbitrary differences in value magnitudes as long as suboptimal actions remain sufficiently separated in score. The envelope $L(h, \epsilon)$ is nondecreasing in ϵ: $L(h, 0)$ captures possible losses among exact-cost score maximizers, while its increase accounts for additional actions admitted by time uncertainty. This is a pointwise statement at a fixed history, not a claim that changing ELP accuracy monotonically improves every policy trajectory.

B.10.5 Scope relative to the empirical bridge audit

The transfer and margin results quantify terminal performance within Π_Ω under the stated evaluation and decision conditions. The transfer bound accounts separately for progress–deadline calibration through $\delta(h; c)$ and relative-cost uncertainty through $\epsilon_T(h)$; the margin condition specifies when prediction accuracy preserves terminally optimal candidate selection.

Appendix B.11.1 instead audits the realized counterparts of the planned-to-service residual exposed by the SLO-normalized potential analysis. Appendix B.11.2 then performs controlled first-action interventions on real GPU states. Its outcome is the terminal joint success obtained after taking one alternative action and then following the same SLOPE continuation; it therefore does not estimate $\mathcal{Q}^*(h, a)$ or Bellman regret. Section 4.2 separately evaluates the complete policy across serving workloads. The empirical audits and the Bellman results are thus complementary: the former test the observable links used to interpret the runtime controller, while the latter characterize stronger sufficient conditions for terminal consistency within the candidate-policy class.

B.11 Empirical Audit of the Local-to-Terminal Bridge

Section 4.5 reports the main audit definitions and aggregate findings. Here we provide the complete measurement definitions, per-stream and concurrency-regime diagnostics, conservation checks, saved-state replay protocol, and per-state outcomes for the two measurable links: (i) whether the planned local decrement is realized by execution, and (ii) whether changing one local allocation can affect terminal joint service under a common continuation. We use Hunyuan-A13B-Instruct-GPTQ-Int4/SCROLLS on two A40 GPUs with tensor parallelism two. The experiments use the $20\times$ TTFT/E2E allowances and an arrival rate of 1.5 × capacity; arrivals last 30 seconds and accepted requests are then drained. Unless stated otherwise, the timing boundary for terminal success is engine-output availability.

B.11.1 Trace audit of planned versus realized service

Sample diagnostic. For a normal mixed decision k, let d_k^D, W_k, f_k denote the planned decode coverage count, useful prefill work, and designated phase-completion marker, and let D_k, P_k, F_k denote their realized first-time service counterparts reconstructed from execution feedback. We audit the sample positive-part discrepancy

$$\widehat{\Delta}_k = \left[\frac{(d_0/n_k^D)d_k^D - D_k}{d^E}\right]_+ + \lambda\frac{[W_k - P_k]_+}{Bd^T} + \frac{[f_k - F_k]_+}{d^T}. \tag{B.52}$$

This is a one-realization counterpart of the terms entering the analytical residual in Corollary B.7; it is not the conditional-expectation quantity $\overline{\xi}_k(a)$. We also report the backlog-weighted aggregate

$$R_\Delta = \frac{\sum_k \overline{Q}_k \widehat{\Delta}_k}{\sum_k \overline{Q}_k \overline{N}_k}, \qquad \overline{Q}_k := \overline{Q}_\lambda(s_k), \tag{B.53}$$

computed separately within each request stream before cross-seed aggregation. The offline reconstruction of $\overline{Q}_k$ uses completed-trace output lengths only for analysis; no such information is supplied to the online scheduler.

The $20\times$ allowances are $d^T = 28.14078203$ s and $d^E = 70.96050758$ s, giving $d_0 = 5.04325058$ for the default weights. Across seeds 42, 43, and 44, the audit covers 107 requests, 4,747 execution frames, and all 329 normal mixed ELP decisions. We independently verify conservation of generated first-time tokens, effective first-time input work (with prefix-reuse credit tracked separately), and input-phase completions. The per-frame potential recursion also balances to numerical precision.

Table B.2: Three-stream execution audit. R_Δ is computed within each seed using Eq. (B.53); the reported cross-seed mean is an equal-seed mean, while the percentile columns are within-seed descriptive statistics over audited decisions.

Seed	Requests	Frames	Mixed decisions	Median $\widehat{\Delta}/\overline{N}$	p95	R_Δ
42	36	2334	90	0.00%	78.22%	6.18%
43	35	1323	112	0.00%	58.87%	2.88%
44	36	1090	127	0.00%	39.53%	0.95%

The equal-seed mean of R_Δ is $3.34\% \pm 2.64\%$ (mean $\pm$ sample SD over the three request streams). Planned prefill work matches realized first-time effective input progress in all 329 audited decisions. We observe no positive residual from frozen-set decode under-delivery, prefill execution, or phase completion. All positive residuals are therefore attributable to the decode SLO-normalization term and occur when $n_k^D < d_0$.

Table B.3: Pooled diagnostic by decode-concurrency regime. These are descriptive statistics after pooling audited decisions and are not inferential estimates that treat frames as independent samples. The pooled R_Δ differs from the equal-seed mean in Table B.2.

Regime	Decisions	Median	p95	Pooled R_Δ	Zero residual
All mixed decisions	329	0.00%	58.87%	3.00%	261/329
$n_k^D < d_0$	71	20.18%	78.22%	21.34%	3/71
$n_k^D \geq d_0$	258	0.00%	0.00%	0.00%	258/258

The regime split matches the structural prediction in Eq. (B.19): when $n_k^D \geq d_0$, the reference-concurrency normalization cannot itself create positive overcount, and all 258 audited decisions in that regime have zero sample residual. The low-concurrency tail is not uniformly tight, so the aggregate audit should not be interpreted as a statewise error bound. Its main role is to show that, in these traces, the planned-to-realized gap is small in backlog-weighted aggregate and is localized to the normalization effect already exposed by the analysis rather than to unmodeled execution-side service loss.

B.11.2 Controlled first-action replay

Protocol. We next test whether a local allocation disagreement can change terminal joint service while holding the continuation policy fixed. For each of seeds 42, 43, and 44, we use two pre-specified real mixed-phase states—one from the arrival phase and one from drain—at which SLOPE and progress-only choose different first actions. Each saved state preserves the request plan, candidate mapping, queues, KV occupancy, and policy state. From the same parent state and future input trace, one branch executes the SLOPE-selected first action and the other executes the progress-only first action; after that single overridden frame, both branches resume the native SLOPE policy until all requests drain. Each action is executed twice, with the execution order reversed across the two technical repeats, yielding 24 completed branches over six states.

All completed branches pass CPU/GPU restoration and single-frame override checks. Restoring a state does not change SLOPE's default candidate under the recovered cost ordering, the first-frame effective prefill service matches the plan in all 24 branches, and no branch exhibits frozen-decode-set first-frame service under-delivery.

Table B.4: Controlled first-action replay. Entries give the two terminal joint-success counts for each first action. Only the first action differs; all subsequent decisions use the same SLOPE continuation.

Seed	Phase	SLOPE repeats	Progress-only repeats	Mean difference
42	arrival	19; 19 / 36	17; 19 / 36	+1.0
42	drain	19; 19 / 36	19; 18 / 36	+0.5
43	arrival	20; 19 / 35	19; 19 / 35	+0.5
43	drain	19; 19 / 35	19; 19 / 35	0.0
44	arrival	27; 27 / 36	24; 26 / 36	+2.0
44	drain	27; 27 / 36	27; 25 / 36	+1.0

The SLOPE-selected first action has a higher two-repeat mean terminal joint-success count in five of the six pre-specified disagreement states and ties in one; it is not lower in any state. Averaging the two state differences within each request stream gives $+0.75$, $+0.25$, and $+1.50$ successful requests for seeds 42, 43, and 44, respectively. The equal-seed mean is therefore $+0.83$ successful requests per replayed state.

These replays are controlled local interventions rather than independent end-to-end workload repetitions. The six states are conditioned on SLOPE and progress-only disagreeing, so the experiment tests the consequence of such a disagreement, not how often disagreements occur in normal execution. The two executions per action are technical repeats, and the three request streams are the independent sampling units.

C Implementation and Training Details

C.1 Output-Length Predictor

Architecture. We use a discrete survival predictor to estimate generation demand from the input text before a request completes. Token and positional embeddings pass through a two-layer Transformer encoder, followed by attention pooling to form a request representation. A prediction head produces the conditional termination probability for each length interval. The encoder has a hidden dimension of 192 and four attention heads. For long inputs, we retain the beginning and end of the token sequence, limiting the encoded length to 768 tokens. Let $h_j(x)$ denote the conditional termination probability in interval j. The probability that the output length exceeds boundary b_j is

$$S_j(x) = \prod_{r=1}^{j} \big(1 - h_r(x)\big). \tag{C.1}$$

This construction ensures that survival probabilities are nonincreasing across length boundaries. The scheduler converts the predicted distribution into a total output-length estimate by weighting representative interval lengths, supporting decode SJF ordering and anticipated KV demand.

Training objective. Training labels come from generations produced by the target LLM. Naturally completed samples identify the termination interval. Samples that reach the generation cap are treated as right-censored observations: they constrain nontermination only over observed intervals, avoiding the treatment of truncated lengths as complete outputs. Training combines an event-weighted survival negative log-likelihood with a Brier loss on survival probabilities to learn both interval termination probabilities and their cumulative relationship. Training and validation sets are split by conversation. We optimize with AdamW and select the checkpoint with the lowest validation survival negative log-likelihood. Predictor weights remain frozen online.

Training setup and cost. Training requests and online test requests are disjoint. The learning rate is 3×10^{-4} and the batch size is 128. The predictor requires a single offline training run before deploying the corresponding LLM, taking approximately 2.5 hours on one GPU.

C.2 Candidate Construction and Execution

Demand-based planning keys. For observable input length p_i and computed progress $c_{i,t}$, a monotone latency interpolant F_P fitted to offline measurements gives

$$\widehat{r}^{P}_{i,t} = \max\{0, F_P(p_i) - F_P(c_{i,t})\}. \tag{C.2}$$

Prefill requests are ordered by this key in ascending order, retaining resident-partial-request management and aging. The output predictor uses existing input tokens and structural features u_i to estimate total output length once; with $y_{i,t}$ generated tokens, the remaining-demand proxy is

$$\widehat{L}_i = H_\phi(u_i), \qquad \widehat{r}^{D}_{i,t} = \max\{1, \widehat{L}_i - y_{i,t}\}. \tag{C.3}$$

Under normal pressure, predicted total length supports decode SJF ordering and, together with input length, anticipated KV demand and admission. Pressure handling uses remaining demand in E2E slack and resource-release cost decisions. These predictions affect request planning, not the numerator definition in Eq. (15). Prediction runs asynchronously; the configured maximum output length is used until a prediction returns.

Common planning and progress calculation. To isolate differences in prefill investment, the scheduler fixes the currently eligible requests and their order, constructing candidates with different budgets under a common request plan. Prefill budgets are allocated in SRPT order according to remaining processing cost, with effective input progress accounting for prefix caching. The phase-completion marker refers to the highest-ranked resident prefill request that has already made input progress; it is activated only when a candidate covers that request's remaining input. Decode coverage uses the common set awaiting service as its denominator and the number of requests that the candidate plans to advance as its numerator, ensuring a consistent evaluation scale across candidates.

Candidate comparison and execution. Each candidate ratio is translated into per-request quotas. Candidates producing identical quotas are merged, and empty plans are excluded. The scheduler selects a candidate using the progress-rate score in Eq. (12), preferring the smaller ratio in a tie. The selected plan passes through the engine's sequence and physical KV constraints before execution, and actual service progress updates the next state. Appendix B further analyzes discrepancies between candidate plans and actual execution caused by resource clipping.

Candidate grid and final feasibility. The configured ratios are $\{0.05, 0.10, \ldots, 1.00\}$. Candidates respect native residual budgets and eligibility, and prefix reuse is queried read-only. Ordered request-quota plans are deduplicated, retaining the smallest ratio as representative and excluding empty plans. Following selection, the initial plan $\widetilde{z}_t(\rho_t)$ is mapped to the executable action

$$z_t = \mathcal{R}_t(\widetilde{z}_t(\rho_t)) \in \mathcal{A}(s_t). \tag{C.4}$$

Anticipated demand informs admission, whereas physical-block checks enforce feasibility. Final clipping can reduce planned decode coverage, input work, or phase completion. Pressure branches shrink prefill quotas, critical-pressure mixed branches suspend prefill, and pure-phase or empty-candidate branches follow their established controls.

Algorithm C.1 SLOPE resource allocation at each iteration

Require: Current request/resource state; output predictor H; ELP

1: Read input/generation progress and available predictions; use the configured fallback for missing predictions.
2: Apply two-phase request ordering, anticipated-resource admission, and pressure handling.
3: Freeze the request-planning context at the chunk-allocation boundary.
4: Construct candidate prefill budgets; deduplicate identical ordered quota plans and remove empty plans.
5: For each valid candidate, compute η^D, W, f, and N, and predict its execution time with ELP.
6: Select the candidate maximizing $N/\widehat{T}$, breaking ties by the smaller representative ratio.
7: Apply final active-set and physical KV checks, then execute the feasible quotas.
8: Update request progress and service feedback.

C.3 Execution Latency Predictor and Cost Learning

ELP takes five scalar inputs: decode concurrency pressure, effective prefill backlog pressure, observed service pressure, the previous ratio, and the candidate ratio. Service pressure combines normalized prefill waiting with observed request-level average TPOT. These features summarize the aggregate state. Output-length predictions affect resource competition through request planning and subsequent state changes, rather than serving as direct ELP inputs.

Formally, the prediction and state vector are

$$\widehat{T}_t(\rho) = \exp \ell_\theta(x_t, \rho), \qquad x_t = (p_t^D, p_t^P, p_t^S, \rho_{t-1}). \tag{C.5}$$

The components p_t^D, p_t^P, p_t^S denote decode concurrency pressure, effective prefill backlog pressure, and service pressure. Pressure normalization constants are distinct from TTFT/E2E evaluation thresholds. For a fixed request plan and candidate set, the score comparison is equivalently

$$\begin{aligned} U_t(\rho_b) > U_t(\rho_a) \quad \Longleftrightarrow \quad & \ell_\theta(x_t, \rho_b) - \ell_\theta(x_t, \rho_a) \\ & < \log N_t(\rho_b) - \log N_t(\rho_a). \end{aligned} \tag{C.6}$$

The predicted log-time difference is compared with the log-progress increase, motivating supervision of relative latency magnitudes rather than latency ordering alone.

ELP consists of one hidden layer with 16 Tanh units and a scalar output layer, totaling 113 parameters. The network predicts standardized log-time, which is converted back to ℓ_θ before exponentiation. It is trained offline and frozen online; execution feedback updates subsequent state features.

Training objective and candidate pairs. Let y_i be a measured log-time label and $\mathcal{P}$ the set of same-state candidate pairs. With $\ell_i = \ell_\theta(x_i, \rho_i)$ and $\Delta_{ab}\ell_s = \ell_\theta(x_s, \rho_b) - \ell_\theta(x_s, \rho_a)$, ELP minimizes

$$\begin{aligned} \mathcal{L}_{\text{abs}} &= \sum_i a_i \left(\frac{\ell_i - y_i}{\sigma_y} \right)^2, \\ \mathcal{L}_{\text{rel}} &= \sum_{(s,a,b)\in\mathcal{P}} b_{sab} \left(\frac{\Delta_{ab}\ell_s - (y_{sb} - y_{sa})}{\sigma_y} \right)^2, \\ \mathcal{L} &= \mathcal{L}_{\text{abs}} + \lambda_{\text{rel}}\mathcal{L}_{\text{rel}}, \qquad \lambda_{\text{rel}} = 1. \end{aligned} \tag{C.7}$$

Here σ_y is the label-normalization scale, and a_i and b_{sab} each sum to one over their respective sets. The relative term directly supervises the identified fixed-plan duration contrast, while the absolute term calibrates the latency level. Measured candidates are ordered by budget, and adjacent candidates form a fixed pair set used in every full-batch update. Repeated candidate timings are averaged in log space. Absolute weights balance trajectories, available service phases, and samples; relative weights balance trajectories, states, and pairs within each budget range. Low-, middle-, and high-budget pairs receive total weights 0.4, 0.4, and 0.2, respectively. The low range contains $0.05 \to 0.10$ and $0.10 \to 0.15$; the middle range contains $0.15 \to 0.30$ and $0.30 \to 0.55$; higher adjacent pairs form the high range. Relative-cost training and held-out evaluation use disjoint trajectories. Starting from an absolute-log-time model, training uses 1,200 full-batch Adam updates with learning rate 0.01 and weight decay 0.001; the final checkpoint is retained.

Low-budget gradient protection. Before each Adam update, we remove the component of the aggregate gradient from the absolute and other relative losses that conflicts with the low-budget relative-loss gradient. The protected aggregate is then added to the low-budget gradient, coordinating the supervision of cost growth across budget ranges. This is the protective projection removed together with relative supervision in the absolute-only control (Appendix D.5).

For the matched candidate-selection study, offline fine-tuning of three seeds in parallel takes approximately 9.1 seconds on an Intel Xeon Platinum 8481C CPU, with one compute thread per process. This timing excludes latency-label collection and initial absolute-model training. Training labels time the historical execution step, including its scheduling phase; Appendix B.8.2 relates this interval to the analytical scheduling frame.

C.4 Runtime Shaping and Reference-Concurrency Parameters

All reported SLOPE experiments use the shared default coefficients

$$(w_D, w_P, w_F) = \left(\frac{1}{2}, \frac{1}{4}, \frac{1}{4} \right),$$

except when they are explicitly varied in Appendix D.6. Substituting these values into Eq. (15) gives

$$N_t(\rho) = \frac{1}{2}\eta_t^D(\rho) + \frac{W_t(\rho)}{4B} + \frac{1}{4}f_t(\rho).$$

The coefficients define the local debt decrement, but the SLO-normalized interpretation in Eq. (8) shows that their common scale is irrelevant. The two nontrivial ratios are

$$\lambda = \frac{w_P}{w_F}, \qquad d_0 = \frac{w_D}{w_F}\frac{d^E}{d^T}.$$

Thus w_P/w_F controls the amount of nonnegative continuous-prefill shaping added to the SLO-risk certificate, while w_D/w_F sets the reference decode concurrency at which planned decode coverage is calibrated to E2E-normalized generation service. For the default coefficients, $\lambda = 1$ and $d_0 = 2d^E/d^T$.

Common positive scaling preserves candidate rankings within a fixed planning context. Changing the two ratios changes the dense controller and its reference-concurrency normalization, not a claimed probability calibration to terminal success. The relative-cost selection guarantee holds for every fixed positive configuration with positive candidate progress, and the potential analysis retains an explicit residual for mismatch between the local decrement and realized SLO-normalized service. Appendix D.6 evaluates terminal joint-SLO robustness over several induced (λ, d_0) parameterizations.

C.5 Online Information and Resource Feasibility

Observable history $\mathcal{H}_t$ contains arrived requests and their inputs, computation and generation progress, queues, resources, and prior execution times and feedback. Predictions may use existing inputs and offline-trained models, but future arrivals and retrospectively observed final output lengths are not policy inputs. Requests with unfinished input form a subset of those awaiting their first delivered token: input completion and client delivery need not coincide.

An action z_t specifies requests, nonnegative integer computation-token quotas $k_{i,t}$, and native admission or suspension decisions. With engine token limit B_{eng}, sequence limit M, and physical KV-block capacity $K_{\max}$, feasible actions satisfy

$$\mathcal{A}(s_t) = \left\{ z : \begin{array}{l} \sum_i k_i(z) \le B_{\mathrm{eng}}, \\ |\{i : k_i(z) > 0\}| \le M, \\ K(s_t, z) \le K_{\max}, \\ \mathcal{C}_{\mathrm{phase}}(s_t, z) = 1 \end{array} \right\}. \tag{C.8}$$

$K(s_t, z)$ accounts for retained and newly allocated physical blocks, including prefix sharing; it is not the sum of predicted sequence lengths. $\mathcal{C}_{\mathrm{phase}}$ permits only arrived, nonterminated requests, respects input-processing and autoregressive dependencies, and includes required recomputation. Ordinary decode produces at most one new output token per served request per iteration, whereas computation quotas count actual engine work.

The latency allowances d_i^T, d_i^E correspond to absolute deadlines $a_i + d_i^T, a_i + d_i^E$; common thresholds are a special case. The distribution $\mathcal{W}$ in Eq. (3) describes workload and execution randomness for fixed model and generation settings and need not be known by the scheduler. Evaluation follows the externally specified arrival set through successful completion or terminal failure, retaining rejected and failed requests in its denominator.

C.6 Mechanism Comparison and Baseline Coverage

We select JITServe (Zhang et al., 2026) and QoServe (Goel et al., 2026) as representative recent SLO-aware serving systems that address demand- and payoff-aware service allocation and deadline-aware dynamic prefill allocation, respectively. Both mechanisms are directly relevant to the online resource-allocation problem studied by SLOPE and provide complementary comparisons for evaluating joint SLO performance. Table C.1 summarizes their decision objects, use of cost information, and allocation principles.

Complementary system comparisons. JITServe estimates service demand from imprecise request information and combines serving-bandwidth allocation with batch organization to improve goodput. Comparison with JITServe therefore evaluates SLOPE against demand- and payoff-aware scheduling. QoServe adapts prefill chunks using deadline slack and coordinates service through hybrid prioritization and overload handling. Comparison with QoServe therefore evaluates SLOPE against deadline-aware dynamic allocation. Both systems adjust resource use according to online state, providing system-level comparisons beyond fixed-order or fixed-budget policies.

SLOPE's allocation criterion. SLOPE differs in the decision level and criterion used to compare resource investments. At each candidate comparison, it fixes request eligibility and ordering, constructs alternative prefill budgets under the same request plan, and evaluates their two-phase service progress relative to execution time. For two candidates with positive progress and execution time,

Table C.1: Scheduling mechanisms of JITServe, QoServe, and SLOPE. The JITServe and QoServe entries describe their published mechanisms.

Method	Decision object	Demand and state information	Role of execution cost	Allocation principle
JITServe	Per-request serving bandwidth and batch composition	Estimated remaining generation demand, remaining time allowance, and goodput payoff	Estimates serving-bandwidth requirements and informs payoff–resource trade-offs	Prioritizes goodput payoff per unit bandwidth and groups requests by input length
QoServe	Prefill chunks, request priorities, and overload handling	Deadline slack, request workload, and service class	Maps available deadline slack to a feasible prefill budget	Expands chunks within deadline constraints, with hybrid prioritization and request relegation
SLOPE	Prefill budget under a common request plan	Two-phase service progress, predicted generation demand, and candidate execution time	Normalizes candidate progress and determines relative-cost comparisons	Selects the candidate with the highest two-phase progress per predicted execution time

the larger budget is preferred only when its relative progress gain exceeds its relative execution-time increase (Eq. (13)). The comparison thus evaluates the progress value of additional computation rather than determining the budget solely from available capacity or deadline slack. This criterion also identifies the cost information required for selection: a common positive scaling of all candidate-time estimates within a state leaves the selected budget unchanged, whereas distortion of candidate-relative costs can change the selection. This property motivates within-state relative-cost supervision for SLOPE's execution-latency predictor.

Coverage across systems, policies, and components. The system comparisons are complemented by conventional scheduling policies and component ablations. FCFS, Global-SJF, Global-SRPT, FastServe-SJMLFQ, and Dual-stage EDF cover arrival-order, size- or remaining-work-based, feedback-queue, and deadline-based scheduling. The ablations further examine dynamic budgeting, generation-demand estimation, execution-time normalization, and relative-cost learning. Together, the system comparisons evaluate overall service performance, while the ablations examine the contributions of individual designs.

D SUPPLEMENTARY EXPERIMENTS AND LOAD DEFINITIONS

D.1 TTFT AND E2E COMPONENTS OF THE END-TO-END EXPERIMENTS

Fig. 3 evaluates joint service for the same request. Here we report TTFT and E2E violations using exactly the same models, workloads, methods, loads, and thresholds. TTFT violations indicate late first responses; E2E violations indicate late completion. A request may violate both, so their rates cannot be added to recover the joint violation rate.

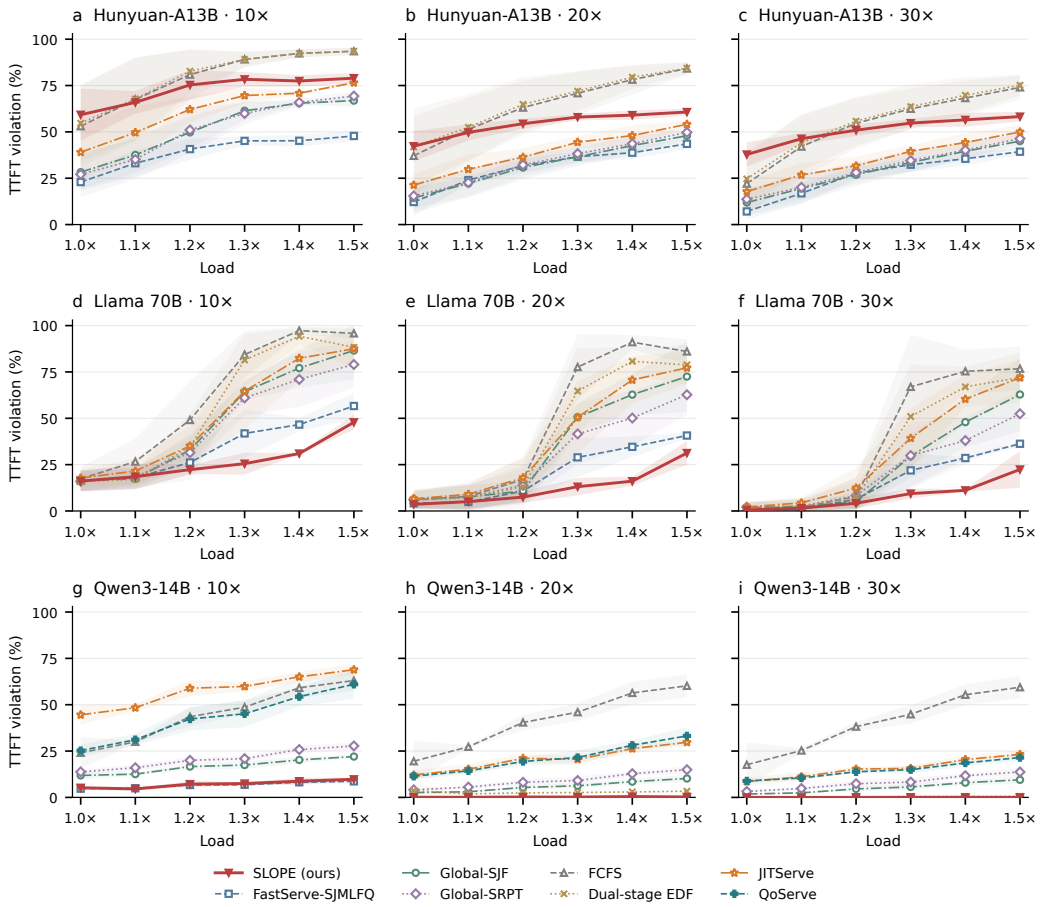


Figure D.1: TTFT violations corresponding to Fig. 3. Rows are Hunyuan-A13B/SCROLLS, Llama 70B/RAGBench, and Qwen3-14B/LMSYS; columns are $10\times$, $20\times$, and $30\times$ SLO thresholds. Displayed loads span $1.0\times$–$1.5\times$ as defined in Appendix D.4; vertical axes are linear percentages. Points and lines show means over three request-stream seeds, and bands show one sample SD, clipped to $[0, 100]\%$.

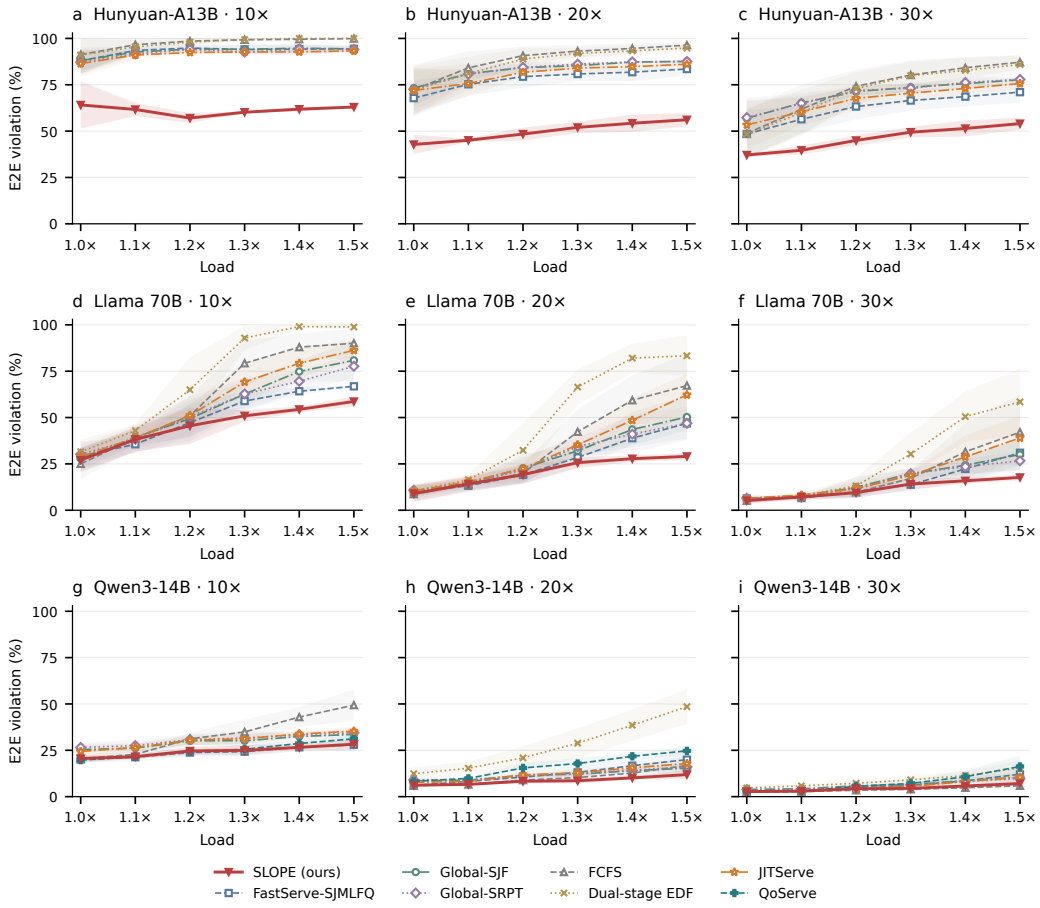


Figure D.2: E2E violations corresponding to Fig. 3. Panel order, runs, load mappings, and statistics match Fig. D.1; the vertical metric is the E2E violation rate. All six displayed loads are retained.

The component results show that joint improvement need not imply leading on each component separately. At 20×, compared with the best evaluated baseline for each component at each load, Hunyuan's mean TTFT violation rate is 23.35 percentage points higher and its E2E violation rate is 28.31 points lower. Llama reduces the two component rates by 7.91 and 5.04 points. Qwen's TTFT rate is approximately 0.07 points higher and its E2E rate is 1.33 points lower. The best baseline is determined independently for each component; these differences are neither added nor substituted for the joint comparison.

D.2 Component Metrics under Bursty Mixed-Tenant Traffic

Fig. D.3 supplements Fig. 4 with TTFT and E2E violations for Llama 70B/RAGBench with TP8. Profiles b0–b4 are defined in Appendix D.3. Requests use their respective tenants' service thresholds, so these results do not split into common 10×/20×/30× panels. Methods affect first response and completion differently across these profiles. The robustness comparison therefore uses joint violations in Fig. 4; component plots explain the allocation consequences. We retain all five profiles for every method.

D.3 Mixed-Tenant Composition and Burst Profiles

Common conditions. The experiment contains interactive, standard, and batch tenants. Their target shares, defined over question groups, are 25%, 25%, and 50%, respectively. They use SLO multipliers of 10×, 20×, and 30×: each multiplier scales both the calibrated TTFT and E2E reference latencies defined in Section 4.1. Thus, each request is evaluated against its own tenant's two

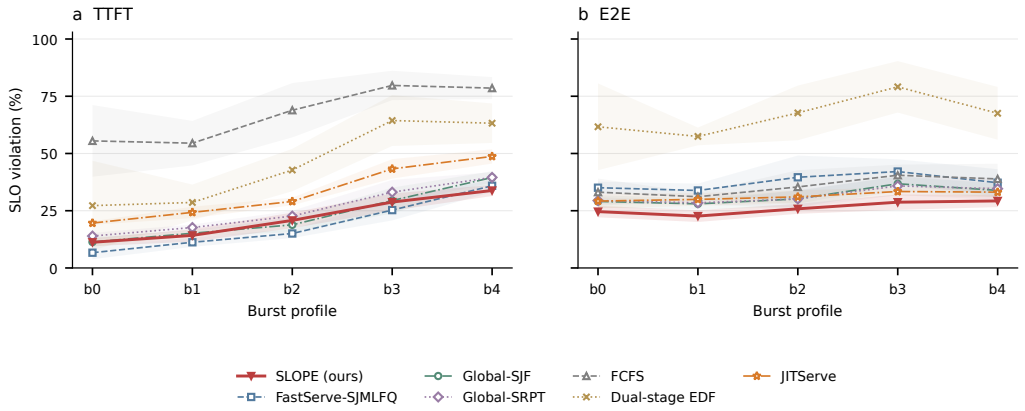


Figure D.3: Component violation rates under bursty multi-tenant arrivals: (a) TTFT and (b) E2E. Categories b0–b4 match Fig. 4; lines track a method across categories. Points are three-seed means, and bands are one-sample SD, clipped to $[0, 100]\%$. Both panels use a common linear vertical scale.

Table D.1: Batch arrival profiles. Lowercase plot labels correspond to the original B0–B4 records. High-period times are relative to each 60-second cycle.

Profile	Pattern	Low	High	High period
b0	Steady arrivals for all tenants	1	1	None
b1	Mild periodic bursts	$2/3$	2	30–45 s
b2	Moderate periodic bursts	$1/3$	3	30–45 s
b3	Strong bursts; batch only in high periods	0	4	30–45 s
b4	Five brief, concentrated pulses	0	12	30–35 s

latency thresholds. Interactive and standard arrivals remain steady throughout the 300-second window; batch is the primary source of bursts. Each seed and profile uses 600 requests at an overall average rate of 2.0 requests/s, the $1.0\times$ reference for this burst protocol. This reference is independent of the end-to-end sweep's load calibration. After 300 seconds, new arrivals stop and accepted requests drain.

The five profiles use the same 600 request IDs within a seed, and methods evaluated with the same seed share a frozen arrival trace. Timestamps are generated offline and not resampled during execution. Profile b0 retains global frozen request order. Profiles b1–b4 retain frozen order within each tenant, with timestamps determining cross-tenant order. Final plots report the mean and sample SD over three request-stream seeds for each method/profile.

Profile definitions. The low- and high-period multipliers in Table D.1 refer to the batch tenant's own mean arrival rate, not the total rate of all tenants. There are five 60-second cycles. The high period begins 30 seconds into each cycle and lasts 15 seconds for b1–b3 or 5 seconds for b4. These multipliers keep the time-weighted mean batch intensity at one: high periods occupy $1/4$ of each cycle for b1–b3 and $1/12$ for b4. Profiles change temporal concentration rather than the total request count. Steady arrivals use sorted uniform timestamps conditional on a fixed count. Periodic profiles generate timestamps under piecewise intensity with fixed tenant counts; b4 distributes requests over five balanced pulses according to the frozen protocol.

D.4 Load Calibration and Statistical Aggregation

Capacity and latency calibration. Capacity is the largest arrival QPS at which steady-state completed requests per second match arriving requests per second. SLO reference latencies are measured by serving 300 requests one at a time on each system. The median TTFT and median E2E are multiplied by the evaluation factor to obtain the respective allowances.

Table D.2: Capacities and actual arrival rates at the six displayed load levels, in plot order.

Model	Capacity (requests/s)	Arrival rates (requests/s)
Hunyuan-A13B	0.70	0.70, 0.77, 0.84, 0.91, 0.98, 1.05
Llama 70B	1.50	1.50, 1.65, 1.80, 1.95, 2.10, 2.25
Qwen3-14B	9.3	9.3, 10.2, 11.2, 12.1, 13.0, 14.0

The plots use six display levels, $1.0\times$ through $1.5\times$. For Hunyuan, Qwen, and Llama, $\ell = \lambda/c$ with capacities $c = 0.70$, $c = 9.3$ and 1.50 requests/s. Table D.2 records the actual rates. Panel labels $10\times/20\times/30\times$ denote SLO thresholds, independently of the horizontal coordinates.

Failed requests count as violations and remain in the submitted-request denominator for joint and component metrics. The runs use seeds 42, 43, and 44. Metrics are computed within each run and then averaged across request-stream seeds, with sample standard deviation. For load-averaged improvements, we first take the lowest baseline mean at each load, compute SLOPE's percentage-point difference or relative reduction, and then average equally over that model's six displayed loads and the baselines evaluated at those loads.

D.5 Supplementary Details for Ablations and Candidate Evaluation

Ablation settings. The ablations in Table 1 use Hunyuan-A13B/SCROLLS and report mean joint violation rates over three request-stream seeds. Fixed-ratio controls set the prefill-budget upper-limit ratio to 0.2, 0.5, or 1.0. Fixed output demand and progress-only selection respectively examine the roles of request-specific demand information and execution-time normalization.

Candidate selection evaluation. Let $C_t(\rho) > 0$ be the measured-cost statistic over a common timing interval and $V_t^C(\rho) = N_t(\rho)/C_t(\rho)$. For selected candidate $\widehat{\rho}_t$, the normalized progress-rate selection loss reported in main-text Table 2 is

$$\mathrm{Loss}_t^{\mathrm{prog}} = 1 - \frac{V_t^C(\widehat{\rho}_t)}{\max_{\rho\in\Omega_t} V_t^C(\rho)}. \tag{D.9}$$

This metric is distinct from terminal joint-attainment regret. The two learned predictors share 5,305 absolute labels, network architecture, initialization, and absolute-supervision settings; full ELP additionally uses 168 relative pairs from 30 training states. The matched control removes both the relative loss in Eq. (C.7) and its low-budget gradient protection (Appendix C.3), evaluating the combined training scheme. All strategies use the same measured interval, from scheduler return to completion of execution-result updates, to compute candidate rates. The relationship between this timing and the analytical scheduling frame is described in Appendices C.3 and B.8.

Scheduling overhead statistics. For the Llama 70B results in Table 4, we first compute the 95th percentile of scheduling-call latency and the ratio of cumulative scheduling time to total run duration within each run, then aggregate the mean and sample standard deviation across three request-stream seeds. Total run duration includes request draining after arrivals stop.

D.6 Robustness to Shaping and Reference-Concurrency Parameters

Under the SLO-normalized interpretation in Eq. (8), the three raw numerator weights reduce, after removing a common scale, to two controller parameters: the dense-prefill shaping strength $\lambda = w_P/w_F$ and the reference decode concurrency $d_0 = (w_D/w_F)(d^E/d^T)$. We therefore interpret the existing weight sweep as a robustness test over these shaping/reference choices rather than as a search for terminal-success probabilities.

Parameter configurations. Let (w_D, w_P, w_F) denote the coefficients in Appendix C.4. Since common positive scaling preserves candidate rankings, we normalize $w_D + w_P + w_F = 1$ and vary the default $(0.50, 0.25, 0.25)$ along two directions. The configurations correspond to Table D.3.

Table D.3: Parameter configurations for the shaping and reference-concurrency sensitivity study.

Configuration	(w_D, w_P, w_F)	$\lambda = w_P/w_F$	$d_0/(d^E/d^T) = w_D/w_F$
Default	$(0.50, 0.25, 0.25)$	1	2
D025	$(0.25, 0.375, 0.375)$	1	$2/3$
D075	$(0.75, 0.125, 0.125)$	1	6
F0125	$(0.50, 0.375, 0.125)$	3	4
F0375	$(0.50, 0.125, 0.375)$	$1/3$	$4/3$

D025/D075 therefore vary the reference-concurrency normalization while keeping the shaping strength fixed at $\lambda = 1$. F0125/F0375 jointly vary the shaping strength and the reference concurrency. The sweep is not intended to identify a theoretically optimal parameter pair.

Experimental setup. We evaluate Hunyuan-A13B and Qwen3-14B at $1.0\times$ and $1.5\times$ loads. Each configuration and load uses three request-stream seeds, giving 60 runs in total. Within each model, load, and seed, all configurations share frozen request sequences and scheduled arrival times. Predictors, candidate construction, and other scheduler settings remain fixed; only the numerator coefficients—equivalently the induced (λ, d_0) pair—change. The default configuration is rerun in the same batch as the reference for comparing mean violation rates.

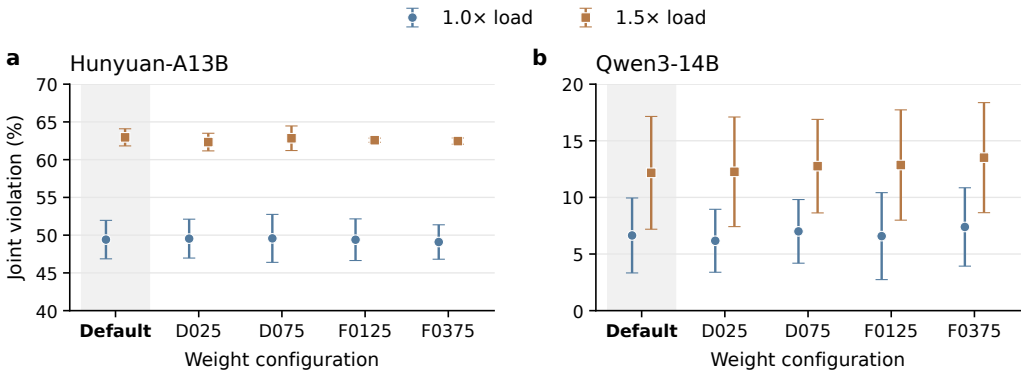


Figure D.4: Robustness to shaping/reference parameterizations at the $20\times$ SLO threshold. Points show mean joint violation rates and error bars show one-sample SD over three request-stream seeds. Panels use separate linear vertical scales; shading marks the default configuration. D025/D075 vary w_D with $w_P = w_F$; F0125/F0375 vary w_F with $w_D = 0.50$. The corresponding (λ, d_0) values are given in the text.

Stable joint-SLO performance across tested parameterizations. Fig. D.4 shows that, across the tested configurations and loads, the largest absolute difference in mean joint violation rate from the default is approximately 0.63 percentage points for Hunyuan and 1.34 percentage points for Qwen. These results support the use of one shared default without load-specific tuning and show that terminal joint-SLO performance is stable over the tested shaping/reference choices. They do not establish that the default (λ, d_0) is optimal or that every parameterization yields an equally tight analytical residual.