CLOUD ON THE CLOCK: ONLINE LLM ROUTING WITH EARLY EDGE TAKEOVER

Anonymous authors
Paper under double-blind review

ABSTRACT

An edge–cloud LLM system routes each query to either a local edge model or a cloud model to balance answer quality against user waiting time. In practice, many routers are trained offline and do not adapt to new service feedback. Some recent routers learn model choices online, but cloud exploration can incur prolonged waiting and highly variable utility feedback, making exploration less efficient. The deadline-only service runs the selected model until delivery or a final deadline, which cannot resolve the difficulty of exploring cloud and learning the quality-delay trade-off under such variability. Therefore, we propose an online edge-cloud routing framework that combines early edge takeover with kernel upper confidence bound (UCB) routing based on semi-adaptive (SA) confidence calibration. During service, the framework interrupts a cloud request that remains unfinished at a threshold and starts an edge generation attempt. After each round, terminal quality–delay feedback updates the selected initial route's weighted kernel utility estimate, while the corresponding prediction error calibrates the SA confidence bonus. Both updates guide UCB routing for the next query. When cloud delays exhibit heavy tails, we analyze the framework using a conditional Pareto model motivated by Pareto execution-time models in cloud straggler analysis. Our analysis establishes conditions for vanishing average regret and for a strictly smaller variance-dependent cumulative regret bound than deadline-only service. Comparing with the strongest baselines on six benchmarks, our method achieves a mean relative utility gain of 27.9% and reduces p95 latency by 36.5%.

1 INTRODUCTION

Response quality and waiting time shape the experience of using an AI assistant (Schlechte et al., 2025; Tan et al., 2026). In an edge–cloud system, the assistant chooses between a compact local LLM and a more capable cloud LLM. A local response may arrive quickly but fall short on a demanding query. Cloud assistance may improve answer quality at the cost of additional inference and transmission delays (Fang et al., 2026). These delays vary with the query, server load, and network conditions, making completion time difficult to predict before routing. The router must select a model before observing the response quality and latency of either candidate. It is therefore critical to study *how a router can learn the quality–delay trade-off from selected feedback*.

Existing routing and cascading methods balance answer quality against inference cost by allocating queries across LLMs (Chen et al., 2024; Dekoninck et al., 2025). HybridLLM and RouteLLM learn routing policies from offline responses or preference data (Ding et al., 2024; Ong et al., 2025). Reinforcement learning can jointly train local reasoning and cloud invocation (Fang et al., 2026), while online kernel routers learn query-dependent choices from selected responses (Hu et al., 2025).

Despite these advances in offline model selection, learning the quality-delay trade-off online from terminal feedback still poses three challenges. First, heavy-tailed cloud delays can impose waiting costs that easily outweigh the cloud model's quality advantage or leave requests unanswered at the deadline (Dean & Barroso, 2013). In LLaMA-7B serving experiments, Sun et al. (2024) frequently reports P95 per-token decode latency 3.8 times the median and a 50-second preemption penalty for a tail request. Second, the router observes terminal feedback only from the selected initial route (Hu et al., 2025), leaving the alternative route's outcome unobserved for the same query. This makes it difficult to determine whether the cloud's expected quality improvement justifies its additional

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.

waiting cost. Obtaining cloud feedback to assess this trade-off can itself incur substantial waiting. Third, the last-minute takeover preserves the chance of a better cloud answer but incurs additional waiting costs, echoing the benefit–cost trade-off in optimal stopping (Lehrer & Wang, 2024). A larger cloud–edge quality gap can justify longer waiting, whereas a fast edge model with adequate quality or a stronger delay penalty can favor earlier takeover.

In this work, we propose an online edge-cloud routing framework that combines early edge takeover with kernel UCB routing based on SA confidence calibration. The framework interrupts unfinished cloud requests at a takeover threshold determined by the cloud-delay scale, the edge completion-time bound, and the final deadline, as detailed in (1), and uses the remaining time for edge execution. Inspired by Jun & Kim (2024), we construct an UCB by adding a SA confidence bonus to weighted kernel utility predictions. After each round, terminal feedback updates the initial route's utility estimate and the SA confidence scale. Our contributions can be summarized as follows:

(1) Joint design of online routing and edge takeover. We construct an online edge–cloud routing framework that combines early edge takeover with kernel UCB routing based on SA confidence calibration. Terminal quality–delay feedback updates the selected initial route's utility estimate and calibrates the SA confidence bonus.

(2) Smaller regret guarantees through early takeover. Under RKHS realizability, the conditional Pareto model, and sufficient timing and variance conditions, we show that early takeover lowers the utility-variance upper bound. Through the SA confidence analysis, this yields a strictly smaller variance-dependent cumulative regret bound than our SA-based UCB router under deadline-only service (Jun & Kim, 2024).

(3) Evaluation across tasks and service preferences. Comparing with the strongest baselines on six benchmarks, our method achieves a mean relative utility gain of 27.9% over the baseline per task and reduces p95 latency by 36.5%. Furthermore, experiments reveal at the same mean cloud delay of 12s, takeover improves utility over the corresponding no-takeover control by 14.3% in the heavier-tailed setting and 5.1% in the less heavy-tailed setting.

2 Related Work

Offline routing and edge–cloud collaboration. HybridLLM and RouteLLM learn from response quality or preference data (Ding et al., 2024; Ong et al., 2025); AutoMix guides escalation through self-verification (Aggarwal et al., 2024), and Fang et al. (2026) jointly train local reasoning and cloud invocation before deployment. These approaches use offline data to train routing predictors, calibrate escalation, or post-train model behavior. Learning the quality–delay trade-off from online service responses requires online updates, which our framework performs after each completed round. **Online model selection.** PAK-UCB learns query-dependent scores from selected-model feedback using kernel UCB (Hu et al., 2025). PILOT combines offline preference initialization with online updates from selected responses (Panda et al., 2025). TRACE-Router learns from terminal accuracy and latency while keeping each agentic task on one model (Raj et al., 2026). These methods learn model choices online but lack early edge takeover, so cloud exploration can still incur prolonged waiting under heavy-tailed delays. Our framework combines online route selection with early edge takeover to limit this waiting. **Noise-adaptive exploration.** Variance-dependent regret bounds are studied in linear (Zhao et al., 2023), contextual (Jia et al., 2024), and kernel bandits (Iwazaki & Takeno, 2025). Thompson sampling adapts to unknown heteroscedastic noise in linear contextual bandits (Xu et al., 2023). These studies analyze learning under reward noise but do not consider how heavy-tailed delays and edge takeover affect service utility and the noise entering the regret bound. In our method, we combine SA exploration with early edge takeover and reduce the variance-dependent regret bound.

3 System Model and Problem Formulation

We consider an AI assistant that serves online queries using a local edge LLM and a cloud LLM, with edge typically providing faster responses but lower answer quality than cloud. Queries are processed sequentially, with each query corresponding to one interaction round.

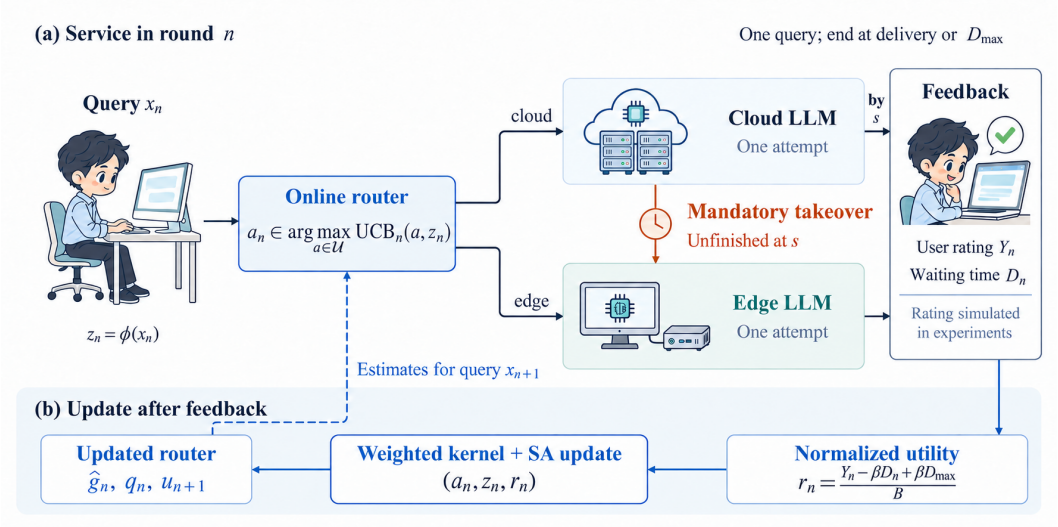


Figure 1: Overview of the online edge-cloud routing service system.

3.1 Online Edge-Cloud Service Process

At round $n = 1, \ldots, N$, an encoder maps query x_n to a representation $z_n = \phi(x_n) \in \mathbb{R}^d$, with $\|z_n\|_2 \leq 1$. The models, decoding settings, and query encoder remain fixed throughout learning, following the common use of selecting pretrained models in online routing works (Hu et al., 2025; Panda et al., 2025). As shown in phase (a) of Figure 1, the router uses the query representation z_n and past terminal feedback to select an initial route $a_n \in \mathcal{U} = \{\mathsf{edge}, \mathsf{cloud}\}$. Then it compares the two routes' predicted utilities, each augmented by an SA confidence bonus, and selects the route with the highest UCB score as detailed in Section 4.3. Each query has a final deadline $D_{\max} > 0$. The selected model makes one generation attempt. A prescribed takeover rule assigns each query z_n a takeover threshold $s(z_n) \in (0, D_{\max})$ before model execution. For the theoretical guarantees in Section 5, the rule satisfies, for every query,

$$4 \max\{x_+, d_e\} \leq s(z_n) \leq D_{\max}/2, \tag{1}$$

where x_+ and d_e are the cloud-scale and edge-time upper bounds specified in Assumption 1. In particular, our analysis and experiments focus on the constant-threshold specialization $s(z_n) \equiv s$. If $a_n = \mathsf{edge}$, the edge model makes one generation attempt until delivery or $D_{\max}$. If $a_n = \mathsf{cloud}$, the cloud model starts generation and returns its answer if completed by s. Otherwise, it is interrupted at s, and a new edge generation attempt starts with the remaining time $D_{\max} - s$. Service then ends at delivery or $D_{\max}$, as detailed in Section 4.1. Let D_{c} denote the potential cloud completion time from query arrival without interruption, and let $\mathcal{G}_n$ contain past observations, the current query, and selected actions before feedback arrives. Inspired by Pareto execution-time models in cloud straggler analysis (Wang et al., 2019), we model cloud latency using a conditional Pareto distribution. The following assumption also imposes a uniform bound on edge service time.

Assumption 1 (Heavy-tailed cloud latency and bounded edge service time)**.** *(1) For any time $t \geq 0$ measured from query arrival, the potential cloud completion time D_{c} has the conditional survival function*

$$\Pr(D_{\mathrm{c}} > t \mid \mathcal{G}_n) = \begin{cases} 1, & 0 \leq t < x(z_n), \\ \left(\dfrac{x(z_n)}{t}\right)^{\alpha(z_n)}, & t \geq x(z_n), \end{cases} \tag{2}$$

where $x(z_n)$ and $\alpha(z_n)$ are the scale and tail-shape parameters, respectively. The constants x_-, x_+ and α_-, α_+ are their respective lower and upper bounds, where $0 < x_- \leq x(z_n) \leq x_+, 1 < \alpha_- \leq \alpha(z_n) \leq \alpha_+ < 2$. (2) Direct-edge completion time from query arrival and additional takeover time from s are uniformly bounded by the constant $d_e > 0$.

Pareto-like tails have been reported in Facebook and Bing task durations (Ananthanarayanan et al., 2014) and HTTP and e-commerce response times (Scharf, 2005; Vallamsetty et al., 2003). Service

measurements on the real datasets MMLU, TriviaQA, and SQuAD in Section 6 also show pronounced cloud-latency tails and edge completion times below $s/4$ at the chosen thresholds. For theoretical analysis, Assumption 1 uses an exact conditional Pareto model with uniformly bounded parameters and tail exponents in $(1, 2)$. This specifies a class of heavy-tailed cloud delays with finite mean and infinite variance before deadline truncation. Together with the timing condition (1), this assumption enables the delay-variance reduction analysis in Section 5.

Each interaction ends when an answer is delivered or the deadline is reached. Let $Y_n \in [0, 1]$ denote the user's rating of the delivered answer, with 0 meaning unsatisfactory and 1 satisfactory, and let $D_n \in [0, D_{\max}]$ denote the total waiting time from query arrival. For either model, the user rating depends on the query and may be zero even for a poor answer delivered by $D_{\max}$. Under the general service rule, if no answer is delivered by $D_{\max}$, set $Y_n = 0$. As shown in phase (b) of Figure 1, terminal feedback (Y_n, D_n) updates the weighted kernel utility estimate for the selected initial route and calibrates the SA confidence scale for subsequent queries. For a cloud-initiated request, this feedback updates the cloud route's estimate even when takeover delivers an edge answer.

3.2 Utility and Kernel Representation

Using the terminal feedback (Y_n, D_n) defined in Section 3.1, we define the observed utility of round n for a latency penalty $\beta > 0$ as

$$U_n = Y_n - \beta D_n, \tag{3}$$

where $U_n \in [-\beta D_{\max}, 1]$ and β represents a rating–delay preference coefficient. Let $f_a(z)$ denote the unknown expected utility of the complete service initiated by action a for query representation z. We use a stationary conditional-mean model in which expected utility depends only on the initial action and query representation: $\mathbb{E}[U_n \mid \mathcal{G}_n] = f_{a_n}(z_n)$, where history $\mathcal{G}_n$ is defined in Section 3.1. For facilitating the SA confidence construction in Section 4.2 later, we normalize the observed utility and its mean to the range $[0, 1]$: $r_n = (U_n + \beta D_{\max})/B$ and $g(a, z) = (f_a(z) + \beta D_{\max})/B$, where $B = 1 + \beta D_{\max}$ is the utility interval width. Thus $g(a_n, z_n) = \mathbb{E}[r_n \mid \mathcal{G}_n]$ is the normalized conditional mean to be learned. To estimate g from the observed feedback, we define a kernel κ on action–query pairs:

$$\kappa\big((a, z), (a', z')\big) = C^2 \mathbf{1}\{a = a'\} k(z, z'), \tag{4}$$

where $k(z, z') = [1 + \exp(-\|z - z'\|_2^2/(2h^2))]/2$ measures query similarity, $h > 0$ is its bandwidth, and $C \geq 1$ is the kernel scale. Let $\mathcal{H}$ be the reproducing kernel Hilbert space (RKHS) associated with κ. For any action-query pair $\xi = (a, z)$ with $a \in \mathcal{U}$ and $\|z\|_2 \leq 1$, define its feature representation as $\Phi(\xi) = \kappa(\xi, \cdot) \in \mathcal{H}$ to estimate g and quantify prediction uncertainty in Section 4.2 later. We make the following RKHS assumption on the normalized mean utility g (Chowdhury & Gopalan, 2017).

Assumption 2 (RKHS realizability)**.** *For the stationary conditional-mean model $\mathbb{E}[U_n \mid \mathcal{G}_n] = f_{a_n}(z_n)$, the normalized mean utility satisfies $g \in \mathcal{H}$ and $\|g\|_{\mathcal{H}} \leq 1$.*

This assumption formalizes the premise that similar query representations have similar expected utility for the same route, following bounded-RKHS models in LLM selection (Hu et al., 2025). It provides a sufficient regularity condition for our confidence and regret guarantees in Section 5.

3.3 Learning Objective

The router aims to maximize the expected average utility over successive rounds. An online routing algorithm $\mathcal{A} = (\mathcal{A}_n)_{n \geq 1}$ consists of decision rules that map the current query representation and past observations to an initial action $a_n = \mathcal{A}_n\big(z_n, (z_i, a_i, Y_i, D_i)_{i<n}\big) \in \mathcal{U}$, where $\mathcal{U}$ is the action set defined in Section 3.1. The learning objective is

$$\underset{\mathcal{A}}{\text{maximize}} \quad J_N(\mathcal{A}), \qquad J_N(\mathcal{A}) = \frac{1}{N}\mathbb{E}_{\mathcal{A}}\left[\sum_{n=1}^{N} U_n\right]. \tag{5}$$

The expectation $\mathbb{E}_{\mathcal{A}}$ implicitly incorporates the cloud-delay model in (2) through the distribution of the terminal feedback (Y_n, D_n) under the service rule in Section 3.1. We evaluate the router against an oracle that knows the mean utility functions. For each query z_n, the oracle selects $a_n^* \in \arg\max_{a \in \mathcal{U}} f_a(z_n)$. The cumulative regret of algorithm $\mathcal{A}$ is

$$R_N(\mathcal{A}) = \sum_{n=1}^{N} \left[f_{a_n^*}(z_n) - f_{a_n}(z_n)\right], \tag{6}$$

where $a_n \in \mathcal{U} = \{\mathsf{edge}, \mathsf{cloud}\}$ is the initial route selected by $\mathcal{A}$. The average regret $R_N(\mathcal{A})/N$ gives the corresponding loss per round. To approach this oracle, we construct an online routing framework that integrates early edge takeover with kernel UCB routing based on SA confidence calibration as detailed in Section 4. Terminal feedback, including takeover outcomes, updates the selected initial route's utility estimate and the SA confidence scale, which are then used to compute UCB scores for the next query.

4 Online Routing with Edge Takeover

We present our design of early edge takeover, weighted kernel estimation, and the SA confidence bound, followed by route selection and updates from terminal service feedback.

4.1 Mandatory Early Edge Takeover

Encoding and initial route selection in round n must finish before s. The selected route then follows the service rule below. For query representation z_n, let $(y_{\mathsf{edge}}(z_n), d_{\mathsf{edge}}(z_n))$ and $(y_{\mathsf{cloud}}(z_n), d_{\mathsf{cloud}}(z_n))$ denote the potential answer quality and time to delivery of the edge and cloud models, respectively, if allowed to finish. Both times are measured from query arrival and may exceed $D_{\max}$; $d_{\mathsf{cloud}}(z_n)$ may have a heavy-tailed distribution. For edge takeover at s, let $(y_{\mathrm{to}}(z_n), d_{\mathrm{to}}(z_n))$ denote its potential answer quality and additional time to delivery, measured from s and including takeover. The delivered quality and actual waiting time are

$$Y_n = \begin{cases} y_{\mathsf{edge}}(z_n)\mathbf{1}\{d_{\mathsf{edge}}(z_n) \le D_{\max}\}, & a_n = \mathsf{edge}, \\ y_{\mathsf{cloud}}(z_n), & a_n = \mathsf{cloud},\ d_{\mathsf{cloud}}(z_n) \le s, \\ y_{\mathrm{to}}(z_n)\mathbf{1}\{d_{\mathrm{to}}(z_n) \le D_{\max} - s\}, & a_n = \mathsf{cloud},\ d_{\mathsf{cloud}}(z_n) > s, \end{cases}$$

$$D_n = \begin{cases} \min\{d_{\mathsf{edge}}(z_n), D_{\max}\}, & a_n = \mathsf{edge}, \\ d_{\mathsf{cloud}}(z_n), & a_n = \mathsf{cloud},\ d_{\mathsf{cloud}}(z_n) \le s, \\ s + \min\{d_{\mathrm{to}}(z_n), D_{\max} - s\}, & a_n = \mathsf{cloud},\ d_{\mathsf{cloud}}(z_n) > s. \end{cases} \tag{7}$$

When a takeover occurs, the utility in (3) updates the initial cloud route's estimate.

4.2 Service Utility Estimation and Confidence Calibration

After round n, we use the selected pair $\xi_n = (a_n, z_n)$ and normalized feedback r_n from Section 3.2 to update the predictor and confidence scale for route selection in round $n+1$.

Utility estimation. Let $\widehat{g}_n$ denote the weighted kernel ridge estimate of the normalized mean utility g based on $\{(\xi_i, r_i)\}_{i=1}^n$, and let V_n denote the operator summarizing their weighted feature information. Initialize $V_0 = \lambda I$ and $\widehat{g}_0 = 0$, where $\lambda > 0$ is the regularization parameter and I is the identity operator on $\mathcal{H}$. For this update, we use the kernel uncertainty $u_n(\xi) = \|\Phi(\xi)\|_{V_{n-1}^{-1}}$ in estimating $g(\xi)$, computed from the first $n-1$ rounds. The observation weight $w_n = \min\{1, 1/u_n(\xi_n)\}$ is fixed after selecting a_n and before observing r_n, where $w_n = 1$ when $u_n(\xi_n) = 0$. With the observation (ξ_n, r_n), the updated estimate is

$$\widehat{g}_n = V_n^{-1} \sum_{i=1}^{n} w_i^2 \Phi(\xi_i) r_i, \tag{8}$$

where $V_n = V_0 + \sum_{i=1}^n w_i^2 \Phi(\xi_i) \otimes \Phi(\xi_i)$. Here, $\otimes$ denotes the outer product on $\mathcal{H}$, defined by $(u \otimes v)f = u\langle v, f\rangle_{\mathcal{H}}$ for $u, v, f \in \mathcal{H}$.

Confidence calibration. The same feedback also calibrates a shared confidence scale for subsequent exploration (Jun & Kim, 2024). Using the prediction recorded before round n's feedback, we update the cumulative statistic of weighted prediction errors:

$$P_n = \sum_{i=1}^{n} \frac{h_i^2 e_i^2}{1 + h_i^2}, \tag{9}$$

where $e_n = w_n[r_n - \widehat{g}_{n-1}(\xi_n)]$ is the weighted prediction error and $h_n = w_n u_n(\xi_n) = \min\{u_n(\xi_n), 1\}$ is the weighted kernel uncertainty fixed before feedback. The confidence scale

is determined by

$$q_n = \lambda + P_n + H/2, \qquad n \geq 0, \tag{10}$$

where $H = \log(6/\delta)$ for $\delta \in (0, 1)$ and $P_0 = 0$. The updated scale $\sqrt{q_n}$ is shared across both routes and multiplies u_{n+1} in the next round's exploration bonus.

4.3 Online Routing and Update

At the beginning of round $n+1$, the router receives z_{n+1} and uses the current state $\widehat{g}_n, V_n, q_n$ defined in Section 4.2, initialized at $n = 0$, to select

$$a_{n+1} \in \arg\max_{a \in \mathcal{U}} \mathrm{UCB}_{n+1}(a, z_{n+1}), \tag{11}$$

where $\mathrm{UCB}_{n+1}(a, z_{n+1}) = \widehat{g}_n(a, z_{n+1}) + \sqrt{q_n}\, u_{n+1}(a, z_{n+1})$. Since $f_a(z) = Bg(a, z) - \beta D_{\max}$ with $B > 0$, maximizing g also maximizes expected utility, linking this score to the objective in (5). We regulate that exact ties select the edge route. After selecting a_{n+1}, we set $\xi_{n+1} = (a_{n+1}, z_{n+1})$, record $\widehat{g}_n(\xi_{n+1})$, and determine w_{n+1}, h_{n+1} before executing the service process in (7). When this service ends, we compute r_{n+1} and its prediction error using the recorded prediction, then update $\widehat{g}_{n+1}, V_{n+1}, q_{n+1}$ for the following query. Algorithm 1 summarizes the whole service process.

Algorithm 1 Online LLM routing with early edge takeover

Require: Edge/cloud models, encoder ϕ, and kernel κ; $0 < s < D_{\max}$, $\beta > 0$, $\lambda > 0$, and $\delta \in (0, 1)$

Set $B = 1 + \beta D_{\max}$ and $H = \log(6/\delta)$
Initialize $V_0 = \lambda I$, $\widehat{g}_0 = 0$, $P_0 = 0$, and $q_0 = \lambda + H/2$
for $n = 0, 1, 2, \ldots$ **do**
  Receive x_{n+1}, start the service clock, and compute $z_{n+1} = \phi(x_{n+1})$
  Compute $u_{n+1}(a, z_{n+1})$ from V_n for both routes ▷ Uncertainty from past queries
  Select a_{n+1} by (11), breaking ties toward edge ▷ Prediction + exploration bonus
  Set $\xi_{n+1} = (a_{n+1}, z_{n+1})$ and record $\widehat{g}_n(\xi_{n+1})$
  $w_{n+1} \leftarrow \min\{1, 1/u_{n+1}(\xi_{n+1})\}$; $h_{n+1} \leftarrow w_{n+1} u_{n+1}(\xi_{n+1})$
  if $a_{n+1} =$ edge **then** run edge until delivery or $D_{\max}$
  else run cloud until delivery or s
    if cloud is unfinished at s **then** interrupt cloud; run a new edge attempt until delivery or $D_{\max}$
    end if
  end if
  Observe terminal feedback (Y_{n+1}, D_{n+1}) according to (7)
  $U_{n+1} \leftarrow Y_{n+1} - \beta D_{n+1}$; $r_{n+1} \leftarrow (U_{n+1} + \beta D_{\max})/B$ ▷ Normalize utility to $[0, 1]$
  $e_{n+1} \leftarrow w_{n+1}[r_{n+1} - \widehat{g}_n(\xi_{n+1})]$ ▷ Error against recorded prediction
  $P_{n+1} \leftarrow P_n + h_{n+1}^2 e_{n+1}^2/(1 + h_{n+1}^2)$ ▷ Accumulate weighted squared errors
  Update $V_{n+1}, \widehat{g}_{n+1}$ using (8); set $q_{n+1} \leftarrow \lambda + P_{n+1} + H/2$ ▷ Update estimates and confidence
end for

5 Theoretical Guarantees

For simpler constant-threshold specialization s as detailed in (1), we derive delay and utility variance bounds under takeover, then establish regret guarantees using kernel information growth and the SA framework. We compare the resulting bounds for the same router under takeover and deadline-only service Jun & Kim (2024). Proofs appear in Appendices A–C. For a cloud-initiated request, $D_0 = \min\{D_c, D_{\max}\}$ is the waiting time under deadline-only service, while D_s is the waiting time under takeover at s, following (7). Both services use the same final deadline $D_{\max}$. The quantity v_0 is the supremum of $\mathrm{Var}(D_0 \mid \mathcal{G}_n)$ over all histories and queries.

Proposition 1 (Variance reduction through takeover)**.** *Under Assumption 1 and the timing condition* (1)*, early takeover strictly reduces the conditional waiting-time variance of cloud-initiated requests:*

$$\mathrm{Var}(D_0 \mid \mathcal{G}_n) - \mathrm{Var}(D_s \mid \mathcal{G}_n) \geq \frac{s^2}{16} \left(\frac{x_-}{s}\right)^{\alpha_+} > 0. \tag{12}$$

Proposition 1 shows that increasing s within (1) strengthens the variance-reduction guarantee and reduces the regret bound while the cloud-route variance term dominates, under the same learning assumptions and parameters. The choice of s balances potential cloud-quality gains against additional

waiting costs. We next bound kernel information to control accumulated prediction uncertainty in the regret analysis.

Lemma 1 (Information growth of the routing kernel)**.** *For the kernel in* (4) *on the action–query domain of Section 3.1, the worst-case kernel information satisfies*

$$\Gamma_N = O\big((\log(N+1))^{d+1}\big) = o(\sqrt{N}), \tag{13}$$

where $\Gamma_N = \max\{1, \sup_{\xi_{1:N}} \frac{1}{2}\log\det(I_N + K_N/\lambda)\}$*,* $K_N = [\kappa(\xi_i, \xi_j)]_{i,j=1}^N$*,* I_N *is the* $N \times N$ *identity, and the supremum is over all action–query sequences.*

The constant components and route blocks preserve the squared-exponential information rate (Vakili et al., 2021). We combine this information bound with the service-variance analysis and SA confidence calibration to obtain the following regret guarantees.

Theorem 1 (Cumulative and average regret)**.** *Consider the router* $\mathcal{A}$ *in Algorithm 1 with the kernel in Lemma 1, the constant threshold* $s(z_n) \equiv s$*, and confidence parameter* $\delta \in (0,1)$*. Under Assumptions 1 and 2 and the timing condition* (1)*, with probability at least* $1-\delta$*, its cumulative regret satisfies* $R_N(\mathcal{A}) \le \mathcal{R}_N(\bar{v}_s)$ *simultaneously for all* $N \ge 1$*. Consequently, for any fixed finite representation dimension* d*, with the remaining service, kernel, and confidence parameters independent of* N*, the average regret satisfies, as* $N \to \infty$*,*

$$\frac{R_N(\mathcal{A})}{N} = O\left(\frac{(\log(N+1))^{d+1}}{\sqrt{N}}\right) \longrightarrow 0, \tag{14}$$

where $v_s = \max\{\frac{d_e^2}{4}, v_0 - \frac{s^2}{16}(x_-/s)^{\alpha_+}\}$ *and* $\bar{v}_s = (\frac{1}{2} + \beta\sqrt{v_s})^2$ *uniformly bound* $\mathrm{Var}(D_n \mid \mathcal{G}_n)$ *and* $\mathrm{Var}(U_n \mid \mathcal{G}_n)$*, respectively;* $B = 1 + \beta D_{\max}$*,* $H = \log(6/\delta)$*, and* $c_v = 4(e-1)$*; and* $\mathcal{R}_N(v) = 2\sqrt{\frac{2N\Gamma_N}{\log 2}[c_v v \Gamma_N + B^2(2\lambda + 2H)]}$ *is the cumulative regret upper-bound function for a deterministic uniform utility-variance bound* $v \ge 0$*.*

We next compare the same router under takeover and deadline-only service, using the same models, query domain, kernel, and parameters $\lambda, \delta, \beta, D_{\max}$. Both normalized service mean utilities satisfy Assumption 2, and each regret is measured against its own service oracle.

Corollary 1 (Smaller regret bound through takeover)**.** *Under the conditions of Theorem 1 and the comparison setting above, suppose* $v_0 > d_e^2/4$*. Then* $v_s < v_0$*, and takeover yields a strictly smaller variance-dependent regret bound for every* $N \ge 1$*:*

$$\frac{\mathcal{R}_N(\bar{v}_0)}{\mathcal{R}_N(\bar{v}_s)} = \sqrt{1 + \frac{c_v\Gamma_N(\bar{v}_0 - \bar{v}_s)}{c_v\bar{v}_s\Gamma_N + B^2(2\lambda + 2H)}} > 1, \tag{15}$$

where $\bar{v}_0 = (1/2 + \beta\sqrt{v_0})^2$ *is the utility-variance upper bound for deadline-only service, and* $\bar{v}_s = (1/2 + \beta\sqrt{v_s})^2$ *is that for takeover, with* $v_0 - v_s = \min\{v_0 - d_e^2/4,\ (s^2/16)(x_-/s)^{\alpha_+}\} > 0$ *and* $\bar{v}_0 - \bar{v}_s = \beta(\sqrt{v_0} - \sqrt{v_s}) + \beta^2(v_0 - v_s) > 0$*.*

Corollary 1 establishes a strict regret-bound improvement already for the constant-threshold specialization $s(z_n) \equiv s$, under Assumption 1, Assumption 2 for both service means, the timing condition (1), and $v_0 > d_e^2/4$. Query-dependent thresholds $s(z_n)$ in (1) form a more flexible class containing this specialization and allow waiting budgets to vary across queries.

6 Experiments

6.1 Experimental Settings

Datasets. We cover common task families studied in edge–cloud collaboration and LLM routing (Fang et al., 2026; Ong et al., 2025; Hu et al., 2025): mathematical reasoning on MATH-Lighteval (Hendrycks et al., 2021b) and MATH-500 (Lightman et al., 2024), multiple-choice question answering on MMLU (Hendrycks et al., 2021a), closed-book question answering on TriviaQA (Joshi et al., 2017), passage-based question answering on SQuAD 1.1 (Rajpurkar et al., 2016), and Python code generation on MBPP (Austin et al., 2021). **Models.** Following the edge-model choice in GAPG (Fang et al., 2026), we use Qwen2.5-1.5B-Instruct (Qwen Team, 2024) with BF16 greedy

decoding. To evaluate routing with a newer generation of cloud reasoning models, we adopt DeepSeek-V4-Pro (DeepSeek-AI et al., 2026) as the cloud model, with thinking enabled. **Simulated feedback for metrics.** We use reference answers or test cases to evaluate the delivered outcomes and provide quality scores in $[0, 1]$. Mathematics uses final-answer verification, MMLU uses option correctness, and MBPP uses execution-test pass/fail. TriviaQA and SQuAD use the maximum normalized token F1 over reference answers. **Baselines.** We evaluate deadline-adapted and last-minute versions of Always Edge, Always Cloud, RouteLLM (Ong et al., 2025), a router-only GAPG adaptation (Fang et al., 2026), and PAK-UCB (Hu et al., 2025).

6.2 Overall Performance

Table 1 shows that our method achieves the highest final mean utility on all six tasks, including last-minute takeover baselines. The mean per-task relative gain over the strongest baseline, normalized by its absolute utility, is 27.9%; macro-averaged utility rises from 0.2380 to 0.2776. Relative to the oracle's macro-average of 0.3474, these values correspond to 68.51% and 79.90%. Figure 2 shows a positive mean utility gap over the strongest baseline throughout the second half of each task's request stream, despite early fluctuations. Negative utility reflects waiting costs exceeding ratings.

Table 1: Final cumulative average utility (β: quality–delay preference). Baselines report deadline-only utility; parentheses show last-minute utility ($s = D_{\max} - 1$), omitted when identical.

Method	MATH-L $\beta = 0.08$	MATH-500 $\beta = 0.08$	MMLU $\beta = 0.04$	TriviaQA $\beta = 0.17$	SQuAD 1.1 $\beta = 0.12$	MBPP $\beta = 0.01$
Always Edge	−0.1340	−0.1429	0.5304	0.1812	0.5257	0.3347
Always Cloud	−0.1220	−0.2099 (−0.2141)	0.4325 (0.4569)	0.1333 (0.1367)	0.5394 (0.5404)	0.2794 (0.2876)
RouteLLM (Ong et al., 2025)	−0.0927	−0.1158 (−0.1162)	0.5120 (0.5167)	0.1820	0.5257	0.3347
GAPG (Fang et al., 2026)	−0.1289	−0.1803 (−0.1821)	0.5013 (0.5035)	0.1463 (0.1466)	0.5348 (0.5353)	0.3178 (0.3200)
PAK-UCB (Hu et al., 2025)	−0.0603	−0.1081 (−0.1126)	0.5307 (0.5378)	0.1550 (0.1540)	0.5419 (0.5408)	0.3140 (0.3184)
Ours	**−0.0098**	**−0.0922**	**0.5601**	**0.2799**	**0.5799**	**0.3474**
Oracle	0.0740	0.0151	0.5963	0.3568	0.6352	0.4069

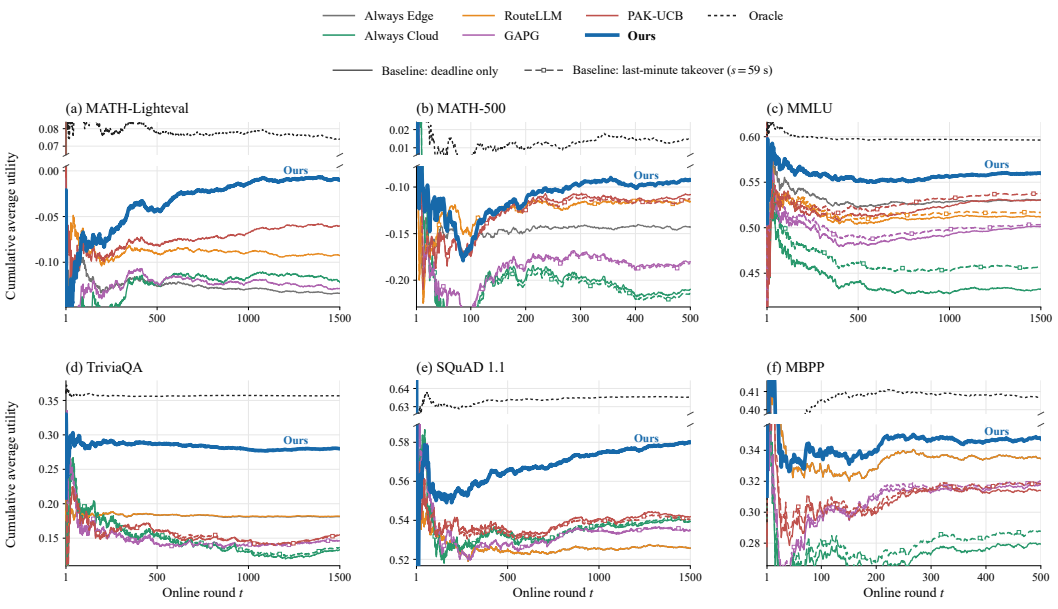


Figure 2: Cumulative average utility over the full request stream.

6.3 Takeover under Heavy-Tailed Latency

Does takeover improve the delivered service? Figure 3(a) shows that takeover reduces p95 latency by 36.5% on average across tasks and eliminates all 366 observed deadline failures. Especially on SQuAD, the maximum recorded edge completion time is 3.7462 s. Covering these observations

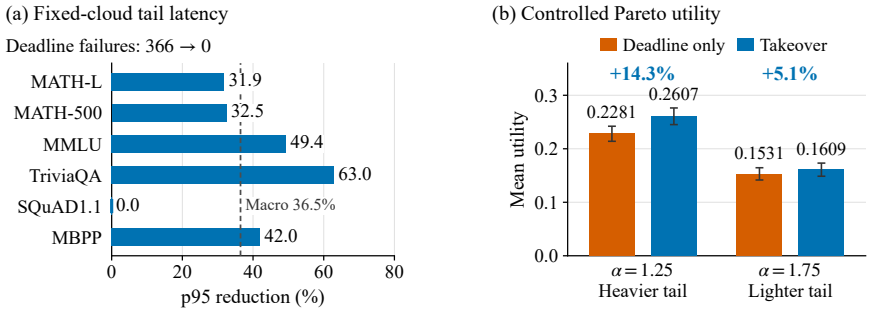


Figure 3: Takeover effects. (a) Per-task p95 latency reduction; the dashed line marks the macro-average. (b) Controlled Pareto utility at $N = 1{,}500$.

Table 2: Ablations of SA calibration and takeover: final cumulative average utility in 10^{-3} units.

Configuration	MATH-L	MATH-500	MMLU	TriviaQA	SQuAD 1.1	MBPP
Standard radius without takeover	-78.70	-144.89	510.68	154.31	543.41	313.87
SA without takeover	-38.30	-129.84	529.43	159.35	557.97	319.22
Standard radius + takeover	-52.25	-125.75	545.90	268.64	559.17	341.54
SA + takeover (Ours)	**-9.76**	**-92.21**	**560.10**	**279.89**	**579.91**	**347.44**

under (1) requires $s \gtrsim 15$ s, which exceeds the recorded cloud p95 of 6.78 s. Thus, p95 remains unchanged under these timing constraints. On MATH-Lighteval and MATH-500, takeover p95 exceeds $5s/4 = 37.5$ s, the limit implied by Assumption 1 together with (1), while mean delay and utility still improve. These results suggest that the sufficient fast-edge timing requirement can be relaxed in practical deployments, with empirical benefits extending beyond the theoretically guaranteed regime. **How does tail heaviness affect takeover gains?** To examine tail heaviness, we use a controlled Pareto model from cloud straggler analysis (Wang et al., 2019) satisfying Assumptions 1 and 2. The tail exponent α controls tail heaviness: smaller values give heavier tails. At the same mean cloud delay of 12 s, takeover improves utility by 14.3% for $\alpha = 1.25$ and 5.1% for $\alpha = 1.75$ after 1,500 rounds (Figure 3(b)), which means our method achieves a larger utility gain in the heavier-tail setting.

6.4 Ablation Study

(1) Mandatory takeover. Table 2 shows that takeover improves SA's final mean utility on all six tasks by 3.9–75.6% (absolute gains of 0.0219–0.1205), while reducing mean waiting. On MATH-500, utility increases from -0.1298 to -0.0922 (+29.0%): waiting-cost savings of 0.0436 outweigh a quality loss of 0.0060. **(2) SA confidence calibration.** To assess SA calibration, we replace its radius with a standard sub-Gaussian radius (Chowdhury & Gopalan, 2017) based on a prescribed noise bound. Table 2 shows that SA with takeover achieves the highest final mean utility among the four configurations on all six tasks. Especially on MATH-Lighteval and MATH-500, it improves utility over standard calibration by 81.3% (+0.0425) and 26.7% (+0.0335), respectively.

7 Conclusion

We propose an online edge-cloud routing framework that learns the quality-delay trade-off from terminal service feedback. Weighted kernel prediction and SA exploration guide initial route selection, while early edge takeover limits heavy-tailed cloud waiting before the final deadline. Motivated by a conditional Pareto model of heavy-tailed cloud latency, we also establish vanishing average regret and show that takeover reduces the utility-variance bound, leading to a strictly smaller variance-dependent cumulative regret bound than deadline-only service. Comparing with the strongest baselines on six benchmarks, our method achieves a mean relative utility gain of 27.9% and reduces p95 latency by 36.5%. Service comparisons and ablations show how takeover improves utility by reducing waiting costs under heavy-tailed cloud delays.

AI USE STATEMENT

In this work, the authors developed the research ideas and experimental design and independently derived and checked the regret bounds. Generative AI tools assisted with the verification of these derivations, literature searches, code debugging, figure preparation, and manuscript editing. The authors reviewed the AI-assisted work, checked the cited sources and determined the final interpretations and conclusions. The authors take responsibility for all claims, code and conclusions.

ETHICS STATEMENT

We evaluate edge–cloud routing through benchmark quality scores and recorded service feedback. The latency weight encodes a prescribed preference; its interpretation as human satisfaction requires user evaluation. Deployment should protect query confidentiality during cloud offloading and assess whether routing and deadline policies produce uneven service quality across user groups. Content safety and application-specific risks require evaluation alongside the quality–delay utility.

REPRODUCIBILITY STATEMENT

The routing procedure and online updates are specified in Section 4 and Algorithm 1, with kernel-matrix computations in Appendix A.3. The modeling assumptions are stated in Section 3, and complete proofs of the theoretical guarantees appear in Appendices A–C. For experimental reproduction, Section 6.1 and Appendix D document the model and decoding settings, feedback scoring, query-feature construction, prompt templates, baseline configurations, request-order seeds, and aggregation procedures. Appendix F specifies how recorded service times are replayed and how the controlled Pareto experiments are constructed. The accompanying supplement material provides exact parameter records, cached service-replay inputs, verification scripts, and numerical data for the controlled learning experiments.

REFERENCES

Yasin Abbasi-Yadkori, Dávid Pál, and Csaba Szepesvári. Improved algorithms for linear stochastic bandits. In *Advances in Neural Information Processing Systems*, volume 24, 2011. URL https://proceedings.neurips.cc/paper/2011/hash/e1d5be1c7f2f456670de3d53c7b54f4a-Abstract.html.

Pranjal Aggarwal, Aman Madaan, Ankit Anand, Srividya Pranavi Potharaju, Swaroop Mishra, Pei Zhou, Aditya Gupta, Dheeraj Rajagopal, Karthik Kappaganthu, Yiming Yang, Shyam Upadhyay, Manaal Faruqui, and Mausam. AutoMix: Automatically mixing language models. In *Advances in Neural Information Processing Systems*, volume 37, 2024. doi: 10.52202/079017-4164. URL https://proceedings.neurips.cc/paper_files/paper/2024/hash/ecda225cb187b40ea8edc1f46b03ffda-Abstract-Conference.html.

Ganesh Ananthanarayanan, Michael Chien-Chun Hung, Xiaoqi Ren, Ion Stoica, Adam Wierman, and Minlan Yu. GRASS: Trimming stragglers in approximation analytics. In *11th USENIX Symposium on Networked Systems Design and Implementation (NSDI 14)*, pp. 289–302. USENIX Association, 2014. URL https://www.usenix.org/conference/nsdi14/technical-sessions/presentation/ananthanarayanan.

Jacob Austin, Augustus Odena, Maxwell Nye, Maarten Bosma, Henryk Michalewski, David Dohan, Ellen Jiang, Carrie Cai, Michael Terry, Quoc Le, and Charles Sutton. Program synthesis with large language models. *arXiv preprint arXiv:2108.07732*, 2021. URL https://arxiv.org/abs/2108.07732.

Lingjiao Chen, Matei Zaharia, and James Zou. FrugalGPT: How to use large language models while reducing cost and improving performance. *Transactions on Machine Learning Research*, 2024. ISSN 2835-8856. URL https://openreview.net/forum?id=cSimKw5p6R.

Sayak Ray Chowdhury and Aditya Gopalan. On kernelized multi-armed bandits. In *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pp. 844–853. PMLR, 2017. URL https://proceedings.mlr.press/v70/chowdhury17a.html.

Jeffrey Dean and Luiz André Barroso. The tail at scale. *Communications of the ACM*, 56(2):74–80, 2013. doi: 10.1145/2408776.2408794.

DeepSeek-AI et al. DeepSeek-V4: Towards highly efficient million-token context intelligence. *arXiv preprint arXiv:2606.19348*, 2026. URL https://arxiv.org/abs/2606.19348.

Jasper Dekoninck, Maximilian Baader, and Martin Vechev. A unified approach to routing and cascading for LLMs. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 12987–13010. PMLR, 2025. URL https://proceedings.mlr.press/v267/dekoninck25a.html.

Dujian Ding, Ankur Mallick, Chi Wang, Robert Sim, Subhabrata Mukherjee, Victor Rühle, Laks V. S. Lakshmanan, and Ahmed Hassan Awadallah. Hybrid LLM: Cost-efficient and quality-aware query routing. In *International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=02f3mUtqnM.

Wenzhi Fang, Dong-Jun Han, Liangqi Yuan, Evan Chen, and Christopher G. Brinton. Bridging on-device and cloud LLMs for collaborative reasoning: A unified methodology for local routing and post-training. In *International Conference on Machine Learning*, 2026. URL https://arxiv.org/abs/2509.24050.

Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. Measuring massive multitask language understanding. In *International Conference on Learning Representations*, 2021a. URL https://arxiv.org/abs/2009.03300.

Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Song, and Jacob Steinhardt. Measuring mathematical problem solving with the MATH dataset. In *Proceedings of the Neural Information Processing Systems Track on Datasets and Benchmarks*, volume 1, 2021b. URL https://datasets-benchmarks-proceedings.neurips.cc/paper_files/paper/2021/hash/be83ab3ecd0db773eb2dc1b0a17836a1-Abstract-round2.html.

Xiaoyan Hu, Ho-Fung Leung, and Farzan Farnia. PAK-UCB contextual bandit: An online learning approach to prompt-aware selection of generative models and LLMs. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 24447–24481. PMLR, 2025. URL https://proceedings.mlr.press/v267/hu25m.html.

Shogo Iwazaki and Shion Takeno. Improved regret analysis in Gaussian process bandits: Optimality for noiseless reward, RKHS norm, and non-stationary variance. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 26642–26672. PMLR, 2025. URL https://proceedings.mlr.press/v267/iwazaki25a.html.

Zeyu Jia, Jian Qian, Alexander Rakhlin, and Chen-Yu Wei. How does variance shape the regret in contextual bandits? In *Advances in Neural Information Processing Systems*, volume 37, pp. 83730–83785, 2024. URL https://proceedings.neurips.cc/paper_files/paper/2024/hash/9861a7c3972ed5d36dda3826d44bb246-Abstract-Conference.html.

Mandar Joshi, Eunsol Choi, Daniel Weld, and Luke Zettlemoyer. TriviaQA: A large scale distantly supervised challenge dataset for reading comprehension. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 1601–1611, 2017. doi: 10.18653/v1/P17-1147. URL https://aclanthology.org/P17-1147/.

Kwang-Sung Jun and Jungtaek Kim. Noise-adaptive confidence sets for linear bandits and application to bayesian optimization. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 22643–22671. PMLR, 2024. URL https://proceedings.mlr.press/v235/jun24a.html.

Ehud Lehrer and Tao Wang. The value of information in stopping problems. *Economic Theory*, 78 (2):619–648, 2024. doi: 10.1007/s00199-023-01543-8.

Hunter Lightman, Vineet Kosaraju, Yura Burda, Harri Edwards, Bowen Baker, Teddy Lee, Jan Leike, John Schulman, Ilya Sutskever, and Karl Cobbe. Let's verify step by step. In *International Conference on Learning Representations*, 2024. URL https://proceedings.iclr.cc/paper_files/paper/2024/file/aca97732e30bcf1303bc22ac3924fd16-Paper-Conference.pdf.

Isaac Ong, Amjad Almahairi, Vincent Wu, Wei-Lin Chiang, Tianhao Wu, Joseph E. Gonzalez, M. Waleed Kadous, and Ion Stoica. RouteLLM: Learning to route LLMs from preference data. In *International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=8sSqNntaMr.

Pranoy Panda, Raghav Magazine, Chaitanya Devaguptapu, Sho Takemori, and Vishal Sharma. Adaptive LLM routing under budget constraints. In *Findings of the Association for Computational Linguistics: EMNLP 2025*, pp. 23934–23949. Association for Computational Linguistics, 2025. doi: 10.18653/v1/2025.findings-emnlp.1301. URL https://aclanthology.org/2025.findings-emnlp.1301/.

Qwen Team. Qwen2.5 technical report. *arXiv preprint arXiv:2412.15115*, 2024. URL https://arxiv.org/abs/2412.15115.

Ritik Raj, Souvik Kundu, Sarbartha Banerjee, Dheemanth Joshi, Ishita Vohra, and Tushar Krishna. TRACE-Router: Task-consistent and adaptive online routing for agentic AI. *arXiv preprint arXiv:2607.22465*, 2026. doi: 10.48550/arXiv.2607.22465. URL https://arxiv.org/abs/2607.22465.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. SQuAD: 100,000+ questions for machine comprehension of text. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pp. 2383–2392, 2016. doi: 10.18653/v1/D16-1264. URL https://aclanthology.org/D16-1264/.

Michael Scharf. On the response time of large-scale composite web services. In *Proceedings of the 19th International Teletraffic Congress (ITC 19)*, pp. 1807–1816, 2005. URL https://content.ikr.uni-stuttgart.de/Content/Publications/Archive/Sf_ITC19_36420.pdf.

Florian Schlechte, Yusra Tasnim Farooqi, and Matthias Hirth. Assessing the influence of end-to-end request latency and response length on waiting time satisfaction of LLM users. In *2025 17th International Conference on Quality of Multimedia Experience (QoMEX)*. IEEE, 2025. doi: 10.1109/QoMEX65720.2025.11219949.

Biao Sun, Ziming Huang, Hanyu Zhao, Wencong Xiao, Xinyi Zhang, Yong Li, and Wei Lin. Llumnix: Dynamic scheduling for large language model serving. In *18th USENIX Symposium on Operating Systems Design and Implementation (OSDI 24)*, pp. 173–191. USENIX Association, 2024. URL https://www.usenix.org/conference/osdi24/presentation/sun-biao.

Felicia Fang-Yi Tan, Moritz A. Messerschmidt, Wen Yin, and Oded Nov. The impact of response latency and task type on human–LLM interaction and perception. In *Proceedings of the 2026 CHI Conference on Human Factors in Computing Systems*. Association for Computing Machinery, 2026. doi: 10.1145/3772318.3790716. URL https://arxiv.org/abs/2604.06183.

Sattar Vakili, Kia Khezeli, and Victor Picheny. On information gain and regret bounds in Gaussian process bandits. In *Proceedings of the 24th International Conference on Artificial Intelligence and Statistics*, volume 130 of *Proceedings of Machine Learning Research*, pp. 82–90. PMLR, 2021. URL https://proceedings.mlr.press/v130/vakili21a.html.

Udaykiran Vallamsetty, Krishna Kant, and Prasant Mohapatra. Characterization of E-Commerce traffic. *Electronic Commerce Research*, 3(1–2):167–192, 2003. doi: 10.1023/A:1021585529079. URL https://www.kkant.net/papers/uday_journal.pdf.

Da Wang, Gauri Joshi, and Gregory W. Wornell. Efficient straggler replication in large-scale parallel computing. *ACM Transactions on Modeling and Performance Evaluation of Computing Systems*, 4(2), 2019. doi: 10.1145/3310336.

Ruitu Xu, Yifei Min, and Tianhao Wang. Noise-adaptive thompson sampling for linear contextual bandits. In *Advances in Neural Information Processing Systems*, volume 36, pp. 23630–23657, 2023. doi: 10.52202/075280-1026. URL https://proceedings.neurips.cc/paper_files/paper/2023/file/4a6824f8f137e78f18e73d9cfc1d22ed-Paper-Conference.pdf.

Heyang Zhao, Jiafan He, Dongruo Zhou, Tong Zhang, and Quanquan Gu. Variance-dependent regret bounds for linear bandits and reinforcement learning: Adaptivity and computational efficiency. In *Proceedings of Thirty Sixth Conference on Learning Theory*, volume 195 of *Proceedings of Machine Learning Research*, pp. 4977–5020. PMLR, 2023. URL https://proceedings.mlr.press/v195/zhao23a.html.

A SA CONFIDENCE COVERAGE AND KERNEL COMPUTATION

We use the bounded terminal feedback and conditional-mean model in Section 3, Assumption 2, and the updates in Section 4. The specified kernel is measurable, its RKHS is separable, and $\|\Phi(\xi)\|_{\mathcal{H}}^2 = \kappa(\xi,\xi) = C^2$. Throughout, $\lambda > 0$ and $H = \log(6/\delta)$ for $\delta \in (0,1)$. Let $\mathcal{F}_i = \sigma(\mathcal{G}_i, Y_i, D_i)$ denote the information after round i. Sequential feedback gives $\mathcal{F}_{i-1} \subseteq \mathcal{G}_i \subseteq \mathcal{F}_i$; all predictions and observation weights used at round i are $\mathcal{G}_i$-measurable.

A.1 WEIGHTED RIDGE AND DETERMINANT IDENTITIES

Set $\eta_i = r_i - g(\xi_i)$, $x_i = w_i\Phi(\xi_i)$, $y_i = w_i r_i$, $\nu_i = w_i\eta_i$, and $p_i = \langle x_i, \widehat{g}_{i-1} - g\rangle$. Then $e_i = \nu_i - p_i$ and $V_t = \lambda I + \sum_{i\le t} x_i \otimes x_i$. Define

$$F_t(v) = \lambda\|v\|_{\mathcal{H}}^2 + \sum_{i\le t}(y_i - \langle x_i, v\rangle)^2, \qquad E_t = \sum_{i\le t} e_i^2.$$

The estimator $\widehat{g}_t$ uniquely minimizes F_t. The weighting rule ensures $h_i^2 = \langle x_i, V_{i-1}^{-1}x_i\rangle \le 1$.

Lemma A.1 (Weighted ridge identities)**.** *For every finite prefix and every $v \in \mathcal{H}$, pathwise,*

$$\begin{aligned} F_t(v) - F_t(\widehat{g}_t) &= \|v - \widehat{g}_t\|_{V_t}^2, \\ F_t(\widehat{g}_t) &= E_t - P_t, \\ \|\widehat{g}_t - g\|_{V_t}^2 &= \lambda\|g\|_{\mathcal{H}}^2 + P_t + \sum_{i\le t}(2p_i\nu_i - p_i^2). \end{aligned} \tag{A.1}$$

For the weighted Gram matrix $G_t = [w_i w_j \kappa(\xi_i,\xi_j)]_{i,j\le t}$,

$$\mathcal{D}_t := \log\det(I_t + G_t/\lambda) = \sum_{i\le t}\log(1 + h_i^2). \tag{A.2}$$

Proof. The proof uses ridge and determinant identities common in linear bandit analysis (Abbasi-Yadkori et al., 2011). The normal equations and completion of the square give the first identity. A rank-one inverse update gives $F_i(\widehat{g}_i) - F_{i-1}(\widehat{g}_{i-1}) = e_i^2/(1+h_i^2)$. Summing from $F_0(0) = 0$ proves the second identity. Since $F_t(g) = \lambda\|g\|_{\mathcal{H}}^2 + \sum_i \nu_i^2$ and $\nu_i^2 - e_i^2 = 2p_i\nu_i - p_i^2$, their difference proves the third. The finite-rank determinant lemma gives $\mathcal{D}_i - \mathcal{D}_{i-1} = \log(1 + h_i^2)$. These calculations apply on the finite span of the observed features, with the orthogonal component contributing only the ridge term. □

A.2 SIMULTANEOUS CONFIDENCE COVERAGE

Theorem A.1 (SA confidence coverage)**.** *Under Assumption 2, with probability at least $1 - \delta/6$, simultaneously for all $t \ge 0$ and all route–query pairs ξ,*

$$\|\widehat{g}_t - g\|_{V_t}^2 \le q_t, \qquad |\widehat{g}_t(\xi) - g(\xi)| \le \sqrt{q_t}\,\|\Phi(\xi)\|_{V_t^{-1}}. \tag{A.3}$$

Consequently, on the same event, for every $n \ge 1$ and ξ,

$$g(\xi) \le \mathrm{UCB}_n(\xi) = \widehat{g}_{n-1}(\xi) + \sqrt{q_{n-1}}\,u_n(\xi). \tag{A.4}$$

Proof. Conditional on $\mathcal{G}_i$, ν_i has mean zero and lies in an interval of length w_i, while p_i and w_i are known. Conditional Hoeffding's inequality therefore gives

$$\mathbb{E}\big[\exp\{2(2p_i\nu_i - p_i^2)\} \mid \mathcal{G}_i\big] \le \exp\{2(w_i^2 - 1)p_i^2\} \le 1.$$

By the tower property, $\exp\{2\sum_{i\le t}(2p_i\nu_i - p_i^2)\}$ is a nonnegative supermartingale with respect to $\mathcal{F}_t$, starting at one. Ville's inequality gives, with failure probability at most $e^{-H} = \delta/6$, the bound $\sum_{i\le t}(2p_i\nu_i - p_i^2) \le H/2$ for every t. Equation (A.1) and $\|g\|_{\mathcal{H}} \le 1$ yield the norm bound. At $t = 0$, it follows from $\|g\|_{V_0}^2 \le \lambda \le q_0$. Cauchy–Schwarz gives the prediction bound for every ξ on this event. Taking $t = n - 1$ proves the routing-score bound. □

A.3 KERNEL-MATRIX EXPRESSIONS

After t observations, let $y = (w_i r_i)_{i \le t}$ and $v(\xi) = (w_i \kappa(\xi_i, \xi))_{i \le t}$. The weighted ridge normal equations and the Woodbury identity give

$$\begin{aligned} \widehat{g}_t(\xi) &= v(\xi)^\top (G_t + \lambda I_t)^{-1} y, \\ u_{t+1}^2(\xi) &= \frac{\kappa(\xi,\xi) - v(\xi)^\top (G_t + \lambda I_t)^{-1} v(\xi)}{\lambda}, \end{aligned} \tag{A.5}$$

where I_t is the $t \times t$ identity matrix.

B DELAY VARIANCE UNDER EDGE TAKEOVER

All distributions and moments in this appendix are conditional on $\mathcal{G}_n$. We write $z = z_n$ and $S_z(t) = \Pr(D_c > t \mid \mathcal{G}_n)$, with the dependence on the remaining history suppressed.

B.1 TRUNCATED CLOUD-DELAY MOMENTS

For any nonnegative cloud completion time, the conditional mean and variance of $\min\{D_c, b\}$ are

$$m_z(b) = \int_0^b S_z(t)\, dt, \qquad V_z(b) = 2\int_0^b t S_z(t)\, dt - m_z(b)^2. \tag{B.1}$$

These identities follow by integrating the survival function. Under the cloud-delay model (2), write $x = x(z)$ and $\alpha = \alpha(z)$. Substituting the Pareto survival function gives, for $b \ge x$,

$$m_z(b) = \frac{\alpha x - x^\alpha b^{1-\alpha}}{\alpha - 1}, \qquad V_z(b) = \frac{2x^\alpha b^{2-\alpha} - \alpha x^2}{2-\alpha} - m_z(b)^2. \tag{B.2}$$

Thus v_0, the supremum of $V_z(D_{\max})$ over admissible histories and queries, is finite despite the infinite second moment of the untruncated cloud delay.

B.2 VARIANCE COMPARISON FOR A GENERAL CLOUD-DELAY DISTRIBUTION

Lemma B.1 (Delay-variance comparison under takeover)**.** *Let $D_0 = \min\{D_c, D_{\max}\}$, $p_s = S_z(s) > 0$, $a_s = s - m_z(s)$, and*

$$e_s = \mathbb{E}[D_0 - s \mid D_c > s, \mathcal{G}_n] = \frac{1}{p_s}\int_s^{D_{\max}} S_z(t)\, dt.$$

Suppose the additional takeover time E lies in $[0, d_e]$ and $s + d_e \le D_{\max}$. For $D_s = \min\{D_c, s\} + E\mathbf{1}\{D_c > s\}$,

$$\mathrm{Var}(D_0 \mid \mathcal{G}_n) - \mathrm{Var}(D_s \mid \mathcal{G}_n) \ge p_s\{2a_s(e_s - d_e) - d_e^2\}. \tag{B.3}$$

The variables E and D_c may be dependent.

Proof. Put $A = \min\{D_c, s\}$ and $Z = E\mathbf{1}\{D_c > s\}$, so $D_s = A + Z$. Writing $\mu = \mathbb{E}[Z \mid \mathcal{G}_n] \le p_s d_e$, the identity $AZ = sZ$ gives $\mathrm{Cov}(A, Z \mid \mathcal{G}_n) = a_s\mu$. Also, $\mathrm{Var}(Z \mid \mathcal{G}_n) \le \mathbb{E}[Z^2 \mid \mathcal{G}_n] \le p_s d_e^2$. Therefore

$$\mathrm{Var}(D_s \mid \mathcal{G}_n) \le V_z(s) + 2a_s p_s d_e + p_s d_e^2.$$

Equation (B.1) gives $V_z'(b) = 2S_z(b)[b - m_z(b)]$ almost everywhere. Since $b - m_z(b)$ is nondecreasing,

$$V_z(D_{\max}) - V_z(s) \ge 2a_s \int_s^{D_{\max}} S_z(t)\, dt = 2a_s p_s e_s.$$

Subtracting the two bounds proves the claim. □

If $p_s = 0$, takeover never occurs and $D_s = D_0$ almost surely under the conditional distribution.

B.3 Proof of Proposition 1

Proof. Assumption 1 and the timing condition (1) imply

$$\Pr(D_{\mathrm{c}} \le s/2 \mid \mathcal{G}_n) = 1 - (2x/s)^{\alpha} \ge \tfrac{1}{2}, \qquad a_s = \mathbb{E}[(s - D_{\mathrm{c}})_+ \mid \mathcal{G}_n] \ge s/4.$$

Using $\alpha < 2$ and $D_{\max} \ge 2s$,

$$e_s = \int_s^{D_{\max}} (s/t)^{\alpha}\, dt \ge \int_s^{2s} (s/t)^2\, dt = s/2.$$

Moreover, $d_e \le s/4$ and $p_s = (x/s)^{\alpha} \ge (x_-/s)^{\alpha_+} > 0$. Lemma B.1 therefore yields

$$\mathrm{Var}(D_0 \mid \mathcal{G}_n) - \mathrm{Var}(D_s \mid \mathcal{G}_n) \ge p_s \left\{ 2\frac{s}{4}\left(\frac{s}{2} - \frac{s}{4}\right) - \left(\frac{s}{4}\right)^2 \right\} \ge \frac{s^2}{16}(x_-/s)^{\alpha_+} = k_s.$$

This bound is uniform over histories and queries. Direct-edge delay lies in $[0, d_e]$ and has conditional variance at most $d_e^2/4$. Hence both initial routes have delay variance at most $v_s = \max\{d_e^2/4, v_0 - k_s\}$.

For $Y \in [0,1]$, the triangle inequality for centered L^2 variables gives

$$\sqrt{\mathrm{Var}(Y - \beta D \mid \mathcal{G}_n)} \le \sqrt{\mathrm{Var}(Y \mid \mathcal{G}_n)} + \beta\sqrt{\mathrm{Var}(D \mid \mathcal{G}_n)} \le \tfrac{1}{2} + \beta\sqrt{\mathrm{Var}(D \mid \mathcal{G}_n)}. \tag{B.4}$$

Thus the utility variance is bounded by $\bar{v}_s = (1/2 + \beta\sqrt{v_s})^2$ for both initial routes, allowing dependence between the user rating and delay. □

C Regret Bounds and Convergence

The confidence and regret arguments use Assumption 2. We first establish an information bound for the specified kernel, then control regret through the observed prediction errors and service variance. Throughout this appendix, $\mathcal{A}$ uses the routing and update rules of Algorithm 1 for the service being analyzed. Its cumulative regret $R_N(\mathcal{A})$ is defined in Section 3.3, relative to that service's oracle.

C.1 Proof of Lemma 1

Proof. For a multi-index $\nu \in \mathbb{N}_0^d$, define

$$\varphi_\nu(z) = \exp\left(-\frac{\|z\|_2^2}{2h^2}\right)\frac{z^\nu}{h^{|\nu|}\sqrt{\nu!}}.$$

The squared-exponential kernel has the expansion

$$\exp\left(-\frac{\|z - z'\|_2^2}{2h^2}\right) = \sum_{\nu \in \mathbb{N}_0^d} \varphi_\nu(z)\varphi_\nu(z').$$

Set $m_N = \lceil \max\{2e/h^2, \log_2(N+1)\} \rceil$. Retaining all features with $|\nu| < m_N$ and the constant component, separately for each route, gives a positive semidefinite kernel of rank at most

$$r_N = 2\left[1 + \binom{d + m_N - 1}{d}\right].$$

Its residual is positive semidefinite. With $u = \|z\|_2^2/h^2 \le h^{-2}$, the diagonal residual of the squared-exponential component satisfies

$$e^{-u}\sum_{j=m_N}^{\infty}\frac{u^j}{j!} \le \frac{u^{m_N}}{m_N!} \le \left(\frac{e}{h^2 m_N}\right)^{m_N} \le 2^{-m_N} \le \frac{1}{N+1}.$$

Consequently, every routing Gram matrix has a decomposition $K_N = A_N + E_N$, where $A_N, E_N \succeq 0$, $\mathrm{rank}(A_N) \le r_N$, $\mathrm{tr}(A_N) \le NC^2$, and $\mathrm{tr}(E_N) \le NC^2/[2(N+1)]$. The log-determinant and trace inequalities give

$$\begin{aligned}\log\det(I_N + K_N/\lambda) &\le \log\det(I_N + A_N/\lambda) + \lambda^{-1}\,\mathrm{tr}(E_N) \\ &\le r_N \log\left(1 + \frac{NC^2}{\lambda r_N}\right) + \frac{C^2}{2\lambda}.\end{aligned}$$

This bound is uniform over all action–query sequences. For fixed d, h, C, λ, $m_N = O(\log(N+1))$ and $r_N = O((\log(N+1))^d)$. Therefore $\Gamma_N = O((\log(N+1))^{d+1}) = o(\sqrt{N})$. □

C.2 Kernel Information and Per-round Regret

For the weighted Gram matrix G_N from Appendix A.1, the unweighted Gram matrix $K_N = [\kappa(\xi_i, \xi_j)]_{i,j \le N}$, and the worst-case information Γ_N defined in Section 5,

$$\mathcal{D}_N = \log\det(I_N + G_N/\lambda) \le \log\det(I_N + K_N/\lambda) \le 2\Gamma_N. \tag{C.1}$$

The bound $\kappa(\xi, \xi) \le C^2$ makes Γ_N finite for every N.

Lemma C.1 (Accumulated uncertainty and per-round regret)**.** *Pathwise,*

$$\sum_{n \le N} h_n^2 \le \frac{\mathcal{D}_N}{\log 2} \le \frac{2\Gamma_N}{\log 2}. \tag{C.2}$$

On the confidence event of Theorem A.1,

$$\Delta_n := \max_a g(a, z_n) - g(a_n, z_n) \le \min\{1, 2\sqrt{q_{n-1}}u_n(\xi_n)\} \le 2\sqrt{q_{n-1}}h_n. \tag{C.3}$$

Proof. For $h_n^2 \in [0, 1]$, concavity gives $\log(1 + h_n^2) \ge h_n^2 \log 2$; summing and using (A.2) proves the first information bound. In feature space, $\sum_n w_n^2 \Phi(\xi_n) \otimes \Phi(\xi_n) \preceq \sum_n \Phi(\xi_n) \otimes \Phi(\xi_n)$. Determinant monotonicity on their finite span proves (C.1) and the second bound.

Let $\xi_n^* = (a_n^*, z_n)$. Confidence coverage and score maximization give

$$g(\xi_n^*) \le \mathrm{UCB}_n(\xi_n^*) \le \mathrm{UCB}_n(\xi_n) = \widehat{g}_{n-1}(\xi_n) + \sqrt{q_{n-1}}u_n(\xi_n).$$

Subtracting the confidence lower bound on $g(\xi_n)$ gives $\Delta_n \le 2\sqrt{q_{n-1}}u_n(\xi_n)$. Also, $0 \le \Delta_n \le 1$ because normalized mean utilities lie in $[0, 1]$. If $u_n(\xi_n) \le 1$, then $h_n = u_n(\xi_n)$. Otherwise $h_n = 1$, and

$$2\sqrt{q_{n-1}} \ge \sqrt{2H} > \sqrt{2\log 6} > 1,$$

since $q_{n-1} \ge H/2$ and $\delta \in (0, 1)$. This proves the last inequality for every $\lambda > 0$. □

C.3 Regret Certificate from Observed Prediction Errors

Proposition C.1 (Observed-loss regret certificate)**.** *On the confidence event of Theorem A.1, simultaneously for all $N \ge 1$,*

$$R_N(\mathcal{A}) \le B \min\left\{N, 2\sqrt{\frac{N q_{N-1}\mathcal{D}_N}{\log 2}}\right\}. \tag{C.4}$$

The quantities q_{N-1} and $\mathcal{D}_N$ are computed from the observed routing history.

Proof. Every increment of P_t is nonnegative, so q_t is nondecreasing. Lemma C.1 and Cauchy–Schwarz yield

$$\sum_{n \le N} \Delta_n \le 2\sqrt{q_{N-1}} \sum_{n \le N} h_n \le 2\sqrt{N q_{N-1} \sum_{n \le N} h_n^2} \le 2\sqrt{N q_{N-1}\mathcal{D}_N / \log 2}.$$

Use $\sum_{n \le N} \Delta_n \le N$ and $R_N(\mathcal{A}) = B \sum_{n \le N} \Delta_n$. □

C.4 Variance Bound for the SA Confidence Scale

Suppose $\mathrm{Var}(U_i \mid \mathcal{G}_i) \le \bar{v}$ uniformly for a deterministic $\bar{v}$. Using the noise variables from Appendix A.1, define

$$d_i = \frac{h_i^2}{1 + h_i^2}, \quad c_i = 1 - 2d_i, \quad Z_t = \sum_{i \le t} d_i w_i^2 \eta_i^2, \quad M_t = \sum_{i \le t} d_i w_i^2 \,\mathrm{Var}(r_i \mid \mathcal{G}_i).$$

The weights d_i, c_i, w_i are $\mathcal{G}_i$-measurable, with $0 \le d_i \le 1/2$ and $0 \le c_i \le 1$.

Lemma C.2 (Noise energy and weighted prediction error)**.** *There are two events, each with failure probability at most $e^{-H} = \delta/6$, on which, respectively, all $t \geq 0$ satisfy*

$$\sum_{i \leq t} c_i(2p_i\nu_i - p_i^2) \leq H/2, \tag{C.5}$$

and

$$Z_t \leq (e-1)M_t + H/2 \leq (e-1)\frac{\bar{v}}{B^2}\mathcal{D}_t + H/2. \tag{C.6}$$

Proof. Conditional on $\mathcal{G}_i$, η_i has mean zero and lies in an interval of length one. Hoeffding's inequality gives $\mathbb{E}[e^{a\eta_i} \mid \mathcal{G}_i] \leq e^{a^2/8}$ and hence

$$\mathbb{E}\Big[e^{2c_i(2p_i\nu_i - p_i^2)} \mid \mathcal{G}_i\Big] \leq \exp\{2c_ip_i^2(c_iw_i^2 - 1)\} \leq 1.$$

The associated product is a nonnegative supermartingale. Ville's inequality proves (C.5) simultaneously for all t with failure probability at most e^{-H}.

For the second event, set $X_i = d_iw_i^2\eta_i^2 \in [0, 1/2]$ and $\mu_i = \mathbb{E}[X_i \mid \mathcal{G}_i] = d_iw_i^2 \operatorname{Var}(r_i \mid \mathcal{G}_i)$. Convexity gives $e^{2X_i} \leq 1 + 2(e-1)X_i$, so

$$\mathbb{E}[e^{2X_i - 2(e-1)\mu_i} \mid \mathcal{G}_i] \leq 1.$$

Ville's inequality applied to this product gives $Z_t \leq (e-1)M_t + H/2$ for all t with the same failure bound. Finally, $\operatorname{Var}(r_i \mid \mathcal{G}_i) \leq \bar{v}/B^2$ and $d_iw_i^2 \leq \log(1 + h_i^2)$ imply $M_t \leq (\bar{v}/B^2)\mathcal{D}_t$. □

Proposition C.2 (Variance bound for the SA scale)**.** *On the intersection of the events in Lemma C.2, simultaneously for all $N \geq 1$ and $0 \leq t \leq N$,*

$$\begin{aligned} &P_t \leq \lambda + 2Z_t + H/2 \leq \lambda + 2(e-1)M_t + \tfrac{3}{2}H, \\ &q_t \leq 2\lambda + 2(e-1)M_t + 2H \leq c_v\frac{\bar{v}}{B^2}\Gamma_N + 2\lambda + 2H, \qquad c_v = 4(e-1). \end{aligned} \tag{C.7}$$

Proof. Since $e_i = \nu_i - p_i$,

$$P_t = \sum_{i \leq t} d_i(\nu_i - p_i)^2 = Z_t - \sum_{i \leq t} d_i(2p_i\nu_i - p_i^2).$$

Combining this equality with (A.1) gives

$$P_t + \|\widehat{g}_t - g\|_{V_t}^2 = \lambda\|g\|_{\mathcal{H}}^2 + 2Z_t + \sum_{i \leq t} c_i(2p_i\nu_i - p_i^2). \tag{C.8}$$

Nonnegativity of the squared norm, $\|g\|_{\mathcal{H}} \leq 1$, and (C.5) give $P_t \leq \lambda + 2Z_t + H/2$. Substitute (C.6) and $q_t = \lambda + P_t + H/2$ to obtain the bounds involving M_t. Finally, $M_t \leq (\bar{v}/B^2)\mathcal{D}_t \leq 2(\bar{v}/B^2)\Gamma_N$ gives the stated bound for q_t. The same inequalities hold at $t = 0$. □

C.5 Proof of Theorem 1

Proof. First consider any fixed service with utility in $[-\beta D_{\max}, 1]$ satisfying Assumption 2 and the conditional-mean model, with a deterministic uniform bound $\operatorname{Var}(U_n \mid \mathcal{G}_n) \leq \bar{v}$.

The confidence event and the two events in Lemma C.2 each fail with probability at most $\delta/6$. Their intersection has probability at least $1 - \delta/2$, simultaneously over all rounds. The proof of Proposition C.1 applies to this service and gives

$$R_N(\mathcal{A}) \leq \min\{BN, 2B\sqrt{Nq_{N-1}\mathcal{D}_N/\log 2}\}.$$

Substituting $q_{N-1} \leq c_v(\bar{v}/B^2)\Gamma_N + 2\lambda + 2H$ from Proposition C.2 and $\mathcal{D}_N \leq 2\Gamma_N$ gives

$$R_N(\mathcal{A}) \leq \min\{BN, \mathcal{R}_N(\bar{v})\} \qquad \text{for all } N \geq 1. \tag{C.9}$$

For takeover under the cloud-delay model (2) and Assumption 1, the variance bound proved in Appendix B allows us to take $\bar{v} = \bar{v}_s$. This establishes the cumulative-regret bound in Theorem 1 with probability at least $1 - \delta/2 \geq 1 - \delta$.

For fixed service and kernel parameters and fixed δ, dividing $\mathcal{R}_N(\bar{v}_s)$, defined in Theorem 1, by N gives

$$\frac{\mathcal{R}_N(\bar{v}_s)}{N} = 2\sqrt{\frac{2\Gamma_N}{N\log 2}\left[c_v\bar{v}_s\Gamma_N + B^2(2\lambda+2H)\right]}.$$

Since $\Gamma_N \geq 1$, Lemma 1 yields

$$\frac{\mathcal{R}_N(\bar{v}_s)}{N} = O\left(\frac{\Gamma_N}{\sqrt{N}}\right) = O\left(\frac{(\log(N+1))^{d+1}}{\sqrt{N}}\right) \longrightarrow 0,$$

which proves (14). The cumulative-regret guarantee gives $0 \leq R_N(\mathcal{A})/N \leq \mathcal{R}_N(\bar{v}_s)/N$ on the event above, so the average regret defined in Section 3.3 also converges to zero with the stated probability. □

C.6 Proof of Corollary 1

Proof. The conditions $k_s > 0$ and $v_0 > d_e^2/4$ give $v_s = \max\{d_e^2/4, v_0 - k_s\} < v_0$. For deadline-only service, v_0 bounds the cloud-delay variance and also the direct-edge variance, which is at most $d_e^2/4$. Applying (B.4) gives $\bar{v}_0 = (1/2 + \beta\sqrt{v_0})^2$, with

$$\bar{v}_0 - \bar{v}_s = \beta(\sqrt{v_0} - \sqrt{v_s}) + \beta^2(v_0 - v_s) > 0. \tag{C.10}$$

Both services use the same kernel, normalization B, and confidence parameter H. Each has its own conditional-mean oracle and feedback history, while Γ_N is common. Apply (C.9) with $\bar{v}_s$ and $\bar{v}_0$, respectively. Each event has failure probability at most $\delta/2$, so both bounds hold jointly with probability at least $1-\delta$. Their statistical bounds satisfy

$$\frac{\mathcal{R}_N(\bar{v}_0)}{\mathcal{R}_N(\bar{v}_s)} = \sqrt{1 + \frac{c_v\Gamma_N(\bar{v}_0 - \bar{v}_s)}{c_v\bar{v}_s\Gamma_N + B^2(2\lambda+2H)}} > 1.$$

Under the horizon condition $\mathcal{R}_N(\bar{v}_s) < BN$,

$$\frac{\min\{BN, \mathcal{R}_N(\bar{v}_0)\}}{\min\{BN, \mathcal{R}_N(\bar{v}_s)\}} = \min\left\{\frac{BN}{\mathcal{R}_N(\bar{v}_s)}, \frac{\mathcal{R}_N(\bar{v}_0)}{\mathcal{R}_N(\bar{v}_s)}\right\} > 1.$$

The horizon condition is equivalent to

$$N > \frac{8}{\log 2}\left[c_v\frac{\bar{v}_s}{B^2}\Gamma_N^2 + (2\lambda+2H)\Gamma_N\right]. \tag{C.11}$$

By Lemma 1, the right-hand side is $O((\log(N+1))^{2d+2}) = o(N)$. Thus a finite N_0 exists such that (C.11) holds for every $N \geq N_0$, with all other parameters fixed.

For a conservative check, $D_0 \in [0, D_{\max}]$ gives $v_0 \leq D_{\max}^2/4$, and (1) gives $d_e \leq D_{\max}/8$. Hence $v_s \leq D_{\max}^2/4$ and $\bar{v}_s \leq B^2/4$. It therefore suffices that

$$N > \frac{8}{\log 2}\left[(e-1)\Gamma_N^2 + (2\lambda+2H)\Gamma_N\right].$$

Finally, repeating a single action–query pair gives $\Gamma_N \geq \frac{1}{2}\log(1 + NC^2/\lambda) \to \infty$. Both statistical bounds eventually lie below BN, and hence

$$\frac{\min\{BN, \mathcal{R}_N(\bar{v}_0)\}}{\min\{BN, \mathcal{R}_N(\bar{v}_s)\}} \longrightarrow \sqrt{\frac{\bar{v}_0}{\bar{v}_s}} > 1. \tag{C.12}$$

These ratios compare the regret upper bounds of the two services. □

C.7 Average Regret and Expected Utility

Corollary C.1 (Expected utility relative to the service oracle)**.** *For exogenous queries, let $J_{N,*}^{(s)}$ denote the oracle's expected average utility under the same service as the learner. Under Theorem 1,*

$$J_{N,*}^{(s)} - J_N(\mathcal{A}) = \frac{\mathbb{E}R_N(\mathcal{A})}{N} \leq \frac{\mathcal{R}_N(\bar{v}_s)}{N} + B\delta. \tag{C.13}$$

For horizon-N runs $\mathcal{A}_N$ of the same algorithm initialized with $\delta_N = (N+1)^{-2}$, with the same service parameters, kernel, and regularization, Lemma 1 yields

$$J_{N,*}^{(s)} - J_N(\mathcal{A}_N) \longrightarrow 0. \tag{C.14}$$

Proof. The conditional-mean model gives $\mathbb{E}U_n = \mathbb{E}f_{a_n}(z_n)$. Exogeneity lets the learner and oracle be evaluated on the same query distribution, yielding the identity in (C.13). Apply Theorem 1 on its probability-$1-\delta$ event and use $R_N(\mathcal{A}) \le BN$ on the complement. For $\delta_N = (N+1)^{-2}$, $H_N = O(\log N)$. Together with Lemma 1, this gives $\Gamma_N^2/N \to 0$, $H_N\Gamma_N/N \to 0$, and $B\delta_N \to 0$. This expectation limit applies to the stated family of horizon-N runs; for a continuing run with unchanged δ, (C.13) retains the $B\delta$ term. □

D Experimental Details

Experimental configurations. Table 3 lists the settings used in the main comparison and ablations. These configurations were selected through exploratory comparisons on the evaluation data, including searches based on gains over baselines. The selected configurations are supplied in `configs/current_sa.json`, and the QA threshold-search records are retained in `docs/selection/qa/`. The QA thresholds were selected using SA and then held for the standard-radius controls; all ablations use the same task settings. MATH-500 uses the selected configuration from a 192-point h, C, λ grid, with the same configuration in the main comparison, mechanisms, ablations, and sensitivity anchors. The confidence parameter gives $H = \log(6/\delta)$.

Table 3: SA task configurations. All tasks use $D_{\max} = 60$ s.

Task	N	β	d	h	λ	C	δ	s (s)
MATH-L	1500	0.08	16	1.866	0.25	2	0.5	30
MATH-500	500	0.08	16	1.866	0.25	1	0.05	30
MMLU	1500	0.03925	16	0.511	0.25	2	0.9	30
TriviaQA	1500	0.1678	64	4.180	0.25	2	0.1	4.75
SQuAD 1.1	1500	0.12	64	1.194	2	3	0.1	15.985
MBPP	500	0.012	128	0.083	4	2	0.05	30

Query representations. During Qwen2.5-1.5B-Instruct prefill, we extract the last nonpadding prompt-token states from zero-based blocks 25–27 before the final RMSNorm. The three vectors are individually ℓ_2-normalized and averaged. A deterministic SHA256-based Rademacher projection (seed 20260906) maps the resulting 1,536-dimensional vector to d dimensions, followed by ℓ_2 normalization. All dimensions reuse the saved pre-generation states; features depend only on the prompt.

Baseline adaptations and tuning. RouteLLM is a locally trained matrix-factorization head. At each task's β, it learns deadline-adjusted utility preferences on 155 historical MATH-Lighteval requests; 77 separate historical requests calibrate the utility-maximizing routing threshold. This 232-request history is disjoint in normalized query text from all main tasks and provides cross-domain calibration for MMLU, QA, and code. The head and threshold are refitted using deadline-adjusted historical utilities, then remain unchanged online. GAPG is a router-only adaptation with a randomly initialized binary policy head, updated from selected feedback using PPO/dual clipping; each query forms a singleton group with raw utility as its advantage. Updates use full batches; the final incomplete batch receives no update. PAK-UCB uses $k(z, z') = (5z^\top z' + 1)^3$, ridge regularization 1, $\delta = 0.05$, and exploration coefficient $\sqrt{2\log(4N/0.05)}$. Each arm is visited once within the horizon; updates use selected deadline-adjusted utility, and exact ties are randomized. We correct the upstream initial kernel inverse to $1/(k_{\text{self}}+1)$. Table 4 and `configs/baselines/deadline60.json` specify the resulting settings. GAPG and PAK-UCB are rerun from initialization using selected deadline-adjusted feedback. For last-minute service, an unfinished cloud call triggers a cold edge restart at $s = D_{\max} - 1$. RouteLLM is refitted and calibrated using historical utilities under this service rule, while GAPG and PAK-UCB are rerun from initialization using the corresponding selected-route terminal feedback.

Table 4: External baseline settings.

	RouteLLM				GAPG			PAK-UCB
Task	d	LR	Epochs	Noise	d	LR	Batch	d
MATH-L	16	3×10^{-4}	50	0.1	16	0.01	32	16
MATH-500	16	3×10^{-4}	50	0.1	16	0.01	32	16
MMLU	16	10^{-4}	50	0.1	16	0.01	32	16
TriviaQA	16	3×10^{-4}	50	0	64	0.01	8	64
SQuAD 1.1	64	3×10^{-4}	50	0	64	0.003	8	64
MBPP	64	3×10^{-4}	100	0.1	128	0.01	8	128

MF uses Adam, hidden dimension 128, batch size 64, and weight decay 10^{-5}. GAPG uses a 32-unit Tanh head, AdamW with weight decay 0.01, one update epoch per batch, PPO clip 0.2, dual clip 3, gradient clip 1, and zero entropy coefficient.

Environment and timing. Edge inference used a 32 GB vGPU with PyTorch 2.8.0 and CUDA 12.8. Recorded service delay includes setup and the selected model's execution path, including API calls and retries for cloud requests. Router CPU time, offline 64/128-dimensional projection, and grading are excluded. Takeover is replayed as a cold edge restart with zero takeover cost; live cancellation and takeover are not measured.

Repeated request orders. Each task trains a separate router. Main, mechanism, and ablation experiments use 15 paired request orders (seeds 5–19); sensitivity uses five (5–9). All orders replay the same cache, and the 15 orders reuse five RouteLLM/GAPG initializations.

Aggregate utility summaries. Let J_d and B_d denote the final cumulative average utilities of our system and the strongest external baseline on task d. The mean relative utility gain across the six tasks is $100\,6^{-1}\sum_d (J_d - B_d)/|B_d|$. The oracle percentages are ratios of macro means, $100\sum_d J_d/\sum_d O_d$ and $100\sum_d B_d/\sum_d O_d$, where O_d is the estimated reference value.

Estimated oracle reference. The estimated oracle uses cross-fitted conditional-utility predictions over four services: deadline-limited edge, cloud with takeover, and either model run to completion without a deadline. It is reported separately from the five baseline methods under both service rules. Its broader service set and estimated means make it a descriptive reference; the theoretical oracle in Section 5 uses the learner's service rules.

Table 5: SA delivered quality and service latency. Quality is accuracy for binary tasks and mean token F1 for QA. Reported p95 values average the sequence-level 95th percentiles.

Task	Quality	Mean delay (s)	p95 (s)	Cloud (%)	Timeout (%)
MATH-L	0.690	8.753	32.29	63.32	0.00
MATH-500	0.686	9.731	39.82	66.23	0.00
MMLU	0.721	4.109	26.94	50.62	0.00
TriviaQA	0.511	1.375	4.81	52.88	0.00
SQuAD 1.1	0.761	1.509	4.90	51.54	0.00
MBPP	0.468	10.036	33.35	51.19	0.00

D.1 Prompt Templates

SA routing uses only pre-generated prompt features and selected utility feedback. Exact message strings are supplied in `data/acquisition/PROMPTS.json`; the layout below is adjusted for readability.

MATH-Lighteval and MATH-500.

System (edge).

```
You are a concise mathematics solver.  Follow this output
protocol exactly.
Give only the essential calculations.  Do not restate the
problem and do not use headings.  Use at most six short
reasoning steps.  End with exactly one \boxed{answer} and
stop immediately after it.
```

User (edge).

```
Problem:
{question}
Mandatory format:  solve concisely and end with exactly one
\boxed{answer}.
```

System (cloud).

```
Solve the mathematics problem carefully.  Show concise
reasoning and end with the final answer in the form
\boxed{answer}.
```

User (cloud).

```
{question}
```

MMLU. The input appends `Choices:` and one labeled option per line.

System (edge).

```
You are a concise multiple-choice solver.  Follow this
output protocol exactly.
Give brief reasoning if useful.  Select exactly one of the
option letters supplied in the question, not the option text
or its numeric value.  Write your answer on a separate final
line in this format:  ANSWER: <letter>
Replace <letter> with the single supplied option letter.  Do
not use a box or math delimiters for this final line.  Stop
immediately after the letter.
```

User (edge).

```
Problem:
{question}
Mandatory format:  put ANSWER: <letter> on a separate final
line, replacing <letter> with exactly one supplied option
letter.  The final answer is a letter, not the option text
or its numeric value.
```

System (cloud).

```
Answer the multiple-choice question carefully.  Show concise
reasoning and select exactly one of the option letters
supplied in the question.  End with exactly one
\boxed{letter}, containing only that single option letter,
and stop immediately after it.
```

User (cloud).

```
{question}
```

TriviaQA.

System (edge).

```
Answer the trivia question from your own knowledge.  Return
only the shortest complete answer, without explanation,
alternatives, or instructions.  Write exactly one line:
ANSWER: <short answer>.  If you do not know, write ANSWER:
UNKNOWN. Stop after that line.
```

User (edge).

```
{question}
Return only the final answer now:  ANSWER: <short answer>.
```

System (cloud).

```
Answer the factual question from your own knowledge.  No
retrieval documents are provided.  Give one short answer, no
alternative guesses or explanations.  Write a final line
ANSWER: <short answer>.  If unable to answer, write ANSWER:
UNKNOWN. Stop after that line.
```

User (cloud).

```
{question}
```

SQuAD 1.1. The input is assembled as follows.

Input.

```
Context:
{context}
Question:
{question}
```

System (edge).

```
Answer the question using only the supplied context.  Return
the shortest complete answer span from that context, without
explanation or alternatives.  Write exactly one line:
ANSWER: <answer span>.  If unable to answer, write ANSWER:
UNKNOWN. Stop after that line.
```

User (edge).

```
{question}
Return the answer now on one line:  ANSWER: <answer span>.
```

System (cloud).

```
Answer the question using only the supplied context.  Return
the shortest complete answer span from that context, without
explanation or alternatives.  Write a final line ANSWER:
<answer span>.  If unable to answer, write ANSWER: UNKNOWN.
Stop after that line.
```

User (cloud).

```
{question}
```

MBPP. Required names and signatures are extracted from the reference AST.

Input.

```
{problem}
Required Python API (implement these definitions):
{API declarations}
```

System (edge).

```
Implement the requested Python program with the supplied API
names and signatures.  Include all necessary
standard-library imports and definitions.  Do not read stdin
or print demonstrations.  Return the complete implementation
in exactly one fenced python code block, without
explanations.  The program is evaluated using private tests
under Python 3.12 with its standard library.  Third-party
packages, filesystem access and network access are
unavailable.
```

User (edge).

```
{question}
```

System (cloud).

```
Implement the requested Python program with the supplied API
names and signatures.  Include all necessary
standard-library imports and definitions.  Do not read stdin
or print demonstrations.  Return the complete implementation
in exactly one fenced python code block, without
explanations.  The program is evaluated using private tests
under Python 3.12 with its standard library.  Third-party
packages, filesystem access and network access are
unavailable.
```

User (cloud).

```
{question}
```

E Additional Routing Mechanism Diagnostics

These diagnostics use $s = 30$ s and $D_{\max} = 60$ s on all six tasks.

How does confidence affect routing? Figure 4 compares the SA routing rule in (11) with the predicted-mean decision $\arg\max_{a \in \mathcal{U}} \widehat{g}_n(a, z_{n+1})$. Both use the same SA learning history and break ties toward edge. The confidence bonus changes the selected route on 19.8–31.9% of queries.

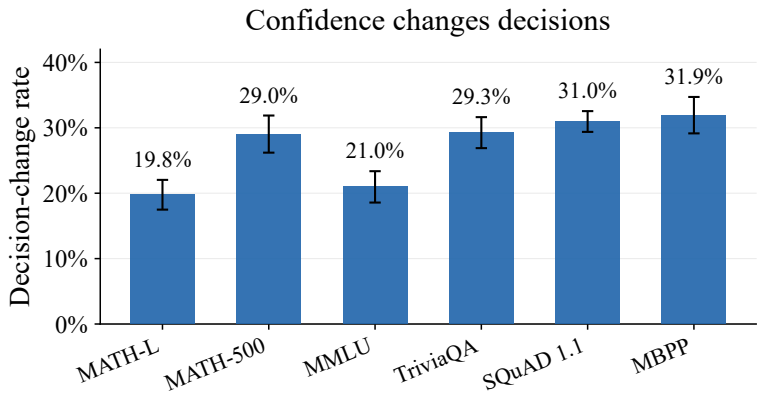


Figure 4: Disagreement between SA and predicted-mean decisions on the same learning history. Error bars show standard deviations.

Uncertainty and confidence bonuses. We track 32 query identities chosen independently of the recorded feedback, evaluating both routes offline along each SA history. Figures 5–7 show decreasing geometric uncertainty u and an increasing shared squared confidence scale q. The bonus $B\sqrt{q}\,u$ depends on both quantities. On MBPP, the mean bonus initially falls and then rises after 50 requests, as growth in $\sqrt{q}$ outweighs the decline in u.

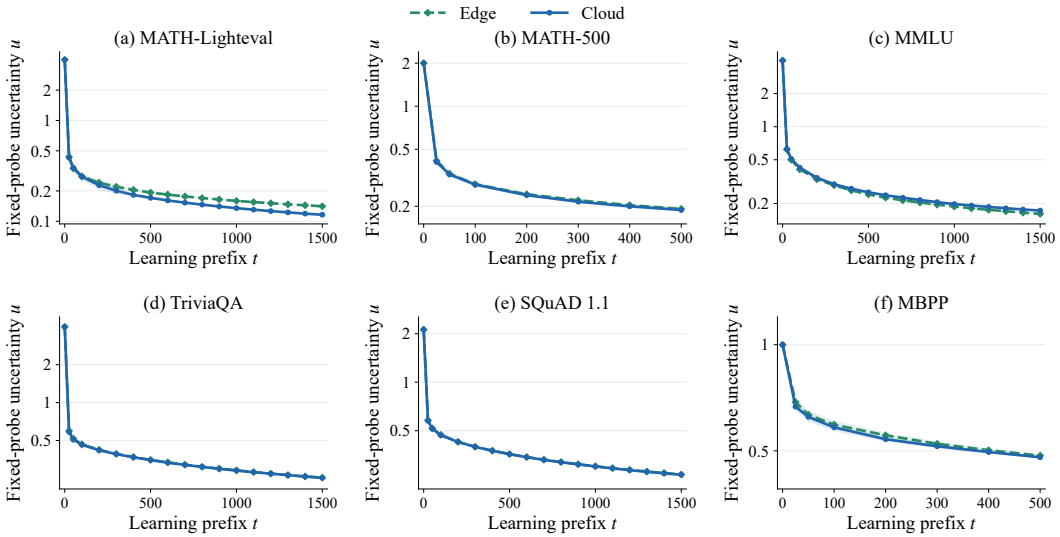


Figure 5: Fixed-probe geometric uncertainty. The vertical axis is logarithmic.

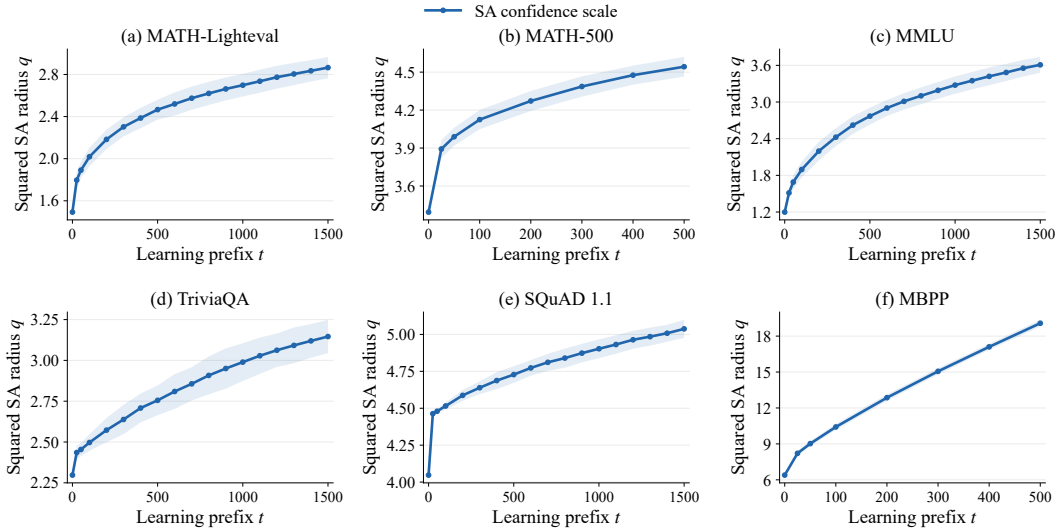


Figure 6: The shared squared SA confidence scale along each task's learning stream.

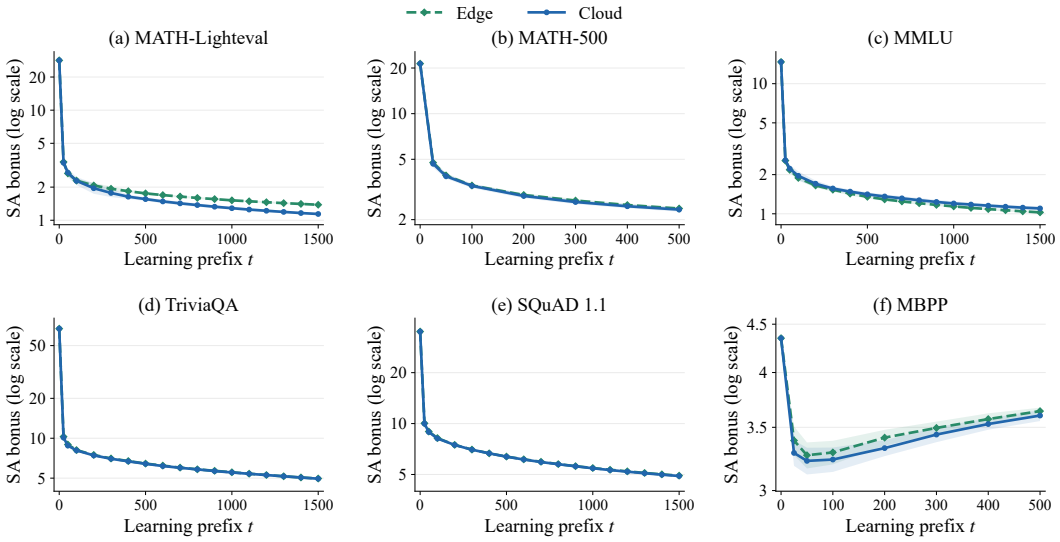


Figure 7: Fixed-probe SA bonuses in utility units on logarithmic vertical axes. Shading shows standard deviations.

F Service Feedback and Controlled Pareto Experiments

F.1 Recorded Cloud Latency and Service Comparison

We replay all 7,000 recorded requests with cloud as the initial route, comparing deadline-only service with takeover under the task-specific thresholds in Table 3. Both use $D_{\max} = 60$ s and zero additional switching overhead. For recorded setup, edge-path, and cloud-path times $(\tau_{\text{setup}}, \tau_e, \tau_c)$, completion times are

$$T_0 = \tau_{\text{setup}} + \tau_c, \qquad T_s = \begin{cases} T_0, & T_0 \le s, \\ s + \tau_{\text{setup}} + \tau_e, & T_0 > s. \end{cases}$$

Takeover repeats setup for a cold edge restart. Delivered delay is $\min\{T_j, D_{\max}\}$, for $j \in \{0, s\}$; quality is the delivered model's task score if $T_j \le D_{\max}$, and zero otherwise.

Table 6 summarizes T_0 before deadline truncation. Across tasks, p99 is 13.2–68.6 times the median. All reported quantiles use linear interpolation.

Table 6: Recorded cloud completion times before deadline truncation. Times are in seconds.

Task	N	Median	p95	p99	p99/Median
MATH-Lighteval	1,500	5.80	87.48	270.19	46.6
MATH-500	500	6.55	78.71	210.49	32.1
MMLU	1,500	4.48	84.62	307.28	68.6
TriviaQA	1,500	2.05	13.04	88.55	43.3
SQuAD 1.1	1,500	1.98	6.78	26.26	13.2
MBPP	500	15.26	170.82	282.45	18.5

Table 7 provides the per-task outcomes for Figure 3(a). Figure 8 shows the full latency tails. For task k, let $q^j_{0.95,k}$ be the p95 capped delay under service j. The mean relative reduction is

$$\frac{100}{6} \sum_{k=1}^{6} \left(1 - \frac{q^s_{0.95,k}}{q^0_{0.95,k}} \right) = 36.46\%.$$

Takeover occurs on 881 requests (12.59%) and eliminates all 366 recorded deadline failures (5.23%).

SQuAD's unchanged p95 follows from the timing constraint and the recorded latency distribution. Its maximum setup-plus-edge time is 3.7462 s, so any uniform d_e covering these records requires

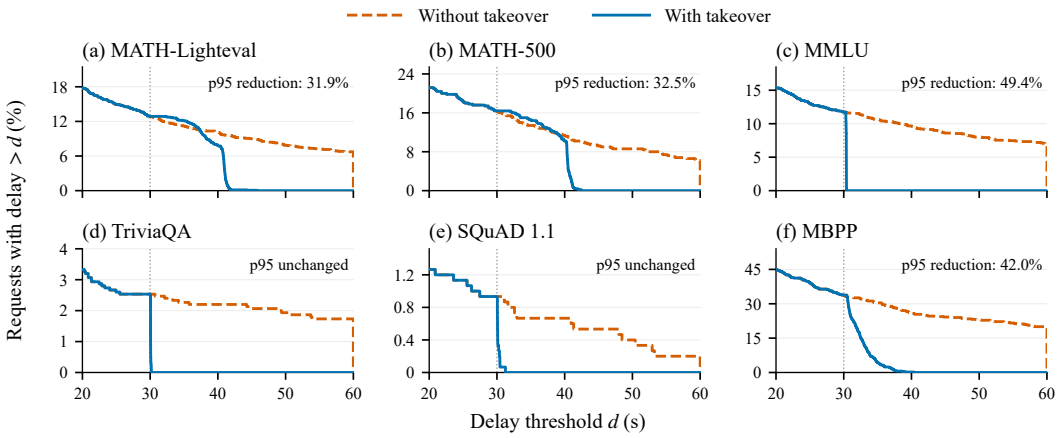


Figure 8: Fixed-cloud latency tails with and without takeover. Triangles mark p95; vertical lines indicate the task-specific takeover time and shared deadline.

Table 7: Fixed-cloud service outcomes without → with takeover.

Task	N	p95 (s)	Reduction (%)	Failures	Takeover (%)
MATH-L	1,500	60.00 → 40.88	31.9	101 → 0	12.87
MATH-500	500	60.00 → 40.48	32.5	31 → 0	16.40
MMLU	1,500	60.00 → 30.38	49.4	105 → 0	11.73
TriviaQA	1,500	13.04 → 4.83	63.0	26 → 0	15.73
SQuAD 1.1	1,500	6.78 → 6.78	0.0	3 → 0	1.67
MBPP	500	60.00 → 34.78	42.0	100 → 0	33.80

$s \geq 4d_e \geq 14.9848$ s and $D_{\max} \geq 2s$ under (1). The two cloud order statistics defining the interpolated p95 are 6.7745 and 6.8512 s, both below this minimum s. Requests completing by s retain their delays, while later requests remain at or above s after takeover and deadline truncation. Thus, p95 stays unchanged throughout this timing range for these records. At the selected threshold, takeover reduces mean delay and removes all three deadline failures.

Cold restarts can prolong requests that cloud would finish shortly after s, explaining the crossing tail curves on math tasks and MBPP. Across all six tasks, waiting-cost savings exceed quality losses (Table 8), giving $\Delta\bar{U} = \Delta\bar{Y} - \beta\Delta\bar{D} > 0$.

Table 8: Quality–delay trade-offs without → with takeover.

Task	β	Quality	Mean delay (s)	$\Delta\bar{U}$
MATH-L	0.08	0.9213 → 0.8687	13.042 → 11.524	0.0688
MATH-500	0.08	0.9300 → 0.8460	14.248 → 12.692	0.0405
MMLU	0.04	0.8833 → 0.8780	11.485 → 8.836	0.0986
TriviaQA	0.17	0.8640 → 0.7726	4.355 → 2.524	0.2157
SQuAD 1.1	0.12	0.9113 → 0.9087	3.099 → 2.786	0.0349
MBPP	0.01	0.5780 → 0.5600	24.883 → 18.213	0.0620

Assumption 1 together with the timing condition (1) implies $D_s \leq 37.5$ s at $s = 30$ s, whereas the two math tasks have p95 above 40 s.Their service gains therefore extend beyond this sufficient fast-edge regime. The controlled model below verifies the theoretical conditions.

F.2 Controlled Pareto Model and Assumptions

This subsection specifies the learning experiments in Figure 3(b) and Appendix F.3. Queries are independent and uniform on $\{e_1, -e_1, e_2, -e_2\} \subset \mathbb{R}^{16}$, where e_1, e_2 are coordinate unit vectors. Let $R(z, z') = \exp(-\|z - z'\|_2^2/(2h^2))$ with $h = 1.8656781657$, and set $d(z) = R(z, e_1) - R(z, -e_1)$.

Edge and cloud qualities are Bernoulli with means $p_e(z) = 0.65 + 0.1d(z)$ and $p_c(z) = 0.85 - 0.1d(z)$. Conditional on the query, raw cloud delay is independent of both qualities and follows Pareto(x, α), with $\alpha \in \{1.25, 1.75\}$ and $x = 12(\alpha - 1)/\alpha$. Each shape has mean delay 12 s and infinite variance. Direct and takeover edge calls take 5 s, with zero additional switching overhead.

We use $\kappa((a, z), (a', z')) = 2\mathbf{1}\{a = a'\}[1 + R(z, z')]$, $\lambda = 1/4$, $\delta = 0.05$, $\beta = 0.08$, $s = 30$ s, and $D_{\max} = 60$ s. Decision and update costs are recorded separately. At each horizon, 1,500 or 10,000 rounds, ten paired seeds per reported shape give 40 trajectories across the two services. The longer runs use fresh seeds; paired services share query, delay, and quality draws. All trajectories for the two reported shapes are included. Results report seed means, with sample standard deviations on regret curves.

Conditional moments. All moments are conditional on z; write $j = 0$ for deadline-only service and $j = s$ for takeover. For $b \geq x$, define

$$p_b = (x/b)^\alpha, \qquad m_b = \frac{\alpha x - x^\alpha b^{1-\alpha}}{\alpha - 1}, \qquad M_{2,b} = \frac{2x^\alpha b^{2-\alpha} - \alpha x^2}{2 - \alpha}.$$

Here m_b and $M_{2,b}$ are the first two moments of $\min\{D_c, b\}$. Cloud-route moments are

$$\begin{aligned}
\mathbb{E}Y_0 &= p_c(1 - p_{D_{\max}}), & \mathbb{E}D_0 &= m_{D_{\max}}, \\
\mathbb{E}D_0^2 &= M_{2,D_{\max}}, & \mathbb{E}[Y_0 D_0] &= p_c(m_{D_{\max}} - D_{\max} p_{D_{\max}}), \\
\mathbb{E}Y_s &= p_c(1 - p_s) + p_e p_s, & \mathbb{E}D_s &= m_s + 5p_s, \\
\mathbb{E}D_s^2 &= M_{2,s} + (10s + 25)p_s, & \mathbb{E}[Y_s D_s] &= p_c(m_s - s p_s) + p_e(s + 5)p_s.
\end{aligned}$$

For direct edge service, $\mathbb{E}Y_e = p_e$ and $D_e = 5$. For either initial route a and service j, the exact utility mean and variance are

$$\begin{aligned}
f_a^j(z) &= \mathbb{E}Y_a^j - \beta \mathbb{E}D_a^j, \\
\mathrm{Var}(U_a^j \mid z) &= \mathbb{E}Y_a^j - 2\beta \mathbb{E}[Y_a^j D_a^j] + \beta^2 \mathbb{E}[(D_a^j)^2] - f_a^j(z)^2.
\end{aligned}$$

The cross moment accounts for dependence between delivered quality and delay.

Theoretical conditions. For each shape, $1 < \alpha < 2$, $4\max\{x, 5\} \leq s \leq D_{\max}/2$, and $v_0 = M_{2,D_{\max}} - m_{D_{\max}}^2 > 25/4$. We use

$$k_s = \frac{s^2}{16}(x/s)^\alpha, \qquad v_s = \max\{25/4, v_0 - k_s\}, \qquad \bar{v}_j = (1/2 + \beta\sqrt{v_j})^2.$$

To verify Assumption 2, write $g^j(a, z) = c_a^j + b_a^j d(z)$, where

$$\begin{aligned}
c_a^j &= \frac{f_a^j(e_2) + \beta D_{\max}}{B}, & b_e^j &= \frac{0.1}{B}, \\
b_c^0 &= -\frac{0.1(1 - p_{D_{\max}})}{B}, & b_c^s &= \frac{0.1(2p_s - 1)}{B}.
\end{aligned}$$

The constant and RBF components give, for both services and both tail shapes,

$$\|g^j\|_{\mathcal{H}}^2 \leq \sum_{a \in \{e,c\}} \left[\frac{(c_a^j)^2}{2} + (b_a^j)^2(1 - e^{-2/h^2}) \right] < 0.77.$$

For this controlled model, the supremum defining Γ_N is restricted to action–query sequences from its four-query support. Set $\rho = e^{-1/h^2}$. The eigenvalues μ_i of one action block are $10 + 4\rho + 2\rho^2$, $2 - 2\rho^2$ (twice), and $2(1 - \rho)^2$. Symmetry and log-determinant concavity yield the common bound

$$\Gamma_N \leq \overline{\Gamma}_N := \max\left\{ 1, \sum_{i=1}^{4} \log\left(1 + \frac{N\mu_i}{8\lambda}\right) \right\}. \tag{F.1}$$

This fractional-count bound is attained when $8 \mid N$ and is $O(\log N)$.

F.3 Learning Regret and Implementation Checks

SA learns only from the selected route's normalized terminal feedback. Using the exact means, the evaluator computes

$$R_n^j = \sum_{i=1}^{n} \left[\max_a f_a^j(z_i) - f_{a_i}^j(z_i) \right], \qquad j \in \{0, s\},$$

relative to each service's own oracle. For each trajectory, let

$$C_N^{\mathrm{SA}} := B \min \left\{ N, 2\sqrt{\frac{N q_{N-1} \mathcal{D}_N}{\log 2}} \right\}$$

denote the observed-loss regret certificate in (C.4).

Table 9 reports the 1,500-round results. At fixed mean cloud delay, takeover improves utility by 0.0078 for $\alpha = 1.75$ and 0.0327 for $\alpha = 1.25$, yielding a larger gain under the heavier-tailed setting.

Table 9: Controlled SA comparison at $N = 1{,}500$. C_N^{SA} is the observed-loss regret certificate.

α	Service	Utility	Quality	Delay (s)	R_N/N	C_N^{SA}/N
1.25	No takeover	0.2281	0.7313	6.29	0.0269	4.9877
1.25	Takeover	0.2607	0.7451	6.05	0.0239	4.7368
1.75	No takeover	0.1531	0.7107	6.97	0.0928	4.8397
1.75	Takeover	0.1609	0.7142	6.92	0.0838	4.6985

Figure 9 shows the separate 10,000-round runs. Average pseudo-regret R_n^j/n generally decreases, with a lower final mean under takeover for both tail shapes. Since $\Gamma_N = O(\log N) = o(\sqrt{N})$, the general service bound (C.9) guarantees vanishing average regret for both services on their joint confidence event. These curves provide numerical support.

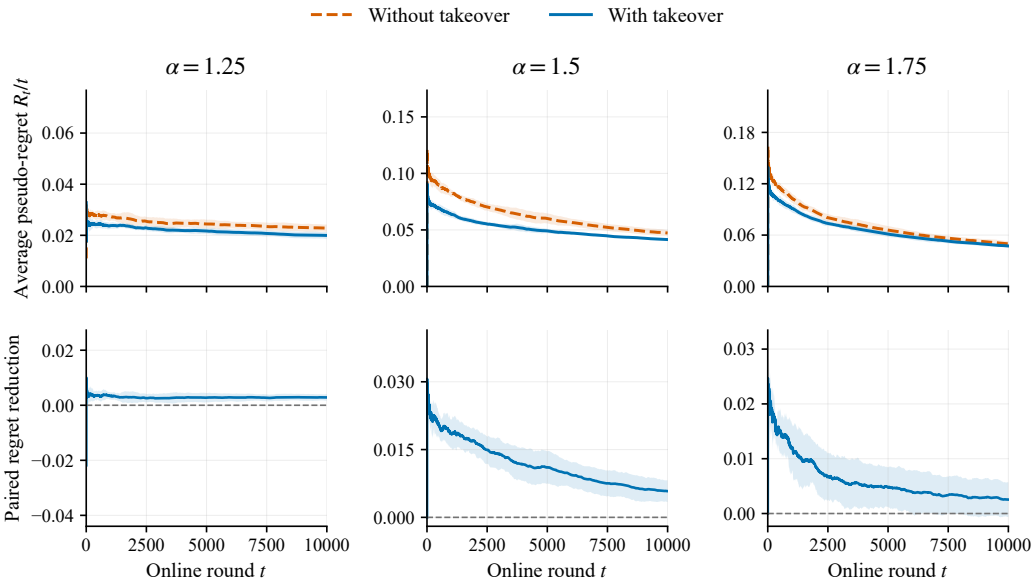


Figure 9: Average pseudo-regret over 10,000 rounds in the controlled Pareto model. Columns: $\alpha = 1.25$ and 1.75. Upper: R_n^j/n. Lower: paired reduction from takeover.

Independent integration checks the moments, and dense kernel calculations check the information bound. The four-context implementation matches the dense learner in four 64-round checks. Across the 80 reported study trajectories, the recorded confidence, regret-bound, and loss-identity checks pass at every evaluated round. Time-uniform coverage follows from Theorem A.1.

G Additional Ablation Results

Matched configurations. We evaluate all four combinations of confidence calibration (SA or a standard-radius control) and service rule (takeover or deadline-only). SA with takeover is our full method; the remaining combinations serve as ablation controls. The standard-radius controls use

$$q_t^{\mathrm{SG}} = \left(\sqrt{\lambda} + \tfrac{1}{2}\sqrt{\log\det(V_t/\lambda) + 2H}\right)^2,$$

with the same H. Without takeover, the selected model runs until completion or the final deadline $D_{\max}$. With takeover, an unfinished cloud request triggers an edge restart at the task-specific threshold in Table 3.

Quality and delay contributions. Table 10 separates the takeover effect within each calibration and the SA-standard difference under each service rule. Every contrast uses paired request orders and $\Delta\bar{U} = \Delta\bar{Y} - \beta\Delta\bar{D}$. Relative utility gains are computed as $100(\bar{U}_{\mathrm{variant}} - \bar{U}_{\mathrm{control}})/|\bar{U}_{\mathrm{control}}|$.

Table 10: Paired ablation effects. Utility, quality, waiting-cost changes, and utility SD are in 10^{-3} units; delay is in seconds.

Dataset	Contrast	$10^3\Delta\bar{U}$	$10^3\Delta\bar{Y}$	$\Delta\bar{D}$	$10^3\beta\Delta\bar{D}$	10^3 Paired SD
MATH-L	Takeover, SA	28.54	-0.31	-0.3607	-28.85	12.9
	Takeover, standard	26.45	-9.96	-0.4551	-36.41	10.7
	SA–standard, takeover	42.49	27.64	-0.1856	-14.85	15.2
	SA–standard, no takeover	40.40	18.00	-0.2800	-22.40	12.0
MATH-500	Takeover, SA	37.63	-6.00	-0.5453	-43.63	31.7
	Takeover, standard	19.14	-25.07	-0.5526	-44.21	42.1
	SA–standard, takeover	33.54	49.87	0.2041	16.33	32.1
	SA–standard, no takeover	15.05	30.80	0.1968	15.75	40.9
MMLU	Takeover, SA	30.67	16.67	-0.3567	-14.00	4.9
	Takeover, standard	35.22	9.11	-0.6652	-26.11	8.8
	SA–standard, takeover	14.20	2.67	-0.2939	-11.53	8.8
	SA–standard, no takeover	18.75	-4.89	-0.6023	-23.64	8.6
TriviaQA	Takeover, SA	120.53	10.47	-0.6559	-110.06	19.1
	Takeover, standard	114.33	-29.92	-0.8597	-144.25	22.9
	SA–standard, takeover	11.24	19.72	0.0505	8.48	8.8
	SA–standard, no takeover	5.04	-20.67	-0.1532	-25.71	23.9
SQuAD 1.1	Takeover, SA	21.94	5.52	-0.1368	-16.42	9.0
	Takeover, standard	15.77	-0.26	-0.1335	-16.02	11.2
	SA–standard, takeover	20.74	21.82	0.0090	1.08	8.3
	SA–standard, no takeover	14.56	16.04	0.0123	1.47	16.2
MBPP	Takeover, SA	28.21	3.47	-2.0620	-24.74	11.5
	Takeover, standard	27.67	-3.87	-2.6281	-31.54	14.7
	SA–standard, takeover	5.89	5.33	-0.0465	-0.56	13.9
	SA–standard, no takeover	5.35	-2.00	-0.6125	-7.35	22.3

Learning trajectories. Figure 10 shows the learning trajectories of all four configurations. On TriviaQA, SA with takeover achieves higher final mean utility than the standard-radius counterpart on 14 of the 15 request orders. The paired differences range from -0.00577 to 0.02367, averaging 0.01124, consistent with the gain reported in Section 6.4.

H Additional Sensitivity Analysis

Each setting starts from an empty history, using five matched request orders and otherwise unchanged task parameters. We vary $\beta/\beta_0 \in \{0.5, 0.8, 0.9, 1, 1.1, 1.2, 1.5\}$ at $s = 30$ s, and $s \in \{15, 20, 25, 30, 35, 40, 45\}$ s at $\beta = \beta_0$, the unrounded main-experiment preference. For the kernel, we vary $h/h_0 \in \{0.25, 0.5, 0.75, 1, 1.5, 2, 4\}$ at $C = C_0$, and $C \in \{1, 1.5, 2, 3, 4, 6, 8\}$

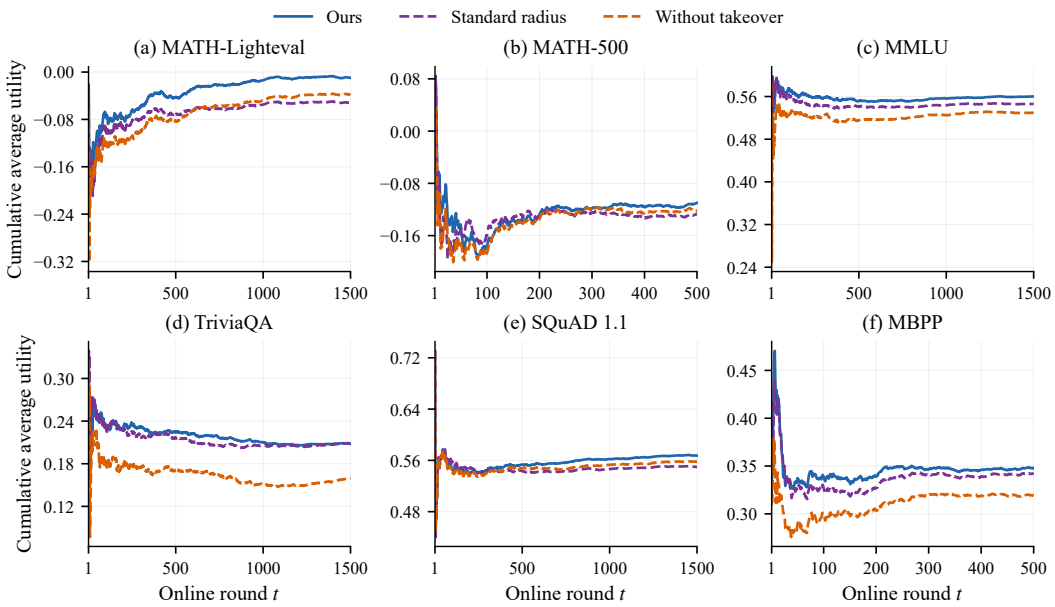


Figure 10: Cumulative average utility for the four confidence–takeover combinations under the shared 60-second deadline.

at $h = h_0$, retaining β_0, $s = 30$ s, and the reported regularization. Here h_0 and C_0 are the reported task-specific defaults. All four parameters are varied separately. The s sweep also covers service settings beyond the sufficient timing range in (1).

Always Edge and Always Cloud use the shared 60-second deadline without takeover, matching the main comparison. We report absolute utility and the gain $\bar{U}_{\mathrm{SA}} - \max\{\bar{U}_{\mathrm{edge}}, \bar{U}_{\mathrm{cloud}}\}$. Only SA changes with s; the two baseline utilities remain fixed. When β changes, all utilities use that same preference. Figures 11, 12, 13, and 14 show all six tasks; Tables 11 and 12 give selected β and s values. Shading denotes one sample standard deviation across request orders, and hollow markers indicate defaults. Lines connect evaluated settings; reported peaks refer to this grid.

How preference and takeover time change utility. Figure 11 shows task-dependent preference effects: the gain over the stronger fixed route peaks at $1.1\beta_0$ on MATH-Lighteval, $0.8\beta_0$ on MMLU, and $0.9\beta_0$ on TriviaQA. Each preference changes the value of waiting and can change which fixed route is stronger. At fixed β_0, earlier takeover improves the tested mean on MATH-500, MMLU, and TriviaQA, whose largest utilities occur at 15 s; MBPP peaks at 30 s (Figure 12). On MATH-500, moving from 30 to 15 s saves 0.0692 in waiting cost at a quality loss of 0.0312, increasing utility by 0.0380.

How bandwidth changes utility. Figure 13 shows different responses across tasks. At $0.25h_0$, mean utility falls by 0.0363 on MATH-Lighteval and 0.0214 on MMLU relative to their defaults. MATH-500 and MMLU peak at h_0, whereas TriviaQA gains 0.0198 at $4h_0$: its quality improvement of 0.0536 exceeds the additional waiting cost of 0.0338. MBPP varies by only 0.0035 across $h/h_0 \in [0.5, 4]$, with little mean benefit from changing the reported bandwidth.

How kernel scale changes utility. Figure 14 shows that larger C does not consistently improve utility. Both math tasks attain their highest tested means at $C = 1$, while MMLU and MBPP peak at their default $C = 2$. On MATH-Lighteval, reducing C from 2 to 1 increases cloud use from 64.1% to 76.1%; the quality gain of 0.0567 exceeds the additional waiting cost of 0.0477, raising utility by 0.0090. Since C changes both prediction and uncertainty estimation, a larger kernel scale need not lead to more cloud calls.

Variation across request orders. On MBPP, the largest gain over the stronger fixed route is 0.0270 at $0.8\beta_0$ (SD 0.0091); the takeover sweep peaks at 30 s with gain 0.0116 (SD 0.0121). TriviaQA's gain at 45 s is 0.0011 with SD 0.0220. Relative to the default kernel, TriviaQA gains 0.0198 at $4h_0$ (paired SD 0.0254), while MBPP gains 0.0006 at $0.5h_0$ (paired SD 0.0168). These variations concern request orders over the same queries. Each β defines a distinct quality–delay preference; s, h, and C are compared at β_0.

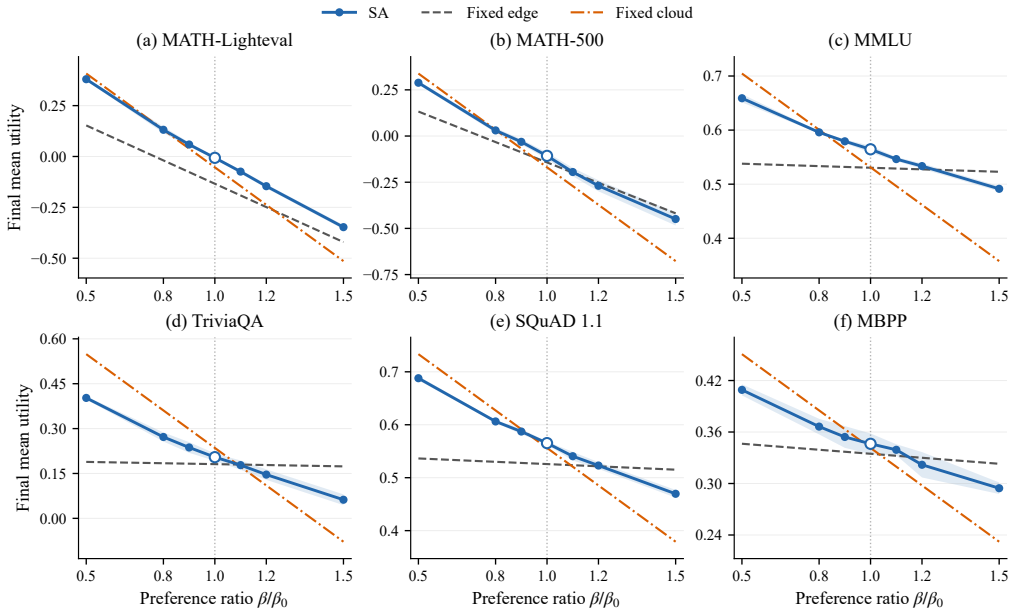


Figure 11: Preference sensitivity: absolute utility of SA and deadline-only fixed routes. Shading denotes standard deviation; hollow markers indicate β_0.

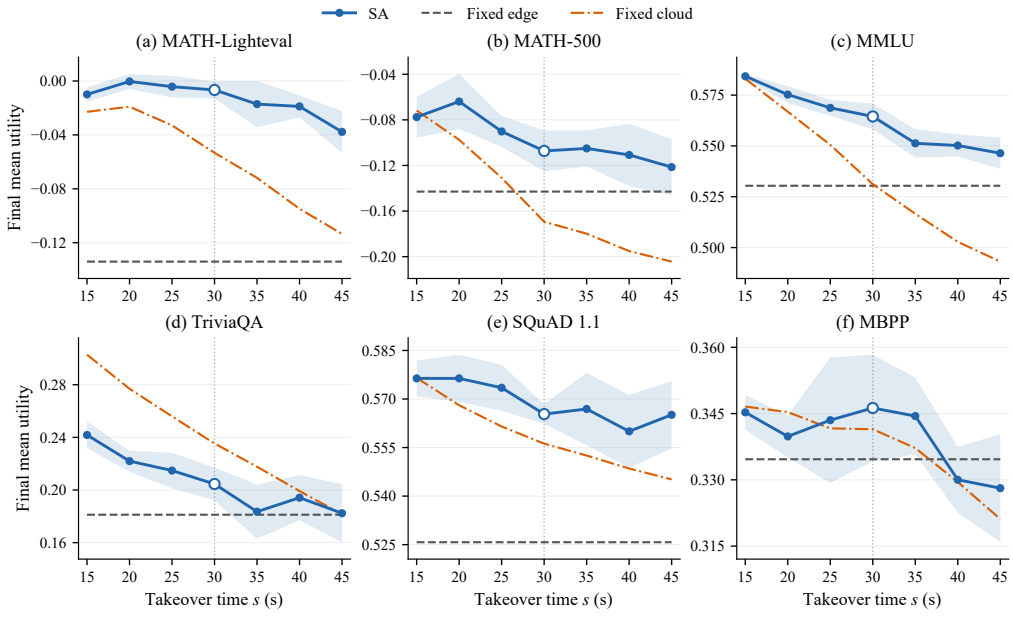


Figure 12: Takeover sensitivity: absolute utility at β_0. Fixed routes retain the deadline without takeover. Shading denotes standard deviation; hollow markers indicate $s = 30$ s.

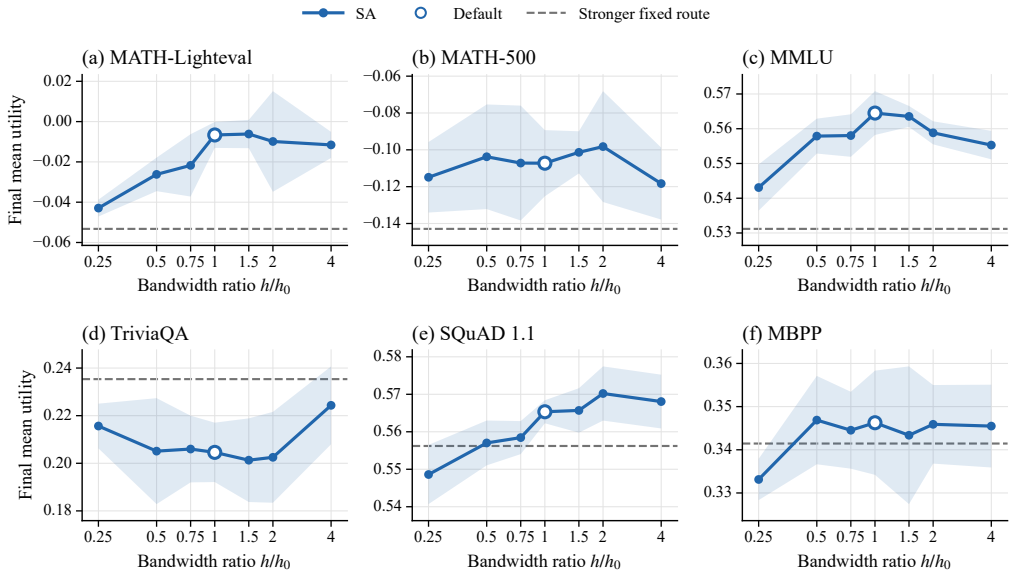


Figure 13: Bandwidth sensitivity at fixed kernel scale. Dashed lines show the stronger deadline-only fixed route.

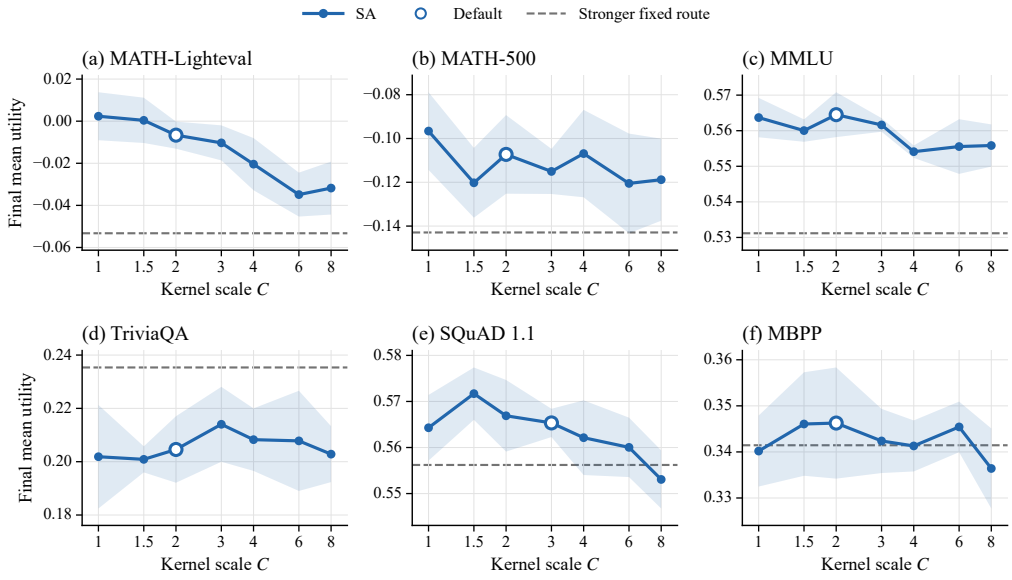


Figure 14: Kernel-scale sensitivity at fixed bandwidth and regularization. Dashed lines show the stronger deadline-only fixed route.

Table 11: Preference sensitivity on selected tasks at $s = 30$ s.

Dataset	β/β_0	SA utility	Edge	Cloud	Gain	SA quality
MATH-L	0.5	0.3799	0.1527	0.3997	-0.0198	0.7579
	0.8	0.1312	-0.0194	0.0866	0.0446	0.7056
	0.9	0.0589	-0.0767	-0.0177	0.0766	0.6941
	1	-0.0066	-0.1340	-0.1220	0.1154	0.6940
	1.1	-0.0744	-0.1914	-0.2264	0.1170	0.6872
	1.2	-0.1460	-0.2487	-0.3307	0.1027	0.6761
	1.5	-0.3473	-0.4207	-0.6437	0.0734	0.6560
MMLU	0.5	0.6589	0.5379	0.6579	0.0010	0.7673
	0.8	0.5959	0.5334	0.5227	0.0625	0.7417
	0.9	0.5793	0.5319	0.4776	0.0474	0.7320
	1	0.5645	0.5304	0.4325	0.0341	0.7249
	1.1	0.5464	0.5289	0.3875	0.0175	0.7128
	1.2	0.5335	0.5274	0.3424	0.0060	0.7087
	1.5	0.4914	0.5229	0.2072	-0.0315	0.6867
TriviaQA	0.5	0.4025	0.1887	0.4986	-0.0961	0.6005
	0.8	0.2719	0.1842	0.2794	-0.0075	0.5418
	0.9	0.2369	0.1827	0.2063	0.0306	0.5342
	1	0.2046	0.1812	0.1333	0.0234	0.5238
	1.1	0.1780	0.1797	0.0602	-0.0017	0.5173
	1.2	0.1463	0.1782	-0.0129	-0.0320	0.5031
	1.5	0.0622	0.1737	-0.2321	-0.1115	0.4784

Table 12: Takeover sensitivity on selected tasks at $\beta = \beta_0$.

Dataset	s (s)	SA utility	Edge	Cloud	Gain	SA quality
MATH-L	15	-0.0099	-0.1340	-0.1220	0.1121	0.6480
	20	-0.0003	-0.1340	-0.1220	0.1217	0.6735
	25	-0.0042	-0.1340	-0.1220	0.1178	0.6847
	30	-0.0066	-0.1340	-0.1220	0.1154	0.6940
	35	-0.0171	-0.1340	-0.1220	0.1049	0.6911
	40	-0.0188	-0.1340	-0.1220	0.1032	0.6935
	45	-0.0378	-0.1340	-0.1220	0.0843	0.6803
MMLU	15	0.5843	0.5304	0.4325	0.0539	0.7256
	20	0.5753	0.5304	0.4325	0.0449	0.7277
	25	0.5687	0.5304	0.4325	0.0383	0.7283
	30	0.5645	0.5304	0.4325	0.0341	0.7249
	35	0.5513	0.5304	0.4325	0.0209	0.7221
	40	0.5502	0.5304	0.4325	0.0198	0.7144
	45	0.5464	0.5304	0.4325	0.0160	0.7188
TriviaQA	15	0.2418	0.1812	0.1333	0.0606	0.5328
	20	0.2220	0.1812	0.1333	0.0408	0.5259
	25	0.2148	0.1812	0.1333	0.0336	0.5248
	30	0.2046	0.1812	0.1333	0.0234	0.5238
	35	0.1835	0.1812	0.1333	0.0023	0.5094
	40	0.1942	0.1812	0.1333	0.0129	0.5169
	45	0.1824	0.1812	0.1333	0.0011	0.5110