QUALITY-AWARE CROSS-MODEL COMPUTATION REUSE

Anonymous authors
Paper under double-blind review

ABSTRACT

An intermediate result computed by one model can be reused by other models to perform their tasks. Existing work mainly focuses on practical execution, leaving a theoretical gap in optimizing reuse decisions. This optimization faces two challenges: *quality uncertainty*, because the effect of reuse on task quality is uncertain across models, and *coupled scheduling*, because tasks need to share the cost of preparing reusable results. These challenges compound each other: quality must be learned online, but the shared preparation structure makes scheduling NP-hard even with known quality, breaking the key assumption in existing methods.

We formulate cross-model computation reuse as an online decision problem and develop the Quality-Aware Reuse Scheduling (QARS) algorithm to address it. For *quality uncertainty*, QARS learns task-dependent reuse quality from selected, possibly delayed feedback and uses optimistic estimates to guide decisions. For *coupled scheduling*, it jointly chooses which results to prepare and which tasks should use them, adapting scheduling accuracy to the remaining quality uncertainty. For the considered problem, our analysis separates learning and optimization error in regret and quantifies the tradeoff between scheduling accuracy and computation. Completing the quality-aware stopping rule yields $\widetilde{O}(\sqrt{T})$ regret while preserving feasibility. Experiments demonstrate the effectiveness of QARS in optimizing cross-model reuse, reducing the combined cost of computation and quality loss by up to 18.0%, and mean regret by 63.9% over the strongest scheduling baseline.

1 INTRODUCTION

Large language models (LLMs) often process the same input for many tasks, such as several questions about one document or successive turns of a conversation. Such repeated processing is common in serving workloads and adds latency and cost (Wang et al., 2025; Ye et al., 2025). Computation reuse avoids it by computing a result once and sharing it across tasks, for example by caching the attention states of a shared prefix (Gim et al., 2024; Zheng et al., 2024; Yao et al., 2025). This reuse typically assumes one model both produces and consumes the result. However, LLM workflows increasingly combine multiple models (Zhang et al., 2025), so one model's result can also serve another.

Cross-model computation reuse targets this case: a *source* model computes a *reusable result*, and a *target* model uses it for multiple tasks (Figure 1). For example, a small source model can summarize or compress a long document (Jiang et al., 2023; Xu et al., 2024), and a larger target model then answers several questions from this result instead of the full document. Similarly, KV caches can be shared or converted across models (Liu et al., 2026; Fu et al., 2026). Such reuse avoids repeated processing, but the target model sees a condensed or converted input, which may reduce task quality.

Existing reuse work mainly focuses on practical execution, leaving a theoretical gap in optimizing reuse decisions. Prior studies develop mechanisms for cross-model reuse (Liu et al., 2026; Fu et al., 2026) and model–cache selection (Lu et al., 2026). Building on these systems, we study the reuse scheduling problem: *which results should be prepared for reuse, and how should tasks be assigned to them when task-dependent quality is unknown and preparation costs are shared?* Optimizing these decisions faces two coupled challenges.

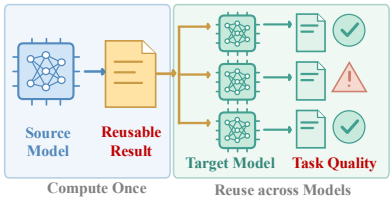


Figure 1: Cross-model computation reuse.

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.

Challenge I: Quality uncertainty arises because the effect of reuse on task quality is uncertain across models. Reuse quality depends on the source and target models, the task, and how the result is reused (Liu et al., 2026; Fu et al., 2026; Heo et al., 2026). In Figure 1, a summary that the target model can read may still lack the evidence needed to answer a question. Quality feedback is available only for executed assignments and may be delayed. The scheduler must therefore learn task-dependent reuse quality from selected feedback while controlling quality loss and execution cost.

Challenge II: Coupled scheduling arises because tasks share the cost of preparing reusable results, coupling their reuse decisions. The scheduler must therefore jointly choose which results to prepare and how to assign tasks to them, accounting for shared preparation costs, task-dependent execution costs, and quality losses.

We formulate cross-model computation reuse as an online decision problem and develop *Quality-Aware Reuse Scheduling* (QARS) to address it. For Challenge I, QARS learns task-dependent reuse quality from selected, possibly delayed feedback and uses optimistic estimates to guide reuse decisions. For Challenge II, its Joint Adaptive Scheduling (JAS) algorithm jointly chooses which results to prepare and how to assign tasks to them. The scheduler uses a computable optimization gap to adjust scheduling accuracy to the remaining quality uncertainty.

Our main contributions are as follows:

- **Problem formulation.** We provide a theoretical framework for cross-model computation reuse. We formulate it as an online decision problem, capturing quality uncertainty and coupled scheduling. We show that the shared preparation structure makes scheduling NP-hard even with known quality, breaking the efficient-oracle assumption in existing methods (Section 3).
- **Joint scheduling with adaptive accuracy.** We develop QARS, which learns reuse quality from selected, possibly delayed feedback and uses JAS for joint preparation and assignment. Unlike existing methods, JAS bounds each feasible schedule's optimization error by a computable gap and stops once this gap matches the remaining quality uncertainty (Section 4).
- **Theoretical guarantees.** We separate learning and optimization error, obtaining $\widetilde{O}(\sqrt{T})$ regret when quality-aware stopping is met at every epoch. Under the component conditions, we bound the scheduling work needed to reach a positive stopping tolerance (Section 5).
- **Empirical validation.** Experiments on four datasets support the analysis, with QARS reducing the combined cost of computation and quality loss by up to 18.0%, and mean regret by 63.9% over the strongest scheduling baseline (Section 6).

2 Related Work

2.1 Computation Reuse

Existing work on computation reuse mainly focuses on practical execution. Within a single model, Prompt Cache and SGLang reuse attention states across prompts (Gim et al., 2024; Zheng et al., 2024), while CacheBlend and KVCOMM extend reuse to document segments and overlapping agent contexts (Yao et al., 2025; Ye et al., 2025). Across models, prompt compression and summarization let a small model shorten the input of a larger one (Jiang et al., 2023; Xu et al., 2024). At the cache level, DroidSpeak selectively recomputes parts of a cache shared across fine-tuned variants, and Cache-to-Cache learns to transform and combine source and target caches (Liu et al., 2026; Fu et al., 2026). These studies show how to produce and use reusable results, but not which results to prepare or which tasks should use them. We address this gap by studying joint result preparation and task assignment under unknown task-dependent quality and shared preparation costs, with guarantees on effectiveness and efficiency.

2.2 Online Learning

Online learning addresses sequential decision-making under uncertainty by learning from observed feedback. Linear bandit methods use optimistic confidence bounds to guide exploration (Abbasi-Yadkori et al., 2011). Contextual linear semi-bandits extend this principle to feasible combinations (Wen et al., 2015; Takemura et al., 2021), while delayed-feedback analyses account for pending

observations (Howson et al., 2023; Zierahn et al., 2025). AESCB lets the index approximation error scale with the confidence widths (Cuvelier et al., 2021). Other methods exploit matroid structure (Tzeng et al., 2024) or reduce oracle calls (Kim et al., 2025). These analyses rely on a subroutine with an a priori guarantee over the action set. In our problem, shared preparation costs make even the optimistic problem NP-hard, so a polynomial-time exact oracle cannot be assumed unless $\mathrm{P} = \mathrm{NP}$. We instead certify each schedule by a computable additive gap to a global lower bound, stop once this gap matches the selected assignments' widths, and bound the work needed to reach it under the component conditions. We provide more detailed mathematical distinctions in Appendix F.1.

2.3 Coupled Scheduling

Coupled scheduling addresses interdependent decisions when tasks share preparation costs or compete for resources. Facility-location models capture shared opening costs and task-level assignment costs (Pasteris et al., 2021). OSTOR uses primal–dual decomposition and adaptive dual descent to schedule reusable queries with approximation guarantees (Cheng et al., 2025). In LLM serving, LMetric combines cache reuse with load balancing (Zhang et al., 2026), RouteLLM learns quality–cost tradeoffs for model selection (Ong et al., 2025), and Unified AI Gateway jointly formulates model, cache and placement decisions under quality and resource constraints (Lu et al., 2026). Our focus is on joint result preparation and task assignment with shared preparation costs and unknown task-dependent quality. The key distinction is that quality is learned from selected, possibly delayed feedback, and computable global gaps adapt scheduling accuracy to the remaining quality uncertainty while preserving resource feasibility.

3 System Model and Problem Formulation

In this section, we present the system model and formulate the optimization problem for cross-model computation reuse. Table A.1 in Appendix A summarizes the key notation.

3.1 System Model

Figure 2 illustrates an LLM serving system supported by model-serving engines such as vLLM and SGLang (Kwon et al., 2023; Zheng et al., 2024), with QARS as its reuse scheduler. The system includes a task manager, source and target LLMs, and a reusable results store. *Source* and *target* denote the roles of computing and using a reusable result. The same model may play both roles. Cross-model reuse requires a compatible representation or a supported conversion (Liu et al., 2026; Fu et al., 2026).

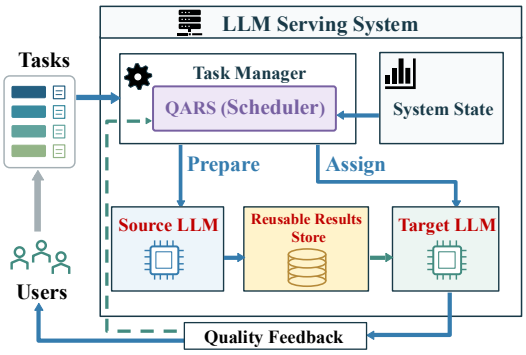


Figure 2: System Overview.

Within the task manager, QARS uses the system state and quality feedback to jointly make scheduling (i.e., preparation and assignment) decisions. Based on these decisions, source models prepare selected results as needed and place them in the reusable results store. Target models execute tasks using the assigned results, and their outputs are evaluated to provide quality feedback, possibly after a delay. We consider *decision epochs* $t \in \{1, \ldots, T\}$, each corresponding to one scheduling step, as modeled below:

Scheduling inputs. At the start of epoch t, the *task set* $\mathcal{R}_t$ contains the tasks to be scheduled, with $|\mathcal{R}_t| \leq M$ for a *workload bound* M. Each *task* $r \in \mathcal{R}_t$ is an admitted unit of work with a designated *target model* m_r^{tgt}. A *reusable result* g is a complete intermediate output for a specified input, computed by its *source model* m_g^{src} under a preparation recipe that specifies its representation and any conversion needed for reuse. Examples include a document summary and a cached KV prefix (Liu et al., 2026). The *result catalogue* $\mathcal{G}_t$ is the finite set of results retained or available for preparation at epoch t.

A *compatible pair* (r, g) means that task r can execute using result g, and the *compatibility set* $\mathcal{E}_t \subseteq \mathcal{R}_t \times \mathcal{G}_t$ collects these pairs.[1] A *context vector* $\boldsymbol{\phi}_{rg,t} \in \mathbb{R}^{D_\phi}$ describes each compatible pair (r, g). It encodes the source–target model identities (i.e., m_g^{src} and m_r^{tgt}), preparation recipe (e.g., the requested summary length) and task context (e.g., document length and question–document overlap). The model pair and preparation recipe together define a *route*. Constructed from model settings and task inputs, the vector is known before scheduling and satisfies $\|\boldsymbol{\phi}_{rg,t}\|_2 \leq 1$.

There are two types of computation costs: *preparation costs* and *execution costs*. The *preparation cost* $F_{g,t} \geq 0$ is the work required to make g available. It is paid once per selected result at epoch t, regardless of how many tasks use it, and is zero for already available results. The *execution cost* $e_{rg,t} \geq 0$ measures the work to execute task r using g, such as reading a summary, or loading cached states and recomputing selected layers. Both costs use the same work units and can be estimated through profiling (Zhong et al., 2024). They are treated as known before scheduling. Native execution incurs its full per-task execution cost with no shared preparation charge.

For P *resource types*, such as GPU compute capacity and memory, the *result reservation* $\boldsymbol{c}_{g,t} \in \mathbb{R}^P$ and the *task reservation* $\boldsymbol{c}_{rg,t} \in \mathbb{R}^P$ specify the nonnegative resources reserved for making result g available and executing task r using g, respectively, with $\bar{\boldsymbol{c}}_t \in \mathbb{R}^P$ denoting the available *capacity*.

The *observed state* s_t collects the scheduling inputs known before the decision: the task set $\mathcal{R}_t$, result catalogue $\mathcal{G}_t$, compatibility set $\mathcal{E}_t$, context vectors $\boldsymbol{\phi}_{rg,t}$, costs $(F_{g,t}, e_{rg,t})$, resource data $(P, \boldsymbol{c}_{g,t}, \boldsymbol{c}_{rg,t}, \bar{\boldsymbol{c}}_t)$, and a feasible fallback schedule.

Scheduling decisions and cost. Given the observed state s_t and the feedback already revealed, the scheduler jointly chooses preparation and assignment decisions $\boldsymbol{a} = (\boldsymbol{x}, \boldsymbol{y})$ at epoch t. The *preparation decision* $y_g = 1$ selects result $g \in \mathcal{G}_t$ for use, preparing it if it is not already available. The *task assignment* $x_{rg} = 1$ assigns task r to result g for a compatible pair $(r, g) \in \mathcal{E}_t$. Let $\ell_{rg,t}$ denote the unknown *expected quality loss* of this assignment conditional on s_t. For a *quality weight* $\omega > 0$, the expected quality-aware cost is

$$J_t(\boldsymbol{a}) = \underbrace{\sum_{g \in \mathcal{G}_t} F_{g,t} y_g}_{\text{preparation cost}} + \underbrace{\sum_{(r,g) \in \mathcal{E}_t} e_{rg,t} x_{rg}}_{\text{execution cost}} + \omega \underbrace{\sum_{(r,g) \in \mathcal{E}_t} \ell_{rg,t} x_{rg}}_{\text{weighted quality loss}}. \tag{1}$$

The quality weight ω converts expected quality loss into work units and balances computation cost and quality loss. Shared preparation costs are charged through $\boldsymbol{y}$, while task-dependent execution costs and quality losses enter through $\boldsymbol{x}$. Appendix H.3 provides evaluation under different ω.

Quality feedback after scheduling. After execution, the *observed quality loss* $\ell^{\mathrm{o}}_{rg,t} \in [0, 1]$ evaluates the output of task r using result g, for example as one minus a normalized task score. Its mean conditional on s_t is $\ell_{rg,t}$. Native execution and same-model reuse may also incur quality loss. Following linear bandit and semi-bandit models (Abbasi-Yadkori et al., 2011; Wen et al., 2015), we model[2] $\ell_{rg,t} = \boldsymbol{\phi}_{rg,t}^\top \boldsymbol{\theta}^\star$ with $\|\boldsymbol{\theta}^\star\|_2 \leq S$, for unknown $\boldsymbol{\theta}^\star$ and a known *parameter-norm bound* S. Feedback is available only for selected assignments and may be delayed. For a known *feedback-delay bound* D, each assignment selected at epoch t yields one observation revealed before epoch $t + D + 1$. Thus, $D = 0$ means that feedback arrives before the next epoch, and at most MD observations are pending before each scheduling decision. We make the following noise assumption (Abbasi-Yadkori et al., 2011; Howson et al., 2023):

Assumption 1 (Feedback noise)**.** Given the information available before each reveal, $\varepsilon = \ell^{\mathrm{o}} - \boldsymbol{\phi}^\top \boldsymbol{\theta}^\star$ is mean-zero and σ-sub-Gaussian for a known *noise scale* σ.

Binary task errors and normalized scores in $[0, 1]$ satisfy the noise assumption with $\sigma = 1/2$ by Hoeffding's lemma, provided the residuals are conditionally mean-zero. For observations sharing a result, the block extension in Appendix B.1 accounts for their dependence with noise scale $\sigma\sqrt{\chi}$, where χ is a *dependence factor*.

[1] Reuse is *cross-model* when $m_g^{\mathrm{src}} \neq m_r^{\mathrm{tgt}}$. This setting also supports native execution (i.e., execution without computation reuse) and same-model reuse.

[2] Generalized linear (Li et al., 2017) and neural (Zhou et al., 2020) bandits offer richer quality models. Extending the confidence and regret analyses to them is future work.

3.2 Problem Formulation

Per-epoch scheduling problem $\mathcal{P}_t$. At epoch t, we minimize the quality-aware cost (1):

$$\min_{\boldsymbol{a}=(\boldsymbol{x},\boldsymbol{y})} \quad J_t(\boldsymbol{a}) \tag{$\mathcal{P}_t$}$$

$$\text{s.t.} \quad x_{rg} \le y_g, \qquad (r,g) \in \mathcal{E}_t, \tag{2-1}$$

$$\sum_{g\in\mathcal{G}_t} \boldsymbol{c}_{g,t} y_g + \sum_{(r,g)\in\mathcal{E}_t} \boldsymbol{c}_{rg,t} x_{rg} \le \bar{\boldsymbol{c}}_t, \tag{2-2}$$

$$\sum_{g:(r,g)\in\mathcal{E}_t} x_{rg} = 1, \qquad r \in \mathcal{R}_t, \tag{2-3}$$

$$x_{rg} \in \{0,1\}\ ((r,g)\in\mathcal{E}_t), \quad y_g \in \{0,1\}\ (g\in\mathcal{G}_t). \tag{2-4}$$

Constraint (2-1) links each assignment to a prepared result, allowing tasks to share its preparation cost. Constraint (2-2) enforces resource capacities componentwise: the first sum accounts for reservations per selected result, and the second for reservations per assignment. Each task is modeled as a single target-model execution, with compatible results serving as alternative reuse options, such as different summaries of the same document. Constraint (2-3) therefore assigns every admitted task to exactly one compatible option, including native execution without reuse.[3] Both preparation and assignment decisions are binary by (2-4).

Long-term scheduling problem $\mathbb{P}$. Across T epochs, we minimize the expected regret:

$$\min_{\pi\in\Pi} \quad \mathbb{E}_\pi[R_T] = \mathbb{E}_\pi\left[\sum_{t=1}^{T}\left[J_t(\boldsymbol{a}_t) - J_t(\boldsymbol{a}_t^\star)\right]\right]. \tag{$\mathbb{P}$}$$

Here, Π denotes the set of online policies that choose $\boldsymbol{a}_t \in \mathcal{A}_t(\mathsf{s}_t)$ using only the current observed state and the feedback already revealed, where $\mathcal{A}_t(\mathsf{s}_t)$ is the feasible set of $\mathcal{P}_t$. Expected quality losses remain unknown at decision time, and the expectation averages over randomness in the online process. The benchmark $\boldsymbol{a}_t^\star$ solves $\mathcal{P}_t$ with the true expected quality losses under the same s_t. Thus, regret evaluates reuse decisions along the observed state sequence rather than comparing policies that induce different retention histories.

Problem $\mathcal{P}_t$ is computationally challenging because shared preparation costs couple task assignments. Problem $\mathbb{P}$ further requires learning unknown quality losses from delayed feedback to guide these decisions. We establish the following complexity result.

Proposition 1 (Computational complexity)**.** *Problem $\mathcal{P}_t$ is NP-hard even with known quality losses.*

Appendix B.2 gives the proof by a Set Cover reduction. Appendix B.3 shows that reassigning tasks together can make reuse worthwhile by sharing preparation costs, even when individual reassignments offer no benefit. This motivates joint scheduling to capture shared savings, with a computable global gap to bound the remaining optimization error. Next, Section 4 presents the joint scheduling algorithm.

4 Quality-Aware Reuse Scheduling

4.1 Algorithm Overview

QARS (Algorithm 1) addresses the online problem $\mathbb{P}$ by combining quality learning with joint scheduling. For Challenge I, it learns task-dependent reuse quality from selected, possibly delayed feedback and constructs confidence bounds $\ell^{\pm}_{rg,t}$ (Lines 4–5, Section 4.2). For Challenge II, Joint Adaptive Scheduling (JAS) uses optimistic quality estimates to jointly choose which results to prepare and how to assign tasks to them (Line 6, Section 4.3). It returns a feasible schedule and a global lower bound. The gap between the schedule's optimistic cost and this bound determines when to stop refinement and bounds the optimization error in regret (Section 5).[4] We detail each part below.

[3] Tasks that jointly use multiple results require modeling their combined quality and extending the scheduling algorithm and analysis. We leave this extension to future work.

[4] We also consider a Thompson-sampling variant of QARS, denoted QARS-TS. It replaces $\ell^{-}_{rg,t}$ with clipped predictions from a single Gaussian parameter draw centered at the ridge estimate, while retaining the same feedback updates, JAS and stopping threshold. The effectiveness of QARS-TS is also evaluated in Section 6.

Algorithm 1: Quality-Aware Reuse Scheduling (QARS)

Input: Confidence parameters $(\lambda_0, S, \sigma, \delta)$, quality weight $\omega > 0$, accuracy factor $\alpha > 0$ and optional deadline.
Output: Feasible decision $\boldsymbol{a}_t$ at each epoch.
Initialize feedback history $\mathcal{H}_0 \leftarrow \varnothing$
for $t = 1, \ldots, T$ **do**
  Receive the observed state s_t: $\mathcal{R}_t, \mathcal{G}_t, \mathcal{E}_t, F_{g,t}, e_{rg,t}, \boldsymbol{c}_{g,t}, \boldsymbol{c}_{rg,t}, \bar{\boldsymbol{c}}_t, \boldsymbol{\phi}_{rg,t}$
  Form $\mathcal{H}_t$ by appending newly revealed $(\tau, r, g, \boldsymbol{\phi}_{rg,\tau}, \ell^{\mathrm{o}}_{rg,\tau})$ to $\mathcal{H}_{t-1}$ in reveal order
  Use the context vectors $\boldsymbol{\phi}_{rg,t}$ to compute $\ell^{\pm}_{rg,t}, \beta_t$ by (3)–(4)
  $\boldsymbol{a}_t \leftarrow \mathrm{JAS}(\mathsf{s}_t, \ell^{\pm}_{rg,t}, \boldsymbol{V}_t, \beta_t; \alpha, \omega)$ (Algorithm 2)
  Execute $\boldsymbol{a}_t$, request quality feedback and output $\boldsymbol{a}_t$

4.2 Quality Learning for Challenge I

QARS estimates task-dependent quality losses from revealed feedback and uses confidence bounds to quantify the remaining uncertainty. The *feedback history* $\mathcal{H}_t$ contains each quality observation revealed before scheduling at epoch t once, in reveal order. Let $\boldsymbol{\phi}_i, \ell^{\mathrm{o}}_i$ be the context vector and observed quality loss of record $i \in \mathcal{H}_t$. With *regularization* $\lambda_0 > 0$, the *observed-context matrix* $\boldsymbol{V}_t$ and the regularized estimator $\widehat{\boldsymbol{\theta}}_t$ are

$$\boldsymbol{V}_t = \lambda_0 \boldsymbol{I} + \sum_{i \in \mathcal{H}_t} \boldsymbol{\phi}_i \boldsymbol{\phi}_i^\top, \qquad \widehat{\boldsymbol{\theta}}_t = \boldsymbol{V}_t^{-1} \sum_{i \in \mathcal{H}_t} \boldsymbol{\phi}_i \ell^{\mathrm{o}}_i. \tag{3}$$

For positive definite $\boldsymbol{B}$, define $\|\boldsymbol{v}\|_{\boldsymbol{B}} = (\boldsymbol{v}^\top \boldsymbol{B} \boldsymbol{v})^{1/2}$. For a *failure probability* $\delta \in (0, 1)$, the *confidence radius* β_t and quality-loss bounds $\ell^{\pm}_{rg,t}$ are

$$\beta_t = \sigma \sqrt{\log \frac{\det \boldsymbol{V}_t}{\det(\lambda_0 \boldsymbol{I})} + 2 \log(1/\delta)} + \sqrt{\lambda_0} S,$$
$$\ell^{\pm}_{rg,t} = \mathrm{clip}_{[0,1]} \left(\boldsymbol{\phi}_{rg,t}^\top \widehat{\boldsymbol{\theta}}_t \pm \beta_t \|\boldsymbol{\phi}_{rg,t}\|_{\boldsymbol{V}_t^{-1}} \right). \tag{4}$$

Clipping keeps the bounds within the quality-loss range $[0, 1]$. JAS uses the lower loss bound $\ell^{-}_{rg,t}$ as an optimistic estimate to guide reuse decisions. Our contribution connects quality learning with joint scheduling: the uncertainty terms $\beta_t \|\boldsymbol{\phi}_{rg,t}\|_{\boldsymbol{V}_t^{-1}}$ for selected assignments set the stopping tolerance for the computable optimization gap. This connection adapts scheduling accuracy to the remaining quality uncertainty. Appendix C.1 establishes the confidence bounds, and Appendix C.2 gives the centered route-wise implementation used in the experiments.

4.3 Joint Scheduling for Challenge II

Using the learned quality estimates, JAS jointly chooses results to prepare and assigns tasks to them (Algorithm 2). Replacing $\ell_{rg,t}$ with $\ell^{\pm}_{rg,t}$ in (1) defines the optimistic cost J_t^- and pessimistic cost J_t^+. JAS improves the best feasible schedule, called the *incumbent*, and tightens a global lower bound. The gap between its optimistic cost and this bound controls the optimization error.

Component structure. A *component* groups tasks with reusable results from the same input, such as questions and alternative summaries of one document. Reuse stays within components, which share result-storage capacity, while task execution resources are allocated separately. Motivated by prompt-module and prefix reuse (Gim et al., 2024; Zheng et al., 2024) and shared KV-cache management (Kwon et al., 2023), we adopt the following assumption.

Assumption 2 (Component structure)**.** The compatibility graph has ν disjoint components $(\mathcal{R}_i, \mathcal{G}_i)$ with $|\mathcal{G}_i| \leq K_{\mathrm{c}}$. There is one scalar capacity, with nonnegative reservations charged only to used results and zero task reservations.

Rounded local scheduling. Under Assumption 2, JAS uses Rounded Local Scheduling (RLS, Algorithm 3), reported as QARS (Bounded). Each covering result subset defines a state by assigning tasks to their cheapest compatible options under optimistic costs, preserving the integral optimum by

Algorithm 2: Joint Adaptive Scheduling (JAS)

Input: $\mathsf{s}_t, \ell^{\pm}_{rg,t}, \boldsymbol{V}_t, \beta_t$; $\alpha, \omega > 0$; optional deadline.
Output: Feasible decision $\boldsymbol{a}^0_t$, with retained global lower bound L^-_t.
Set $\widetilde{e}_{rg} \leftarrow e_{rg} + \omega \ell^-_{rg}$ and form $J^{\pm}$ by (1)
Initialize $\boldsymbol{a}^0 \leftarrow$ feasible fallback
if *the workload satisfies Assumption 2* **then**
  $(\boldsymbol{a}^0, L^-) \leftarrow \mathrm{RLS}(\mathsf{s}_t, \widetilde{e}_{rg}, \boldsymbol{a}^0, \boldsymbol{V}_t, \beta_t; \alpha, \omega)$ (Algorithm 3)
else
  $(\boldsymbol{a}^0, L^-) \leftarrow \mathrm{GLS}(\mathsf{s}_t, \widetilde{e}_{rg}, \boldsymbol{a}^0, \boldsymbol{V}_t, \beta_t; \alpha, \omega)$ (Algorithm 4)
return $\boldsymbol{a}^0_t \leftarrow \boldsymbol{a}^0$; *retain* $L^-_t \leftarrow L^-$

Lemma D.1. Dynamic programming (DP) selects one state per component under the shared capacity, rounding costs above component minima and retaining the minimum exact reservation per rounded total. The feasible schedule and global lower bound differ by at most $\nu B/\kappa$ by Lemma D.2, where B is the initial gap. Higher rounding precision κ improves accuracy at greater computational cost.

General local scheduling. For broader constraints, JAS uses General Local Scheduling (GLS, Algorithm 4), reported as QARS (General), which also handles component workloads. Accepted moves jointly change result selections and task assignments to share preparation costs, preserve feasibility and reduce optimistic cost, as detailed in Appendix E.1. Assignment and capacity prices give a Lagrangian lower bound and screen results that cannot improve the incumbent by Proposition E.1. If needed, linear programming (LP) strengthens the bound and mixed-integer linear programming (MILP) finds feasible candidates. Both procedures retain feasible schedules and valid global lower bounds, but only RLS has the refinement computation bound. The state, confidence bounds, and $\boldsymbol{V}_t$ stay fixed within each call. Algorithm 2 omits epoch subscripts. Appendices E.1, E.2 and D give the procedures, GLS initialization and RLS proofs, respectively.

Quality-aware stopping. Quality-aware stopping limits scheduling work by matching accuracy to quality uncertainty. Let $\boldsymbol{a}^0_t$ be the incumbent, $m^-_t = \min_{\boldsymbol{a} \in \mathcal{A}_t(\mathsf{s}_t)} J^-_t(\boldsymbol{a})$ the optimistic optimum, and $L^-_t \le m^-_t$ the global lower bound. The *actual optimization error* satisfies

$$0 \le \eta_t = J^-_t(\boldsymbol{a}^0_t) - m^-_t \le J^-_t(\boldsymbol{a}^0_t) - L^-_t. \tag{5}$$

The gap includes lower-bound slack $m^-_t - L^-_t$, which may persist for an LP bound even at optimality. For an *accuracy factor* $\alpha > 0$, JAS stops when

$$J^-_t(\boldsymbol{a}^0_t) - L^-_t \le \alpha\omega\beta_t \sum_{(r,g):x^0_{rg,t}=1} \|\boldsymbol{\phi}_{rg,t}\|_{\boldsymbol{V}_t^{-1}}. \tag{6}$$

The threshold expresses selected assignments' quality uncertainty in work units, keeping optimization error at the same scale.

After each completed update, JAS checks this condition and returns the retained feasible schedule and valid bound on success or at the deadline. Missing or unverified outputs leave both unchanged. Section 5.1 accounts for the remaining optimization error if the deadline is reached first.

5 Theoretical Guarantees

In this section, we provide theoretical guarantees on decision quality and computational efficiency.

5.1 Decision Quality

We analyze QARS's decision quality for the per-epoch problem $\mathcal{P}_t$ and the long-term problem $\mathbb{P}$.

Theorem 1 (Decision quality)**.** *Under Assumption 1 and* $\lambda_0 \ge \max\{1, M\}$*,* QARS *satisfies, with probability at least* $1 - \delta$*,*

$$\mathcal{P}_t: \quad J_t(\boldsymbol{a}_t) - J_t(\boldsymbol{a}^\star_t) \le J^+_t(\boldsymbol{a}_t) - L^-_t, \quad t \le T, \tag{7}$$

$$\mathbb{P}: \quad R_T \le \sum_{t \le T} \eta_t + 2\omega\overline{\beta}_T \Xi_T. \tag{8}$$

Appendix F.1 provides the proof and defines the confidence-radius bound $\overline{\beta}_T$ and cumulative context uncertainty bound Ξ_T, including delayed feedback. The per-epoch bound is computable, while the regret bound separates learning and optimization error (Chen et al., 2013; Cuvelier et al., 2021). Completing (6) gives $R_T \leq (2+\alpha)\omega\overline{\beta}_T\Xi_T$, keeping optimization error at the scale of learning uncertainty. If a deadline is reached first, the bound retains the residual optimization error η_t.

Corollary 1 (Regret)**.** *Under the conditions of Theorem 1, suppose* (6) *is satisfied at every nonempty epoch, with probability at least* $1-\delta$*,*

$$R_T = O\Big((2+\alpha)\,\omega\big(\sigma D_\phi \log T + S\sqrt{\lambda_0 D_\phi \log T}\big)\sqrt{MT(1+MD/\lambda_0)}\Big) = \widetilde{O}(\sqrt{T}). \quad (9)$$

Appendix F.2 provides the proof. The corollary shows that matching scheduling accuracy to quality uncertainty achieves sublinear regret under bounded feedback delay, without requiring exact optimization at every epoch.

5.2 Computational Efficiency

We quantify the tradeoff between scheduling accuracy and computational complexity for RLS. Complexity counts scheduling arithmetic operations.

Theorem 2 (Scheduling accuracy and computational complexity)**.** *Under Assumption 2, at each nonempty epoch* t*,* RLS *at a completed integer resolution* $\kappa_t \geq 1$ *returns a feasible schedule and a valid global lower bound satisfying*

$$0 \leq \eta_t \leq J_t^-(\boldsymbol{a}_t) - L_t^- \leq \frac{\nu_t B_t}{\kappa_t}. \quad (10)$$

Its computational complexity is $O(G_t^{\mathrm{g}} + N_t^{\mathrm{r}}(\kappa_t+1))$*. With* $\tau_t > 0$ *and no deadline,* RLS *reaches* (6) *through doubling refinement with computational complexity*

$$O\left(G_t^{\mathrm{g}} + N_t^{\mathrm{r}}\nu_t\left(1+\frac{B_t}{\tau_t}\right)\right). \quad (11)$$

Appendix G.1 provides the proof and parameter definitions. Here ν_t, N_t^{r} and G_t^{g} denote the component count, retained-state count and state-generation work, respectively. The resolution κ_t is the chosen or last completed resolution. A smaller initial gap B_t or larger tolerance lower bound τ_t reduces the complexity bound. Centering makes this bound invariant to common cost offsets within each component. Appendix G.1 compares the finite-instance bound (11) with exact scheduling to identify when computation can be reduced.

6 Experimental Evaluation

6.1 Experimental Settings

Data and workload. We evaluate the proposed framework using workloads built from four datasets—QuALITY, QASPER, MultiRC and RACE (Pang et al., 2022; Dasigi et al., 2021; Khashabi et al., 2018; Lai et al., 2017)—with Qwen3.5-4B and Qwen3.5-9B serving as the source and target models, respectively.[5] Appendix H.1 provides more details.[6]

Baselines and metrics. We compare QARS with four baselines adapted to the reuse problem. UCB-Greedy assigns tasks sequentially using optimistic losses. UCB-LocalSearch improves this schedule through result additions, removals and swaps with joint task reassignment (Arya et al., 2004; Su et al., 2024). Following recent bandit comparisons (Kim & Oh, 2024; Pires et al., 2025), QARS-TS uses Gaussian parameter samples, while Point Estimate uses clipped ridge predictions updated online. Both retain JAS. Additional controls include Frozen Point Estimate (a fixed predictor), NU (native execution), and SMU (QARS restricted to native execution and target-model summaries). We also compare RLS, GLS, Exact DP and MILP for scheduling efficiency. We report task quality $1-\ell^{\mathrm{o}}$, modeled computation cost, quality-aware cost (1), and regret ($\mathbb{P}$), evaluated semi-synthetically with losses sampled from the quality model (Section 3.1) fit to real LLM outputs.

[5] This work primarily focuses on the theoretical analysis of quality-aware reuse scheduling. We leave evaluation across different model families and at larger model scales to future work.

[6] Code is available at `https://anonymous.4open.science/r/QARS-F03F/`.

6.2 Decision Quality

Online updates and exploration. The results show that QARS achieves the lowest quality-aware cost (90.78), 18.0% below Frozen Point, supporting online quality learning and exploration. Point Estimate and QARS-TS remain 10.2% and 10.0% higher, so both online fitting and optimistic exploration help. SMU without source-summary routes is 10.2% higher, and native-only NU is 12.7% higher. Shared preparation lowers its modeled computation cost from 50.1 to 20.5 s/task. Quality-weight experiments are in Appendix H.3.

Coupled scheduling with selected feedback. We compare five same-catalogue methods under selected feedback in Table 1 and Figure 3. QARS achieves the lowest mean regret with low scheduling time, demonstrating effective joint scheduling with selected feedback.

Table 1: Average regret per task.

	QARS	QARS-TS	Point Est.	UCB-LocalSearch	UCB-Greedy
Regret	**0.524**	1.070	2.167	1.450	6.084
Time	**0.245**	0.249	0.987	1.698	0.476

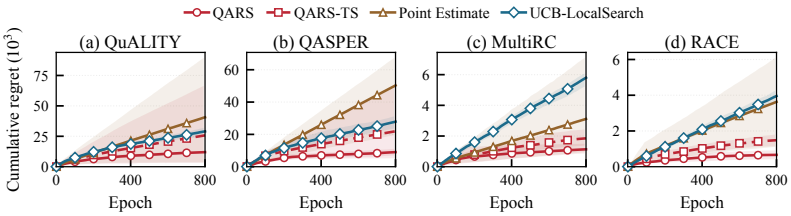


Figure 3: Cumulative regret across epochs.

6.3 Computational Efficiency

Quality-aware stopping. We evaluate quality-aware stopping in Table H.1 and compare scheduling time across deadlines in Figure 4. It reduces mean scheduling time by 58.8% relative to zero tolerance, with comparable regret and quality-aware cost. QARS/QARS-TS have the shortest scheduling times across all tested deadlines. Appendices H.2.1 and H.2.2 provide detailed analyses.

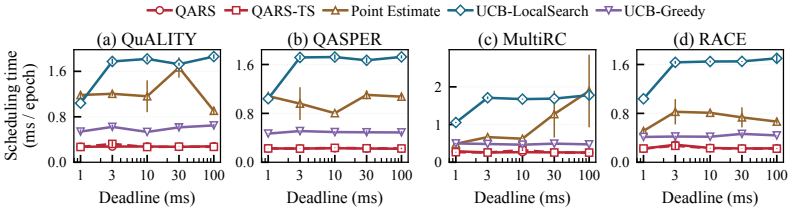


Figure 4: Scheduling time across deadlines for the five primary methods.

Joint adaptive scheduling. We compare Bounded (RLS) and General (GLS) with Exact DP and MILP on identical integer-component instances in Figure 5. Both JAS procedures reduce mean scheduling time (2.72 and 5.37 ms versus 11.55 and 12.42 ms), while all four attain the same mean objective (509.3). Appendices H.2.3 and H.2.4 provide solver and catalogue comparisons.

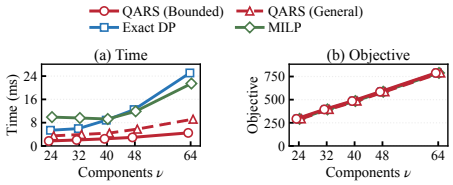


Figure 5: Time and objective versus ν.

7 Conclusion

We study online cross-model computation reuse with shared preparation costs, resource constraints and unknown task-dependent quality. QARS jointly chooses result preparations and task assignments, adapts scheduling accuracy to quality uncertainty, and attains $\widetilde{O}(\sqrt{T})$ regret with bounded scheduling work for component workloads. This work focuses on the theoretical framework. Future work will extend QARS to broader quality models, such as generalized linear and neural bandits, and to richer context features such as embeddings, and will evaluate it in larger-scale serving deployments, including KV-level cross-model reuse, more model pairs and end-to-end latency.

AI USE STATEMENT

The authors used AI tools to aid or polish writing and for retrieval and discovery of related work. AI tools also assisted with creating the initial experimental code framework and checking mathematical arguments. The authors wrote the manuscript and refined and debugged the experimental code. They take responsibility for the final content of this paper, including all claims, proofs, references, and experimental results.

REPRODUCIBILITY STATEMENT

The model assumptions and algorithms are described in Sections 3–4, and proofs of the theoretical results are provided in Appendices B–G. Experimental settings and additional results are provided in Appendix H. Our experimental code is available in an anonymous repository at `https://anonymous.4open.science/r/QARS-F03F/`.

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://papers.nips.cc/paper_files/paper/2011/hash/e1d5be1c7f2f456670de3d53c7b54f4a-Abstract.html`.

Vijay Arya, Naveen Garg, Rohit Khandekar, Adam Meyerson, Kamesh Munagala, and Vinayaka Pandit. Local search heuristics for k-median and facility location problems. *SIAM Journal on Computing*, 33(3):544–562, 2004. doi: 10.1137/S0097539702416402.

Wei Chen, Yajun Wang, and Yang Yuan. Combinatorial multi-armed bandit: General framework and applications. In *Proceedings of the 30th International Conference on Machine Learning (ICML)*, pp. 151–159, 2013.

Jin Cheng, Ningning Ding, John C. S. Lui, and Jianwei Huang. OSTOR: Online scheduling framework for trading continuous queries. In *2025 IEEE 41st International Conference on Data Engineering (ICDE)*, pp. 1895–1909. IEEE, 2025. doi: 10.1109/ICDE65448.2025.00145. URL `https://doi.org/10.1109/ICDE65448.2025.00145`.

Thibaut Cuvelier, Richard Combes, and Eric Gourdin. Statistically efficient, polynomial-time algorithms for combinatorial semi-bandits. *Proceedings of the ACM on Measurement and Analysis of Computing Systems*, 5(1):9:1–9:31, 2021. doi: 10.1145/3447387. URL `https://doi.org/10.1145/3447387`.

Pradeep Dasigi, Kyle Lo, Iz Beltagy, Arman Cohan, Noah A. Smith, and Matt Gardner. A dataset of information-seeking questions and answers anchored in research papers. In *Proceedings of NAACL-HLT*, pp. 4599–4610, 2021. doi: 10.18653/v1/2021.naacl-main.365. URL `https://aclanthology.org/2021.naacl-main.365/`.

Tianyu Fu, Zihan Min, Hanling Zhang, Jichao Yan, Guohao Dai, Wanli Ouyang, and Yu Wang. Cache-to-cache: Direct semantic communication between large language models. In *International Conference on Learning Representations*, 2026. URL `https://proceedings.iclr.cc/paper_files/paper/2026/hash/474ada926b331d78f06d95e8913111cc-Abstract-Conference.html`.

In Gim, Guojun Chen, Seung-seob Lee, Nikhil Sarda, Anurag Khandelwal, and Lin Zhong. Prompt cache: Modular attention reuse for low-latency inference. In *Proceedings of Machine Learning and Systems*, volume 6, 2024. URL `https://proceedings.mlsys.org/paper_files/paper/2024/hash/a66caa1703fe34705a4368c3014c1966-Abstract-Conference.html`.

Taekyung Heo, Rasoul Shafipour, Ritchie Zhao, Maximilian Golub, Mohammad Mahdi Kamani, Ritika Borkar, Makesh Tarun Chandran, Pantea Zardoshti, and Bita Darvish Rouhani. Cross-model KV cache transfer in LLM families: A closed-form linear mapping for prefill reuse. arXiv preprint arXiv:2608.03893, 2026. URL `https://arxiv.org/abs/2608.03893`.

Benjamin Howson, Ciara Pike-Burke, and Sarah Filippi. Delayed feedback in generalised linear bandits revisited. In *Proceedings of the 26th International Conference on Artificial Intelligence and Statistics*, volume 206 of *Proceedings of Machine Learning Research*, pp. 6095–6119, 2023. URL https://proceedings.mlr.press/v206/howson23b.html.

Huiqiang Jiang, Qianhui Wu, Chin-Yew Lin, Yuqing Yang, and Lili Qiu. LLMLingua: Compressing prompts for accelerated inference of large language models. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 13358–13376, 2023.

Daniel Khashabi, Snigdha Chaturvedi, Michael Roth, Shyam Upadhyay, and Dan Roth. Looking beyond the surface: A challenge set for reading comprehension over multiple sentences. In *Proceedings of NAACL-HLT*, pp. 252–262, 2018. doi: 10.18653/v1/N18-1023. URL https://aclanthology.org/N18-1023/.

Jung-hun Kim, Milan Vojnovic, and Min-hwan Oh. Oracle-efficient combinatorial semi-bandits. In D. Belgrave, C. Zhang, H. Lin, R. Pascanu, P. Koniusz, M. Ghassemi, and N. Chen (eds.), *Advances in Neural Information Processing Systems*, volume 38, pp. 24981–25016. Curran Associates, Inc., 2025. doi: 10.52202/085713-0840. URL https://proceedings.neurips.cc/paper_files/paper/2025/file/237196fdf3472b8de08bdd1103bb7b67-Paper-Conference.pdf.

Seok-Jin Kim and Min-hwan Oh. Local anti-concentration class: Logarithmic regret for greedy linear contextual bandit. In *Advances in Neural Information Processing Systems*, volume 37, 2024. doi: 10.52202/079017-2465.

Woosuk Kwon, Zhuohan Li, Siyuan Zhuang, Ying Sheng, Lianmin Zheng, Cody Hao Yu, Joseph E. Gonzalez, Hao Zhang, and Ion Stoica. Efficient memory management for large language model serving with PagedAttention. In *Proceedings of the 29th Symposium on Operating Systems Principles*, pp. 611–626. Association for Computing Machinery, 2023. doi: 10.1145/3600006.3613165.

Guokun Lai, Qizhe Xie, Hanxiao Liu, Yiming Yang, and Eduard Hovy. RACE: Large-scale ReAding comprehension dataset from examinations. In *Proceedings of EMNLP*, pp. 785–794, 2017. doi: 10.18653/v1/D17-1082. URL https://aclanthology.org/D17-1082/.

Lihong Li, Yu Lu, and Dengyong Zhou. Provably optimal algorithms for generalized linear contextual bandits. In *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pp. 2071–2080. PMLR, 2017. URL https://proceedings.mlr.press/v70/li17c.html.

Yuhan Liu, Yuyang Huang, Jiayi Yao, Shaoting Feng, Zhuohan Gu, Kuntai Du, Hanchen Li, Yihua Cheng, Junchen Jiang, Shan Lu, Madan Musuvathi, and Esha Choukse. DroidSpeak: KV cache sharing across fine-tuned model variants. In *23rd USENIX Symposium on Networked Systems Design and Implementation (NSDI 26)*, pp. 319–338. USENIX Association, 2026. URL https://www.usenix.org/conference/nsdi26/presentation/liu-yuhan.

Jiaxun Lu, Xiang Zhang, and Yunfeng Shao. Unified AI gateway: A framework for joint model routing and KV cache management. arXiv preprint arXiv:2609.06940, 2026. URL https://arxiv.org/abs/2609.06940.

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 with preference data. In *International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=8sSqNntaMr.

Richard Yuanzhe Pang, Alicia Parrish, Nitish Joshi, Nikita Nangia, Jason Phang, Angelica Chen, Vishakh Padmakumar, Johnny Ma, Jana Thompson, He He, and Samuel R. Bowman. QuALITY: Question answering with long input texts, yes! In *Proceedings of NAACL-HLT*, pp. 5336–5358, 2022. doi: 10.18653/v1/2022.naacl-main.391. URL https://aclanthology.org/2022.naacl-main.391/.

Stephen Pasteris, Ting He, Fabio Vitale, Shiqiang Wang, and Mark Herbster. Online learning of facility locations. In *Proceedings of the 32nd International Conference on Algorithmic Learning Theory*, volume 132 of *Proceedings of Machine Learning Research*, pp. 1002–1050. PMLR, 2021. URL https://proceedings.mlr.press/v132/pasteris21a.html.

Pedro R. Pires, Gregorio F. Azevedo, Pietro L. Campos, Rafael T. Sereicikas, and Tiago A. Almeida. Exploitation over exploration: Unmasking the bias in linear bandit recommender offline evaluation. In *Proceedings of the Nineteenth ACM Conference on Recommender Systems*. Association for Computing Machinery, 2025. doi: 10.1145/3705328.3748166.

Hongyuan Su, Yu Zheng, Jingtao Ding, Depeng Jin, and Yong Li. Large-scale urban facility location selection with knowledge-informed reinforcement learning. In *Proceedings of the 32nd ACM International Conference on Advances in Geographic Information Systems*. Association for Computing Machinery, 2024. doi: 10.1145/3678717.3691254.

Kei Takemura, Shinji Ito, Daisuke Hatano, Hanna Sumita, Takuro Fukunaga, Naonori Kakimura, and Ken-ichi Kawarabayashi. Near-optimal regret bounds for contextual combinatorial semi-bandits with linear payoff functions. *Proceedings of the AAAI Conference on Artificial Intelligence*, 35(11): 9791–9798, 2021. doi: 10.1609/aaai.v35i11.17177. URL https://ojs.aaai.org/index.php/AAAI/article/view/17177.

Ruo-Chun Tzeng, Naoto Ohsaka, and Kaito Ariu. Matroid semi-bandits in sublinear time. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 48855–48877. PMLR, 2024. URL https://proceedings.mlr.press/v235/tzeng24a.html.

Jiahao Wang, Jinbo Han, Xingda Wei, Sijie Shen, Dingyan Zhang, Chenguang Fang, Rong Chen, Wenyuan Yu, and Haibo Chen. KVCache cache in the wild: Characterizing and optimizing KVCache cache at a large cloud provider. In *2025 USENIX Annual Technical Conference (USENIX ATC 25)*, pp. 465–482. USENIX Association, 2025. URL https://www.usenix.org/conference/atc25/presentation/wang-jiahao.

Zheng Wen, Branislav Kveton, and Azin Ashkan. Efficient learning in large-scale combinatorial semi-bandits. In *Proceedings of the 32nd International Conference on Machine Learning*, volume 37 of *Proceedings of Machine Learning Research*, pp. 1113–1122. PMLR, 2015. URL https://proceedings.mlr.press/v37/wen15.html.

Fangyuan Xu, Weijia Shi, and Eunsol Choi. RECOMP: Improving retrieval-augmented LMs with compression and selective augmentation. In *International Conference on Learning Representations (ICLR)*, 2024.

Jiayi Yao, Hanchen Li, Yuhan Liu, Siddhant Ray, Yihua Cheng, Qizheng Zhang, Kuntai Du, Shan Lu, and Junchen Jiang. CacheBlend: Fast large language model serving for RAG with cached knowledge fusion. In *Proceedings of the Twentieth European Conference on Computer Systems*, EuroSys '25, pp. 94–109. Association for Computing Machinery, 2025. doi: 10.1145/3689031.3696098. URL https://doi.org/10.1145/3689031.3696098.

Hancheng Ye, Zhengqi Gao, Mingyuan Ma, Qinsi Wang, Yuzhe Fu, Ming-Yu Chung, Yueqian Lin, Zhijian Liu, Jianyi Zhang, Danyang Zhuo, and Yiran Chen. KVCOMM: Online cross-context KV-cache communication for efficient LLM-based multi-agent systems. In *Advances in Neural Information Processing Systems*, volume 38, 2025. doi: 10.52202/085713-0605. URL https://proceedings.neurips.cc/paper_files/paper/2025/hash/1a074a28c3a6f2056562d00649ae6416-Abstract-Conference.html.

Dingyan Zhang, Jinbo Han, Kaixi Zhang, Xingda Wei, Sijie Shen, Chenguang Fang, Wenyuan Yu, Jingren Zhou, and Rong Chen. Simple is better: Multiplication may be all you need for LLM request scheduling. In *20th USENIX Symposium on Operating Systems Design and Implementation (OSDI 26)*, pp. 55–73. USENIX Association, 2026. URL https://www.usenix.org/conference/osdi26/presentation/zhang-dingyan.

Guibin Zhang, Luyang Niu, Junfeng Fang, Kun Wang, Lei Bai, and Xiang Wang. Multi-agent architecture search via agentic supernet. In *Proceedings of the 42nd International Conference on*

Machine Learning, volume 267 of *Proceedings of Machine Learning Research*, pp. 75834–75852. PMLR, 2025. URL https://proceedings.mlr.press/v267/zhang25bi.html.

Lianmin Zheng, Liangsheng Yin, Zhiqiang Xie, Chuyue Sun, Jeff Huang, Cody Hao Yu, Shiyi Cao, Christos Kozyrakis, Ion Stoica, Joseph E. Gonzalez, Clark Barrett, and Ying Sheng. SGLang: Efficient execution of structured language model programs. In *Advances in Neural Information Processing Systems*, volume 37, 2024. doi: 10.52202/079017-2000. URL https://proceedings.neurips.cc/paper_files/paper/2024/hash/724be4472168f31ba1c9ac630f15dec8-Abstract-Conference.html.

Yinmin Zhong, Shengyu Liu, Junda Chen, Jianbo Hu, Yibo Zhu, Xuanzhe Liu, Xin Jin, and Hao Zhang. DistServe: Disaggregating prefill and decoding for goodput-optimized large language model serving. In *18th USENIX Symposium on Operating Systems Design and Implementation (OSDI 24)*, pp. 193–210. USENIX Association, 2024. URL https://www.usenix.org/conference/osdi24/presentation/zhong-yinmin.

Dongruo Zhou, Lihong Li, and Quanquan Gu. Neural contextual bandits with UCB-based exploration. In *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pp. 11492–11502. PMLR, 2020. URL https://proceedings.mlr.press/v119/zhou20a.html.

Lukas Zierahn, Dirk van der Hoeven, Tal Lancewicki, Aviv Rosenberg, and Nicolò Cesa-Bianchi. A unified analysis of nonstochastic delayed feedback for combinatorial semi-bandits, linear bandits, and MDPs. *Journal of Machine Learning Research*, 26(104):1–60, 2025. URL https://jmlr.org/papers/v26/24-0496.html.

A NOTATION

Table A.1 summarizes the notation.

Table A.1: Notation.

Symbol	Meaning
$t,\ T$	Decision epoch and horizon.
$r,\ g,\ m_r^{\mathrm{tgt}},\ m_g^{\mathrm{src}}$	Task, reusable result, target model, and source model.
$\mathcal{R}_t,\ \mathcal{G}_t,\ \mathcal{E}_t$	Task set, result catalogue, and compatibility set.
$\mathsf{s}_t,\ \mathcal{A}_t(\mathsf{s}_t)$	Observed state and its feasible set.
$\boldsymbol{a}=(\boldsymbol{x},\boldsymbol{y}),\ \boldsymbol{a}_t^0,\ \boldsymbol{a}_t^\star$	Decision, incumbent, and per-epoch benchmark. x_{rg} assigns task r to result g, and y_g selects result g.
$F_{g,t},\ e_{rg,t},\ \omega$	Preparation cost, execution cost, and quality weight.
$J_t(\boldsymbol{a}),\ J_t^{\pm}(\boldsymbol{a})$	Expected quality-aware cost, and the optimistic and pessimistic costs J_t^- and J_t^+.
$R_T,\ \Pi$	Cumulative regret along the observed states, and the set of online policies.
$P,\ \boldsymbol{c}_{g,t},\ \boldsymbol{c}_{rg,t},\ \bar{\boldsymbol{c}}_t$	Number of resource types, result reservation, task reservation, and capacity. The scalar capacity is $c^{\max}$.
$\ell^{\mathrm{o}}_{rg,t},\ \ell_{rg,t},\ \ell^{\pm}_{rg,t}$	Observed quality loss, expected quality loss, and confidence bounds.
$\boldsymbol{\phi}_{rg,t},\ \boldsymbol{\theta}^\star,\ D_\phi,\ S$	Context vector, unknown parameter, context dimension, and parameter-norm bound, with $\ell_{rg,t}=\boldsymbol{\phi}_{rg,t}^\top\boldsymbol{\theta}^\star$.
$M,\ D,\ \chi$	Workload bound, feedback-delay bound, and dependence factor.
$\mathcal{H}_t,\ \boldsymbol{V}_t,\ \widehat{\boldsymbol{\theta}}_t$	Feedback history, observed-context matrix, and regularized estimator.
$\lambda_0,\ \sigma,\ \delta,\ \beta_t$	Regularization, noise scale, failure probability, and confidence radius.
$\alpha,\ \tau_t$	Accuracy factor and tolerance lower bound.
$\nu_t,\ (\mathcal{R}_i,\mathcal{G}_i),\ K_{\mathrm{c}},\ M_{\mathrm{c}}$	Component count, component task and result sets, and maximum numbers of results and tasks per component. Within one epoch, ν_t is also written ν.
$\mathcal{Q}_i,\ \mathcal{U},\ B_t,\ \kappa_t$	Retained component states, a covering result subset, initial gap, and rounding resolution.
$N_t^{\mathrm{r}},\ G_t^{\mathrm{g}}$	Retained-state count and state-generation work.
$\Lambda_T,\ \overline{\beta}_T,\ H_T,\ \Xi_T$	Logarithmic potential, confidence-radius bound, directional delay term, and cumulative context uncertainty bound.

B SYSTEM MODEL AND PROBLEM FORMULATION

B.1 DEPENDENT FEEDBACK ANALYSIS

Observations sharing a reusable result can be dependent. Blocks of jointly revealed feedback account for this dependence while preserving the linear quality model and regularized estimator.

Feedback blocks and noise conditions. Index the blocks in reveal order by $b=1,2,\ldots$. Block b contains $n_b \le N_{\max}^{\mathrm{blk}}$ observed quality losses from selected assignments, $\boldsymbol{\ell}_b^{\mathrm{o}} \in [0,1]^{n_b}$, with one context vector per row in $\boldsymbol{X}_b \in \mathbb{R}^{n_b\times D_\phi}$. Its membership, size and context matrix are measurable with respect to $\mathcal{F}_{b-1}$, the information available before the block is revealed. Each selected assignment contributes one observation to exactly one block, whose observations are revealed together. The feedback-delay and pending-observation bounds still apply. If batching increases either bound, the corresponding constants must be updated.

The same parameter determines the conditional means, $\mathbb{E}[\boldsymbol{\ell}_b^{\mathrm{o}} \mid \mathcal{F}_{b-1}] = \boldsymbol{X}_b\boldsymbol{\theta}^\star$, with $\boldsymbol{\epsilon}_b = \boldsymbol{\ell}_b^{\mathrm{o}} - \boldsymbol{X}_b\boldsymbol{\theta}^\star$. For a noise scale $\sigma>0$ and a dependence factor $\chi \ge 1$, every $\boldsymbol{v}\in\mathbb{R}^{n_b}$ satisfies

$$\mathbb{E}\big[\exp(\boldsymbol{v}^\top\boldsymbol{\epsilon}_b) \mid \mathcal{F}_{b-1}\big] \le \exp\left(\frac{\sigma^2\chi}{2}\|\boldsymbol{v}\|_2^2\right). \tag{12}$$

This conditional moment-generating-function (MGF) bound is stronger than a covariance bound. It allows dependence within blocks and a random reveal order, but need not hold when the next block is selected using its unrevealed outcomes.

Let $n_{\mathrm{blk}}(t)$ be the number of blocks revealed before epoch t. The regularized estimator is

$$\boldsymbol{V}_t = \lambda_0 \boldsymbol{I} + \sum_{b=1}^{n_{\mathrm{blk}}(t)} \boldsymbol{X}_b^\top \boldsymbol{X}_b, \qquad \widehat{\boldsymbol{\theta}}_t = \boldsymbol{V}_t^{-1} \sum_{b=1}^{n_{\mathrm{blk}}(t)} \boldsymbol{X}_b^\top \boldsymbol{\ell}_b^{\mathrm{o}}. \tag{13}$$

Each revealed observation contributes once to the estimator, which excludes pending observations. With Λ_T and Ξ_T defined in Appendix F.1, the confidence radius and its uniform upper bound are

$$\begin{aligned} \beta_t^{(\chi)} &= \sigma\sqrt{\chi}\sqrt{\log\frac{\det \boldsymbol{V}_t}{\det(\lambda_0 \boldsymbol{I})} + 2\log(1/\delta)} + \sqrt{\lambda_0} S, \\ \overline{\beta}_T^{(\chi)} &= \sigma\sqrt{\chi}\sqrt{\Lambda_T + 2\log(1/\delta)} + \sqrt{\lambda_0} S. \end{aligned} \tag{14}$$

Predictability, conditional means and the MGF bound replace Assumption 1. The linear quality model in Section 3.1 is unchanged. Confidence requires $\lambda_0 > 0$, and Theorem 1 requires $\lambda_0 \geq \max\{1, M\}$.

Proposition B.1 (Block feedback)**.** *Under these conditions, the estimator* (13) *satisfies, with probability at least* $1-\delta$*, simultaneously for every epoch* t*,*

$$\|\widehat{\boldsymbol{\theta}}_t - \boldsymbol{\theta}^\star\|_{\boldsymbol{V}_t} \leq \beta_t^{(\chi)}. \tag{15}$$

The dependence factor scales only the noise term in the confidence radius, leaving the estimator unchanged. On the same event, the confidence bounds yield the regret bound

$$R_T \leq \sum_{t=1}^{T} \eta_t + 2\omega \overline{\beta}_T^{(\chi)} \Xi_T. \tag{16}$$

If every nonempty epoch satisfies (6) with $\beta_t^{(\chi)}$ in place of β_t,

$$R_T \leq (2+\alpha)\omega \overline{\beta}_T^{(\chi)} \Xi_T. \tag{17}$$

Proof. The effective noise scale is $\sigma_0 = \sigma\sqrt{\chi} > 0$. Let $\boldsymbol{A}_n = \sum_{b \leq n} \boldsymbol{X}_b^\top \boldsymbol{X}_b$ and $\boldsymbol{m}_n = \sum_{b \leq n} \boldsymbol{X}_b^\top \boldsymbol{\epsilon}_b$. Predictability of $\boldsymbol{X}_b$ allows $\boldsymbol{v} = \boldsymbol{X}_b \boldsymbol{u}$ in (12), so

$$z_n^{\exp}(\boldsymbol{u}) = \exp\left(\boldsymbol{u}^\top \boldsymbol{m}_n - \frac{\sigma_0^2}{2} \boldsymbol{u}^\top \boldsymbol{A}_n \boldsymbol{u}\right)$$

is a nonnegative supermartingale starting at one. Integrate this process against the Gaussian density $\boldsymbol{u} \sim N(\boldsymbol{0}, (\sigma_0^2 \lambda_0)^{-1} \boldsymbol{I})$, fixed before any feedback is revealed. Conditional Tonelli's theorem preserves the supermartingale property, and Gaussian integration gives the mixture

$$z_n^{\mathrm{mix}} = \left(\frac{\det(\lambda_0 \boldsymbol{I})}{\det(\lambda_0 \boldsymbol{I} + \boldsymbol{A}_n)}\right)^{1/2} \exp\left(\frac{\|\boldsymbol{m}_n\|^2_{(\lambda_0 \boldsymbol{I} + \boldsymbol{A}_n)^{-1}}}{2\sigma_0^2}\right).$$

Ville's inequality then gives, with probability at least $1-\delta$, for every n,

$$\|\boldsymbol{m}_n\|_{(\lambda_0 \boldsymbol{I} + \boldsymbol{A}_n)^{-1}} \leq \sigma_0 \sqrt{\log\frac{\det(\lambda_0 \boldsymbol{I} + \boldsymbol{A}_n)}{\det(\lambda_0 \boldsymbol{I})} + 2\log(1/\delta)}.$$

At epoch t, take $n = n_{\mathrm{blk}}(t)$. Then $\widehat{\boldsymbol{\theta}}_t - \boldsymbol{\theta}^\star = \boldsymbol{V}_t^{-1}(\boldsymbol{m}_{n_{\mathrm{blk}}(t)} - \lambda_0 \boldsymbol{\theta}^\star)$. Since $\boldsymbol{V}_t \succeq \lambda_0 \boldsymbol{I}$, the triangle inequality bounds the regularization contribution by $\sqrt{\lambda_0} S$, proving (15).

Clipping preserves coverage of the expected quality loss. Appendix F.1 uses only the selected contexts and reveal indicators, so the new radius yields (16) and (17). □

Choosing the dependence factor. Suppose each observation lies in $[0,1]$ and has its stated conditional mean. The conditional range of $\boldsymbol{v}^\top \boldsymbol{\ell}_b^{\mathrm{o}}$ then has length at most $\|\boldsymbol{v}\|_1$, and conditional Hoeffding's lemma with Cauchy–Schwarz gives

$$\mathbb{E}\left[e^{\boldsymbol{v}^\top \boldsymbol{\epsilon}_b} \mid \mathcal{F}_{b-1}\right] \leq \exp\left(\frac{\|\boldsymbol{v}\|_1^2}{8}\right) \leq \exp\left(\frac{N_{\max}^{\mathrm{blk}} \|\boldsymbol{v}\|_2^2}{8}\right). \tag{18}$$

Thus $\sigma = 1/2$ and $\chi = N_{\max}^{\mathrm{blk}}$ satisfy (12) under arbitrary dependence within a block. Conditional independence given $\mathcal{F}_{b-1}$ allows $\chi = 1$. An intermediate factor requires a conditional MGF bound, which sample correlation alone cannot establish.

B.2 Proof of Computational Complexity

Proof of Proposition 1. We reduce Set Cover with N_{set} sets to $\mathcal{P}_t$. Given a universe $\{e_1, \dots, e_{N_{\mathrm{el}}}\}$ and sets $\mathcal{S}_1, \dots, \mathcal{S}_{N_{\mathrm{set}}}$ whose union is the universe, construct one epoch with a task r_j per element and a result g_i per set. Set $F_{g_i} = 1$, $e_{r_j g_i} = 0$, $\ell_{r_j g_i} = 0$ and $\boldsymbol{c}_{g_i} = \boldsymbol{c}_{r_j g_i} = \mathbf{0}$, and let $(r_j, g_i) \in \mathcal{E}$ exactly when $e_j \in \mathcal{S}_i$. Choose $\bar{\boldsymbol{c}}$ large enough that (2-2) never binds; activating all set results is a feasible fallback because the sets cover the universe. Add a native option g_0 compatible with every task, with $F_{g_0} = 0$, $e_{r_j g_0} = N_{\mathrm{set}} + 1$, zero quality loss and zero reservations. Any schedule using it costs more than a schedule induced by all sets, of cost at most N_{set}, so no optimum uses native execution.

A feasible schedule without native assignments consists of the selected set $\mathcal{U} = \{i : y_{g_i} = 1\}$ together with an assignment of each task to a selected compatible result. By (2-3) and (2-1) such an assignment exists if and only if $\mathcal{U}$ covers the universe, and by construction $J(\boldsymbol{a}) = |\mathcal{U}|$. The optimum of $\mathcal{P}_t$ therefore equals the minimum cover size. Since Set Cover is NP-hard, so is $\mathcal{P}_t$. □

B.3 Reassigning Tasks Together

The following instance illustrates the joint savings discussed in Section 3.2. Expected quality losses are zero and capacity is unconstrained, so costs reflect only preparation and execution. There are $n \geq 2$ tasks. Each native option has preparation cost 0 and execution cost 1, and one reusable result has preparation cost $3/2$ and execution cost 0. The all-native schedule costs n. Reassigning one task to the reusable result raises the cost by $1/2$, so this individual reassignment offers no benefit. Reassigning all n tasks together costs $3/2 < n$, making reuse worthwhile by sharing the preparation cost.

C Quality Learning

C.1 Simultaneous Confidence Bounds

The regularized estimator uses only revealed quality feedback from selected assignments. Index these observations in reveal order by $i = 1, 2, \dots$, with context vector $\boldsymbol{\phi}_i$ and observed quality loss ℓ_i^{o}. Let $\mathcal{F}_{i-1}$ be the information available before ℓ_i^{o} is revealed, including $\boldsymbol{\phi}_i$. By Assumption 1, the residual $\varepsilon_i = \ell_i^{\mathrm{o}} - \boldsymbol{\phi}_i^\top \boldsymbol{\theta}^\star$ satisfies

$$\mathbb{E}[\varepsilon_i \mid \mathcal{F}_{i-1}] = 0, \qquad \mathbb{E}[e^{u\varepsilon_i} \mid \mathcal{F}_{i-1}] \leq e^{\sigma^2 u^2/2} \quad (u \in \mathbb{R}).$$

These conditions are imposed in reveal order. A feedback-delay bound alone does not ensure them when that order depends on unrevealed quality losses.

After n observations have been revealed, the cumulative noise vector and observed-context matrix are

$$\boldsymbol{m}_n = \sum_{i=1}^{n} \boldsymbol{\phi}_i \varepsilon_i, \qquad \boldsymbol{V}^{(n)} = \lambda_0 \boldsymbol{I} + \sum_{i=1}^{n} \boldsymbol{\phi}_i \boldsymbol{\phi}_i^\top.$$

For any $\lambda_0 > 0$, the uniform self-normalized inequality of Abbasi-Yadkori et al. (2011) gives, with probability at least $1 - \delta$,

$$\|\boldsymbol{m}_n\|_{(\boldsymbol{V}^{(n)})^{-1}} \leq \sigma \sqrt{\log \frac{\det \boldsymbol{V}^{(n)}}{\det(\lambda_0 \boldsymbol{I})} + 2\log(1/\delta)} \quad \text{for every } n \geq 0. \tag{19}$$

At epoch t, $n_{\mathrm{o}}(t) = |\mathcal{H}_t|$ and $\boldsymbol{V}_t = \boldsymbol{V}^{(n_{\mathrm{o}}(t))}$. Substituting into (3) gives

$$\widehat{\boldsymbol{\theta}}_t - \boldsymbol{\theta}^\star = \boldsymbol{V}_t^{-1}\big(\boldsymbol{m}_{n_{\mathrm{o}}(t)} - \lambda_0 \boldsymbol{\theta}^\star\big).$$

Since $\boldsymbol{V}_t \succeq \lambda_0 \boldsymbol{I}$, the triangle inequality yields

$$\begin{aligned} \|\widehat{\boldsymbol{\theta}}_t - \boldsymbol{\theta}^\star\|_{\boldsymbol{V}_t} &\leq \|\boldsymbol{m}_{n_{\mathrm{o}}(t)}\|_{\boldsymbol{V}_t^{-1}} + \lambda_0 \|\boldsymbol{\theta}^\star\|_{\boldsymbol{V}_t^{-1}} \\ &\leq \sigma \sqrt{\log \frac{\det \boldsymbol{V}_t}{\det(\lambda_0 \boldsymbol{I})} + 2\log(1/\delta)} + \sqrt{\lambda_0} S = \beta_t. \end{aligned}$$

On this event, denoted $\mathcal{E}_{\mathrm{conf}}$, weighted Cauchy–Schwarz gives, for every compatible pair,

$$\big|\boldsymbol{\phi}_{rg,t}^\top (\widehat{\boldsymbol{\theta}}_t - \boldsymbol{\theta}^\star)\big| \leq \beta_t \|\boldsymbol{\phi}_{rg,t}\|_{\boldsymbol{V}_t^{-1}}.$$

Since $\ell_{rg,t} \in [0,1]$, clipping to this range preserves coverage of the expected quality loss:

$$\|\widehat{\boldsymbol{\theta}}_t - \boldsymbol{\theta}^\star\|_{\boldsymbol{V}_t} \le \beta_t, \qquad \ell^-_{rg,t} \le \ell_{rg,t} \le \ell^+_{rg,t}, \qquad 0 \le \ell_{rg,t} - \ell^-_{rg,t} \le 2\beta_t \|\boldsymbol{\phi}_{rg,t}\|_{\boldsymbol{V}_t^{-1}}. \tag{20}$$

Before clipping, the lower bound lies at most twice the radius below the conditional mean, and clipping at zero can only reduce this difference. The event holds for every n and hence every epoch, without a further union bound. Confidence requires only $\lambda_0 > 0$. Theorem 1 uses $\lambda_0 \ge \max\{1, M\}$ to bound cumulative uncertainty over selected assignments.

C.2 Centered Confidence Bounds for Routes

Let N_{rt} be the number of routes, with $D_\phi = 2N_{\mathrm{rt}}$ and two coordinates per route. A compatible pair's context has two route coordinates $\boldsymbol{\psi} = (1-\xi, \xi)$ for $\xi \in [0,1]$, with all other coordinates zero. The primary experiments use four routes, so $N_{\mathrm{rt}} = 4$. The additional endpoint assumption $\boldsymbol{\theta}^\star_k \in [0,1]^2$, with known center $\boldsymbol{\theta}^0_k = (1/2, 1/2)$, gives $\|\boldsymbol{\theta}^\star_k - \boldsymbol{\theta}^0_k\|_2 \le 1/\sqrt{2}$. Let $\mathcal{H}_{k,t}$ contain the records in $\mathcal{H}_t$ on route k. For any $\lambda_0 > 0$, the observed-context matrix, centered estimator, and confidence radius on this route are

$$\begin{aligned}
\boldsymbol{V}_{k,t} &= \lambda_0 \boldsymbol{I}_2 + \sum_{i \in \mathcal{H}_{k,t}} \boldsymbol{\psi}_i \boldsymbol{\psi}_i^\top, \\
\widehat{\boldsymbol{\theta}}_{k,t} &= \boldsymbol{\theta}^0_k + \boldsymbol{V}^{-1}_{k,t} \sum_{i \in \mathcal{H}_{k,t}} \boldsymbol{\psi}_i (\ell^{\mathrm{o}}_i - 1/2), \\
\beta_{k,t} &= \sigma\sqrt{\chi}\sqrt{\log \frac{\det \boldsymbol{V}_{k,t}}{\lambda_0^2} + 2\log(N_{\mathrm{rt}}/\delta)} + \sqrt{\lambda_0/2}.
\end{aligned} \tag{21}$$

Then $\boldsymbol{V}_t = \mathrm{diag}(\boldsymbol{V}_{1,t}, \ldots, \boldsymbol{V}_{N_{\mathrm{rt}},t})$, and $\widehat{\boldsymbol{\theta}}_t$ concatenates the route estimators. This centered estimator replaces (3). Predictability, conditional linear means, and the vector-MGF condition (12) still apply. Routes index parameter coordinates, not blocks of jointly revealed feedback. The Gaussian-mixture argument of Appendix B.1, applied to each route with failure probability δ/N_{rt}, retains the noise scale $\sigma\sqrt{\chi}$ and bounds the regularization term by $\sqrt{\lambda_0/2}$. The resulting union bound does not require independence across routes. It gives an event $\mathcal{E}_{\mathrm{route}}$ of probability at least $1-\delta$ on which $\|\widehat{\boldsymbol{\theta}}_{k,t} - \boldsymbol{\theta}^\star_k\|_{\boldsymbol{V}_{k,t}} \le \beta_{k,t}$ for every route and epoch.

For a compatible pair (r,g) on route k, write $w_{rg,t} = \beta_{k,t}\|\boldsymbol{\phi}_{rg,t}\|_{\boldsymbol{V}_t^{-1}}$. Clip $\boldsymbol{\phi}_{rg,t}^\top \widehat{\boldsymbol{\theta}}_t \pm w_{rg,t}$ to $[0,1]$. For each coordinate j on route k, also clip $\widehat{\theta}_{j,t} \pm \beta_{k,t}\sqrt{(\boldsymbol{V}_t^{-1})_{jj}}$ to $[0,1]$, collecting the bounds in $\underline{\boldsymbol{\theta}}_t$ and $\overline{\boldsymbol{\theta}}_t$. Intersect the prediction interval with $[\boldsymbol{\phi}_{rg,t}^\top \underline{\boldsymbol{\theta}}_t, \boldsymbol{\phi}_{rg,t}^\top \overline{\boldsymbol{\theta}}_t]$ to obtain $\ell^-_{rg,t}, \ell^+_{rg,t}$. On $\mathcal{E}_{\mathrm{route}}$, both intervals contain the conditional mean: nonnegative context coordinates preserve the endpoint bounds in the second interval. The intersection is nonempty, and $0 \le \ell_{rg,t} - \ell^-_{rg,t} \le 2w_{rg,t}$. Stopping uses the selected widths $w_{rg,t}$ in place of $\beta_t\|\boldsymbol{\phi}_{rg,t}\|_{\boldsymbol{V}_t^{-1}}$, preserving the excess-cost argument. The primary confirmation experiments use this route parameterization.

D Component Scheduling

D.1 Rounded Local Scheduling

Under Assumption 2, omit epoch subscripts and write c_g and $c^{\max}$ for the scalar result reservation and capacity. Each covering result subset $\mathcal{U} \subseteq \mathcal{G}_i$ defines a state by assigning tasks to their cheapest compatible options under optimistic costs,

$$J_i(\mathcal{U}) = \sum_{g \in \mathcal{U}} F_g + \sum_{r \in \mathcal{R}_i} \min_{g \in \mathcal{U}:(r,g)\in\mathcal{E}} \widetilde{e}_{rg}, \qquad c_i(\mathcal{U}) = \sum_{g \in \mathcal{U}} c_g. \tag{22}$$

Remove unused results and update each state's cost and reservation. Discard duplicate states and states with $c_i(\mathcal{U}) > c^{\max}$ to obtain the retained states $\mathcal{Q}_i$. These states preserve the integral optimum $m^- = \min_{\boldsymbol{a} \in \mathcal{A}(\mathsf{s})} J^-(\boldsymbol{a})$ (Lemma D.1). The component minimum is $\underline{J}_i = \min_{\mathcal{U} \in \mathcal{Q}_i} J_i(\mathcal{U})$. For an incumbent with optimistic cost U_0, the initial lower bound is $L_0 = \sum_i \underline{J}_i$ and the initial gap is $B = U_0 - L_0$. The retained-state count is $N^{\mathrm{r}} = \sum_i |\mathcal{Q}_i|$, and G^{g} bounds state-generation work. Dynamic programming selects one state per component under the shared capacity, rounding costs

above component minima and retaining the minimum exact reservation per rounded total. The feasible schedule and global lower bound differ by at most $\nu B/\kappa$ (Lemma D.2).

Algorithm 3: Rounded Local Scheduling (RLS)

Input: Observed state s, optimistic assignment costs $\widetilde{e}_{rg}$, feasible incumbent $\boldsymbol{a}^0$, $\boldsymbol{V}, \beta$; $\alpha, \omega > 0$; optional deadline.
Output: Feasible decision $\boldsymbol{a}^0$ and valid global lower bound L^-.

Generate retained states $\mathcal{Q}_i$ once by (22)
$\underline{J}_i \leftarrow \min_{\mathcal{U}\in\mathcal{Q}_i} J_i(\mathcal{U})$
$U_0 \leftarrow J^-(\boldsymbol{a}^0)$, $L_0 \leftarrow \sum_i \underline{J}_i$, $B \leftarrow U_0 - L_0$, $L^- \leftarrow L_0$, $\kappa \leftarrow \nu$
while (6) *fails and time remains* **do**
  $h \leftarrow B/\kappa$
  $q_i(\mathcal{U}) \leftarrow \lfloor (J_i(\mathcal{U}) - \underline{J}_i)/h \rfloor$ for each $\mathcal{U} \in \mathcal{Q}_i$
  Compute $f_i^{\mathrm{res}}(z)$ by (25)
  $z_* \leftarrow \min\{z \le \kappa : f_\nu^{\mathrm{res}}(z) \le c^{\max}\}$
  Reconstruct $\boldsymbol{a}_\kappa$ from z_* and set $L_\kappa \leftarrow L_0 + hz_*$
  $\boldsymbol{a}^0 \leftarrow \arg\min_{\boldsymbol{a}\in\{\boldsymbol{a}^0,\boldsymbol{a}_\kappa\}} J^-(\boldsymbol{a})$
  $L^- \leftarrow \max\{L^-, L_\kappa\}$, $\kappa \leftarrow 2\kappa$
return $\boldsymbol{a}^0, L^-$

For a component with K_{c} results and M_{c} tasks, there are at most $2^{K_{\mathrm{c}}} - 1$ nonempty result subsets, compared with at most ${K_{\mathrm{c}}}^{M_{\mathrm{c}}}$ assignments.

Lemma D.1 (Exact subset representation)**.** *Under Assumption 2,*

$$\min_{\{\mathcal{U}_i\}:\,\sum_i c_i(\mathcal{U}_i)\le c^{\max}} \sum_i J_i(\mathcal{U}_i) = m^-.$$

Proof. For any feasible schedule, remove unused selected results without increasing cost or reservations, and let $\mathcal{U}_i$ contain the remaining results in component i. Assigning each task to its cheapest compatible result in $\mathcal{U}_i$ cannot increase its optimistic assignment cost, so the state optimum is at most m^-. Conversely, every feasible state selection gives compatible minimum-cost assignments. Removing unused results cannot increase preparation cost or reservations, so the resulting schedule is feasible and costs at most the state objective. The two optima are therefore equal. Discarding duplicate states and states with $c_i(\mathcal{U}) > c^{\max}$ preserves this value. □

Lemma D.2 (Component scheduling)**.** *Under Assumption 2, every integer $\kappa \ge 1$ yields a feasible $\widehat{\boldsymbol{a}}$ and global lower bound L_κ satisfying*

$$L_\kappa \le m^- \le J^-(\widehat{\boldsymbol{a}}), \qquad J^-(\widehat{\boldsymbol{a}}) - L_\kappa \le \frac{\nu B}{\kappa}, \tag{23}$$

in $O(G^{\mathrm{g}} + N^{\mathrm{r}}(\kappa+1))$ arithmetic operations.

Proof of Lemma D.2. If $B = 0$, the incumbent and L_0 coincide with m^-, so return the incumbent with $L_\kappa = L_0$. Otherwise set the rounding unit $h = B/\kappa$. The excess cost and rounded excess above the component minimum are

$$\Delta_i(\mathcal{U}) = J_i(\mathcal{U}) - \underline{J}_i \ge 0, \qquad q_i(\mathcal{U}) = \left\lfloor \frac{\Delta_i(\mathcal{U})}{h} \right\rfloor. \tag{24}$$

Discard states with $\Delta_i(\mathcal{U}) > B$. An optimal state selection has total excess at most B, so its rounded excess is at most κ.

The value $f_i^{\mathrm{res}}(z)$ is the minimum exact reservation of one state per component among the first i components with total rounded excess z. Initialize $f_0^{\mathrm{res}}(0) = 0$ and $f_0^{\mathrm{res}}(z) = +\infty$ otherwise, and compute

$$f_i^{\mathrm{res}}(z) = \min_{\mathcal{U}\in\mathcal{Q}_i:\,\Delta_i(\mathcal{U})\le B,\ q_i(\mathcal{U})\le z} \{f_{i-1}^{\mathrm{res}}(z - q_i(\mathcal{U})) + c_i(\mathcal{U})\}, \qquad 0 \le z \le \kappa. \tag{25}$$

Removing the last state leaves rounded total $z - q_i(\mathcal{U})$; appending it adds reservation $c_i(\mathcal{U})$. No resource coefficient is rounded. The smallest feasible rounded total is $z_* = \min\{z \le \kappa : f_\nu^{\mathrm{res}}(z) \le$

$c^{\max}\}$, which exists because an optimal selection is retained. Reconstruct its states and remove unused results. Let $\widehat{\mathcal{U}}_i$ be this selection and $\boldsymbol{a}_\kappa$ the resulting feasible schedule, and set

$$L_\kappa = L_0 + hz_*. \tag{26}$$

For an optimal selection $\mathcal{U}_i^*$,

$$L_\kappa \le L_0 + h\sum_i q_i(\mathcal{U}_i^*) \le L_0 + \sum_i \Delta_i(\mathcal{U}_i^*) = m^-.$$

For the reconstructed selection $\widehat{\mathcal{U}}_i$, rounding down gives

$$J^-(\boldsymbol{a}_\kappa) \le \sum_i J_i(\widehat{\mathcal{U}}_i) < L_0 + h(z_* + \nu) = L_\kappa + \nu B/\kappa.$$

Let $\widehat{\boldsymbol{a}}$ be the cheaper of $\boldsymbol{a}_\kappa$ and the incumbent. Replacing L_κ by any larger valid global lower bound can only reduce the gap, and (23) follows. Each state is processed for at most $\kappa+1$ rounded totals, so the arithmetic work is $O(G^{\mathrm{g}} + N^{\mathrm{r}}(\kappa+1))$. □

For fixed component size bounds $K_{\mathrm{c}}, M_{\mathrm{c}}$ and resolution κ, this work is linear in the component count.

Corollary D.1 (Component complexity)**.** *Under Assumption 2, $|\mathcal{G}_i| \le K_{\mathrm{c}}$ and $|\mathcal{R}_i| \le M_{\mathrm{c}}$, the component solver uses $O(2^{K_{\mathrm{c}}}\nu(K_{\mathrm{c}}M_{\mathrm{c}}+\kappa))$ arithmetic operations.*

Proof of Corollary D.1. With $|\mathcal{G}_i| \le K_{\mathrm{c}}$ and $|\mathcal{R}_i| \le M_{\mathrm{c}}$, Lemma D.2 gives $N^{\mathrm{r}} \le \nu(2^{K_{\mathrm{c}}}-1)$ and $G^{\mathrm{g}} = \sum_i 2^{|\mathcal{G}_i|}|\mathcal{G}_i|(|\mathcal{R}_i|+1) \le \nu 2^{K_{\mathrm{c}}}K_{\mathrm{c}}(M_{\mathrm{c}}+1)$, so $G^{\mathrm{g}} + N^{\mathrm{r}}(\kappa+1) = O(2^{K_{\mathrm{c}}}\nu(K_{\mathrm{c}}M_{\mathrm{c}}+\kappa))$. Enumerating all preparation decisions instead considers $2^{\sum_i |\mathcal{G}_i|}$ subsets. □

D.2 Exact Dynamic Programming

For nonnegative integer reservations and integer capacity $c^{\max}$, Exact DP selects one retained state per component under the shared capacity. Let $f_i(c)$ be the minimum cost of the first i components with total reservation exactly c. Initialize $f_0(0) = 0$ and $f_0(c) = +\infty$ for $c \in \{1, \dots, c^{\max}\}$, and compute

$$f_i(c) = \min_{\mathcal{U}\in\mathcal{Q}_i:\, c_i(\mathcal{U})\le c} \{f_{i-1}(c - c_i(\mathcal{U})) + J_i(\mathcal{U})\}, \qquad 0 \le c \le c^{\max}. \tag{27}$$

Appending a state $\mathcal{U}$ to a selection of reservation $c - c_i(\mathcal{U})$ gives total reservation c, so $m^- = \min_{c\le c^{\max}} f_\nu(c)$. The recurrence takes $O((c^{\max}+1)N^{\mathrm{r}})$ arithmetic work, plus $O(G^{\mathrm{g}})$ to generate states. Because this work depends on the value of $c^{\max}$, Exact DP is pseudo-polynomial.

The component dispatch and scaling experiments use assignment catalogues rather than result-subset enumeration. Both preserve the integral optimum. With four tasks and four results, an assignment catalogue examines up to 256 assignments and result-subset enumeration at most 15 nonempty subsets. Exact DP combines retained states by capacity, and RLS by rounded excess cost. The bound $G^{\mathrm{g}} = O(\nu 2^{K_{\mathrm{c}}}K_{\mathrm{c}}(M_{\mathrm{c}}+1))$ applies to result-subset enumeration, not to these assignment catalogues. Reported times include catalogue construction and any overrun.

E Joint Scheduling

E.1 General Local Scheduling

Fix an epoch and omit its subscript. Assignment and capacity prices give a Lagrangian lower bound. Let $[v]_+ = \max\{v, 0\}$, with assignment prices $\boldsymbol{u} \in \mathbb{R}^{|\mathcal{R}|}$ and capacity prices $\boldsymbol{\pi} \ge 0$. Define

$$b_g(\boldsymbol{u}, \boldsymbol{\pi}) = F_g + \boldsymbol{\pi}^\top \boldsymbol{c}_g - \sum_{r:(r,g)\in\mathcal{E}} [u_r - \widetilde{e}_{rg} - \boldsymbol{\pi}^\top \boldsymbol{c}_{rg}]_+,$$

$$L(\boldsymbol{u}, \boldsymbol{\pi}) = \sum_r u_r - \boldsymbol{\pi}^\top \bar{\boldsymbol{c}} + \sum_g \min\{0, b_g(\boldsymbol{u}, \boldsymbol{\pi})\}. \tag{28}$$

The term b_g compares the preparation cost and priced resource reservation of result g with the assignment benefit under these prices. Proposition E.1 gives $L(\boldsymbol{u}, \boldsymbol{\pi}) \le m^-$ and screens results that cannot improve the incumbent.

Algorithm 4: General Local Scheduling (GLS)

Input: Observed state s, optimistic assignment costs $\widetilde{e}_{rg}$, feasible incumbent $\boldsymbol{a}^0$, $\boldsymbol{V}, \beta$; $\alpha, \omega > 0$; optional deadline.
Output: Feasible decision $\boldsymbol{a}^0$ and valid global lower bound L^-.

$u_r \leftarrow \min_{g:(r,g)\in\mathcal{E}} \widetilde{e}_{rg}$, $\boldsymbol{\pi} \leftarrow \mathbf{0}$
$L^- \leftarrow L(\boldsymbol{u}, \boldsymbol{\pi})$ from (28)
while (6) *fails and time remains* **do**
  Update prices by (33)
  $L^- \leftarrow \max\{L^-, L(\boldsymbol{u}, \boldsymbol{\pi})\}$
  Screen results by Proposition E.1
  Improve $\boldsymbol{a}^0$ by feasible collective moves with $\Delta J^- < 0$ using (29)–(30)
  if (6) *still fails and time remains* **then**
    Use LP/MILP to seek a feasible candidate $\widehat{\boldsymbol{a}}$ and a valid global bound $\widehat{L}$
    If $\widehat{\boldsymbol{a}}$ is verified feasible, set $\boldsymbol{a}^0 \leftarrow \arg\min_{\boldsymbol{a}\in\{\boldsymbol{a}^0,\widehat{\boldsymbol{a}}\}} J^-(\boldsymbol{a})$
    If $\widehat{L}$ is verified valid, set $L^- \leftarrow \max\{L^-, \widehat{L}\}$
return $\boldsymbol{a}^0, L^-$

Accepted moves jointly change result selections and task assignments. Consider reassigning a nonempty compatible task group $\mathcal{M}$ to result g. Let $\iota(r)$ be the result currently assigned to task r, and let $\mathcal{Z}$ contain currently used results other than g whose tasks all lie in $\mathcal{M}$. The move selects g and removes the results in $\mathcal{Z}$, changing the optimistic cost by

$$\Delta J^- = F_g(1 - y_g) - \sum_{h\in\mathcal{Z}} F_h + \sum_{r\in\mathcal{M}} (\widetilde{e}_{rg} - \widetilde{e}_{r,\iota(r)}). \tag{29}$$

The first two terms account for changes in preparation costs, and the last for changes in execution cost and optimistic quality loss. The current resource reservation is $\boldsymbol{c}(\boldsymbol{a}) = \sum_g \boldsymbol{c}_g y_g + \sum_{(r,g)\in\mathcal{E}} \boldsymbol{c}_{rg} x_{rg}$, and the move changes it by

$$\Delta\boldsymbol{c} = \boldsymbol{c}_g(1 - y_g) - \sum_{h\in\mathcal{Z}} \boldsymbol{c}_h + \sum_{r\in\mathcal{M}} (\boldsymbol{c}_{rg} - \boldsymbol{c}_{r,\iota(r)}). \tag{30}$$

A move is accepted only if the new assignments are compatible, $\boldsymbol{c}(\boldsymbol{a}) + \Delta\boldsymbol{c} \le \bar{\boldsymbol{c}}$, and $\Delta J^- < 0$. It preserves feasibility and reduces the optimistic cost.

Proposition E.1 (Global lower bound and screening)**.** *For any $\boldsymbol{u} \in \mathbb{R}^{|\mathcal{R}|}$ and $\boldsymbol{\pi} \ge 0$, $L(\boldsymbol{u}, \boldsymbol{\pi}) \le m^-$. Every feasible schedule selecting g has optimistic cost at least $L(\boldsymbol{u}, \boldsymbol{\pi}) + [b_g(\boldsymbol{u}, \boldsymbol{\pi})]_+$.*

Define the preparation and optimistic assignment costs adjusted by capacity prices as $F_g^{\boldsymbol{\pi}} = F_g + \boldsymbol{\pi}^\top \boldsymbol{c}_g$ and $\widetilde{e}_{rg}^{\boldsymbol{\pi}} = \widetilde{e}_{rg} + \boldsymbol{\pi}^\top \boldsymbol{c}_{rg}$.

Local minimization. Dualize the assignment equations by $\sum_r u_r(1 - \sum_g x_{rg})$ and the capacity inequality by $\boldsymbol{\pi}^\top(\sum_g \boldsymbol{c}_g y_g + \sum_{r,g} \boldsymbol{c}_{rg} x_{rg} - \bar{\boldsymbol{c}})$. With the local constraints $x_{rg} \le y_g$ and binary restrictions, the Lagrangian is

$$\mathrm{lag}(\boldsymbol{x}, \boldsymbol{y}; \boldsymbol{u}, \boldsymbol{\pi}) = \sum_r u_r - \boldsymbol{\pi}^\top \bar{\boldsymbol{c}} + \sum_g \left(F_g^{\boldsymbol{\pi}} y_g + \sum_{r:(r,g)\in\mathcal{E}} (\widetilde{e}_{rg}^{\boldsymbol{\pi}} - u_r) x_{rg} \right). \tag{31}$$

For each result g, $y_g = 0$ contributes zero. If $y_g = 1$, set $x_{rg} = 1$ when $u_r > \widetilde{e}_{rg}^{\boldsymbol{\pi}}$ and $x_{rg} = 0$ when $u_r < \widetilde{e}_{rg}^{\boldsymbol{\pi}}$, with either value at a tie, giving $b_g(\boldsymbol{u}, \boldsymbol{\pi})$. The local minimum is $\min\{0, b_g(\boldsymbol{u}, \boldsymbol{\pi})\}$, which yields (28).

Proof of Proposition E.1. For any feasible schedule $\boldsymbol{a}$, the assignment terms vanish and the capacity term in (31) is nonpositive, so

$$L(\boldsymbol{u}, \boldsymbol{\pi}) = \min_{\substack{x_{rg}\le y_g \ ((r,g)\in\mathcal{E}) \\ \boldsymbol{x},\boldsymbol{y} \text{ binary}}} \mathrm{lag}(\boldsymbol{x}, \boldsymbol{y}; \boldsymbol{u}, \boldsymbol{\pi}) \le \mathrm{lag}(\boldsymbol{a}; \boldsymbol{u}, \boldsymbol{\pi}) \le J^-(\boldsymbol{a}).$$

A feasible primal LP objective alone need not lower-bound the integer minimum unless LP optimality is established. Forcing $y_g = 1$ replaces $\min\{0, b_g\}$ by b_g, giving the lower bound $L + [b_g]_+$. If this

exceeds $J^-(\bar{\boldsymbol{a}})$ for a feasible incumbent $\bar{\boldsymbol{a}}$, no optimal schedule under the optimistic costs can select g. □

E.2 GLS Initialization and Price Updates

Before initializing the prices in Algorithm 4, GLS compares the supplied feasible fallback, a cached feasible decision for the current template, and an incremental greedy decision under the current optimistic costs. It retains the cheapest completed feasible candidate. The initial lower bound uses $u_r = \min_{g:(r,g)\in\mathcal{E}} \widetilde{e}_{rg}$ and $\boldsymbol{\pi} = \mathbf{0}$, giving $b_g = F_g \geq 0$ and $L = \sum_r u_r$.

Price updates. Let $(\widetilde{\boldsymbol{x}}, \widetilde{\boldsymbol{y}})$ be a minimizer of the relaxed Lagrangian for the current prices. A supergradient of L is

$$h_r = 1 - \sum_{g:(r,g)\in\mathcal{E}} \widetilde{x}_{rg}, \qquad \boldsymbol{s}^{\mathrm{cap}} = \sum_g \boldsymbol{c}_g \widetilde{y}_g + \sum_{(r,g)\in\mathcal{E}} \boldsymbol{c}_{rg}\widetilde{x}_{rg} - \bar{\boldsymbol{c}}. \tag{32}$$

At this minimizer,

$$L(\boldsymbol{u}', \boldsymbol{\pi}') \leq L(\boldsymbol{u}, \boldsymbol{\pi}) + \boldsymbol{h}^\top(\boldsymbol{u}' - \boldsymbol{u}) + (\boldsymbol{s}^{\mathrm{cap}})^\top(\boldsymbol{\pi}' - \boldsymbol{\pi}).$$

For any step size $\gamma_k > 0$, the projected update is

$$\boldsymbol{u}^{k+1} = \boldsymbol{u}^k + \gamma_k \boldsymbol{h}^k, \qquad \boldsymbol{\pi}^{k+1} = [\boldsymbol{\pi}^k + \gamma_k \boldsymbol{s}^{\mathrm{cap},\,k}]_+, \tag{33}$$

with the capacity prices projected coordinatewise. An update need not improve L, so the solver retains the largest valid lower bound seen so far.

F Decision Quality

F.1 Proof of Theorem 1

We use the linear quality model in Section 3.1. At each epoch, either RLS or GLS supplies JAS with a feasible schedule and a valid global lower bound. Let $N_T = \sum_{t\leq T} |\mathcal{R}_t|$ be the total task count. Define the *logarithmic potential* Λ_T and the confidence-radius bound $\overline{\beta}_T$ as

$$\Lambda_T = D_\phi \log\!\left(1 + \frac{N_T}{\lambda_0 D_\phi}\right), \qquad \overline{\beta}_T = \sigma\sqrt{\Lambda_T + 2\log(1/\delta)} + \sqrt{\lambda_0}S. \tag{34}$$

The *selected-context matrix* $\boldsymbol{\Phi}_t = \sum_{(r,g):x_{rg,t}=1} \boldsymbol{\phi}_{rg,t}\boldsymbol{\phi}_{rg,t}^\top$ collects the contexts selected at epoch t. The matrix $\boldsymbol{W}_t = \lambda_0 \boldsymbol{I} + \sum_{s<t} \boldsymbol{\Phi}_s$ includes every selected assignment before epoch t, whereas $\boldsymbol{V}_t$ includes only revealed feedback. Their difference captures pending feedback, whose effect on selected assignments is measured by the *directional delay term* H_T. Define

$$H_T = \sum_{t\leq T} \mathrm{tr}\big((\boldsymbol{V}_t^{-1} - \boldsymbol{W}_t^{-1})\boldsymbol{\Phi}_t\big), \qquad \Xi_T = \sqrt{N_T(2\Lambda_T + H_T)}. \tag{35}$$

Here Ξ_T bounds cumulative context uncertainty. We write $\boldsymbol{a}_t = \boldsymbol{a}_t^0$ for the incumbent.

We first bound the per-epoch excess cost for $\mathcal{P}_t$, then regret for $\mathbb{P}$, accounting for selected contexts and pending feedback. All sums over (r, g) at epoch t refer to compatible pairs in $\mathcal{E}_t$.

Proof. **Step 1: Confidence containment.** Under Assumption 1 in reveal order, the event $\mathcal{E}_{\mathrm{conf}}$ in (20) has probability at least $1 - \delta$ for any $\lambda_0 > 0$, including the theorem's choice $\lambda_0 \geq \max\{1, M\}$. On this event, Cauchy–Schwarz gives, for every compatible pair and epoch,

$$\left|\boldsymbol{\phi}_{rg,t}^\top(\widehat{\boldsymbol{\theta}}_t - \boldsymbol{\theta}^\star)\right| \leq \beta_t \|\boldsymbol{\phi}_{rg,t}\|_{\boldsymbol{V}_t^{-1}}.$$

Clipping preserves containment because expected quality losses lie in $[0, 1]$:

$$\ell_{rg,t}^- \leq \ell_{rg,t} \leq \ell_{rg,t}^+, \qquad 0 \leq \ell_{rg,t} - \ell_{rg,t}^- \leq 2\beta_t \|\boldsymbol{\phi}_{rg,t}\|_{\boldsymbol{V}_t^{-1}}. \tag{36}$$

This event covers contexts selected adaptively from the observed state and revealed feedback, without an additional union bound over decisions.

Step 2: Per-epoch decision quality. Optimistic costs separate learning and optimization error. Since J_t^-, J_t and J_t^+ differ only in quality loss, every feasible decision $\boldsymbol{a}$ satisfies $J_t^-(\boldsymbol{a}) \le J_t(\boldsymbol{a}) \le J_t^+(\boldsymbol{a})$ on $\mathcal{E}_{\mathrm{conf}}$. The global lower bound gives

$$L_t^- \le m_t^- \le J_t^-(\boldsymbol{a}_t^\star) \le J_t(\boldsymbol{a}_t^\star).$$

Together with $J_t(\boldsymbol{a}_t) \le J_t^+(\boldsymbol{a}_t)$, this proves the computable bound (7) for $\mathcal{P}_t$. Using the actual optimization error $\eta_t = J_t^-(\boldsymbol{a}_t) - m_t^-$ from (5), we obtain

$$\begin{aligned} J_t(\boldsymbol{a}_t) - J_t(\boldsymbol{a}_t^\star) &= \big[J_t(\boldsymbol{a}_t) - J_t^-(\boldsymbol{a}_t)\big] + \eta_t + \big[m_t^- - J_t(\boldsymbol{a}_t^\star)\big] \\ &\le \eta_t + \omega \sum_{(r,g):x_{rg,t}=1} (\ell_{rg,t} - \ell_{rg,t}^-) \\ &\le \eta_t + 2\omega\beta_t \sum_{(r,g):x_{rg,t}=1} \|\boldsymbol{\phi}_{rg,t}\|_{\boldsymbol{V}_t^{-1}}. \end{aligned} \tag{37}$$

Both decisions use the same feasible set $\mathcal{A}_t(\mathsf{s}_t)$, as required by ($\mathbb{P}$).

Step 3: Selected-assignment potential. The matrix $\boldsymbol{W}_t$ bounds uncertainty before accounting for delayed feedback. It satisfies $\boldsymbol{W}_{t+1} = \boldsymbol{W}_t + \boldsymbol{\Phi}_t$ and $\boldsymbol{W}_1 = \lambda_0 \boldsymbol{I}$. Each epoch selects one assignment for each admitted task, so the context-norm bound implies

$$0 \preceq \boldsymbol{\Phi}_t \preceq |\mathcal{R}_t| \boldsymbol{I} \preceq M \boldsymbol{I} \preceq \lambda_0 \boldsymbol{I} \preceq \boldsymbol{W}_t.$$

Thus the eigenvalues of $\boldsymbol{W}_t^{-1/2} \boldsymbol{\Phi}_t \boldsymbol{W}_t^{-1/2}$ lie in $[0,1]$. On this interval, $u \le 2\log(1+u)$ because $2\log(1+u) - u$ starts at zero and has derivative $(1-u)/(1+u) \ge 0$. Applying this inequality gives

$$\begin{aligned} \operatorname{tr}(\boldsymbol{W}_t^{-1} \boldsymbol{\Phi}_t) &= \operatorname{tr}(\boldsymbol{W}_t^{-1/2} \boldsymbol{\Phi}_t \boldsymbol{W}_t^{-1/2}) \\ &\le 2 \log\det(\boldsymbol{I} + \boldsymbol{W}_t^{-1/2} \boldsymbol{\Phi}_t \boldsymbol{W}_t^{-1/2}) \\ &= 2 \log \frac{\det \boldsymbol{W}_{t+1}}{\det \boldsymbol{W}_t}. \end{aligned}$$

Moreover, $\operatorname{tr}(\boldsymbol{W}_{T+1}) \le \lambda_0 D_\phi + N_T$. The arithmetic–geometric mean inequality for its eigenvalues gives

$$\log \frac{\det \boldsymbol{W}_{T+1}}{\det(\lambda_0 \boldsymbol{I})} \le D_\phi \log\left(\frac{\operatorname{tr}(\boldsymbol{W}_{T+1})}{\lambda_0 D_\phi}\right) \le D_\phi \log\left(1 + \frac{N_T}{\lambda_0 D_\phi}\right) = \Lambda_T.$$

Summing over epochs and telescoping gives

$$\sum_{t=1}^{T} \operatorname{tr}(\boldsymbol{W}_t^{-1} \boldsymbol{\Phi}_t) \le 2 \log \frac{\det \boldsymbol{W}_{T+1}}{\det(\lambda_0 \boldsymbol{I})} \le 2\Lambda_T. \tag{38}$$

This argument uses the full batch $\boldsymbol{\Phi}_t$, without requiring feedback between assignments in the same epoch.

Step 4: Pending feedback and selected uncertainty. Pending feedback adds the nonnegative uncertainty term H_T. The matrix $\boldsymbol{P}_t = \boldsymbol{W}_t - \boldsymbol{V}_t$ sums the context outer products of observations pending before epoch t. Each revealed observation enters the history once, so $\boldsymbol{P}_t \succeq 0$ and $\boldsymbol{V}_t \preceq \boldsymbol{W}_t$. Inversion gives $\boldsymbol{V}_t^{-1} - \boldsymbol{W}_t^{-1} \succeq 0$, making every term in H_T nonnegative. By the definition of $\boldsymbol{\Phi}_t$,

$$\begin{aligned} \sum_{t=1}^{T} \sum_{(r,g):x_{rg,t}=1} \|\boldsymbol{\phi}_{rg,t}\|_{\boldsymbol{V}_t^{-1}}^2 &= \sum_{t=1}^{T} \operatorname{tr}(\boldsymbol{V}_t^{-1} \boldsymbol{\Phi}_t) \\ &= \sum_{t=1}^{T} \operatorname{tr}(\boldsymbol{W}_t^{-1} \boldsymbol{\Phi}_t) + H_T \le 2\Lambda_T + H_T. \end{aligned} \tag{39}$$

Applying Cauchy–Schwarz to the N_T selected assignments gives

$$\sum_{t=1}^{T} \sum_{(r,g):x_{rg,t}=1} \|\boldsymbol{\phi}_{rg,t}\|_{\boldsymbol{V}_t^{-1}} \le \sqrt{N_T(2\Lambda_T + H_T)} = \Xi_T. \tag{40}$$

If $N_T = 0$, both sides are zero. Finally, $\boldsymbol{V}_t \preceq \boldsymbol{W}_{T+1}$ and the same determinant bound used above imply

$$\beta_t \le \sigma\sqrt{\Lambda_T + 2\log(1/\delta)} + \sqrt{\lambda_0} S = \overline{\beta}_T.$$

Summing (37) and applying the radius bound and (40) yields

$$R_T \le \sum_{t=1}^{T} \eta_t + 2\omega\overline{\beta}_T \Xi_T,$$

which proves (8) for $\mathbb{P}$ and separates optimization error from learning error. The matrix inequalities hold for every selected-context and reveal-time sequence allowed by the model. Both conclusions therefore hold on the same event $\mathcal{E}_{\mathrm{conf}}$, with probability at least $1-\delta$. □

Quality-aware stopping. Quality-aware stopping keeps optimization error at the scale of learning uncertainty. If (6) is satisfied at every nonempty epoch, the valid lower bound L_t^- gives

$$\eta_t \le J_t^-(\boldsymbol{a}_t) - L_t^- \le \alpha\omega\beta_t \sum_{(r,g):x_{rg,t}=1} \|\boldsymbol{\phi}_{rg,t}\|_{\boldsymbol{V}_t^{-1}}.$$

Substituting this inequality into (37) and summing with (40) yields $R_T \le (2+\alpha)\omega\overline{\beta}_T\Xi_T$. If a deadline is reached first, the theorem accounts for the remaining η_t, which can also be replaced by its computable upper bound $J_t^-(\boldsymbol{a}_t) - L_t^-$.

Pending feedback and selected contexts. Pending feedback changes the widths of the currently selected contexts. Define the normalized pending-context and selected-context matrices

$$\boldsymbol{Q}_t = \boldsymbol{V}_t^{-1/2}\boldsymbol{P}_t\boldsymbol{V}_t^{-1/2}, \qquad \boldsymbol{Z}_t = \boldsymbol{V}_t^{-1/2}\boldsymbol{\Phi}_t\boldsymbol{V}_t^{-1/2}.$$

Since $\boldsymbol{W}_t = \boldsymbol{V}_t^{1/2}(\boldsymbol{I}+\boldsymbol{Q}_t)\boldsymbol{V}_t^{1/2}$, matrix inversion and cyclicity of trace give

$$\begin{aligned}\operatorname{tr}\big((\boldsymbol{V}_t^{-1}-\boldsymbol{W}_t^{-1})\boldsymbol{\Phi}_t\big) &= \operatorname{tr}\big([\boldsymbol{I}-(\boldsymbol{I}+\boldsymbol{Q}_t)^{-1}]\boldsymbol{Z}_t\big)\\ &= \operatorname{tr}\big(\boldsymbol{Q}_t(\boldsymbol{I}+\boldsymbol{Q}_t)^{-1}\boldsymbol{Z}_t\big).\end{aligned}$$

The contribution is zero when $\boldsymbol{Q}_t\boldsymbol{Z}_t = 0$. To bound it for every pattern of pending and selected contexts, let $\rho_t = \|\boldsymbol{Q}_t\|_{\mathrm{op}}$. The inequalities $\boldsymbol{P}_t \preceq \rho_t\boldsymbol{V}_t$ and $\boldsymbol{W}_t \preceq (1+\rho_t)\boldsymbol{V}_t$ imply

$$\boldsymbol{V}_t^{-1}-\boldsymbol{W}_t^{-1} \preceq \rho_t\boldsymbol{W}_t^{-1}, \qquad \operatorname{tr}\big((\boldsymbol{V}_t^{-1}-\boldsymbol{W}_t^{-1})\boldsymbol{\Phi}_t\big) \le \rho_t \operatorname{tr}(\boldsymbol{W}_t^{-1}\boldsymbol{\Phi}_t).$$

At most MD observations are pending, each with context norm at most one. Thus $\boldsymbol{P}_t \preceq MD\boldsymbol{I}$ and $\boldsymbol{V}_t \succeq \lambda_0\boldsymbol{I}$ imply $\rho_t \le MD/\lambda_0$. Equation (38) then gives

$$H_T \le 2\big(\max_{t\le T}\rho_t\big)\Lambda_T \le \frac{2MD}{\lambda_0}\Lambda_T, \qquad \Xi_T \le \sqrt{2N_T(1+MD/\lambda_0)\Lambda_T}. \tag{41}$$

Corollary 1 uses this bound. The realized H_T depends on how selected contexts align with pending observations. A bound using only the pending count MD omits this dependence.

Same pending count, different effect. The same number of pending observations can add different amounts of uncertainty, as a two-dimensional example shows. Let $\boldsymbol{V}_t = \operatorname{diag}(v_1, v_2)$ with $v_1, v_2 \ge \lambda_0$, and suppose $n_{\mathrm{pend}} \le MD$ pending observations all have context $\boldsymbol{e}_1$, so $\boldsymbol{P}_t = n_{\mathrm{pend}}\boldsymbol{e}_1\boldsymbol{e}_1^\top$. If $n_{\mathrm{cur}} \le M$ current assignments all have context $\boldsymbol{e}_2$, their contribution to H_T is zero. If their contexts instead equal $\boldsymbol{e}_1$, the contribution is

$$n_{\mathrm{cur}}\left(\frac{1}{v_1} - \frac{1}{v_1+n_{\mathrm{pend}}}\right) = \frac{n_{\mathrm{cur}}n_{\mathrm{pend}}}{v_1(v_1+n_{\mathrm{pend}})}.$$

In this example, pending feedback adds uncertainty relative to $\boldsymbol{W}_t$ only when selected contexts align with the pending observations.

Dependent feedback blocks. Dependent feedback changes confidence widths while preserving the matrix argument, which uses only selected contexts and reveal times. Under the conditional block-feedback assumptions in Appendix B.1, Proposition B.1 provides the required simultaneous confidence event with radii $\beta_t^{(\chi)}$ and $\overline{\beta}_T^{(\chi)}$. Using these radii preserves the proof, the stopping guarantee and the delay bound. The dependence factor adjusts confidence widths, while H_T captures pending feedback through the selected contexts.

Relation to approximate-oracle bandits. Approximate-oracle analyses bound regret by learning error plus oracle error. CUCB assumes an (α, β)-approximation oracle and bounds regret against the $\alpha\beta$-scaled optimum (Chen et al., 2013). AESCB computes the ESCB index approximately, with error allowed to scale with the confidence widths, using budgeted linear maximization over the action set (Cuvelier et al., 2021). Theorem 1 uses the same separation but differs in three respects. (i) *Oracle.* Both analyses specify a priori approximation guarantees for their optimization subroutines. For $\mathcal{P}_t$, even linear minimization over the feasible set with known costs is Set-Cover-hard (Proposition 1). Thus, a polynomial-time exact oracle cannot be assumed unless $\mathrm{P} = \mathrm{NP}$. We compare with the exact optimum $\boldsymbol{a}_t^\star$ and account for the additive optimization error. (ii) *Certificate.* JAS returns a feasible schedule and a valid global lower bound, so $\eta_t \le J_t^-(\boldsymbol{a}_t) - L_t^-$ holds at every epoch, including when a deadline interrupts the solver. (iii) *Tolerance and work.* AESCB enforces its approximation tolerance by construction. JAS checks the optimization gap against the selected assignments' confidence widths in (6). For RLS under Assumption 2, Theorem 2 bounds the work needed to pass this check when the tolerance lower bound is positive and no deadline is imposed.

F.2 Proof of Corollary 1

The corollary follows from Theorem 1 and the delay bound (41). The notation $\widetilde{O}$ suppresses logarithmic factors in T.

Proof of Corollary 1. On the confidence event of Theorem 1, satisfying (6) at every nonempty epoch gives

$$R_T \le (2+\alpha)\omega\overline{\beta}_T\sqrt{N_T(2\Lambda_T + H_T)}.$$

The delay bound in Appendix F.1 and the workload bound give

$$H_T \le \frac{2MD}{\lambda_0}\Lambda_T, \qquad N_T \le MT.$$

Substituting both inequalities yields

$$R_T \le (2+\alpha)\omega\overline{\beta}_T\sqrt{2MT\left(1+\frac{MD}{\lambda_0}\right)\Lambda_T}. \tag{42}$$

For fixed D_ϕ, M, λ_0, $\Lambda_T \le D_\phi \log(1 + MT/(\lambda_0 D_\phi)) = O(\log(1+T))$. For the horizon rate, we also hold $D, \alpha, \omega, \sigma, S, \delta$ fixed, so

$$\overline{\beta}_T = \sigma\sqrt{\Lambda_T + 2\log(1/\delta)} + \sqrt{\lambda_0}S = O(\sqrt{\log(1+T)}).$$

Equation (42) and $\overline{\beta}_T \le \sigma\sqrt{\Lambda_T} + \sigma\sqrt{2\log(1/\delta)} + \sqrt{\lambda_0}S$, with $\Lambda_T \le D_\phi \log(1 + MT/(\lambda_0 D_\phi))$, give the rate in (9), so $R_T = \widetilde{O}(\sqrt{T})$ with probability at least $1-\delta$. For $\lambda_0 = \max\{1, M\}$,

$$R_T = \widetilde{O}\Big((2+\alpha)\,\omega\sqrt{1+D}\,\big(\sigma D_\phi\sqrt{MT} + SM\sqrt{D_\phi T}\big)\Big).$$

With $k = M$ items per round and $d = D_\phi$, the two terms mirror the $\widetilde{O}(d\sqrt{kT})$ and $\widetilde{O}(Sk\sqrt{dT})$ cases of C^{2}UCB (Takemura et al., 2021), and, for $D = 0$, the first improves the task-count dependence of CombLinUCB's $\widetilde{O}(Kd\sqrt{n})$ bound (Wen et al., 2015) by a factor $\sqrt{M}$, with $K = M$ and $n = T$. Quality-aware stopping adds the factor $2+\alpha$, and the realized term H_T in Theorem 1 can replace the worst-case delay factor. □

G Computational Efficiency

G.1 Proof of Theorem 2

Let ν_t, N_t^{r} and G_t^{g} denote the component count, retained-state count and state-generation work. The initial gap is $B_t = U_{0,t} - L_{0,t}$, where $U_{0,t}$ is the incumbent's optimistic cost and $L_{0,t}$ sums the component minima. The *tolerance lower bound* is

$$\tau_t = \alpha\omega\beta_t \sum_{r\in\mathcal{R}_t} \min_{g:(r,g)\in\mathcal{E}_t} \|\boldsymbol{\phi}_{rg,t}\|_{\boldsymbol{V}_t^{-1}}. \tag{43}$$

This quantity lower-bounds the stopping threshold of every feasible schedule. After a failed initial test, RLS starts doubling refinement at $\kappa = \nu_t$. Write κ_t for the chosen or last completed resolution. Within a call, optimistic costs and confidence widths stay fixed.

Proof of Theorem 2. **Completed solves bound the optimization error.** Fix a nonempty epoch t and omit its subscript. Lemma D.1 shows that the component representation preserves the optimistic optimum m^-. Thus the component minima and input incumbent satisfy

$$L_0 \le m^- \le U_0, \qquad B = U_0 - L_0 \ge 0.$$

If $B = 0$, the input incumbent and L_0 give zero optimization error and satisfy (6), even at zero tolerance.

For $B > 0$, recurrence (25) keeps the minimum exact reservation for each rounded excess-cost total at resolution κ. Backtracking therefore gives a feasible decision $\boldsymbol{a}_\kappa$. The lower bound in (26) and Appendix D.1 give

$$L_\kappa \le m^- \le J^-(\boldsymbol{a}_\kappa), \qquad J^-(\boldsymbol{a}_\kappa) - L_\kappa \le \frac{\nu B}{\kappa}.$$

Rounding changes only excess costs. Keeping reservations exact preserves feasibility in both the capacity test and the reconstructed schedule.

JAS retains the best feasible schedule and largest valid global lower bound found so far. Hence $J^-(\boldsymbol{a}^0) \le J^-(\boldsymbol{a}_\kappa)$ and $L^- \ge L_\kappa$, giving

$$0 \le J^-(\boldsymbol{a}^0) - m^- \le J^-(\boldsymbol{a}^0) - L^- \le J^-(\boldsymbol{a}_\kappa) - L_\kappa \le \frac{\nu B}{\kappa}.$$

The retained schedule and bound may come from different completed solves. Restoring epoch subscripts gives (10).

DP work scales with the retained states and resolution. Generate states once in $O(G_t^{\mathrm{g}})$ work, storing each state's minimum-cost assignments, cost, exact reservation and selected confidence-width sum. Recurrence (25) considers at most $\kappa_t + 1$ totals per retained state, requiring $O(N_t^{\mathrm{r}}(\kappa_t + 1))$ work. Backtracking selects one state per component, and their stored width sums give the stopping test in $O(\nu_t)$ work. Expanding the final assignments is covered by $O(G_t^{\mathrm{g}})$. Since each component retains at least one state, $\nu_t \le N_t^{\mathrm{r}}$, so these operations fit the same bound. The per-epoch computational complexity is therefore $O(G_t^{\mathrm{g}} + N_t^{\mathrm{r}}(\kappa_t + 1))$. Summing completed solves across epochs gives total computational complexity

$$O\left(\sum_{t \le T} [G_t^{\mathrm{g}} + N_t^{\mathrm{r}}(\kappa_t + 1)]\right).$$

Here κ_t is the chosen or last completed resolution. Under doubling, the geometric-sum argument below includes all earlier completed solves in the same bound, with states generated only once.

A positive tolerance lower bound ensures finite refinement. For a feasible decision $\boldsymbol{a}$, define its selected confidence-width sum as

$$w_t(\boldsymbol{a}) = \sum_{(r,g):x_{rg}=1} \|\boldsymbol{\phi}_{rg,t}\|_{\boldsymbol{V}_t^{-1}}.$$

Each task uses exactly one compatible option, so the tolerance lower bound in (43) satisfies

$$\alpha\omega\beta_t w_t(\boldsymbol{a}) \ge \alpha\omega\beta_t \sum_{r \in \mathcal{R}_t} \min_{g:(r,g)\in\mathcal{E}_t} \|\boldsymbol{\phi}_{rg,t}\|_{\boldsymbol{V}_t^{-1}} = \tau_t$$

for every feasible incumbent throughout refinement. The context matrix, quality estimates, component states, U_0, L_0 and B_t remain fixed during the call.

If the input incumbent and L_0 satisfy (6), no rounded solve is needed. Otherwise, for $\tau_t > 0$,

$$B_t > \alpha\omega\beta_t w_t(\boldsymbol{a}_t^0) \ge \tau_t.$$

At stage $j = 0, 1, \ldots$, use $\kappa_j = \nu_t 2^j$. The completed-solve bound gives

$$J_t^-(\boldsymbol{a}_t^0) - L_t^- \le \frac{\nu_t B_t}{\kappa_j} = B_t 2^{-j}.$$

Thus the stopping condition holds no later than stage

$$j_* = \left\lceil \log_2 \frac{B_t}{\tau_t} \right\rceil, \qquad B_t 2^{-j_*} \le \tau_t \le \alpha\omega\beta_t w_t(\boldsymbol{a}_t^0).$$

The incumbent's selected uncertainty may allow earlier stopping. The lower bound τ_t controls the worst-case number of stages.

Doubling adds only a constant factor to the last solve's work. Since $\kappa_j \geq 1$, the sum of the resolution ranges obeys

$$\sum_{j=0}^{j_*}(\kappa_j + 1) \leq 2\sum_{j=0}^{j_*}\kappa_j = 2\nu_t(2^{j_*+1} - 1) < 4\kappa_{j_*},$$

$$\kappa_{j_*} = \nu_t 2^{j_*} < 2\nu_t \frac{B_t}{\tau_t}.$$

Each stage's DP work covers its $O(\nu_t)$ stopping test. Including state generation, refinement after a failed initial test costs $O(G_t^{\mathrm{g}} + N_t^{\mathrm{r}}\nu_t B_t/\tau_t)$. Including successful initial tests and $B_t = 0$ gives

$$O\left(G_t^{\mathrm{g}} + N_t^{\mathrm{r}}\nu_t\left(1 + \frac{B_t}{\tau_t}\right)\right),$$

which proves (11). Empty task epochs require no component solve and contribute zero regret. □

Implications for decision quality. If a resolution κ_t completes at every nonempty epoch, then on the confidence event of Theorem 1, substituting (10) into (8) gives

$$R_T \leq \sum_{t \leq T} \frac{\nu_t B_t}{\kappa_t} + 2\omega\overline{\beta}_T \Xi_T. \tag{44}$$

Deadlines and zero tolerance. An interrupted solve keeps the retained feasible schedule and valid global lower bound, but the unfinished resolution gives no new accuracy guarantee. The finite work bound requires $\tau_t > 0$. For $\tau_t = 0$, stopping can still succeed when the incumbent and lower bound close the gap. Integer reservations also permit (27). These bounds count scheduling arithmetic operations, with optimistic costs and confidence widths given. They exclude quality estimation, model execution and feedback collection. A fixed tolerance $\epsilon > 0$ uses $\kappa = \max\{1, \lceil \nu B/\epsilon \rceil\}$ and $O(G^{\mathrm{g}} + N^{\mathrm{r}}(1 + \nu B/\epsilon))$ work. Quality-aware stopping adapts this accuracy to the remaining quality uncertainty.

Workload and precision. A smaller initial gap B or a larger tolerance lower bound τ reduces the complexity bound. Centering makes the bound invariant to common cost offsets within each component. For one nonempty epoch, omit t and let $M_{\mathrm{c}} = \max_i |\mathcal{R}_i|$. Appendix D.1 gives $N^{\mathrm{r}} \leq \nu(2^{K_{\mathrm{c}}} - 1)$ and $G^{\mathrm{g}} = O(\nu 2^{K_{\mathrm{c}}} K_{\mathrm{c}}(M_{\mathrm{c}} + 1))$ for result-subset enumeration, so (11) becomes

$$O\left(2^{K_{\mathrm{c}}}\nu\left[K_{\mathrm{c}}(M_{\mathrm{c}} + 1) + \nu\left(1 + \frac{B}{\tau}\right)\right]\right).$$

At a fixed resolution κ, Corollary D.1 gives work linear in ν for bounded $K_{\mathrm{c}}, M_{\mathrm{c}}$. Appendix D.2 describes the different generation cost of the assignment catalogues used in timing experiments.

Comparison with exact scheduling. RLS indexes rounded costs, while Exact DP indexes integer capacity. Under Assumption 2, completed solves have the following per-epoch complexity bounds.

Method	Arithmetic work or candidate count
Complete result-selection enumeration	$2^{\sum_i \lvert\mathcal{G}_i\rvert}$ candidates
Exact integer-capacity DP	$O(G^{\mathrm{g}} + N^{\mathrm{r}}(c^{\max} + 1))$
RLS, to (6) ($\tau > 0$)	$O(G^{\mathrm{g}} + N^{\mathrm{r}}\nu(1 + B/\tau))$

Exact DP uses the recurrence in Appendix D.2. After a failed initial test with $\tau > 0$, $\kappa = \lceil \nu B/\tau \rceil$ suffices, and doubling reaches the stopping condition at a resolution below $2\nu B/\tau$. Thus RLS can reduce DP work when K_{c} is small and $\nu B/\tau \ll c^{\max}$, unless state generation dominates. Small capacities or tight tolerances can favor Exact DP.

H Additional Experimental Evaluation

H.1 Experimental Settings

Data version and splits. Workloads use the published releases of QuALITY, QASPER, MultiRC and RACE cited in Section 6. The source and target models are Qwen3.5-4B and Qwen3.5-9B,

respectively. Each dataset is divided into disjoint development and confirmation pools in a 35:80 ratio. The default workload draws four documents per epoch and schedules their questions as tasks.

Routes. We use four routes: native, source-short, source-long and target-self. Native execution uses the 9B target to answer from the original document. Source-short and source-long use the 4B source to prepare concise and detailed summaries, respectively, from which the target answers. Target-self uses the 9B model for both summary preparation and answering. Summary length is five percent of the source document or 16 tokens, whichever is larger. SMU uses native execution and target-self. The five primary methods use all four routes.

Context construction. The context vector uses a scalar ξ from document length and question–document overlap, with two coordinates for each route. For character length L_{doc} and overlap $o \in [0,1]$, where o is the fraction of question tokens that appear in the document, let $\tilde{d}$ be the min–max normalization of $\log(1+L_{\mathrm{doc}})$ in the dataset pool, and set

$$\xi = \tfrac{1}{2}\min\{1, \max\{0, \tilde{d}\}\} + \tfrac{1}{2}(1-o). \tag{45}$$

For route k, the context has $\boldsymbol{\psi} = (1-\xi, \xi)$ in its two coordinates and zeros in the other route blocks. Thus $D_\phi = 8$ and $\|\boldsymbol{\phi}_{rg,t}\|_2 \le 1$.

Quality feedback and cost calibration. Quality losses are computed from the source and target LLM answers: $0/1$ error on QuALITY and RACE, $1-\mathrm{F1}$ on QASPER, and per-question F1 loss on MultiRC. The primary comparison uses delay $D=0$. Prefill and decode times per token are obtained from one calibration of the two models. They determine the modeled costs of native answers, summary preparation and answers using a summary.

Quality-aware cost and regret. We construct a semi-synthetic evaluation using the task contexts, computation costs, and resource constraints described above. For each route k, we fit $\boldsymbol{\theta}_k^{\mathrm{sim}} \in [0,1]^2$ by constrained least squares on the development-pool LLM losses and freeze it before confirmation experiments. The simulator defines $\mu_{rg,t} = \boldsymbol{\phi}_{rg,t}^\top \boldsymbol{\theta}^{\mathrm{sim}} \in [0,1]$. For every executed assignment, it draws a fresh observation $\ell^{\mathrm{o}}_{rg,t} \sim \mathrm{Bernoulli}(\mu_{rg,t})$, independently across executions conditional on the contexts and fixed simulator parameters. Only selected observations are revealed to the learner, with delays independent of unrevealed outcomes.

The evaluator uses $\mu_{rg,t}$ in (1) and computes the exact per-epoch optimum over the same feasible set. Regret is the cumulative difference between the chosen schedule's mean cost and this optimum, as in ($\mathbb{P}$); the per-task average divides this quantity by $N_T = \sum_{t=1}^{T} |\mathcal{R}_t|$. The simulator parameters and means are unavailable to the learner. This construction satisfies the linear quality model by design; it does not assume that actual LLM losses are exactly linear in the chosen features.

H.2 Computational Efficiency

H.2.1 Stopping Rule and Solver Diagnostics

Table H.1 reports ablations of individual GLS mechanisms through 400 epochs.

Table H.1: GLS mechanism ablations.

	QARS	No screening	No collective moves	Fixed tolerance	Zero tolerance	No greedy initialization
Regret	0.785	0.845	0.808	0.896	0.787	0.875
Time (ms)	1.770	1.786	2.551	1.469	4.291	1.337

Screening removes an average of 0.693 results per epoch. Collective moves reduce MILP calls from 0.199 to 0.094 per epoch. On the primary trajectories, the mean quality-loss interval width is 0.278, and 98.7% of decisions satisfy the stopping rule (6).

H.2.2 Resources and Scheduling Deadlines

Figure H.1 reports regret under capacity, workload-size, and scheduling-deadline sweeps. Capacities are $2, 3, 4, 5, 6$ and 8; the workload-size sweep varies document count and capacity jointly. Deadlines

are $1, 3, 10, 30, 100$ ms. Each point uses 400 epochs, with task streams paired across methods. QARS has the lowest mean regret in several settings. Figure 4 reports scheduling time for the same five methods and deadlines, with 400 epochs per dataset.

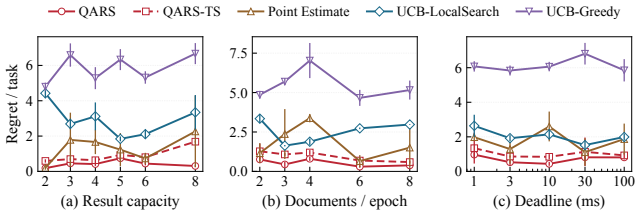


Figure H.1: Regret across resource and deadline settings for the five methods in Table 1.

H.2.3 Component Dispatch and Scaling

Figure 5 compares Bounded (RLS) and General (GLS) with Exact DP and MILP as the component count ν grows. All methods use fixed quality estimates and assignment catalogues. The comparison uses integer-component instances and a 50 ms scheduling budget, with ten instances at each $\nu \in \{24, 32, 40, 48, 64\}$. Reported times include catalogue construction and any budget overrun (Appendix D.2); MILP times also include model construction in SciPy/HiGHS. At $\nu = 24$, the times are 1.70 ms for Bounded, 3.49 ms for General, 5.38 ms for Exact DP and 9.91 ms for MILP.

H.2.4 Subset versus Assignment Catalogues

Figure H.2 compares subset and assignment catalogue generation (Corollary D.1). The scaling experiment above fixes $K_c = M_c = 4$. Here $\nu = 24$ and quality estimates are fixed. We vary M_c with $K_c = 4$, then K_c with $M_c = 4$. Times are medians over ten instances.

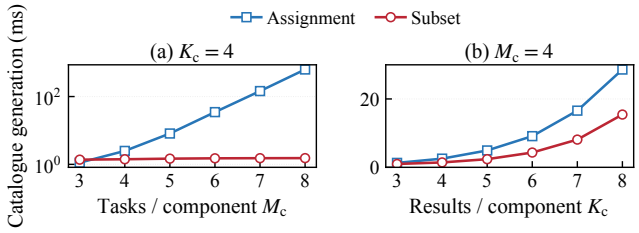


Figure H.2: Catalogue generation time. (a) $K_c = 4$, log scale. (b) $M_c = 4$, linear scale.

At $M_c = 8$, assignment generation takes 629 ms and subset generation takes 1.53 ms. At $K_c = 8$, the times are 28.6 ms and 15.4 ms. Both catalogues yield the same objective on all 120 instances. The subsequent dynamic program takes about 0.2 ms and reconstruction about 0.01 ms.

H.3 Quality Weight

Figure H.3 multiplies each dataset's default quality weight ω_0 by 0.25, 0.5, 1, 2 and 4. The five methods in Table 1 are evaluated at capacity 5 for 400 epochs. At $\omega = \omega_0$, QARS has the lowest mean regret (0.863). The factor ω in Theorem 1 reflects the weighting of quality loss in the objective.

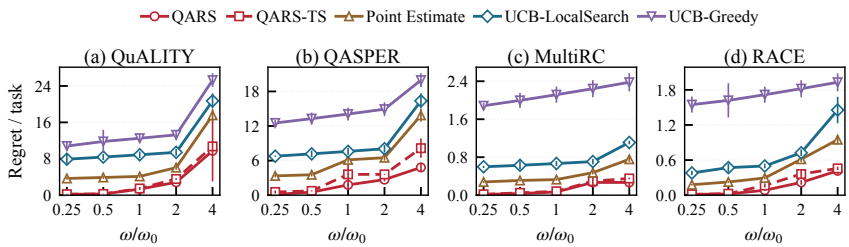


Figure H.3: Regret versus quality-weight multiplier. ω_0 is the dataset-specific default.