SOFT-INTERLEAVED THOMPSON SAMPLING FOR COMPUTATIONALLY EFFICIENT COMBINATORIAL BANDITS

Anonymous authors
Paper under double-blind review

ABSTRACT

Combinatorial Thompson Sampling (CTS) is a highly effective policy for sequential decision-making in structurally constrained environments. However, standard CTS relies on "black-box" optimization oracles, which is problematic from two aspects. First, computationally, it repeatedly executes exact offline oracles from scratch, which often incurs polynomial or exponential costs. Second, in problems with local optima, the oracle tends to prematurely commit too early to such optimum, and this highly confines the exploration in next rounds, blinding the policy from reaching the global optimum. To address these limitations, we propose *Soft-Interleaved Thompson Sampling* (SITS), a novel framework that abandons the black-box exact-oracle approach, and is based on extracting information from non-convergent, iterative "soft" solvers. By interleaving targeted queries directly within the intermediate steps of an iterative solver, our framework identifies and samples ambiguous topological boundaries before a final action is chosen. We demonstrate this approach with two canonical classes of problems, the Maximum Weight Matching problem with a message passing solver, and the Shortest Path problem with a Bellman-Ford solver. Empirical evaluations, including real-world biological sequence alignments, establish that SITS successfully solves problem instances in which CTS fails, while reducing the per-round computational overhead by an order of magnitude, empirically matching the computational scaling of a single offline exact solver.

1 INTRODUCTION

Sequential decision-making under uncertainty is classically modeled using the Multi-Armed Bandit (MAB) framework (Robbins, 1952; Slivkins, 2019; Lattimore & Szepesvári, 2020), whose main aspect is to capture the fundamental trade-off between exploration and exploitation. In traditional MAB settings, an agent repeatedly chooses from a set of actions (also called arms), with the goal of learning their unknown reward distributions over time, and maximizing the cumulative payoff. A prominent Bayesian approach that balances exploration and exploitation by sampling from posterior distributions is Thompson Sampling (TS) (Thompson, 1933);(Russo et al., 2018).

Many real-world scenarios require selecting a *combination* of actions, possibly subject to structural constraints. This is formalized as the Combinatorial Multi-Armed Bandit (CMAB) problem (Chen et al., 2013), encompassing applications from network routing and influence maximization to resource allocation and bipartite graph matching, under semi-bandit feedback (Lattimore & Szepesvári, 2020), where the agent observes the individual outcome of every queried base arm, rather than the aggregate reward of the overall structure.

Combinatorial Thompson Sampling (CTS) (Wang & Chen, 2018) extends TS to CMAB by sampling base-arm weights from the posterior, passing the sampled instance to a combinatorial optimization oracle (solver), and querying the resulting action to obtain semi-bandit feedback. This black-box oracle paradigm is theoretically well studied (Wang & Chen, 2018; Perrault et al., 2020), but introduces two major pitfalls.

First, as we discuss in detail throughout this paper, CTS operates poorly for a sub-class of CMAB instances, in which the environment is structurally constrained in a way that leads to a sub-optimal

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.

local optimum. In these instances, at early stages of exploration, when the posterior is not yet concentrated, the sample of environment's arm rewards is highly noisy. The black-box optimization oracle then returns a sub-optimal combinatorial structure, whose constituent arms are subsequently sampled and reinforced under the updated posterior. Escaping these local optima requires rare simultaneous, reward anomalies across multiple arms.

This phenomenon is a combinatorial manifestation of the fundamental shortcoming of TS identified by Russo & Van Roy (2014), in which posterior sampling is limited to actions with a non-zero probability of being strictly optimal. This precludes sampling *revealing actions* that yield high information gain regarding the global optimum. In combinatorial domains, this flaw is magnified, as the oracle blinds the algorithm to individual base arms that act as necessary structural pathways to discover the true optimal solution. Russo & Van Roy (2014) proposed a theoretically elegant alternative, called Information-Directed Sampling (IDS), which explicitly minimizes the ratio of squared expected regret to information gain. While IDS actively targets sampling to minimize ignoring relevant information, it is very computationally intensive. In CMAB settings, computing the information-ratio of the IDS policy requires calculating an exponentially large set of candidate solutions, and approximations are needed.

Second, vanilla CTS also suffers from a computational burden. Executing an *exact* combinatorial optimization oracle from scratch at every round is inefficient, particularly because minor posterior updates often leave the optimal structure unchanged. Several recent works have recognized this computational cost and proposed ways to reduce either the per-call complexity or the frequency of oracle queries, including approximation-based algorithms (Xu & Li, 2021), efficient answer-neighbor structures (Jourdan et al., 2021), and epoch schemes (Kim et al., 2025). While these approaches mitigate the computational burden, their primary objective is computational. The oracle is still treated as an isolated black-box used to produce an action for the next round, and reducing its frequency does not address the loss incurred when intermediate information is ignored.

The cost of repeated oracle calls becomes particularly important at *scale*, where even a *single* combinatorial optimization instance can be computationally demanding. In fact, recent work has developed specialized solvers and learning-augmented methods to accelerate large-scale assignment problems (Haller et al., 2020; Yavlovich et al., 2026), reflecting the substantial computational effort required by individual optimization solves. This motivates taking a different approach to sequential learning, in which rather than repeatedly invoking a solver from scratch, the computational progress made while solving one instance is retained and reused across interactions.

These limitations motivate *Soft-Interleaved Thompson Sampling* (SITS), our novel TS algorithm that addresses these issues. SITS interleaves posterior sampling with partially executed iterative optimization. Rather than waiting for the solver to converge, it extracts structural information from its intermediate state to guide base-arm queries, while retaining the solver state across rounds. This couples exploration with the optimization process and can simultaneously reduce solver workload and improve exploration in deceptive instances.

Specifically, we make the following contributions:

- **A novel policy:** We introduce SITS, which couples posterior sampling with intermediate states of iterative combinatorial solvers, using structural ambiguity to guide base-arm queries.
- **Solver-aware exploration:** We demonstrate conceptually and empirically that exposing intermediate solver structure can prevent premature combinatorial commitment and reduce simple regret on deceptive instances where standard CTS fails.
- **Computational efficiency:** We show that retaining solver state and interleaving optimization with exploration reduces solver workload, scaling equivalently to a single offline execution, while maintaining comparable statistical performance on benign instances.

2 Motivation: Hard Oracle Failures

Before detailing our framework, we establish why standard combinatorial algorithms fail in certain environments. We will use the Maximum Weight Matching (MWM) problem on a bipartite graph as our running example. In this problem, the nodes of a weighted graph are partitioned into two sets

of equal size. The goal is to match any node in one set to a single node at the other set. The edge between the nodes represents the reward associated with this choice, and the goal of the agent is to maximize the total sum of the selected edge weights. Standard optimization algorithms, like the Hungarian algorithm (Kuhn, 1955), were designed to solve MWM problems when all edge weights are known. When the edge weights are unknown, and can be queried by an agent, these exact algorithms are used as "black-box" oracles for exploration in CTS. The oracle takes a matrix of sampled beliefs regarding the weights, computes the exact global optimum, and outputs a single binary decision (an edge is either in the optimal matching or it is not). However, all intermediate information about second-best options or high-variance alternatives is discarded. The agent only sees the final output and remains "blind" to the underlying uncertainty of the graph. We next illustrate this in two basic examples.

2.1 3×3 EXAMPLE

Consider a 3×3 bipartite matching environment with current posterior means (μ) as follows:

$$\mu = \begin{bmatrix} 10 & 0 & -100 \\ -100 & 10 & 0 \\ 0 & -100 & 10 \end{bmatrix} \tag{1}$$

The diagonal edges are known with weight 10 ($\sigma = 0$), giving a matching of total weight 30; the three edges with weight -100 are known penalties ($\sigma = 0$); and the remaining three edges form an unexplored cycle with $\mu = 0$ and $\sigma = 5$. Suppose the true weights of the unexplored cycle are all 12, making it the global optimum with value 36.

For CTS to select an unexplored edge, its sample must place that edge in the optimal matching. The mixed matching containing one unexplored edge is forced to use a -100 edge and has expected value $0 - 100 + 10 = -90$, making it highly unlikely to beat the diagonal matching. The only remaining possibility is the full three-edge cycle, which is also unlikely to have a larger weight. Its sampled weight is distributed as $\mathcal{N}(0, 75)$, and thus exceeds the diagonal's value 30 with probability

$$\Pr[\mathcal{N}(0, 75) > 30] = \Pr[Z > 3.46] \approx 0.02\%.$$

Hence, CTS must simultaneously draw a highly optimistic sample for all three unexplored edges before the oracle exposes any of their value. While such an event can eventually occur in a 3×3 instance, the corresponding simultaneous exploration requirement becomes increasingly unlikely as the number of dependent edges grows.

2.2 50×50 EXAMPLE

To visually demonstrate this at scale, we consider a 50×50 MWM instance where diagonal entries all have weight 0.5, and there are three hidden 10-edge "highways" of weight 1. All other edges have weight 0. A "safe" choice is to match using only the diagonal, yielding a total weight of 25, whereas the global optimum requires three sacrificial edges and reaches 38.5 (illustrated in Figure 1 (Left)). As shown in Figure 1 (Right), standard CTS fails this case. Because it relies on the oracle to dictate its exploration, it doesn't accept a 0 edge unless the 10 unknown edges in the highway simultaneously draw high noise samples to offset the penalty.

To succeed in such scenarios, an algorithm must decouple its exploration strategy from the final output of the oracle. As we shall see, and as detailed for this instance in Appendix B, our proposed policy successfully escapes such local maxima.

3 METHOD: SOFT-INTERLEAVED THOMPSON SAMPLING

In this section, We introduce *Soft-Interleaved Thompson Sampling* (SITS), a framework that interleaves posterior sampling with partially executed combinatorial solvers. Rather than treating the solver as a black-box oracle that produces only a final action, SITS uses its intermediate state to identify structural ambiguity and guide future queries.

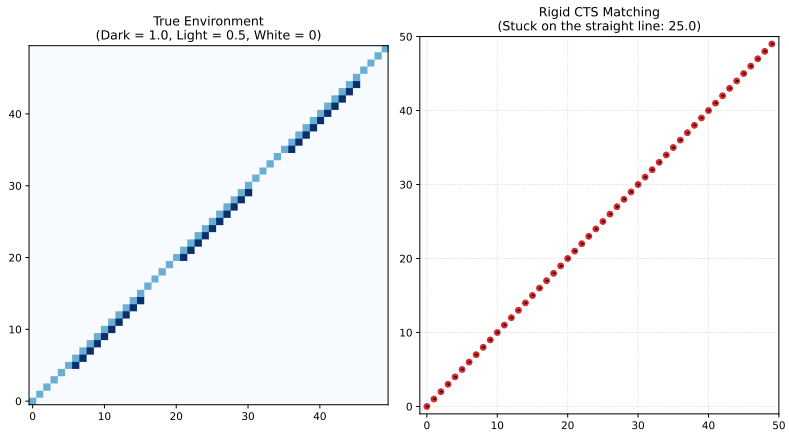


Figure 1: A 50×50 example. **Left:** The environment contains a safe diagonal (0.5), three high-value detours (1), and background edges (0). **Right:** Standard CTS remains on the sub-optimal diagonal because exposing the detours requires accepting locally unattractive edges.

3.1 Problem Formulation and Interaction Protocol

Let $\mathcal{E}$ be a finite set of base arms and $\mathcal{A} \subseteq 2^{\mathcal{E}}$ the feasible super-arms. Each arm e has unknown weight θ_e, and $f(A, \theta) = \sum_{e \in A} \theta_e$ is the reward for choosing the base arms in $\mathcal{A}$. The objective of the agent is to identify $A^* = \arg\max_{A \in \mathcal{A}} f(A, \theta)$. We assume the Bayesian setting, in which the agent has a prior regarding the weights $\{\theta_e\}_{e \in \mathcal{E}}$ At round t, the agent queries Q arms, receives noisy semi-bandit observations, $\{y_e\}_{e \in Q}$, and updates its posterior distribution on $\{\theta_e\}_{e \in \mathcal{E}}$. There are various ways to measure the regret of the agent compared to an agent that knows the weights in advance. For concreteness and simplicity, we focus on the *expected simple regret* $\Delta_t = \mathbb{E}[f(A^*, \theta) - f(\hat{A}_t, \theta)]$.

3.2 Standard CTS vs. SITS

Standard CTS treats the optimization process as a black-box oracle. Letting $\mathcal{P}^{(t-1)}$ denote the posterior distribution before the tth round, the agent first samples an environment $\hat{\theta}^{(t)} \sim \mathcal{P}^{(t-1)}$. Then, it solves the resulting combinatorial problem to completion $A_t = \arg\max_{A \in \mathcal{A}} f(A, \hat{\theta}^{(t)})$, and collects the reward. The semi-bandit feedback is then used to update the posterior to the next round $\mathcal{P}^{(t)}$. In principle, the solver is then reset from the perspective of the agent after each round, and only the final action is exposed ot the agent, while information contained in the optimization trajectory is discarded.

SITS instead solves the sampled problem in a *soft* manner. Concretely, this can be achieved by advancing an iterative solver for only a small number of steps. The intermediate state can then be used to reveal structural ambiguity before querying the environment. In particular, the solver can reveal competing edges, paths, or other combinatorial components that are close to one another according to the current sampled environment. These structural signals are combined with posterior uncertainty to determine which base arms should be queried.

The resulting interaction loop of SITS is thus

$$\text{sample} \rightarrow \text{soft solve} \rightarrow \text{score structural ambiguity} \rightarrow \text{query} \rightarrow \text{update},$$

compared to the standard CTS interaction loop

$$\text{sample} \rightarrow \text{solve to completion} \rightarrow \text{query} \rightarrow \text{update}.$$

3.3 Formalizing the Framework

Let $\mathcal{P}^{(t)}$ denote the posterior over base-arm weights $\{\theta_e\}_{e \in \mathcal{E}}$ at round t, and let $\mathcal{S}$ be an iterative combinatorial solver with persistent internal state. At each round, SITS draws a Thompson sample

$\hat{\theta}^{(t)} \sim \mathcal{P}^{(t-1)}$ and advances $\mathcal{S}$ for K steps using this sampled environment. Rather than waiting for convergence, we extract structural information $\mathcal{I}^{(t)}$ from the intermediate state, such as solver margins, competing decisions, or other measures of local combinatorial ambiguity.

We then compute an acquisition score

$$AS_e^{(t)} = \text{Score}\left(\mathcal{P}^{(t-1)}, \mathcal{I}^{(t)}\right),$$

which combines structural ambiguity with posterior uncertainty to identify the base arms whose observations are most useful for resolving the current combinatorial uncertainty. The Q highest-scoring arms, subject to problem-specific structural constraints, are queried, and their observations are incorporated into the posterior. Crucially, the solver state is retained across rounds rather than recomputed from scratch, allowing computational progress and structural information to accumulate throughout the interaction. A description of the SITS is listed in Algorithm 1.

Naturally, the acquisition score is domain-dependent: Section 4 instantiates it for MWM using solver margins and for Shortest Path (SP) using a query-by-committee (QBC) construction. Thus, SITS defines a general interleaving principle while allowing the structural signal exposed by the underlying iterative solver to be specialized to the combinatorial problem.

Algorithm 1 Soft-Interleaved Thompson Sampling

Require: Action space $\mathcal{A}$, prior $\mathcal{P}^{(0)}$, iterative solver $\mathcal{S}$, steps per round K, query budget Q

```
Initialize the persistent state of S
for t = 1, 2, ... do
    Sample θ̂^(t) ~ P^(t-1)
    Advance S for K steps using θ̂^(t)
    Extract structural information I^(t)
    Compute AS_e^(t) ← Score(P^(t-1), I^(t)) for e ∈ E
    Query Q_t, the chosen Q arms
    Observe {y_e : e ∈ Q_t}
    Update P^(t) ← Update(P^(t-1), {y_e}_{e∈Q_t})
end for
```

4 Application Domains

To demonstrate the applicability of the SITS framework, we apply it to two structurally distinct combinatorial problems: MWM solved via Message Passing (MP) and the SP problem solved via Bellman-Ford (BF).

4.1 Maximum Weight Matching

As previously discussed, MWM seeks a maximum-weight matching between two disjoint sets of N nodes. We assume that the unknown edge weights are random, and for simplicity of exposition that they are statistically independent. Thus, the posterior corresponds to an independent Gaussian posterior for each edge and can be updated it via Normal–Normal conjugacy when queried.

For the soft solver $\mathcal{S}$, we use the max-sum MP algorithm (Cheng et al., 2006). MP iteratively propagates continuous beliefs between nodes. From these beliefs, we extract the structural margin $\delta_{ij}^{(t)}$, defined as the difference between an edge's best and second-best alternatives, small margins indicate locally ambiguous assignments. We thus choose the acquisition score to be

$$AS_{ij}^{(t)} = w_m\left(1 - \tilde{\delta}_{ij}^{(t)}\right) + w_v\tilde{\tau}_{ij}^{2,(t)} + w_b\left(1 - |\tilde{M}_{ij}^{(t)}|\right) \tag{2}$$

where $\tau_{ij}^{2,(t)}$ is the posterior variance and the continuous belief magnitude is defined as the sum of incoming messages and the sampled weight: $M_{ij}^{(k)} = m_{T_i \to x_{ij}}^{(k)} + m_{B_j \to x_{ij}}^{(k)} + \hat{\theta}_{ij}$. Here, k is the current MP iteration index. The messages $m_{T_i \to x_{ij}}$ and $m_{B_j \to x_{ij}}$ represent the penalties applied to

edge (i, j) if the Top (row) node i or the Bottom (column) node j have better alternative connections in the graph. The tilde notation denotes standard min-max scaling to $[0, 1]$ computed across all edges at round t. The weights (w_m, w_v, w_b) control the relative contributions of each term. The three terms prioritize small structural margins, high posterior uncertainty, and beliefs near zero, respectively. For more details, see Algorithm 2 in Appendix C.

To respect the bipartite matching constraints during exploration, we construct the query set greedily from the acquisition scores. Specifically, we rank all N^2 edges in descending order (using `argpartition` to identify the top $K = cN$ candidates in $\mathcal{O}(N^2)$ time) and iterate through this ordering, adding an edge to the query set if neither of its endpoints has already been selected. We continue until N edges have been selected, yielding a valid perfect matching. Thus, in the MWM setting, $Q = N$, corresponding to the number of edges in a perfect matching.

Unlike CTS, which fully solves each posterior-sampled instance before every query, SITS advances MP for only a *bounded* number of iterations. Each partial MP iteration and posterior update costs $\mathcal{O}(N^2)$, while the solver state is *retained* across rounds rather than recomputed from scratch. Thus, over T bandit rounds, SITS requires $\mathcal{O}(TN^2)$ solver work, compared with $\mathcal{O}(TN^3)$ when a cubic-time exact oracle (like the Hungarian algorithm or full MP algorithm) is executed each round. Empirical computational comparisons and scaling results are given in Section 5.3 and Appendix D.

4.2 The Shortest Path Problem

Given a directed graph $G = (V, E)$ with unknown edge weights, the objective is to identify the minimum-weight path between a source s and destination d. We use BF as the soft solver (Bellman, 1958), see Algorithm 3 in Appendix C.

BF repeatedly relaxes the graph's edges to update tentative shortest-path distances. For the acquisition policy, we use a QBC approach. At each round, two independent weight vectors are sampled from the current posterior and the soft BF solver produces tentative optimal paths P_1 and P_2. Their disagreement defines the candidate boundary

$$\mathcal{D}_t = P_1 \Delta P_2.$$

The acquisition score is

$$AS_e^{(t)} = \begin{cases} \tau_{e,t}^2, & e \in \mathcal{D}_t, \\ 0, & \text{otherwise.} \end{cases} \tag{3}$$

The active learning protocol allocates a fixed query budget Q per round, which is scaled proportionally to the expected number of elements of the target structure. When the sampled paths disagree ($\mathcal{D}_t \neq \emptyset$), we sorts the edges within the disagreement boundary by their acquisition score and query the top Q most uncertain boundary edges. If the sampled paths agree ($\mathcal{D}_t = \emptyset$), we instead score the edges in P_1 by their posterior variance.

Standard CTS executes BF to convergence at every bandit round, requiring $|V|-1$ relaxation sweeps and hence the oracle has computational complexity of $\mathcal{O}(|V||E|)$ per round. SITS performs only $k \ll |V|$ sweeps, resulting a computational complexity of $\mathcal{O}(|E|)$ per round for constant k. Computational comparisons and scaling results are reported in Section 5.3 and Appendix E.

5 Experiments

We evaluate the SITS framework across both domains to empirically validate our resolution of the dual bottleneck. Specifically, we demonstrate that: (1) Statistically, our proposed versions of SITS succeeds in structurally constrained environments, and reduce simple regret to zero or near-zero in both synthetic and real-world topologies, where standard baselines plateau; and (2) computationally, our framework matches the baseline statistical efficiency in benign environments while reducing the computational workload by orders of magnitude.

All empirical evaluations are averaged over 10 independent random seeds, with shaded regions in all figures representing ± 1 standard deviation to confirm statistical consistency across initializations.

To validate that this failure mode is a flaw of hard oracles rather than a specific artifact of Bayesian sampling, we benchmark SITS against two gold-standard algorithms:

1. **Standard CTS** (Wang & Chen, 2018): The standard probabilistic baseline, utilizing Gaussian TS wrapped around an exact oracle.
2. **Standard CombUCB** (Chen et al., 2013): The standard deterministic Frequentist baseline, utilizing Upper Confidence Bounds ($\hat{\mu} \pm \alpha\sqrt{\ln t/N_e}$) wrapped around the exact oracle.

These baselines are conventionally studied under the cumulative-regret CMAB objective, here we evaluate their exact exploration mechanisms under Combinatorial Pure Exploration (CPE) (Wang & Zhu, 2022; Jourdan et al., 2021) objective using simple regret. Where the goal is to efficiently probe the environment in order to identify the true global optimum). This isolates the impact of the black-box oracle on premature structural commitment, showing that the oracle itself prevents boundary sampling. For a fair comparison, all hyperparameters (including the CombUCB exploration scale α) are tuned proportionally to the environmental noise.

5.1 Synthetic Combinatorial Experiments

We tested the algorithms under strict query budgets on two penalized environments. For MWM, we constructed a dense bipartite graph ($N = 200$, yielding $40,000$ base arms) featuring a topological bottleneck that separates a rewarding local optimum from the true global maximum. For SP, we constructed a "Hidden Highway" Directed Acyclic Graph (DAG) ($|V| = 502$) featuring a congested grid that masks an optimal highway (we use the DAG structure here purely to visually isolate the combinatorial trapping while showing the active–learning).

As shown in Figures 2 and 3, both standard baselines fail. Despite operating on different statistical principles, both CTS and CombUCB rely on black-box oracles to evaluate exploration queries. Both algorithms are blinded to adjacent boundary edges, causing them to plateau and suffer linear regret.

In contrast, our SITS queries the boundary edges and successfully reaches zero simple regret while reducing the per-round computational overhead by an order of magnitude.

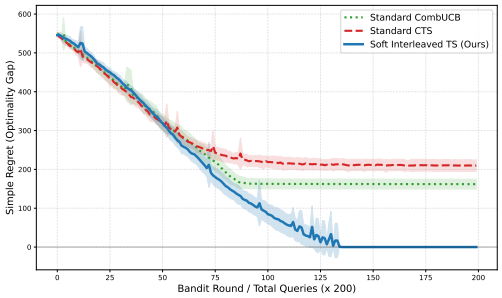


Figure 2: MWM Regret ($N = 200$). Both CombUCB (green) and CTS (red) plateau. SITS (blue) reaches zero simple regret.



Figure 3: SP Regret ($|V| = 502$). CombUCB (green) and CTS (red) both plateau. SITS (blue) targets boundaries to find the true path.

5.2 Real-World Applicability: SARS-CoV-2 RNA Alignment

We exemplify SITS on a real-world bioinformatics dataset. While *de novo* sequence assembly is often modeled via SP or dynamic programming (Needleman & Wunsch, 1970; Pevzner et al., 2001), reference-based shotgun sequencing and read mapping are abstracted as MWM problems (Mohajer et al., 2013; Lo et al., 2013; Mandric & Zelikovsky, 2015). The objective is thus to uniquely assign a set of independent, overlapping sequence reads to candidate genomic loci without violating the strict one-to-one mapping constraint. We aligned the official SARS-CoV-2 reference genome (Wu et al., 2020) against 500 candidate loci with a 10% mutation/error rate characteristic of high-divergence viral variants and long-read sequencing technologies.

Because viral RNA sequences contain conserved domains, this matrix is filled with correlated structural traps. As shown in Figure 4, this biological complexity punishes both standard black-box baselines. Due to relying on black-box oracles, both CTS and CombUCB lock onto sub-optimal alignments early in the process, with their simple regret plateauing at ~ 86 and ~ 125, respectively.

SITS identifies ambiguous sequence overlaps using its continuous acquisition scores and discovers the true genomic alignments (gap: 1.2).

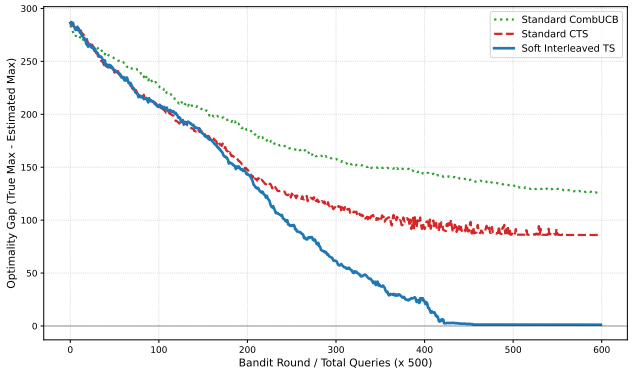


Figure 4: Performance on Real-World SARS-CoV-2 RNA Alignment ($N = 500$). The hard-oracle baselines (CombUCB and CTS) fail while SITS discovers the true genomic alignments.

5.3 Computational Efficiency and Statistical Parity

We next evaluate whether SITS can reduce solver computation without sacrificing statistical performance in benign, unstructured environments favorable to CTS. We consider dense MWM instances ($N = 150$) and SP instances ($|V| \approx 300$).

Figure 5 shows that SITS and CTS achieve comparable simple regret, reaching zero regret at similar rates, for both the MWM and SP problems. Crucially, SITS reduces solver workload by interleaving queries with partial solver execution. For both problems, the cumulative number of solver sweeps decreases from approximately 10^5 for CTS to 10^3 for SITS, yielding a two-order-of-magnitude reduction in the workload without a corresponding loss in regret.

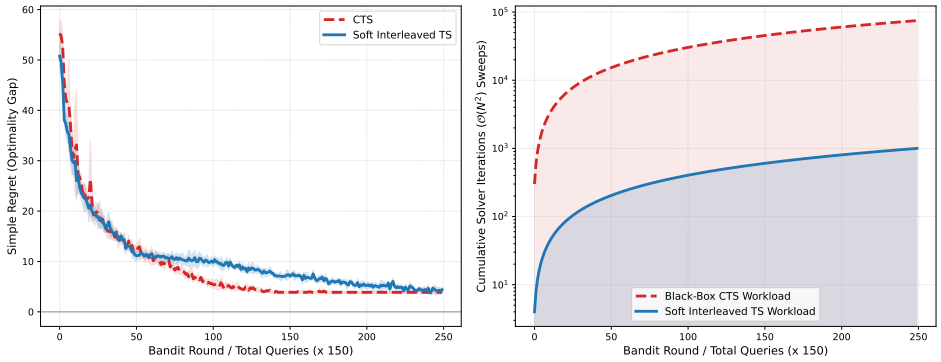
5.4 Mechanisms of Success and Empirical Scaling

Additional experiments (detailed in Appendices D and E) examine the mechanisms of SITS and its scaling behavior. Spatial query heatmaps show that SITS concentrates queries near high-affinity boundaries in block-structured environments without explicit topological information. Ablations further show that combining posterior variance with structural margins is important for avoiding regret plateaus, while sensitivity experiments demonstrate robustness to observation noise and hyperparameter misspecification. Finally, the number of solver iterations grows linearly with graph size in both domains, consistent with the computational scaling of a single offline solver execution.

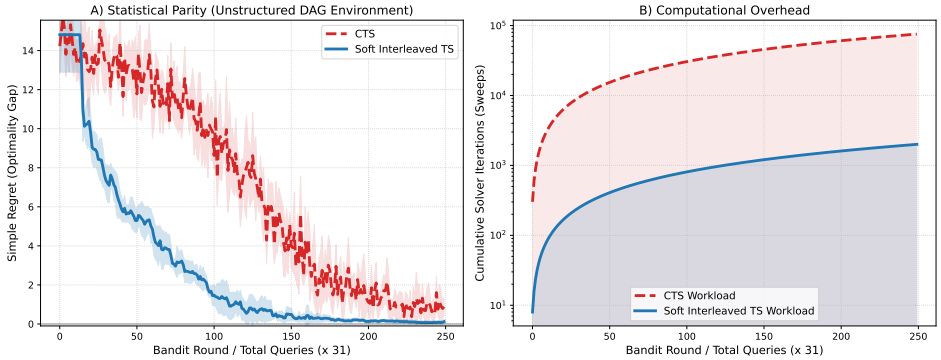
6 Related Work

CTS provides an established posterior-sampling baseline for stochastic CMABs with access to a combinatorial optimization oracle. Several recent works address the computational cost of this oracle through approximation (Xu & Li, 2021), efficient answer-neighbor structures (Jourdan et al., 2021), and epoch-based oracle queries (Kim et al., 2025). These approaches reduce oracle computation or frequency while retaining the conventional separation between the sampling policy and optimization oracle. In contrast, SITS exposes and reuses the solver's intermediate state, making it part of the exploration mechanism.

Additionally, CPE methods such as CLUCB (Chen et al., 2013) and TS-Explore (Wang & Zhu, 2022) focus on selecting informative base arms directly, typically assuming the agent can reason over candidate answers without exploiting the internal trajectory of the solver. Our focus is complementary, as we study how to make posterior sampling computationally efficient when the underlying combinatorial optimization problem is expensive. SITS retains the combinatorial solver in the loop, using its intermediate state to determine which base arms to query. While these methods provide important conceptual CPE baselines, they are not direct comparisons for our solver-aware setting.



(a) MWM, $N = 150$.



(b) SP, $|V| \approx 300$.

Figure 5: Computational efficiency and statistical parity in unstructured environments. In both MWM and SP, SITS achieves comparable simple regret to CTS while reducing cumulative solver workload by approximately two orders of magnitude ($\sim 10^5$ to $\sim 10^3$ sweeps).

Our approach is also related to methods that use disagreement or structural uncertainty for active sampling, including Joint Entropy Search in Bayesian Optimization (Hvarfner et al., 2022) and scalable TS via optimal transport (Zhang et al., 2019). Unlike these methods, SITS directly couples posterior sampling to the intermediate computational state of an iterative combinatorial solver.

7 Conclusion and Limitations

We introduced SITS, a framework that interleaves posterior sampling with partially executed combinatorial solvers. By exposing intermediate solver states rather than treating the oracle as a black box, SITS uses structural ambiguity to guide exploration while retaining computational progress across rounds. Experiments on MWM and SP show that SITS solves deceptive combinatorial structures where CTS fails while substantially reducing solver workload across the evaluated environments.

The main limitation of SITS is its problem dependence. Unlike standard CTS, which uses a generic combinatorial oracle, SITS requires an iterative solver whose intermediate state exposes a useful structural signal, together with a problem-specific acquisition score. However, as we demonstrated this through MP margins for MWM and QBC disagreement for SP, such solvers are common and can be utilized for SITS.

Our experiments provide empirical evidence of convergence on the tested synthetic and real-world instances, but finite-time theoretical guarantees for SITS remain open. These would also be problem specific. Such analysis must account for the coupled stochastic dynamics of posterior updates and the persistent state of the iterative solver. Extending the framework to additional combinatorial domains and developing corresponding theoretical guarantees are natural directions for future work.

AI USE STATEMENT

In this work, we used generative AI tools for document formatting, LaTeX typesetting, and text condensation to meet page limits. We have not used generative AI tools for conceptualizing the core theoretical framework, writing the experimental codebase, or generating empirical data. We have reviewed all AI-assisted work. We take responsibility for the final content of this work, including text, claims, or artifacts produced with the aid of generative AI.

ETHICS STATEMENT

Our research focuses on foundational improvements to combinatorial optimization algorithms. The empirical evaluations rely strictly on synthetic, algorithmically generated graphs and publicly available, open-source biological data (the official SARS-CoV-2 reference genome). This work does not involve human subjects, sensitive personal information, or proprietary datasets that raise privacy, security, or bias concerns. We foresee no immediate harmful societal impacts from the methodological improvements proposed in this paper.

REFERENCES

Richard Bellman. On a routing problem. *Quarterly of Applied Mathematics*, 16(1):87–90, 1958.

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

Yuan-sheng Cheng, Michael Neely, and Keith M Chugg. Iterative message passing algorithm for bipartite maximum weighted matching. In *IEEE International Symposium on Information Theory*, 2006.

Stefan Haller, Mangal Prakash, Lisa Hutschenreiter, Tobias Pietzsch, Carsten Rother, Florian Jug, Paul Swoboda, and Bogdan Savchynskyy. A primal-dual solver for large-scale tracking-by-assignment. In *International Conference on Artificial Intelligence and Statistics*, 2020.

Carl Hvarfner, Frank Hutter, and Luigi Nardi. Joint entropy search for maximally-informed Bayesian optimization. In *Advances in Neural Information Processing Systems*, 2022.

Marc Jourdan, Mojmír Mutnỳ, Johannes Kirschner, and Andreas Krause. Efficient pure exploration for combinatorial bandits with semi-bandit feedback. In *Algorithmic Learning Theory*, 2021.

Jung-hun Kim, Milan Vojnović, and Min-hwan Oh. Oracle-efficient combinatorial semi-bandits. In *Advances in Neural Information Processing Systems*, 2025.

Harold W Kuhn. The Hungarian method for the assignment problem. *Naval Research Logistics Quarterly*, 2(1–2):83–97, 1955.

Tor Lattimore and Csaba Szepesvári. *Bandit algorithms*. Cambridge University Press, 2020.

Christine Lo, Sangwoo Kim, Shay Zakov, and Vineet Bafna. Evaluating genome architecture of a complex region via generalized bipartite matching. *BMC Bioinformatics*, 14:S13, 2013.

Igor Mandric and Alex Zelikovsky. ScaffMatch: Scaffolding algorithm based on maximum weight matching. In *Research in Computational Molecular Biology*, 2015.

Soheil Mohajer, Abolfazl Motahari, and David Tse. Reference-based DNA shotgun sequencing: Information theoretic limits. In *IEEE International Symposium on Information Theory*, 2013.

Saul B Needleman and Christian D Wunsch. A general method applicable to the search for similarities in the amino acid sequence of two proteins. *Journal of Molecular Biology*, 48(3):443–453, 1970.

Pierre Perrault, Etienne Boursier, Michal Valko, and Vianney Perchet. Statistical efficiency of Thompson sampling for combinatorial semi-bandits. In *Advances in Neural Information Processing Systems*, 2020.

Pavel A Pevzner, Haixu Tang, and Michael S Waterman. An Eulerian path approach to DNA fragment assembly. *Proceedings of the National Academy of Sciences*, 98(17):9748–9753, 2001.

Herbert Robbins. Some aspects of the sequential design of experiments. *Bulletin of the American Mathematical Society*, 1952.

Daniel Russo and Benjamin Van Roy. Learning to optimize via information-directed sampling. In *Advances in Neural Information Processing Systems*, 2014.

Daniel Russo, Benjamin Van Roy, Abbas Kazerouni, Ian Osband, and Zheng Wen. A tutorial on Thompson sampling. *Foundations and Trends in Machine Learning*, 11(1):1–99, 2018.

Aleksandrs Slivkins. Introduction to multi-armed bandits. *Foundations and Trends in Machine Learning*, 12(1–2):1–286, 2019.

William R Thompson. On the likelihood that one unknown probability exceeds another in view of the evidence of two samples. *Biometrika*, 25(3–4):285–294, 1933.

Siwei Wang and Wei Chen. Thompson sampling for combinatorial semi-bandits. In *International Conference on Machine Learning*, 2018.

Siwei Wang and Jun Zhu. Thompson sampling for (combinatorial) pure exploration. In *International Conference on Machine Learning*, pp. 23470–23483, 2022.

Fan Wu, Su Zhao, Bin Yu, Yan-Mei Chen, Wen Wang, Zhi-Gang Song, Yi Hu, Zhao-Wu Tao, Jun-Hua Tian, Yuan-Yuan Pei, Ming-Li Yuan, Yu-Ling Zhang, Fa-Hui Dai, Yi Liu, Qi-Min Wang, Jiao-Jiao Zheng, Lin Xu, Edward C Holmes, and Yong-Zhen Zhang. A new coronavirus associated with human respiratory disease in China. *Nature*, 579(7798):265–269, 2020.

Haike Xu and Jian Li. Simple combinatorial algorithms for combinatorial bandits: Corruptions and approximations. In *Uncertainty in Artificial Intelligence*, pp. 1444–1454, 2021.

Ilay Yavlovich, Jad Agbaria, Muhamed Mhamed, Nir Weinberger, and Jose Yallouz. Learning-augmented scalable Linear Assignment Problem optimization via neural dual warm-starts. In *International Conference on Machine Learning*, 2026.

Ruiyi Zhang, Zheng Wen, Changyou Chen, Chen Fang, Tong Yu, and Lawrence Carin. Scalable Thompson sampling via optimal transport. In *Proceedings of the Twenty-Second International Conference on Artificial Intelligence and Statistics*, 2019.

The appendices are structured as follows:

- **Appendix A:** Provides the complete hyperparameters and experimental details for all evaluated environments, graph topologies, Bayesian priors, observation noise, and specific solver configurations.
- **Appendix B:** Provides an empirical resolution of the 50×50 example via SITS.
- **Appendix C:** Details the full pseudocode and solver mechanics for the specific versions of the SITS algorithms used for Maximum Weight Matching (MWM) and Shortest Path (SP).
- **Appendix D:** Provides extended empirical evaluations for the MWM problem, including ablation studies, heatmaps, and asymptotic scaling.
- **Appendix E:** Provides extended empirical evaluations for the SP problem, including topological query distributions and asymptotic scaling.

A Hyperparameters and Experimental Details

All experiments were implemented in Python. To ensure evaluation, performance metrics such as expected simple regret are reported as the mean across multiple independent runs with distinct random seeds, and shaded regions in all figures indicate ± 1 standard deviation. The environment-specific hyperparameters and active learning parameters are detailed below.

Maximum Weight Matching:

- **Graph Topology:** Bipartite graphs with varying sizes N.
- **Observation Noise:** Edges return noisy Gaussian semi-bandit feedback with standard deviation $\sigma_{\text{obs}} = 0.15$.
- **Bayesian Priors:** Independent Gaussian priors initialized to $\mu_0 = 0.5$ and $\tau_0^2 = 1.0$.
- **Query Budget:** Scaled linearly to match the structural cardinality of a perfect matching, $Q = N$ queries per round.
- **Solver Sweeps (k_{mp}):** SITS stops the solver every $k_{\text{mp}} = 4$ sweeps per round.
- **Message Passing Dynamics:** To prevent oscillatory behavior in dense bipartite structures, a momentum-based damping factor of $\rho = 0.5$ was applied to all message updates.
- **Acquisition Weights:** For the combined structural acquisition score, the relative weights were parameterized as $w_m = 1.0$ (margin priority), $w_v = 1.0$ (variance priority), and $w_b = 0.2$ (belief magnitude priority).

Shortest Path:

- **Graph Topology:** Layered directed acyclic graphs (DAGs) consisting of changing intermediate layers (experiment dependent) with 20 nodes per layer, plus a single source s and destination d.
- **Observation Noise:** Gaussian semi-bandit feedback with standard deviation $\sigma_{\text{obs}} = 0.15$.
- **Bayesian Priors:** Gaussian priors initialized with variance $\tau_0^2 = 0.5$.
- **Query Budget:** $Q = L$, where L is the number of edges in the source-destination path.
- **Solver Sweeps (k_{bf}):** SITS maintains a continuous state updated with $k_{\text{bf}} = 2$ sweeps per round.

B Extended Motivating Example - 50×50

We evaluate the SITS framework's ability to solve the example introduced in Section 2. As visualized in Figure 6, our algorithm successfully overcomes the limitations of the exact oracle. By actively allocating its query budget to structural boundaries, SITS absorbs the local 0 sacrifice penalty to unlock detours. It successfully reconstructs all three hidden highways, avoiding the sub-optimal 25 diagonal to correctly achieve the true global optimum matching of 38.5.

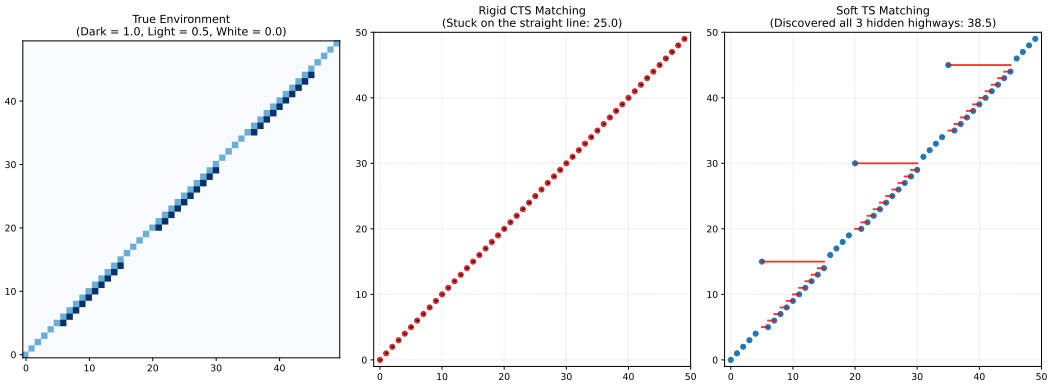


Figure 6: Solving the 50×50 example. Standard CTS remains trapped on the sub-optimal safe diagonal. SITS actively targets structural ambiguity, sacrificing local reward to discover all three hidden highways.

C Application Domain Algorithms & Solver Mechanics

This appendix provides the complete algorithmic details of the SITS implementations evaluated in our experiments. Algorithm 2 gives the full procedure for SITS in the MWM domain, using the message-passing (MP) solver, while Algorithm 3 gives the corresponding procedure for SP, using the Bellman–Ford (BF) solver. We then describe the solver mechanics and structural acquisition procedures used in each domain. We first present the Gaussian Bayesian posterior updates shared by both domains, followed by the MWM-specific MP updates, $\mathcal{O}(N^2)$ two-best optimization, and continuous margin extraction. For SP, we describe the partial BF updates, path extraction, and Query-by-Committee (QBC) disagreement used to identify structurally ambiguous edges.

Algorithm 2 SITS for Maximum Weight Matching

Require: N, initial priors $\mu_{ij}^{(0)}, \tau_{ij}^{2,(0)}$, observation noise σ_{obs}, MP damping ρ.

Initialize: For all (i, j), sample $\hat{\theta}_{ij} \sim \mathcal{N}(\mu_{ij}^{(0)}, \tau_{ij}^{2,(0)})$.
Initialize: Set all internal solver messages $m^{(0)} \leftarrow 0$.
for iteration $t = 1, 2, \ldots$ **do**
  # Continuous Solving
  for $\ell = 1, \ldots, k_{\text{mp}}$ **do**
    Update messages via equation 6 and equation 7 using $\mathcal{O}(N^2)$ two-best trick.
    Add damping: $m \leftarrow (1 - \rho)m + \rho\, m_{\text{new}}$.
  end for
  Compute beliefs $M_{ij}^{(k)}$ using current $\hat{\theta}$.
  Extract margins δ_{ij} to quantify structural ambiguity.
  # Targeted Querying (Structural Acquisition)
  Compute Acquisition Scores $AS_{ij}^{(t)}$ via equation 2 and decode tentative matching.
  Select query set $\mathcal{Q}_t$ that maximizes $AS_{ij}^{(t)}$.
  for each $(i, j) \in \mathcal{Q}_t$ **do**
    Observe $y_{ij} \sim \mathcal{N}(\theta_{ij}, \sigma_{\text{obs}}^2)$.
    Update posterior parameters (μ_{ij}, τ_{ij}^2) via equation 4 and equation 5.
  end for
  Resample all base arm weights $\hat{\theta}_{ij} \sim \mathcal{N}(\mu_{ij}, \tau_{ij}^2)$.
  if matching remains unchanged for T_{stable} iterations **then**
    break
  end if
end for

Algorithm 3 SITS for Shortest Path

Require: Graph $G = (V, E)$, source s, destination d, budget Q, priors $\mu_{uv}^{(0)}, \tau_{uv}^{2,(0)}$.
Initialize State: $D[s] \leftarrow 0$, and $D[v] \leftarrow \infty$ for all $v \neq s$.
for iteration $t = 1, 2, \ldots$ **do**
  Continuous Solving
  Sample primary edge weights $\hat{\theta}_{uv} \sim \mathcal{N}(\mu_{uv}^{(t-1)}, \tau_{uv}^{2,(t-1)})$.
  for $\ell = 1, \ldots, k_{\text{bf}}$ **do**
    for each edge $(u, v) \in E$ **do**
      if $D[u] + \hat{\theta}_{uv} < D[v]$ **then**
        $D[v] \leftarrow D[u] + \hat{\theta}_{uv}$
      end if
    end for
  end for
  # Targeted Querying (Structural Acquisition)
  Sample independent committee weights $\hat{\theta}^{(1)}, \hat{\theta}^{(2)} \sim \mathcal{N}(\mu^{(t-1)}, \tau^{2,(t-1)})$.
  Branch continuous state D using $\hat{\theta}^{(1)}$ and $\hat{\theta}^{(2)}$ to decode tentative optimal paths P_1 and P_2
  Compute the structural boundary via symmetric difference: $\mathcal{D}_t = P_1 \Delta P_2$.
  Compute Acquisition Scores $AS_e^{(t)}$ via equation 3.
  Select query set $\mathcal{Q}_t$ containing the top Q scoring edges.
  for each $(u, v) \in \mathcal{Q}_t$ **do**
    Observe semi-bandit traversal time $y_{uv} \sim \mathcal{N}(\theta_{uv}, \sigma_{\text{obs}}^2)$.
    Update posterior parameters $(\mu_{uv}^{(t)}, \tau_{uv}^{2,(t)})$ via Bayesian conjugacy.
  end for
  if primary path remains unchanged for T_{stable} iterations **then**
    break
  end if
end for

C.1 Detailed Solver Mechanics

Posterior Updates. For both MWM and SP, we maintain an independent Gaussian posterior over each base-arm weight. When querying base arm e and observing y_e, the posterior is updated by standard Gaussian conjugacy:

$$\tau_e^{2,(t+1)} = \left(\frac{1}{\tau_e^{2,(t)}} + \frac{1}{\sigma_{\text{obs}}^2} \right)^{-1}, \tag{4}$$

$$\mu_e^{(t+1)} = \tau_e^{2,(t+1)} \left(\frac{\mu_e^{(t)}}{\tau_e^{2,(t)}} + \frac{y_e}{\sigma_{\text{obs}}^2} \right). \tag{5}$$

Here, $\mu_e^{(t)}$ and $\tau_e^{2,(t)}$ denote the posterior mean and variance before round t, and σ_{obs}^2 is the observation noise variance. At each round, Thompson samples are drawn independently from these posteriors and sent to the corresponding combinatorial solver.

C.1.1 Maximum Weight Matching

Message-Passing Updates and Intuition. We use the iterative max-sum MP algorithm of Cheng et al. (2006) to solve the sampled MWM instance. The key idea is to allow each edge to assess its competitiveness relative to alternative edges incident to the same nodes. Messages are passed between the row nodes T_i, column nodes B_j, and their incident edges. A message from a node to an edge therefore summarizes the best alternative available to that node if the edge were not selected. Large alternative values make an edge less attractive, while an edge supported by both endpoints can accumulate a strong belief. Repeated message updates propagate this local competition through the graph, gradually producing beliefs that reflect the global matching structure.

Let k denote the current solver iteration index. The normalized updates for the max-sum MP algorithm use the current sampled weights $\hat{\theta}_{ij}$:

$$m^{(k)}_{T_i \to x_{ij}} = -\max_{m \neq j} \{ m^{(k-1)}_{B_m \to x_{im}} + \hat{\theta}_{im} \} \tag{6}$$

$$m^{(k)}_{B_j \to x_{ij}} = -\max_{l \neq i} \{ m^{(k-1)}_{T_l \to x_{lj}} + \hat{\theta}_{lj} \} \tag{7}$$

Naively, each edge would need to inspect all competing edges incident to its endpoint. However, all edges in a given row share the same collection of alternatives, so these maxima can be reused across edges. For a fixed row i, we compute

$$arr_i[m] = m^{(k-1)}_{B_m \to x_{im}} + \hat{\theta}_{im}$$

once in $\mathcal{O}(N)$ time and retain its largest two values, v_1 and v_2. For edge (i, j), the maximum excluding j is then v_2 if j achieves v_1, and v_1 otherwise. The same procedure is applied to columns, reducing the cost of a complete MP sweep from $\mathcal{O}(N^3)$ to $\mathcal{O}(N^2)$.

MWM Margin Extraction. The MP beliefs M_{ij} provide a continuous measure of how strongly each edge is supported by the current solver state. We convert these beliefs into structural margins that measure how much an edge is preferred over its alternatives. For a given row i, let $j^* = \arg\max_j M_{ij}$. The row margin is defined as the gap between the best and second-best belief: $\Delta_i^{\text{row}} = M_{i,j^*} - \max_{j \neq j^*} M_{i,j}$. The final per-edge margin is evaluated as $\delta_{ij} = \min(\Delta_i^{\text{row}}, \Delta_j^{\text{col}})$.

Greedy Decoding. The continuous beliefs do not themselves guarantee a valid matching. To obtain a feasible solution we sort edges by $|M_{ij}|$ and greedily add an edge whenever both endpoints remain unmatched. This produces a valid matching in $\mathcal{O}(N^2)$ time.

Damping. Because dense bipartite graphs contain many short cycles, max-sum MP can oscillate rather than converge monotonically. We therefore apply momentum-based damping with $\rho \in (0, 1)$ to the message updates. This stabilizes the intermediate solver state and makes the extracted margins more reliable during early, nonconverged iterations.

C.1.2 Shortest Path

Bellman–Ford Solver and Intuition. For SP, we use the BF algorithm as the iterative solver. BF maintains, for each node v, a current estimate $D[v]$ of the shortest distance from the source and a predecessor $P[v]$ identifying the edge currently responsible for that estimate. A sweep relaxes every edge, if reaching v through an incoming edge (u, v) gives a shorter distance, the distance and predecessor are updated. Thus, each sweep propagates SP information through another layer of the graph. After sufficient sweeps, the predecessor structure defines the SP. Under SITS, we stop after only k_{bf} sweeps and store the resulting state across rounds.

Query-by-Committee and Path Extraction. During structural acquisition, SITS draws two independent Thompson samples from the current posterior and runs the soft BF solver on both sampled weight vectors for k_{bf} sweeps. Each solver state produces a tentative predecessor structure, which is traced backward from the destination d to the source s to obtain paths P_1 and P_2. Their symmetric difference

$$D_t = P_1 \triangle P_2$$

identifies edges on which the two sampled environments disagree. Edges outside this set correspond to stable portions of the current solution and are therefore not prioritized for exploration. Within the disagreement boundary, we rank edges using posterior uncertainty and allocate the query budget accordingly.

Handling Nonconverged Paths. An edge case occurs during initialization. If the SP has L hops but only $k_{\text{bf}} \ll L$ sweeps are performed per round, the destination may initially have $D[d] = \infty$, resulting an incomplete predecessor chain. Because the BF state is stored across rounds, the reachable frontier expands by approximately k_{bf} hops per round. During this initialization phase, the extracted paths are empty and the QBC disagreement is empty, so no active-learning queries are issued. Once the frontier reaches d, the extracted paths become available and QBC begins directing queries toward their disagreement boundary.

D Extended Experiments: Maximum Weight Matching

D.1 Ablation Study

We conducted an ablation study on the bottleneck environment defined in Section 5.1, with $N = 100$, to understand which mechanisms cause the framework to succeed. We evaluated our full combined heuristic against two variants: a *Pure Variance* policy ($w_m = 0, w_v = 1$) and a *Pure Structural Margin* policy ($w_m = 1, w_v = 0$).

As demonstrated in Figure 7, isolating either mechanism results in failure. The *Pure Variance* variant wastes its budget uniformly exploring irrelevant background noise. The *Pure Structural Margin* variant fails due to getting trapped endlessly refining the structurally ambiguous local neighborhood. Our full combined framework synthesizes both components to allocate its budget to the exact edges required to find the global optimum.

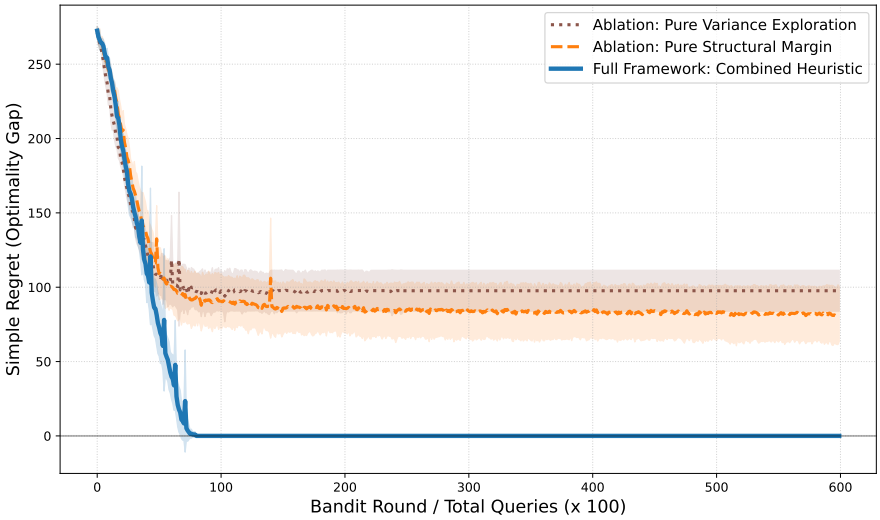


Figure 7: Ablation Study on the $N = 100$ deceptive topology. Only the Full Combined Framework (blue solid) successfully reaches zero regret.

D.2 Ablation Study: The Necessity of Structural Acquisition

Next, we tested whether an agent can achieve the same results by simply truncating the solver early and playing the resulting greedy matching.

We evaluated multiple solver configurations on the bottleneck environment with $N = 300$:

1. **CTS:** Runs MP to convergence and selects the discrete matching.
2. **Truncated CTS (Greedy):** Runs MP for a fixed number of times ($K \in \{10, 50, 100\}$), stopping the solver and extracting queries based purely on the greedy discrete matching of the stopped state, ignoring the structural margins.
3. **SITS:** Runs MP a small number of times ($K = 4$), extracts queries using the Structural Acquisition Score defined in Section 4.1.

As shown in Figure 8, truncating the solver without using the acquisition policy causes failures regardless of the truncation threshold. Truncated CTS reaches results similar to the Exact CTS baseline. This happens because the MP solver identifies the bottleneck boundaries fairly quickly. The greedy extraction of the stopped state already mirrors the fully converged sub-optimal solution.

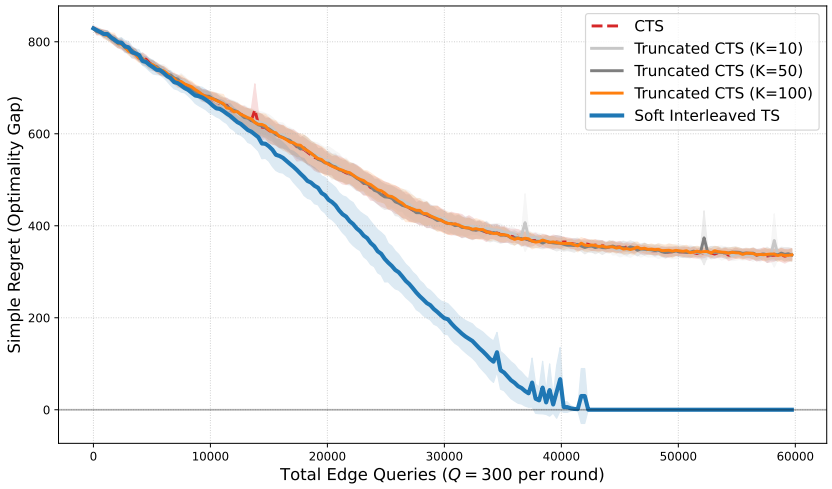


Figure 8: Ablation Study on the $N = 300$ bottleneck environment. Stopping the solver and greedily solving (Truncated CTS at $K = 10, 50, 100$) results in either inefficient exploration or convergence to a sub-optimal solution. SITS extracts the structural margins, allowing it to find the optimal solution.

D.3 Sensitivity Analysis: Priors and Observation Noise

To ensure the SITS framework is robust to hyperparameter misspecification and environmental stochasticity, we ran sensitivity sweeps on unstructured bipartite graphs ($N = 150$).

Sensitivity to Initial Priors: We evaluated the algorithm's recovery from mismatched initial beliefs by sweeping the prior mean: Pessimistic ($\mu_0 = 0.1$), Neutral ($\mu_0 = 0.5$), and Optimistic ($\mu_0 = 0.9$). As demonstrated in Figure 9, an optimistic prior delays convergence. By assuming every unqueried edge is rewarding, the algorithm experiences continuous disappointment and wastes queries systematically disproving its own optimism. On the other hand, under a pessimistic prior, the algorithm assumes the entire unqueried graph is low-reward. Rather than randomly exploring in hopes of finding a high mean, it relies on the structural variance and margin components of its acquisition score. This allows the algorithm to focus on topological boundaries, dropping the simple regret faster.

Sensitivity to Observation Noise: To evaluate stability under increasing environmental chaos, we swept the observation noise $\sigma \in \{0.05, 0.1, 0.2, 0.3\}$. As demonstrated in Figure 10, the algorithm does not break or permanently stall under high noise. Instead, it exhibits statistical degradation, requiring proportionally more queries to reduce the posterior variance and resolve the structural boundaries.

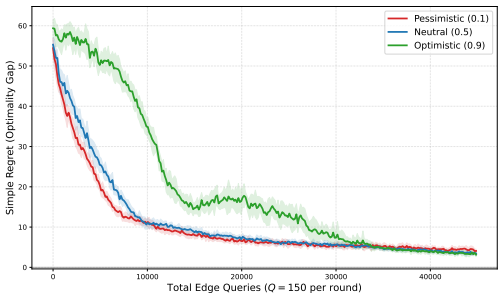


Figure 9: Prior Sensitivity ($N = 150$). Optimistic priors induce over-exploration, while pessimistic priors force the algorithm to rely efficiently on structural ambiguity.

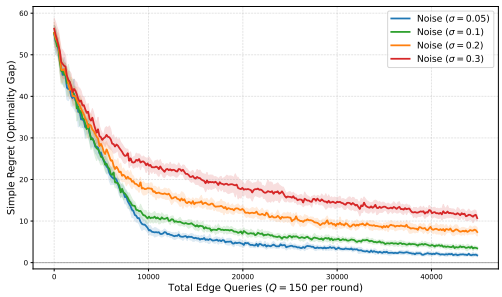


Figure 10: Noise Sensitivity ($N = 150$). The framework exhibits degradation, scaling its query requirements as the environmental stochasticity (σ) increases.

D.4 Spatial and Temporal Exploitation

To evaluate spatial exploitation, we deployed the algorithms on a Block-Structured "Type" Model ($N = 600$). In this topology, nodes are uniformly assigned to one of K latent categories, and the true edge weights are generated from a $K \times K$ underlying affinity matrix with added Gaussian noise. This creates distinct "blocks" of high- and low-reward regions.

As visualized in the logarithmic query heatmap in Figure 11, the SITS framework deduces and exploits this hidden topology without any prior knowledge of the block assignments. Instead of exploring uniformly, the algorithm allocates its query budget almost exclusively to the boundaries of the high-affinity blocks to resolve their internal ambiguity.

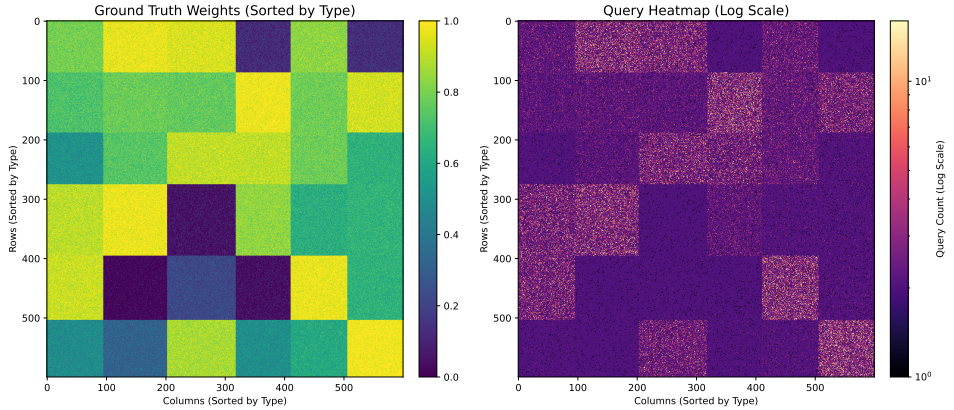


Figure 11: Query Heatmap on a Block-Structured Model ($N = 600$). The algorithm organically deduces the hidden topology, ignoring low-affinity regions to focus strictly on the boundaries of high-affinity blocks.

D.5 Asymptotic Scaling

We evaluated its asymptotic scaling behavior across bipartite graphs of increasing dimensionality. As shown in Figure 12, the number of iterations scales strictly linearly with the graph size ($I \approx C \cdot N$). The 'Target Optimality Threshold' is defined as reaching a simple regret bound of $\Delta_t \leq \epsilon$, signifying the framework has found the global optimum. As detailed in Appendix C, our solver bounds all internal beliefs, margin extractions, and Bayesian posterior updates to strictly $\mathcal{O}(N^2)$ per iteration. Because the iteration count scales linearly ($\mathcal{O}(N)$) while the per-iteration cost is quadratic ($\mathcal{O}(N^2)$), the cumulative computational complexity of the entire active-learning loop is bounded by $\mathcal{O}(N^3)$. This empirically proves that our SITS framework executes the entire sequential decision-making process in the same asymptotic time it takes a standard black-box oracle to solve the offline problem once.

E Extended Experiments: Shortest Path

E.1 Sensitivity Analysis: Priors and Observation Noise

We verify the framework's robustness to hyperparameter misspecification and environmental stochasticity in the SP domain utilizing unstructured layered DAGs ($|V| \approx 500$).

Sensitivity to Initial Priors: We evaluated the algorithm's recovery from mismatched initial beliefs by sweeping the prior mean. Because the SP problem is a cost-minimization objective, an initial prior of $\mu_0 = 0.1$ is Optimistic (the agent believes all unknown edges are cheap/fast), while a prior of $\mu_0 = 0.9$ is Pessimistic. As demonstrated in Figure 13, this produces the exact same behavioral dynamic observed in the MWM domain. The optimistic prior ($\mu_0 = 0.1$) causes over-exploration, as the algorithm must continually disappoint itself by querying edges that fail to meet its cheap expectations. Conversely, the pessimistic prior ($\mu_0 = 0.9$) forces the agent to rely on the QBC structural disagreement score to find the true shortest path.

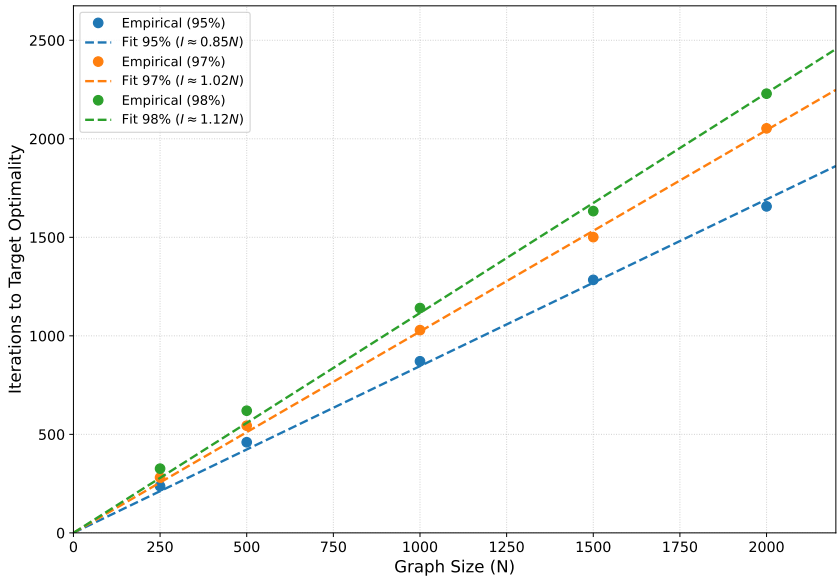


Figure 12: Iterations required to reach target optimality thresholds as a function of graph size N. The strictly linear relationships confirm the $\mathcal{O}(N^3)$ total active-learning runtime limit.

Sensitivity to Observation Noise: To evaluate stability under varying degrees of environmental stochasticity, we swept the semi-bandit observation noise $\sigma \in \{0.05, 0.1, 0.2, 0.3\}$. As demonstrated in Figure 14, the QBC policy degrades smoothly, maintaining its structural targeting capabilities while requiring proportionally larger query budgets to reduce posterior variance.

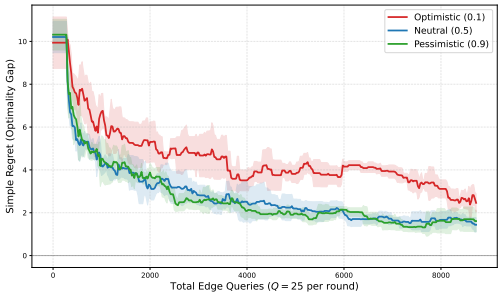


Figure 13: Prior Sensitivity ($|V| \approx 500$). Optimistic priors cause over-exploration, while pessimistic priors force the algorithm to rely on structural ambiguity.

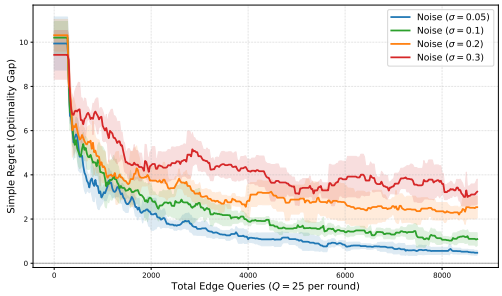


Figure 14: Noise Sensitivity ($|V| \approx 500$). The QBC policy exhibits robust, smooth statistical degradation as the environmental noise (σ) increases.

E.2 Implicit Structure Exploitation

We analyzed the spatial distribution of queries across the different regions of the "hidden highway". This environment is designed to visually isolate the optimal detour (the "highway"), the transitional bottleneck edges (the "ramps"), and a connected sub-optimal grid (the "deep city"). As visualized in Figure 15, the SITS framework shows an understanding of the graph's structure without any prior topological knowledge, allocating the majority of the query density to the optimal highway edges (17.68 queries/edge), while dedicating a small fraction of its budget to the uninformative deep city edges (1.54 queries/edge).

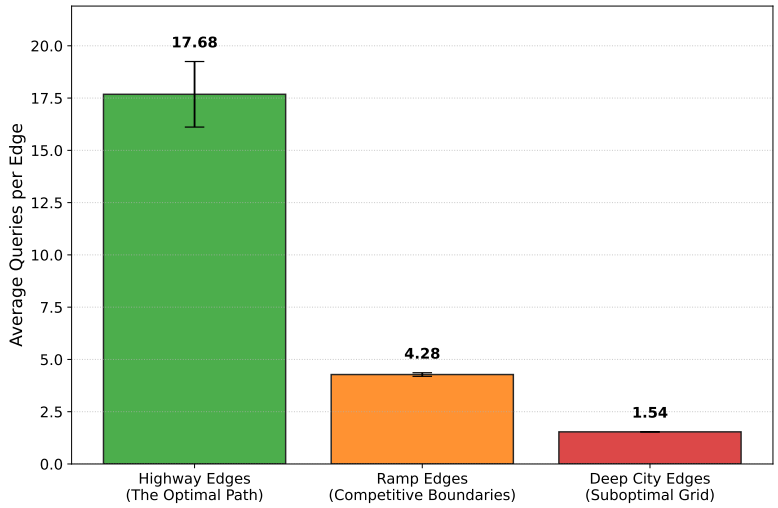


Figure 15: Average query density across different topological regions in the Hidden Highway trap. SITS spends almost no query budget on the uninformative city grid (1.54 queries/edge).

E.3 Asymptotic Scaling

To confirm that the computational efficiency of the SITS translates robustly to routing environments, we evaluated the asymptotic iteration scaling on dense, layered DAGs ranging from $|V| \approx 100$ to $|V| \approx 3000$ nodes. As shown in Figure 16, the total number of active-learning iterations scales linearly with the number of nodes ($I = \mathcal{O}(|V|)$) across all optimality precision thresholds.

By limiting the continuous BF solver to a small constant number of sweeps k, our framework ensures that the per-iteration computational cost is bounded by $\mathcal{O}(|E|)$. Combining this linear iteration scaling with the $\mathcal{O}(|E|)$ per-round footprint yields an active-learning cumulative complexity of $\mathcal{O}(|V||E|)$. This empirically proves that the SITS paradigm discovers the true optimal path in an unknown stochastic environment using the same asymptotic time complexity as executing the exact BF algorithm a single time in a fully known deterministic environment.

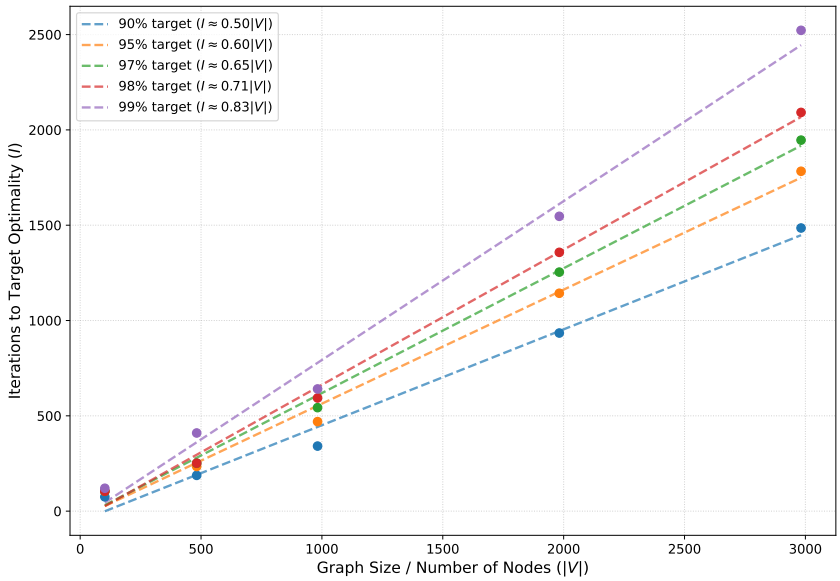


Figure 16: Iterations required to reach target optimality thresholds as a function of graph size $|V|$. The strictly linear relationships confirm the $\mathcal{O}(|V||E|)$ total active-learning runtime limit.