Matroid Semi-Bandits in Sublinear Time

Ruo-Chun Tzeng [1][§]   **Naoto Ohsaka** [2]   **Kaito Ariu** [2]

Abstract

We study the matroid semi-bandits problem, where at each round the learner plays a subset of K arms from a feasible set, and the goal is to maximize the expected cumulative linear rewards. Existing algorithms have per-round time complexity at least $\Omega(K)$, which becomes expensive when K is large. To address this computational issue, we propose `FasterCUCB` whose sampling rule takes time sublinear in K for common classes of matroids: $\mathcal{O}(D \operatorname{polylog}(K) \operatorname{polylog}(T))$ for uniform matroids, partition matroids, and graphical matroids, and $\mathcal{O}(D\sqrt{K}\operatorname{polylog}(T))$ for transversal matroids. Here, D is the maximum number of elements in any feasible subset of arms, and T is the horizon. Our technique is based on dynamic maintenance of an approximate maximum-weight basis over inner-product weights. Although the introduction of an approximate maximum-weight basis presents a challenge in regret analysis, we can still guarantee an upper bound on regret as tight as `CUCB` in the sense that it matches the gap-dependent lower bound by Kveton et al. (2014a) asymptotically.

1. Introduction

Matroid semi-bandits model many real-world tasks. An instance of matroid semi-bandit is described by $([K], \mathcal{X}, \boldsymbol{\mu})$, where $[K] \triangleq \{1, \cdots, K\}$ is the ground set, each $k \in [K]$ is associated with a probability distribution ν_k with mean μ_k, and $\mathcal{X} \subseteq \{0, 1\}^K$ is the set of bases of a given matroid $\mathcal{M} = ([K], \mathcal{I})$ of rank D. At each round $t \in [T]$, the learner pulls an action $\boldsymbol{x}(t) \in \mathcal{X}$ and observes a *semi-bandit* feedback, i.e., $y_k(t) \sim \nu_k$ iff $x_k(t) = 1$. This formulation can be used to model online advertising and news selection (Kale et al., 2010) with $\mathcal{M}$ as a uniform matroid. Ad placement

(Bubeck et al., 2013; Streeter et al., 2009) and diversified recommendation (Abbassi et al., 2013) can be modeled with $\mathcal{M}$ as a partition matroid. Network routing (Kveton et al., 2014a) can be modeled with $\mathcal{M}$ as a graphical matroid. Task assignment (Chen et al., 2016) can be modeled with $\mathcal{M}$ as a transversal matroid.

Popular algorithms include Combinatorial Upper Confidence Bound (`CUCB`) (Gai et al., 2012; Chen et al., 2013; Kveton et al., 2014a; 2015), Combinatorial Thompson Sampling (`CTS`) (Wang and Chen, 2018; Kong et al., 2021; Perrault, 2022), and the instance-specifically optimal algorithm KL-based Efficient Sampling for Matroids (`KL-OSM`) (Talebi and Proutiere, 2016). All of these algorithms rely on a greedy algorithm (see Algorithm 1) to determine the action to be pulled. The greedy algorithm takes time at least $\Omega(K)$ and at most $\mathcal{O}(K(\log K + \mathcal{T}_{\text{member}}))$, where $\mathcal{T}_{\text{member}}$ is the time taken to determine whether $\boldsymbol{x} + \boldsymbol{e}_k \in \mathcal{I}$ for some $(\boldsymbol{x}, k) \in \mathcal{I} \times [K]$, and $\boldsymbol{e}_k$ is the k-th canonical unit vector. However, when the number K of arms is large, performing the greedy algorithm at each round can become expensive. There is a need to develop a matroid semi-bandit algorithm with per-round time complexity sublinear in K.

In this work, we present `FasterCUCB` (Algorithm 5), the first sublinear-time algorithm for matroid semi-bandit. The design of `FasterCUCB` is based on `CUCB`, but with a much faster sampling rule which takes time sublinear in K for many classes of matroids. For uniform matroids, partition matroids, and graphical matroids, it has per-round time complexity of $\mathcal{O}(D \operatorname{polylog}(K) \operatorname{polylog}(T))$, which is optimal up to a polylogarithmic factor as compared to the trivial lower bound of $\Omega(D)$. For transversal matroids, the per-round time complexity is $\mathcal{O}(D\sqrt{K} \operatorname{polylog}(T))$, which is still sublinear in K when $D = \mathcal{O}(K^{\frac{1}{2}-\epsilon})$ for any $\epsilon > 0$. `FasterCUCB` trades the accuracy for computational efficiency. In other words, the action computed by the sampling rule of `FasterCUCB` is an *approximation* to the optimal solution computed by the sampling rule of `CUCB`. This introduces difficulty in the regret analysis because prior analysis of `CUCB` (Kveton et al., 2014a) requires the exact solution. What is interesting is that we can still guarantee the same regret upper bound asymptotically as prior analysis of `CUCB`.

To develop a sublinear-time sampling rule, we present a dynamic algorithm for maintaining maximum-weight base

---

	CUCB	FasterCUCB
Per-round Time Complexity	$\mathcal{O}(K(\log K + \mathcal{T}_{\text{member}}))$	$\mathcal{O}(D\,\text{polylog}\,(T)\,\mathcal{T}_{\text{update}}(\mathcal{A}))$
Uniform Matroid	$\mathcal{O}(K\log K)$	$\mathcal{O}(D\log K\,\text{polylog}\,(T))$
Partition Matroid	$\mathcal{O}(K\log K)$	$\mathcal{O}(D\log K\,\text{polylog}\,(T))$
Graphical Matroid	$\mathcal{O}(K\log K)$	$\mathcal{O}(D\,\text{polylog}\,(K)\,\text{polylog}\,(T))$
Transversal Matroid	$\mathcal{O}(K(\log K + DK))$	$\mathcal{O}(D\sqrt{K}\,\text{polylog}\,(T))$

Table 1. Per-round time complexity of CUCB (Kveton et al., 2014a) and FasterCUCB (Algorithm 5) for different classes of matroids. K is the number of arms and D is the maximum number of elements in any action in $\mathcal{X}$. $\mathcal{T}_{\text{member}}$ for different matroids is discussed in Appendix C. $\mathcal{T}_{\text{update}}(\mathcal{A})$ for different matroids is discussed in Section 3.

over inner product weights (Section 4). There have been many sublinear-time algorithms for dynamic maximum-weight base maintenance (see Section 3), which, however, may not be directly used in FasterCUCB because *all* arm weights representing the UCB index can change simultaneously at each round. Our insight for addressing this issue is that the UCB index of each arm k at round t can be decomposed into an inner product of the following two-dimensional vectors: (1) a *feature*, which depends on k and is supposedly a pair of the empirical reward estimate and radius of confidence interval, and (2) a *query*, which depends only on round t. Our proposed dynamic algorithm consists of two speeding-up techniques. One is *feature rounding*, which rounds each feature into a few bins so as to reduce the number of distinct features to consider. The other is the *minimum hitting set* technique, which allows us to compute a small number of queries in advance and correctly identify an (approximate) maximum-weight base for *any* query.

Sections are organized as follows. We introduce matroid semi-bandits and basic concepts in Section 2. We review relevant literature in Section 3. We develop a dynamic algorithm for maintaining a maximum-weight base over inner product weights in Section 4. We propose FasterCUCB based on the algorithms developed in Section 4 and analyzed its regret and time complexity in Section 5.

2. Preliminaries

We use $[n]$ to denote the set $\{1, \cdots, n\}$. We use $\boldsymbol{i}^\star(\boldsymbol{\mu})$ to denote any element in $\operatorname{argmax}_{\boldsymbol{x}\in\mathcal{X}} \langle \boldsymbol{\mu}, \boldsymbol{x} \rangle$, and when it is clear from the context, we drop $\boldsymbol{\mu}$ from $\boldsymbol{i}^\star(\boldsymbol{\mu})$ and write $\boldsymbol{i}^\star$. We use $\operatorname{supp}(\cdot)$ to denote the support set of a given vector. We use $\boldsymbol{e}_k$ to denote the vector with 1 only on the k-th row and 0's elsewhere, and use $\boldsymbol{0}_K$ to denote a K-dimensional vector with 0 on every row. We use $\log$ with base e. See Appendix A for a table of notation.

Matroid. A matroid is described by $\mathcal{M} \triangleq ([K], \mathcal{I})$, where $[K]$ is called the *ground set* and $\mathcal{I} \subseteq \{0,1\}^K$ is the set of *independent sets* satisfying (i) *hereditary property*, i.e., if $\operatorname{supp}(\boldsymbol{y}) \subset \operatorname{supp}(\boldsymbol{x})$ and $\boldsymbol{x} \in \mathcal{I}$, then $\boldsymbol{y} \in \mathcal{I}$; and

(ii) *augmentation property*, i.e., if $\boldsymbol{x}, \boldsymbol{y} \in \mathcal{I}$ and $\operatorname{supp}(\boldsymbol{y}) \subset \operatorname{supp}(\boldsymbol{x})$, then there exists $j \in \operatorname{supp}(\boldsymbol{x}) \setminus \operatorname{supp}(\boldsymbol{y})$ such that $\boldsymbol{y} + \boldsymbol{e}_j \in \mathcal{I}$. We said $\boldsymbol{x} \in \mathcal{I}$ is a basis if $\operatorname{supp}(\boldsymbol{x})$ is *maximal*, i.e., there does not exist $\boldsymbol{y} \in \mathcal{I}$ such that $\operatorname{supp}(\boldsymbol{x}) \subset \operatorname{supp}(\boldsymbol{y})$. All bases have the same cardinality, which is called the *rank* of the matroid. For $\boldsymbol{v} \in \mathbb{R}_+^K$, a maximum-weight basis $\boldsymbol{i}^\star(\boldsymbol{v}) \in \operatorname{argmax}_{\boldsymbol{x}\in\mathcal{X}} \sum_{k=1}^K v_k x_k$ can be found by a greedy algorithm (Algorithm 1) in $\mathcal{O}(K(\log K + \mathcal{T}_{\text{member}}))$ time, where $\mathcal{T}_{\text{member}}$ is the time taken for the membership oracle to determine whether $\boldsymbol{x} + \boldsymbol{e}_k \in \mathcal{I}$ for some $\boldsymbol{x} \in \mathcal{I}$ and some $k \in [K] \setminus \operatorname{supp}(\boldsymbol{x})$.

Algorithm 1 A greedy maximum-weight basis algorithm

Input: $\boldsymbol{v} \in \mathbb{R}^K$ and the bases $\mathcal{X} \subseteq \{0,1\}^K$.
Sort $\boldsymbol{v}$ in non-increasing order: $v_{\gamma(1)} \geq \cdots \geq v_{\gamma(K)}$;
$\boldsymbol{x} = \boldsymbol{0}_K; i = 1$;
while $\|\boldsymbol{x}\|_0 < D$ **do**
  if $\boldsymbol{x} + \boldsymbol{e}_{\gamma(i)} \in \mathcal{I}$ **then**
    $\boldsymbol{x} \leftarrow \boldsymbol{x} + \boldsymbol{e}_{\gamma(i)}$
  $i \leftarrow i + 1$;
end

Matroid semi-bandits. An instance of matroid semi-bandit is described by $([K], \mathcal{X}, \boldsymbol{\mu})$, where $[K]$ is the ground set, $\mathcal{X} \subseteq \{0,1\}^K$ is the set of bases of the given matroid $\mathcal{M} \triangleq ([K], \mathcal{I})$ of rank D, and $\mathcal{I} \subseteq \{0,1\}^K$ is the set of independent sets. Each $k \in [K]$ is associated with a distribution ν_k with mean μ_k. The learner knows the matroid $\mathcal{M}$, and aims to learn the best action $\boldsymbol{i}^\star(\boldsymbol{\mu}) \in \operatorname{argmax}_{\boldsymbol{x}\in\mathcal{X}} \langle \boldsymbol{\mu}, \boldsymbol{x} \rangle$ by playing a game with the environment: At each round $t \in \mathbb{N}$, the learner plays an action $\boldsymbol{x}(t)$, and the environment draws a noisy reward $y_k(t) \sim \nu_k$ for each arm $k \in [K]$ and reveals $y_k(t)$ to the learner iff $k \in \operatorname{supp}(\boldsymbol{x}(t))$. We assume arms' rewards are bounded:

Assumption 2.1. Assume the support of each arm ν_k is a subset of $[a, b]$ and $0 < a < b < \infty$.

The performance is measured by expected regret:

$$R(T) \triangleq T \langle \boldsymbol{\mu}, \boldsymbol{i}^\star(\boldsymbol{\mu}) \rangle - \mathbb{E}\left[\sum_{t=1}^T \langle \boldsymbol{\mu}, \boldsymbol{x}(t) \rangle\right],$$

which is the difference between the expected cumulative reward of the learner and that of an algorithm who knows $\boldsymbol{\mu}$ and always selects the best action $\boldsymbol{i}^\star(\boldsymbol{\mu})$.

Common classes of matroids. Refer to Chapter 39 (Schrijver, 2003) or Chapter 1 (Oxley, 2011) for more details.

- A *uniform matroid* $([K], \mathcal{I})$ of rank D has independent sets $\mathcal{I} = \{S \subseteq [K] : |S| \le D\}$ and the bases $\mathcal{X}$ consist of subsets whose cardinalities are exactly D.

- A *partition matroid* $([K], \mathcal{I})$ of rank D is given a partition $S_1, \cdots, S_D$ of the ground set $[K]$, the independent sets $\mathcal{I} = \{S : |S \cap S_i| \le 1, \forall i \in [D]\}$, and the bases $\mathcal{X}$ are subsets that choose exactly one element from each of the D sets.

- A *graphical matroid* is given a graph $G = (V, E)$ with K edges, the bases $\mathcal{X}$ consist of all spanning forests in G, and the rank D is $|V|$ minus the number of connected components in G.

- A *transversal matroid* is given a bipartite graph $G = ([K] \cup V, E)$ with a bipartition $([K], V), |V| \le K$, the independent sets $\mathcal{I}$ consist of $S \subseteq [K]$ such that there is a matching of S to $|S|$ vertices in V, and $\mathcal{X}$ is the set of endpoints in $[K]$ of all maximum matchings in G.

We discuss the query time $\mathcal{T}_{\text{member}}$ of membership oracle in Appendix C. For more examples on semi-bandits under different matroid constraints, we refer the readers to (Kveton et al., 2014a) for linear matroids, and (Kveton et al., 2014b) for polymatroid semi-bandits.

CUCB. The action selected by CUCB (Gai et al., 2012; Chen et al., 2013; Kveton et al., 2014b) at round $t \in \mathbb{N}$ is:

$$\boldsymbol{x}(t) \in \operatorname*{argmax}_{\boldsymbol{x} \in \mathcal{X}} \sum_{k=1}^{K} \left(\hat{\mu}_k(t-1) + \frac{\lambda_t}{\sqrt{N_k(t-1)}} \right) x_k, \tag{1}$$

where $\hat{\mu}_k(t) \triangleq \frac{1}{N_k(t)} \sum_{s=1}^{t} y_k(s) \mathbb{1}\{x_k(s) = 1\}$ is the empirical reward estimate, $N_k(t) \triangleq \sum_{s=1}^{t} \mathbb{1}\{x_k(s) = 1\}$ is the number of pulls of arm k, and $\lambda_t > 0$ controls the confidence width. The value $\hat{\mu}_k(t-1) + \frac{\lambda_t}{\sqrt{N_k(t-1)}}$ is called the *UCB index* of arm k. In (Kveton et al., 2014a), Eq. (1) is solved by a $\mathcal{O}(K(\log K + \mathcal{T}_{\text{member}}))$-time greedy algorithm shown in Algorithm 1. In Section 4, we will develop a faster algorithm for solving Eq. (1) with the following reformulation:

$$\boldsymbol{x}(t) \in \operatorname*{argmax}_{\boldsymbol{x} \in \mathcal{X}} \sum_{k=1}^{K} \langle \boldsymbol{f}_k, \boldsymbol{q} \rangle x_k,$$

where $\boldsymbol{f}_k = (\hat{\mu}_k(t-1), \frac{1}{\sqrt{N_k(t-1)}})$ and $\boldsymbol{q} = (1, \lambda_t)$.

3. Related Works

Semi-bandits and sublinear-time bandits. We provide an extensive survey on related bandit literature in Appendix B. To summarize here, for semi-bandit algorithms, CUCB (Gai et al., 2012; Chen et al., 2013; Kveton et al., 2014a), CTS (Wang and Chen, 2018; Kong et al., 2021; Perrault, 2022) and KL-OSM (Talebi and Proutiere, 2016) all rely on a $\mathcal{O}(K(\log K + \mathcal{T}_{\text{member}}))$-time greedy algorithm to compute the action to be pulled. In contrast, our FasterCUCB, as far as we know, is the first semi-bandit algorithm having per-round time complexity of $o(K)$. For linear bandits, there exist several works (Jun et al., 2017; Yang et al., 2022) on reducing the per-round complexity to be sublinear in the number of arms. But, their results transferred to our setting are worse than what we have obtained both in terms of regret bound and the time complexity (see the discussion in Appendix B).

Dynamic maintenance of maximum-weight base of a matroid Here, we review existing dynamic algorithms for maintaining a maximum-weight base of a matroid. In a standard (fully-)dynamic setting, we are given a weighted matroid $\mathcal{M} = ([K], \mathcal{I})$, where each arm's weight dynamically changes over time in an online manner. The objective is to maintain any (exact or approximate) maximum-weight basis of $\mathcal{M}$ over up-to-date arm weights as efficiently as possible. We use $\mathcal{T}_{\text{update}}(\mathcal{A})$ to denote the time complexity of a dynamic algorithm $\mathcal{A}$ required for updating an (approximate) maximum-weight base according to the change of a single arm weight. The best-known bound of $\mathcal{T}_{\text{update}}(\mathcal{A})$ for each matroid class is summarized as follows: For graphic matroids, a maximum-weight basis can be updated in $\mathcal{O}(\sqrt{K})$ worst-case time (Frederickson, 1985; Eppstein et al., 1997) and in $\mathcal{O}(\text{polylog}(K))$ amortized time (Holm et al., 2001). For laminar matroids (which include uniform and partition matroids as special cases), the worst-case time complexity for exact dynamic algorithms is bounded by $\mathcal{O}(\log K)$ (Henzinger et al., 2023). For transversal matroids, a $\mathcal{O}(K^{1.407})$-time exact dynamic algorithm is known (van den Brand et al., 2019), while a $(1 + \eta)$-approximation dynamic algorithm runs in $\mathcal{O}(\eta^{-2}\sqrt{K})$ time (Gupta and Peng, 2013). We can safely assume that, after updating multiple arm weights, $\mathcal{A}$ returns an (approximate) maximum-weight basis in $\mathcal{O}(D)$ time, where D is the rank of a matroid.

4. Dynamic Maintenance of Maximum-weight Base over Inner Product Weight

In this section, we develop a sublinear-time sampling rule, which is used as a subroutine in FasterCUCB. Recall that *any* static algorithm that solves the linear maximization of Eq. (1) from scratch requires at least $\Omega(K)$ time, which is computationally expensive. To circumvent this issue,

we present a dynamic algorithm for maintaining an (approximate) maximum-weight base of a matroid where arm weights change over time. The next subsection begins with formalizing the problem setting.

4.1. Problem Setting and Technical Result

Consider the following problem setting: Let $\mathcal{M} = ([K], \mathcal{I})$ be a matroid of rank D over K arms, and $\mathcal{X}$ be the set of its bases. Each arm $k \in [K]$ has a (nonnegative) two-dimensional vector $\boldsymbol{f}_k = (\alpha_k, \beta_k) \in \mathbb{R}_+^2$ referred to as a *feature*, which may change as time goes by. Given a (nonnegative) two-dimensional vector $\boldsymbol{q} \in \mathbb{R}_+^2$ as a *query*, we are required to find any (approximate) maximum-weight base of $\mathcal{M}$, where arm k's weight is given by projecting its feature onto $\boldsymbol{q}$, i.e., $\langle \boldsymbol{f}_k, \boldsymbol{q} \rangle$.

Observe that in the matroid semi-bandit setting, each arm k's feature $\boldsymbol{f}_k = (\alpha_k, \beta_k)$ corresponds to a pair of the empirical reward estimate $\alpha_k = \hat{\mu}_k(t-1)$ and radius $\beta_k = \frac{1}{\sqrt{N_k(t-1)}}$ of confidence interval, and a query is $\boldsymbol{q} = (1, \lambda_t)$ at round t, both of which change over rounds.

Hereafter, we make the following two assumptions.

Assumption 4.1. Lower and upper bounds, denoted by α_{lb} and α_{ub} (resp. β_{lb} and β_{ub}), on the possible positive values of α_k's (resp. β_k's) at anytime are known; namely, it always holds that $\alpha_k \in \{0\} \cup [\alpha_{\text{lb}}, \alpha_{\text{ub}}]$ and $\beta_k \in \{0\} \cup [\beta_{\text{lb}}, \beta_{\text{ub}}]$. The precise values of these bounds will be discussed in Section 5.

Assumption 4.2. There exists a dynamic algorithm $\mathcal{A}$ for maintaining a $(1 + \eta)$-approximate maximum-weight base of $\mathcal{M}$ with arm weights changing over time, where parameter $\eta \in (0, 1)$ specifies the approximation guarantee. Denote by $\mathcal{T}_{\text{init}}(\mathcal{A}; \eta)$ and $\mathcal{T}_{\text{update}}(\mathcal{A}; \eta)$ the time complexity of $\mathcal{A}$ required for initializing the data structure and updating a single arm's weight, respectively. We can safely assume that after updating multiple arm weights, $\mathcal{A}$ returns a maximum-weight base in $\mathcal{O}(D)$ time. See Section 3 for existing implementations.

Our dynamic algorithm is parameterized by a *precision parameter* $\epsilon \in (0, 1)$, and consists of the following three procedures:

INITIALIZE: Given lower and upper bounds $[\alpha_{\text{lb}}, \alpha_{\text{ub}}]$ and $[\beta_{\text{lb}}, \beta_{\text{ub}}]$ as in Assumption 4.1, K features $(\alpha_k, \beta_k)_{k \in [K]}$, a matroid $\mathcal{M} = ([K], \mathcal{I})$, a dynamic algorithm $\mathcal{A}$ for maximum-weight base maintenance as in Assumption 4.2, and a precision parameter ϵ, this procedure initializes the data structure used in the remaining two procedures.

FIND-BASE: Given a query $\boldsymbol{q}$, this procedure is supposed to return a $(1 + \epsilon)$-approximate maximum-weight base

of $\mathcal{M}$, where arm k's weight is defined as $\langle \boldsymbol{f}_k, \boldsymbol{q} \rangle$ for the up-to-date k's feature $\boldsymbol{f}_k$.

UPDATE-FEATURE: Given an arm k and a new feature $\boldsymbol{f}_k'$, this procedure reflects the change of arm k's feature on the data structure.

Remark 4.3. Our problem setting is different from a canonical setting of dynamic maximum-weight base maintenance in a sense that the arm weights are revealed when a query is issued in **FIND-BASE**. Consequently, existing dynamic algorithms may not be used directly.

The technical result in this section is stated below.

Theorem 4.4 (∗). *There exist implementations of* **INITIALIZE**, **FIND-BASE**, *and* **UPDATE-FEATURE** *such that the following are satisfied:* **FIND-BASE** *always returns a* $(1 + \epsilon)$-*approximate maximum-weight base of a matroid* $\mathcal{M}$ *with arm k's weight defined as* $\langle \boldsymbol{f}_k, \boldsymbol{q} \rangle$ *for an up-to-date k's feature* $\boldsymbol{f}_k$ *and a query* $\boldsymbol{q}$. *Moreover,* **INITIALIZE** *runs in* $\mathcal{O}(K + \text{poly}(W) \cdot \mathcal{T}_{\text{init}}(\mathcal{A}; \frac{\epsilon}{3}))$ *time,* **FIND-BASE** *runs in* $\mathcal{O}(\text{poly}(W) + D)$ *time, and* **UPDATE-FEATURE** *runs in* $\mathcal{O}(\text{poly}(W) \cdot \mathcal{T}_{\text{update}}(\mathcal{A}; \frac{\epsilon}{3}))$ *time, where*

$$W = \mathcal{O}\left(\epsilon^{-1} \cdot \log\left(\frac{\alpha_{ub}}{\alpha_{lb}} \cdot \frac{\beta_{ub}}{\beta_{lb}} \right) \right). \tag{2}$$

Remark 4.5. The proof of Theorem 4.4 can be easily adapted to the case when (the update procedure of) dynamic algorithm $\mathcal{A}$ has only amortized complexity. In such case, Theorem 4.4 holds in the *amortized* sense rather than the *worst-case* sense.

The remainder of this section is organized as follows: In Section 4.2, we apply a rounding technique to arm features to reduce the number of distinct features to consider, in Section 4.3, we investigate the representability of permutations induced by inner product weights to deal with multiple queries efficiently, and Section 4.4 finally develops our dynamic algorithm for maximum-weight base maintenance. All proofs of the lemmas appearing in this section are deferred to Appendix D.

4.2. Rounding Arm Features

Here, we apply a rounding technique to arm features so as to reduce the number of distinct features to consider. Hereafter, let $\eta \triangleq \frac{\epsilon}{3}$, so that $(1 + \eta)^2 \leq 1 + \epsilon$ for $\epsilon \in (0, 1)$. Define

$$W \triangleq \max\left\{ \left\lceil \log_{1+\eta}\left(\frac{\alpha_{\text{ub}}}{\alpha_{\text{lb}}} \right) \right\rceil, \left\lceil \log_{1+\eta}\left(\frac{\beta_{\text{ub}}}{\beta_{\text{lb}}} \right) \right\rceil \right\},$$

$$\mathbb{W} \triangleq \{-\infty\} \cup [W] = \{-\infty, 1, 2, 3, \dots, W\}. \tag{3}$$

Since any features are within $(\{0\} \cup [\alpha_{\text{lb}}, \alpha_{\text{ub}}]) \times (\{0\} \cup [\beta_{\text{lb}}, \beta_{\text{ub}}])$ as guaranteed by Assumption 4.2, we shall partition the possible region of the features into $|\mathbb{W}|^2$ bins. For

each $q, r \in \mathbb{W}$, define $\mathrm{BIN}_{q,r} \subset \mathbb{R}_+^2$ as

$$\mathrm{BIN}_{q,r} \triangleq (\alpha_{\mathrm{lb}}(1+\eta)^{q-1}, \alpha_{\mathrm{lb}}(1+\eta)^q] \tag{4}$$
$$\times (\beta_{\mathrm{lb}}(1+\eta)^{r-1}, \beta_{\mathrm{lb}}(1+\eta)^r],$$

$$\text{where } (\alpha_{\mathrm{lb}}(1+\eta)^{-\infty}, \alpha_{\mathrm{lb}}(1+\eta)^{-\infty}] \triangleq \{0\}, \tag{5}$$
$$(\beta_{\mathrm{lb}}(1+\eta)^{-\infty}, \beta_{\mathrm{lb}}(1+\eta)^{-\infty}] \triangleq \{0\}. \tag{6}$$

Note that these bins are pairwise disjoint, and that $\cup_{q,r \in \mathbb{W}} \mathrm{BIN}_{q,r}$ covers $(\{0\} \cup [\alpha_{\mathrm{lb}}, \alpha_{\mathrm{ub}}]) \times (\{0\} \cup [\beta_{\mathrm{lb}}, \beta_{\mathrm{ub}}])$; i.e., any possible feature belongs to a unique $\mathrm{BIN}_{q,r}$. For each $q, r \in \mathbb{W}$, let $\mathrm{dom}_{q,r} \in \mathbb{R}_+^2$ denote the unique *dominating point* of $\mathrm{BIN}_{q,r}$; namely,

$$\mathrm{dom}_{q,r} \triangleq (\alpha_{\mathrm{lb}}(1+\eta)^q, \beta_{\mathrm{lb}}(1+\eta)^r). \tag{7}$$

For any feature $\boldsymbol{f}_k = (\alpha_k, \beta_k)$, we will use $\mathrm{dom}(\boldsymbol{f}_k) = \mathrm{dom}(\alpha_k, \beta_k)$ to denote the dominating point $\mathrm{dom}_{q,r}$ such that $\boldsymbol{f}_k \in \mathrm{BIN}_{q,r}$. See Figure 1 in Appendix D for illustration of $\mathrm{BIN}_{q,r}$'s, $\mathrm{dom}_{q,r}$'s, and $\mathrm{dom}(\boldsymbol{f}_k)$'s.

Observe easily that for any feature $\boldsymbol{f}_k \in \mathbb{R}_+^2$ and query $\boldsymbol{q} \in \mathbb{R}_+^2$,

$$\frac{1}{1+\eta} \cdot \langle \mathrm{dom}(\boldsymbol{f}_k), \boldsymbol{q} \rangle < \langle \boldsymbol{f}_k, \boldsymbol{q} \rangle \leq \langle \mathrm{dom}(\boldsymbol{f}_k), \boldsymbol{q} \rangle. \tag{8}$$

By Eq. (8), we can replace each arm's feature by its dominating point without much deteriorating the quality of the (approximate) maximum-weight base, as shown below.

Lemma 4.6 (∗). *Let $\boldsymbol{f}_1, \ldots, \boldsymbol{f}_K \in (\{0\} \cup [\alpha_{lb}, \alpha_{ub}]) \times (\{0\} \cup [\beta_{lb}, \beta_{ub}])$ be K features, $\boldsymbol{q} \in \mathbb{R}_+^2$ be a query, and $\boldsymbol{x}_{\mathrm{dom}}^*$ be a $(1+\eta)$-approximate maximum-weight base of matroid $\mathcal{M}$ with arm k's weight defined as $\langle \mathrm{dom}(\boldsymbol{f}_k), \boldsymbol{q} \rangle$. Then, for any base $\boldsymbol{x}$ of $\mathcal{M}$, it holds that*

$$\sum_{k \in \mathit{supp}(\boldsymbol{x}_{\mathrm{dom}}^*)} \langle \boldsymbol{f}_k, \boldsymbol{q} \rangle \geq \frac{1}{1+\epsilon} \cdot \sum_{k \in \mathit{supp}(\boldsymbol{x})} \langle \boldsymbol{f}_k, \boldsymbol{q} \rangle. \tag{9}$$

In particular, $\boldsymbol{x}_{\mathrm{dom}}^*$ is a $(1+\epsilon)$-approximate maximum-weight base with arm k's weight defined as $\langle \boldsymbol{f}_k, \boldsymbol{q} \rangle$.

4.3. Handling Multiple Queries

From weighting to permutation. Now we deal with multiple queries. Our idea is that, if two queries $\boldsymbol{q}_1, \boldsymbol{q}_2 \in \mathbb{R}_+^2$ are "very close" to each other, then they should derive the common maximum-weight base (provided that features are fixed). This intuition can be justified with respect to *orderings* of arms. For two total orders $\preceq$ and $\preceq^\circ$ over $[K]$, we say that $\preceq^\circ$ is *consistent with* $\preceq$ if $k \preceq^\circ k'$ implies $k \preceq k'$ for any $k \neq k'$. The following fact is easy to confirm:

Lemma 4.7 (∗). *Let $\boldsymbol{w} = (w_1, \ldots, w_K) \in \mathbb{R}_+^K$ be K arm weights and $\preceq$ be a total order over $[K]$ such that $k \succeq k'$ if and only if $w_k \geq w_{k'}$. Let $\preceq^\circ$ be a total order over $[K]$ that*

is consistent with $\preceq$. If $\boldsymbol{x}^\circ$ is a base of matroid $\mathcal{M}$ obtained by running the greedy algorithm over any ordering of $[K]$ consistent with $\preceq^\circ$, it is a maximum-weight base of $\mathcal{M}$ over arm k's weight w_k; namely, for any base $\boldsymbol{x}$ of $\mathcal{M}$,

$$\langle \boldsymbol{x}^\circ, \boldsymbol{w} \rangle \geq \langle \boldsymbol{x}, \boldsymbol{w} \rangle. \tag{10}$$

Lemma 4.7 implies that any maximum-weight base can be obtained by running the greedy algorithm over some total order $\preceq^\circ$; moreover, we can safely assume that $\preceq^\circ$ is *strict* (i.e., $k \prec^\circ k'$ or $k \succ^\circ k'$ for all $k \neq k'$), or equivalently, a *permutation* over $[K]$. Our strategy for dealing with multiple queries is: (1) we enumerate all possible permutations in advance, and (2) we guess a permutation consistent with the arm weights determined based on a query. To this end, the following question arises: *What kind of and how many permutations are representable given a fixed set of features?*

Characterizing representable permutations. To answer the above question, we characterize representable permutations. Hereafter, let $\mathfrak{S}_K$ denote the set of all permutations over $[K]$, and $\boldsymbol{f}_1, \ldots, \boldsymbol{f}_K \in \mathbb{R}_+^2$ be any fixed, distinct K features. We say that a query $\boldsymbol{q} \in \mathbb{R}^2$ over $\boldsymbol{f}_1, \ldots, \boldsymbol{f}_K$ *represents* a permutation $\pi \in \mathfrak{S}_K$ if $\langle \boldsymbol{f}_{\pi(i)}, \boldsymbol{q} \rangle > \langle \boldsymbol{f}_{\pi(j)}, \boldsymbol{q} \rangle$ for all $1 \leq i < j \leq K$,[1] and that π is *representable* if such $\boldsymbol{q}$ exists.

For a permutation $\pi \in \mathfrak{S}_K$ to be representable, we wish for some query $\boldsymbol{q} \in \mathbb{R}^2$ to ensure that for any $i < j$, arm $\pi(i)$'s weight is (strictly) higher than arm $\pi(j)$'s weight. This requirement is equivalent to $\langle \boldsymbol{f}_{\pi(i)} - \boldsymbol{f}_{\pi(j)}, \boldsymbol{q} \rangle > 0$; thus, if the following system of linear inequalities is feasible, any of its solutions $\boldsymbol{q}$ represents π:

$$\langle \boldsymbol{f}_{\pi(i)} - \boldsymbol{f}_{\pi(j)}, \boldsymbol{q} \rangle > 0 \text{ for all } 1 \leq i < j \leq K. \tag{11}$$

Observe now that the above system is feasible *if and only if* the intersection of $\mathcal{P}_{i,j}$ for all $i < j$ is nonempty, where $\mathcal{P}_{i,j}$ is an open half-plane defined as

$$\mathcal{P}_{i,j} \triangleq \{\boldsymbol{q} \in \mathbb{R}^2 : \langle \boldsymbol{f}_{\pi(i)} - \boldsymbol{f}_{\pi(j)}, \boldsymbol{q} \rangle > 0\}. \tag{12}$$

Because each $\mathcal{P}_{i,j}$ is obtained by dividing $\mathbb{R}^2$ by a unique line that is orthogonal to line $\overleftrightarrow{\boldsymbol{f}_{\pi(i)} \boldsymbol{f}_{\pi(j)}}$ and intersects the origin $\boldsymbol{0}$, the set of feasible solutions for Eq. (11) is equal to (the interior of) a *polyhedral cone* defined by the boundaries of a particular pair of $\mathcal{P}_{i,j}$'s.

Here, we characterize the representable permutations by the concept of arrangement of lines. Let $\mathcal{L} \triangleq \{\boldsymbol{l}_1, \ldots, \boldsymbol{l}_{\binom{K}{2}}\}$ denote $\binom{K}{2}$ lines, each of which is orthogonal to line $\overleftrightarrow{\boldsymbol{f}_k \boldsymbol{f}_{k'}}$ for some $k \neq k'$ and intersects $\boldsymbol{0}$. Given such $\mathcal{L}$, a *cell* $\mathcal{C}$

---

[1] This definition does not allow "ties"; i.e., no pair $k \neq k'$ satisfies $\langle \boldsymbol{f}_k, \boldsymbol{q} \rangle = \langle \boldsymbol{f}_{k'}, \boldsymbol{q} \rangle$.

in arrangement of $\mathcal{L}$ is defined as a maximum connected region of $\mathbb{R}^2$ that does not intersect with $\mathcal{L}$ (which is the interior of a polyhedral cone). Then, for each cell $\mathcal{C}$, every query $\boldsymbol{q}$ in $\mathcal{C}$ represents the same permutation $\pi_{\mathcal{C}} \in \mathfrak{S}_K$ depending only on $\mathcal{C}$; namely, *there is a bijection between the representable permutations and the cells in arrangement of* $\mathcal{L}$. See Figure 2 in Appendix D for illustration.

With this connection in mind, we demonstrate that reserving a single vector for each cell suffices to cover all representable permutations. A *minimum hitting set* of the cells in arrangement of $\mathcal{L}$ is defined as any minimum set $\mathcal{H}$ of vectors in $\mathbb{R}^2$ such that $\mathcal{H}$ and each cell have a non-empty intersection.

Lemma 4.8 (∗). *Let $\mathcal{H}$ be a minimum hitting set of the cells in arrangement of* $\mathcal{L}$. *Then, for any query $\boldsymbol{q} \in \mathbb{R}^2$, there exists a vector $\boldsymbol{h} \in \mathcal{H}$ such that for any $k \neq k'$,*

$$\langle \boldsymbol{f}_k, \boldsymbol{h} \rangle > \langle \boldsymbol{f}_{k'}, \boldsymbol{h} \rangle \implies \langle \boldsymbol{f}_k, \boldsymbol{q} \rangle \geq \langle \boldsymbol{f}_{k'}, \boldsymbol{q} \rangle. \quad (13)$$

Lemma 4.8 along with Lemma 4.7 ensure that for any query $\boldsymbol{q} \in \mathbb{R}^2$, there is a vector $\boldsymbol{h}$ in $\mathcal{H}$ such that a maximum-weight base with arm k's weight $\langle \boldsymbol{f}_k, \boldsymbol{h} \rangle$ is a maximum-weight base with arm k's weight $\langle \boldsymbol{f}_k, \boldsymbol{q} \rangle$.

Generating a minimum hitting set. Subsequently, we generate a minimum hitting set. One may think that it requires exponentially long time because the number of permutations in $\mathfrak{S}_K$ is $K!$. However, it turns out that the number of cells in arrangement of $\mathcal{L}$ is $\mathcal{O}(K^2)$ (i.e., so is the number of representable permutations), and a minimum hitting set can be constructed in $\text{poly}(K)$ time.

Lemma 4.9 (∗). *The number of cells in arrangement of* $\mathcal{L}$ *is at most $\mathcal{O}(K^2)$. Moreover, we can generate a minimum hitting set in $\text{poly}(K)$ time (by using Algorithm 6 in Appendix D).*

4.4. Putting It All Together: Algorithm Description and Complexity

We are now ready to implement the three procedures. We here stress that applying either of the feature rounding or minimum hitting set technique *separately* does not make sense: On one hand, if we only apply feature rounding, we would have to recompute each arm's weight *every time* a query is issued, which is expensive. On the other hand, if the minimum hitting set technique is only applied (to raw features $\boldsymbol{f}_k$'s), then a minimum hitting set $\mathcal{H}$ cannot be constructed in advance due to a dynamic nature of features, and its size would be $\mathcal{O}(K^2)$, which is prohibitive.

By applying both techniques, (1) we know *a priori* the set of possible dominating points, whose size $\mathcal{O}(W^2)$ depends *only* on W; moreover, (2) we can create a minimum hitting set $\mathcal{H}$ of size $\mathcal{O}(W^4)$ beforehand at initialization.

Pseudocodes of **INITIALIZE**, **FIND-BASE**, and **UPDATE-FEATURE** are described in Algorithms 2 to 4, respectively. The proof of Theorem 4.4 follows from Lemmas 4.6 to 4.9, whose details are deferred to Appendix D.

In **INITIALIZE**, we construct a hitting set $\mathcal{H}$ of $\text{dom}_{q,r}$'s and $\frac{1}{1+\eta} \cdot \text{dom}_{q,r}$'s rather than solely of $\text{dom}_{q,r}$'s, which incurs a constant-factor blowup in the time complexity. Though this change is not needed in the proof of Theorem 4.4, the following immediate corollary of Lemma 4.8 is crucial in the regret analysis of Section 5.

Corollary 4.10 (∗). *Let $\mathcal{H}$ be a minimum hitting set constructed in Algorithm 2. Then, for any query $\boldsymbol{q} \in \mathbb{R}^2$, there exists a vector $\boldsymbol{h} \in \mathcal{H}$ such that for any $\text{dom} = \text{dom}_{q,r}$ and $\text{dom}' = \text{dom}_{q',r'}$ with $q, r, q', r' \in \mathbb{W}$,*

$$\langle \text{dom}, \boldsymbol{h} \rangle > \langle \text{dom}', \boldsymbol{h} \rangle \implies \langle \text{dom}, \boldsymbol{q} \rangle \geq \langle \text{dom}', \boldsymbol{q} \rangle,$$

$$\langle \text{dom}, \boldsymbol{h} \rangle > \frac{\langle \text{dom}', \boldsymbol{h} \rangle}{1 + \eta} \implies \langle \text{dom}, \boldsymbol{q} \rangle \geq \frac{\langle \text{dom}', \boldsymbol{q} \rangle}{1 + \eta}.$$

Algorithm 2 INITIALIZE.

Input: lower and upper bounds $[\alpha_{\text{lb}}, \alpha_{\text{ub}}]$ and $[\beta_{\text{lb}}, \beta_{\text{ub}}]$; K features $(\boldsymbol{f}_k)_{k \in [K]}$; precision parameter $\epsilon \in (0, 1)$.
Define $\mathbb{W}$ by Eq. (3);
for *each $q, r \in \mathbb{W}$* **do**
  Define $\text{BIN}_{q,r}$ and $\text{dom}_{q,r}$ by Eqs. (4) and (7);
end
Define $\eta \triangleq \frac{\epsilon}{3}$;
Construct a minimum hitting set $\mathcal{H}$ of size $\mathcal{O}(W^4)$ for $\text{dom}_{q,r}$'s and $\frac{1}{1+\eta} \cdot \text{dom}_{q,r}$'s by Lemma 4.9;
for *each $\boldsymbol{h} \in \mathcal{H}$* **do**
  Create an instance $\mathcal{A}_{\boldsymbol{h}}$ of dynamic maximum-weight base algorithm with precision parameter $\eta \triangleq \frac{\epsilon}{3}$, $\mathcal{M}$, and arm k's weight $\langle \text{dom}(\boldsymbol{f}_k), \boldsymbol{h} \rangle$;
end

Algorithm 3 FIND-BASE.

Input: query $\boldsymbol{q} \in \mathbb{R}^2_+$.
Find $\boldsymbol{h} \in \mathcal{H}$ s.t. $\boldsymbol{q}$ and $\boldsymbol{h}$ belong to (the closure of) the same cell in arrangement of $\mathcal{V}$;
Call $\mathcal{A}_{\boldsymbol{h}}$ and return the maximum-weight base $\boldsymbol{x}^\circ$ of $\mathcal{M}$ with arm k's weight $\langle \text{dom}(\boldsymbol{f}_k), \boldsymbol{h} \rangle$;

Algorithm 4 UPDATE-FEATURE.

Input: arm $k \in [K]$; new feature $\boldsymbol{f}'_k \in \mathbb{R}^2_+$.
for *each $\boldsymbol{h} \in \mathcal{H}$* **do**
  Change arm k's weight stored in $\mathcal{A}_{\boldsymbol{h}}$ to $\langle \text{dom}(\boldsymbol{f}'_k), \boldsymbol{h} \rangle$;
end

---

5. Our Proposed Algorithm: `FasterCUCB`

In this section, we present `FasterCUCB` in Algorithm 5, which uses procedures introduced in Section 4.

The purpose of initialization procedure is to ensure each arm is pulled at least once. It takes at most K rounds, and in each round, the computation of $i^\star(e_k)$ takes $\mathcal{O}(K \cdot \mathcal{T}_{\mathrm{member}})$ time because the permutation γ in Algorithm 1 can be specified explicitly as $\gamma(j) = k$ if $j = 1$, $\gamma(j) = j-1$ if $2 \le j < k$, and $\gamma(j) = j + 1$ if $j \ge k$. So, it only require to compute at most K membership tests. After the initialization, the computation of each round t consists of one call to **FIND-BASE** for computing the action $x(t)$, the update of $\hat\mu_k(t)$ and $N_k(t)$ for each $k \in \mathrm{supp}\,(x(t))$, and one call to **UPDATE-FEATURE** for updating the feature of each arm $k \in \mathrm{supp}\,(x(t))$ stored in the instances of the dynamic maxium-weight base algorithm.

Algorithm 5 `FasterCUCB`

Input: the total number of rounds T, λ_t, and $m \in \mathbb{N}$
Initialization:
    $t = 0$;
    while $\exists k \in [K]$ *such that* $N_k(t) = 0$ **do**
      | Pull $i^\star(e_k)$; $t = t + 1$;
    end
    INITIALIZE$\left(a, b, \frac{1}{\sqrt{T}}, 1, (\hat\mu_k(t), N_k(t))_{k\in[K]}, \frac{1}{\log^m T}\right)$
while $t < T$ **do**
    $x(t) \leftarrow$**FIND-BASE**$((1, \lambda_t))$;
    Pull $x(t)$ and receive $y_k(t) \sim \nu_k$ for each $k \in \mathrm{supp}\,(x(t))$;
    for $k \in \mathrm{supp}\,(x(t))$ **do**
      | $N_k(t) \leftarrow N_k(t-1) + 1$;
      | $\hat\mu_k(t) \leftarrow \frac{t-1}{t}\hat\mu_k(t-1) + \frac{1}{t}y_k(t)$;
      | **UPDATE-FEATURE**$\left(k, \left(\mu_k(t), \frac{1}{\sqrt{N_k(t)}}\right)\right)$;
    end
    $t = t + 1$;
end

5.1. Per-round Time Complexity

By Theorem 4.4, one call to **FIND-BASE** takes $\mathcal{O}(\,\mathrm{poly}\,(W) + D)$ and D calls to **UPDATE-FEATURE** take $\mathcal{O}(D \,\mathrm{poly}\,(W)\, \mathcal{T}_{\mathrm{update}}(\mathcal{A}; \frac{\epsilon}{3}))$. Since

$$W = \mathcal{O}\left(\log^m T \log\left(\frac{b}{a}\sqrt{T}\right)\right) = \mathcal{O}\left(\log^{m+1} T\right),$$

the per-round time complexity of Algorithm 5 is

$$\mathcal{O}\left(D \,\mathrm{polylog}\,(T)\, \mathcal{T}_{\mathrm{update}}\left(\mathcal{A}; \frac{\epsilon}{3}\right)\right)$$

. Here, we will set $\epsilon = \frac{1}{\log^m T}$ for the regret analysis.

5.2. Regret Upper Bound

Notation. Fix $\mu \in \Lambda$ and $i^\star \in \mathrm{argmax}_{x \in \mathcal{X}} \langle \mu, x \rangle$. We introduce a few notation. Let $\{\bar{j}\}_{j=1}^D$ be the permutation of $\mathrm{supp}\,(i^\star)$ such that $\mu_{\bar 1} \ge \cdots \ge \mu_{\bar D}$. Define $\triangle_{j,k} \triangleq \mu_j - \mu_k$ and $d_k \triangleq \max\{j \in [D] : \triangle_{\bar j, k} > 0\}$ for $j \in \mathrm{supp}\,(i^\star)$ and $k \notin \mathrm{supp}\,(i^\star)$, and $\triangle_{\min} \triangleq \min_{k \notin \mathrm{supp}\,(i^\star)} \triangle_{\overline{d_k}, k}$.

Theorem 5.1. *Let* $\lambda_t = \sqrt{1.5(b-a)^2 \log t}$ *and* $m \in \mathbb{N}$. *Define* $T_0 \triangleq \max\{K, \exp((\frac{b}{\triangle_{\min}})^{\frac{1}{m}})\}$. *For* $T \in \mathbb{N}$, *the expected regret of Algorithm 5 is upper bounded by*

$$R(T) \le \sum_{k \notin supp(i^\star)} \left(\sum_{j=1}^{d_k} \triangle_{\bar j, k} T_0 + \frac{12\triangle_{\overline{d_k}, k}(b-a)^2 \log T}{\left(\frac{\mu_{\overline{d_k}}}{1 + \log^{-m} T} - \mu_k\right)^2} \right)$$
$$+ \sum_{k \notin supp(i^\star)} \sum_{j=1}^{d_k} \triangle_{\bar j, k}\left(\frac{1}{T} + \frac{\pi^2}{6}\right) + DbT_0.$$

As a consequence of Theorem 5.1, setting $T \to \infty$ yields:

$$\lim_{T\to\infty} \frac{R(T)}{\log T} \le \sum_{k \notin supp(i^\star)} \frac{12(b-a)^2}{\triangle_{\overline{d_k}, k}} \le \mathcal{O}\left(\frac{K-D}{\triangle_{\min}}\right),$$

which matches Theorem 4 in (Kveton et al., 2014a), $\liminf_{T\to\infty} \frac{R(T)}{\log T} = \Omega(\frac{K-D}{\triangle_{\min}})$, asymptotically up to a constant factor. Note that `FasterCUCB` is faster than `CUCB` when $\triangle_{\min} = \Omega(\frac{1}{\mathrm{polylog}(K)})$ and when $T = \mathrm{poly}\,(K)$. Also, similar to (Cuvelier et al., 2021b), our per-round time complexity also goes to infinity as $T \to \infty$, one way to address this issue is to use `CUCB` when the per-round time complexity of ours is larger than that of `CUCB`.

Useful lemmas. Here we present two lemmas that will be used to show Theorem 5.1 in Section 5.3. First, inspired by Kveton et al. (2014a), we define a bijection g_t from $\mathrm{supp}\,(i^\star)$ to $\mathrm{supp}\,(x(t))$ with the following properties:

Lemma 5.2. *There exists a bijection* $g_t : \mathrm{supp}\,(i^\star) \to \mathrm{supp}\,(x(t))$ *such that (i)* $g_t(j) = j$ *for* $j \in \mathrm{supp}\,(i^\star) \cap \mathrm{supp}\,(x(t))$; *(ii) for any* $j \in \mathrm{supp}\,(i^\star) \setminus \mathrm{supp}\,(x(t))$,

$$x_{g_t(j)}(t) = 1 \implies \left\langle \mathrm{dom}(f_{g_t(j)}), h \right\rangle \ge \frac{\langle \mathrm{dom}(f_j), h \rangle}{1 + \frac{1}{3\log^m T}}.$$

The proof of Lemma 5.2 is in Appendix E.1, where an explicit construction of g_t is provided. Property (i) allows us to decompose the instantaneous regret $\langle i^\star - x(t), \mu \rangle = \sum_{k \notin \mathrm{supp}(i^\star)} \sum_{j \in \mathrm{supp}(i^\star)} \triangle_{j,k} \mathbb{1}\{g_t(j) = k\}$, and Property (ii), Lemma 5.2, allows us to derive a bound of $\sum_{t=1}^T \mathbb{1}\{g_t(j) = k\}$ that relates with UCB values.

Second, for technical reasons, we need the precision parameter $\epsilon = \log^{-m} T$ to be small enough so that $\triangle_{i,j}$ and $\mu_i - (1+\epsilon)\mu_j$ have the same sign. The below lemma (proved in Appendix E.2) gives the threshold to make it happen:

Lemma 5.3. *Let* $\epsilon < \frac{\triangle_{\min}}{b}$. *Then, for any* $i \in \mathrm{supp}\,(i^\star)$ *and any* $j \notin \mathrm{supp}\,(i^\star)$, $\mu_i - \mu_j > 0 \implies \frac{\mu_i}{1+\epsilon} - \mu_j > 0$.

5.3. Proof of Theorem 5.1

For $T \le T_0$, $R(T)$ is trivially bounded by $T\langle \mu, i^\star \rangle \le DbT_0$. In the following, we assume $T > T_0$.

As g_t is a bijection from $\mathrm{supp}\,(\boldsymbol{i}^\star)$ to $\mathrm{supp}\,(\boldsymbol{x}(t))$ and $g_t(j) = j$ for $j \in \mathrm{supp}\,(\boldsymbol{i}^\star) \cap \mathrm{supp}\,(\boldsymbol{x}(t))$, we can rewrite

$$\mathbb{E}[\langle \boldsymbol{i}^\star - \boldsymbol{x}(t), \boldsymbol{\mu}\rangle] = \sum_{k \notin \mathrm{supp}\,(\boldsymbol{i}^\star)} \sum_{j \in \mathrm{supp}\,(\boldsymbol{i}^\star)} \triangle_{j,k} \mathbb{E}[\mathbb{1}\{g_t(j) = k\}]$$

$$\leq \sum_{k \notin \mathrm{supp}\,(\boldsymbol{i}^\star)} \sum_{j=1}^{d_k} \triangle_{\bar{j},k} \mathbb{E}\left[\mathbb{1}\{g_t(\bar{j}) = k\}\right]$$

so that the expected regret is bounded from the above by:

$$R(T) \leq \sum_{k \notin \mathrm{supp}\,(\boldsymbol{i}^\star)} \sum_{j=1}^{d_k} \triangle_{\bar{j},k} \mathbb{E}\left[\sum_{t=1}^{T} \mathbb{1}\{g_t(\bar{j}) = k\}\right]$$

$$= \sum_{k \notin \mathrm{supp}\,(\boldsymbol{i}^\star)} \sum_{j=1}^{d_k} \triangle_{\bar{j},k}\left((I)_{\bar{j},k} + (II)_{\bar{j},k}\right),$$

where $\begin{cases} (I)_{\bar{j},k} = \sum_{t=1}^{T} \mathbb{E}\left[\mathbb{1}\{g_t(\bar{j}) = k, N_k(t) \leq n_{\bar{j},k}\}\right] \\ (II)_{\bar{j},k} = \sum_{t=1}^{T} \mathbb{E}\left[\mathbb{1}\{g_t(\bar{j}) = k, N_k(t) > n_{\bar{j},k}\}\right] \end{cases}$

and $n_{j,k} = \max\left\{\frac{6(b-a)^2 \log T}{(\frac{\mu_j}{1+\log^{-m} T} - \mu_k)^2}, T_0\right\}$. The proof is completed by bounding related terms of $(I)_{\bar{j},k}$ and $(II)_{\bar{j},k}$ by Lemma 5.4 (proved in Appendix E.2) and Lemma 5.5.

Lemma 5.4. *Let* $k \notin \mathrm{supp}\,(\boldsymbol{i}^\star)$ *and* $j \in [d_k]$. *For* $T > T_0$,

$$\sum_{j=1}^{d_k} \triangle_{\bar{j},k}(I)_{\bar{j},k} \leq \sum_{j=1}^{d_k} \triangle_{\bar{j},k} T_0 + \frac{12(b-a)^2 \triangle_{\overline{d_k},k} \log T}{(\frac{\mu_{\overline{d_k}}}{1+\log^{-m} T} - \mu_k)^2}.$$

Lemma 5.5. *Let* $k \notin \mathrm{supp}\,(\boldsymbol{i}^\star)$ *and* $j \in [d_k]$. *For* $T > T_0$,

$$(II)_{\bar{j},k} \leq \frac{1}{T} + \frac{\pi^2}{6}.$$

Proof sketch: See Appendix E.2 for the entire proof. Let $\epsilon \triangleq \frac{1}{\log^m T}$. First, we claim:

$$g_t(\bar{j}) = k \implies u_k(N_k(t-1), T) \geq \frac{\min_{s<t} u_{\bar{j}}(s,t)}{1+\epsilon}, \tag{14}$$

where $u_k(s,t) = \tilde{\mu}_k(s) + \frac{\lambda_t}{\sqrt{s}}$ and $\tilde{\mu}_k(t) = \frac{1}{t}\sum_{s=1}^{t} y_k(s)$.

<u>Show Eq. (14):</u> Observe that $g_t(\bar{j}) = k$ implies:

$$\left(1 + \frac{\epsilon}{3}\right)\langle \boldsymbol{f}_k, \boldsymbol{q}\rangle \geq \langle \mathrm{dom}(\boldsymbol{f}_k), \boldsymbol{q}\rangle$$

$$\geq \frac{\langle \mathrm{dom}(\boldsymbol{f}_{\bar{j}}), \boldsymbol{q}\rangle}{1 + \frac{\epsilon}{3}} \geq \frac{\langle \boldsymbol{f}_{\bar{j}}, \boldsymbol{q}\rangle}{1 + \frac{\epsilon}{3}}, \tag{15}$$

where Eq. (8) is used in the first and the last inequality, and the second uses Lemma 5.2 and Corollary 4.10. By $(1+\frac{\epsilon}{3})^2 \leq 1+\epsilon$ and expanding $\boldsymbol{f}_k = (\hat{\mu}_k(t-1), \frac{1}{\sqrt{N_k(t-1)}})$ and $\boldsymbol{q} = (1, \lambda_t)$, we derive from (15) that:

$$u_k(N_k(t-1), t) \geq \frac{u_{\bar{j}}(N_{\bar{j}}(t-1), t)}{1+\epsilon},$$

and further by $\log T > \log t$ and $N_{\bar{j}}(t-1) \in [t-1]$,

$$u_k(N_k(t-1), T) \geq \frac{u_{\bar{j}}(N_{\bar{j}}(t-1), t)}{1+\epsilon} \geq \frac{\min_{s<t} u_{\bar{j}}(s,t)}{1+\epsilon},$$

which shows Eq. (14). Second, define

$$\mathcal{T}_{\bar{j},k} = \{t \in \{n_{\bar{j},k}+1, \cdots, T\} : g_t(\bar{j}) = k, N_k(t-1) > n_{\bar{j},k}\}.$$

From Eq. (14), $(II)_{\bar{j},k}$ is upper bounded by

$$\mathbb{E}\left[\sum_{t \in \mathcal{T}_{\bar{j},k}} \mathbb{1}\left\{u_k(N_k(t-1), T) \geq \frac{\min_{s<t}\{u_{\bar{j}}(s,t)\}}{1+\epsilon}\right\}\right]$$

$$\leq \mathbb{E}\left[\sum_{t \in \mathcal{T}_{\bar{j},k}} \sum_{s<t} (\mathbb{1}\{\mathcal{A}_{1,t,s}\} + \mathbb{1}\{\mathcal{A}_{2,t,s}\} + \mathbb{1}\{\mathcal{A}_{3,t,s}\})\right], \tag{16}$$

where $\begin{cases} \mathcal{A}_{1,t,s} = \left\{\tilde{\mu}_k(N_k(t-1)) \geq \mu_k + \frac{\lambda_T}{\sqrt{N_k(t-1)}}\right\} \\ \mathcal{A}_{2,t,s} = \left\{\mu_{\bar{j}} \geq \tilde{\mu}_{\bar{j}}(s) + \frac{\lambda_t}{\sqrt{s}}\right\} \\ \mathcal{A}_{3,t,s} = \left\{\mu_k + \frac{2\lambda_T}{\sqrt{N_k(t-1)}} > \frac{\mu_{\bar{j}}}{1+\epsilon}\right\} \end{cases}$.

See Appendix E.2 for the derivation of Eq. (16). Observe when $t \in \mathcal{T}_{\bar{j},k}$,

$$\mathbb{1}\{\mathcal{A}_{3,t,s}\} \leq \mathbb{1}\left\{\mu_k + \frac{2\lambda_T}{\sqrt{n_{\bar{j},k}+1}} > \frac{\mu_{\bar{j}}}{1+\epsilon}\right\} = 0,$$

where the inequality is because $N_k(t-1) > n_{\bar{j},k}$, and the equality is because

$$n_{\bar{j},k} \geq \frac{4\lambda_T^2}{(\frac{\mu_{\bar{j}}}{1+\epsilon} - \mu_k)^2} \implies \frac{4\lambda_T^2}{n_{\bar{j},k}+1} < \left(\frac{\mu_{\bar{j}}}{1+\epsilon} - \mu_k\right)^2,$$

and also $\frac{\mu_{\bar{j}}}{1+\epsilon} - \mu_k > 0$ which is ensured by Lemma 5.3 as $T > T_0$. Finally, from Eq. (16) and using Hoeffding's inequality, we get

$$(II)_{\bar{j},k} \leq \mathbb{E}\left[\sum_{t \in \mathcal{T}_{\bar{j},k}} \sum_{s<t} (\mathbb{1}\{\mathcal{A}_{1,t,s}\} + \mathbb{1}\{\mathcal{A}_{2,t,s}\})\right]$$

$$\leq \sum_{t=n_{\bar{j},k}+1}^{T} \sum_{s<t} \left(e^{-3\log T} + e^{-3\log t}\right).$$

See Appendix E.2 for the derivation of the second inequality. The proof is completed by evaluating

$$\begin{cases} \sum_{t=1}^{T} \sum_{s<t} e^{-3\log T} \leq \sum_{t=1}^{T} \frac{t}{T^3} \leq \frac{T(T+1)}{2T^3} \leq \frac{1}{T} \\ \sum_{t=1}^{T} \sum_{s<t} e^{-3\log t} \leq \sum_{t=1}^{\infty} \frac{t}{t^3} \leq \sum_{t=1}^{\infty} \frac{1}{t^2} \leq \frac{\pi^2}{6} \end{cases}.$$

$\square$

6. Conclusion

In this paper, we have presented `FasterCUCB`, the first sublinear-time algorithm for matroid semi-bandits. Several possible future directions. First, one might extend our approach to speed up UCB-style algorithms for different problems such as combinatorial best-arm identification (Chen et al., 2014; Du et al., 2021) and nonstationary semi-bandits (Zhou et al., 2020; Chen et al., 2021). Second, another direction is to study the possibility of speeding up other forms of weights, such as those derived from gradients (Tzeng et al., 2023) and those in the follow-the-perturbed-leader algorithm (Neu and Bartók, 2016).

Acknowledgement

Impact Statement

This paper develops an algorithm for matroid semi-bandits. The societal consequences of this work indirectly come from the applications of matroid semi-bandits, and none of which we think should be discussed here.

References

Zeinab Abbassi, Vahab S Mirrokni, and Mayur Thakur. Diversity maximization under matroid constraints. In *Proc. of KDD*, 2013.

Alper Atamtürk and Andrés Gómez. Maximizing a class of utility functions over the vertices of a polytope. *Operations Research*, 2017.

Sébastian Bubeck, Tengyao Wang, and Nitin Viswanathan. Multiple identifications in multi-armed bandits. In *Proc. of ICML*, 2013.

Lijie Chen, Anupam Gupta, and Jian Li. Pure exploration of multi-armed bandit under matroid constraints. In *Proc. of COLT*, 2016.

Shouyuan Chen, Tian Lin, Irwin King, Michael R Lyu, and Wei Chen. Combinatorial pure exploration of multi-armed bandits. In *Proc. of NeurIPS*, 2014.

Wei Chen, Yajun Wang, and Yang Yuan. Combinatorial multi-armed bandit: General framework and applications. In *Proc. of ICML*, 2013.

Wei Chen, Liwei Wang, Haoyu Zhao, and Kai Zheng. Combinatorial semi-bandit in the non-stationary environment. In *Proc. of UAI*, 2021.

Sayak Ray Chowdhury, Gaurav Sinha, Nagarajan Natarajan, and Amit Sharma. Combinatorial categorized bandits with expert rankings. In *Proc. of UAI*, 2023.

Richard Combes, Mohammad Sadegh Talebi Mazraeh Shahi, Alexandre Proutiere, and Marc Lelarge. Combinatorial bandits revisited. In *Proc. of NeurIPS*, 2015.

Richard Combes, Stefan Magureanu, and Alexandre Proutiere. Minimal exploration in structured stochastic bandits. In *Proc. of NeurIPS*, 2017.

Thibaut Cuvelier, Richard Combes, and Eric Gourdin. Asymptotically optimal strategies for combinatorial semi-bandits in polynomial time. In *Proc. of ALT*, 2021a.

Thibaut Cuvelier, Richard Combes, and Eric Gourdin. Statistically efficient, polynomial-time algorithms for combinatorial semi-bandits. In *Proc. of SIGMETRICS*, 2021b.

Rémy Degenne and Vianney Perchet. Combinatorial semi-bandit with known covariance. In *Proc. of NeurIPS*, 2016.

Yihan Du, Yuko Kuroki, and Wei Chen. Combinatorial pure exploration with full-bandit or partial linear feedback. In *Proc. of AAAI*, 2021.

Jack Edmonds. Matroids and the greedy algorithm. *Mathematical Programming*, 1971.

David Eppstein, Zvi Galil, Giuseppe F. Italiano, and Amnon Nissenzweig. Sparsification—a technique for speeding up dynamic graph algorithms. *Journal of the ACM*, 1997.

Greg N. Frederickson. Data structures for on-line updating of minimum spanning trees, with applications. *SIAM Journal on Computing*, 1985.

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

Todd L Graves and Tze Leung Lai. Asymptotically efficient adaptive choice of control laws incontrolled markov chains. *SIAM Journal on Control and Optimization*, 1997.

Manoj Gupta and Richard Peng. Fully dynamic $(1 + \epsilon)$-approximate matchings. In *Proc. of FOCS*, 2013.

Monika Henzinger, Paul Liu, Jan Vondrák, and Da Wei Zheng. Faster submodular maximization for several classes of matroids. In *Proc. of ICALP*, 2023.

Jacob Holm, Kristian De Lichtenberg, and Mikkel Thorup. Poly-logarithmic deterministic fully-dynamic algorithms for connectivity, minimum spanning tree, 2-edge, and biconnectivity. *Journal of the ACM*, 2001.

Shinji Ito. Hybrid regret bounds for combinatorial semi-bandits and adversarial linear bandits. In *Proc. of NeurIPS*, 2021.

Kwang-Sung Jun, Aniruddha Bhargava, Robert Nowak, and Rebecca Willett. Scalable generalized linear bandits: Online computation and hashing. In *Proc. of NeurIPS*, 2017.

Satyen Kale, Lev Reyzin, and Robert E Schapire. Non-stochastic bandit slate problems. In *Proc. of NeurIPS*, 2010.

Jon Kleinberg and Éva Tardos. *Algorithm design*. Pearson Education India, 2006.

Fang Kong, Yueran Yang, Wei Chen, and Shuai Li. The hardness analysis of Thompson sampling for combinatorial semi-bandits with greedy oracle. In *Proc. of NeurIPS*, 2021.

Branislav Kveton, Zheng Wen, Azin Ashkan, Hoda Eydgahi, and Brian Eriksson. Matroid bandits: Fast combinatorial optimization with learning. In *Proc. of UAI*, 2014a.

Branislav Kveton, Zheng Wen, Azin Ashkan, and Michal Valko. Learning to act greedily: Polymatroid semi-bandits. *arXiv preprint arXiv:1405.7752*, 2014b.

Branislav Kveton, Zheng Wen, Azin Ashkan, and Csaba Szepesvari. Tight regret bounds for stochastic combinatorial semi-bandits. In *Proc. of AISTATS*, 2015.

Gergely Neu and Gábor Bartók. Importance weighting without importance weights: An efficient algorithm for combinatorial semi-bandits. *JMLR*, 2016.

James G Oxley. Matroid theory, 2011.

Orestis Papadigenopoulos and Constantine Caramanis. Recurrent submodular welfare and matroid blocking semi-bandits. In *Proc. of NeurIPS*, 2021.

Pierre Perrault. When combinatorial Thompson sampling meets approximation regret. In *Proc. of NeurIPS*, 2022.

Pierre Perrault, Vianney Perchet, and Michal Valko. Exploiting structure of uncertainty for efficient matroid semi-bandits. In *Proc. of ICML*, 2019.

Pierre Perrault, Etienne Boursier, Michal Valko, and Vianney Perchet. Statistical efficiency of thompson sampling for combinatorial semi-bandits. In *Proc. of NeurIPS*, 2020a.

Pierre Perrault, Michal Valko, and Vianney Perchet. Covariance-adapting algorithm for semi-bandits with application to sparse outcomes. In *Proc. of COLT*, 2020b.

Alexander Schrijver. *Combinatorial Optimization: Polyhedra and Efficiency*. Springer, 2003.

Matthew Streeter, Daniel Golovin, and Andreas Krause. Online learning of assignments. In *Proc. of NeurIPS*, 2009.

Mohammad Sadegh Talebi and Alexandre Proutiere. An optimal algorithm for stochastic matroid bandit optimization. In *Proc. of AAMAS*, 2016.

Taira Tsuchiya, Shinji Ito, and Junya Honda. Further adaptive best-of-both-worlds algorithm for combinatorial semi-bandits. In *Proc. of AISTATS*, 2023.

Ruo-Chun Tzeng, Po-An Wang, Alexandre Proutiere, and Chi-Jen Lu. Closing the computational-statistical gap in best arm identification for combinatorial semi-bandits. In *Proc. of NeurIPS*, 2023.

Jan van den Brand, Danupon Nanongkai, and Thatchaphol Saranurak. Dynamic matrix inverse: Improved algorithms and matching conditional lower bounds. In *Proc. of FOCS*, 2019.

Siwei Wang and Wei Chen. Thompson sampling for combinatorial semi-bandits. In *Proc. of ICML*, 2018.

Zheng Wen, Branislav Kveton, and Azin Ashkan. Efficient learning in large-scale combinatorial semi-bandits. In *Proc. of ICML*, 2015.

Shuo Yang, Tongzheng Ren, Sanjay Shakkottai, Eric Price, Inderjit S Dhillon, and Sujay Sanghavi. Linear bandit algorithms with sublinear time complexity. In *Proc. of ICML*, 2022.

Huozhi Zhou, Lingda Wang, Lav Varshney, and Ee-Peng Lim. A near-optimal change-detection based algorithm for piecewise-stationary combinatorial semi-bandits. In *Proc. of AAAI*, 2020.

A. Notation

Problem setting	
K	the number of arms
$\mathcal{X}$	the bases of the given matroid $([K], \mathcal{I})$
D	$\max_{\boldsymbol{x} \in \mathcal{X}} \|\boldsymbol{x}\|_0$
$\boldsymbol{\mu}$	the mean vector of the K arms $\nu_1, \cdots, \nu_K$
$\boldsymbol{i}^\star(\boldsymbol{\mu})$	an action attaining $\max_{\boldsymbol{x} \in \mathcal{X}} \langle \boldsymbol{\mu}, \boldsymbol{x} \rangle$

Notation related to `FasterCUCB`	
$N_k(t)$	the number of arm pulls of arm k
$\boldsymbol{x}(t)$	the action selected by the algorithm at round t
$\boldsymbol{y}(t)$	the reward vector at round t
$\hat{\mu}_k(t)$	the empirical reward $\frac{1}{N_k(t)} \sum_{s=1}^t y_k(t) \mathbb{1}\{x_k(t) = 1\}$ of arm k at round t
λ_t	the parameter that controls the confidence interval

Notation related to dynamic algorithm	
$\boldsymbol{f}_k = (\alpha_k, \beta_k)$	a nonnegative two-dimensional feature of arm k
$\boldsymbol{q}$	a nonnegative two-dimensional query
$(\alpha_{\text{lb}}, \alpha_{\text{ub}})$	lower and upper bounds of α_k's
$(\beta_{\text{lb}}, \beta_{\text{ub}})$	lower and upper bounds of β_k's
W	(the square root of) the number of bins
$\text{BIN}_{q,r}$	bins that partition the possible region of the features
$\text{dom}_{q,r}$	dominating point of $\text{BIN}_{q,r}$
$\text{dom}(\boldsymbol{f}_k)$	dominating point of $\text{BIN}_{q,r}$ to which $\boldsymbol{f}_k$ belongs
$\mathcal{L} = \{\boldsymbol{l}_1, \ldots, \boldsymbol{l}_{\binom{K}{2}}\}$	the set of $\binom{K}{2}$ lines, each of which is orthogonal to line $\overleftrightarrow{\boldsymbol{f}_k \boldsymbol{f}_{k'}}$ for some $k \neq k'$ and intersects $\boldsymbol{0}$
$\mathcal{H}$	a minimum hitting set of the cells in arrangement of $\mathcal{L}$

Notation related to regret analysis	
$\{\bar{j}\}_{j=1}^D$	the permutation of $\text{supp}(\boldsymbol{i}^\star)$ such that $\mu_{\bar{1}} \geq \cdots \geq \mu_{\bar{D}}$
ϵ	the precision parameter which is set to $\frac{1}{\log^m T}$ in `FasterCUCB` (Algorithm 5)
$g_t(j)$	the mapping from $\text{supp}(\boldsymbol{i}^\star)$ to $\text{supp}(\boldsymbol{x}(t))$ such that
	(i) $g_t(j) = j$ if $j \in \text{supp}(\boldsymbol{i}^\star) \cap \text{supp}(\boldsymbol{x}(t))$
	(ii) $x_{g_t(j)}(t) = 1$ implies $\left\langle \text{dom}(\boldsymbol{f}_{g_t(j)}), \boldsymbol{h} \right\rangle \geq \frac{1}{1+\frac{\epsilon}{3}} \left\langle \text{dom}(\boldsymbol{f}_j), \boldsymbol{h} \right\rangle$
$\triangle_{j,k}$	the difference $\mu_j - \mu_k$ between arm j's and arm k's expected reward
$\triangle_{\min}$	the smallest positive gap $\triangle_{i,j}$ between any pair of $i \in \text{supp}(\boldsymbol{i}^\star)$ and $j \notin \text{supp}(\boldsymbol{i}^\star)$
d_k	the largest $j \in [D]$ such that $\triangle_{\delta(j),k} > 0$
$\tilde{\mu}_k(t)$	the average $\frac{1}{t} \sum_{s=1}^t y_k(s)$ of rewards of arm k in the first t rounds
$u_k(s,t)$	the UCB value of $\tilde{\mu}_k(s) + \frac{\lambda_t}{\sqrt{s}}$ under s samples of arm k and with confidence parameter λ_t

Table 2. Table of notation.

B. Further Related Works

In this section, we review relevant literatures on combinatorial semi-bandits and sublinear-time bandits. We focus on the stochastic setting. For ease of comparison, we assume the best action $i^\star \in \operatorname{argmax}_{x \in \mathcal{X}} \langle x, \mu \rangle$ is unique, and define $\triangle_{\min} \triangleq \min_{j,k \in [K]: i_j^\star = 1, i_k^\star = 0, \mu_j - \mu_k > 0} (\mu_j - \mu_k)$, and $\triangle \triangleq \min_{x \neq i^\star : \langle i^\star - x, \mu \rangle > 0} \langle i^\star - x, \mu \rangle$.

Matroid semi-bandits. Kveton et al. (2014a) showed an instance such that any uniformly good algorithm[2] suffer $R(T) = \Omega\left(\frac{(K-D)\log T}{\triangle_{\min}}\right)$. They also showed that CUCB (Gai et al., 2012; Chen et al., 2013) have a regret bound scaling as $\mathcal{O}\left(\frac{(K-D)\log T}{\triangle_{\min}}\right)$. Talebi and Proutiere (2016) showed an instance-specific lower bound $\liminf_{T \to \infty} \frac{R(T)}{\log T} \geq c(\mu)$ for uniformly good algorithms, where $c(\mu)$ is the optimum of a semi-infinite linear program (Graves and Lai, 1997; Combes et al., 2015), and proposed KL-OSM whose regret upper bound matches this lower bound. The per-round compleixty of KL-OSM is K line search for generating the indices plus the time for solving a linear maximization. Both CUCB and KL-OSM rely on the greedy algorithm (Algorithm 1) to solve the linear maximization for determing the action to be pulled. The time complexity of the greedy algorithm is upper bounded by $\mathcal{O}(K(\log K + \mathcal{T}_{\mathrm{member}}))$ time and lower bounded by $\Omega(K)$. (Perrault et al., 2019) showed that the sampling rule of many combinatorial semi-bandit algorithms is a maximization problem over a summation of a linear function and a submodular function, and proposed two efficient algorithms for matroid semi-bandits: One is based on local search and the other is a greedy algorithm. Both have per-round time complexity at least $\Omega(KD)$. In contrast, our FasterCUCB is the first matroid semi-bandit algorithm with per-round time complexity sublinear in K for many classes of matroids.

Combinatorial semi-bandits. Here, we review works that focus on the standard setting of stochastic combinatorial semi-bandits. These consider a linear reward function and any action sets $\mathcal{X}$, where linear maximization $\max_{x \in \mathcal{X}} \langle x, v \rangle$ for any $v \in \mathbb{R}^K$ can be solved in time polynomial in K. We omit the discussion on works that focus on a specific action set (Chowdhury et al., 2023), with additional structural assumptions on the rewards (Wen et al., 2015; Perrault et al., 2020b), or with a different reward function (Papadigenopoulos and Caramanis, 2021). Perrault et al. (2020a) showed that CTS has a regret bound of $\mathcal{O}(\frac{K \log^2 D \log T}{\triangle})$ for mutually independent gaussian rewards and a regret bound of $\mathcal{O}(\frac{KD \log^2 D \log T}{\triangle})$ for correlated gaussian rewards. Perrault (2022) sharpen the regret bound of CTS for the case of mutually independent gaussian rewards to be $\mathcal{O}(\frac{K \log D \log T}{\triangle})$. The per-round time complexity of CTS is at least $\Omega(K)$ due to sampling from the posterior distributions. Degenne and Perchet (2016) showed that ESCB2 has regret bound of $\mathcal{O}(\frac{K \log^2 D \log T}{\triangle})$ for independent subgaussian rewards, but its sampling rule is NP-hard (Atamtürk and Gómez, 2017) to optimize. Cuvelier et al. (2021b) proposed AESCB that approximates ESCB2 with per-round time complexity of $\mathcal{O}(KD \log^3 K \operatorname{poly}(\log T))$ while maintaining the same regret bound. Their technique is based on rounding and budgeted-linear maximization. OSSB (Combes et al., 2017) is an asymptotically instance-specifically optimal algorithm for general structured bandits, including combinatorial semi-bandits, but at each round, it requires to solve a semi-infinite linear program (Graves and Lai, 1997). Cuvelier et al. (2021a) developed a method that runs in time polynomial in K to solve the semi-infinite linear program for Gaussian rewards. They managed to maintain OSSB's asymptotic optimality for m-sets, but not for spanning trees and bipartite matchings. Ito (2021) and Tsuchiya et al. (2023) proposed algorithms based on the optimistic FTRL framework that achieve $\mathcal{O}(\frac{KD \log T}{\triangle})$ regret in the stochastic setting and $\mathcal{O}(\sqrt{KDT \log T})$ in the adversarial setting. At each round, the proposed algorithms first use FTRL rule to obtain a vector $a(t)$ in the convex hull of $\mathcal{X}$ and then sample an action $x(t)$ based on $a(t)$. Tsuchiya et al. (2023) mentioned that the computational efficiency of the sampling step has long been a problem in semi-bandits using the optimistic FTRL framework.

Sublinear-time linear bandits. Several works (Jun et al., 2017; Yang et al., 2022) focusing on making per-round complexity of linear bandits sublinear in the number of arms. Maximum Inner Product Search (MIPS) is the primary tool used to design such algorithms. For N arms in $\mathbb{R}^d$, Q-GLOC (Jun et al., 2017) achieves a high-probability regret bound of $\tilde{\mathcal{O}}(d^{\frac{5}{4}}\sqrt{T})$ and per-round time complexity of $\tilde{\mathcal{O}}(d^2 N^\rho \log N)$ for some $\rho \in (0,1)$, where $\tilde{\mathcal{O}}$ hides polylogarithmic factors in T and d. Yang et al. (2022) considered the setting with arms addition (resp. addition and deletion), and proposed Sub-Elim (resp. Sub-TS), which has a high-probability regret bound of $\tilde{\mathcal{O}}(d\sqrt{T})$ (resp. $\tilde{\mathcal{O}}(d^{\frac{3}{2}}\sqrt{T})$) and per-round time complexity of $N^{1-\Theta(\frac{1}{T^2 \log^2 T})}$ (resp. $N^{1-\Theta(\frac{1}{T})}$). These results are applicable to our setting with $d = K$ and $N = |\mathcal{X}|$. Q-GLOC (Jun et al., 2017) applied to our setting has regret bound of $\tilde{O}(K^{\frac{5}{4}}\sqrt{T})$ and per-round time compleixty $\tilde{\mathcal{O}}(K^2 |\mathcal{X}|^\rho)$.

---

[2]A uniformly good algorithm has the expected regret $R(T) = o(T^\alpha)$ hold for any $\alpha > 0$.

`Sub-Elim` (resp. `Sub-TS`) Yang et al. (2022) applied to our setting has regret bound of $\tilde{\mathcal{O}}(K\sqrt{T})$ (resp. $\tilde{\mathcal{O}}(K^{\frac{3}{2}}\sqrt{T})$) and has per-round time complexity of $|\mathcal{X}|^{1-\Theta(\frac{1}{T^2 \log T})}$ (resp. $|\mathcal{X}|^{1-\Theta(\frac{1}{T})}$). These results have worse regret bounds than what we have obtained, and their per-round time complexity can be exponential in K.

C. Membership Oracles for Different Matroids

In this section, we discuss $\mathcal{T}_{\mathrm{member}}$ for the matroids shown in Section 2.

- For uniform matroid, the membership oracle is given $\boldsymbol{x} \in \mathcal{I}$ and $k \in [K] \backslash \mathrm{supp}\,(\boldsymbol{x})$, and has to check whether $|\,\mathrm{supp}\,(\boldsymbol{x} + \boldsymbol{e}_k)| \leq D$. Suppose the number $n = |\,\mathrm{supp}\,(\boldsymbol{x})|$ is maintained. Then, it takes $\mathcal{O}(1)$ time to check whether $n + 1 \leq D$, and hence $\mathcal{T}_{\mathrm{member}} = \mathcal{O}(1)$.

- For partition matroids, the membership oracle is given $\boldsymbol{x} \in \mathcal{I}$ and $k \in [K] \backslash \mathrm{supp}\,(\boldsymbol{x})$, and has to check whether $|\,\mathrm{supp}\,(\boldsymbol{x} + \boldsymbol{e}_k) \cap S_i| \leq 1$ for all $i \in [D]$. Suppose there is an integer array A of size K such that $j \in S_{A[j]}$ for each $j \in [K]$, and suppose there is an integer array B of size D such that $B[i] = \sum_{j \in \mathrm{supp}(\boldsymbol{x})} \mathbb{1}\{j \in S_i\}$ for each $i \in [D]$. Then, to decide whether whether $\boldsymbol{x} + \boldsymbol{e}_k \in \mathcal{I}$, it only requires to check whether $B[A[k]] + 1 \leq 1$. This can be implemented in $\mathcal{O}(1)$ time, and thus $\mathcal{T}_{\mathrm{member}} = \mathcal{O}(1)$.

- For graphical matroids, the membership oracle has to detect if there is a cycle. Using the union-find data structure, whether $\mathrm{supp}\,(\boldsymbol{x}) \cup \{k\}$ has a cycle can be detected in $\mathcal{O}(\log K)$ time, so we have $\mathcal{T}_{\mathrm{member}} = \mathcal{O}(\log K)$. Refer to Section 4.6 in (Kleinberg and Tardos, 2006) for more detailed explanation.

- For transversal matroids, there is little discussion about its membership oracle. Here we present an implementation to answer a query $(\boldsymbol{x}, k)$ about whether $\boldsymbol{x} + \boldsymbol{e}_k \in \mathcal{I}$, where $\boldsymbol{x} \in \mathcal{I}$ and $k \in [K] \backslash \mathrm{supp}\,(\boldsymbol{x})$. Suppose a maximum matching M on $\mathrm{supp}\,(\boldsymbol{x}) \cup V$ is maintained. Then, answering whether $\boldsymbol{x} + \boldsymbol{e}_k \in \mathcal{I}$ is equivalent to checking whether an augmenting path on $\mathrm{supp}\,(\boldsymbol{x} + \boldsymbol{e}_k) \cup V$ from M can be found. Finding a augmentation path can be done by a breadth-first search (BFS) starting from k (see Section 17.2 in (Schrijver, 2003)), and it takes $\mathcal{O}(DK)$ time because there are at most K leaves in the BFS tree and the length of the path from k to each leaf is at most $2D$. Thus, we have $\mathcal{T}_{\mathrm{member}} = \mathcal{O}(DK)$.
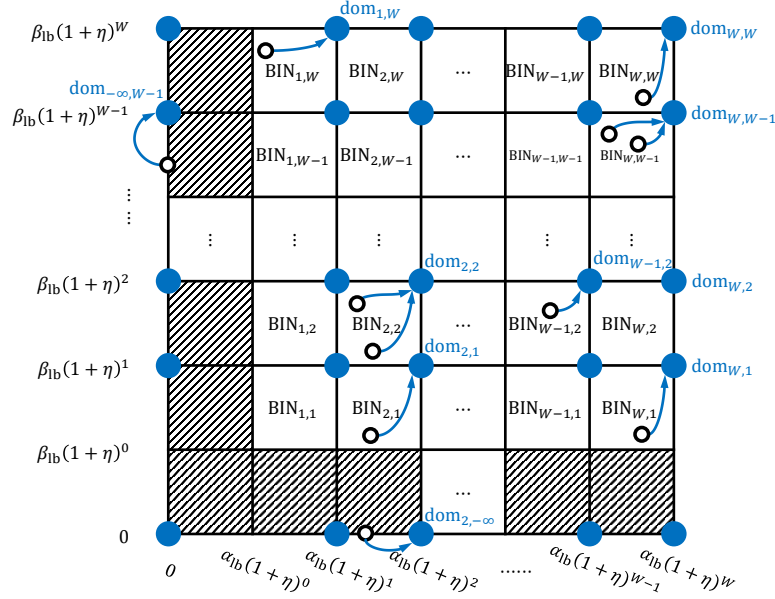
Figure 1. Illustration of feature rounding. There are $|\mathbb{W}|^2$ bins, and features are assumed not to be in (the interior of) the shaded area. Each feature $\boldsymbol{f}_k$ is rounded to its dominating point $\mathrm{dom}(\boldsymbol{f}_k)$, which is specified by a curved arrow.

D. Omitted Proofs in Section 4

Proof of Lemma 4.6. By Eq. (8) and the optimality of $\boldsymbol{x}^*_{\mathrm{dom}}$, for any base $\boldsymbol{x}$, we have

$$
\sum_{k\in\,\mathrm{supp}(\boldsymbol{x}^*_{\mathrm{dom}})} \langle \boldsymbol{f}_k, \boldsymbol{q}\rangle
$$

$$
> \frac{1}{1+\eta}\cdot \sum_{k\in\,\mathrm{supp}(\boldsymbol{x}^*_{\mathrm{dom}})} \langle \mathrm{dom}(\boldsymbol{f}_k), \boldsymbol{q}\rangle \qquad\qquad \text{(by Eq. (8))}
$$

$$
\geq \frac{1}{(1+\eta)^2}\cdot \sum_{k\in\,\mathrm{supp}(\boldsymbol{x})} \langle \mathrm{dom}(\boldsymbol{f}_k), \boldsymbol{q}\rangle \qquad\qquad \text{(by optimality of } \boldsymbol{x}^*_{\mathrm{dom}})
$$

$$
\geq \frac{1}{(1+\eta)^2}\cdot \sum_{k\in\,\mathrm{supp}(\boldsymbol{x})} \langle \boldsymbol{f}_k, \boldsymbol{q}\rangle \qquad\qquad\qquad \text{(by Eq. (8))}
$$

$$
\geq \frac{1}{1+\epsilon}\cdot \sum_{k\in\,\mathrm{supp}(\boldsymbol{x})} \langle \boldsymbol{f}_k, \boldsymbol{q}\rangle. \qquad\qquad \text{(as } (1+\eta)^2 \leq 1+\epsilon) \quad \square
$$

Proof of Lemma 4.7. The proof follows from the uniqueness of the maximum-weight base in the case of distinct weights; see, e.g., (Edmonds, 1971). $\qquad\square$

Proof of Lemma 4.8. For a query $\boldsymbol{q}\in\mathbb{R}^2$, let $\mathcal{C}\subset\mathbb{R}^2$ be a cell in arrangement of $\mathcal{L}$ whose closure contains $\boldsymbol{q}$ (which may not be uniquely determined). Then, there is a permutation $\pi\in\mathfrak{S}_K$ such that for any vector $\boldsymbol{h}\in\mathcal{H}\cap\mathcal{C}$, we have $\langle \boldsymbol{f}_{\pi(i)}, \boldsymbol{h}\rangle > \langle \boldsymbol{f}_{\pi(j)}, \boldsymbol{h}\rangle$ whenever $i<j$. Since $\boldsymbol{q}$ is in the closure of $\mathcal{C}$, it holds that $\langle \boldsymbol{f}_{\pi(i)}, \boldsymbol{q}\rangle \geq \langle \boldsymbol{f}_{\pi(j)}, \boldsymbol{q}\rangle$ for any $i<j$, implying the proof. $\qquad\square$

Proof of Lemma 4.9. Since each cell in arrangement of $\mathcal{L}$ is a polyhedral cone generated by two lines of $\mathcal{L}$ that does not contain any other lines of $\mathcal{L}$, there are only $\mathcal{O}(K^2)$ cells, and each of their internal points can be found by using Algorithm 6, as desired.
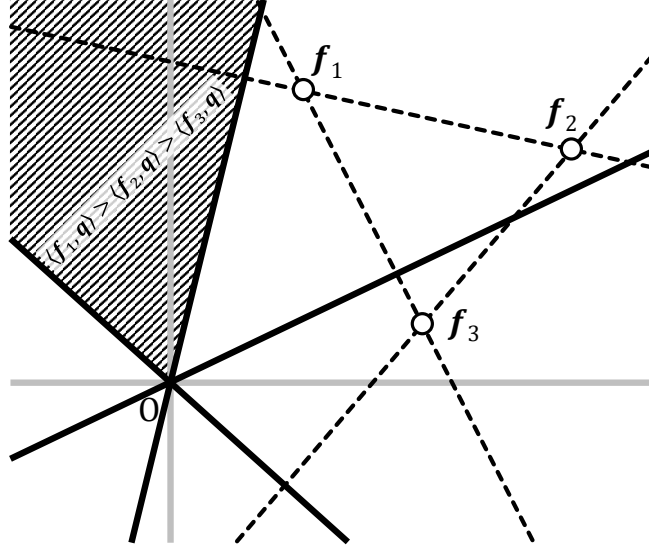
$\qquad\square$

Figure 2. Illustration of characterization of representable permutations. There are three features $\boldsymbol{f}_1, \boldsymbol{f}_2, \boldsymbol{f}_3$ on $\mathbb{R}^2$. Each dashed line denotes $\overleftrightarrow{\boldsymbol{f}_i \boldsymbol{f}_j}$ for some $i \neq j$; each black bold line is orthogonal to some dashed line and intersects the origin. Such black bold lines generate six regions, each corresponding to a distinct permutation. For example, for any query $\boldsymbol{q}$ in the hatched area, it holds that $\langle \boldsymbol{f}_1, \boldsymbol{q} \rangle > \langle \boldsymbol{f}_2, \boldsymbol{q} \rangle > \langle \boldsymbol{f}_3, \boldsymbol{q} \rangle$; i.e., $\boldsymbol{q}$ represents a permutation π such that $(\pi(1), \pi(2), \pi(3)) = (1, 2, 3)$.

Algorithm 6 GENERATE-HITTING-SET.

Input: K distinct features $(\boldsymbol{f}_k)_{k \in [K]}$.
let $\Theta \leftarrow \emptyset$;
for *all* $k \neq k'$ **do**

    let L be a unique line that is orthogonal to line $\overleftrightarrow{\boldsymbol{f}_k \boldsymbol{f}_{k'}}$ and intersects $\boldsymbol{0}$;
    add the angle θ of L and $-\theta$ to Θ;

end
let $\mathcal{H} \leftarrow \emptyset$;
for *all neighboring (but distinct)* θ_1 *and* θ_2 *in* Θ **do**

    let $\boldsymbol{h} \triangleq \left(\cos(\frac{\theta_1 + \theta_2}{2}), \sin(\frac{\theta_1 + \theta_2}{2})\right)$ be an internal point of a polyhedral cone generated by two half-lines whose angles are θ_1 and θ_2;
    add $\boldsymbol{h}$ to $\mathcal{H}$;

end
return $\mathcal{H}$;

Proof of Theorem 4.4. The correctness of **FIND-BASE** is shown first. Given a query $\boldsymbol{q} \in \mathbb{R}_+^2$, Algorithm 3 finds $\boldsymbol{h} \in \mathcal{H}$ such that $\langle \boldsymbol{f}_k, \boldsymbol{h} \rangle > \langle \boldsymbol{f}_{k'}, \boldsymbol{h} \rangle$ implies $\langle \boldsymbol{f}_k, \boldsymbol{q} \rangle \geq \langle \boldsymbol{f}_{k'}, \boldsymbol{q} \rangle$ due to Lemma 4.8. Calling $\mathcal{A}_{\boldsymbol{h}}$ finds a $(1 + \eta)$-approximate maximum-weight base $\boldsymbol{x}^\circ$ of $\mathcal{M}$ with arm k's weight defined as $\langle \text{dom}(\boldsymbol{f}_k), \boldsymbol{h} \rangle$. Since a total order over $[K]$ induced by arm weights $\langle \text{dom}(\boldsymbol{f}_k), \boldsymbol{h} \rangle$ is consistent with that induced by arm weights $\langle \text{dom}(\boldsymbol{f}_k), \boldsymbol{q} \rangle$, by Lemma 4.7, $\boldsymbol{x}^\circ$ is also a $(1 + \eta)$-approximate maximum-weight base of $\mathcal{M}$ with arm k's weight defined as $\langle \text{dom}(\boldsymbol{f}_k), \boldsymbol{q} \rangle$. By Lemma 4.6,

$$\sum_{k \in \text{supp}(\boldsymbol{x}^\circ)} \langle \boldsymbol{f}_k, \boldsymbol{q} \rangle \geq \frac{1}{1 + \epsilon} \cdot \sum_{k \in \text{supp}(\boldsymbol{x})} \langle \boldsymbol{f}_k, \boldsymbol{q} \rangle, \tag{17}$$

for any base $\boldsymbol{x}$ of $\mathcal{M}$; namely, $\boldsymbol{x}^\circ$ is a $(1 + \epsilon)$-approximate maximum-weight base of $\mathcal{M}$ with arm k's weight defined as $\langle \boldsymbol{f}_k, \boldsymbol{q} \rangle$, completing the correctness of **FIND-BASE**.

Subsequently, we bound the time complexity of each subroutine as follows.

INITIALIZE: Construction of $\text{BIN}_{q,r}$ and $\text{dom}_{q,r}$ for all $q, r \in \mathbb{W}$ completes in $\mathcal{O}(K + W^2)$ time. Then, a hitting set $\mathcal{H}$ for $\text{dom}_{q,r}$'s and $\frac{1}{1+\eta} \cdot \text{dom}_{q,r}$'s of size $|\mathcal{H}| = \mathcal{O}(W^4)$ can be constructed in poly (W) time due to Lemma 4.9. There will be $|\mathcal{H}|$ instances of algorithm $\mathcal{A}$ (with different arm weights), creating which takes $\mathcal{O}(W^4 \cdot \mathcal{T}_{\text{init}}(\mathcal{A}; \eta))$ time.

FIND-BASE: Checking whether each $\boldsymbol{h} \in \mathcal{H}$ and query $\boldsymbol{q} \in \mathbb{R}_+^2$ belong to (the closure of) the same cell in arrangement of

16

$\mathcal{V}$ can be done in $\mathcal{O}(W^2)$ time by comparing the induced total orders. By brute-force search, a desired $\boldsymbol{h}$ can be found in $\mathcal{O}(W^6)$ time. Since calling $\mathcal{A}_{\boldsymbol{h}}$ requires $\mathcal{O}(D)$ time, the entire time complexity is bounded by $\mathcal{O}(\operatorname{poly}(W) + D)$.

UPDATE-FEATURE: For $|\mathcal{H}|$ instances of $\mathcal{A}$, a single arm's weight would be changed, each of which runs in $\mathcal{T}_{\text{update}}(\mathcal{A}; \eta)$ time.

Observe finally that

$$
\begin{aligned}
W &= \max \left\{ \left\lceil \log_{1+\eta} \left(\frac{\alpha_{\text{ub}}}{\alpha_{\text{lb}}} \right) \right\rceil, \left\lceil \log_{1+\eta} \left(\frac{\beta_{\text{ub}}}{\beta_{\text{lb}}} \right) \right\rceil \right\} \\
&= \mathcal{O} \left(\frac{\log(\frac{\alpha_{\text{ub}}}{\alpha_{\text{lb}}}) + \log(\frac{\beta_{\text{ub}}}{\beta_{\text{lb}}})}{\log(1 + \eta)} \right) \\
&= \mathcal{O} \left(\eta^{-1} \cdot \log \left(\frac{\alpha_{\text{ub}}}{\alpha_{\text{lb}}} \cdot \frac{\beta_{\text{ub}}}{\beta_{\text{lb}}} \right) \right) \\
&= \mathcal{O} \left(\epsilon^{-1} \cdot \log \left(\frac{\alpha_{\text{ub}}}{\alpha_{\text{lb}}} \cdot \frac{\beta_{\text{ub}}}{\beta_{\text{lb}}} \right) \right),
\end{aligned}
\tag{18}
$$

where we used the fact that $\frac{1}{\log(1+\eta)} < \frac{1}{\eta}$ when $\eta \in (0, 1)$, completing the proof. $\square$

E. Proofs Related to Regret Analysis

E.1. Proofs Related to the Bijection g_t

Lemma 5.2. There exists a bijection $g_t : \text{supp}\,(\boldsymbol{i}^\star) \to \text{supp}\,(\boldsymbol{x}(t))$ such that (i) $g_t(j) = j$ for $j \in \text{supp}\,(\boldsymbol{i}^\star) \cap \text{supp}\,(\boldsymbol{x}(t))$; (ii) for any $j \in \text{supp}\,(\boldsymbol{i}^\star) \setminus \text{supp}\,(\boldsymbol{x}(t))$,

$$x_{g_t(j)}(t) = 1 \implies \left\langle \text{dom}(\boldsymbol{f}_{g_t(j)}), \boldsymbol{h} \right\rangle \geq \frac{\langle \text{dom}(\boldsymbol{f}_j), \boldsymbol{h} \rangle}{1 + \frac{1}{3\log^m T}}.$$

Proof: Let $\eta \triangleq \frac{1}{3\log^m T}$. The proof is inspired by Section 4.2 in (Kveton et al., 2014a), and several changes are made to deal with the usage of the dynamic $(1 + \eta)$-approximate maximum-weight basis algorithm in the **FIND-BASE** procedure.

Let $\xi_t : [D] \to \text{supp}\,(\boldsymbol{x}(t))$ be the ordering such that $\xi_t(i)$'s arm weight $\left\langle \text{dom}(\boldsymbol{f}_{\xi_t(i)}), \boldsymbol{h} \right\rangle$ is the i-th largest, where $\boldsymbol{h} \in \mathcal{H}$ lies in the same cell as the query $\boldsymbol{q} = (1, \lambda_t)$ when invoking **FIND-BASE** procedure.

Explicit construction of g_t: We define

$$g_t(j) = \xi_t(\pi_t^{-1}(j)), \forall j \in \text{supp}\,(\boldsymbol{i}^\star),$$

where the function $\pi_t : [D] \to \text{supp}\,(\boldsymbol{i}^\star)$ is a bijection such that the following hold:

(i) $\sum_{i=1}^{k-1} \boldsymbol{e}_{\xi_t(i)} + \boldsymbol{e}_{\pi_t(k)} \in \mathcal{I}$ for all $k \in [D]$

(ii) $\pi_t(k) = \xi_t(k)$ if $\xi_t(k) \in \text{supp}\,(\boldsymbol{i}^\star) \cap \text{supp}\,(\boldsymbol{x}(t))$

The existence of π_t is proved in Lemma E.1 and also by Lemma 1 of (Kveton et al., 2014a).

Show (i) $g_t(j) = j$ for $j \in \text{supp}\,(\boldsymbol{i}^\star) \cap \text{supp}\,(\boldsymbol{x}(t))$: Fix any $j \in \text{supp}\,(\boldsymbol{i}^\star) \cap \text{supp}\,(\boldsymbol{x}(t))$. From the definition of π_t, we have $\pi_t(j) = \xi_t(j)$ and hence $g_t(j) = \xi_t(\pi_t^{-1}(j)) = \xi_t(\xi_t^{-1}(j)) = j$.

Show (ii) $x_{g_t(j)}(t) = 1 \implies \left\langle \text{dom}(\boldsymbol{f}_{g_t(j)}), \boldsymbol{h} \right\rangle \geq \frac{\langle \text{dom}(\boldsymbol{f}_j), \boldsymbol{h} \rangle}{1+\eta}$: Fix any $j \in \text{supp}\,(\boldsymbol{i}^\star) \setminus \text{supp}\,(\boldsymbol{x})$. Let $k = \pi_t^{-1}(j)$. Observe that the bijection π_t captures the situation that: The algorithm can choose $\pi_t(k) \in \text{supp}\,(\boldsymbol{i}^\star)$ as the k-th element but instead chooses $\xi_t(k) \in \text{supp}\,(\boldsymbol{x}(t))$. By the procedure of Algorithm 3 and Assumption 4.2, this happens when

$$\left\langle \text{dom}(\boldsymbol{f}_{\xi_t(k)}), \boldsymbol{h} \right\rangle \geq \frac{1}{1+\eta} \left\langle \text{dom}(\boldsymbol{f}_{\pi_t(k)}), \boldsymbol{h} \right\rangle,$$

and replacing $k = \pi_t^{-1}(j)$ completes the proof. $\qquad\square$

Lemma E.1. *Let $\boldsymbol{x}, \boldsymbol{i}^\star \in \mathcal{X}$, and $\xi : [D] \to \text{supp}\,(\boldsymbol{x})$ be an arbitrary bijection. There exists a bijection $\pi : [D] \to \text{supp}\,(\boldsymbol{i}^\star)$ such that $\sum_{i=1}^{k-1} \boldsymbol{e}_{\xi(i)} + \boldsymbol{e}_{\pi(k)} \in \mathcal{I}$ for all $k \in [D]$.*

Proof: This lemma is equivalent to Lemma 1 of (Kveton et al., 2014a). For reader's convenience, we provide a proof here.

For $k = D$, consider $\sum_{i=1}^{D-1} \boldsymbol{e}_{\xi(i)} \in \mathcal{I}$ (due to hereditary property), and $\boldsymbol{i}^\star \in \mathcal{I}$. As the former has $D - 1$ element while the latter has D elements, by augmentation property, there exists $\pi(D) \in \text{supp}\,(\boldsymbol{i}^\star)$ such that $\sum_{i=1}^{D-1} \boldsymbol{e}_{\xi(i)} + \boldsymbol{e}_{\pi(D)} \in \mathcal{I}$. For the case when $\xi(D) \in \text{supp}\,(\boldsymbol{i}^\star) \cap \text{supp}\,(\boldsymbol{x})$, we set $\pi(D) = \xi(D)$.

The proof is completed by repeating the following process for $k = D - 1, \cdots, 1$. As $\sum_{i=1}^{k-1} \boldsymbol{e}_{\xi(i)} \in \mathcal{I}$ (due to hereditary property) has $k - 1$ elements, and $\boldsymbol{i}^\star - \sum_{i=k+1}^{D} \boldsymbol{e}_{\pi(i)} \in \mathcal{I}$ (due to hereditary property) has k elements, by augmentation property, there exists $\pi(k)$ such that $\sum_{i=1}^{k-1} \boldsymbol{e}_{\xi(i)} + \boldsymbol{e}_{\pi(k)} \in \mathcal{I}$. If $\xi(k) \in \text{supp}\,(\boldsymbol{i}^\star) \cap \text{supp}\,(\boldsymbol{x})$, we set $\pi(k) = \xi(k)$. $\qquad\square$

E.2. Lemmas Related to Regret Analysis of Algorithm 5

In this section, we fix a best action $\boldsymbol{i}^\star \in \text{argmax}_{\boldsymbol{x} \in \mathcal{X}} \langle \boldsymbol{\mu}, \boldsymbol{x} \rangle$ and define $\triangle_{j,k} \triangleq \mu_j - \mu_k$. Let $\{\bar{j}\}_{j=1}^{D}$ be the permutation of $\text{supp}\,(\boldsymbol{i}^\star)$ such that $\mu_{\bar{1}} \geq \cdots \geq \mu_{\bar{D}}$. Define $d_k \triangleq \max\{j \in \text{supp}\,(\boldsymbol{i}^\star) : \triangle_{\bar{j},k} > 0\}$ and $\triangle_{\min} \triangleq \min_{k \notin \text{supp}(\boldsymbol{i}^\star)} \triangle_{\overline{d_k},k}$.

Lemma 5.3. *Let* $\epsilon < \frac{\triangle_{\min}}{b}$. *Then, for any* $i \in supp\,(\boldsymbol{i}^\star)$ *and any* $j \notin supp\,(\boldsymbol{i}^\star)$,

$$\mu_i - \mu_j > 0 \implies \frac{\mu_i}{1+\epsilon} - \mu_j > 0.$$

Proof: Fix $i \in supp\,(\boldsymbol{i}^\star)$ and $j \notin supp\,(\boldsymbol{i}^\star)$ such that $\mu_i - \mu_j > 0$. We want the following to hold:

$$\frac{\mu_i}{1+\epsilon} - \mu_j > 0 \iff \mu_i - (1+\epsilon)\mu_j > 0 \iff \mu_i - \mu_j > \epsilon\mu_j.$$

As $\mu_i - \mu_j > \epsilon\mu_j$ must hold for all such i and j, taking the minimum over all possible i and j on the left-hand side, and use the fact that $\mu_j \leq b$ for all j on the right-hand side, we derive

$$\frac{\triangle_{\min}}{b} > \epsilon$$

is the condition on ϵ to ensure $\mu_i - \mu_j > 0 \implies \frac{\mu_i}{1+\epsilon} - \mu_j > 0$ holds for all i and j. $\qquad\square$

Lemma 5.4. *Let* $k \notin supp\,(\boldsymbol{i}^\star)$ *and* $j \in [d_k]$. *For* $T > T_0$,

$$\sum_{j=1}^{d_k} \triangle_{\overline{j},k}(I)_{\overline{j},k} \leq \sum_{j=1}^{d_k} \triangle_{\overline{j},k}T_0 + \frac{12(b-a)^2\triangle_{\overline{d_k},k}\log T}{(\frac{\mu_{\overline{d_k}}}{1+\log^{-m} T} - \mu_k)^2}.$$

Proof: Recall $(I)_{\overline{j},k} = \sum_{t=1}^{T} \mathbb{E}\left[\mathbb{1}\left\{g_t(\overline{j}) = k, N_k(t) \leq n_{\overline{j},k}\right\}\right]$, where $n_{j,k} = \max\left\{\frac{6(b-a)^2\log T}{(\frac{\mu_j}{1+\log^{-m} T}-\mu_k)^2}, T_0\right\}$.

First, we claim that: for any $\{a_j\}_{j=1}^{d_k}$ with $a_1 \geq \cdots \geq a_{d_k} \geq 0$,

$$\sum_{j=1}^{d_k} a_j(I)_{\overline{j},k} \leq a_1 n_{\overline{1},k} + \sum_{j=2}^{d_k} a_j(n_{\overline{j},k} - n_{\overline{j-1},k}). \tag{19}$$

Show Eq. (19): We show by induction. For the base case, we have

$$a_1(I)_{\overline{1},k} + a_2(I)_{\overline{2},k} \leq a_1 n_{\overline{1},k} + a_2(n_{\overline{2},k} - n_{\overline{1},k}). \tag{20}$$

Eq. (20) is derived as follows. Since $a_1, a_2 \geq 0$ and

$$(I)_{\overline{1},k} + (I)_{\overline{2},k} = \sum_{t=1}^{T} \mathbb{E}\left[\mathbb{1}\left\{g_t(\overline{1}) = k, N_k(t) \leq n_{\overline{1},k}\right\} + \mathbb{1}\left\{g_t(\overline{2}) = k, N_k(t) \leq n_{\overline{2},k}\right\}\right] \leq \max\{n_{\overline{1},k}, n_{\overline{2},k}\} = n_{\overline{2},k},$$

therefore we can bound $(I)_{\overline{2},k}$ as $(I)_{\overline{2},k} \leq n_{\overline{2},k} - (I)_{\overline{1},k}$, yields that:

$$a_1(I)_{\overline{1},k} + a_2(I)_{\overline{2},k} \leq (a_1 - a_2)(I)_{\overline{1},k} + \triangle_{\overline{2},k}n_{\overline{2},k}.$$

Then, since $a_1 \geq a_2$ and $(I)_{\overline{1},k} \leq n_{\overline{1},k}$, we derive

$$a_1(I)_{\overline{1},k} + a_2(I)_{\overline{2},k} \leq (a_1 - a_2)n_{\overline{1},k} + a_2 n_{\overline{2},k},$$

which shows Eq. (20). Now, assume for any $\{b_j\}_{j=1}^{\ell}$ with $b_1 \geq \cdots \geq b_\ell \geq 0$, the following

$$\sum_{j=1}^{\ell} b_j(I)_{\overline{j},k} \leq b_1 n_{\overline{1},k} + \sum_{j=2}^{\ell} b_j(n_{\overline{j},k} - n_{\overline{j-1},k}) \tag{21}$$

holds for $\ell < d_k$.

Fix any $\{a_j\}_{j=1}^{\ell+1}$ with $a_1 \geq \cdots \geq a_{\ell+1} \geq 0$. Consider $\sum_{j=1}^{\ell+1} a_j(I)_{\bar{j},k}$. Since $a_j \geq 0$ for all $j \in [\ell+1]$ and

$$\sum_{j=1}^{\ell+1}(I)_{\bar{j},k} = \sum_{t=1}^{T} \mathbb{E}\left[\sum_{j=1}^{\ell+1} \mathbb{1}\{g_t(\bar{j}) = k, N_k(t) \leq n_{\bar{1},k}\}\right] \leq \max_{j\in[\ell+1]} n_{\bar{j},k} = n_{\overline{\ell+1},k},$$

we can bound $(I)_{\overline{\ell+1},k}$ as $(I)_{\overline{\ell+1},k} \leq n_{\overline{\ell+1},k} - \sum_{j=1}^{\ell}(I)_{\bar{j},k}$, which results in:

$$\sum_{j=1}^{\ell+1} a_j(I)_{\bar{j},k} \leq \sum_{j=1}^{\ell}(a_j - a_{\ell+1})(I)_{\bar{j},k} + a_{\ell+1} n_{\overline{\ell+1},k}.$$

Since $a_1 - a_{\ell+1} \geq \cdots \geq a_\ell - a_{\ell+1} \geq 0$, using inductive hypothesis Eq. (21) with $b_j = a_j - a_{\ell+1}$ for all $j \in [\ell]$, we get

$$\sum_{j=1}^{\ell+1} a_j(I)_{\bar{j},k} \leq (a_1 - a_{\ell+1})n_{\bar{1},k} + \sum_{j=2}^{\ell}(a_j - a_{\ell+1})(n_{\bar{j},k} - n_{\overline{j-1},k}) + a_{\ell+1} n_{\overline{\ell+1},k}.$$

$$= a_1 n_{\bar{1},k} + \sum_{j=2}^{\ell} a_j(n_{\bar{j},k} - n_{\overline{j-1},k}) - a_{\ell+1}\left(n_{\bar{1},k} + \sum_{j=2}^{\ell}(n_{\bar{j},k} - n_{\overline{j-1},k}) - n_{\overline{\ell+1},k}\right)$$

$$= a_1 n_{\bar{1},k} + \sum_{j=2}^{\ell} a_j(n_{\bar{j},k} - n_{\overline{j-1},k}) + a_{\ell+1}(n_{\overline{\ell+1},k} - n_{\bar{\ell},k})$$

$$= a_1 n_{\bar{1},k} + \sum_{j=2}^{\ell+1} a_j(n_{\bar{j},k} - n_{\overline{j-1},k}).$$

Thus, Eq. (19) is proved by induction.

Define $\epsilon \triangleq \log^{-m} T$ and $\triangle_{j,k}(\epsilon) \triangleq \frac{\mu_j}{1+\epsilon} - \mu_k$. Using Eq. (19) with $a_j = \triangle_{\bar{j},k}$ for $j \in [d_k]$ and recalling $n_{j,k} \triangleq \max\left\{\frac{6(b-a)^2 \log T}{\triangle_{j,k}(\epsilon)^2}, T_0\right\}$, we have

$$\sum_{j=1}^{d_k} \triangle_{\bar{j},k}(I)_{\bar{j},k} \leq \triangle_{\bar{1},k} n_{\bar{1},k} + \sum_{j=2}^{d_k} \triangle_{\bar{j},k}(n_{\bar{j},k} - n_{\overline{j-1},k})$$

$$\leq \sum_{j=1}^{d_k} \triangle_{\bar{j},k} T_0 + 6(b-a)^2 \log T \left(\frac{\triangle_{\bar{1},k}}{\triangle_{\bar{1},k}(\epsilon)^2} + \sum_{j=2}^{d_k} \triangle_{\bar{j},k}\left(\frac{1}{\triangle_{\bar{j},k}(\epsilon)^2} - \frac{1}{\triangle_{\overline{j-1},k}(\epsilon)^2}\right)\right).$$

We upper bound the last term by:

$$\frac{\triangle_{\bar{1},k}}{\triangle_{\bar{1},k}(\epsilon)^2} + \sum_{j=2}^{d_k} \triangle_{\bar{j},k}\left(\frac{1}{\triangle_{\bar{j},k}(\epsilon)^2} - \frac{1}{\triangle_{\overline{j-1},k}(\epsilon)^2}\right) = \sum_{j=1}^{d_k-1} \frac{\triangle_{\bar{j},k}(\epsilon) - \triangle_{\overline{j+1},k}(\epsilon)}{(\triangle_{\bar{j},k}(\epsilon))^2} + \frac{\triangle_{\overline{d_k},k}}{\triangle_{\overline{d_k},k}(\epsilon)^2}$$

$$\leq \sum_{j=1}^{d_k-1} \frac{\triangle_{\bar{j},k}(\epsilon) - \triangle_{\overline{j+1},k}(\epsilon)}{\triangle_{\bar{j},k}(\epsilon)\triangle_{\overline{j+1},k}(\epsilon)} + \frac{\triangle_{\overline{d_k},k}}{\triangle_{\overline{d_k},k}(\epsilon)^2}$$

$$= \sum_{j=1}^{d_k-1}\left(\frac{1}{\triangle_{\overline{j+1},k}(\epsilon)} - \frac{1}{\triangle_{\bar{j},k}(\epsilon)}\right) + \frac{\triangle_{\overline{d_k},k}}{\triangle_{\overline{d_k},k}(\epsilon)^2} \leq \frac{2\triangle_{\overline{d_k},k}}{\triangle_{\overline{d_k},k}(\epsilon)^2},$$

where the first inequality is due to $\triangle_{\bar{j},k}(\epsilon) \geq \triangle_{\overline{j+1},k}(\epsilon)$, and the second upperbounds the telescoping series:

$$\sum_{j=1}^{d_k-1}\left(\frac{1}{\triangle_{\overline{j+1},k}(\epsilon)} - \frac{1}{\triangle_{\bar{j},k}(\epsilon)}\right) = \frac{1}{\triangle_{\overline{d_k},k}(\epsilon)} - \frac{1}{\triangle_{\bar{1},k}(\epsilon)} \leq \frac{1}{\triangle_{\overline{d_k},k}(\epsilon)}$$

Hence, we derive an upper bound for the first part relevant to $(I)_{\bar{j},k}$:

$$\sum_{j=1}^{d_k} \triangle_{\bar{j},k}(I)_{\bar{j},k} \leq \sum_{j=1}^{d_k} \triangle_{\bar{j},k} T_0 + \frac{12(b-a)^2 \triangle_{\overline{d_k},k} \log T}{\triangle_{\overline{d_k},k}(\epsilon)^2}.$$

$\square$

Lemma 5.5. Let $k \notin supp\,(i^\star)$ and $j \in [d_k]$. For $T > T_0$,

$$(II)_{\bar{j},k} \leq \frac{1}{T} + \frac{\pi^2}{6}.$$

Proof of Lemma 5.5: Let $\epsilon = \frac{1}{\log^m T}$. Recall

$$(II)_{\bar{j},k} = \sum_{t=1}^{T} \mathbb{E}\left[\mathbb{1}\left\{g_t(\bar{j}) = k, N_k(t) > n_{\bar{j},k}\right\}\right],$$

where $n_{j,k} = \max\left\{\frac{6(b-a)^2 \log T}{(\frac{\mu_j}{1+\log^{-m} T} - \mu_k)^2}, T_0\right\}$.

First, we claim that

$$g_t(\bar{j}) = k \implies u_k(N_k(t-1), T) \geq \frac{\min_{s<t} u_{\bar{j}}(s,t)}{1+\epsilon}, \tag{14}$$

where $u_k(s,t) = \tilde{\mu}_k(s) + \sqrt{\frac{1.5(b-a)^2 \log t}{s}}$ and $\tilde{\mu}_k(t) = \frac{1}{t}\sum_{s=1}^{t} y_k(s)$.

Show Eq. (14): Observe that $g_t(\bar{j}) = k$ implies

$$\left(1 + \frac{\epsilon}{3}\right)\langle \boldsymbol{f}_k, \boldsymbol{q}\rangle \geq \langle \text{dom}(\boldsymbol{f}_k), \boldsymbol{q}\rangle \geq \frac{\left\langle \text{dom}(\boldsymbol{f}_{\bar{j}}), \boldsymbol{q}\right\rangle}{1+\frac{\epsilon}{3}} \geq \frac{\left\langle \boldsymbol{f}_{\bar{j}}, \boldsymbol{q}\right\rangle}{1+\frac{\epsilon}{3}},$$

where Eq. (8) is used in the first and last inequality, and the second inequality is due to Lemma 5.2 and Corollary 4.10. By $(1 + \frac{\epsilon}{3})^2 \leq 1 + \epsilon$ and expanding $\boldsymbol{f}_k = (\hat{\mu}_k(t-1), \frac{1}{\sqrt{N_k(t-1)}})$ and $\boldsymbol{q} = (1, \sqrt{1.5(b-a)^2 \log t})$, we have

$$u_k(N_k(t-1), t) \geq \frac{u_{\bar{j}}(N_{\bar{j}}(t-1), t)}{1+\epsilon}.$$

As $\log T > \log t$ and $N_{\bar{j}}(t-1) \in [t-1]$, we further derive

$$u_k(N_k(t-1), T) \geq \frac{u_{\bar{j}}(N_{\bar{j}}(t-1), t)}{1+\epsilon} \geq \frac{\min_{s<t} u_{\bar{j}}(s,t)}{1+\epsilon},$$

which shows Eq. (14).

Second, let $\mathcal{T}_{\bar{j},k} = \{t \in \{n_{\bar{j},k} + 1, \cdots, T\} : g_t(\bar{j}) = k, N_k(t-1) > n_{\bar{j},k}\}$. From Eq. (14), we derive

$$\begin{aligned}
(II)_{\bar{j},k} &= \sum_{t=n_{\bar{j},k}+1}^{T} \mathbb{P}\left[g_t(\bar{j}) = k, N_k(t-1) > n_{\bar{j},k}\right] \\
&\leq \sum_{t=n_{\bar{j},k}+1}^{T} \mathbb{P}\left[u_k(N_k(t-1), T) \geq \frac{\min_{s<t} u_{\bar{j}}(s,t)}{1+\epsilon} \text{ and } t \in \mathcal{T}_{\bar{j},k}\right] \\
&\leq \sum_{t=n_{\bar{j},k}+1}^{T} \sum_{s<t} \mathbb{P}\left[u_k(N_k(t-1), T) \geq \frac{u_{\bar{j}}(s,t)}{1+\epsilon} \text{ and } t \in \mathcal{T}_{\bar{j},k}\right], \tag{22}
\end{aligned}$$

where the last inequality uses union bound.

Third, we now upper bound each term $\mathbb{P}\left[u_k(N_k(t-1),T) \geq \frac{u_{\bar{j}}(s,t)}{1+\epsilon} \text{ and } t \in \mathcal{T}_{\bar{j},k}\right]$ in Eq. (22). Remind that

$$u_k(N_k(t-1),T) \geq \frac{u_{\bar{j}}(s,t)}{1+\epsilon} \iff \underbrace{\tilde{\mu}_k(N_k(t-1)) + \frac{\lambda_T}{\sqrt{N_k(t-1)}}}_{A_t} \geq \underbrace{\frac{\tilde{\mu}_{\bar{j}}(s) + \frac{\lambda_t}{\sqrt{s}}}{1+\epsilon}}_{B_{t,s}}.$$

Define the event $\mathcal{E}_{t,s} = \{A_t \geq B_{t,s} \text{ and } t \in \mathcal{T}_{\bar{j},k}\}$. We will partition the event $\mathcal{E}_{t,s}$ by comparing A_t to $A'_t = \mu_k + \frac{2\lambda_T}{\sqrt{N_k(t-1)}}$ and comparing $B_{t,s}$ to $B' = \frac{\mu_{\bar{j}}}{1+\epsilon}$ as follows:

- $\mathcal{E}_{t,s} \cap \left\{A_t \geq A'_t \text{ and } t \in \mathcal{T}_{\bar{j},k}\right\} \subseteq \left\{\tilde{\mu}_k(N_k(t-1)) \geq \mu_k + \frac{\lambda_T}{\sqrt{N_k(t-1)}} \text{ and } t \in \mathcal{T}_{\bar{j},k}\right\}$

- $\mathcal{E}_{t,s} \cap \{B_{t,s} \leq B' \text{ and } t \in \mathcal{T}_{\bar{j},k}\} \subseteq \left\{\mu_{\bar{j}} \geq \tilde{\mu}_{\bar{j}}(s) + \frac{\lambda_t}{\sqrt{s}} \text{ and } t \in \mathcal{T}_{\bar{j},k}\right\}$

- $\mathcal{E}_{t,s} \cap \{A_t < A'_t \text{ and } B_{t,s} > B' \text{ and } t \in \mathcal{T}_{\bar{j},k}\} \subseteq \left\{\mu_k + \frac{2\lambda_T}{\sqrt{N_k(t-1)}} > \frac{\mu_{\bar{j}}}{1+\epsilon} \text{ and } t \in \mathcal{T}_{\bar{j},k}\right\}$. The inclusion is because under the event $\mathcal{E}_{t,s} \cap \{A_t < A'_t \text{ and } B_{t,s} > B' \text{ and } t \in \mathcal{T}_{\bar{j},k}\}$, we have

$$\mu_k + \frac{2\lambda_T}{\sqrt{N_k(t-1)}} = A'_t > A_t \geq B_{t,s} > B' = \frac{\mu_{\bar{j}}}{1+\epsilon},$$

where the first and last inequalities are due to the event $\{A_t < A'_t \text{ and } B_{t,s} > B' \text{ and } t \in \mathcal{T}_{\bar{j},k}\}$, and the second inequality is due to the event $\mathcal{E}_{t,s} = \{A_t \geq B_{t,s} \text{ and } t \in \mathcal{T}_{\bar{j},k}\}$.

Hence, we have the following inclusion:

$$\{A_t \geq A'_t \text{ and } t \in \mathcal{T}_{\bar{j},k}\} \cup \{B_{t,s} \leq B' \text{ and } t \in \mathcal{T}_{\bar{j},k}\} \cup \{A_t < A'_t \text{ and } B_{t,s} > B' \text{ and } t \in \mathcal{T}_{\bar{j},k}\}$$
$$= \{t \in \mathcal{T}_{\bar{j},k}\} \supset \{A_t \geq B_{t,s} \text{ and } t \in \mathcal{T}_{\bar{j},k}\} = \mathcal{E}_{t,s}.$$

From union bound,

$$\mathbb{P}[\mathcal{E}_{t,s}] \leq \mathbb{P}\left[\{A_t \geq A'_t \text{ and } t \in \mathcal{T}_{\bar{j},k}\} \cap \mathcal{E}_{t,s}\right] + \mathbb{P}\left[\{B_{t,s} \leq B' \text{ and } t \in \mathcal{T}_{\bar{j},k}\} \cap \mathcal{E}_{t,s}\right]$$
$$+ \mathbb{P}\left[\{A_t < A'_t \text{ and } B_{t,s} > B' \text{ and } t \in \mathcal{T}_{\bar{j},k}\} \cap \mathcal{E}_{t,s}\right]$$
$$\leq \mathbb{P}\left[\mu_k + \frac{\lambda_T}{\sqrt{N_k(t-1)}} \leq \tilde{\mu}_k(N_k(t-1)) \text{ and } t \in \mathcal{T}_{\bar{j},k}\right]$$
$$+ \mathbb{P}\left[\mu_{\bar{j}} \geq \tilde{\mu}_{\bar{j}}(s) + \frac{\lambda_t}{\sqrt{s}} \text{ and } t \in \mathcal{T}_{\bar{j},k}\right] + \mathbb{P}\left[\mu_k + \frac{2\lambda_T}{\sqrt{N_k(t-1)}} > \frac{\mu_{\bar{j}}}{1+\epsilon} \text{ and } t \in \mathcal{T}_{\bar{j},k}\right]. \quad (23)$$

In Eq. (23), recall $\lambda_t = \sqrt{1.5(b-a)^2 \log t}$ and observe that the last term

$$\mathbb{P}\left[\mu_k + 2\sqrt{\frac{1.5(b-a)^2 \log T}{N_k(t-1)}} \geq \frac{\mu_{\bar{j}}}{1+\epsilon} \text{ and } t \in \mathcal{T}_{\bar{j},k}\right] \leq \mathbb{P}\left[\mu_k + 2\sqrt{\frac{1.5(b-a)^2 \log T}{n_{\bar{j},k}+1}} \geq \frac{\mu_{\bar{j}}}{1+\epsilon}\right] = 0,$$

where the inequality is because $t \in \mathcal{T}_{\bar{j},k}$ implies $N_k(t-1) \geq n_{\bar{j},k} + 1$, and the equality is because

$$n_{\bar{j},k} \geq \frac{6(b-a)^2 \log T}{(\frac{\mu_{\bar{j}}}{1+\epsilon} - \mu_k)^2} \implies \frac{6(b-a)^2 \log T}{n_{\bar{j},k}+1} < \left(\frac{\mu_{\bar{j}}}{1+\epsilon} - \mu_k\right)^2$$

and also we have $\frac{\mu_{\bar{j}}}{1+\epsilon} - \mu_k > 0$ which is ensured by Lemma 5.3 as $T > T_0$. Finally, from Eq. (22) and Eq. (23),

$$
\begin{aligned}
(II)_{\bar{j},k} &\leq \sum_{t=n_{\bar{j},k}+1}^{T} \sum_{s<t} \mathbb{P}\left[\tilde{\mu}_k(N_k(t-1)) \geq \mu_k + \sqrt{\frac{1.5(b-a)^2 \log T}{N_k(t-1)}} \text{ and } t \in \mathcal{T}_{\bar{j},k}\right] \\
&\quad + \sum_{t=n_{\bar{j},k}+1}^{T} \sum_{s<t} \mathbb{P}\left[\mu_{\bar{j}} \geq \tilde{\mu}_{\bar{j}}(s) + \sqrt{\frac{1.5(b-a)^2 \log t}{s}} \text{ and } t \in \mathcal{T}_{\bar{j},k}\right] \\
&\leq \sum_{t=n_{\bar{j},k}+1}^{T} \sum_{s<t} \left(\mathbb{P}\left[\tilde{\mu}_k(t-1) \geq \mu_k + \sqrt{\frac{1.5(b-a)^2 \log T}{t-1}}\right] + \mathbb{P}\left[\mu_{\bar{j}} \geq \tilde{\mu}_{\bar{j}}(s) + \sqrt{\frac{1.5(b-a)^2 \log t}{s}}\right]\right) \\
&\leq \sum_{t=n_{\bar{j},k}+1}^{T} \sum_{s<t} \left(e^{-3\log T} + e^{-3\log t}\right),
\end{aligned}
$$

where the second inequality is because $\{N_k(t-1)\}_{t \in \mathcal{T}_{\bar{j},k}}$ is strictly increasing (as $N_k(t) = N_k(t-1)+1$ when $g_t(\bar{j}) = k$) and thus is a subsequence of $\{n_{\bar{j},k}+1, \cdots, T\}$, and the last inequality is due to an application of Hoeffding's inequality (Lemma E.2) with $s = \sqrt{1.5(t-1)(b-a)^2 \log T}$ and $n = t-1$ to bound the first term and with $s = \sqrt{1.5s(b-a)^2 \log t}$ and $n = s$ to bound the second term. The proof is completed by evaluating

$$
\sum_{t=1}^{T} \sum_{s<t} e^{-3\log T} \leq \sum_{t=1}^{T} \frac{t}{T^3} \leq \frac{T(T+1)}{2T^3} \leq \frac{1}{T},
$$

$$
\sum_{t=1}^{T} \sum_{s<t} e^{-3\log t} \leq \sum_{t=1}^{\infty} \frac{t}{t^3} \leq \sum_{t=1}^{\infty} \frac{1}{t^2} \leq \frac{\pi^2}{6}.
$$

$\square$

Lemma E.2 (Hoeffding's inequality)**.** *Let* $X_1, \cdots, X_n$ *be independent random variables such that* $X_i \in [a,b]$ *for all* $i \in [n]$. *Then, for all* $s > 0$,

$$
\mathbb{P}\left[\sum_{i=1}^{n}(X_i - \mathbb{E}[X_i]) \geq s\right] \leq \exp\left(-\frac{2s^2}{n(b-a)^2}\right).
$$