Combinatorial Pure Exploration for Dueling Bandits

Wei Chen [*] [1] **Yihan Du** [2] **Longbo Huang** [2] **Haoyu Zhao** [2]

Abstract

In this paper, we study combinatorial pure exploration for dueling bandits (CPE-DB): we have multiple candidates for multiple positions as modeled by a bipartite graph, and in each round we sample a duel of two candidates on one position and observe who wins in the duel, with the goal of finding the best candidate-position matching with high probability after multiple rounds of samples. CPE-DB is an adaptation of the original combinatorial pure exploration for multi-armed bandit (CPE-MAB) problem to the dueling bandit setting. We consider both the Borda winner and the Condorcet winner cases. For Borda winner, we establish a reduction of the problem to the original CPE-MAB setting and design PAC and exact algorithms that achieve both the sample complexity similar to that in the CPE-MAB setting (which is nearly optimal for a subclass of problems) and polynomial running time per round. For Condorcet winner, we first design a fully polynomial time approximation scheme (FPTAS) for the offline problem of finding the Condorcet winner with known winning probabilities, and then use the FPTAS as an oracle to design a novel pure exploration algorithm CAR-Cond with sample complexity analysis. CAR-Cond is the first algorithm with polynomial running time per round for identifying the Condorcet winner in CPE-DB.

1. Introduction

Multi-Armed Bandit (MAB) (Lai & Robbins, 1985; Thompson, 1933; Auer et al., 2002; Agrawal & Goyal, 2012) is a classic model that characterizes the exploration-exploitation tradeoff in online learning. The pure exploration task (Even-Dar et al., 2006; Chen & Li, 2016; Sabato, 2019) is an important variant of the MAB problems, where the objective is to identify the best arm with high confidence, using as few samples as possible. A rich class of pure exploration problems have been extensively studied, e.g., best K-arm identification (Kalyanakrishnan et al., 2012) and multi-bandit best arm identification (Bubeck et al., 2013). Recently, Chen et al. (2014) proposes a general combinatorial pure exploration for multi-armed bandit (CPE-MAB) framework, which encompasses previous pure exploration problems. In the CPE-MAB problem, a learner is given a set of arms and a collection of arm subsets with certain combinatorial structures. At each time step, the learner plays an arm and observes the random reward, with the objective of identifying the best combinatorial subset of arms. Gabillon et al. (2016); Chen et al. (2017) follow this setting and further improve the sample complexity.

However, in many real-world applications involving implicit (human) feedback including social surveys (Alwin & Krosnick, 1985), market research (Ben-Akiva et al., 1994) and recommendation systems (Radlinski et al., 2008), the information observed by the learner is intrinsically relative. For example, in voting and elections, it is more natural for the electors to offer preference choices than numerical evaluations on candidates. For this scenario, the dueling bandit formulation (Yue et al., 2012; Ramamohan et al., 2016; Sui et al., 2018) provides a promising model for online decision making with relative feedback.

In this paper, we contribute a model adapting the original CPE-MAB problem to the dueling bandit setting. Specifically, we formulate the *combinatorial pure exploration for dueling bandit (CPE-DB)* problem as follows. A CPE-DB instance consists of a bipartite graph G modeling multiple candidates that could fit into multiple positions, and an *unknown preference probability matrix* specifying when we play a duel between two candidates for one position, the probability that the first would win over the second. At each time step, a learner samples a duel of two candidates on one position and observes a random outcome of which candidate wins in this duel sampled according to the preference probability matrix. The objective is to use as few duel samples as possible to identify the best candidate-position matching with high confidence, for two popular optimality metrics

[*]Alphabetical order [1]Microsoft Research, Beijing, China [2]IIIS, Tsinghua University, Beijing, China. Correspondence to: Wei Chen <weic@microsoft.com>, Yihan Du <duyh18@mails.tsinghua.edu.cn>, Longbo Huang <longbohuang@mail.tsinghua.edu.cn>, Haoyu Zhao <zhaohy16@tsinghua.org.cn>.

Proceedings of the 37th International Conference on Machine Learning, Online, PMLR 119, 2020. Copyright 2020 by the author(s).

in the dueling bandit literature, i.e., Condorcet winner and Borda winner.

The CPE-DB model represents a novel preference-based version of the common candidate-position matching problems, which occurs in various real-world scenarios, including social choice (McLean, 1990), multi-player game (Graepel & Herbrich, 2006) and online advertising (Joachims et al., 2017). For instance, a committee selection procedure (Gehrlein, 1985) may want to choose among multiple candidates one candidate for each position to form a committee. For any two candidates on one position, we can play a duel on them, e.g., by surveying a bystander, to learn a sample of which candidate would win on this position, and the sample follows an unknown preference probability. We hope to play as few duels as possible (or by surveying as few people as possible) to identify the best performing committee.

The CPE-DB problem raises interesting challenges on exponentially large decision space and relative feedback. The key issue here is how to exploit the problem structure and design algorithms that guarantee both high computational efficiency and low sample complexity. Therefore, the design and analysis of algorithms for CPE-DB demand novel computational acceleration techniques. The contributions of this work are summarized as follows:

(1). We formulate the combinatorial pure exploration for dueling bandit (CPE-DB) problem, adapted from the original combinatorial pure exploration for multi-armed bandit (CPE-MAB) problem to the dueling bandit setting, and associate it with various real-world applications involving preference-based bipartite matching selection.

(2). For the Borda winner metric, we reduce CPE-DB to the original CPE-MAB problem, and design algorithms CLUCB-Borda-PAC and CLUCB-Borda-Exact with polynomial running time per round. We provide their sample complexity upper bounds and a problem-dependent lower bound for CPE-DB with Borda winner. Our upper and lower bound results together show that CLUCB-Borda-Exact achieves near-optimal sample complexity for a subclass of problems.

(3). For the Condorcet winner metric, we design a fully polynomial time approximation scheme (FPTAS) for a proper extended version of the offline problem, and then adopt the FPTAS to design a novel online algorithm CAR-Cond with sample complexity analysis. To our best knowledge, CAR-Cond is the first algorithm with polynomial running time per round for identifying the Condorcet winner in CPE-DB.

1.1. Related Works

Combinatorial pure exploration The combinatorial pure exploration for multi-armed bandit (CPE-MAB) problem is first formulated by Chen et al. (2014) and generalizes the multi-armed bandit pure exploration task to general combinatorial structures. Gabillon et al. (2016) follow the setting of (Chen et al., 2014) and propose algorithms with improved sample complexity but a loss of computational efficiency. Chen et al. (2017) further design algorithms for this problem that have tighter sample complexity and pseudo-polynomial running time. Wu et al. (2015) study another combinatorial pure exploration case in which given a graph, at each time step, a learner samples a path with the objective of identifying the optimal edge.

Dueling bandit The dueling bandit problem (Yue et al., 2012; Ramamohan et al., 2016; Sui et al., 2018), first proposed by (Yue et al., 2012), is an important variation of the multi-armed bandit setting. According to the assumptions on preference structures and definitions of the optimal arm (winner), previous methods can be categorized as methods on Condorcet winner (Komiyama et al., 2015; Xu et al., 2019), methods on Borda winner (Jamieson et al., 2015; Xu et al., 2019), methods on Copeland winner (Wu & Liu, 2016; Agrawal & Chaporkar, 2019), etc. Recently, Saha & Gopalan (2019) propose a variant of combinatorial bandits with relative feedback. In their setting, a learner plays a subset of arms (assuming each arm has an unknown positive value) in a time step and observes the ranking feedback, and the goal is to minimize the cumulative regret. Therefore, their model is quite different from ours.

2. Problem Formulation

In this section, we formally define the combinatorial pure exploration problem for dueling bandits. Suppose that there are n candidates $C = \{c_1, \dots, c_n\}$ and ℓ positions $S = \{s_1, \dots, s_\ell\}$ with $n \geq \ell$. Each candidate is available for several positions, and we use bipartite graph $G(C, S, E)$ to denote this relation, where each edge $e = (c_i, s_j) \in E$ denotes that candidate c_i is capable for position s_j. We define $m = |E|$. We use E_j to denote the set of edges connected to position j, i.e., $E_j = \{e = (c, s_j) \in E : c \in C\}$ and we also use $s(e)$ to denote the position index of e.

Two edges e and e' are comparable if they have the same position indices, i.e. $s(e) = s(e')$. For any two comparable edges $e = (c, s_j)$ and $e' = (c', s_j)$, there is an unknown preference probability $p_{e,e'}$, which means that with probability $p_{e,e'}$, e wins e', or c wins c' on position j. We have $p_{e,e'} = 1 - p_{e',e}$. For any $e \in E$, we define $p_{e,e} = \frac{1}{2}$.

Given the graph $G(C, S, E)$, we define an order of edges in E by first ranking them by their position indices from smallest to the largest and then ranking them by their candidate index from the smallest to the largest. Given the order of the edges, we use e_i to denote the i-th edge in the order, and define $\chi_M \in \{0, 1\}^m$ as the vector representation of

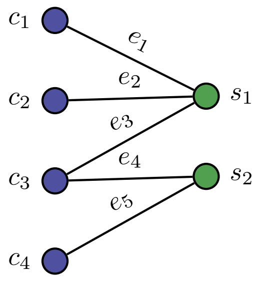

Figure 1. Graph

	e_1	e_2	e_3	e_4	e_5
e_1	0.5	0.45	1	0	0
e_2	0.55	0.5	0.55	0	0
e_3	0	0.45	0.5	0	0
e_4	0	0	0	0.5	0
e_5	0	0	0	1	0.5

Figure 2. Preference Matrix

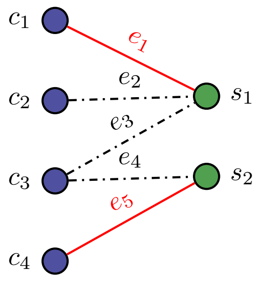

Figure 3. Borda Winner

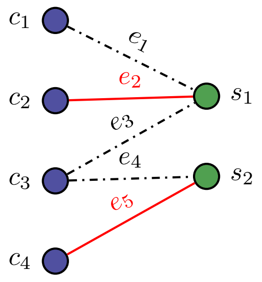

Figure 4. Condorcet Winner

the edges $M \subset E$, where $(\chi_M)_i = 1$ if and only if $e_i \in M$. We also use a preference matrix $P \in ([0,1])^{m \times m}$ to record all preference probabilities. Specifically, for any two comparable edges e_i, e_j, $P_{i,j} = p_{e_i,e_j}$ is the preference probability of e_i over e_j. For two incomparable edges $e_{i'}, e_{j'}$, $P_{i',j'}$ is set to 0 for the convenience of later computations. Figure 1 show an example bipartite graph and Figure 2 shows its corresponding preference matrix.

Note that for each position s_j, any two edges connecting to s_j can be compared with a preference probability. This is similar to the dueling bandit setting (Yue et al., 2012), where each edge is an arm, and we can compare the arms (edges) to find the best arm (edge), i.e., finding the best candidate for a position. Thus, from now on, we will use arms and edges interchangeably. We define $K = \sum_{j=1}^{\ell} \frac{|E_j|(|E_j|-1)}{2}$, which is the number of all possible duels between any two comparable arms.

We assume that there is at least one matching with cardinality ℓ in G, meaning that we can find at least one candidate for each position without a conflict. The *decision class* $\mathcal{M} \subset 2^E$ is the set of all maximum matchings in G. We can also view a matching as a team that specifies which candidate shall play which position for all the positions. Given a matching M and a position s_j, we use $e(M,j)$ to represent the edge in M that connects to position s_j. For any two matchings $M_1, M_2 \in \mathcal{M}$, we define the preference probability of M_1 over M_2 as follows:

$$f(M_1, M_2, P) := \frac{1}{\ell} \sum_{j=1}^{\ell} p_{e(M_1,j),e(M_2,j)}. \qquad (1)$$

It is easy to show that $f(M_1, M_2, P) = 1 - f(M_2, M_1, P)$. Written in vector representation, we have $f(M_1, M_2, P) = \frac{1}{\ell}\chi_{M_1}^T \cdot P \cdot \chi_{M_2}$.

Now, we define the "best" matching in the decision class $\mathcal{M}$. There are several different definitions, e.g., Borda winner (Emerson, 2013; 2016), Condorcet winner (Black, 1948), and Copeland winner (Copeland, 1951; Saari & Merlin, 1996). In this paper, we focus on the Borda winner and the Condorcet winner, the definitions of which are given below.

Borda winner The Borda winner refers to the winner that maximizes the average preference probability over the decision class, which we call "Borda score". Mathematically, in our framework, the Borda score of any matching $M_x \in \mathcal{M}$ and the Borda winner are defined as:

$$B(M_x) = \frac{1}{|\mathcal{M}|} \sum_{M_y \in \mathcal{M}} f(M_x, M_y, P), \qquad (2)$$

$$M_*^B = \operatorname*{argmax}_{M_x \in \mathcal{M}} B(M_x). \qquad (3)$$

For the pure exploration task, we assume that there is a unique Borda winner, similar to the assumption in other pure exploration tasks (Even-Dar et al., 2006; Bubeck et al., 2013; Chen et al., 2014; 2017). Figure 3 shows the Borda winner as the matching with the red edges, because according to the preference matrix in Figure 2, it has the largest Borda score of 0.64.

Condorcet winner The Condorcet winner is the matching that always wins when compared to others. In our framework, the Condorcet winner is defined as the matching M_*^C such that $f(M_*^C, M, P) \geq \frac{1}{2}$ for any matching $M \in \mathcal{M}$. We assume that the Condorcet winner exists as several previous works (Zoghi et al., 2014; Komiyama et al., 2015; Chen & Frazier, 2017) do, and the Condorcet winner wins over any other matching with probability strictly better than $\frac{1}{2}$, i.e. $f(M_*^C, M, P) > \frac{1}{2}$ for any $M \in \mathcal{M} \setminus \{M_*^C\}$. Figure 4 shows the Condorcet winner as the matching with the red edges. It is different from the Borda winner in this example, since this matching wins over all other matchings, but its average winning score (the Borda score) 0.615 is not as good as the Borda winner.

Our goal is to find the best matching (Borda winner or the Condorcet winner) by exploring the duels at all the positions, and we want the number of duels that we need to explore as small as possible. This is the problem of combinatorial pure exploration for dueling bandits (CPE-DB). More precisely, at the beginning, the graph $G(C, S, E)$ is given to the learner, but the preference matrix P is unknown. Because the learner does not know the preference probability for arms connected to the same position, she needs to sample the duel between edges. In each round the learner samples one duel pair (e, e') for some position, and she observes a Bernoulli random variable $X_{e,e'}$ with $\Pr\{X_{e,e'} = 1\} = p_{e,e'}$. The observed feedback could be used to help to select future pairs to sample. Our objective is to find the Borda winner M_*^B or the Condorcet winner M_*^C with as few samples as possible.

3. Efficient Exploration for Borda Winner

In this section, we first show the reduction of the Borda winner identification problem to the combinatorial pure exploration for multi-armed bandit (CPE-MAB) problem, originally proposed and studied in (Chen et al., 2014). Next, we introduce an efficient PAC pure exploration algorithm CLUCB-Borda-PAC for Borda winner, and show that with an almost uniform sampler for perfect matchings (Jerrum et al., 2004), CLUCB-Borda-PAC has both tight sample complexity and fully-polynomial time complexity. Then, based on the PAC algorithm CLUCB-Borda-PAC, we further propose an exact pure exploration algorithm CLUCB-Borda-Exact for Borda Winner, and provide its sample complexity upper bound. Finally, we present the sample complexity lower bound for identifying the Borda winner.

3.1. Reduction to Conventional Combinatorial Pure Exploration

In order to show the reduction of CPE-DB for Borda winner to the conventional CPE-MAB (Chen et al., 2014) problem, we first define the rewards for edges. Then, we define the reward of a matching to be the sum of its edge rewards. Based on the reward definitions, it can be shown that the problem of identifying the Borda winner is equivalent to identifying the matching with the maximum reward. Specifically, for any edge $e = (c_i, s_j) \in E$ and matching $M \in \mathcal{M}$, we define their rewards and the reduction relationship between the two problems as follows:

$$\begin{aligned} w(e) =& \frac{1}{|\mathcal{M}|} \sum_{M \in \mathcal{M}} p_{e,e(M,j)}, \\ w(M) =& \sum_{e \in M} w(e) \overset{(a)}{=} \ell \cdot B(M), \\ M_*^B =& \operatorname*{argmax}_{M \in \mathcal{M}} B(M) = \operatorname*{argmax}_{M \in \mathcal{M}} w(M), \end{aligned} \tag{4}$$

where the equality (a) is due to the definitions of the Borda score (Eq. (2)) and preference probability between two matchings (Eq. (1)) (see Appendix B.1 for a detailed proof of equality (a)).

It remains to show how to efficiently learn the reward $w(e)$ for edge e, by sampling arm pairs in CPE-DB.

First, we can see that for any edge $e = (c_i, s_j)$, $w(e)$ is exactly the *expected* preference probability of e over $e(\bar{M}, j)$, where $\bar{M}$ is a uniformly sampled matching from $\mathcal{M}$. In other words, we could treat e as a base arm in the CPE-MAB setting with mean reward $w(e)$, and we could obtain an unbiased sample for e if we can uniformly sample $\bar{M}$ from $\mathcal{M}$ and then play the duel $(e, e(\bar{M}, j))$ to observe the outcome. However, a naive sampling method on $\mathcal{M}$ would take exponential time. To resolve this issue, we employ a fully-polynomial almost uniform sampler for perfect matchings (Jerrum et al., 2004) $\mathcal{S}(\eta)$ to obtain an almost uniformly sampled matching M' from $\mathcal{M}$. Below we give the formal definition of $\mathcal{S}(\eta)$.

Definition 1. *An almost uniform sampler for perfect matchings is a randomized algorithm $\mathcal{S}(\eta)$ that, if given any bipartite graph G and bias parameter η, it returns a random perfect matching from a distribution π' that satisfies*

$$d_{\text{tv}}(\pi', \pi) = \frac{1}{2} \sum_{x \in \Theta} |\pi'(x) - \pi(x)| \leq \eta,$$

where d_{tv} is the total variation, Θ is the set of all perfect matchings in G and π is the uniform distribution on Θ.

Next, we show how to obtain M' using $\mathcal{S}(\eta)$. We add some ficticious vertices in S and ficticious edges in E to construct a new bipartite graph $G'(C, S', E')$ where $|C| = |S'|$. There is a one-to-n relationship between a maximum matchings in G and a perfect matchings in G'. Then, with $\mathcal{S}(\eta)$, we can almost uniformly sample a maximum matching M' from G in fully-polynomial time. We defer the details for sampling with $\mathcal{S}(\eta)$ to Appendix B.2.

3.2. Efficient PAC Pure Exploration Algorithm

In the previous subsection, we present a reduction of CPE-DB for Borda winner to the conventional CPE-MAB (Chen et al., 2014) problem. However, directly applying the existing CLUCB algorithm in (Chen et al., 2014) cannot obtain an efficient algorithm for our problem. The main obstacle is that there is currently no efficient algorithm to sample from an exact uniform distribution over all the maximum matchings in a general bipartite graph, and thus the original CLUCB algorithm is not directly applicable. To tackle this problem, we need to use an approximate sampler and modify the original CLUCB algorithm to handle the bias introduced by the approximate sampler.

Algorithm 1 illustrates an efficient PAC pure exploration algorithm CLUCB-Borda-PAC for the Borda winner case. Given a confidence level δ and an accuracy requirement ε, CLUCB-Borda-PAC returns an approximate Borda winner Out such that $B(\mathsf{Out}) \geq B(M_*^B) - \varepsilon$ with probability at least $1 - \delta$.

CLUCB-Borda-PAC is built on the CLUCB (Chen et al., 2014) algorithm designed for the conventional CPE-MAB problem, and CLUCB-Borda-PAC efficiently transforms the original numerical observations to the equivalent relative observations. In particular, the maximization oracle $\mathsf{MWMC}(\cdot)$ called in CLUCB-Borda-PAC is exactly the maximum-weighted maximum-cardinality matching algorithm, performed in fully-polynomial time. The main structure follows the CLUCB algorithm: in each round, we first use the empirical mean $\bar{\boldsymbol{w}}_t$ as the input to the oracle $\mathsf{MWMC}(\cdot)$ to find a matching M_t. Then we use the lower confidence bounds for all edges in M_t and upper confidence bounds for all edges outside M_t as the input and call $\mathsf{MWMC}(\cdot)$ again to find an adjusted matching $\tilde{M}_t$. If the difference in weights of the adjusted and non-adjusted matchings are small (line 15), the algorithm stops and returns M_t as the final matching. If not, the algorithm finds the edge z_t in the symmetric difference of $\tilde{M}_t$ and M_t. Then, the algorithm samples a matching M' using sampler $\mathcal{S}(\eta)$, and plays a duel between z_t and the corresponding edge in M' with the same position as z_t. After playing the duel, the algorithm observes the result and updates empirical mean $\bar{w}_{t+1}(z_t)$. With the fast maximization oracle $\mathsf{MWMC}(\cdot)$ and sampler $\mathcal{S}(\eta)$, the CLUCB-Borda-PAC algorithm can be performed in fully-polynomial time.

To formally state the sample complexity upper bound of the CLUCB-Borda-PAC algorithm, we need to first define the width of G, the Borda gap and the Borda hardness.

Definition 2 (Width). *For a bipartite graph G, let $\mathcal{M}(G)$ denote the set of all its maximum matchings. For any $M_1, M_2 \in \mathcal{M}(G)$ such that $M_1 \neq M_2$, we define* $\mathrm{width}(M_1, M_2)$ *as the number of edges of the maximum connected component in their union graph. Then, we define the width of bipartite graph G as*

$$\mathrm{width}(G) = \max_{\substack{M_1, M_2 \in \mathcal{M}(G) \\ M_1 \neq M_2}} \mathrm{width}(M_1, M_2).$$

This width definition for bipartite maximum matching is inline with the general width definition in (Chen et al., 2014). We establish the equivalency between our width definition for bipartite maximum matching and that in (Chen et al., 2014), and defer the proof to Appendix B.3.

Definition 3 (Borda gap). *We define the Borda gap Δ_e^B for any edge $e \in E$ as*

$$\Delta_e^B = \begin{cases} w(M_*^B) - \max\limits_{M \in \mathcal{M}: e \in M} w(M) & \text{if } e \notin M_*, \\ w(M_*^B) - \max\limits_{M \in \mathcal{M}: e \notin M} w(M) & \text{if } e \in M_*, \end{cases}$$

where we make the convention that the maximum value of an empty set is $-\infty$.

Algorithm 1 CLUCB-Borda-PAC

```
1: Input: confidence $\delta$, accuracy $\varepsilon$, bipartite graph $G$, maximization oracle $\mathsf{MWMC}(\cdot)$: $\mathbb{R}^m \to \mathcal{M}$ and almost uniform sampler for perfect matchings $\mathcal{S}(\eta)$
2: Set bias parameter $\eta \leftarrow \frac{1}{8}\varepsilon$
3: Initialize $T_1(e) \leftarrow 0$ and $\bar{w}_1(e) \leftarrow 0$ for all $e \in E$
4: for $t = 1, 2, ...$ do
5:     $M_t \leftarrow \mathsf{MWMC}(\bar{\boldsymbol{w}}_t)$
6:     Compute confidence radius $c_t(e) \leftarrow \sqrt{\frac{\ln(\frac{4Kt^3}{\delta})}{2T_t(e)}}$ for all $e \in E$   // $\frac{x}{0} := 1$ for any $x$
7:     for all $e \in E$ do
8:         if $e \in M_t$ then
9:             $\tilde{w}_t(e) \leftarrow \bar{w}_t(e) - c_t(e) - \frac{1}{4}\varepsilon$
10:        else
11:            $\tilde{w}_t(e) \leftarrow \bar{w}_t(e) + c_t(e) + \frac{1}{4}\varepsilon$
12:        end if   // $\bar{w}_t(e) := 0$ if $T_t(e) = 0$
13:    end for
14:    $\tilde{M}_t \leftarrow \mathsf{MWMC}(\tilde{\boldsymbol{w}}_t)$
15:    if $\tilde{w}_t(\tilde{M}_t) - \tilde{w}_t(M_t) \le \ell\varepsilon$ then
16:        Out $\leftarrow M_t$
17:        return Out
18:    end if
19:    $z_t \leftarrow \arg\max_{e \in (\tilde{M}_t \setminus M_t) \cup (M_t \setminus \tilde{M}_t)} c_t(e)$
20:    Sample a matching $M'$ from $\mathcal{M}$ using $\mathcal{S}(\eta)$
21:    Pull the duel $(z_t, e')$, where $e' = e(M', s(z_t))$
22:    Update $\bar{w}_{t+1}(z_t) \leftarrow \frac{\bar{w}_t(z_t) \cdot T_t(z_t) + X_t(z_t)}{T_t(z_t)+1}$ where $X_t(z_t)$ takes value 1 if $z_t$ wins, 0 otherwise, and $T_{t+1}(z_t) \leftarrow T_t(z_t) + 1$
23: end for
```

Definition 4 (Borda hardness). *We define the hardness H^B for identifying Borda winner in CPE-DB as*

$$H^B := \sum_{e \in E} \frac{1}{(\Delta_e^B)^2}.$$

The Borda gap and Borda hardness definitions are naturally inherited from those in (Chen et al., 2014). For each edge $e \notin M_*^B$, the Borda gap Δ_e^B is the sub-optimality of the best matching that includes edge e, while for each edge $e \in M_*^B$ the Borda gap Δ_e^B is the sub-optimality of the best matching that does not include edge e. The Borda hardness H^B is the sum of inverse squared Borda gaps, which represents the problem hardness for identifying the Borda winner.

Now we present a problem-dependent upper bound of the sample complexity for the CLUCB-Borda-PAC algorithm.

Theorem 1 (CLUCB-Borda-PAC). *With probability at least $1 - \delta$, the* CLUCB-Borda-PAC *algorithm (Algorithm 1) returns an approximate Borda winner* Out *such that*

$B(\mathsf{Out}) \geq B(M_*^B) - \varepsilon$ *with sample complexity*

$$O\left(H_\varepsilon^B \ln\left(\frac{H_\varepsilon^B}{\delta}\right)\right),$$

where $H_\varepsilon^B := \sum_{e\in E} \min\left\{\frac{\mathrm{width}(G)^2}{(\Delta_e^B)^2}, \frac{1}{\varepsilon^2}\right\}$.

We can see that when the accuracy parameter ε is small enough, H_ε^B coincides with the hardness metric H^B. We defer the detailed proof of Theorem 1 to Appendix B.4.

3.3. Efficient Exact Pure Exploration Algorithm

Based on the PAC algorithm CLUCB-Borda-PAC, we further design an efficient exact pure exploration algorithm CLUCB-Borda-Exact for Borda winner and analyze its sample complexity upper bound. Generally speaking, CLUCB-Borda-Exact performs CLUCB-Borda-PAC as a sub-procedure, and guesses the smallest Borda gap $\Delta_{\mathsf{min}}^B := \min_{e\in E} \Delta_e^B$. Iterating epoch $q = 1, 2, \dots$, we set accuracy $\varepsilon_q = \frac{1}{2^q}$ and confidence $\delta_q = \frac{\delta}{2q^2}$. CLUCB-Borda-Exact will guess $\Delta_{\mathsf{min}}^B > \ell\varepsilon_q$, and call CLUCB-Borda-PAC as a sub-procedure with parameters ε_q, δ_q. If the adjusted matching $\tilde{M}_t$ has exactly the same weight as the non-adjusted matching M_t ($\tilde{w}_t(\tilde{M}_t) = \tilde{w}_t(M_t)$, similar as in line 15 of Algorithm 1), then the algorithm stops and returns M_t as the final matching. If $\tilde{w}_t(\tilde{M}_t) \neq \tilde{w}_t(M_t)$ but they differ within $\ell\varepsilon_q$, then the current epoch stops and CLUCB-Borda-Exact will enter the next epoch and cut the guess in half ($\varepsilon_{q+1} = \varepsilon_q/2$). (See Appendix B.5 for the algorithm pseudocode.) Using this technique, we can obtain an algorithm to identify the exact Borda winner with a loss of logarithmic factors in its sample complexity upper bound.

Below we present a problem-dependent upper bound of the sample complexity for the CLUCB-Borda-Exact algorithm and defer the detailed algorithm and proof to Appendix B.5.

Theorem 2 (CLUCB-Borda-Exact)**.** *With probability at least* $1-\delta$*, the* CLUCB-Borda-Exact *algorithm (Algorithm 5) returns the Borda winner with sample complexity*

$$O\Bigg(\mathrm{width}(G)^2 H^B \cdot \ln\left(\frac{\ell}{\Delta_{\mathsf{min}}^B}\right)\cdot \left(\ln\left(\frac{\mathrm{width}(G) H^B}{\delta}\right) + \ln\ln\left(\frac{\ell}{\Delta_{\mathsf{min}}^B}\right)\right)\Bigg),$$

where $\Delta_{\mathsf{min}}^B := \min_{e\in E} \Delta_e^B$.

3.4. Lower Bound

To formally state our result for lower bound, we first introduce the definition of δ-correct algorithm as follows. For any $\delta \in (0,1)$, we call an algorithm $\mathbb{A}$ a δ-correct algorithm if, for any problem instance of CPE-DB with Borda winner, algorithm $\mathbb{A}$ identifies the Borda winner with probability at least $1-\delta$.

Now we give a problem-dependent lower bound on the sample complexity for CPE-DB with Borda winner.

Theorem 3 (Borda lower bound)**.** *Consider the problem of combinatorial pure exploration for identifying the Borda winner. Suppose that, for some constant* $\gamma \in (0, \frac{1}{4})$, $\frac{1}{2} - \gamma \leq p_{e_i,e_j} \leq \frac{1}{2} + \gamma$, $\forall e_i, e_j \in E$ *and* $\frac{|\mathcal{M}|}{|\mathcal{M}|-|\mathcal{M}_e|} \leq \frac{1-4\gamma}{4\gamma\ell}$, $\forall e \in E$*. Then, for any* $\delta \in (0, 0.1)$*, any* δ*-correct algorithm has sample complexity* $\Omega\left(H^B \ln\left(\frac{1}{\delta}\right)\right)$*, where* $\mathcal{M}_e := \{M \in \mathcal{M} : e \in M\}$.

We defer the detailed proof of Theorem 3 to Appendix B.6.

From the upper bounds (Theorems 1,2) and lower bound (Theorem 3), we see that when ignoring the logarithmic factors, our algorithms are tight on the hardness metric H^B. However, whether the width(G) factor is tight or not remains unclear and we leave it for future investigation.

4. Efficient Exploration for Condorcet Winner

In this section, we introduce the efficient pure exploration algorithm CAR-Cond to find a Condorcet winner. We first introduce the efficient pure exploration part assuming there exists "an oracle" that performs like a black-box, and we show the correctness and the sample complexity of CAR-Cond given the oracle. Next, we present the details of the oracle and show that the time complexity of the oracle is polynomial. Then, we apply the verification technique (Karnin, 2016) to improve our sample complexity further. Finally, we give the sample complexity lower bound for finding the Condorcet winner.

4.1. Efficient Pure Exploration Algorithm: CAR-Cond

We first introduce our algorithm CAR-Cond for CPE-DB for the Condorcet winner assuming that there is a proper "oracle". Note that finding the Condorcet winner if existed is equivalent to the following optimization problem,

$$\max_{x=\chi_{M_1}} \min_{y=\chi_{M_2}} \frac{1}{\ell} x^T P y,$$

where $M_1, M_2 \in \mathcal{M}$ are feasible matchings and the value is optimal when $x = y = \chi_{M_*^C}$. This is because if M_1 is not the Condorcet winner M_*^C, it will lose to M_*^C with score $\chi_{M_1}^T P \chi_{M_*^C} < 1/2$, and only when $x = \chi_{M_*^C}$, $\min_{y=\chi_{M_2}} \frac{1}{\ell} x^T P y$ reaches $1/2$ when $y = \chi_{M_*^C}$. However, the optimization problem is "discrete" and we first use the continuous relaxation technique to solve the following optimization problem

$$\max_{x\in\mathcal{P}(\mathcal{M})}\min_{y\in\mathcal{P}(\mathcal{M})}\frac{1}{\ell}x^TPy, \tag{5}$$

where $\mathcal{P}(\mathcal{M})=\{\sum_i\lambda_i\chi_{M_i}:M_i\in\mathcal{M},\sum_i\lambda_i=1,\lambda_i\geq 0\}$ is the convex hull of the vectors $\chi_M, M\in\mathcal{M}$. There is an algorithm that can solve x,y approximately in polynomial time, but solving the optimization problem of Eq. (5) is not enough for our CPE-DB problem. Therefore, we need the following more powerful oracle.

We assume that there is an oracle O_ε that takes the inputs $\varepsilon, A_1, R_1, A_2, R_2, Q$, where ε is the error of the oracle, $A_1, R_1, A_2, R_2\subset E$ and $Q\in[0,1]^{m\times m}$. The oracle can approximately solve the following optimization

$$\max_{x\in\mathcal{P}(\mathcal{M},A_1,R_1)}\min_{y\in\mathcal{P}(\mathcal{M},A_2,R_2)}\frac{1}{\ell}x^TQy, \tag{6}$$

where $\mathcal{P}(\mathcal{M},A,R)=\{\sum_i\lambda_i\chi_{M_i}:M_i\in\mathcal{M},A\subset M_i,R\subset(M_i)^c,\sum_i\lambda_i=1,\lambda_i\geq 0\}$ is the convex hull of the vector representations of the matchings, such that all edges in A are included in the matching and none of the edges in R is included in the matching. More specifically, we assume that the oracle O_ε will compute a solution x_0 that satisfies both the constraint and the following guarantee:

$$\begin{aligned}&\min_{y\in\mathcal{P}(\mathcal{M},A_2,R_2)}\frac{1}{\ell}x_0^TQy\\ \geq&\max_{x\in\mathcal{P}(\mathcal{M},A_1,R_1)}\min_{y\in\mathcal{P}(\mathcal{M},A_2,R_2)}\frac{1}{\ell}x^TQy-\varepsilon.\end{aligned}$$

In the algorithm, we only require that the oracle O_ε returns the value $\min_{y\in\mathcal{P}(\mathcal{M},A_2,R_2)}\frac{1}{\ell}x_0^TQy$, not the x_0.

Given the oracle O_ε, the high level idea of CAR-Cond (Algorithm 2) is as follows: If we know how to set the approximation parameter properly, then in every round we partition the edge set E into A, R, and U, where A is the set of the edges that should be included in the Condorcet winner, R is the set of the edges that should be excluded, and U are the remaining undecided edges. In each round, we only sample the duel between two comparable edges in the set U (Line 6). Then, we use the upper and lower confidence bounds to estimate the real preference matrix P (Line 7). After that, for every undecided edge e, we enforce it to be included in the optimal solution or to be excluded in the solution, and use the oracle to see if the included and excluded cases vary much. If so, we classify edge e into A or R in the next round (Line 9). Since we do not know how to set the approximation parameter properly, we use the "doubling trick" to shrink the approximation parameter ε_q by a factor of 2 in each epoch q (Line 4).

For the value of the confidence radius and the upper and lower confidence bound for the matrix P, we use the following quantity for the confidence radius of the winning probability of the duel between any two comparable arms.

Algorithm 2 CAR-Cond

```
1:  Input: Bipartite graph $G$, Oracle $\mathsf{O}_\varepsilon$ with accuracy $\varepsilon$
2:  $A_0\leftarrow\phi, R_0\leftarrow\phi, U_0\leftarrow E, e_0=0$.
3:  for $q=1,2,\dots$ do
4:      $\varepsilon_q\leftarrow\frac{1}{2^q}, e_q\leftarrow\frac{1}{\varepsilon_q^2}$
5:      for $t=e_{q-1}+1,e_{q-1}+2,\dots,e_q$ do
6:          For every $e_1\neq e_2$ and $e_1,e_2\in E_j$ for some $j$ and
            $e_1,e_2\in U_{t-1}$, sample duel between $e_1,e_2$
7:          Compute $\bar{P}_t,\underline{P}_t$
8:          $A_t\leftarrow A_{t-1}, R_t\leftarrow R_{t-1}, U_t\leftarrow U_{t-1}$
9:          for $e\in U_{t-1}$ do
10:             // We use $A,R$ as shorthands for $A_{t-1},R_{t-1}$
11:             InU = $\mathsf{O}_{\varepsilon_q}(A\cup\{e\},R,A,R,\bar{P}_t)$
12:             InL = $\mathsf{O}_{\varepsilon_q}(A\cup\{e\},R,A,R,\underline{P}_t)$
13:             ExU = $\mathsf{O}_{\varepsilon_q}(A,R\cup\{e\},A,R,\bar{P}_t)$
14:             ExL = $\mathsf{O}_{\varepsilon_q}(A,R\cup\{e\},A,R,\underline{P}_t)$
15:             if InL > ExU + $\varepsilon_q$ then
16:                 $A_t\leftarrow A_t\cup\{e\}, U_t\leftarrow U_t\setminus\{e\}$
17:             else if ExL > InU + $\varepsilon_q$ then
18:                 $R_t\leftarrow R_{t-1}\cup\{e\}, U_t\leftarrow U_t\setminus\{e\}$
19:             end if
20:             if $|A_t|=\ell$ then Out $\leftarrow A$, return Out
21:         end for
22:     end for
23: end for
```

$$c_t(e_i,e_j)=\sqrt{\frac{\ln(4Kt^3/\delta)}{2T_t(e_i,e_j)}}, \tag{7}$$

where $T_t(e_i,e_j)$ is the number of duels between two comparable arms e_i,e_j at the beginning of round t. Now given some duels (at least one) between e_i,e_j, we define $\hat{p}_t(e_i,e_j)$ as the empirical winning probability of e_i over e_j *up to* round t's exploration phase, and we define

$$\begin{aligned}\bar{p}_t(e_i,e_j)&:=\min\{1,\hat{p}_t(e_i,e_j)+c_t(e_i,e_j)\},\\ \underline{p}_t(e_i,e_j)&:=\max\{0,\hat{p}_t(e_i,e_j)-c_t(e_i,e_j)\}.\end{aligned} \tag{8}$$

$\bar{p}_t(e_i,e_j)$ and $\underline{p}_t(e_i,e_j)$ can be interpreted as the upper and lower confidence bounds of the winning probability of e_1 over e_2. Then we denote $\bar{P}_t$ as the matrix where $\bar{P}_{t,ij}:=\bar{p}_t(e_i,e_j)$ where i,j are edge indices, $\bar{P}_{t,ii}:=0.5$, and $\bar{P}_{t,ij}=0$ for any 2 incomparable indices. Similarly, we define $\underline{P}_t$ as the matrix where $\underline{P}_{t,ij}:=\underline{p}_t(e_i,e_j)$, $\underline{P}_{t,ii}:=0.5$, and $\underline{P}_{t,ij}=0$ for any two incomparable indices.

Sample complexity for CAR-Cond To present our main result on the sample complexity of CAR-Cond, we need to first introduce the notion of *gap* for each edge and each comparable pair under the Condorcet setting.

Definition 5 (Condorcet gap)**.** *We define the Condorcet gap Δ_e^C of an edge e as the following quantity.*

$$\Delta_e^C = \begin{cases} 1/2 - \max_{\chi_M, e \in M} \frac{1}{\ell} \chi_M^T P \chi_{M_*^C}, & \text{if } e \notin M_*^C \\ 1/2 - \max_{\chi_M, e \notin M} \frac{1}{\ell} \chi_M^T P \chi_{M_*^C}, & \text{if } e \in M_*^C \end{cases}$$

Then we define the gap $\Delta_{e,e'}^C$ for a pair of arms $e \neq e'$ and $e, e' \in E_j$ as the following quantity $\Delta_{e,e'}^C = \max\{\Delta_e^C, \Delta_{e'}^C\}$.

The definition of gap is very similar to the gap defined in (Chen et al., 2014). Intuitively speaking, the definition of the gap of each edge e is a measurement of how easily e will be classified into the accepted set A or the rejected set R. Given the definition of the gap, we have the following main theorem for the Condorcet setting.

Theorem 4 (CAR-Cond)**.** *With probability at least $1-\delta$, algorithm* CAR-Cond *returns the correct Condorcet winner with a sample complexity bounded by*

$$O\left(\sum_{j=1}^{\ell} \sum_{e \neq e', e, e' \in E_j} \frac{1}{(\Delta_{e,e'}^C)^2} \ln\left(\frac{K}{\delta(\Delta_{e,e'}^C)^2}\right)\right).$$

Generally speaking, our algorithm sequentially classifies each edge into M_*^C or $(M_*^C)^c$. The definition of the gap shows the sub-optimality of wrongly classifying each edge, and $\frac{1}{(\Delta_e^C)^2}$ is roughly the number of times to correctly classify the edge e. Because each query is a sample between two edges e, e', the number of query between e, e' is roughly $1/(\Delta_{e,e'}^C)^2$, this is so as when we correctly classify an edge, we will not need to query any pair that contains this edge. Summing over all comparable pairs of edges, we get our upper bound when omitting all logarithm terms.

When there is only one position, our problem reduces to the original dueling bandit problem. In special cases when the Condorcet winner beat every arm with the largest margin (formally, for all arm $i \in [m]$, $i^C = \arg\max_{j \in [m]} \Pr\{j \text{ wins } i\}$), our sample complexity bound is at the same order as the state-of-the-art (Karnin, 2016) when omitting the logarithmic terms.

4.2. Implementation of Oracle

In this part, we present the high level idea of our method to solve the optimization problem (Eq. (6)). If we define

$$g(x) = \min_{y \in \mathcal{P}(\mathcal{M}, A_2, R_2)} \frac{1}{\ell} x^T Q y,$$

then g is concave in x, since $x^T Q y$ is linear in x and the minimum of linear functions is a concave function. Also note that the constraint set $\mathcal{P}(\mathcal{M}, A_2, R_2)$ is a convex set since it is defined as the convex hull of the vector representations. Thus, using the projected sub-gradient ascent method, we can solve the optimization problem by an error of ε in $O(\frac{1}{\varepsilon})$ number of iterations. To do so, we need to address two problems: how to compute the gradient at a given point, and how to compute the projection efficiently.

Algorithm 3 CAR-Parallel

```
1: Input: confidence $\delta < 0.01$, algorithm CAR-Verify
2: Define CAR-Verify$_k, k \in \mathbb{N}$ as the CAR-Verify algorithm with confidence $\frac{\delta}{2^{k+1}}$
3: Simulate $\{$CAR-Verify$_k\}_{k \in \mathbb{N}}$ in parallel
4: for $t = 1, 2, \ldots$ do
5:     for each $k \in \mathbb{N}$ $s.t.$ $t$ mod $2^k = 0$ do
6:         Start or resume CAR-Verify$_k$, allowing only one sample, and then suspend CAR-Verify$_k$
7:         if CAR-Verify$_k$ returns an answer Out$_k$ then
8:             Out $\leftarrow$ Out$_k$
9:             return Out
10:        end if
11:    end for
12: end for
```

The first problem is rather easy to solve, because if we want to compute the sub-gradient at a given point x_0, it suffices to compute the parameter $y_0 = \arg\min_{y \in \mathcal{P}(\mathcal{M}, A_2, R_2)} x_0^T Q y$, and the sub-gradient will be $\frac{1}{\ell} Q y_0 \in \partial_x g(x_0)$. Computing the parameter y_0 can be done in polynomial time, since the minimum cost maximum matching can be solved in polynomial time.

The second problem is the main challenge. Note that there may be an exponentially large number of vertices in the polytope $\mathcal{P}(\mathcal{M}, A_2, R_2)$ because the number of feasible matchings may be exponential, and we cannot solve the projection step in general. However, if we can tolerate some error in the projection step, we may solve the approximate projection in polynomial time by the Frank-Wolfe algorithm. Then, we can set the approximate projection error to be relatively small, so the cumulative error due to the projection can also be bounded. In this way, we can solve the optimization problem Eq. (6) with $poly(1/\varepsilon, m, K, \ell)$ time complexity.

Please see Appendix A.2 for more backgrounds on projected sub-gradient ascent, Frank-Wolfe, and Appendix C.2 for the detailed implementation of the oracle.

4.3. Further Improvements through Verification

Based on the CAR-Cond algorithm, we further design an algorithm CAR-Parallel for identifying Condorcet winner, which uses the parallel simulation technique (Chen & Li, 2015; Chen et al., 2017) and achieves a tighter expected

Algorithm 4 CAR-Verify

```
1: **Input:** confidence $\delta < 0.01$, algorithm CAR-Cond
2: $\delta_0 \leftarrow 0.01$
3: $\hat{M} = \text{CAR-Cond}(\delta_0)$
4: for $t = 1, 2, \dots$ do
5:     Compute $\bar{P}_t, \underline{P}_t$
6:     if $\max_{M \in \mathcal{M} \setminus \{\hat{M}\}} f(M, \hat{M}, \underline{P}_t) \geq \frac{1}{2}$ then
7:         return error
8:     end if
9:     $M_t = \operatorname{argmax}_{M \in \mathcal{M} \setminus \{\hat{M}\}} f(M, \hat{M}, \bar{P}_t)$
10:    if $f(M_t, \hat{M}, \bar{P}_t) \leq \frac{1}{2}$ then
11:        Out $\leftarrow \hat{M}$
12:        return Out
13:    else
14:        $(e_t, f_t) \leftarrow \operatorname{argmax}_{e_t \in M_t \setminus \hat{M}, f_t \in \hat{M} \setminus M_t, s(e_t)=s(f_t)} c_t(e_t, f_t)$
15:        Pull the duel $(e_t, f_t)$ and update empirical means
16:    end if
17: end for
```

sample complexity for small confidence. CAR-Parallel calls a variant of CAR-Cond, named CAR-Verify, which applies the verification technique (Karnin, 2016) to improve the sample complexity of the original CAR-Cond. Specifically, CAR-Verify calls CAR-Cond(δ_0) to obtain a hypothesized Condorcet winner $\hat{M}$ using a constant confidence $\delta_0 > \delta$. Then, CAR-Verify verifies the correctness of $\hat{M}$ using confidence δ. While CAR-Verify loses a part of confidence in order to obtain better sample complexity for small confidence, CAR-Parallel boosts the confidence to δ by simulating a sequence of CAR-Verify in parallel and keeps the obtained better sample complexity in expectation.

Algorithm 3 illustrates the detailed algorithm CAR-Parallel that applies the parallel simulation technique (Chen & Li, 2015; Chen et al., 2017) and achieves a tighter expected sample complexity for small confidence. Algorithm 4 illustrates the sub-procedure CAR-Verify called in CAR-Parallel. CAR-Verify is based on the original algorithm CAR-Cond and employs the verification technique to improve the sample complexity for small confidence.

In order to formally state our result for the CAR-Parallel algorithm, we first introduce the following definitions.

For any $e \notin M_*^C$, we define the verification gap $\tilde{\Delta}_e^C$ as

$$\min_{M \in \mathcal{M} \setminus \{M_*^C\} : e \in M} \left\{ \frac{\ell}{d_{M_*^C, M}} \cdot \left(\frac{1}{2} - \frac{1}{\ell} \chi_M^T P \chi_{M_*^C} \right) \right\},$$

where d_{M_x, M_y} denotes the number of positions with different edges between M_x and M_y, *i.e.*, $d_{M_x, M_y} := \sum_{j=1}^{\ell} \mathbb{I}\{e(M_x, j) \neq e(M_y, j)\}$.

For ease of notation, we define the following quantity

$$H_{\text{ver}}^C := \sum_{e \notin M_*^C} \frac{1}{(\tilde{\Delta}_e^C)^2}.$$

Then, we have the main theorem of the sample complexity of algorithm CAR-Parallel.

Theorem 5 (CAR-Parallel)**.** *Assume the existence of Condorcet winner. Then, given* $\delta < 0.01$*, with probability at least* $1 - \delta$*, the* CAR-Parallel *algorithm (Algorithm 3) will return the Condorcet winner with an expected sample complexity*

$$O\left(\sum_{j=1}^{\ell} \sum_{\substack{e \neq e' \\ e, e' \in E_j}} \frac{\ln \left(K / (\Delta_{e,e'}^C)^2 \right)}{(\Delta_{e,e'}^C)^2} + H_{\text{ver}}^C \ln \left(\frac{H_{\text{ver}}^C}{\delta} \right) \right).$$

To the best of our knowledge, the best sample complexity for pure exploration of Condorcet dueling bandit is $O(n^2/\Delta^2 + n/\Delta^2 \log(1/\delta))$ by (Karnin, 2016) using the verification technique. When reducing our setting to the simple Condorcet dueling bandit ($\ell = 1$), Theorem 5 recovers this result.

We defer the detailed results and proofs to Appendix C.3.

5. Conclusion and Future Work

In this paper, we formulate the combinatorial pure exploration for dueling bandit (CPE-DB) problem. We consider two optimality metrics, Borda winner and Condorcet winner. For Borda winner, we first reduce the problem to CPE-MAB, and then propose efficient PAC and exact algorithms. We provide sample complexity upper and lower bounds for these algorithms. For a subclass of problems the upper bound of the exact algorithm matches the lower bound when ignoring the logarithmic factor. For Condorcet winner, we first design an FPTAS for a properly extended offline problem, and then employ this FPTAS to design a novel online algorithm CAR-Cond. To our best knowledge, CAR-Cond is the first algorithm with polynomial running time per round for identifying the Condorcet winner in CPE-DB.

There are several promising directions worth further investigation for CPE-DB. One direction is to improve the sample complexity of the CAR-Cond algorithm without compromising its computational efficiency, and try to find a lower bound in this case that matches the upper bound. Other directions of interest include studying a more general CPE-DB model than the current candidate-position matching version, or a family of practical preference functions $f(M_1, M_2, P)$ other than linear functions.

Acknowledgement

The work of Yihan Du and Longbo Huang is supported in part by the National Natural Science Foundation of China Grant 61672316, the Zhongguancun Haihua Institute for Frontier Information Technology and the Turing AI Institute of Nanjing.

References

Agrawal, N. and Chaporkar, P. Klucb approach to copeland bandits. *arXiv preprint arXiv:1902.02778*, 2019.

Agrawal, S. and Goyal, N. Analysis of thompson sampling for the multi-armed bandit problem. In *Conference on Learning Theory*, pp. 39–1, 2012.

Alwin, D. F. and Krosnick, J. A. The measurement of values in surveys: A comparison of ratings and rankings. *Public Opinion Quarterly*, 49(4):535–552, 1985.

Auer, P., Cesa-Bianchi, N., and Fischer, P. Finite-time analysis of the multiarmed bandit problem. *Machine Learning*, 47(2-3):235–256, 2002.

Ben-Akiva, M., Bradley, M., Morikawa, T., Benjamin, J., Novak, T., Oppewal, H., and Rao, V. Combining revealed and stated preferences data. *Marketing Letters*, 5(4):335–349, 1994.

Black, D. On the rationale of group decision-making. *Journal of Political Economy*, 56(1):23–34, 1948.

Bubeck, S., Wang, T., and Viswanathan, N. Multiple identifications in multi-armed bandits. In *Proceedings of the 30th International Conference on Machine Learning*, pp. 258–265, 2013.

Bubeck, S. et al. Convex optimization: Algorithms and complexity. *Foundations and Trends® in Machine Learning*, 8(3-4):231–357, 2015.

Chen, B. and Frazier, P. I. Dueling bandits with weak regret. In *Proceedings of the 34th International Conference on Machine Learning*, pp. 731–739. JMLR. org, 2017.

Chen, L. and Li, J. On the optimal sample complexity for best arm identification. *arXiv preprint arXiv:1511.03774*, 2015.

Chen, L. and Li, J. Open problem: Best arm identification: Almost instance-wise optimality and the gap entropy conjecture. In *Conference on Learning Theory*, pp. 1643–1646, 2016.

Chen, L., Gupta, A., Li, J., Qiao, M., and Wang, R. Nearly optimal sampling algorithms for combinatorial pure exploration. In *Conference on Learning Theory*, pp. 482–534, 2017.

Chen, S., Lin, T., King, I., Lyu, M. R., and Chen, W. Combinatorial pure exploration of multi-armed bandits. In *Advances in Neural Information Processing Systems*, pp. 379–387, 2014.

Copeland, A. H. A reasonable social welfare function. Technical report, mimeo, 1951. University of Michigan, 1951.

Deveci, M., Kaya, K., Uçar, B., and Çatalyürek, Ü. V. Gpu accelerated maximum cardinality matching algorithms for bipartite graphs. In *European Conference on Parallel Processing*, pp. 850–861. Springer, 2013.

Emerson, P. The original borda count and partial voting. *Social Choice and Welfare*, 40(2):353–358, 2013.

Emerson, P. *From Majority Rule to Inclusive Politics*. Springer, 2016.

Even-Dar, E., Mannor, S., and Mansour, Y. Action elimination and stopping conditions for the multi-armed bandit and reinforcement learning problems. *Journal of Machine Learning Research*, 7(Jun):1079–1105, 2006.

Gabillon, V., Lazaric, A., Ghavamzadeh, M., Ortner, R., and Bartlett, P. Improved learning complexity in combinatorial pure exploration bandits. In *Artificial Intelligence and Statistics*, pp. 1004–1012, 2016.

Gehrlein, W. V. The condorcet criterion and committee selection. *Mathematical Social Sciences*, 10(3):199–209, 1985.

Graepel, T. and Herbrich, R. Ranking and matchmaking. *Game Developer Magazine*, 25:34, 2006.

Jamieson, K., Katariya, S., Deshpande, A., and Nowak, R. Sparse dueling bandits. In *Artificial Intelligence and Statistics*, pp. 416–424, 2015.

Jerrum, M., Sinclair, A., and Vigoda, E. A polynomial-time approximation algorithm for the permanent of a matrix with nonnegative entries. *Journal of the ACM (JACM)*, 51(4):671–697, 2004.

Joachims, T., Granka, L., Pan, B., Hembrooke, H., and Gay, G. Accurately interpreting clickthrough data as implicit feedback. In *ACM SIGIR Forum*, volume 51, pp. 4–11. Acm New York, NY, USA, 2017.

Kalyanakrishnan, S., Tewari, A., Auer, P., and Stone, P. Pac subset selection in stochastic multi-armed bandits. In *Proceedings of the 29th International Conference on Machine Learning*, volume 12, pp. 655–662, 2012.

Karnin, Z. S. Verification based solution for structured mab problems. In *Advances in Neural Information Processing Systems*, pp. 145–153, 2016.

Kaufmann, E., Cappé, O., and Garivier, A. On the complexity of best-arm identification in multi-armed bandit models. *The Journal of Machine Learning Research*, 17 (1):1–42, 2016.

Komiyama, J., Honda, J., Kashima, H., and Nakagawa, H. Regret lower bound and optimal algorithm in dueling bandit problem. In *Conference on Learning Theory*, pp. 1141–1154, 2015.

Lai, T. L. and Robbins, H. Asymptotically efficient adaptive allocation rules. *Advances in Applied Mathematics*, 6(1): 4–22, 1985.

McLean, I. The borda and condorcet principles: three medieval applications. *Social Choice and Welfare*, 7 (2):99–108, 1990.

Radlinski, F., Kurup, M., and Joachims, T. How does clickthrough data reflect retrieval quality? In *Proceedings of the 17th ACM Conference on Information and Knowledge Management*, pp. 43–52, 2008.

Ramamohan, S. Y., Rajkumar, A., and Agarwal, S. Dueling bandits: Beyond condorcet winners to general tournament solutions. In *Advances in Neural Information Processing Systems*, pp. 1253–1261, 2016.

Saari, D. G. and Merlin, V. R. The copeland method. *Economic Theory*, 8(1):51–76, 1996.

Sabato, S. Epsilon-best-arm identification in pay-per-reward multi-armed bandits. In *Advances in Neural Information Processing Systems*, pp. 2876–2886, 2019.

Saha, A. and Gopalan, A. Combinatorial bandits with relative feedback. In *Advances in Neural Information Processing Systems*, pp. 983–993, 2019.

Saip, H. B. and Lucchesi, C. L. Matching algorithms for bipartite graph. *Relatorio Tecnico*, 700(03), 1993.

Sui, Y., Zoghi, M., Hofmann, K., and Yue, Y. Advancements in dueling bandits. In *Proceedings of the 27th International Joint Conference on Artificial Intelligence*, pp. 5502–5510, 2018.

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

Wu, H. and Liu, X. Double thompson sampling for dueling bandits. In *Advances in Neural Information Processing Systems*, pp. 649–657, 2016.

Wu, Y., Gyorgy, A., and Szepesvari, C. On identifying good options under combinatorially structured feedback in finite noisy environments. In *Proceedings of the 32nd International Conference on Machine Learning*, pp. 1283–1291, 2015.

Xu, L., Honda, J., and Sugiyama, M. Dueling bandits with qualitative feedback. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pp. 5549–5556, 2019.

Yue, Y., Broder, J., Kleinberg, R., and Joachims, T. The k-armed dueling bandits problem. *Journal of Computer and System Sciences*, 78(5):1538–1556, 2012.

Zoghi, M., Whiteson, S., Munos, R., and De Rijke, M. Relative upper confidence bound for the k-armed dueling bandit problem. In *Proceedings of the 31st International Conference on Machine Learning*, pp. II–10, 2014.

Appendix

A. Preliminaries

A.1. Maximum-Weighted Maximum-Cardinality Matching Algorithm

The maximum-weighted maximum-cardinality (MWMC) matching algorithm (Saip & Lucchesi, 1993; Deveci et al., 2013) is a variation of the known maximum-weighted matching algorithm. Given any bipartite graph G with weighted edges, the MWMC algorithm finds the maximum-weighted matching among all maximum-cardinality matchings and operates in fully-polynomial time.

Note the the variant of MWMC, the minimum-weighted maximum-cardinality matching can also be solve efficiently. We first take the negative value of each edge and shift all of them to the positive direction, to make sure every "new" weight is positive. Then we call the MWMC algorithm and find the maximum-weighted maximum-cardinality matching for the new graph. Since the maximum-cardinality are the same for the 2 graphs, the MWMC solution for the new graph is the minimum-weighted maximum-cardinality matching in the original graph.

A.2. Basic concepts and algorithms for convex optimization

In this part, we review some basic definitions, properties, and algorithms in convex optimization. First, we give the definition of convex sets and convex functions. All of the definitions, algorithms, and properties are adapted from (Bubeck et al., 2015).

Definition 6 (Convex Sets and Convex functions)**.** *A set $\mathcal{X} \subset \mathbb{R}^n$ is said to be convex if it contains all of its segments, i.e.*

$$\forall (x, y, \gamma) \in \mathcal{X} \times \mathcal{X} \times [0,1], (1-\gamma)x + \gamma y \in \mathcal{X}.$$

A function $f : \mathcal{X} \to \mathbb{R}$ is said to be convex if $\mathcal{X}$ is a convex set and

$$\forall (x, y, \gamma) \in \mathcal{X} \times \mathcal{X} \times [0,1], f((1-\gamma)x + \gamma y) \le (1-\gamma)f(x) + \gamma f(y).$$

The gradient of a function f is a basic definition. However, there are cases when f does not have gradient at every point, and we have the following definition of subgradient for convex function f.

Definition 7 (Subgradients)**.** *Let $\mathcal{X} \in \mathbb{R}^n$, and $f : \mathcal{X} \to \mathbb{R}$. Then $g \in \mathbb{R}^n$ is a subgradient of f at $x \in \mathcal{X}$ if for any $y \in \mathcal{X}$ one has*

$$f(x) - f(y) \le g^T(x - y).$$

The set of subgradients of f at x is denoted $\partial f(x)$.

Then, we have the definition of Lipschitz and Smoothness.

Definition 8 (Lipschitz and Smoothness)**.** *A continuous function $f(\cdot)$ is ℓ-Lipschitz if:*

$$\forall x_1, x_2, |f(x_1) - f(x_2)| \le \ell ||x_1 - x_2||_2$$

A differentiable function $f(\cdot)$ is β-smooth if:

$$\forall x_1, x_2,\ ||\nabla f(x_1) - \nabla f(x_2)||_2 \le \beta ||x_1 - x_2||_2.$$

Next, we recall the definition of projection. The projection $\Pi(x, \mathcal{X})$ from a point $x \in \mathbb{R}^n$ to a convex set $\mathcal{X} \subset \mathbb{R}^n$ is defined to be

$$\Pi(x, \mathcal{X}) = \arg\min_{y \in \mathcal{X}} ||x - y||_2.$$

The projection of x to $\mathcal{X}$ is the point in $\mathcal{X}$ that is the closest to x. Then, we have the following property of projection.

Proposition 1 (Property of projection)**.** *Let $\mathcal{X} \subset \mathbb{R}^n$ be a convex set. For any $x \in \mathcal{X}, y \in \mathbb{R}^n$, we have*

$$||y - x||_2 \ge ||\Pi(x, \mathcal{X}) - x||_2 + ||\Pi(x, \mathcal{X}) - y||_2^2.$$

The property of projection is a key lemma in the analysis of many convex optimization algorithms, including the one we use in the following sections.

Then, we briefly introduce 2 algorithms for convex optimization: Projected subgradient descent and Frank-Wolfe. We will use these 2 algorithms in our analysis.

Projected subgradient descent The projected subgradient descent acts almost the same as the projected gradient descent algorithm, except that in this case, the gradient may not exist and we use the subgradient. The projected subgradient descent algorithm iterates the following equations for $t \geq 1$:

$$\begin{aligned} y^{(t+1)} =& x^{(t)} - \eta g^{(t)}, \text{where } g^{(t)} \in \partial f(x^{(t)}), \\ x^{(t+1)} =& \Pi(y^{(t+1)}, \mathcal{X}) \end{aligned}$$

We will not directly apply the performance guarantee of the PGD algorithm, so we omit the theoretical guarantee here.

Frank-Wolfe Algorithm For a convex function f defined on a convex set $\mathcal{X}$, given a fixed sequence $\{\gamma_t\}_{t\geq 1}$, the Frank-Wolfe Algorithm iterate as the following for $t \geq 1$:

$$\begin{aligned} y^{(t)} \in& \arg\min_{y\in\mathcal{X}} \nabla f(x^{(t)})^T y \\ x^{(t+1)} =& (1-\gamma_t)x^{(t)} + \gamma_t y^{(t)} \end{aligned}$$

We have the following theoretical guarantee for Frank-Wolfe.

Proposition 2. *Let f be convex and β-smooth function with respect to norm $||\cdot||_2$, and define $D = \sup_{x,y\in\mathcal{X}} ||x-y||_2$, and $\gamma_s = \frac{2}{s+1}$ for $s \geq 1$. Then for any $t \geq 2$, one has*

$$f(x^{(t)}) - f(x^*) \leq \frac{2\beta D^2}{t+1}.$$

B. Omitted Proofs in Section 3

B.1. Reduction to Conventional Combinatorial Pure Exploration

In the following, we give the omitted proof of the equality (a) in Eq. (4).

Recall that the preference probability between two matchings $M_1, M_2 \in \mathcal{M}$ are defined as

$$f(M_1, M_2, P) := \frac{1}{\ell}\sum_{j=1}^{\ell} p_{e(M_1,j),e(M_2,j)}.$$

The Borda score of any matching $M_x \in \mathcal{M}$ and the Borda winner are defined as

$$\begin{aligned} B(M_x) =& \frac{1}{|\mathcal{M}|}\sum_{M_y\in\mathcal{M}} f(M_x, M_y, P) \\ M_*^B =& \operatorname*{argmax}_{M_x\in\mathcal{M}} B(M_x). \end{aligned}$$

The rewards of any edge $e = (c_e, s_j) \in E$ and any matching $M \in \mathcal{M}$ are defined as

$$\begin{aligned} w(e) =& \frac{1}{|\mathcal{M}|}\sum_{M\in\mathcal{M}} p_{e,e(M,j)} \\ w(M) =& \sum_{e\in M} w(e) \overset{(a)}{=} \ell \cdot B(M) \end{aligned}$$

Therefore, we have

$$B(M_x) = \frac{1}{|\mathcal{M}|}\sum_{M_y\in\mathcal{M}} f(M_x, M_y, P)$$

$$
\begin{aligned}
=&\frac{1}{|\mathcal{M}|}\sum_{M_y\in\mathcal{M}}\frac{1}{\ell}\sum_{j=1}^{\ell}p_{e(M_x,j),e(M_y,j)}\\
=&\frac{1}{\ell}\sum_{j=1}^{\ell}\frac{1}{|\mathcal{M}|}\sum_{M_y\in\mathcal{M}}p_{e(M_x,j),e(M_y,j)}\\
=&\frac{1}{\ell}\sum_{j=1}^{\ell}w(e(M_x,j))\\
=&\frac{1}{\ell}\sum_{e\in M_x}w(e)\\
=&\frac{1}{\ell}w(M_x),
\end{aligned}
$$

which completes the proof of the equality (a) in Eq. (4).

With the shown linear relationship between the Borda score of any matching and rewards of its contained edges, we can reduce combinatorial pure exploration for Borda dueling bandits to conventional combinatorial pure exploration.

B.2. Details for applying the almost uniform sampler

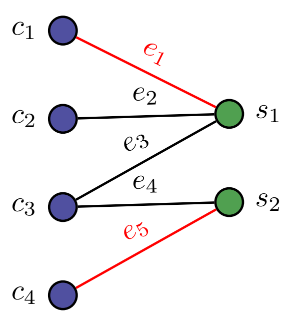

Figure 5. Original bipartite graph G

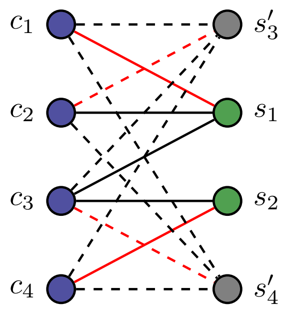

Figure 6. Constructed bipartite graph G'

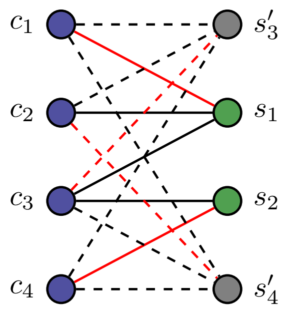

Figure 7. Constructed bipartite graph G'

In this section, we show that how to apply the fully-polynomial almost uniform sampler for perfect matchings (Jerrum et al., 2004) $\mathcal{S}(\eta)$ to obtian an almost uniformly sampled matching M' from $\mathcal{M}$ in bipartite graph G.

Recall that in bipartite graph G, $n=|C|$, $\ell=|S|$. If $n=\ell$, each maximum matching is a perfect matching. Then, we can directly use $\mathcal{S}(\eta)$ to sample a matching almost uniformly.

If $n>\ell$ (note that $n<\ell$ cannot occur due to the assumption of $\mathcal{M}\neq\varnothing$), we add $n-\ell$ ficticious vertices $\{s_{\ell+1},...,s_n\}$ in S. In addition, for each ficticious vertex s_j $(\ell+1\leq j\leq n)$, we add n ficticious edges $(c_1,s_j),...,(c_n,s_j)$ that connected to each vertex in C. Let $G'(C,S',E')$ denote this new bipartite graph. There is a one-to-n relationship between the maximum matchings in G and the perfect matchings in G'. See Figures 5 to 7 for an example. Figure 5 illustrates the original bipartite graph G and a valid maximum matching $M=\{e_1,e_5\}$. Figures 6,7 illustrate the constructed bipartite graph G' and two perfect matchings corresponding to M. The gray vertices s'_3,s'_4 and dashed edges respectively denote the ficticious vertices and edges, and the red edges denote the perfect matchings.

We first use $\mathcal{S}(\eta)$ to almost uniformly sample a perfect matching M'_{perf} in G'. Then, we eliminate the ficticious edges in M'_{perf} and obtain its corresponding maximum matching M' in original G. Because each maximum matching in original G has the same number of corresponding perfect matchings in G', the property of the uniform distribution still holds. Therefore, with $\mathcal{S}(\eta)$, we can obtian an almost uniformly sampled matching M' from $\mathcal{M}$ in bipartite graph G.

B.3. Width of Bipartite Graph

Definition 2 (Width)**.** *For a bipartite graph G, let $\mathcal{M}(G)$ denote the set of all its maximum matchings. For any $M_1,M_2\in\mathcal{M}(G)$ such that $M_1\neq M_2$, we define* $\text{width}(M_1,M_2)$ *as the number of edges of the maximum connected component in*

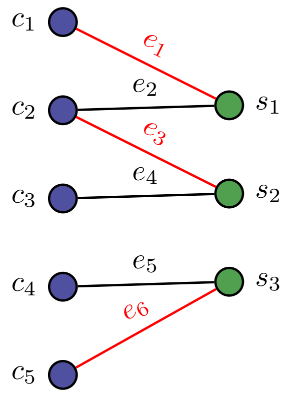

Figure 8. Maximum matching M_1

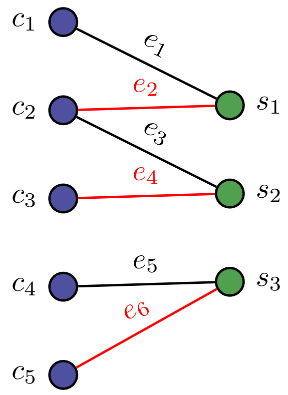

Figure 9. Maximum matching M_2

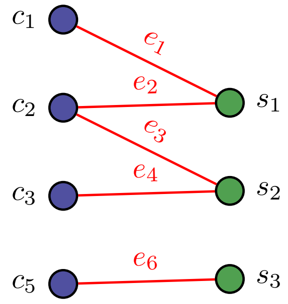

Figure 10. Union graph $G(M_1, M_2)$

their union graph. Then, we define the width of bipartite graph G as

$$\text{width}(G) = \max_{\substack{M_1, M_2 \in \mathcal{M}(G) \\ M_1 \neq M_2}} \text{width}(M_1, M_2).$$

Below we show that our width definition (Definition 2) for bipartite graph is equivalent to that in (Chen et al., 2014).

First, we recall the definitions of exchange set, exchange class and width in (Chen et al., 2014) for the problem instance of bipartite graph and maximum matching.

Exchange set b is defined as an ordered pair of disjoint sets $b = (b_+, b_-)$ where $b_+ \cap b_- = \varnothing$ and $b_+, b_- \subseteq E$. Then, we define operator $\oplus$ such that, for any matching M and any exchange set $b = (b_+, b_-)$, we have $M \oplus b := M \setminus b_- \cup b_+$. Similarly, we also define operator such that $M \ominus b := M \setminus b_+ \cup b_-$.

Exchange class $\mathcal{B}$ for $\mathcal{M}$ is defined as a collection of exchange sets that satisfies the following property. For any $M_1, M_2 \in \mathcal{M}$ such that $M_1 \neq M_2$ and for any $e \in M_1 \setminus M_2$, there exists an exchange set $(b_+, b_-) \in \mathcal{B}$ which satisfies five constraints: (a) $e \in b_-$, (b) $b_+ \subseteq M_2 \setminus M_1$, (c) $b_- \subseteq M_1 \setminus M_2$, (d) $M_1 \oplus b \in \mathcal{M}$ and (e) $M_2 \ominus b \in \mathcal{M}$. We use $\text{Exchange}(\mathcal{M})$ to denote the family of all possible exchange classes for $\mathcal{M}$.

Then, the widths of exchange class $\mathcal{B}$ and decision class $\mathcal{M}$ are defined as follows:

$$\text{width}(\mathcal{B}) = \max_{(b_+, b_-) \in \mathcal{B}} |b_+| + |b_-|,$$

$$\text{width}(\mathcal{M}) = \min_{\mathcal{B} \in \text{Exchange}(\mathcal{M})} \text{width}(\mathcal{B}).$$

We can see that in bipartite graph G, for any $M_1, M_2 \in \mathcal{M}$ such that $M_1 \neq M_2$, their union graph $G(M_1, M_2)$ represents $M_1 \cup M_2$, which can be divided to $(M_1 \setminus M_2) \cup (M_2 \setminus M_1)$ and $M_1 \cap M_2$ (common edges). Let $\mathcal{G}$ denote the connected components of $G(M_1, M_2)$. Then, $\mathcal{G}$ consists of the connected components in $(M_1 \setminus M_2) \cup (M_2 \setminus M_1)$, denoted by $\mathcal{G}_{\text{dif}} = \{G_1(M_1, M_2), G_2(M_1, M_2), \cdots\}$, and those in $M_1 \cap M_2$, denoted by $\mathcal{G}_{\text{com}} = \{e^1, e^2, \cdots\}$. Note that each connected component in $M_1 \cap M_2$ is a single edge. See Figures 8 to 10 for an example. Figures 8,9 illustrate two maximum matchings M_1, M_2 in bipartite graph G respectively and Figure 10 illustrates their union graph $G_(M_1, M_2)$. Then, $G_(M_1, M_2)$ has two connected components, which respectively fall in $\mathcal{G}_{\text{dif}}$ and $\mathcal{G}_{\text{com}}$. Specifically, $\mathcal{G}_{\text{dif}} = \{G_1(M_1, M_2)\}$ where $G_1(M_1, M_2) = \{e_1, e_2, e_3, e_4\}$, and $\mathcal{G}_{\text{com}} = \{e^6\}$.

Then, for any $e \in M_1 \setminus M_2$, there exists some $G_i(M_1, M_2) \in \mathcal{G}_{\text{dif}}$ containing e. Let $b = G_i(M_1, M_2)$, $b_- = M_1 \cap G_i(M_1, M_2)$ and $b_+ = M_2 \cap G_i(M_1, M_2)$. We can see that $M_1 \oplus b \in \mathcal{M}$, $M_2 \ominus b \in \mathcal{M}$, because the other connected components in $G(M_1, M_2)$ do not change and $M_1 \oplus b$, $M_2 \ominus b$ are also valid maximum matchings. Thus, $G_i(M_1, M_2)$ is an exchange set for M_1, M_2 and e that satisfies the five constraints (a)-(e). Similarly, any union of multiple connected components in $\mathcal{G}_{\text{dif}}$ containing $G_i(M_1, M_2)$ is an exchange set for M_1, M_2, e that satisfies the five constraints (a)-(e), and among these exchange sets, $G_i(M_1, M_2)$ has the smallest size. For the example illustrated in Figures 8 to 10,

$G_1(M_1, M_2) = \{e_1, e_2, e_3, e_4\}$ is the exchange set for $M_1, M_2, e,\ s.t.\ e \in \{e_1, e_2, e_3, e_4\}$, and $\text{width}(M_1, M_2) = 4$. In a similar manner, we can see that for any $M_1, M_2 \in \mathcal{M}(G), M_1 \neq M_2$, $\text{width}(M_1, M_2) \leq 4$. Therefore, $\text{width}(G) = \max_{M_1, M_2 \in \mathcal{M}(G), M_1 \neq M_2} \text{width}(M_1, M_2) = 4$.

From the above analysis, we can obtain that the exchange class $\mathcal{B} \in \text{Exchange}(\mathcal{M})$ with minimum $\text{width}(\mathcal{B})$ satisfies that for any $M_1, M_2 \in \mathcal{M}, M_1 \neq M_2$ and for any $e \in M_1 \setminus M_2$, $\mathcal{B}$ only contains the connected component $G_i(M_1, M_2)\ s.t.\ e \in G_i(M_1, M_2)$, not the union of multiple connected components. Thus, the minimum $\text{width}(\mathcal{B})$ over $\mathcal{B} \in \text{Exchange}(\mathcal{M})$ is exactly the maximum $\text{width}(M_1, M_2)$ over any $M_1, M_2 \in \mathcal{M}, M_1 \neq M_2$. Therefore, for the problem instance of bipartite graph and maximum matching, our definition $\text{width}(G) = \max_{M_1, M_2 \in \mathcal{M}(G), M_1 \neq M_2} \text{width}(M_1, M_2)$ is equivalent to that in (Chen et al., 2014).

B.4. Proof of Theorem 1

In order to prove Theorem 1, we first give a brief introduction of the combinatorial pure exploration setting and the CLUCB algorithm in (Chen et al., 2014) and extend the original result to that with biased estimates.

In the setting of combinatorial pure exploration, there are m arms and each arm $e \in [m]$ is associated with a reward distribution with mean $w(e)$. The CLUCB algorithm maintains empirical mean $\bar{w}_t(e)$ and confidence radius $\text{rad}_t(e)$ for each arm $e \in [m]$ and each timestep t. The construction of confidence radius ensures that $|\bar{w}_t(e) - w(e)|| < \text{rad}_t(e)$ holds with high probability for each arm $e \in [m]$ and each timestep t.

In order to prove Theorem 1, we first introduce the following lemma as an extended result of the CLUCB algorithm (Chen et al., 2014) with biased estimates.

Lemma 1 (CLUCB-bias)**.** *In the CLUCB algorithm (Chen et al., 2014), if $\bar{w}(e)$ is a biased estimator of $w(e)$ and $|\mathbb{E}[\bar{w}(e)] - w(e)| \leq \varepsilon < \frac{\Delta_e}{3\text{width}(\mathcal{M})}$. Given any timestep $t > 0$ and suppose that $\forall e \in [m]$, $|\bar{w}(e) - \mathbb{E}[\bar{w}(e)]| < c_t(e)$. For any $e \in [m]$, if $c_t(e) < \frac{\Delta_e}{3\text{width}(\mathcal{M})} - \varepsilon$, then arm e will not be pulled on round t.*

Proof. We first bound the difference between the estimator $\bar{w}_t(e)$ and the reward mean $w(e)$ as follows:

$$\begin{aligned} |\bar{w}_t(e) - w(e)| \leq & |\bar{w}_t(e) - \mathbb{E}[\bar{w}_t(e)]| + |\mathbb{E}[\bar{w}_t(e)] - w(e)| \\ < & c_t(e) + \varepsilon. \end{aligned}$$

Then, the confidence radius $\text{rad}_t(e)$ in the Lemma 10 of (Chen et al., 2014) can be written as $\text{rad}_t(e) = c_t(e) + \varepsilon$ and we obtain that given any timestep $t > 0$, for any $e \in [m]$, if $c_t(e) < \frac{\Delta_e}{3\text{width}(\mathcal{M})} - \varepsilon$, then arm e will not be pulled on round t. □

Theorem 1 (CLUCB-Borda-PAC)**.** *With probability at least $1 - \delta$, the* CLUCB-Borda-PAC *algorithm (Algorithm 1) returns an approximate Borda winner* Out *such that $B(\text{Out}) \geq B(M_*^B) - \varepsilon$ with sample complexity*

$$O\left(H_\varepsilon^B \ln\left(\frac{H_\varepsilon^B}{\delta}\right)\right),$$

where $H_\varepsilon^B := \sum_{e \in E} \min\left\{\frac{\text{width}(G)^2}{(\Delta_e^B)^2}, \frac{1}{\varepsilon^2}\right\}$.

Proof. First, we prove the correctness of the CLUCB-Borda-PAC algorithm (Algorithm 1).

Recall that the empirical mean

$$\bar{w}_t(e) = \frac{\sum_{s=1}^{T_t(e)} X_s(e)}{T_t(e)},$$

where $X_s(e)$ denotes the s-th observation of the duel between e and e' that is selected via the almost uniform sampler $\mathcal{S}(\eta)$. Specifically, $X_s(e)$ takes value 1 if e wins in the s-th observation and takes value 0 otherwise. Note that $X_1(e), X_2(e), \ldots, X_t(e)$ are i.i.d. random variables.

According to the definition of $\mathcal{S}(\eta)$ (Definition 1), in the s-th observation of the duel between e and another edge e', $\mathcal{S}(\eta)$ returns a matching M' from distribution π'_s that satisfies

$$d_{tv}(\pi'_s, \pi) = \frac{1}{2} \sum_{M \in \mathcal{M}} |\pi'_s(x) - \pi(x)| \leq \eta,$$

where π is the uniform distribution on $\mathcal{M}$.

Since e' is the edge at the same position as e in M', we have

$$\mathbb{E}[X_s(e)] = \sum_{M \in \mathcal{M}} \pi'_s(M) \cdot p_{e,e(M,j)},$$

where j is the position index of e.

Since $c_t(e) = \sqrt{\frac{\ln(\frac{4Kt^3}{\delta})}{2T_t(e)}}$, according to the Hoeffding's inequality, we have

$$\begin{aligned}
&\Pr\left[|\bar{w}_t(e) - \mathbb{E}[X_1(e)]| \geq c_t(e)\right] \\
&= \Pr\left[\left|\sum_{s=1}^{T_t(e)} X_s(e)/T_t(e) - \mathbb{E}[X_1(e)]\right| \geq \sqrt{\frac{\ln(\frac{4Kt^3}{\delta})}{2T_t(e)}}\right] \\
&= \sum_{j=1}^{t} \Pr\left[\left|\sum_{s=1}^{j} X_s(e)/j - \mathbb{E}[X_1(e)]\right| \geq \sqrt{\frac{\ln(\frac{4Kt^3}{\delta})}{2j}}, T_t(e) = j\right] \\
&\leq \sum_{j=1}^{t} \Pr\left[\left|\sum_{s=1}^{j} X_s(e)/j - \mathbb{E}[X_1(e)]\right| \geq \sqrt{\frac{\ln(\frac{4Kt^3}{\delta})}{2j}}\right] \\
&\leq \sum_{j=1}^{t} \frac{\delta}{2Kt^3} = \frac{\delta}{2Kt^2}.
\end{aligned}$$

In other words, with probability at least $1 - \frac{\delta}{2Kt^2}$, we have

$$|\bar{w}_t(e) - \mathbb{E}[X_1(e)]| < c_t(e).$$

Recall that $w(e) = \frac{1}{|\mathcal{M}|} \sum_{M \in \mathcal{M}} p_{e,e(M,j)}$ and $\eta = \frac{1}{8}\varepsilon$. Next, we bound the bias between $w(e)$ and $\mathbb{E}[X_1(e)]$.

$$\begin{aligned}
|\mathbb{E}[X_1(e)] - w(e)| &= \left|\sum_{M \in \mathcal{M}} \pi'_1(M) \cdot p_{e,e(M,j)} - \frac{1}{|\mathcal{M}|} \sum_{M \in \mathcal{M}} p_{e,e(M,j)}\right| \\
&= \left|\sum_{M \in \mathcal{M}} \pi'_1(M) \cdot p_{e,e(M,j)} - \sum_{M \in \mathcal{M}} \pi(M) \cdot p_{e,e(M,j)}\right| \\
&= \left|\sum_{M \in \mathcal{M}} p_{e,e(M,j)} \cdot (\pi'_1(M) - \pi(M))\right| \\
&\leq \sum_{M \in \mathcal{M}} p_{e,e(M,j)} \cdot |\pi'_1(M) - \pi(M)| \\
&\leq \sum_{M \in \mathcal{M}} |\pi'_1(M) - \pi(M)| \\
&\leq \frac{1}{4}\varepsilon.
\end{aligned}$$

Combining the above reseults, we have that with probability at least $1 - \frac{\delta}{2Kt^2}$,

$$\begin{aligned}|\bar{w}_t(e) - w(e)| \leq & |\bar{w}_t(e) - \mathbb{E}[X_1(e)]| + |\mathbb{E}[X_1(e)] - w(e)| \\ < & c_t(e) + \frac{1}{4}\varepsilon.\end{aligned}$$

By a union bound over timestep t and edge e, we have that with probability at least $1-\delta$, for any timestep $t > 0$, for any edge $e \in E$, $|\bar{w}_t(e) - w(e)| < c_t(e) + \frac{1}{4}\varepsilon$.

Thus, with probability at least $1-\delta$, when the CLUCB-Borda-PAC algorithm terminates, we have

$$w(M_*^B) - w(\mathsf{Out}) \leq \tilde{w}_t(M_*^B) - \tilde{w}_t(\mathsf{Out}) \leq \tilde{w}_t(\tilde{M}_t) - \tilde{w}_t(\mathsf{Out}) \leq \ell\varepsilon.$$

Thus, according to Eq. (4),

$$B(M_*^B) - B(\mathsf{Out}) = \frac{1}{\ell}(w(M_*^B) - w(\mathsf{Out})) \leq \varepsilon,$$

which completes the proof of the correctness for the CLUCB-Borda-PAC algorithm.

Next, we prove the sample complexity of the CLUCB-Borda-PAC algorithm (Algorithm 1).

In the following case (i) and case (ii), we respectively prove that if $c_t(e) < \frac{\Delta_e^B}{3\text{width}(\mathcal{M})} - \frac{1}{4}\varepsilon$ or if $c_t(e) < \frac{1}{4}\varepsilon$, edge e will not be pulled as the left arm of duel (z_t, e') in the CLUCB-Borda-PAC algorithm, *i.e.*, $z_t \neq e$.

Case (i) If $c_t(e) < \frac{\Delta_e^B}{3\text{width}(\mathcal{M})} - \frac{1}{4}\varepsilon$, where $\frac{1}{4}\varepsilon < \frac{\Delta_e^B}{3\text{width}(\mathcal{M})}$, according to Lemma 1, we obtain $z_t \neq e$.

Case (ii) If $c_t(e) < \frac{1}{4}\varepsilon$, suppose that edge e is pulled at timestep t. Then,

$$\begin{aligned}\tilde{w}_t(\tilde{M}_t) - \tilde{w}_t(M_t) = & \bar{w}_t(\tilde{M}_t) - \bar{w}_t(M_t) + \sum_{e \in (\tilde{M}_t \setminus M_t) \cup (M_t \setminus \tilde{M}_t)} \left(c_t(e) + \frac{1}{4}\varepsilon\right) \\ < & \bar{w}_t(\tilde{M}_t) - \bar{w}_t(M_t) + \sum_{e \in (\tilde{M}_t \setminus M_t) \cup (M_t \setminus \tilde{M}_t)} \left(\frac{1}{4}\varepsilon + \frac{1}{4}\varepsilon\right) \\ \leq & \bar{w}_t(\tilde{M}_t) - \bar{w}_t(M_t) + 2\ell \cdot \left(\frac{1}{4}\varepsilon + \frac{1}{4}\varepsilon\right) \\ \leq & \ell\varepsilon,\end{aligned}$$

which contradicts the stop condition.

Therefore, we have that if $c_t(e) < \max\{\frac{\Delta_e^B}{3\text{width}(\mathcal{M})} - \frac{1}{4}\varepsilon, \frac{1}{4}\varepsilon\}$, then $z_t \neq e$.

Since $\frac{1}{8} \cdot \max\{\frac{\Delta_e^B}{\text{width}(G)}, \varepsilon\} < \max\{\frac{\Delta_e^B}{3\text{width}(\mathcal{M})} - \frac{1}{4}\varepsilon, \frac{1}{4}\varepsilon\}$, we have that if $c_t(e) < \frac{1}{8} \cdot \max\{\frac{\Delta_e^B}{\text{width}(G)}, \varepsilon\}$, edge e will not be pulled as the left arm of duel (z_t, e') in the CLUCB-Borda-PAC algorithm, *i.e.*, $z_t \neq e$.

Fix any edge $e \in E$. Let $T(e)$ denote the number of times edge e being pulled as the left arm of duel (z_t, e'), *i.e.*, $z_t = e$. Let t_e denote the last timestep when $z_t = e$. It is easy to see that $T_{t_e}(e) = T(e) - 1$. According to the above analysis, we see that $c_{t_e}(e) \geq \frac{1}{8} \cdot \max\{\frac{\Delta_e^B}{\text{width}(G)}, \varepsilon\}$. Thus, we have

$$c_{t_e}(e) = \sqrt{\frac{\ln(\frac{4Kt^3}{\delta})}{2(T(e)-1)}} \geq \frac{1}{8} \cdot \max\left\{\frac{\Delta_e^B}{\text{width}(G)}, \varepsilon\right\}$$

$$T(e) \leq 32 \cdot \min\left\{\frac{\text{width}(G)^2}{(\Delta_e^B)^2}, \frac{1}{\varepsilon^2}\right\} \cdot \ln\left(\frac{4KT^3}{\delta}\right) + 1$$

Recall that $H^B_\varepsilon := \sum_{e\in E} \min\{\frac{\text{width}(G)^2}{(\Delta^B_e)^2}, \frac{1}{\varepsilon^2}\}$. Taking summation over $e \in E$, we have

$$T \leq 32 H^B_\varepsilon \ln\left(\frac{4Kt^3}{\delta}\right) + m. \tag{9}$$

Below we prove that

$$T \leq 985 H^B_\varepsilon \ln\left(\frac{4H^B_\varepsilon}{\delta}\right) + 2m. \tag{10}$$

If $m \geq \frac{1}{2}T$, then Eq. (10) holds immediately. Next, we consider the case when $m < \frac{1}{2}T$. Since $T > m$, we can write

$$T = CH^B_\varepsilon \ln\left(\frac{4H^B_\varepsilon}{\delta}\right) + m,$$

where C is some positive constant.

If $C \leq 985$, then we see that Eq. (10) holds. On the contrary, if $C > 985$, from Eq. (9), we have

$$\begin{aligned}
T \leq & m + 32H^B_\varepsilon \ln\left(\frac{4KT^3}{\delta}\right) \\
= & m + 32H^B_\varepsilon \ln\left(\frac{4K}{\delta}\right) + 96H^B_\varepsilon \ln\left(CH^B_\varepsilon \ln\left(\frac{4H^B_\varepsilon}{\delta}\right) + m\right) \\
\leq & m + 32H^B_\varepsilon \ln\left(\frac{4K}{\delta}\right) + 96H^B_\varepsilon \ln\left(2CH^B_\varepsilon \ln\left(\frac{4H^B_\varepsilon}{\delta}\right)\right) \\
= & m + 64H^B_\varepsilon \ln\left(\frac{4K}{\delta}\right) + 96H^B_\varepsilon \ln(2C) + 96H^B_\varepsilon \ln(H^B_\varepsilon) + 96H^B_\varepsilon \ln\left(\ln\left(\frac{4H^B_\varepsilon}{\delta}\right)\right) \\
\leq & m + 64H^B_\varepsilon \ln\left(\frac{4H^B_\varepsilon}{\delta}\right) + 96\ln(2C)H^B_\varepsilon \ln\left(\frac{4H^B_\varepsilon}{\delta}\right) + 96H^B_\varepsilon \ln\left(\frac{4H^B_\varepsilon}{\delta}\right) + 96H^B_\varepsilon \ln\left(\frac{4H^B_\varepsilon}{\delta}\right) \\
= & m + (256 + 96\ln(2C))H^B_\varepsilon \ln\left(\frac{4H^B_\varepsilon}{\delta}\right) \\
< & m + CH^B_\varepsilon \ln\left(\frac{4H^B_\varepsilon}{\delta}\right) \\
= & T,
\end{aligned}$$

which makes a contradiction. Therefore, we have $C \leq 985$ and complete the proof of Eq. (10). Theorem 1 follows immediately from Eq. (10).

□

B.5. Exact Algorithm for Identifying Borda Winner

In Algorithm 5, we present the detailed algorithm CLUCB-Borda-Exact for identifying the exact Borda winner. Then, in the following we give the detailed proof of its sample complexity upper bound (Theorem 2).

Theorem 2 (CLUCB-Borda-Exact)**.** *With probability at least $1-\delta$, the* CLUCB-Borda-Exact *algorithm (Algorithm 5) returns the Borda winner with sample complexity*

$$O\Bigg(\text{width}(G)^2 H^B \cdot \ln\left(\frac{\ell}{\Delta^B_{\text{min}}}\right) \cdot \left(\ln\left(\frac{\text{width}(G) H^B}{\delta}\right) + \ln\ln\left(\frac{\ell}{\Delta^B_{\text{min}}}\right)\right)\Bigg),$$

where $\Delta^B_{\text{min}} := \min_{e\in E} \Delta^B_e$.

Algorithm 5 CLUCB-Borda-Exact

```
1: Input: confidence $\delta$, bipartite graph $G$, decision class $\mathcal{M}$, maximization oracle $\mathsf{O}(\cdot)$: $\mathbb{R}^m \to \mathcal{M}$ and almost uniform sampler for perfect matchings $\mathcal{S}(\eta)$
2: for $q = 1, 2, \ldots$ do
3:     $\varepsilon_q \leftarrow \frac{1}{2^q}$
4:     $\delta_q \leftarrow \frac{\delta}{2q^2}$
5:     Set bias parameter $\eta_q \leftarrow \frac{1}{8}\varepsilon_q$
6:     Initialize $T_1(e) \leftarrow 0$ and $\bar{w}_1(e) \leftarrow 0$ for all $e \in E$
7:     for $t = 1, 2, ...$ do
8:         $M_t \leftarrow \mathsf{O}(\bar{\boldsymbol{w}}_t)$
9:         Compute confidence radius $c_t(e) \leftarrow \sqrt{\frac{\ln(\frac{4Kt^3}{\delta_q})}{2T_t(e)}}$ for all $e \in E$    // $\frac{x}{0} := 1$ for any $x$
10:        for all $e \in E$ do
11:            if $e \in M_t$ then
12:                $\tilde{w}_t(e) \leftarrow \bar{w}_t(e) - c_t(e) - \frac{1}{4}\varepsilon_q$
13:            else
14:                $\tilde{w}_t(e) \leftarrow \bar{w}_t(e) + c_t(e) + \frac{1}{4}\varepsilon_q$
15:            end if   // $\bar{w}_t(e) := 0$ if $T_t(e) = 0$
16:        end for
17:        $\tilde{M}_t \leftarrow \mathsf{O}(\tilde{\boldsymbol{w}}_t)$
18:        if $\tilde{w}_t(\tilde{M}_t) = \tilde{w}_t(M_t)$ then
19:            Out $\leftarrow M_t$
20:            return Out
21:        end if
22:        if $\tilde{w}_t(\tilde{M}_t) - \tilde{w}_t(M_t) \leq \ell\varepsilon_q$ then
23:            break
24:        end if
25:        $z_t \leftarrow \arg\max_{e \in (\tilde{M}_t \setminus M_t) \cup (M_t \setminus \tilde{M}_t)} c_t(e)$
26:        Sample a matching $M'$ from $\mathcal{M}$ using $\mathcal{S}(\eta_q)$
27:        Pull the duel $(z_t, e')$, where $e' = e(M', s(z_t))$
28:        Update empirical means $\bar{w}_t(z_t)$ according to the winning or lossing of $z_t$ and set $T_{t+1}(z_t) \leftarrow T_t(z_t) + 1$
29:     end for
30: end for
```

Proof. First, we prove the correctness of the CLUCB-Borda-Exact algorithm (Algorithm 5).

Note that in epoch q, the CLUCB-Borda-Exact algorithm performs a subroutine of the CLUCB-Borda-PAC algorithm (Algorithm 1) with confidence δ_q and accuracy ε_q. Then, using similar analysis in the proof of the correctness (Theorem 1) of the CLUCB-Borda-PAC algorithm, we have that for any epoch q, with probability at least $1 - \delta_q$, for any edge $e \in E$, $|\bar{w}_t(e) - w(e)| < c_t(e) + \frac{1}{4}\varepsilon_q$.

Since $\sum_{q=1}^{\infty} \delta_q = \sum_{q=1}^{\infty} \frac{\delta}{2q^2} \leq \delta$, by a union bound over q, we have that with probability at least $1 - \delta$, for any epoch q, for any edge $e \in E$, $|\bar{w}_t(e) - w(e)| < c_t(e) + \frac{1}{4}\varepsilon_q$.

Thus, with probability at least $1 - \delta$, when the CLUCB-Borda-Exact algorithm terminates, *i.e.*, $\tilde{w}_t(\tilde{M}_t) = \tilde{w}_t(M_t)$, we have that for any $M \neq M_t$,

$$\tilde{w}_t(M_t) \geq \tilde{w}_t(M)$$

$$\sum_{e \in M_t \setminus M} \left(\bar{w}_t(e) - c_t(e) - \frac{1}{4}\varepsilon_q \right) \geq \sum_{e \in M \setminus M_t} \left(\bar{w}_t(e) + c_t(e) + \frac{1}{4}\varepsilon_q \right)$$

$$\sum_{e \in M_t \setminus M} w(e) > \sum_{e \in M \setminus M_t} w(e)$$

$$w(M_t) > w(M)$$

Therefore, we obtain $\mathsf{Out} = M_t = M_*^B$ and complete the proof of the correctness for the CLUCB-Borda-Exact algorithm.

Next, we prove the sample complexity of the CLUCB-Borda-Exact algorithm (Algorithm 5). Using similar analysis in the proof of the sample complexity (Theorem 1) of the CLUCB-Borda-PAC algorithm, we have that with probability at least $1-\delta_q$, the number of samples in epoch q is bounded by

$$T_q \le O\left(\sum_{e \in E} \min\left\{ \frac{\text{width}(G)^2}{(\Delta_e^B)^2}, \frac{1}{\varepsilon^2} \right\} \ln\left(\frac{1}{\delta_q} \cdot \sum_{e \in E} \min\left\{ \frac{\text{width}(G)^2}{(\Delta_e^B)^2}, \frac{1}{\varepsilon^2} \right\} \right) \right).$$

Let $q^* = \left\lfloor \log_2 (\frac{\ell}{\Delta^B_{\text{min}}}) \right\rfloor + 1$ denote the first epoch that satisfies $\varepsilon^*_q < \frac{\Delta^B_{\text{min}}}{\ell}$. In the following, we show that in epoch q^*, the CLUCB-Borda-Exact algorithm terminates *i.e.*, $\tilde{w}_t(\tilde{M}_t) = \tilde{w}_t(M_t)$ holds before $\tilde{w}_t(\tilde{M}_t) - \tilde{w}_t(M_t) \le \ell \varepsilon_q$.

Suppose that, in epoch q^*, $\tilde{w}_t(\tilde{M}_t) - \tilde{w}_t(M_t) \le \ell \varepsilon^*_q$ holds before $\tilde{w}_t(\tilde{M}_t) = \tilde{w}_t(M_t)$, which implies that the CLUCB-Borda-Exact algorithm enters epoch q^*+1. Then, at the last timestep of epoch q^*, $\tilde{w}_t(\tilde{M}_t) - \tilde{w}_t(M_t) \le \ell \varepsilon^*_q < \Delta^B_{\text{min}}$ and $\tilde{w}_t(\tilde{M}_t) \neq \tilde{w}_t(M_t)$.

Since $\tilde{w}_t(\tilde{M}_t) \neq \tilde{w}_t(M_t)$, $\tilde{M}_t \neq M_t$. Thus, with probability at least $1-\delta$, we have

$$\sum_{e \in \tilde{M}_t \setminus M_t} \left(\bar{w}_t(e) + c_t(e) + \frac{1}{4}\varepsilon_q \right) - \sum_{e \in M_t \setminus \tilde{M}_t} \left(\bar{w}_t(e) - c_t(e) - \frac{1}{4}\varepsilon_q \right) < \Delta^B_{\text{min}}$$

$$\sum_{e \in \tilde{M}_t \setminus M_t} w(e) - \sum_{e \in M_t \setminus \tilde{M}_t} w(e) < \Delta^B_{\text{min}}$$

$$w(\tilde{M}_t) - w(M_t) < \Delta^B_{\text{min}},$$

which contradicts the definition of Δ^B_{min}.

Therefore, in epoch q^*, the CLUCB-Borda-Exact algorithm terminates. Note that if the CLUCB-Borda-Exact algorithm terminates before epoch q^*, our proof of sample complexity still holds.

Now we bound the total number of samples from epoch 1 to q^* as

$$\begin{aligned}
T \le & \sum_{q=1}^{q^*} T_q \\
= & O\left(\sum_{q=1}^{q^*} \sum_{e \in E} \min\left\{ \frac{\text{width}(G)^2}{(\Delta_e^B)^2}, \frac{1}{\varepsilon^2} \right\} \ln\left(\frac{1}{\delta_q} \cdot \sum_{e \in E} \min\left\{ \frac{\text{width}(G)^2}{(\Delta_e^B)^2}, \frac{1}{\varepsilon^2} \right\} \right) \right) \\
= & O\left(\sum_{q=1}^{q^*} \sum_{e \in E} \frac{\text{width}(G)^2}{(\Delta_e^B)^2} \ln\left(\frac{1}{\delta_q} \cdot \sum_{e \in E} \frac{\text{width}(G)^2}{(\Delta_e^B)^2} \right) \right) \\
= & O\left(\sum_{q=1}^{q^*} \text{width}(G)^2 H^B \ln\left(\frac{2q^2}{\delta} \cdot \text{width}(G)^2 H^B \right) \right) \\
= & O\left(\sum_{q=1}^{q^*} \text{width}(G)^2 H^B \left(\ln\left(\frac{\text{width}(G) H^B}{\delta} \right) + \ln q \right) \right) \\
= & O\left(q^* \text{width}(G)^2 H^B \left(\ln\left(\frac{\text{width}(G) H^B}{\delta} \right) + \ln q^* \right) \right)
\end{aligned}$$

$$=O\left(\ln\left(\frac{\ell}{\Delta^B_{\text{min}}}\right)\text{width}(G)^2 H^B\left(\ln\left(\frac{\text{width}(G)H^B}{\delta}\right)+\ln\ln\left(\frac{\ell}{\Delta^B_{\text{min}}}\right)\right)\right),$$

which completes the proof of Theorem 2.

□

B.6. Proof of Theorem 3

Theorem 3 (Borda lower bound)**.** *Consider the problem of combinatorial pure exploration for identifying the Borda winner. Suppose that, for some constant* $\gamma \in (0, \frac{1}{4})$, $\frac{1}{2}-\gamma \le p_{e_i,e_j} \le \frac{1}{2}+\gamma$, $\forall e_i, e_j \in E$ *and* $\frac{|\mathcal{M}|}{|\mathcal{M}|-|\mathcal{M}_e|} \le \frac{1-4\gamma}{4\gamma\ell}$, $\forall e \in E$. *Then, for any* $\delta \in (0, 0.1)$, *any* δ*-correct algorithm has sample complexity* $\Omega\left(H^B \ln\left(\frac{1}{\delta}\right)\right)$, *where* $\mathcal{M}_e := \{M \in \mathcal{M} : e \in M\}$.

Proof. For ease of notation, we first introduce the following definition.

Definition 9 (Next-to-optimal)**.** *For any edge* $e \in E$, *we define the next-to-optimal set associated with e as follows:*

$$M^B_e = \begin{cases} \underset{M\in\mathcal{M}:e\in M}{\text{argmax}}\ w(M) & \text{if } e \notin M^B_*, \\ \underset{M\in\mathcal{M}:e\notin M}{\text{argmax}}\ w(M) & \text{if } e \in M^B_*. \end{cases}$$

Note that, according to the definition of Δ^B_e (Definition 3), we have $w(M^B_*) - w(M^B_e) = \Delta^B_e$.

Fix an instance $\mathcal{I}$ of combinatorial pure exploration for Borda dueling bandits and a δ-correct algorithm $\mathbb{A}$. In instance $\mathcal{I}$, M^B_* is the Borda winner and M_x is a suboptimal super arm. Let T_{e_z,e_k} be the expected number of samples drawn from the duel (e_z, e_k) when $\mathbb{A}$ runs on instance $\mathcal{I}$.

We consider the following alternative instance $\mathcal{I}'$. For an edge $e = (c_i, s_j)$, we change all the distributions of duels $(e, \tilde{e})$ $s.t.$ $\tilde{e} \in E_j \setminus \{e\}$ as follows:

$$p'_{e,\tilde{e}} = \begin{cases} p_{e,\tilde{e}} + \dfrac{|\mathcal{M}|}{|\mathcal{M}|-|\mathcal{M}_e|}\cdot\Delta^B_e & \text{if } e \notin M^B_*, \\ p_{e,\tilde{e}} - \dfrac{|\mathcal{M}|}{|\mathcal{M}|-|\mathcal{M}_e|}\cdot\Delta^B_e & \text{if } e \in M^B_*. \end{cases}$$

Then, for the next-to-optimal matching M^B_e,

$$\begin{aligned} w'(M^B_e) - w'(M^B_*) \ge& w(M^B_e) - w(M^B_*) + \frac{1}{|\mathcal{M}|}\sum_{M\in\mathcal{M}\setminus\mathcal{M}_e}\left|p'_{e,e(M,j)} - p_{e,e(M,j)}\right| \\ =& w(M^B_e) - w(M^B_*) + \frac{1}{|\mathcal{M}|}\sum_{M\in\mathcal{M}\setminus\mathcal{M}_e}\frac{|\mathcal{M}|}{|\mathcal{M}|-|\mathcal{M}_e|}\cdot\Delta^B_e \\ =& w(M^B_e) - w(M^B_*) + \Delta^B_e \\ =& 0. \end{aligned}$$

Thus, we can see that in instance $\mathcal{I}'$, M^B_e is the Borda winner instead.

Using Lemma 1 in (Kaufmann et al., 2016), fixing $e = (c_i, s_j)$, we can obtain

$$\sum_{\tilde{e}\in E_j\setminus\{e\}} T_{e,\tilde{e}}\cdot d(p_{e,\tilde{e}}, p'_{e,\tilde{e}}) \ge d(1-\delta, \delta).$$

For $\delta \in (0, 0.1)$, we have $d(1-\delta,\delta) \ge 0.4\ln(\frac{1}{\delta})$. Suppose that, for some constant $\gamma \in (0, \frac{1}{4})$, $\frac{1}{2}-\gamma \le p_{e_i,e_j} \le \frac{1}{2}+\gamma$, $\forall e_i, e_j \in E$ and $\frac{|\mathcal{M}|}{|\mathcal{M}|-|\mathcal{M}_e|} \le \frac{1-4\gamma}{4\gamma l}$, $\forall e \in E$. Then, for any $e \in E$, $\Delta^B_e \le 2\gamma\ell$. For any $e_i, e_j \in E$ $(e_i \ne e_j)$, $\gamma \le p'_{e_i,e_j} \le 1-\gamma$ and $d(p_{e_i,e_j}, p'_{e_i,e_j}) \le \frac{(p_{e_i,e_j}-p'_{e_i,e_j})^2}{p'_{e_i,e_j}(1-p'_{e_i,e_j})} \le \frac{1}{\gamma(1-\gamma)}(p_{e_i,e_j} - p'_{e_i,e_j})^2$.

Therefore, fixing $e = (c_i, s_j)$, we have

$$\frac{1}{\gamma(1-\gamma)} \sum_{\tilde{e} \in E_j \setminus \{e\}} T_{e,\tilde{e}} \cdot (p_{e,\tilde{e}} - p'_{e,\tilde{e}})^2 \geq 0.4 \ln\left(\frac{1}{\delta}\right)$$

$$\frac{1}{\gamma(1-\gamma)} \left(\frac{|\mathcal{M}|}{|\mathcal{M}| - |\mathcal{M}_e|} \cdot \Delta_e^B\right)^2 \sum_{\tilde{e} \in E_j \setminus \{e\}} T_{e,\tilde{e}} \geq 0.4 \ln\left(\frac{1}{\delta}\right)$$

$$\sum_{\tilde{e} \in E_j \setminus \{e\}} T_{e,\tilde{e}} \geq 0.4\gamma(1-\gamma) \left(\frac{4\gamma\ell}{1-4\gamma}\right)^2 \frac{1}{(\Delta_e^B)^2} \ln\left(\frac{1}{\delta}\right)$$

We can perform the similar distribution changes on any edge $e \in E$. Therefore, we can obtain

$$\begin{aligned}
\sum_{e_z < e_k} T_{e_z,e_k} =& \sum_{j=1}^{\ell} \sum_{\substack{e_z, e_k \in E_j \\ e_z < e_k}} T_{e_z,e_k} \\
=& \frac{1}{2} \sum_{j=1}^{\ell} \sum_{e_z \in E_j} \sum_{e_k \in E_j \setminus \{e_z\}} T_{e_z,e_k} \\
=& \frac{1}{2} \sum_{e \in E} \sum_{\tilde{e} \in E_{s(e)} \setminus \{e\}} T_{e,\tilde{e}} \\
\geq& 0.2\gamma(1-\gamma) \left(\frac{4\gamma\ell}{1-4\gamma}\right)^2 \ln\left(\frac{1}{\delta}\right) \sum_{e \in E} \frac{1}{(\Delta_e^B)^2} \\
=& \Omega\left(H^B \ln\left(\frac{1}{\delta}\right)\right),
\end{aligned}$$

which completes the proof of Theorem 3.

□

C. Omitted Proofs in Section 4

In this section, we will introduce the efficient pure exploration algorithm CAR-Cond to find a Condorcet winner. We will first introduce the efficient pure exploration part assuming there exist "an oracle" that performs like a black-box, and we will show the correctness and the sample complexity of CAR-Cond given the oracle. Next, we will present the details of the oracle, and show that the time complexity of the oracle is polynomial. Then, we will apply the verification framework to further improve our sample complexity. Finally, we will give the sample complexity lower bound for finding the Condorcet winner.

C.1. Accept-reject algorithm for combinatorial pure exploration

In this section, we prove Theorem 4. The proof is divided into 2 parts: the first part shows the correctness of CAR-Cond, and the second part bounds the sample complexity. We begin with the first part.

Correctness of CAR-Cond

Definition 10 (Sampling is nice)**.** *Define event $\mathcal{N}_t := \{\underline{p}_t(e_1, e_2) \leq p_{e_1,e_2} \leq \bar{p}_t(e_1, e_2), \forall e_1 \neq e_2, e_1, e_2 \in E_j\}$. Furthermore, we use $\mathcal{N} = \cap_{t \geq 1} \mathcal{N}_t$ to denote the case when $\mathcal{N}_t$ happens for at every round t.*

We have the following lemma to show that $\mathcal{N}$ is a high probability event.

Lemma 2. *$\mathcal{N}$ is a high probability event. Formally, we have*

$$\Pr\{\neg\mathcal{N}\} \leq \delta.$$

Proof. The proof is an application of the Hoeffding Inequality and the union bound. We first bound $\neg\mathcal{N}_t$, and we have

$$\begin{aligned}
\Pr\{\neg\mathcal{N}_t\} =& \Pr\{\exists e_1, e_j, |\hat{p}_t(e_i,e_j) - p_{e_i,e_j}| > c_t(e_i,e_j)\}\\
\leq& \sum_{Comparable\ e_i,e_j} \Pr\{|\hat{p}_t(e_i,e_j) - p_{e,e_j}| > c_t(e_i,e_j)\}\\
\leq& \sum_{Comparable\ e_i,e_j} \Pr\left\{|\hat{p}_t(e_i,e_j) - p_{e_i,e_j}| > \sqrt{\frac{\ln(4Kt^3/\delta)}{2T_t(e_i,e_j)}}\right\}\\
\leq& \sum_{Comparable\ e_i,e_j} \sum_{k=1}^{t} \Pr\left\{|\hat{p}_t(e_i,e_j) - p_{e_i,e_j}| > \sqrt{\frac{\ln(4Kt^3/\delta)}{2T_t(e_i,e_j)}}, T_t(e_i,e_j) = k\right\}\\
\leq& \sum_{Comparable\ e_i,e_j} \sum_{k=1}^{t} \exp\left(-2k\left(\sqrt{\frac{\ln(4Kt^3/\delta)}{2T_t(e_i,e_j)}}\right)^2\right)\\
\leq& \sum_{Comparable\ e_i,e_j} \sum_{k=1}^{t} \frac{\delta}{4Kt^3}\\
\leq& \frac{\delta}{2t^2}.
\end{aligned}$$

Then we have

$$\begin{aligned}
\Pr\{\neg\mathcal{N}\} =& \Pr\{\exists t \geq 1, \neg\mathcal{N}_t\}\\
\leq& \sum_{t\geq 1} \Pr\{\neg\mathcal{N}_t\}\\
\leq& \sum_{t\geq 1} \frac{\delta}{2t^2}\\
\leq& \delta.
\end{aligned}$$

□

Then we have the key lemma for the correctness of CAR-Cond. The lemma says that, when $\mathcal{N}$ happens, CAR-Cond will not wrongly classify the edges.

Lemma 3. *Suppose the optimal super arm is denoted by M_*^C. If $\mathcal{N}$ happens, then at the end of every round t, we have*

$$A_t \subseteq M_*^C, R_t \subseteq (M_*^C)^c.$$

Proof. We use induction to prove that $A_t \subseteq M_*^C, R_t \subseteq (M_*^C)^c$ at the end of every round t if $\mathcal{N}$ happens.

Note that the optimal matching can be solved by the following minimax optimization problem

$$\max_{x\in\chi_\mathcal{M}} \min_{y\in\chi_\mathcal{M}} \frac{1}{\ell} x^T P y.$$

It is known that when $x = \chi_{M_*^C}$, the minimax optimization problem will reach its optimal value $\frac{1}{2}$. Because our assumption, we have for any $y \in \chi_\mathcal{M}, y \neq \chi_{M_*^C}$,

$$\frac{1}{\ell}\chi_{M_*^C}^T P y \geq \frac{1}{2} + \Delta^{Cond}, \frac{1}{\ell} y^T P \chi_{M_*^C} \leq \frac{1}{2} - \Delta^{Cond}.$$

Suppose that at time $t-1$, the induction is correct, i.e. $A_{t-1} \subseteq M_*^C, R_{t-1} \subseteq (M_*^C)^c$. We use $\mathcal{P}(\mathcal{M}, A, R)$ to denote the arm distributions that is a linear combination of the matchings in $\mathcal{M}$ such that A must appear in the super arm and R must not appear in the super arm. Then for any set $\mathcal{P}$, $\mathcal{Q}$ such that $|x|_1 = |y|_1 = 1, \forall x \in \mathcal{P}, y \in \mathcal{Q}$, we have

$$\max_{x\in\mathcal{P}} \min_{y\in\mathcal{Q}} \frac{1}{\ell} x^T \underline{P} y \leq \max_{x\in\mathcal{P}} \min_{y\in\mathcal{Q}} \frac{1}{\ell} x^T P y \leq \max_{x\in\mathcal{P}} \min_{y\in\mathcal{Q}} \frac{1}{\ell} x^T \bar{P} y.$$

Suppose that time t belongs to epoch q. If $e \in M_*^C$, then we have

$$\begin{aligned}
\text{ExL} \leq & \max_{x \in \mathcal{P}(\mathcal{M}, A_{t-1}, R_{t-1} \cup \{e\})} \min_{y \in \mathcal{P}(\mathcal{M}, A_{t-1}, R_{t-1})} \frac{1}{\ell} x^T \underline{P} y \\
\leq & \max_{x \in \mathcal{P}(\mathcal{M}, A_{t-1}, R_{t-1} \cup \{e\})} \frac{1}{\ell} x^T \underline{P} \chi_{M_*^C} \\
\leq & \max_{x \in \mathcal{P}(\mathcal{M}, A_{t-1}, R_{t-1} \cup \{e\})} \frac{1}{\ell} x^T P \chi_{M_*^C} \\
\leq & \frac{1}{2}.
\end{aligned}$$

We also have

$$\begin{aligned}
\text{InU} + \varepsilon_q \geq & \max_{x \in \mathcal{P}(\mathcal{M}, A_{t-1} \cup \{e\}, R_{t-1})} \min_{y \in \mathcal{P}(\mathcal{M}, A_{t-1}, R_{t-1})} \frac{1}{\ell} x^T \bar{P} y \\
\geq & \min_{y \in \mathcal{P}(\mathcal{M}, A_{t-1}, R_{t-1})} \frac{1}{\ell} \chi_{M_*^C}^T \bar{P} y \\
\geq & \min_{y \in \mathcal{P}(\mathcal{M}, A_{t-1}, R_{t-1})} \frac{1}{\ell} \chi_{M_*^C}^T P y \\
\geq & \frac{1}{2}.
\end{aligned}$$

Then we know that $\text{InU} + \varepsilon \geq \text{ExL}$ and the algorithm will not put e into the set R_t. On the other hand, if $e \notin M_*^C$, then

$$\begin{aligned}
\text{InL} \leq & \max_{x \in \mathcal{P}(\mathcal{M}, A_{t-1} \cup \{e\}, R_{t-1})} \min_{y \in \mathcal{P}(\mathcal{M}, A_{t-1}, R_{t-1})} \frac{1}{\ell} x^T \underline{P} y \\
\leq & \max_{x \in \mathcal{P}(\mathcal{M}, A_{t-1} \cup \{e\}, R_{t-1})} \frac{1}{\ell} x^T \underline{P} \chi_{M_*^C} \\
\leq & \max_{x \in \mathcal{P}(\mathcal{M}, A_{t-1} \cup \{e\}, R_{t-1})} \frac{1}{\ell} x^T P \chi_{M_*^C} \\
\leq & \frac{1}{2},
\end{aligned}$$

and

$$\begin{aligned}
\text{ExU} + \varepsilon_q \geq & \max_{x \in \mathcal{P}(\mathcal{M}, A_{t-1}, R_{t-1} \cup \{e\})} \min_{y \in \mathcal{P}(\mathcal{M}, A_{t-1}, R_{t-1})} \frac{1}{\ell} x^T \bar{P} y \\
\geq & \min_{y \in \mathcal{P}(\mathcal{M}, A_{t-1}, R_{t-1})} \frac{1}{\ell} \chi_{M_*^C}^T \bar{P} y \\
\geq & \min_{y \in \mathcal{P}(\mathcal{M}, A_{t-1}, R_{t-1})} \frac{1}{\ell} \chi_{M_*^C}^T P y \\
\geq & \frac{1}{2}.
\end{aligned}$$

Thus, we know that $\text{ExU} + \varepsilon \geq \text{InL}$ and the algorithm will not put e into the set A_t. □

With the help of Lemma 3, we have the following lemma summarize the correctness of CAR-Cond.

Lemma 4 (Correctness of CAR-Cond)**.** *When $\mathcal{N}$ happens, if* CAR-Cond *stops, then* CAR-Cond *will return the Condorcet winner M_*^C.*

Proof. When CAR-Cond stops at round t, it means that $|A_t| = \ell$. Then from the previous lemma (Lemma 3), we know that when $\mathcal{N}$ happens, $A_t \subseteq M_*^C$. However, $M_*^C = \ell$ and thus $A_t = M_*^C$. In this way, CAR-Cond returns the correct (unique) Condorcet winner. □

Sample complexity of CAR-Cond In the previous part, we show that if the algorithm stops, then with high probability, the output is correct. Now in this part, we show that with high probability, the algorithm with stop, and formally, we bound the sample complexity of CAR-Cond. First, we recall the definition of Gap in the Condorcet winner case.

Definition 5 (Condorcet gap)**.** *We define the Condorcet gap Δ_e^C of an edge e as the following quantity.*

$$\Delta_e^C = \begin{cases} 1/2 - \max_{\chi_M, e\in M} \frac{1}{\ell}\chi_M^T P \chi_{M_*^C}, & \text{if } e \notin M_*^C \\ 1/2 - \max_{\chi_M, e\notin M} \frac{1}{\ell}\chi_M^T P \chi_{M_*^C}, & \text{if } e \in M_*^C \end{cases}$$

Then we define the gap $\Delta_{e,e'}^C$ for a pair of arms $e \neq e'$ and $e, e' \in E_j$ as the following quantity $\Delta_{e,e'}^C = \max\{\Delta_e^C, \Delta_{e'}^C\}$.

Lemma 5 (Sample Complexity of CAR-Cond)**.** *If $\mathcal{N}$ happens, the sample complexity of* CAR-Cond *is bounded by*

$$O\left(\sum_{j=1}^{\ell}\sum_{e_1\neq e_2, e_1, e_2\in E_j} \frac{1}{(\Delta_{e_1,e_2}^C)^2}\ln\left(\frac{K}{\delta(\Delta_{e_1,e_2}^C)^2}\right)\right).$$

Proof. We first prove that, at round t in epoch q such that $e \in U_t$ and $c_t < \frac{\Delta_e}{6}$ for $\Delta_e > 6\varepsilon_q$, the algorithm CAR-Cond will classify arm e into either A_{t+1} or R_{t+1}. For simplicity, we denote $c_t := \sqrt{\frac{\ln(4Kt^3/\delta)}{2t}}$, and it is the confidence radius for those arms in set U_t after the exploration in round t.

Case 1. If arm $e \in M_*^C, \Delta_e > 6\varepsilon_q, c_t < \frac{\Delta_e}{6}$, and $e \notin A_{t-1}$, we show that $e \in A_t$. Note that if $\mathcal{N}$ happens, we have

$$\begin{aligned} \text{InL} + \varepsilon_q \geq & \max_{x\in\mathcal{P}(\mathcal{M}, A_{t-1}\cup\{e\}, R_{t-1})}\ \min_{y\in\mathcal{P}(\mathcal{M}, A_{t-1}, R_{t-1})} \frac{1}{\ell}x^T \underline{P} y \\ \geq & \min_{y\in\mathcal{P}(\mathcal{M}, A_{t-1}, R_{t-1})} \frac{1}{\ell}\chi_{M_*^C}^T \underline{P} y \\ \geq & \min_{y\in\mathcal{P}(\mathcal{M}, A_{t-1}, R_{t-1})} \frac{1}{\ell}\chi_{M_*^C}^T P y - \frac{1}{\ell}\ell 2c_t \\ \geq & \frac{1}{2} - 2c_t, \end{aligned}$$

and

$$\begin{aligned} \text{ExU} \leq & \max_{x\in\mathcal{P}(\mathcal{M}, A_{t-1}, R_{t-1}\cup\{e\})}\ \min_{y\in\mathcal{P}(\mathcal{M}, A_{t-1}, R_{t-1})} \frac{1}{\ell}x^T \bar{P} y \\ \leq & \max_{x\in\mathcal{P}(\mathcal{M}, A_{t-1}, R_{t-1}\cup\{e\})} \frac{1}{\ell}x^T \bar{P}\chi_{M_*^C} \\ \leq & \max_{x\in\mathcal{P}(\mathcal{M}, A_{t-1}, R_{t-1}\cup\{e\})} \frac{1}{\ell}x^T \bar{P}\chi_{M_*^C} + \frac{1}{\ell}\ell 2c_t \\ \leq & \frac{1}{2} - \Delta_e + 2c_t. \end{aligned}$$

The reasons between the inequality between line 2 and line 3 are: 1. The matrix $P_t, \underline{P}_{t-1}$ are all diagonal block matrices and they can be partitioned into ℓ small nonzero matrices; 2. Although for the edge $e' \in A_{t-1}\cup R_{t-1}$, the confidence radius is larger than c_t, however, in the computation we will never use that larger confidence radius. If $e' \in A_{t-1}$, then at the same position in the matching, y also chooses e' and we know the exact value $P_{e',e'} = \frac{1}{2}$. If $e' \in R_{t-1}$, both x and y will have 0 weight on the entry corresponding to e', and the confidence radius related to e' does not matter.

Then we have

$$\begin{aligned} \text{InL} - \text{ExL} - \varepsilon_q \geq & \frac{1}{2} - 2c_t - \left(\frac{1}{2} - \Delta_e + 2c_t\right) - 2\varepsilon_q \\ = & \Delta_e - 2\varepsilon_q - 4c_t \end{aligned}$$

$$
\begin{aligned}
&> \Delta_e - \frac{2\Delta_e}{6} - \frac{4\Delta_e}{6} \\
&= 0,
\end{aligned}
$$

where we use the assumption that $\Delta_e > 6\varepsilon_q, c_t < \frac{\Delta_e}{6}$.

Case 2. If arm $e \notin M_*^C, \Delta_e > 6\varepsilon_q, c_t < \frac{\Delta_e}{6}$, and $e \notin R_{t-1}$, we show that $e \in R_t$.

$$
\begin{aligned}
\text{ExL} + \varepsilon_q &\geq \max_{x \in \mathcal{P}(\mathcal{M}, A_{t-1}, R_{t-1} \cup \{e\})} \min_{y \in \mathcal{P}(\mathcal{M}, A_{t-1}, R_{t-1})} \frac{1}{\ell} x^T \underline{P} y \\
&\geq \min_{y \in \mathcal{P}(\mathcal{M}, A_{t-1}, R_{t-1})} \frac{1}{\ell} \chi_{M_*^C}^T \underline{P} y \\
&\geq \min_{y \in \mathcal{P}(\mathcal{M}, A_{t-1}, R_{t-1})} \frac{1}{\ell} \chi_{M_*^C}^T P y - \frac{1}{\ell} \ell 2c_t \\
&\geq \frac{1}{2} - 2c_t,
\end{aligned}
$$

and

$$
\begin{aligned}
\text{InU} &\leq \max_{x \in \mathcal{P}(\mathcal{M}, A_{t-1} \cup \{e\}, R_{t-1})} \min_{y \in \mathcal{P}(\mathcal{M}, A_{t-1}, R_{t-1})} \frac{1}{\ell} x^T \bar{P} y \\
&\leq \max_{x \in \mathcal{P}(\mathcal{M}, A_{t-1} \cup \{e\}, R_{t-1})} \frac{1}{\ell} x^T \bar{P} \chi_{M_*^C} \\
&\leq \max_{x \in \mathcal{P}(\mathcal{M}, A_{t-1} \cup \{e\}, R_{t-1})} \frac{1}{\ell} x^T \bar{P} \chi_{M_*^C} + \frac{1}{\ell} \ell 2c_t \\
&\leq \frac{1}{2} - \Delta_e + 2c_t.
\end{aligned}
$$

Then we have

$$
\begin{aligned}
\text{ExL} - \text{InL} - \varepsilon_q &\geq \frac{1}{2} - 2 \cdot c_t - \left(\frac{1}{2} - \Delta_e + 2c_t \right) - 2\varepsilon_q \\
&= \Delta_e - 2\varepsilon_q - 4c_t \\
&> \Delta_e - \frac{2\Delta_e}{6} - \frac{4\Delta_e}{6} \\
&= 0,
\end{aligned}
$$

where we use the assumption that $\Delta_e > 6\varepsilon_q, c_t < \frac{\Delta_e}{6}$.

Now we bound the round t_e such that an edge e is added to A_{t_e+1} or R_{t_e+1}. Note that previously, we prove that when $\mathcal{N}$ at round t in epoch q such that $e \in U_t$ and $c_t < \frac{\Delta_e}{6}$ for $\Delta_e > 6\varepsilon_q$, the algorithm CAR-Cond will classify arm e into either A_{t+1} or R_{t+1}. Note that when we select $t'_e = \frac{162}{(\Delta_e^C)^2} \ln\left(\frac{162K}{\delta(\Delta_e^C)^2}\right)$, we know that t'_e is in epoch q'_e such that $\varepsilon_{q'_e} \leq \frac{\Delta_e^C}{6}$ since

$$
\frac{1}{\left(\frac{(\Delta_e^C)^2}{6}\right)^2} < \frac{162}{(\Delta_e^C)^2} \ln\left(\frac{162K}{\delta(\Delta_e^C)^2}\right).
$$

Recall that the confidence radius is defined as follow $c_t = \sqrt{\frac{\ln(4Kt^3/\delta)}{2t}}$, and we have the following

$$
\begin{aligned}
c_{t'_e} &= \sqrt{\frac{\ln(4K(t'_e)^3/\delta)}{2t'_e}} \\
&= \sqrt{\frac{\ln(4K/\delta)}{2t'_e} + \frac{3\ln(t'_e)}{2t'_e}}
\end{aligned}
$$

$$\begin{aligned}
&\le \sqrt{\frac{\ln(4K/\delta)}{2\frac{162}{(\Delta_e^C)^2}\ln\left(\frac{162K}{\delta(\Delta_e^C)^2}\right)} + \frac{3\ln(162/(\Delta_e^C)^2) + 3\ln\ln\left(\frac{162K}{(\delta\Delta_e^C)^2}\right)}{2\frac{162}{(\Delta_e^C)^2}\ln\left(\frac{162K}{\delta(\Delta_e^C)^2}\right)}} \\
&< \sqrt{\frac{(\Delta_e^C)^2}{2\times 162} + \frac{3(\Delta_e^C)^2}{2\times 162} + \frac{3(\Delta_e^C)^2}{2\times 162}} \\
&< \sqrt{(\Delta_e^C)^2 \times \frac{3}{108}} \\
&= \frac{\Delta_e^C}{6}.
\end{aligned}$$

Also note that c_t is monotonically decreasing when t increases and $t \ge 3$, and ε_q (as a function of t) is also monotonically decreasing when t increases, so we know that $t_e \le t'_e$ when $t'_e \ge 3$.

Now from the definition of our algorithm, we will sample edges $e_1 \ne e_2$ if and only if they are connected to the same position and $t \le \min\{t_{e_1}, t_{e_2}\}$. Combining the previous bound on t_{e_1}, t_{e_2}, we know that when $\mathcal{N}$ happens, the sample complexity of CAR-Cond is bounded by

$$O\left(\sum_{j=1}^{\ell} \sum_{e_1 \ne e_2, e_1, e_2 \in E_j} \frac{1}{(\Delta_{e_1,e_2}^C)^2} \ln\left(\frac{K}{\delta(\Delta_{e_1,e_2}^C)^2}\right)\right).$$

$\square$

Combining the Correctness lemma (Lemma 4), the Sample Complexity lemma (Lemma 5), and the fact that $\mathcal{N}$ is a high probability event (Lemma 2), we have the following theorem.

Theorem 4 (CAR-Cond). *With probability at least $1-\delta$, algorithm* CAR-Cond *returns the correct Condorcet winner with a sample complexity bounded by*

$$O\left(\sum_{j=1}^{\ell} \sum_{e \ne e', e, e' \in E_j} \frac{1}{(\Delta_{e,e'}^C)^2} \ln\left(\frac{K}{\delta(\Delta_{e,e'}^C)^2}\right)\right).$$

C.2. Details for the oracle implementation

In this section, we introduce the implementation of the oracle used in CAR-Cond. Recall that we use the following oracle: The oracle can approximately solve the following optimization

$$\max_{x \in \mathcal{P}(\mathcal{M}, A_1, R_1)} \min_{y \in \mathcal{P}(\mathcal{M}, A_2, R_2)} \frac{1}{\ell} x^T Q y,$$

where $\mathcal{P}(\mathcal{M}, A, R) = \{\sum_i \lambda_i \chi_{M_i} : M_i \in \mathcal{M}, A \subset M_i, R \subset (M_i)^c, \sum_i \lambda_i = 1\}$ is the convex hull of the vector representations of the matchings, such that all the edge A are included in the matching and all of R are not included in the matching.

First, we give the full detailed algorithm for the implementation of the oracle. Algorithm 6 is the main algorithm and Algorithm 7 is the approximation algorithm.

Recall that the general idea for the implementation of our oracle is to apply the projected sub-gradient descent, and while in the projection step, we use the Frank-Wolfe algorithm to perform the approximate projection step. The proof is organized as follow: 1. We first prove that the function we optimize $\min_{y \in \mathcal{P}(\mathcal{M}, A_2, R_2)} \frac{1}{\ell} x^T Q y$ is a concave function and has the properties that we need to use in the proof (Bounded (Lemma 6) and Lipschitz (Lemma 7)). After the basic properties, we show the main lemma of the approximation projection (Lemma 8). Finally, we combine the projected sub-gradient descent with the approximation oracle (Lemma 9).

Lemma 6. *For any Accepted/Rejected sets A, R, the diameter of the set $\mathcal{P}(\mathcal{M}, A, R)$ is bounded by $2K$. Formally, we have*

$$\sup_{x, y \in \mathcal{P}(\mathcal{M}, A, R)} ||x - y||_2 \le 2\ell.$$

Algorithm 6 Condorcet Oracle (Detailed)

```
1: Input: Bipartite graph $G$, weight matrix $W$ where $w_{i,j}$ denote an estimation of the probability that $i$ wins $j$, Accepted/Rejected Set for $x, y$: $A_x, R_x, A_y, R_y$
2: Goal: Find the approximate optimal solution of $\max_{x \in \mathcal{P}(\mathcal{M}, A_x, R_x)} f_{A_y,R_y}(x)$
3: Initialize $x^{(1)} = \chi_M$ for any possible $M$ such that $A_x \subset M$ and $R_x \subset M^c$
4: Time hozizon $T = \lceil \frac{(4\ell K)^2}{\varepsilon^2} \rceil$, Step size $\eta = \frac{2\ell}{K\sqrt{M}}$, Accuracy $\varepsilon' = \frac{\varepsilon}{2K\sqrt{M}}$ for the approximate projection oracle.
5: for $t = 1, 2, \dots, T$ do
6:     Compute the subgradient $\nabla f_{A_y,R_y}(x)$ at the point $x^{(t)}$
7:     $y^{(t+1)} \leftarrow x^{(t)} + \eta \nabla f_{A_y,R_y}(x^{(t)})$
8:     $x^{(t+1)} \leftarrow \Pi_{\varepsilon'}(y^{(t+1)}, \mathcal{P}(\mathcal{M}, A_x, R_x))$
9: end for
10: return $\left(f_{A_y,R_y}(x^{(t+1)}), x^{(t+1)}\right)$
```

Algorithm 7 Approximate projection by Frank-Wolfe

```
1: Input: Point $x \in \mathbb{R}^K$, Bipartite graph $G$ with maximum matching $\ell$, Accepted set $A$ and Rejected set $R$, Accuracy Parameter $\varepsilon$
2: Output: Approximate projection $y$ such that $||y - \Pi(x, \mathcal{P}(\mathcal{M}, A, R)||_2 \le \varepsilon$
3: $x^{(1)} \leftarrow \chi_M$, where $M$ is any maximum cardinal matching for graph $G$.
4: for $t = 1, 2, \dots, \lceil 16\ell^2/\varepsilon^2 \rceil$ do
5:     $c \leftarrow x^{(t)} - x$
6:     Solve the minimum cost maximum matching for graph $G$ with cost vector $c$
7:     Denote the solution as $\chi_t$
8:     $x^{(t+1)} \leftarrow \left(1 - \frac{2}{t+1}\right) x^{(t)} + \frac{2}{t+1}\chi_t$
9: end for
10: return $x^{(\lceil 8\ell^2/\varepsilon^2 \rceil + 1)}$
```

Proof. First note that, for any $x \in \mathcal{P}(\mathcal{M}, A, R)$, we have $||x||_1 = \ell$, because x is a linear combination of matching with cardinal ℓ. Then we have

$$\begin{aligned}\sup_{x,y \in \mathcal{P}(\mathcal{M},A,R)} ||x - y||_2 \le & \sup_{x,y \in \mathcal{P}(\mathcal{M},A,R)} ||x - y||_1 \\ \le & \sup_{x,y \in \mathcal{P}(\mathcal{M},A,R)} (||x||_1 + ||y||_1) \\ \le & \ell + \ell \\ = & 2\ell.\end{aligned}$$

□

Lemma 7. *Fixing the matrix W, the accepted/rejected sets A_y, R_y, the function*

$$f_{A_y,R_y}(x) = \min_{y \in \mathcal{P}(\mathcal{M},A_y,R_y)} \frac{1}{\ell} x^T W y$$

is concave and K-Lipschitz.

Proof. First, we know that $f_{A_y,R_y}(x)$ is concave, because

$$f_{A_y,R_y}(x) = \min_{y \in \mathcal{P}(\mathcal{M},A_y,R_y)} \frac{1}{\ell} x^T W y,$$

is the minimum of linear functions, and thus is concave. Furthermore, we show that $f_{A_y,R_y}(x)$ is K-Lipschitz. For any x_1, x_2, let $y_2 = \mathrm{argmin}_{y \in \mathcal{P}(\mathcal{M},A_y,R_y)} \frac{1}{\ell} x_2^T W y$, and we have

$$f_{A_y,R_y}(x_1) - f_{A_y,R_y}(x_2) = \min_{y \in \mathcal{P}(\mathcal{M},A_y,R_y)} \frac{1}{\ell} x_1^T W y - \min_{y \in \mathcal{P}(\mathcal{M},A_y,R_y)} \frac{1}{\ell} x_2^T W y$$

$$\begin{aligned}
&\le \frac{1}{\ell}x_1^T W y_2 - \frac{1}{\ell}x_2^T W y_2 \\
&= (x_1 - x_2)^T \frac{1}{\ell} W y_2 \\
&\le ||x_1 - x_2||_2 \cdot ||\frac{1}{\ell} W y_2||_2 \\
&\le K||x_1 - x_2||_2,
\end{aligned}$$

where the last inequality comes from the fact that each entry in W belongs to $[0,1]$ and the 1-norm of y_2 is $||y_2||_1 = \ell$. Similarly, we can also prove that

$$f_{A_y,R_y}(x_2) - f_{A_y,R_y}(x_2) \le K||x_1 - x_2||_2,$$

and we can conclude that $f_{A_y,R_y}(x)$ is K-Lipschitz. □

Then, we come to the proof of the approximation oracle. First we recall the procedure of the Frank-Wolfe Algorithm and recall the performance guarantee. Then we recall the projection lemma that we use in the analysis. We refer to Section A for more background on Frank-Wolfe Algorithm and other basic properties of convex optimization.

For a convex function f defined on a convex set $\mathcal{X}$, given a fixed sequence $\{\gamma_t\}_{t\ge1}$, the Frank-Wolfe Algorithm iterate as the following for $t \ge 1$:

$$\begin{aligned}
y^{(t)} &\in \arg\min_{y\in\mathcal{X}} \nabla f(x^{(t)})^T y \\
x^{(t+1)} &= (1-\gamma_t)x^{(t)} + \gamma_t y^{(t)}
\end{aligned}$$

The following is the performance guarantee of the Frank-Wolfe algorithm.

Proposition 2. *Let f be convex and β-smooth function with respect to norm $||\cdot||_2$, and define $D = \sup_{x,y\in\mathcal{X}} ||x-y||_2$, and $\gamma_s = \frac{2}{s+1}$ for $s \ge 1$. Then for any $t \ge 2$, one has*

$$f(x^{(t)}) - f(x^*) \le \frac{2\beta D^2}{t+1}.$$

Also recall that we have the following property for projecting to a convex set.

Proposition 1 (Property of projection)**.** *Let $\mathcal{X} \subset \mathbb{R}^n$ be a convex set. For any $x \in \mathcal{X}, y \in \mathbb{R}^n$, we have*

$$||y-x||_2 \ge ||\Pi(x,\mathcal{X}) - x||_2 + ||\Pi(x,\mathcal{X}) - y||_2^2.$$

Now we give the lemma of the approximation projection. The lemma is nearly a direct application of the proposition of the Frank-Wolfe performance guarantee and the projection proposition, but we need to carefully choose the parameters.

Lemma 8 (Approximate Projection)**.** *Let $\Pi(x,\mathcal{P}(\mathcal{M},A,R))$ denote the projection of x onto the distribution polytope $\mathcal{P}(\mathcal{M},A,R)$. Algorithm 7 will return a solution x_r such that $||x_r - \Pi(x,\mathcal{P}(\mathcal{M},A,R))||_2 \le \varepsilon$. Moreover, x_r can be represented by $\sum_e \lambda_e \chi_{M_e}$ such that $M \in \mathcal{M}, A \subseteq M_e, R \subseteq M_e^*$ and $\boldsymbol{\lambda}$ is sparse.*

Proof. Denote $x_r = x^{(\lceil 16\ell^2/\varepsilon^2\rceil+1)}$. First we know that x_r is a linear combination of the vertices, and it is easy to see that the coefficient vector $\boldsymbol{\lambda}$ can have at most $\lceil 16\ell^2/\varepsilon^2\rceil$ non-zero entries. Thus, we know that $x_r \in \mathcal{P}(\mathcal{M},A,R)$. From the property of Frank-Wolfe algorithm (Proposition 2), we know that

$$\frac{1}{2}||x-x_r||_2^2 \le \frac{1}{2}||x - \Pi(x,\mathcal{P}(\mathcal{M},A,R))||_2^2 + \frac{1}{2}\varepsilon^2,$$

since $D = \sup_{x,y\in\mathcal{P}(\mathcal{M},A,R)} ||x-y||_2 \le 2\ell$ and the function $f(y) = \frac{1}{2}||x-y||_2^2$ is 1-smooth. Then, from the property of projection (Proposition 1), we know that

$$||x_r - x||_2^2 \ge ||x - \Pi(x,\mathcal{P}(\mathcal{M},A,R))||_2^2 + ||x_r - \Pi(x,\mathcal{P}(\mathcal{M},A,R))||_2^2.$$

Then we know that

$$||x_r - \Pi(x,\mathcal{P}(\mathcal{M},A,R))||_2^2 \le \varepsilon^2,$$

and complete the proof of this lemma. □

By the help of the previous lemmas, we have the following main lemma for our minimax oracle. The main lemma follows the proof strategy of the projected sub-gradient descent, but we need to substitute the original accurate projection oracle to our approximate projection oracle.

Lemma 9 (Minimax Oracle)**.** *Using the Minimax Oracle (Algorithm 6) with the approximate projection oracle (Algorithm 7), the output $(f_{A_y,R_y}(x_r), x_r)$ satiesfies*

$$f_{A_y,R_y}(x_r) \geq \max_{x \in \mathcal{P}(\mathcal{M},A_x,R_x)} f_{A_y,R_y}(x) - \varepsilon.$$

Proof. Let $T = \frac{(2\ell K)^2}{\varepsilon^2}$ denote the total steps, $\eta =$ denote the step size, and $\varepsilon' =$ denote the accuracy of the approximate projection oracle. We show that

$$f_{A_y,R_y}(x^*) - \max_{t \leq T} f_{A_y,R_y}(x^{(t)}) \leq \varepsilon.$$

We have

$$\begin{aligned} f(x^*) - f(x^{(t)}) \leq& \nabla f(x^{(t)})^T (x^* - x^{(t)}) \\ =& \frac{1}{\eta}(y^{(t+1)} - x^{(t)})^T (x^* - x^{(t)}) \\ =& \frac{1}{2\eta}\left(||x^{(t)} - x^*||_2^2 + ||x^{(t)} - y^{(t+1)}||_2^2 - ||y^{(t+1)} - x^*||_2^2 \right) \\ =& \frac{1}{2\eta}\left(||x^{(t)} - x^*||_2^2 - ||y^{(t+1)} - x^*||_2^2 \right) + \frac{\eta}{2}||f(x^{(t)})||_2^2. \end{aligned}$$

Note that from Proposition 1, we have

$$||y^{(t+1)} - x^*||_2^2 \geq ||\Pi(y^{(t+1)}, \mathcal{P}(\mathcal{M}, A_x, R_x)) - x^*||_2^2.$$

Furthermore, since

$$||x^{(t+1)} - \Pi((y^{(t+1)}, \mathcal{P}(\mathcal{M}, A_x, R_x))||_2 \leq \varepsilon',$$

we have

$$\begin{aligned} & \left| ||x^{(t+1)} - x^*||_2^2 - ||\Pi(y^{(t+1)}, \mathcal{P}(\mathcal{M}, A_x, R_x)) - x^*||_2^2 \right| \\ =& \left| (x^{(t+1)} - \Pi(y^{(t+1)}, \mathcal{P}(\mathcal{M}, A_x, R_x)))^T (x^{(t+1)} + \Pi(y^{(t+1)}, \mathcal{P}(\mathcal{M}, A_x, R_x)) - 2x^*) \right| \\ \leq& ||x^{(t+1)} - \Pi(y^{(t+1)}, \mathcal{P}(\mathcal{M}, A_x, R_x))||_2 \cdot ||x^{(t+1)} + \Pi(y^{(t+1)}, \mathcal{P}(\mathcal{M}, A_x, R_x)) - 2x^*||_2 \\ \leq& ||x^{(t+1)} - \Pi(y^{(t+1)}, \mathcal{P}(\mathcal{M}, A_x, R_x))||_2 \cdot \left(||x^{(t+1)} - x^*||_2 + ||\Pi(y^{(t+1)}, \mathcal{P}(\mathcal{M}, A_x, R_x)) - x^*||_2 \right) \\ \leq& 4\ell\varepsilon', \end{aligned}$$

where in the last step, we use the fact that

$$\sup_{x,y \in \mathcal{P}(\mathcal{M},A_x,R_x)} ||x - y||_2 \leq 2\ell.$$

Sum up all $t \leq T$, apply the fact that $||\nabla f_{A_y,R_y}(x)||_2 \leq K$ (because the function $f_{A_y,R_y}(x)$ is K-Lipschitz), we have

$$\begin{aligned} \sum_{i=1}^{T} \left(f(x^*) - f(x^{(t)}) \right) \leq& \sum_{i=1}^{T} \left(\frac{1}{2\eta}\left(||x^{(t)} - x^*||_2^2 - ||y^{(t+1)} - x^*||_2^2 \right) + \frac{\eta}{2}||f(x^{(t)})||_2^2 \right) \\ \leq& \sum_{i=1}^{T} \left(\frac{1}{2\eta}\left(||x^{(t)} - x^*||_2^2 - ||x^{(t+1)} - x^*||_2^2 + 4\ell\varepsilon' \right) + \frac{\eta}{2}||f(x^{(t)})||_2^2 \right) \\ \leq& \frac{4\ell^2}{2\eta} + \frac{M}{2\eta}4\ell\varepsilon' + \frac{T\eta}{2}K^2, \end{aligned}$$

and we can get

$$\begin{aligned} f_{A_y,R_y}(x^*) - \max_{t\le T} f_{A_y,R_y}(x^{(t)}) \le& \frac{1}{M}\sum_{i=1}^{T}\left(f(x^*) - f(x^{(t)})\right) \\ \le& \frac{4\ell^2}{2T\eta} + \frac{1}{2\eta}4\ell\varepsilon' + \frac{\eta}{2}K^2. \end{aligned}$$

Plug in $T = \lceil \frac{(4\ell K)^2}{\varepsilon^2} \rceil, \eta = \frac{2\ell}{K\sqrt{M}}, \varepsilon' = \frac{\varepsilon}{2K\sqrt{M}}$, we can get

$$\begin{aligned} f_{A_y,R_y}(x^*) - \max_{t\le T} f_{A_y,R_y}(x^{(t)}) \le& \frac{4\ell^2}{2T\eta} + \frac{1}{2\eta}4\ell\varepsilon' + \frac{\eta}{2}K^2 \\ =& \frac{4\ell^2}{2T\frac{2\ell}{K\sqrt{M}}} + \frac{2\ell\varepsilon'}{\frac{2\ell}{K\sqrt{M}}} + \frac{\frac{2\ell}{K\sqrt{M}}}{2}K^2 \\ =& \frac{2\ell K}{\sqrt{M}} + K\sqrt{M}\varepsilon' \\ \le& 2\ell K\frac{\varepsilon}{4\ell K} + \frac{\varepsilon}{2} \\ =& \varepsilon. \end{aligned}$$

□

C.3. Details of the verification algorithm

Recall that we introduce the following definitions.

For any $e \notin M_*^C$, we define the verification gap $\tilde{\Delta}_e^C$ as

$$\tilde{\Delta}_e^C = \min_{M\in\mathcal{M}\setminus\{M_*^C\}: e\in M} \left\{ \frac{\ell}{d_{M_*^C,M}} \cdot \left(\frac{1}{2} - \frac{1}{\ell}\chi_M^T P \chi_{M_*^C}\right) \right\},$$

where d_{M_x,M_y} denotes the number of positions with different edges between M_x and M_y, *i.e.*, $d_{M_x,M_y} := \sum_{j=1}^{\ell} \mathbb{I}\{e(M_x,j) \neq e(M_y,j)\}$.

For ease of notation, we define the following quantity

$$H_{\text{ver}}^C := \sum_{e\notin M_*^C} \frac{1}{(\tilde{\Delta}_e^C)^2}.$$

Next, we present two lemmas for CAR-Verify on the sample complexity and correctness with high probability.

Lemma 10 (CAR-Verify)**.** *Assume the existence of Condorcet winner. Then, with probability at least* $1-\delta_0-\delta$*, the* CAR-Verify *algorithm (Algorithm 4) will return the Condorcet winner with sample complexity*

$$O\left(\sum_{j=1}^{\ell} \sum_{\substack{e\neq e' \\ e,e'\in E_j}} \frac{1}{(\Delta_{e,e'}^C)^2} \ln\left(\frac{K}{(\Delta_{e,e'}^C)^2}\right) + H_{\text{ver}}^C \ln\left(\frac{H_{\text{ver}}^C}{\delta}\right) \right).$$

Proof. First, we define event $\mathcal{E} := \{\hat{M} = M_*^C\}$. From Theorem 4, we have $\Pr[\mathcal{E}] \ge 1-\delta_0$. We also define the event $\mathcal{F}_t := \{|\hat{p}_{e_i,e_j} - p_{e_i,e_j}| < c_{e_i,e_j}(t), \forall e_i \neq e_j, s(e_i) = s(e_j)\}$ for any timestep t. Since $c_t(e_i,e_j) = \sqrt{\frac{\ln(4Kt^3/\delta)}{2T_t(e_i,e_j)}}$, from the Chernoff-Hoeffding bound, we can obtain that for any t, for any $e_i, e_j,$ *s.t.* $e_i \neq e_j, s(e_i) = s(e_j)$,

$$\Pr[|\hat{p}_{e_i,e_j} - p_{e_i,e_j}| \ge c_{e_i,e_j}(t)] = \sum_{s=1}^{t} \Pr\left[|\hat{p}_{e_i,e_j} - p_{e_i,e_j}| \ge \sqrt{\frac{\log(\frac{4Kt^3}{\delta})}{2s}}, T_t(e_i,e_j) = s \right]$$

$$\begin{aligned}&\le \sum_{s=1}^{t} \frac{\delta}{2Kt^3}\\&\le \frac{\delta}{2Kt^2}.\end{aligned}$$

By a union bound over e_i, e_j, we have $\Pr[\overline{\mathcal{F}_t}] \le \frac{\delta}{2t^2}$.

Define event $\mathcal{F} := \bigcap_{t=1}^{\infty} \mathcal{F}_t$. Then, we have $\Pr[\mathcal{F}] \ge 1 - \sum_{t=1}^{\infty} \Pr[\overline{\mathcal{F}_t}] \ge 1 - \sum_{t=1}^{\infty} \frac{\delta}{2t^2} \ge 1 - \delta$.

Below we prove that for any $e_i \notin M_*^C$, let e_j be the edge in M_*^C at the same position as e_i, *i.e.*, $e_j \in M_*^C, s(e_i) = s(e_j)$, and then conditioning on $\mathcal{E} \cap \mathcal{F}$, when $c_t(e_i, e_j) < \frac{1}{2}\tilde{\Delta}^C_{e_i}$, the duel (e_i, e_j) will not be pulled.

Suppose that, $\mathcal{E} \cap \mathcal{F}$ occur, and at some timestep t, $c_t(e_i, e_j) < \frac{1}{2}\tilde{\Delta}^C_{e_i}$ and CAR-Verify pulls the duel (e_i, e_j), *i.e.*, $(e_t, f_t) = (e_i, e_j)$. Then, from the occurences of $\mathcal{E} \cap \mathcal{F}$ and the definition of $\tilde{\Delta}^C_{e_i}$, we have

$$\begin{aligned}c_t(e_i, e_j) <& \frac{1}{2} \cdot \min_{M \in \mathcal{M} \setminus \{M_*^C\} : e_i \in M} \left\{ \frac{\ell}{d_{M_*^C, M}} \cdot \left(\frac{1}{2} - \frac{1}{\ell} \chi_M^T P \chi_{M_*^C} \right) \right\}\\ \le& \frac{\ell}{2 d_{M_*^C, M_t}} \cdot \left(\frac{1}{2} - \frac{1}{\ell} \chi_{M_t}^T P \chi_{M_*^C} \right).\end{aligned}$$

According to the selection of (e_t, f_t) in CAR-Verify, we have that for any e, e' $s.t.$ $e \in M_t \setminus M_*^C, e' \in M_*^C \setminus M_t, s(e) = s(e')$,

$$\begin{aligned}c_t(e, e') \le& c_t(e_i, e_j)\\ <& \frac{\ell}{2 d_{M_*^C, M_t}} \cdot \left(\frac{1}{2} - \frac{1}{\ell} \chi_{M_t}^T P \chi_{M_*^C} \right).\end{aligned}$$

Thus, we have

$$\begin{aligned}f(M_t, M_*^C, \bar{P}_t) <& f(M_t, M_*^C, P_t) + \frac{2}{\ell} \sum_{\substack{e \in M_t \setminus M_*^C, e' \in M_*^C \setminus M_t\\ s(e) = s(e')}} c_t(e', e)\\ <& \frac{1}{\ell} \chi_{M_t}^T P \chi_{M_*^C} + \frac{2}{\ell} \cdot d_{M_*^C, M_t} \cdot \frac{\ell}{2 d_{M_*^C, M_t}} \cdot \left(\frac{1}{2} - \frac{1}{\ell} \chi_{M_t}^T P \chi_{M_*^C} \right)\\ =& \frac{1}{2},\end{aligned}$$

which contradicts the return condition of CAR-Cond.

Thus, conditioning on $\mathcal{E} \cap \mathcal{F}$, when $c_t(e_i, e_j) < \frac{1}{2}\tilde{\Delta}^C_{e_i}$, the duel (e_i, e_j) will not be pulled. Let T_{cond} and T_{ver} denote the number of samples incurred by the sub-procedure CAR-Cond(δ_0) and the verification part (from Line 4 to end), respectively. Then, using the similar analysis as the proof of Theorem 1, we have that for any e, e' $s.t.$ $e \notin M_*^C, e' \in M_*^C, s(e) = s(e')$

$$T(e, e') \le \frac{1}{(\tilde{\Delta}^C_e)^2} \ln \left(\frac{4KT^3}{\delta} \right) + 1$$

Note that fixing $e \notin M_*^C$, e' is the edge in M_*^C at the same position as e, *i.e.*, $e' \in M_*^C, s(e) = s(e')$. Thus, taking summation over $e \notin M_*^C$, we have

$$T_{\text{ver}} \le H^C_{\text{ver}} \ln \left(\frac{4KT^3}{\delta} \right) + 1$$

Thus, we can obtain $T_{\text{ver}} = O(H^C_{\text{ver}} \ln(\frac{H^C_{\text{ver}}}{\delta}))$. Then, from Theorem 4, we have that conditioning on $\mathcal{E} \cap \mathcal{F}$,

$$T = T_{\text{cond}} + T_{\text{ver}}$$

$$=O\left(\sum_{j=1}^{\ell}\sum_{\substack{e\neq e'\\ e,e'\in E_j}}\frac{1}{(\Delta^C_{e,e'})^2}\ln\left(\frac{K}{\delta_0(\Delta^C_{e,e'})^2}\right)\right)+O\left(H^C_{\text{ver}}\ln\left(\frac{H^C_{\text{ver}}}{\delta}\right)\right)$$

$$=O\left(\sum_{j=1}^{\ell}\sum_{\substack{e\neq e'\\ e,e'\in E_j}}\frac{1}{(\Delta^C_{e,e'})^2}\ln\left(\frac{K}{(\Delta^C_{e,e'})^2}\right)+H^C_{\text{ver}}\ln\left(\frac{H^C_{\text{ver}}}{\delta}\right)\right),$$

which completes the proof of Lemma 10. □

Lemma 11 (CAR-Verify-correctness)**.** *Assume the existence of Condorcet winner. Then, with probability at least $1-\delta$, the* CAR-Verify *algorithm (Algorithm 4) will return the Condorcet winner or an error.*

Proof. Recall that $\Pr[\mathcal{F}]\geq 1-\delta$.

Then, conditioning on $\mathcal{F}$, if CAR-Verify terminates with an error, Lemma 11 holds. If CAR-Verify terminates with an answer $\mathsf{Out}=M_t$, we have $f(M,\hat{M},P_t)<f(M,\hat{M},\bar{P}_t)\leq \max_{M\in\mathcal{M}\setminus\{\hat{M}\}}f(M,\hat{M},\bar{P}_t)\leq\frac{1}{2}$ for any $M\in\mathcal{M}\setminus\{\hat{M}\}$, and thus the answer $\mathsf{Out}=M_t=M^C_*$.

Note that conditioning on $\mathcal{F}$, CAR-Verify must terminate. This is because if $\mathcal{F}\cap\mathcal{E}$ occur, according to Lemma 10, CAR-Verify will terminate and return the Condorcet winner with a bounded samples. Otherwise, if $\mathcal{F}\cap\bar{\mathcal{E}}$ occur, we have that $M^C_*\in\mathcal{M}\setminus\{\hat{M}\}$ and $f(M^C_*,\hat{M},P_t)>\frac{1}{2}$. Then, the condition of returning an answer cannot be satisfied and the condition of returning an error will be satisfied with limit timesteps because the confidence radius shrinks as the timestep increases.

Therefore, we complete the proof of Lemma 11. □

Now, we present the expected sample complexity for the CAR-Parallel algorithm.

Theorem 5 (CAR-Parallel)**.** *Assume the existence of Condorcet winner. Then, given $\delta<0.01$, with probability at least $1-\delta$, the* CAR-Parallel *algorithm (Algorithm 3) will return the Condorcet winner with an expected sample complexity*

$$O\left(\sum_{j=1}^{\ell}\sum_{\substack{e\neq e'\\ e,e'\in E_j}}\frac{\ln\left(K/(\Delta^C_{e,e'})^2\right)}{(\Delta^C_{e,e'})^2}+H^C_{\text{ver}}\ln\left(\frac{H^C_{\text{ver}}}{\delta}\right)\right).$$

Proof. Since CAR-Parallel directly applies the "parallel simulation" technique (Chen & Li, 2015; Chen et al., 2017) on CAR-Verify to boost the confidence, Theorem 5 follows from Lemma 10, 11 and Lemma 4.8 (result for parallel simulation) in (Chen et al., 2017). □

C.4. Lower Bound

To formally state our result for lower bound, we first introduce the following notions. For any $\delta\in(0,1)$, we call an algorithm $\mathbb{A}$ a δ-correct algorithm if, for any problem instance of CPE-DB with Condorcet winner, algorithm $\mathbb{A}$ identifies the Condorcet winner with probability at least $1-\delta$. In addition, for any $M\in\mathcal{M}\setminus\{M^C_*\}$, we use $\mathcal{O}(M)$ to denote the set of matchings that can beat M, *i.e.*, $\mathcal{O}(M)=\{M_x\in\mathcal{M}\setminus\{M\}:f(M_x,M,P)\}\geq\frac{1}{2}$. According to the definition of Condorcet winner, $M^C_*\in\mathcal{O}(M)$ for any $M\in\mathcal{M}\setminus\{M^C_*\}$.

In the following, we present a lower bound for the problem of combinatorial pure exploration for identifying the Condorcet winner in a special case.

Theorem 6 (Condorcet lower bound)**.** *Consider the problem of combinatorial pure exploration for identifying the Condorcet winner. Suppose that, for any $M\in\mathcal{M}\setminus\{M^C_*\}$, for any $M_x\in\mathcal{O}(M)$, $f(M,M^C_*,P)\leq f(M,M_x,P)$ and $M^C_*\setminus M\subseteq$*

$M_x \setminus M$. For some constant $0 < \gamma < \frac{1}{2(2+\ell)}$, for any $e_i, e_j \in E, s(e_i) = s(e_j)$, $\frac{1}{2} - \gamma \le p_{e_i,e_j} \le \frac{1}{2} + \gamma$. Then, for any $\delta \in (0, 0.1)$, any δ-correct algorithm has sample complexity

$$\Omega\left(\sum_{e \notin M_*^C} \frac{1}{\ell^2 \cdot (\Delta_e^C)^2} \ln\left(\frac{1}{\delta}\right) \right).$$

Proof. Fix an instance $\mathcal{I}$ of the Condorcet CPE-DB problem under the supposition and a δ-correct algorithm $\mathbb{A}$. In instance $\mathcal{I}$, M_*^C is the Condorcet winner and M is a suboptimal matching. Let T_{e_i,e_j} be the expected number of samples drawn from the duel (e_i, e_j) when $\mathbb{A}$ runs on instance $\mathcal{I}$.

We consider the following alternative instance $\mathcal{I}'$. For the duel (e_i, e_j) such that $e_i \in M \setminus M_*^C, e_j \in M_*^C \setminus M, s(e_i) = s(e_j)$, we change the Bernoulli distribution of duel (e_i, e_j) as follows:

$$p'_{e_i,e_j} = p_{e_i,e_j} + \ell \cdot \left(\frac{1}{2} - f(M, M_*^C, P) + \lambda \right)$$

Then, $f'(M, M_*^C, P) > \frac{1}{2}$. For any $M_x \in \mathcal{O}(M)$, since $f(M, M_*^C, P) < f(M, M_x, P)$ and $e_j \in M_x \setminus M$, we have $f'(M, M_x, P) > f(M, M_x, P) + (\frac{1}{2} - f(M, M_*^C, P)) \ge \frac{1}{2}$. Thus, we can see that in instance $\mathcal{I}'$, M is the Condorcet winner instead.

Using Lemma 1 in (Kaufmann et al., 2016), we can obtain

$$T_{e_i,e_j} \cdot d(p_{e_i,e_j}, p'_{e_i,e_j}) \ge d(1 - \delta, \delta).$$

For $\delta \in (0, 0.1)$, we have $d(1-\delta, \delta) \ge 0.4 \ln(\frac{1}{\delta})$. From the supposition, for some constant $0 < \gamma < \frac{1}{2(2+\ell)}$, for any $e_i, e_j \in E, s(e_i) = s(e_j)$, $\frac{1}{2} - \gamma \le p_{e_i,e_j} \le \frac{1}{2} + \gamma$. Then, for any $M_1, M_2 \in \mathcal{M}$ *s.t.* $M_1 \ne M_2$, $\frac{1}{2} - \gamma \le f(M_1, M_2, P) \le \frac{1}{2} + \gamma$. Thus, for the changed duel (e_i, e_j), $\gamma \le p'_{e_i,e_j} \le 1 - \gamma$ and $d(p_{e_i,e_j}, p'_{e_i,e_j}) \le \frac{(p_{e_i,e_j} - p'_{e_i,e_j})^2}{p'_{e_i,e_j}(1 - p'_{e_i,e_j})} \le \frac{1}{\gamma(1-\gamma)} (p_{e_i,e_j} - p'_{e_i,e_j})^2$.

Therefore,

$$\frac{1}{\gamma(1-\gamma)} \cdot T_{e_i,e_j} \cdot (p_{e_i,e_j} - p'_{e_i,e_j})^2 \ge 0.4 \ln\left(\frac{1}{\delta}\right)$$

$$\frac{1}{\gamma(1-\gamma)} \cdot T_{e_i,e_j} \cdot \ell^2 \cdot \left(\frac{1}{2} - f(M, M_*^C, P) + \lambda \right)^2 \ge 0.4 \ln\left(\frac{1}{\delta}\right)$$

$$T_{e_i,e_j} \ge \frac{0.4\gamma(1-\gamma)}{\ell^2 \cdot \left(\frac{1}{2} - f(M, M_*^C, P) + \lambda\right)^2} \ln\left(\frac{1}{\delta}\right)$$

We can perform the similar distribution changes on any duel (e_i, e_j) such that $e_i \in M \setminus M_*^C, e_j \in M_*^C \setminus M, s(e_i) = s(e_j)$ and any $M \in \mathcal{M} \setminus \{M_*^C\}$. In addition, the inequality holds for any $\lambda > 0$. Therefore, from the above analysis and the definition of $\Delta_{e_i}^C$ (Definition 5), we can obtain that for any $e_i, e_j \in E$ such that $e_j \in M_*^C, e_i \notin M_*^C, s(e_i) = s(e_j)$,

$$\begin{aligned} T_{e_i,e_j} \ge & \max_{M \in \mathcal{M} \setminus \{M_*^C\} : e_i \in M} \left\{ \frac{0.4\gamma(1-\gamma)}{\ell^2 \cdot \left(\frac{1}{2} - f(M, M_*^C, P)\right)^2} \ln\left(\frac{1}{\delta}\right) \right\} \\ \ge & \frac{0.4\gamma(1-\gamma)}{\ell^2 \cdot (\Delta_{e_i}^C)^2} \ln\left(\frac{1}{\delta}\right) \end{aligned}$$

Thus, we can see that for any edge $e \notin M_*^C$, the number of samples for the duel between e and the edge in M_*^C at the same position as e, which we denote by T_e, satisfies

$$T_e \ge \frac{0.4\gamma(1-\gamma)}{\ell^2 \cdot (\Delta_e^C)^2} \ln(\frac{1}{\delta}).$$

Summing over $e \notin M_*^C$, we have

$$\begin{aligned} T \geq & \sum_{e \notin M_*^C} \frac{0.4\gamma(1-\gamma)}{\ell^2 \cdot (\Delta_e^C)^2} \ln\left(\frac{1}{\delta}\right) \\ =& \Omega\left(\sum_{e \notin M_*^C} \frac{1}{\ell^2 \cdot (\Delta_e^C)^2} \ln\left(\frac{1}{\delta}\right)\right), \end{aligned}$$

which completes the proof of Theorem 6. □

Note that in the sample complexity upper bound of CAR-Parallel (Theorem 5), for any $e \notin M_*^C$, the verification gap $\tilde{\Delta}_e^C \geq \bar{\Delta}_e^C$, and thus the verification hardness satisfies

$$H_{\text{ver}}^C \leq \sum_{e \notin M_*^C} \frac{1}{(\Delta_e^C)^2}$$

Thus, given confidence $\delta < 0.01$, the term $H_{\text{ver}}^C \ln\left(\frac{H_{\text{ver}}^C}{\delta}\right)$ in the sample complexity upper bound of Algorithm 3 matches the lower bound within a factor of ℓ^2.