Combinatorial Pure Exploration with Continuous and Separable Reward Functions and Its Applications*

Weiran Huang1,†, Jungseul Ok2, Liang Li3 and Wei Chen4,†
1 Tsinghua University, Beijing, China
2 KTH, Stockholm, Sweden
3 Ant Financial Group, Hangzhou, China
4 Microsoft Research, Beijing, China
huang.inbox@outlook.com, jungseul@kth.se, liangli.ll@antfin.com, weic@microsoft.com

Abstract

We study the Combinatorial Pure Exploration problem with Continuous and Separable reward functions (CPE-CS) in the stochastic multi-armed bandit setting. In a CPE-CS instance, we are given several stochastic arms with unknown distributions, as well as a collection of possible decisions. Each decision has a reward according to the distributions of arms. The goal is to identify the decision with the maximum reward, using as few arm samples as possible. The problem generalizes the combinatorial pure exploration problem with linear rewards, which has attracted significant attention in recent years. In this paper, we propose an adaptive learning algorithm for the CPE-CS problem, and analyze its sample complexity. In particular, we introduce a new hardness measure called the consistent optimality hardness, and give both the upper and lower bounds of sample complexity. Moreover, we give examples to demonstrate that our solution has the capacity to deal with non-linear reward functions.

1 Introduction

The stochastic multi-armed bandit model is a predominant model for characterizing the trade-off between exploration and exploitation in a variety of application fields with stochastic environments. In this model, we are given a set of stochastic arms associated with unknown distributions. Upon each play of an arm, the player can get a reward sampled from the corresponding distribution. The most well studied objective is to maximize the cumulative reward, or minimize the cumulative regret, e.g., [Lai and Robbins, 1985; Auer et al., 2002b; Auer et al., 2002a; Bubeck et al., 2012]. Another popular objective is to identify the optimal arm with high probability by adaptively sampling arms based on the feedback collected. This is called the pure exploration version of the multi-armed bandit problem [Bubeck et al., 2011; Audibert and Bubeck, 2010; Gabillon et al., 2012].

* Corresponding authors.
† Due to space constraints, supplementary materials and proofs are moved into the extended version [Huang et al., 2018].
process. In this case, the objective function is also non-linear. Furthermore, we show that the COCI algorithm also solves the CPE-L problem with the same sample complexity as the CLUCB algorithm proposed by Chen et al. [2014].

In summary, our contributions include: (a) studying the combinatorial pure exploration problem with continuous and separable functions and proposing the COCI algorithm as its solution, (b) analyzing the sample complexity of COCI and providing both its lower and upper bounds with a novel hardness measure, and (c) applying the CPE-CS framework to water resources planning and partitioned opinion sampling with non-linear reward functions to demonstrate the usefulness of the CPE-CS framework and the COCI algorithm.

Related Work. Pure exploration bandit studies adaptive learning methods to identify the optimal solution. Best arm identification [Bubeck et al., 2011; Audibert and Bubeck, 2010; Gabillon et al., 2012], top- \( k \) arm identification [Kalyanakrishnan and Stone, 2010; Kalyanakrishnan et al., 2012; Bubeck et al., 2013; Kaufmann and Kalyanakrishnan, 2013; Zhou et al., 2014], the multi-bandit best arm identification [Gabillon et al., 2011] have been studied in the literature. Chen et al.; Chen et al. [2014; 2016a] generalize these studies to Combinatorial Pure Exploration with Linear reward functions (CPE-L). Soare et al. [2014] also study the linear reward functions, but the player is required to select a decision to play instead of a single arm to sample in each round. A very recent paper [Chen et al., 2017] studies the CPE problems beyond linear reward functions, but their model assumes arms with Gaussian distributions and only works with the mean estimator, while our CPE-CS only requires bounded distributions and also works for variance estimators. Moreover, for efficient implementations, they need a pseudo-polynomial algorithm for the exact query besides the maximization oracle, but our solution only needs a maximization oracle.

A related online learning problem is multi-armed bandit (MAB), e.g., [Lai and Robbins, 1985; Auer et al., 2002b; Auer et al., 2002a; Bubeck et al., 2012]. The goal of MAB is to maximize cumulative rewards over multiple rounds, and the key is to balance exploration and exploitation during the learning process. In contrast, in pure exploration, the key is the adaptive exploration in the learning process to quickly find the optimal solution, and thus it is fundamentally different from MAB [Bubeck et al., 2011]. Combinatorial MAB is a popular topic in recent years [Cesa-Bianchi and Lugosi, 2012; Gai et al., 2012; Chen et al., 2016c; Chen et al., 2016b; Gopalan et al., 2014; Kveton et al., 2014; Kveton et al., 2015; Combes et al., 2015], but their goals and techniques are very different from ours.

2 Problem Definition

An instance of combinatorial pure exploration bandit problems consists of (a) a set of \( m \) arms \( [m] = \{1, \ldots, m\} \), each arm \( i \) being associated with an unknown distribution \( D_i \) with range \([0,1]\) and a key unknown parameter \( \theta^*_i \in [0,1] \) of \( D_i \), (b) a finite set of decisions \( \mathcal{Y} \subseteq \mathbb{R}^m \), with each decision \( y = (y_1, \ldots, y_m) \) as a vector, and (c) a real-valued (expected) reward function \( r(\theta; y) \) with vector \( \theta \) taken from the parameter space \([0,1]^m\) and \( y \in \mathcal{Y} \). In each round \( t = 1, 2, \ldots \), a player selects one arm \( i \in [m] \) to play, and observes a sample independently drawn from \( D_i \) as the feedback. The player needs to decide based on the observed feedback so far if she wants to continue to play arms. If so, she needs to decide which arm to play next; if not, she needs to output a decision \( y^* \in \mathcal{Y} \) such that with high probability \( y^* \) is the optimal decision maximizing the reward \( r(\theta^*; y^*) \), where \( \theta^* = (\theta^*_1, \ldots, \theta^*_m) \) is the vector of the true underlying parameters of the unknown distributions \( D = (D_1, \ldots, D_m) \).

**Definition 1.** Given a combinatorial pure exploration instance \( ([m], \mathcal{Y}, r(\cdot; \cdot), D, \theta^*) \) and a confidence error bound \( \delta \), the combinatorial pure exploration (CPE) problem requires the design of an algorithm with the following components: (a) a stopping condition, which decides whether the algorithm should stop in the current round, (b) an arm selection component, which selects the arm to play in the current round when the stopping condition is false, and (c) an output component, which outputs the decision \( y^* \) when the stopping condition is true. The algorithm could only use \( ([m], \mathcal{Y}, r(\cdot; \cdot)) \) and the feedback from previous rounds as inputs, and should guarantee that with probability at least \( 1 - \delta \) the output \( y^* \) is an optimal decision, i.e., \( y^* \in \arg \max_{y \in \mathcal{Y}} r(\theta^*; y) \).

A standard assumption for CPE problems is that the optimal decision under the true parameter vector \( \theta^* \) is unique, i.e., \( y^* = \arg \max_{y \in \mathcal{Y}} r(\theta^*; y) \). The performance of a CPE algorithm is measured by its sample complexity, which is the number of rounds taken when the algorithm guarantees its output to be the optimal one with probability at least \( 1 - \delta \).

We say that a reward function \( r(\theta; y) \) is continuous if \( r(\theta; y) \) is continuous in \( \theta \) for every \( y \in \mathcal{Y} \), and (additively) separable if there exist functions \( r_1, \ldots, r_m \) such that \( r(\theta; y) = \sum_{i=1}^m r_i(\theta_i, y_i) \). We use CPE-CS to denote the class of CPE problems with Continuous and Separable reward functions and each parameter \( \theta^*_i \) of arm \( i \) can either be mean \( \mathbb{E}_{X \sim D_i}[X] \) or variance \( \text{Var}_{X \sim D_i}[X] \).

We use \( \text{EST}_{i}(X_{i,1}, X_{i,2}, \ldots, X_{i,s}) \) to denote the unbiased estimator for parameter \( \theta^*_i \) from \( s \) i.i.d. observations \( X_{i,1}, X_{i,2}, \ldots, X_{i,s} \) of the \( i \)-th arm. In particular, for the mean estimator, \( \text{EST}_{i}(X_{i,1}, X_{i,2}, \ldots, X_{i,s}) = \frac{1}{s} \sum_{j=1}^s X_{i,j} \), and for the variance estimator, \( \text{EST}_{i}(X_{i,1}, X_{i,2}, \ldots, X_{i,s}) = \frac{1}{s-1} \left( \sum_{j=1}^s X_{i,j}^2 - \frac{1}{s} \left( \sum_{j=1}^s X_{i,j} \right)^2 \right) \). Notice that the variance estimator needs at least two samples. We also define \( \phi : [0,1]^m \rightarrow \mathcal{Y} \) to be a deterministic tie-breaking maximization oracle such that for any \( \theta \in [0,1]^m \), \( \phi(\theta) = (\phi_1(\theta), \ldots, \phi_m(\theta)) \in \arg \max_{y \in \mathcal{Y}} r(\theta; y) \) and it always outputs the same optimal solution, called the leading optimal solution, under the same parameter \( \theta \).

CPE-CS encompasses the important CPE problems with Linear reward functions (CPE-L). In CPE-L, parameter \( \theta^*_i \) of arm \( i \) is taken from the parametric family \([0,1]^m\), which can be represented as an \( m \)-dimensional binary vector. Thus, the decision space \( \mathcal{Y} \) is a subset of \([0,1]^m\), and each vector \( y = (y_1, \ldots, y_m) \in \mathcal{Y} \) represents a subset
Algorithm 1: COCI: Consistently Optimal Confidence Interval Algorithm for CPE-CS

Input: Confidence error bound \( \delta \in (0, 1) \), maximization oracle \( \phi \).

Output: Confidence error bound \( y^* = (y_1, y_2, \ldots, y_m) \in \mathcal{Y} \).

1. \( t \leftarrow T_m; \// \tau = 1 \) for the mean estimator and \( \tau = 2 \) for the variance estimator.

2. for \( i = 1, 2, \ldots, m \) do
   3. observe the \( i \)-th arm \( \tau \) times \( X_{i,1}, \ldots, X_{i,\tau} \).
   4. \( T_{i,t} \leftarrow \tau; \)
   5. estimate \( \hat{\theta}_{i,t} \leftarrow \text{Est}_i(X_{i,1}, \ldots, X_{i,T_{i,t}}); \)
   6. \( \text{rad}_{i,t} \leftarrow \sqrt{\frac{1}{2T_{i,t}} \ln \frac{4T_{i,t}}{4\tau}} \); // confidence radius
   7. \( \Theta_t \leftarrow \{ \theta \in [0, 1]^m \colon |\theta_i - \hat{\theta}_{i,t}| \leq \text{rad}_{i,t}, \forall i \in [m] \}; \)
   8. for \( t = \tau m + 1, \tau m + 2, \tau m + 3, \ldots \) do
      9. \( C_t \leftarrow \emptyset; \)
     10. for \( i = 1, 2, \ldots, m \) do
         11. if \( \max_{\theta \in \Theta_{t-1}} \phi_i(\theta) \neq \min_{\theta \in \Theta_{t-1}} \phi_i(\theta) \) then
             12. \( C_t \leftarrow C_t \cup \{i\}; \)
         13. if \( C_t = \emptyset \) then
             14. return \( y^* = \phi(\theta) \) for an arbitrary \( \theta \in \Theta_{t-1}; \)
             15. \( j \leftarrow \arg \max_{i \in C_t} \text{rad}_{i,T_{i,t-1}}; \)
             16. \( T_{j,t} \leftarrow T_{j,T_{j,t-1}} + 1; T_{i,t} \leftarrow T_{i,t-1} \) for all \( i \neq j; \)
         17. play the \( j \)-th arm and observe the outcome \( X_{j,T_{j,t}}; \)
         18. update \( \hat{\theta}_{j,t} \leftarrow \text{Est}_j(X_{j,1}, \ldots, X_{j,T_{j,t}}); \)
         19. update \( \hat{\theta}_{i,t} \leftarrow \hat{\theta}_{i,t-1} \) for all \( i \neq j; \)
     20. update \( \text{rad}_{i,t} \leftarrow \sqrt{\frac{1}{2T_{i,t}} \ln \frac{4T_{i,t}}{4\tau}} \) for all \( i \in [m]; \)
     21. \( \Theta_t \leftarrow \{ \theta \in [0, 1]^m \colon |\theta_i - \hat{\theta}_{i,t}| \leq \text{rad}_{i,t}, \forall i \in [m] \}; \)

of arms \( S_y = \{ i \in [m] : y_i = 1 \} \). Moreover, the reward function \( r(\theta; y) = \sum_{i=1}^{m} \theta_i \cdot y_i \) is continuous and separable.

3 Solving CPE-CS

In this section, we propose the Consistently Optimal Confidence Interval (COCI) Algorithm for CPE-CS, and analyze its sample complexity. En route to our sample complexity bound, we introduce a new concept of arm-level consistently optimal radius \( \Lambda_i \) of each arm \( i \), which leads to a new hardness measure \( \Lambda \). We first introduce the components and notations which will be used in the algorithm.

The algorithm we propose for CPCI (Algorithm 1) is based on the confidence intervals of the parameter estimates. The algorithm maintains the confidence interval space \( \Theta_t \) for every round \( t \) to guarantee that the true parameter \( \theta^* \) is always in \( \Theta_t \) for all \( t > 0 \) with probability at least \( 1 - \delta \). After the initialization (lines 1–7), in each round \( t \), the algorithm first computes the candidate set \( C_t \subseteq [m] \) (lines 9–12). According to the key condition in line 11, \( C_t \) contains the \( i \)-th arm if \( \max_{\theta \in \Theta_{t-1}} \phi_i(\theta) \neq \min_{\theta \in \Theta_{t-1}} \phi_i(\theta) \) (this is a logical condition, and its actual implementation will be discussed in Section 3.1). The stopping condition is \( C_t = \emptyset \) (line 13), which means that within the confidence interval space, all leading optimal solutions are the same. In this case, the algorithm returns the leading optimal solution under any \( \theta \in \Theta_{t-1} \) as the final output (line 14). Notice that if the true parameter \( \theta^* \) is in \( \Theta_{t-1} \), then the output is the true optimal solution \( y^* = \phi(\theta^*) = y^* \). If \( C_t \neq \emptyset \), then the algorithm picks any arm \( j \) with the largest confidence radius (line 15), plays this arm, observes its feedback, and updates its estimate \( \hat{\theta}_{j,t} \) and confidence radius \( \text{rad}_{j,t} \) accordingly (lines 16–21). Intuitively, arm \( j \) is the most uncertain arm causing inconsistency, thus the algorithm picks it to play first. Since the key stopping condition is that the leading optimal solutions for all \( \theta \in \Theta_{t-1} \) are consistently optimal, we call our algorithm Consistently Optimal Confidence Interval (COCI) algorithm.

Before analyzing the sample complexity of the COCI algorithm, we first introduce the (arm-level) consistent optimality radius for every arm \( i \), which is formally defined below.

Definition 2. For all \( i \in [m] \), the consistent optimality radius \( \Lambda_i \) for arm \( i \) is defined as:

\[
\Lambda_i = \inf_{\theta : \phi_i(\theta) \neq \phi_i(\theta^*)} \|\theta - \theta^*\|_\infty.
\]

Intuitively, \( \Lambda_i \) measures how far \( \theta \) can be away from \( \theta^* \) (in infinity norm) while the leading optimal solution under \( \theta \) is still consistent with the true optimal one in the i-th dimension, as precisely stated below.

Proposition 1. \( \forall i \in [m] \), if \( |\theta_j - \theta_{j'}| < \Lambda_i \) holds for all \( j \in [m] \), then \( \phi_i(\theta) = \phi_i(\theta^*) \).

The following lemma shows that the consistent optimality radii are all positive, provided by that the reward function is continuous and the true optimal decision \( y^* \) is unique.

Lemma 1. If the reward function \( r(\theta; y) \) is continuous on \( \theta \) for every \( y \in \mathcal{Y} \), and the optimal decision \( y^* \) under the true parameter vector \( \theta^* \) is unique, then \( \Lambda_i \) is positive for every \( i \in [m] \).

Given that the consistent optimality radii are all positive, we can introduce the key hardness measure used in the sample complexity analysis. We define consistent optimality hardness as \( \Lambda = \sum_{i=1}^{m} \frac{1}{\Lambda_i^2} \). The following theorem shows our primary sample complexity result for the COCI algorithm.

Theorem 1. With probability at least \( 1 - \delta \), the COCI algorithm (Algorithm 1) returns the unique true optimal solution \( y^* = y^* \), and the number of rounds (or samples) \( T \) satisfies

\[
T \leq 2m + 12\Lambda \ln 24\Lambda + 4\Lambda \ln \frac{4}{\tau \delta} = O \left( \Lambda \log \frac{\Lambda}{\delta} \right).
\]

Theorem 1 shows that the sample complexity is positively related to the consistent optimality hardness, or inversely proportional to the square of consistent optimality radius \( \Lambda_i^2 \). Intuitively, when \( \Lambda_i \) is small, we need more samples to make the optimal solutions in the confidence interval consistent on the i-th dimension, and hence higher sample complexity.

We remark that if we do not compute the candidate set \( C_t \) and directly pick the arm with the largest radius among all arms in line 15, every arm will be selected in a round-robin
fashion and COCI becomes a uniform sampling algorithm. In the extended version, we show that the sample complexity upper bound of the uniform version is obtained by replacing $H_\Lambda$ in Eq. (1) by $H_\Lambda^U = \frac{m}{\min_{i\in[m]} \Lambda_i}$, and the factor $H_\Lambda^U$ is tight for the uniform sampling. This indicates that the adaptive sampling method of COCI would perform much better than the uniform sampling when arms have heterogeneous consistent optimality radii such that $H_\Lambda \ll H_\Lambda^U$.

Due to the space constraint, we only provide the key lemma below leading to the proof of the theorem. We define a random event $\xi = \{\forall t \geq \tau m, \forall i \in [m], \theta_{i,t}^* - \theta_{i,t-1} \leq \rad_{i,t}\}$, which indicates that $\theta^*$ is inside the confidence interval space of all the rounds. Then we have the following lemma.

**Lemma 2.** Suppose event $\xi$ occurs. For every $i \in [m]$ and every $t > \tau m$, if $\rad_{i,t-1} < \Lambda_i/2$, then the $i$-th arm will not be played in round $t$.

**Proof.** Suppose, for a contradiction, that the $i$-th arm is played in round $t$, namely, $i \in C_t$, and $i = \arg\max_{j \in C_t} \rad_{j,t-1}$. Thus for each $j \in C_t$, we have $\rad_{j,t-1} \leq \rad_{i,t-1} < \Lambda_i/2$.

We claim that for all $\theta \in \Theta_{t-1}$, $\phi_i(\theta) = \phi_i(\theta^*)$. If so, $\max_{\theta \in \Theta_{t-1}} \phi_i(\theta) = \min_{\theta \in \Theta_i} \phi_i(\theta)$, then by line 11 $i \notin C_t$, a contradiction.

We now prove the claim. For any vector $x \in \mathbb{R}^m$ and any index subset $C \subseteq [m]$, we use $x_C$ to denote the subvector of $x$ projected onto $C$. For vector-valued functions such as $\phi(\theta)$, we use $\phi_C(\theta)$ for $\phi(\theta)|_C$. For any $\theta \in \Theta_{t-1}$, we construct an intermediate vector $\theta' = (\theta_{C_t}, \theta_{\bar{C}_t})$, i.e., the $j$-th component $\theta'_j$ is $\theta_j$ when $j \in C_t$, or $\theta^*_j$ when $j \notin C_t$.

Since event $\xi$ occurs, we have $|\theta'_{j,t-1} - \theta^*_j| \leq \rad_{j,t-1}$ for $j \in [m]$. Thus for all $j \in C_t$, $|\theta'_{j,t-1} - \theta^*_j| \leq \rad_{j,t-1} < \Lambda_i$, and for all $j \notin C_t$, $|\theta'_{j,t-1} - \theta^*_j| = 0$. This means that $\|\theta' - \theta^*\|_{\infty} < \Lambda_i$. According to Proposition 1, $\phi_i(\theta') = \phi_i(\theta^*)$. We next prove that $\phi_i(\theta') = \phi_i(\theta^*)$, which directly leads to $\phi_i(\theta) = \phi_i(\theta^*)$.

Since event $\xi$ occurs and $\theta^* \in [0,1]^m$, $\theta^*$ is in $\Theta_{t-1}$.

By the definition of $\theta'$ and $\theta \in \Theta_{t-1}$, $\theta'$ is also in $\Theta_{t-1}$.

According to Algorithm 1, for each $j \notin C_t$, we have $\max_{\theta \in \Theta_{t-1}} \phi_i(\theta) = \min_{\theta \in \Theta_i} \phi_i(\theta)$, thus $\phi_{-C_t}(\theta') = \phi_{-C_t}(\theta^*)$.

Note that the reward function is separable, we have

$$r(\theta; y) = \sum_{j \in C_t} r_j(\theta_j; y_j) + \sum_{j \notin C_t} r_j(\theta_j; y_j).$$

Let $Y_{C_t}(\theta) = \{y_C : y \in Y \land y_{-C_t} = \phi_{-C_t}(\theta)\}$. It is straightforward to verify that $\phi_{C_t}(\theta)$ is the leading optimal solution for the following problem:

$$\max_{\theta \in \Theta_i} \sum_{j \in C_t} r_j(\theta_j; z_j),$$

subject to $z \in Y_{C_t}(\theta)$.

Similarly, we have

$$r(\theta'; y) = \sum_{j \in C_t} r_j(\theta_j; y_j) + \sum_{j \notin C_t} r_j(\theta_j; y_j),$$

and $\phi_{C_t}(\theta')$ is the leading optimal solution for

$$\max_{\theta \in \Theta_i} \sum_{j \in C_t} r_j(\theta_j; z_j),$$

subject to $z \in Y_{C_t}(\theta')$.

Since $\phi_{-C_t}(\theta) = \phi_{-C_t}(\theta^*)$, optimization problems (2) and (3) are identical, thus they have the same leading solution $\phi_{C_t}(\theta) = \phi_{C_t}(\theta^*)$. Notice that $i \in C_t$, therefore, $\phi_i(\theta) = \phi_i(\theta^*)$ holds. $\square$

The above lemma is the key connecting consistent optimality radius $\Lambda_i$ with confidence radius $\rad_{i,t-1}$ and the stopping condition. Its proof relies on both the definition of consistent optimality radius and the assumption of separable reward functions. With this lemma, the sample complexity can be obtained by considering the first round when every arm satisfies the condition $\rad_{i,t-1} < \Lambda_i/2$.

Borrowing a lower bound analysis in [Chen et al., 2017], we can further show that the hardness measure $H_\Lambda$ is necessary for CPE-CS, even CPE-L, as shown below.

**Theorem 2.** Given $m$ arms and $\delta \in (0, 0.1)$, there exists an instance such that every algorithm $A$ for CPE-L which outputs the optimal solution with probability at least $1-\delta$ takes at least

$$\Omega(H_\Lambda + H_\Lambda m^{-1} \log 1/\delta)$$

samples in expectation.

### 3.1 Implementing the Condition in Line 11

The key condition in line 11 of Algorithm 1 is a logical one revealing the conceptual meaning of the stopping condition, but it does not lead to a direct implementation. In many CPE-CS instances, the condition can be translated to a condition only on the boundary of $\Theta_{t-1}$, and further due to the bi-monotonicity of $\phi$ introduced below, it has an efficient implementation. Such instances include best-arm identification, top-$k$ arm identification, water resources planning (Section 4.1), partitioned opinion sampling (Section 4.2), etc.

We say that the leading optimal solution $\phi(\theta)$ satisfies bi-monotonicity, if for each $i \in [m]$, $\phi_i(\theta)$ is monotonically non-increasing (or non-decreasing) in $\theta_i$, and monotonically non-decreasing (or non-increasing) in $\theta_j$ for all $j \neq i$. For convenience, we use $\theta_{-i,t} = \max_{\theta \in \Theta} \theta_i$ and $\theta_{i,t} = \min_{\theta \in \Theta} \theta_i$ to denote the upper and lower confidence bound of arm $i$ in round $t$. We also use $\theta_{-i,t}$ and $\theta_{-i,t}$ to denote the upper and lower confidence bounds of all arms excluding arm $i$.

**Theorem 3.** If the leading optimal solution $\phi(\theta)$ satisfies bi-monotonicity, the condition in line 11 of Algorithm 1 can be efficiently implemented by

$$\phi_i(\theta_{-i,t-1}, \theta_{i,t-1}) \neq \phi_i(\theta_{-i,t-1}, \theta_{i,t-1}).$$

The above theorem indicates that, when bi-monotonicity holds for $\phi(\theta)$, we only need two calls to the offline oracle $\phi(\theta)$ to implement the condition in line 11, and thus the COCI algorithm has an efficient implementation in this case.
4 Applications

4.1 Water Resource Planning

Water resource systems benefit people to meet drinking water and sanitation needs, and also support and maintain resilient biodiverse ecosystems. In regional water resource planning, one needs to determine the Biological Oxygen Demand (BOD, a measure of pollution) to be removed from the water system at each source. Online learning techniques proposed in recent years make adaptive optimization for water resource planning possible.

Let $y_i$ be the pounds of BOD to be removed at source $i$. One general model (adapted from [Bradley et al., 1977]) to minimize total costs to the region to meet specified pollution standards can be expressed as:

$$\max \sum_{i=1}^{m} \theta_i^* y_i - \sum_{i=1}^{m} f_i(y_i),$$

subject to

$$\sum_{i=1}^{m} y_i \geq b, 0 \leq y_i \leq c_i, \forall i \in [m],$$

where $\theta_i^*$ is the quality response caused by removing one pound of BOD at source $i$, and $f_i(y_i)$ is the cost of removing $y_i$ pounds of BOD at source $i$. Each $y_i$ is constrained by $c_i$, the maximum pounds of BOD that can be removed at source $i$. Moreover, the total pounds of BOD to be removed are required to be larger than a certain threshold $b$.

The above model formulates the trade-off between the benefit and the cost of removing the pollutants. The cost function $f_i$ is usually known and non-linear, which may depend on the cost of oxidation, labor cost, facility cost, etc., while the quality response $\theta_i^*$ is unknown beforehand, and needs to be learned from tests at source $i$. In each test, the tester measures the quality response at a source $i$ and gets an observation of $\theta_i^*$, which can be regarded as a random variable $\theta_i$ derived from an unknown distribution with mean $\theta_i^*$. The goal is to do as few tests as possible to estimate the quality responses, and then give a final allocation $(y_1^*, \ldots, y_m^*)$ of BOD among sources as the plan to be implemented (e.g., building BOD removal facilities at the sources).

The above problem falls into the CPE-CS framework. The $i$-th source corresponds to the $i$-th arm. Each quality response at source $i$ is the unknown parameter $\theta_i^*$ associated with the arm $i$, and $\tau = 1$. Each allocation $(y_1, \ldots, y_m)$ satisfying the constraints corresponds to a decision. We discretize $(y_i)$'s so that the decision class $\mathcal{Y}$ is finite. The reward function is $r(\theta, y) = \sum_{i=1}^{m} \theta_i y_i - \sum_{i=1}^{m} f_i(y_i)$, which is continuous and separable. Suppose the offline problem of Eq. (4) when $\theta^*$ is known can be solved by a known oracle $\phi(\theta^*)$. Then, the COCI algorithm can be directly applied to the water resource planning problem. The following lemma gives a sufficient condition for the bi-monotonicity of $\phi$.

**Lemma 3.** When $(dy_i/dy_j)$'s are all monotonically increasing or decreasing, and the constraint $\sum_{i=1}^{m} y_i \geq b$ is tight at the leading optimal solution $\phi(\theta)$ for all $\theta$, then $\phi(\theta)$ satisfies bi-monotonicity.

By Theorem 3, when the offline oracle for the water resources planning problem satisfies bi-monotonicity, we can instantiate the condition in line 11 of Algorithm 1 as $\phi_i(\theta_{i:1-1}, 0) \neq \phi_i(\theta_{i:1-1}, -1)$.

Although this application is set up in the context of water resource planning, we can see that the formulation in Eq. (4) is general enough to model other applications, especially ones in the urban planning context. For example, for planning air quality control for a city, we need to target a number of air pollution emission sources (e.g., factories), and do adaptive testing at the sources to determine the optimal pollution removal target at each sources which maximizes the total utility of the planning. Other applications, such as crime control, may also be modeled similarly as instances of our CPE-CS framework and solved effectively by our COCI algorithm.

4.2 Partitioned Opinion Sampling

Public opinion dynamics has been well studied, and there are a number of opinion dynamic models proposed in the literature, such as the voter model [Clifford and Sudbury, 1973], and its variants [Yildiz et al., 2011; Li et al., 2015; Huang et al., 2017]. In these models, people’s opinions $f_1(t), f_2(t), \ldots, f_n(t) \in [0, 1]$ change over time $t$, and will converge to a steady state after sufficient social interactions in which the joint distribution of people’s opinions no longer changes. Thus, they are regarded as Bernoulli random variables derived from the steady-state joint distribution, and sampling at time $t$ can be considered as observing part of a realization of $f_1(t), f_2(t), \ldots, f_n(t)$. In partitioned opinion sampling, the population is divided into several disjoint groups $V_1, V_2, \ldots, V_m$ with $n_i = |V_i|$. When we draw $y_i$ samples (with replacement) from group $V_i$ at time $t$, we obtain $y_i$ i.i.d. random variables $f_{i:1}, f_{i:2}, \ldots, f_{i:n_i}$, where $v_{i,j}$ is the $j$-th sample from group $V_i$. Partitioned sampling uses $f_i(t) = \sum_{i=1}^{m} \frac{n_i}{m} \cdot \left( \frac{1}{y_i} \sum_{j=1}^{y_i} f_{i:j} \right)$ as the unbiased estimator for the mean population opinion at time $t$, and the task is to find the optimal allocation $(y_1^*, \ldots, y_m^*)$ with sample size budget $\sum_{i=1}^{m} y_i^* \leq k$ which minimizes the sample variance $\text{Var}[f_i(t)]$, a common sample quality measure [Bethe, 1986; Ballin and Barcaroli, 2013; Huang et al., 2017].

One way to achieve best estimate quality for a future time $t$ is to do adaptive sampling to quickly estimate the opinion variance of each group, and then decide the optimal sample size allocation for the real sample event at time $t$. This corresponds to certain opinion polling practices, for instance, polling after each presidential debates, and preparing for a better sample quality at the election day. We remark that in this setting, past samples are useful to estimate opinion variance within groups, but cannot be directly use to estimate the mean opinion at a future time $t$, since $f_i(t)$ is time-based and using historical samples directly may lead to biased estimates.

More specifically, let $X_i$ be the result of one random sample from group $V_i$ in the steady state. Note that the randomness of $X_i$ comes from both the sampling randomness and the opinion randomness in the steady state. One can easily verify that $\text{Var}[f_i(t)] = \sum_{i=1}^{m} \frac{n_i^2}{n_i y_i} \cdot \text{Var}[X_i]$, where $\text{Var}[X_i]$ is the variance of group $V_i$, and referred to as the within-group variance. The goal is to use as few samples as possible to estimate within-group variances, and then give the final sample
size allocation which minimizes \( \text{Var}[f(t)] \).

This falls into the CPE-CS framework. In particular, each group \( V_i \) corresponds to an arm \( i \), and each within-group variance \( \text{Var}[X_i] \) corresponds to the unknown parameter \( \theta_i^* \) of arm \( i \). The decision space \( \mathcal{Y} \) is \( \{ (y_1, \ldots, y_m) \in \mathbb{Z}_{+}^m : \sum_{i=1}^{m} y_i \leq k \} \). The reward function \( r(\theta; y) \) is set to be \( -\sum_{i=1}^{m} \frac{n_i^2}{n_i y_i} \) where the negative sign is because the partitioned opinion sampling problem is a minimization problem. It is non-linear but continuous and separable. Therefore, the problem is an instance of CPE-CS. The oracle for the offline problem can be achieved by a greedy algorithm, denoted as \( \phi(\theta) \), and it satisfies the bi-monotonicity (the design and the analysis of the offline oracle is non-trivial, see the extended version). Thus, the COCI algorithm can be directly applied as follows: 1) \( \text{EST}_i \) is set to be the variance estimator, i.e., \( \text{EST}_i(X_1, \ldots, X_s, i) = \frac{1}{s-1} (\sum_{j=1}^{s} X_i^2 - \frac{1}{s} (\sum_{j=1}^{s} X_{i,j})^2) \), and \( \tau = 2 \); 2) the condition in line 11 of Algorithm 1 is instantiated by \( \phi_i(\bar{\theta}_{-i,-t-1}, \bar{y}_{i,t-1}) \neq \phi_i(\bar{\theta}_{-i,-t-1}, \bar{y}_{i,t-1}) \).

5 Applying COCI to CPE-L

In Section 2, we already show that the linear class CPE-L is a special case of CPE-CS. In this section, we discuss the implication of applying COCI algorithm to solve CPE-L problems, and compare the sample complexity and implementation efficiency against the CLUCB algorithm in [Chen et al., 2014].

Since the parameter \( \theta^* \) is the vector of means of arms, we use the mean estimator and set \( \tau = 1 \) in COCI.

Recall that for a binary vector \( y \in \mathcal{Y}, S_y \) is defined as \( \{ i \in [m] : y_i = 1 \} \). Chen et al. [2014] use the term reward gap in the formulation of sample complexity. For each arm \( i \in [m] \), its reward gap \( \Delta_i \) is defined as:

\[
\Delta_i = \begin{cases} 
0 & \text{if } \theta^* \text{ is bi-monotone} \\
\max_{y \in \mathcal{Y}_i} \min_{y \in S_y} r(\theta^*; y) & \text{if } \theta^* \text{ is not bi-monotone}
\end{cases}
\]

Chen et al. [2014] also define a (reward gap) hardness measure \( H_{\Delta} = \sum_{i=1}^{m} \frac{1}{\Delta_i^2} \). Moreover, for each decision class \( \mathcal{Y} \), Chen et al. [2014] define a key quantity width, denoted as width(\( \mathcal{Y} \)), that is needed for sample complexity. Intuitively, width(\( \mathcal{Y} \)) denotes the minimum number of elements that one may need to exchange in one step of a series of steps when changing the current decision \( S \in \mathcal{Y} \) into another decision \( S' \in \mathcal{Y} \), and for every step of exchange in the series, the resulting decision (subset) should still be in \( \mathcal{Y} \). The technical definition is not very relevant with the discussion below, and thus is left in the supplementary material. We remark that width(\( \mathcal{Y} \)) = \( O(m) \).

Given the above setup, Chen et al. [2014] show that with probability \( 1 - \delta \), their CLUCB algorithm achieves sample complexity bound

\[
T \leq 2m + 499\text{width}(\mathcal{Y})^2 H_{\Delta} \ln(4\text{width}(\mathcal{Y})^2 H_{\Delta}/\delta) = O(\text{width}(\mathcal{Y})^2 H_{\Delta} \log(m H_{\Delta}/\delta)).
\]

When applying the COCI algorithm to solve CPE-L problems, we are able to obtain the following key connection between consistent optimality radius and the reward gap:

Lemma 4. For the CPE-L problems, we have \( \forall i \in [m], \Lambda_i \geq \Delta_i / \text{width}(\mathcal{Y}), \) and thus \( H_{\Lambda} \leq H_{\Delta} \cdot \text{width}(\mathcal{Y})^2 \).

Combining with Theorem 1, we have that COCI could achieve the following sample complexity bound for CPE-L:

\[
T \leq 2m + 12\text{width}(\mathcal{Y})^2 H_{\Delta} \ln(24\text{width}(\mathcal{Y})^2 H_{\Delta})
\]

\[
+ 4\text{width}(\mathcal{Y})^2 H_{\Delta} \ln(4\delta^{-1})
\]

\[
= O(\text{width}(\mathcal{Y})^2 H_{\Delta} \log(m H_{\Delta}/\delta)).
\]

The above result has the same sample complexity\(^2\) as in Eq. (5) (with even a slightly better constant). However, with our analysis, we only need the complicated combinatorial quantity width(\( \mathcal{Y} \)) and the linear reward assumption in the last step. This also suggests that our consistent optimality radius \( \Lambda_i \) and its associated consistent optimality hardness \( H_{\Lambda} \) are more fundamental measures of problem hardness than the reward gap \( \Delta_i \) and its associated reward gap hardness \( H_{\Delta} \).

Next we discuss the implementation of the condition in line 11 of COCI for CPE-L. First, because linear functions are monotone, it is easy to see that we only need to check parameters \( \theta \) on the boundaries of \( \Theta_{i,t-1} \) (at most \( 2|\mathcal{Y}| \) calls to the oracle \( \phi \)). For simple constraints such as any subsets of size \( k \), it is easy to verify that \( \phi(\theta) \) is bi-monotone in this case, and thus we have efficient implementation of the condition as given in Theorem 3. For more complicated combinatorial constraints, it is still an open question on whether efficient implementation of the condition in line 11 exists when oracle \( \phi \) is given. The CLUCB algorithm, on the other hand, does have an efficient implementation for all CPE-L problems as long as the oracle \( \phi \) is given.

Therefore, compared with CLUCB in terms of efficient implementation, COCI can be viewed as taking the trade-off between the complexity of the reward functions and the complexity of combinatorial constraints. In particular, COCI could handle more complicated nonlinear reward functions on real vectors, and allow efficient implementation (due to bi-monotonicity) under simple constraints, while CLUCB deals with complicated combinatorial constraints but could only work with linear reward functions on binary vectors.

6 Future Work

There are a number of open problems and future directions. For example, one can consider the fixed budget setting of CPE-CS; the game stops after a fixed number \( T \) of rounds where \( T \) is given before the game starts, and the learner needs to minimize the probability of error \( \Pr[y^* \neq y^*] \). One may also consider the PAC setting: with probability at least \( 1 - \delta \) the algorithm should output a decision with reward at most \( \varepsilon \) away from the optimal reward. This setting may further help to eliminate the requirement of finite decision class \( \mathcal{Y} \). Another direction is to combine the advantage of COCI and CLUCB to design a unified algorithm that allows efficient implementation for all CPE-CS problems. How to incorporate approximation oracle instead of the exact oracle into the CPE framework is also an interesting direction.
Acknowledgments

This work was supported in part by the National Basic Research Program of China Grant 2011CBA00300, 2011CBA00301, the National Natural Science Foundation of China Grant 61033001, 61361136003, 61433014.

References


