Causal Bandits with Unknown Graph Structure

Yangyi Lu
Department of Statistics
University of Michigan
yylu@umich.edu

Amirhossein Meisami
Adobe Inc.
meisami@adobe.com

Ambuj Tewari
Department of Statistics
University of Michigan
tewaria@umich.edu

Abstract

In causal bandit problems, the action set consists of interventions on variables of a causal graph. Several researchers have recently studied such bandit problems and pointed out their practical applications. However, all existing works rely on a restrictive and impractical assumption that the learner is given full knowledge of the causal graph structure upfront. In this paper, we develop novel causal bandit algorithms without knowing the causal graph. Our algorithms work well for causal trees, causal forests and a general class of causal graphs. The regret guarantees of our algorithms greatly improve upon those of standard multi-armed bandit (MAB) algorithms under mild conditions. Lastly, we prove our mild conditions are necessary: without them one cannot do better than standard MAB algorithms.

1 Introduction

A multi-armed bandit (MAB) problem is one of the classic models of sequential decision making (Auer et al., 2002; Agrawal and Goyal, 2012, 2013a). Statistical measures such as regret and sample complexity measure how fast learning algorithms achieve near optimal performance in bandit problems. However, both regret and sample complexity for MAB problems necessarily scale with the number of actions without further assumptions. To address problems with a large action set, researchers have studied various types of structured bandit problems where additional assumptions are made on the structure of the reward distributions of the various actions. Algorithms for structured bandit problems exploit the dependency among arms to reduce the regret or sample complexity. Examples of structured bandit problems include linear bandits (Abbasi-Yadkori et al., 2011; Agrawal and Goyal, 2013b), sparse linear bandits (Abbasi-Yadkori et al., 2012), and combinatorial bandits (Cesa-Bianchi and Lugosi, 2012; Combes et al., 2015).

In this paper, we study a different kind of structured bandit problems: *causal bandits*. In this setting, actions are composed of interventions on variables of a causal graph. Many real world problems can be modeled via causal bandits. In healthcare applications, the physician adaptively adjusts the dosage of multiple drugs to achieve some desirable clinical outcome (Liu et al., 2020). In email campaign problems, marketers adjust for features of commercial emails to attract more customers and convert them into loyal buyers (Lu et al., 2019; Nair et al., 2021). Genetic engineering also involves direct manipulation of one or more genes using biotechnology, such as changing the genetic makeup of cells to produce improved organisms (Wikipedia contributors, 2021). Recently there has been a flurry of works (Lattimore et al., 2016; Sen et al., 2017; Lee and Bareinboim, 2018; Lu et al., 2019; Lee and Bareinboim, 2019; Nair et al., 2021) on causal bandits that show how to achieve simple regret or cumulative regret not scaling with the action set size.

However, a major drawback of existing work is that they require significant prior knowledge. All existing works require that the underlying causal graph is given upfront. Some regret analysis works even assume knowing certain probabilities for the causal model. In practice, they are all strong assumptions.

35th Conference on Neural Information Processing Systems (NeurIPS 2021).

In this paper, our goal is to develop causal bandit algorithms that 1) do not require prior knowledge of the causal graph and 2) achieve stronger worst-case regret guarantees than non-causal algorithms such as Upper Confidence Bound (UCB) and Thompson Sampling (TS) whose regret often scales *at least polynomially* with the number of nodes n in the causal graph. Unfortunately, this goal cannot be achieved for general causal graphs. Consider the causal graph consists of isolated variables and the reward directly depends on one of them. Then in the worst case there is no chance to do better than standard algorithms since no meaningful causal relations among variables can be exploited. In this paper, we study what classes of causal graphs on which we can achieve the goal.

Our Contributions. We summarize our contributions below.

1. We first study causal bandit problems where the unknown causal graph is a directed tree, or a causal forest. This setting has wide applications in biology and epidemiology (Greenewald et al., 2019; Burgos et al., 2008; Kontou et al., 2016; Pavlopoulos et al., 2018). We design a novel algorithm **C**entral **N**ode UCB (CN-UCB) that *simultaneously exploits* the reward signal and the tree structure to efficiently find the direct cause of the reward and then applies the UCB algorithm on a reduced intervention set corresponding to the direct cause.
2. Theoretically, we show under certain identifiability assumptions, the regret of our algorithm only scales *logarithmically* with the number of nodes n in the causal graph. To our knowledge, this is the first regret guarantee for unknown causal graph that provably outperforms standard MAB algorithms. We complement our positive result with lower bounds showing the indentifiability assumptions are necessary.
3. Furthermore, we generalize CN-UCB to a more general class of graphs that includes causal trees, causal forests, proper interval graphs, etc. Our algorithm first constructs undirected clique (junction) trees and again simultaneously exploits the reward signal and the junction-tree structure to efficiently find the direct cause of the reward. We also extend our regret guarantees to this class of graphs.

In many scenarios, our algorithms do *not* recover the full underlying causal graph structure. Therefore, our results deliver the important conceptual message that *exact causal graph recovery is not necessary in causal bandits* since the main target is to maximize the reward.

2 Related work

The causal bandit framework was proposed by Lattimore et al. (2016) and has been studied in various settings since then. In the absence of confounders, Lattimore et al. (2016) and Sen et al. (2017) studied the best arm identification problem assuming that the exact causal graph and the way interventions influence the direct causes of the reward variable are given. Lu et al. (2019) and Nair et al. (2021) proposed efficient algorithms that minimize the cumulative regret under the same assumptions. When confounders exist, Lee and Bareinboim (2018, 2019) developed effective ways to reduce the intervention set using the causal graph before applying any standard bandit algorithm, such as UCB. Even though the performance of above works improved upon that of standard bandit MAB algorithms, they all make the strong assumption of knowing the causal graph structure in advance.

In our setting, the underlying causal graph is not known. Then a natural approach is to first learn the causal graph through interventions. There are many intervention design methods developed for causal graph learning under different assumptions (He and Geng, 2008; Hyttinen et al., 2013; Shanmugam et al., 2015; Kocaoglu et al., 2017; Lindgren et al., 2018; Greenewald et al., 2019; Squires et al., 2020). However, this approach is not sample efficient because it is not necessary to recover the full causal graph in order to maximize rewards. de Kroon et al. (2020) tackled this problem with unknown graph structure based on separating set ideas. However, this implicitly requires the existence of a set of non-intervenable variables that d-separate the interventions and the reward. Moreover, their regret bound does not improve upon non-causal algorithms such as UCB. In this paper, we take a different approach which uses the reward signal to efficiently learn the direct causes of the reward.

Our approach is inspired by Greenewald et al. (2019) which proposed a set of central node algorithms that can recover the causal tree structure within $O(\log n)$ single-node interventions. Squires et al. (2020) also extended the central node idea to learn a general class of causal graphs that involves constructing junction trees and clique graphs. Following these works, our causal bandit algorithms

also adaptively perform interventions on central nodes to learn the direct causes of the reward variable. However, our algorithms differ from theirs because we also take the reward signal into account.

3 Preliminaries

In this section, we follow the notation and terminology of Lattimore et al. (2016); Greenewald et al. (2019) for describing causal models and causal bandit problems.

3.1 Causal Models

A causal model consists of a directed acyclic graph (DAG) D over a set of random variables $\mathcal{X} = \{X_1, \ldots, X_n\}$ and a joint distribution P that factorizes over D. The parents (children) of a variable X_i on graph D, denoted by $\text{Pa}_D(X_i)$ (or $\text{Ch}_D(X_i)$), are the set of variables X_j such that there is a directed edge from X_j to X_i (or from X_i to X_j) on graph D. The set of ancestors (descendants) of a variable X_i, denoted by $\text{An}_D(X_i)$ (or $\text{De}_D(X_i)$), are the set of variables X_j such that there is a path from X_j to X_i (or from X_i to X_j) on D. Without loss of generality, we assume the domain set for every X_i is $\text{Dom}(X_i) = [K] := \{1, \ldots, K\}$. For every X_i, we write the set of neighbors of X_i in graph D as $N_D(X_i)$ including variables X_j such that there is an edge between X_j and X_i regardless of the direction. The maximum degree of an undirected graph G is denoted by $d_{\max}(G)$. Throughout, we denote the true causal graph by D, use $V(\cdot)$ as the set of vertices of a graph and define $\text{skeleton}(\cdot)$ as the undirected graph obtained by replacing the arrows in the directed graph with undirected edges.

Definition 1 (Directed Tree and Causal Tree)**.** *A directed tree is a DAG whose underlying undirected graph is a tree and all its edges point away from the root. A causal tree is a causal model whose underlying causal graph is a directed tree.*

For a node X_i on a directed or undirected tree D and its neighbor $Y \in N_D(X_i)$, we write $B_D^{X_i:Y}$ as the set of nodes that can be reached from Y through any path on the graph (regardless of the directions of edges on the path), when the edge between X_i and Y is cut out from D. Note that the neighbor Y itself is always included in branch $B_D^{X_i:Y}$.

Definition 2 (Central Node (Greenewald et al., 2019))**.** *A central node v_c of an undirected tree $\mathcal{T}$ with respect to a distribution q over the nodes is one for which $\max_{j \in N_{\mathcal{T}}(v_c)} q(B_{\mathcal{T}}^{v_c:X_j}) \leq 1/2$. At least one such v_c is guaranteed to exist for any distribution q (Jordan, 1869; Greenewald et al., 2019).*

Informally, a central node v_c guarantees that the weight of every branch around v_c cannot be larger than $1/2$ according to the distribution $q(\cdot)$.

Definition 3 (Essential Graph)**.** *The class of causal DAGs that encode the same set of conditional independences is called the Markov equivalence class. Denote the Markov equivalence class of a DAG D by $[D]$. The essential graph of D, denoted by $\mathcal{E}(D)$, has the same skeleton as D, with directed edges $X_i \to X_j$ if such edge direction between X_i and X_j holds for all DAGs in $[D]$, and undirected edges otherwise.*

The chain components of $\mathcal{E}(D)$, denoted by $\text{CC}(\mathcal{E}(D))$, are the connected components of $\mathcal{E}(D)$ after removing all directed edges. Every chain component of $\mathcal{E}(D)$ is a chordal graph (Andersson et al., 1997). A DAG whose essential graph has a single chain component is called a *moral* DAG (Greenewald et al., 2019).

Definition 4 (Causal Forest (Greenewald et al., 2019))**.** *A causal graph is said to be a causal forest if each of the undirected components of the essential graph are trees.*

Many widely used causal DAGs including causal trees and bipartite causal graphs are examples of causal forest (Greenewald et al., 2019). Bipartite graph applications can range from biological networks, biomedical networks, biomolecular networks to epidemiological networks (Burgos et al., 2008; Kontou et al., 2016; Pavlopoulos et al., 2018).

3.2 Causal Bandit Problems

In causal bandit problems, the action set consists of interventions (Lattimore et al., 2016). An intervention on node X removes all edges from $\text{Pa}_D(X)$ to X and results in a post-intervention

distribution denoted by $P(\{X\}^c|\text{do}(X=x))$ over $\{X\}^c \triangleq \mathcal{X} \setminus \{X\}$. An empty intervention is represented by $\text{do}()$. The reward variable $\mathbf{R}$ is real-valued and for simplicity, we assume the reward is only directly influenced by one of the variables in $\mathcal{X}$, which we call the *reward generating variable* denoted by X_R [1]. The learner does not know the identity of X_R. Since there is only one X_R in our setting, the optimal intervention must be contained in the set of single-node interventions as follows: $\mathcal{A} = \{\text{do}(X=x) \mid X \in \mathcal{X}, x \in [K]\}$. Thus, we focus on above intervention set with $|\mathcal{A}| = nK$ throughout the paper.

We denote the expected reward for intervention $a = \text{do}(X=x)$ by $\mu_a = \mathbb{E}[\mathbf{R}|a]$. Then $a^* = \text{argmax}_{a\in\mathcal{A}}\,\mu_a$ is the optimal action and we assume that $\mu_a \in [0,1]$ for all a. A random reward for $a = \text{do}(X=x)$ is generated by $\mathbf{R}|_a = \mu_a + \varepsilon$, where ε is 1-subGaussian. At every round t, the learner pulls $a_t = \text{do}(X_t = x_t)$ based on the knowledge from previous rounds and observes a random reward R_t and the values of all $X \in \mathcal{X}$. The objective of the learner is to minimize the cumulative regret $R_T = T\mu_{a^*} - \sum_{t=1}^{T}\mu_{a_t}$ without knowing the causal graph D.

We make the following assumptions below.

Assumption 1. *The following three causal assumptions hold:*

- Causal sufficiency: *for every pair of observed variables, all their common causes are also observed.*
- Causal Markov condition: *every node in causal graph G is conditionally independent of its nondescendents, given its parents.*
- Causal faithfulness condition: *the set of independence relations derived from Causal Markov condition is the exact set of independence relations.*

Assumption 1 is commonly made in causal discovery literature (Peters et al., 2012; Hyttinen et al., 2013; Eberhardt, 2017; Greenewald et al., 2019). Equivalently speaking, there is no latent common causes and the correpondence between d-separations and conditional independences is one-to-one.

Assumption 2 (Causal Effect Identifiability)**.** *There exists an $\varepsilon > 0$, such that for any two variables $X_i \to X_j$ in graph D, we have $|P(X_j = x|\text{do}(X_i = x')) - P(X_j = x)| > \varepsilon$ holds for some $x \in Dom(X_j), x' \in Dom(X_i)$.*

Assumption 2 is necessary. It states that if there is a direct causal relation between two variables on the graph, the causal effect strength cannot go arbitrarily small. A similar version of this assumption was also made by Greenewald et al. (2019)) in their causal graph learning algorithms. We show the necessity of this assumption in Section 6. Intuitively, without this assumption, any causal relation among variables cannot be determined through finite intervention samples and $\Omega(\sqrt{nKT})$ worst-case regret is the best one can hope for.

Assumption 3 (Reward Identifiability)**.** *We assume that for all $X \in An(X_R)$, there exists $x \in [K]$ such that $|\mathbb{E}[\mathbf{R}|\text{do}()] - \mathbb{E}[\mathbf{R}|\text{do}(X=x)]| \geq \Delta$, for some universal constant $\Delta > 0$.*

Lastly, we show that Assumption 3 is also necessary. It guarantees a difference on the expected reward between the observations and after an intervention on an ancestor of X_R. In Section 6, we prove that the worst-case regret is again lower bounded by $\Omega(\sqrt{nKT})$ without this assumption.

In the following sections, we describe our causal bandit algorithms that focus on intervention design to minimize regret. Like most intervention design works, our algorithm take the essential graph $\mathcal{E}(D)$ and observational probabilities over $\mathcal{X}$ (denoted by $P(\mathcal{X})$) as the input, which can be estimated from enough observational data[2] (He and Geng, 2008; Greenewald et al., 2019; Squires et al., 2020).

4 CN-UCB for trees and forests

We start with introducing our algorithm CN-UCB (Algorithm 1) when the causal graph D is a directed tree. Before diving into details, we summarize our results as follows:

Theorem 1 (Regret Guarantee for Algorithm 1)**.** *Define* $B = \max\left\{\frac{32}{\Delta^2}\log\left(\frac{8nK}{\delta}\right), \frac{2}{\varepsilon^2}\log\left(\frac{8n^2K^2}{\delta}\right)\right\}$ *and* $T_1 = KB(2+d)\log_2 n$, *where* $0 < \delta < 1$.

[1] For multiple reward generating variables cases, one can repeatedly run our proposed algorithms and recover each of them one by one. We discuss this setting in Section C

[2] Observational data is usually much more cheaper than interventional data (Greenewald et al., 2019).

Suppose we run CN-UCB with $T \gg T_1$ interventions in total and $T_2 := T - T_1$. Then with probability at least $1-\delta$, we have

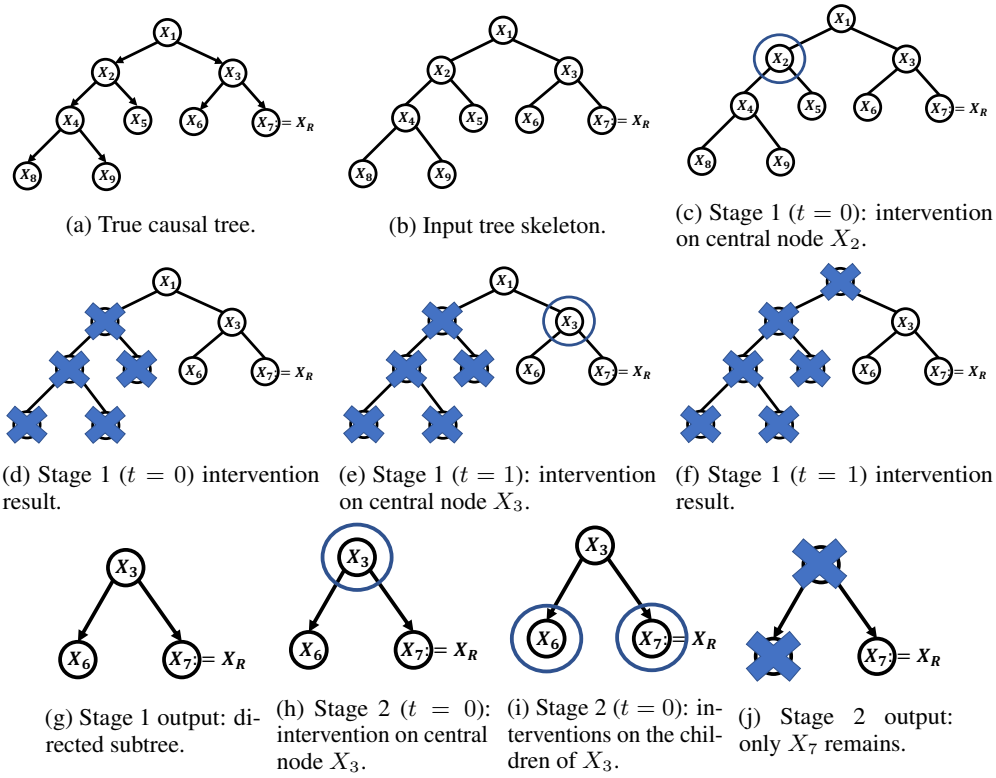
$$R_T = \widetilde{O}\left(K \max\left\{\frac{1}{\Delta^2}, \frac{1}{\varepsilon^2}\right\} d(\log n)^2 + \sqrt{KT}\right), \tag{1}$$

where $d := d_{\max}(\text{skeleton}(D))$ and $\widetilde{O}$ ignores poly-log terms non-regarding to n.

From above results, we see that especially for large n and small maximum degree d, Algorithm 1 outperforms the standard MAB bandit approaches that incur $\widetilde{O}(\sqrt{nKT})$ regret.

4.1 Description for CN-UCB

Our method contains three main stages. There is no v-structure in a directed tree, so the essential graph obtained from observational data is just the tree skeleton $\mathcal{T}_0$ and we take it as our input in Algorithm 1. In stage 1, Algorithm 1 calls Algorithm 3 to find a directed sub-tree $\widetilde{\mathcal{T}}_0$ that contains X_R by performing interventions on the central node (may change over time) on the tree skeleton. In stage 2, Algorithm 1 calls Algorithm 4 to find the reward generating node X_R by central node interventions on $\widetilde{\mathcal{T}}_0$. We prove that X_R can be identified correctly with high probability from the first two stages, so a UCB algorithm can then be applied on the reduced intervention set $\mathcal{A}_R = \{\text{do}(X_R = k) \mid k = 1, \ldots, K\}$ for the remaining rounds in stage 3. In the remainder of this section, we explain our algorithm design for each stage in detail and use an example in Figure 1 to show how CN-UCB proceeds step by step.



(a) True causal tree. (b) Input tree skeleton. (c) Stage 1 ($t = 0$): intervention on central node X_2.

(d) Stage 1 ($t = 0$) intervention result. (e) Stage 1 ($t = 1$): intervention on central node X_3. (f) Stage 1 ($t = 1$) intervention result.

(g) Stage 1 output: directed subtree. (h) Stage 2 ($t = 0$): intervention on central node X_3. (i) Stage 2 ($t = 0$): interventions on the children of X_3. (j) Stage 2 output: only X_7 remains.

Figure 1: Causal Tree Example for CN-UCB procedure.

In the beginning of CN-UCB, we collect reward data under empty intervention to obtain the observational reward mean estimate $\hat{R}$, which will be used in later steps to check whether a variable is an ancestor of the true reward generating variable X_R or not.

In our example, the true causal graph and its essential graph are displayed in Figure 1a and 1b, where the true reward generating variable is X_7.

Algorithm 1 Central Node Upper Confidence Bound (CN-UCB)

1: **Input:** Tree skeleton $\mathcal{T}_0$, K, Δ, ε, B, T_2, observational probabilities $P(\mathcal{X})$.
2: Perform do() for B times, collect $R_1, \ldots, R_B$
3: Obtain reward estimate for the empty intervention do(): $\hat{R} \leftarrow \frac{1}{B}\sum_{b=1}^{B} R_b$.
4: **Stage 1:** Find a directed subtree that contains X_R: $\widetilde{\mathcal{T}}_0 \leftarrow$ Find Sub-tree($\mathcal{T}_0, K, B, \varepsilon, \Delta, \hat{R}$). //call Algorithm 3 in Section B.
5: **Stage 2:** Find the key node that generates the reward: $X_R \leftarrow$ Find Key Node($\widetilde{\mathcal{T}}_0, K, B, \Delta, \hat{R}$). //call Algorithm 4 in Section B.
6: **Stage 3:** Apply UCB algorithm on $\mathcal{A}_R = \{\text{do}(X_R = k) \mid k = 1, \ldots, K\}$ for T_2 rounds.

Stage 1. The goal of this stage is to find a *directed* subtree that contains X_R within $O(\log n)$ single-node interventions. We achieve this goal by adaptively intervening the central node that may change over time. To illustrate our main idea, we make the following claim for single-node interventions.

Claim 1 (Two outcomes from single-node interventions on variable X)**.** *1) Whether X is an ancestor of X_R or not can be determined. 2) The directed subtree induced by X as its root can be found.*

To see the first outcome, we obtain reward estimates under interventions over X. If the difference between any of the reward estimates and $\hat{R}$ is larger than a threshold, Assumption 3 guarantees us that X is an ancestor of X_R.

To see the second outcome, we estimate interventional probabilities $\hat{P}(Y = y|\text{do}(X = x))$ for all $Y \in N_{\mathcal{T}_0}(X)$ and $y, x \in [K]$. Comparing these quantities with the corresponding observational probabilities $P(Y = y)$, the directions of edges between X and its neighbors can be determined due to Assumption 2. Note that in a tree structure, at most one of the neighbors has causal effect on X while others are all causally influenced by X. Once the edges between X and its neighbors are oriented, all other edges in the subtree induced by X (as the root) can be oriented using the property that each variable has at most one parent in a tree structure. Thus, if we learn that X is indeed an ancestor of X_R, a directed subtree can be obtained immediately. Otherwise, we conclude that all the variables in the directed subtree induced by X cannot be X_R.

Then a key question is: how do we adaptively select which variables to intervene? Arbitrarily selecting variables may easily require $O(n)$ single-node interventions. For example, let us consider a graph $X_1 \to X_2 \to \cdots \to X_n$ and the reward generating variable is X_1. If we start from intervening on X_n and everytime move one-node backward towards intervening X_1, we need $O(n)$ single-node interventions in total to figure out the directed subtree containing X_1. To overcome this issue, we adopt the idea of central node algorithm proposed by Greenewald et al. (2019). Interventions on a central node $v_c(t)$ guarantees us to eliminate at least half of the nodes from future searching, because the directed subtree induced by $v_c(t)$ contains more than half of the nodes regarding to the current weighting function $q_t(\cdot)$. And as long as $v_c(t)$ is not found as an ancestor of X_R, the weights $q_{t+1}(\cdot)$ for variables in the directed subtree induced by $v_c(t)$ as the root will be set as zeros. Thus, stage 1 can finish within $O(\log n)$ single-node interventions on $\mathcal{T}_0$.

For the example in Figure 1, CN-UCB takes the tree skeleton in Figure 1b as its input and identifies X_2 as the central node. As Figure 1d shows, the directed subtree induced by root X_2 are crossed out from the candidate set for X_R since intervening on X_2 shows that X_2 is not an ancestor of the true X_R. CN-UCB then identifies X_3 as the central node over the remaining variables(Figure 1e). We will see X_3 is indeed an ancestor of X_7 from interventions on X_3. Thus, X_1 can be crossed out because X_1 is on the upstream direction of X_3 and cannot be the true X_R (Figure 1f). Finally, stage 1 outputs the directed subtree in Figure 1g whose root is X_3.

Stage 2. The goal of this stage is to identify X_R in the directed sub-tree $\widetilde{\mathcal{T}}_0$ within $O(\log n)$ single-node interventions. Similar to stage 1, we determine whether a node X is an ancestor of X_R by performing interventions on it. If we find a variable X that is not an ancestor of X_R according to its reward estimates, all nodes in the sub-tree induced by X can be eliminated from future searching. Otherwise, we continue to intervene on the children of X. Since X_R is unique, at most one child $Y \in \text{Ch}(X)$ is an ancestor of X_R. If such Y exists, we repeat above procedure on the directed

sub-tree induced by Y as the root and update weights for all other variables as zeros in the algorithm. Lastly, if none of the children $Y \in \text{Ch}(X)$ appears to be an ancestor of X_R, X itself must be X_R.

We again perform intervention on the central node at the beginning of each round in Algorithm 4 and intervene on its children if necessary. By the definition of central node, every round we can either eliminate at least half of the nodes or finish searching for X_R. Thus, stage 2 can be finished within $O(d \log_2 n)$ single node interventions on $\widetilde{\mathcal{T}}_0$, since the central node at each round has at most d children.

For our example, stage 2 identifies X_3 as the central node showing in Figure 1h. With high probability we can conclude X_3 is indeed an ancestor of $X_7 := X_R$, so stage 2 continues to intervene on its children X_6 and X_7 (Figure 1i). From the intervention results we will see that X_7 is also an ancestor of the true X_R so that X_3 and X_6 can be removed. Finally, stage 2 outputs X_7 with high probability (Figure 1j).

Stage 3. Once the first two stages output a variable, we simply apply the UCB algorithm over the reduced intervention set defined in Algorithm 1.

4.2 Extension of CN-UCB to causal forest

Generalizing CN-UCB to causal forest[3] defined in Definition 4 is straightforward. The causal forest version for CN-UCB is presented in Algorithm 5 (Section B). We simply run stage 1 and stage 2 of Algorithm 1 for every chain component (tree structure) of the causal forest essential graph until stage 2 finds the reward generating variable X_R with high probability. Then, a standard UCB algorithm can be applied on the reduced intervention set corresponding to X_R. The regret guarantee for Algorithm 5 is presented in Theorem 2.

Theorem 2 (Regret Guarantee for Algorithm 5)**.** *Define $C(D)$ as the number of chain components in $\mathcal{E}(D)$. Suppose we run Algorithm 5 (Section 5) with $T \gg 2KB(d + C(D)) \log_2 n := T_1$ total interventions and $T_2 = T - T_1$, where $0 < \delta < 1$ and $B = \max\left\{ \frac{32}{\Delta^2} \log\left(\frac{8nK}{\delta}\right), \frac{2}{\varepsilon^2} \log\left(\frac{8n^2K^2}{\delta}\right) \right\}$. Then under Assumptions 1, 2 and 3, with probability at least $1 - \delta$, we have*

$$R_T = \widetilde{O}\left(K \max\left\{ \frac{1}{\Delta^2}, \frac{1}{\varepsilon^2} \right\} (d + C(D))(\log n)^2 + \sqrt{KT} \right), \tag{2}$$

where $\widetilde{O}$ ignores poly-log terms non-regarding to n and $d := \max_{\mathcal{T}_0 \in CC(\mathcal{E}(D))} d_{\max}(\mathcal{T}_0)$.

5 Extension of CN-UCB to a general class of causal graphs

In this section, we extend CN-UCB to more general causal graphs. Our approach involves searching for the reward generating node X_R within each undirected chain component of $\mathcal{E}(D)$. In order to use the central node idea, we try to construct tree structures on each component.

5.1 Preliminaries for the general graph class

Before describing our algorithm, we first review definitions of clique (junction) tree and clique graph for an undirected graph G following Squires et al. (2020).

Definition 5 (Clique)**.** *A clique $C \subset V(G)$ is a subset of nodes in which every pair of nodes are connected with an edge. We say a clique C is maximal if for any $v \in V(G) \setminus C$, $C \cup \{v\}$ is not a clique. The maximal cliques set of G is denoted by $\mathcal{C}(G)$ with its clique number defined as $\omega(G) = \max_{C \in \mathcal{C}(G)} |C|$, where $|C|$ denotes the number of nodes in clique C. We use $|\mathcal{C}(G)|$ to denote the number of cliques in set $\mathcal{C}(G)$.*

Definition 6 (Clique Tree (Junction Tree))**.** *A clique tree $\mathcal{T}_G$ for G is a tree with maximal cliques $\mathcal{C}(G)$ as vertices that satisfies the junction tree property, i.e. for all $v \in V(G)$, the induced subgraph on the set of cliques containing v is a tree. We denote the set of clique trees for graph G by $\mathcal{T}(G)$.*

[3]Note that in our setting, causal forest refers to a type of causal graph. This should not be confused with the causal forest machine learning method which is similar to random forest.

Algorithm 2 CN-UCB for General Causal Graph

Input: essential graph $\mathcal{E}(D), K, \Delta, \varepsilon, B, T_2$, observational probabilities $P(\mathcal{X})$.
Initialize: found $\leftarrow$ False
Perform do() for B times, collect reward data $R_1, \ldots, R_B$; $\hat{R} \leftarrow \frac{1}{B}\sum_{b=1}^{B} R_b$.
for $G \in \text{CC}(\mathcal{E}(D))$ **do**
  Stage 1: Create a junction tree $\mathcal{T}_G$ and find a directed sub-junction-tree that contains X_R: $\widetilde{\mathcal{T}}_G \leftarrow$ Find Sub-junction-tree($G, \mathcal{T}_G, K, B, \varepsilon, \Delta, \hat{R}$). //call Algorithm 7 in Section B.
  if $\widetilde{\mathcal{T}}_G$ is not empty **then**
    Stage 2: Find a clique in $\widetilde{\mathcal{T}}_G$ that contains X_R: $C_0 \leftarrow$Find Key Clique($\widetilde{\mathcal{T}}_G, K, B, \Delta, \hat{R}$). //call Algorithm 8 in Section B.
    Stage 3: Apply UCB algorithm on $\mathcal{A}_{C_0} = \{\text{do}(X = k) \mid X \in V(C_0), k = 1, \ldots, K\}$ for T_2 rounds.
    return Finished.
  end if
end for
return No clique that contains X_R is found. Apply UCB algorithm on the entire action space $\mathcal{A} = \{\text{do}(X = x) \mid X \in \mathcal{X}, x \in [K]\}$.

Definition 7 (Clique Graph). *A clique graph Γ_G is the graph union of $\mathcal{T}(G)$, i.e. $V(\Gamma_G) = \mathcal{C}(G)$ and $U(\Gamma_G) = \cup_{\mathcal{T}\in\mathcal{T}(G)} U(\mathcal{T})$, where $U(\cdot)$ denotes the set of undirected edges.*

Definition 8 (Directed Clique Tree). *A directed junction tree $\mathcal{T}_D$ of a moral DAG D has the same vertices and adjacencies as a clique tree $\mathcal{T}_G$ of $G = \text{skeleton}(D)$. We use asterisks as wildcards for edge points, e.g. $X_i* \to X_j$ denotes either $X_i \to X_j$ or $X_i \leftrightarrow X_j$. For each ordered pair of adjacent cliques $C_1 * - * C_2$ we orient the edge mark of C_2 as: 1) $C_1* \to C_2$ if for $\forall v_{12} \in C_1 \cap C_2$ and $\forall v_2 \in C_2 \setminus C_1$, we have $v_{12} \to v_2$ in the DAG D or 2) $C_1 * - C_2$ otherwise.*

Definition 9 (Directed Clique Graph). *The directed clique graph Γ_D of a moral DAG D is the graph union of all directed clique trees of D. The procedure of graph union is the same as Definition 8.*

Definition 10 (Intersection Comparable (Squires et al., 2020)). *A pair of edges $C_1 - C_2$ and $C_2 - C_3$ on a clique graph are intersection comparable if $C_1 \cap C_2 \subseteq C_2 \cap C_3$ or $C_1 \cap C_2 \supseteq C_2 \cap C_3$.*

Intersection-incomparable chordal graphs were introduced as "uniquely representable chordal graphs" in Kumar and Madhavan (2002). This class includes many familiar graph classes such as causal trees, causal forests and proper interval graphs (Squires et al., 2020). A clique graph is said to be intersection-incomparable if above intersection-comparable relation does not hold for any pair of edges. In this section, we design our algorithm for general causal graphs. If for every chain component $G \in \text{CC}(\mathcal{E}(D))$, the corresponding clique graph Γ_G is intersection-incomparable, we prove that our algorithm can achieve improved regret guarantee.

5.2 CN-UCB for general causal graphs

Our approach for causal bandit problems with general causal graphs is similar as previous settings. We try to find a subset of variables $\mathcal{X}_{\text{sub}}$ that contain the true reward generating node X_R with high probability and then apply UCB algorithm on interventions over $\mathcal{X}_{\text{sub}}$.

Our method is presented in Algorithm 2. We search for X_R in every chain component $G \in \text{CC}(\mathcal{E}(D))$ separately. In stage 1, we create a junction tree $\mathcal{T}_G$ and search for a directed sub-junction-tree $\widetilde{\mathcal{T}}_G$ that contains X_R via Algorithm 7. If such a $\widetilde{\mathcal{T}}_G$ exists, we keep searching for a clique C_0 in $\widetilde{\mathcal{T}}_G$ that contains X_R in stage 2 via Algorithm 8 and apply UCB algorithm on single-node interventions over variables in C_0. Otherwise, we repeat above procedure for the next chain component.

Stage 1 and stage 2 in Algorithm 2 also use the idea of central node interventions. But different from previous settings, Algorithm 7 and Algorithm 8 proceed by performing interventions on variables in the central clique that changes over time and every central clique intervention can eliminate at least half of the cliques on $\mathcal{T}_G$ or $\widetilde{\mathcal{T}}_G$ from future searching. A central clique for a junction tree is defined in the same way as a central node for a tree in that we just treat the cliques on a junction tree as the "node" in Definition 2.

Remark on the intersection-incomparable property. We now explain why we need the intersection-incomparable property like other central node based algorithm that also involves constructing clique graphs (Squires et al., 2020). In general, it is possible for a directed junction tree $\mathcal{T}$ to have v-structures over cliques, which can make us unable to finish searching for a directed sub-junction-tree that contains X_R by intervening on variables in $O(\log|\mathcal{T}|)$ cliques, where $|\mathcal{T}|$ denotes the number of cliques on $\mathcal{T}$. The reason is, a clique node may have more than one clique parents in $\mathcal{T}$, so that we cannot eliminate more than half of the cliques by interventions over the central clique. In total, we may need to perform $O(n)$ single-node interventions to find X_R and incur $O(n)$ regret. However, for every chain component G of the essential graph, if its clique graph Γ_G is intersection-incomparable, then there is no v-structure over cliques in any junction tree $\mathcal{T}_G$ of G (Squires et al., 2020), i.e. every node has at most one parent node.

Algorithm 2 can take any graph input no matter whether intersection-incomparable property holds or not. If the property does not hold, Algorithm 2 may output nothing at line 5 or line 7 for all component $G \in \text{CC}(\mathcal{E}(D))$ (will not output an incorrect sub-junction tree or clique by its design). Thus, if the learner finds that Algorithm 2 does not output anything after line 5 or 7 for all components, standard UCB algorithm can be used on the entire action set (see line 12 in Algorithm 2).

Theorem 3 (Regret guarantee for Algorithm 2 (Γ_G is intersection-incomparable for all $G \in \text{CC}(\mathcal{E}(D))$)). *Define $0 < \delta < 1$ and $B = \max\left\{\frac{32}{\Delta^2}\log\left(\frac{8nK}{\delta}\right), \frac{2}{\varepsilon^2}\log\left(\frac{8n^2K^2}{\delta}\right)\right\}$. Suppose we run Algorithm 2 for $T \gg B + KB\log_2 n\left(\omega(G_R) + d\omega(G_R) + \sum_{G\in CC(\mathcal{E}(D))}\omega(G)\right) := T_1$ total number of interventions and $T_2 := T - T_1$. Then under Assumptions 1, 2 and 3, with probability at least $1-\delta$, we have*

$$R_T = \widetilde{O}\left(\left(d(\mathcal{T}_{G_R})\omega(G_R) + \sum_{G\in CC(\mathcal{E}(D))}\omega(G)\right)K\max\left\{\frac{1}{\Delta^2}, \frac{1}{\varepsilon^2}\right\}\log n\log \mathcal{C}_{max} + \sqrt{\omega(G_R)KT}\right) \tag{3}$$

where $d(\mathcal{T}_{G_R})$ denotes the maximum degree of junction tree $\mathcal{T}_{G_R}$, G_R denotes the chain component that contains the true X_R and $\mathcal{C}_{max} \triangleq \max_{G\in CC(\mathcal{E}(D))}|\mathcal{C}(G)|$. $\tilde{O}(\cdot)$ ignores poly-log terms non-regarding to n or number of cliques.

Above regret bound shows that our algorithm outperforms standard MAB algorithms especially when n is large and the degree $d(\mathcal{T}_{G_R})$ and the clique numbers $\omega(G)$ are small . Also, above result reduces to Theorem 2 when D is a causal forest, because $\omega(G)$ is always 2 for tree-structure chain components.

6 Lower bounds

In this section we show without Assumption 2 or 3, any algorithm will incur an $\Omega\left(\sqrt{nkT}\right)$ regret in the worst-case, which is exponentially worse than our results in terms of n.

Definition 11 (nK-arm Gaussian Causal Bandit Class). *For a bandit instance in nK-arm Gaussian causal bandit class, actions are composed by single-node interventions over n variables $\mathcal{X} = \{X_1, \ldots, X_n\}$: $\mathcal{A} = \{\text{do}(X_i = x)|x \in [K]; i = 1, \ldots, n\}$ with $|\mathcal{A}| = nK$. The reward for every action is Gaussian distributed and is directly influenced by one of $\mathcal{X}$.*

Theorem 4 (Minimax Lower Bound Without Assumption 2). *Let $\mathcal{E}$ be the set of nK-armed Gaussian bandits (Definition 11) where the instances in $\mathcal{E}$ does not satisfy Assumption 2. We show that the minimax regret is $\inf_{\pi\in\Pi}\sup_{\nu\in\mathcal{E}}\mathbb{E}[R_T(\pi,\nu)] = \Omega(\sqrt{nKT})$, where Π denotes the set of all policies.*

Theorem 5 (Minimax Lower Bound (Assumption 2 holds but Assumption 3 does not hold)). *Let $\mathcal{E}$ be the set of nK-armed Gaussian bandits (Definition 11) where the instances in $\mathcal{E}$ satisfy Assumption 2 but does not satisfy Assumption 3. We show that the minimax regret is $\inf_{\pi\in\Pi}\sup_{\nu\in\mathcal{E}}\mathbb{E}[R_T(\pi,\nu)] = \Omega(\sqrt{nKT})$, where Π denotes the set of all policies.*

7 Discussion

In this paper, we studied causal bandit problems with unknown causal graph structures. We proposed an efficient algorithm CN-UCB for the causal tree setting. The regret of CN-UCB scales logarithmically with the number of variables n when the intervention size is nK. CN-UCB was then extended to broader settings where the underlying causal graph is a causal forest or belongs to a general class of graphs. For the later two settings, our regret again only scales logarithmically with n. Lastly, we provide lower bound results to justify the necessity of our assumptions on the causal effect identifiability and the reward identifiability. Our approaches are the first set of causal bandit algorithms that do not rely on knowing the causal graph in advance and can still outperform standard MAB algorithms.

There are several directions for future work. First, one can generalize CN-UCB to the multiple reward generating node setting, i.e., more than one variable directly influences the reward. We expect this can be done by repeatedly applying stage 1 and 2 in CN-UCB to find all the reward generating nodes one by one and apply a standard MAB algorithm on the reduced intervention set. In this setting, it is natural to consider interventions that can be performed on more than one variables. Second, it will be interesting to develop instance dependent regret bounds, for example, through estimating the probabilities over the causal graph in CN-UCB. Lastly, one can also develop efficient algorithms that do not need to know the causal graph when confounders exist.

Acknowledgement

We thank our NeurIPS reviewers and meta-reviewer for helpful suggestions to improve the paper.

Funding transparency statement

Funding (financial activities supporting the submitted work): Funding in direct support of this work: NSF CAREER grant IIS-1452099, Adobe Data Science Research Award.

Competing Interests (financial activities outside the submitted work): None

References

Abbasi-Yadkori, Y., Pál, D., and Szepesvári, C. (2011). Improved algorithms for linear stochastic bandits. In *NIPS*, volume 11, pages 2312–2320.

Abbasi-Yadkori, Y., Pal, D., and Szepesvari, C. (2012). Online-to-confidence-set conversions and application to sparse stochastic bandits. In *Artificial Intelligence and Statistics*, pages 1–9. PMLR.

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

Agrawal, S. and Goyal, N. (2013a). Further optimal regret bounds for thompson sampling. In *Artificial intelligence and statistics*, pages 99–107. PMLR.

Agrawal, S. and Goyal, N. (2013b). Thompson sampling for contextual bandits with linear payoffs. In *International Conference on Machine Learning*, pages 127–135.

Andersson, S. A., Madigan, D., Perlman, M. D., et al. (1997). A characterization of markov equivalence classes for acyclic digraphs. *Annals of statistics*, 25(2):505–541.

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

Burgos, E., Ceva, H., Hernández, L., Perazzo, R. P., Devoto, M., and Medan, D. (2008). Two classes of bipartite networks: nested biological and social systems. *Physical Review E*, 78(4):046113.

Cesa-Bianchi, N. and Lugosi, G. (2012). Combinatorial bandits. *Journal of Computer and System Sciences*, 78(5):1404–1422.

Combes, R., Talebi, M. S., Proutiere, A., and Lelarge, M. (2015). Combinatorial bandits revisited. *arXiv preprint arXiv:1502.03475*.

de Kroon, A. A., Belgrave, D., and Mooij, J. M. (2020). Causal discovery for causal bandits utilizing separating sets. *arXiv preprint arXiv:2009.07916*.

Eberhardt, F. (2017). Introduction to the foundations of causal discovery. *International Journal of Data Science and Analytics*, 3(2):81–91.

Greenewald, K., Katz, D., Shanmugam, K., Magliacane, S., Kocaoglu, M., Adsera, E. B., and Bresler, G. (2019). Sample efficient active learning of causal trees. In *Advances in Neural Information Processing Systems*, pages 14302–14312.

He, Y.-B. and Geng, Z. (2008). Active learning of causal networks with intervention experiments and optimal designs. *Journal of Machine Learning Research*, 9(Nov):2523–2547.

Hyttinen, A., Eberhardt, F., and Hoyer, P. O. (2013). Experiment selection for causal discovery. *The Journal of Machine Learning Research*, 14(1):3041–3071.

Jordan, C. (1869). Sur les assemblages de lignes. *J. Reine Angew. Math*, 70(185):81.

Kocaoglu, M., Dimakis, A., and Vishwanath, S. (2017). Cost-optimal learning of causal graphs. In *International Conference on Machine Learning*, pages 1875–1884. PMLR.

Kontou, P. I., Pavlopoulou, A., Dimou, N. L., Pavlopoulos, G. A., and Bagos, P. G. (2016). Network analysis of genes and their association with diseases. *Gene*, 590(1):68–78.

Kumar, P. S. and Madhavan, C. V. (2002). Clique tree generalization and new subclasses of chordal graphs. *Discrete Applied Mathematics*, 117(1-3):109–131.

Lattimore, F., Lattimore, T., and Reid, M. D. (2016). Causal bandits: Learning good interventions via causal inference. In *Advances in Neural Information Processing Systems*, pages 1181–1189.

Lattimore, T. and Szepesvári, C. (2018). Bandit algorithms. *preprint*.

Lee, S. and Bareinboim, E. (2018). Structural causal bandits: where to intervene? In *Advances in Neural Information Processing Systems*, pages 2568–2578.

Lee, S. and Bareinboim, E. (2019). Structural causal bandits with non-manipulable variables. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 4164–4172.

Lindgren, E. M., Kocaoglu, M., Dimakis, A. G., and Vishwanath, S. (2018). Experimental design for cost-aware learning of causal graphs. *arXiv preprint arXiv:1810.11867*.

Liu, S., See, K. C., Ngiam, K. Y., Celi, L. A., Sun, X., and Feng, M. (2020). Reinforcement learning for clinical decision support in critical care: comprehensive review. *Journal of medical Internet research*, 22(7):e18477.

Lu, Y., Meisami, A., Tewari, A., and Yan, Z. (2019). Regret analysis of causal bandit problems. *arXiv preprint arXiv:1910.04938*.

Nair, V., Patil, V., and Sinha, G. (2021). Budgeted and non-budgeted causal bandits. In *International Conference on Artificial Intelligence and Statistics*, pages 2017–2025. PMLR.

Pavlopoulos, G. A., Kontou, P. I., Pavlopoulou, A., Bouyioukos, C., Markou, E., and Bagos, P. G. (2018). Bipartite graphs in systems biology and medicine: a survey of methods and applications. *GigaScience*, 7(4):giy014.

Peters, J., Mooij, J., Janzing, D., and Schölkopf, B. (2012). Identifiability of causal graphs using functional models. *arXiv preprint arXiv:1202.3757*.

Sen, R., Shanmugam, K., Dimakis, A. G., and Shakkottai, S. (2017). Identifying best interventions through online importance sampling. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pages 3057–3066. JMLR. org.

Shanmugam, K., Kocaoglu, M., Dimakis, A. G., and Vishwanath, S. (2015). Learning causal graphs with small interventions. *arXiv preprint arXiv:1511.00041*.

Squires, C., Magliacane, S., Greenewald, K., Katz, D., Kocaoglu, M., and Shanmugam, K. (2020). Active structure learning of causal dags via directed clique tree. *arXiv preprint arXiv:2011.00641*.

Wikipedia contributors (2021). Genetic engineering — Wikipedia, the free encyclopedia. `https://en.wikipedia.org/w/index.php?title=Genetic_engineering&oldid=1018922014`. [Online; accessed 26-April-2021].

A Proof for Theorems

A.1 Proof for Theorem 1

We first present a sample complexity result for interventions in stage 1 and 2.

Lemma 1 (Sample Complexity for Algorithm 1 Before Stage 3)**.** *Set* $B = \max\{\frac{32}{\Delta^2}\log\left(\frac{8nK}{\delta}\right), \frac{2}{\varepsilon^2}\log\left(\frac{8n^2K^2}{\delta}\right)\}$*, with probability at least* $1-\delta$*, stage 2 in Algorithm 1 outputs the true key node within* $KB(2+d)\log_2 n + B$ *interventions, where* d *is defined as the maximum degree of the causal tree skeleton.*

Proof for Lemma 1. By Hoeffding's inequality for bounded random variables, for any fixed $X, Y \in \mathcal{X}, x \in [K], y \in [K]$, we have

$$|P(Y=y|\text{do}(X=x)) - \hat{P}(Y=y|\text{do}(X=x))| \leq \sqrt{\frac{1}{2B}\log\left(\frac{8n^2K^2}{\delta}\right)},$$

with probability at least $1-\frac{\delta}{4n^2K^2}$. Furthermore, for fixed $X \in \mathcal{X}$ and $x \in [K]$, by Hoeffding's inequality for sub-gaussian random variables, we have

$$|\hat{R}^{\text{do}(X=x)} - \mathbb{E}[\mathbf{R}|\text{do}(X=x)]| \leq \sqrt{\frac{2}{B}\log\left(\frac{8nK}{\delta}\right)},$$

with probability at least $1-\frac{\delta}{4nK}$. For empty intervention, we write $\hat{R}$ for $\hat{R}^{\text{do}()}$, then

$$|\hat{R} - \mathbb{E}[\mathbf{R}|\text{do}()]| \leq \sqrt{\frac{2}{B}\log\left(\frac{4}{\delta}\right)}$$

holds with probability at least $1-\frac{\delta}{2}$. We define the union of above good events by E. By union bound, we have $P(E^c) \leq \delta$.

Under event E, suppose X is the key node, one can show that

$$|\hat{R} - \hat{R}^{\text{do}(X=x)}| \geq$$
$$|\mathbb{E}[\mathbf{R}|\text{do}()] - \mathbb{E}[\mathbf{R}|\text{do}(X=x)]| - |\hat{R} - \mathbb{E}[\mathbf{R}|\text{do}()]| - |\mathbb{E}[\mathbf{R}|\text{do}(X=x)] - \hat{R}^{\text{do}(X=x)}| \geq \frac{\Delta}{2}.$$

And for any X, Y, x, y such that $X \to Y$,

$$|P(Y=y) - \hat{P}(Y=y|\text{do}(v_c(t)=z))|$$
$$\geq |P(Y=y) - P(Y=y|\text{do}(v_c(t)=z))| - |P(Y=y|\text{do}(v_c(t)=z)) - \hat{P}(Y=y|\text{do}(v_c(t)=z))|$$
$$\geq \frac{\varepsilon}{2}.$$

Combine everything before stage 3, the total number of interventions is $2KB\log_2 n + dKB\log_2 n + B$. □

Proof for Theorem 1. Condition on the good event E in the proof of Lemma 1, stage 2 in Algorithm 1 returns the true reward generating node X_R. Combining with the total intervention sample results in Lemma 1 we have

$$\begin{aligned} R_T &\leq KB(2+d)\log_2 n + B + \sqrt{KT\log T} \\ &= O\left(K\max\left\{\frac{1}{\Delta^2}, \frac{1}{\varepsilon^2}\right\} d\log\left(\frac{nK}{\delta}\right)\log_2 n + \sqrt{KT\log T}\right). \end{aligned}$$

holds with probability at least $1-\delta$. □

A.2 Proof for Theorem 2

Proof. The proof is straightforward from Theorem 1. In Algorithm 5, there is only one tree component that contains X_R. Thus, in the worst case, we find X_R in the last component and stage 1 needs to be performed for $C(D)$ times. Combine with the results in Section A.1, the total number of interventions is $C(D)KB\log_2 n + KB\log_2 n + dKB\log_2 n + B$. Then, the regret can be bounded by:

$$\begin{aligned} R_T &\leq KB(d + C(D) + 1)\log_2 n + B + \sqrt{KT\log T} \\ &= O\left(K\max\left\{\frac{1}{\Delta^2}, \frac{1}{\varepsilon^2}\right\}(d + C(D))\log\left(\frac{nK}{\delta}\right)\log_2 n + \sqrt{KT\log T}\right). \end{aligned}$$

□

A.3 Proof for Theorem 3

Proof. We follow the proof idea in Theorem 1.
By Hoeffding's inequality for bounded random variables, for any fixed $X, Y \in \mathcal{X}, x \in [K], y \in [K]$, we have

$$|P(Y = y|\text{do}(X = x)) - \hat{P}(Y = y|\text{do}(X = x))| \leq \sqrt{\frac{1}{2B}\log\left(\frac{8n^2K^2}{\delta}\right)},$$

with probability at least $1 - \frac{\delta}{4n^2K^2}$. Furthermore, for fixed $X \in \mathcal{X}$ and $x \in [K]$, by Hoeffding's inequality for sub-gaussian random variables, we have

$$|\hat{R}^{\text{do}(X=x)} - \mathbb{E}[\mathbf{R}|\text{do}(X = x)]| \leq \sqrt{\frac{2}{B}\log\left(\frac{8nK}{\delta}\right)},$$

with probability at least $1 - \frac{\delta}{4nK}$. For empty intervention, we write $\hat{R}$ for $\hat{R}^{\text{do}()}$, then

$$|\hat{R} - \mathbb{E}[\mathbf{R}|\text{do}()]| \leq \sqrt{\frac{2}{B}\log\left(\frac{4}{\delta}\right)}$$

holds with probability at least $1 - \frac{\delta}{2}$. We define the union of above good events by E. By union bound, we have $P(E^c) \leq \delta$. We condition on event E for the following analysis.

By the design of Algorithm 2, we search for the reward generating node in every chain component of $\text{CC}(\mathcal{E}(D))$. In the worst case, we find X_R in the last component. In this case, Algorithm 7 (stage 1) is executed for every component G. and Algorithm 8 (stage 2) is executed only for the chain component that contains X_R. We use $|\mathcal{C}(G)|$ to denote the number of maximal cliques of component G for every $G \in \text{CC}(\mathcal{E}(D))$.

Specifically, for every chain component G, Algorithm 7 performs $KB\log_2|\mathcal{C}(G)|$ clique interventions, i.e. at most $KB\omega(G)\log_2|\mathcal{C}(G)|$ node interventions. For component G_R that contains X_R, Algorithm 8 performs $(KB + d(\mathcal{T}_{G_R})KB)\log_2|\mathcal{C}(G_R)|$ clique interventions, i.e. at most $(KB + d(\mathcal{T}_{G_R})KB)\omega(G_R)\log_2|\mathcal{C}(G_R)|$ node interventions.

Thus, the total number of interventions before applying UCB algorithm on the reduced intervention set is: $\sum_{G\in\text{CC}(\mathcal{E}(G))} KB\omega(G)\log_2|\mathcal{C}(G)| + dKB\omega(G_R)\log_2|\mathcal{C}(G_R)|$. Thus, conditioning on event E which holds with at least $1 - \delta$, we have

$$\begin{aligned} R_T &\leq \sum_{G\in\text{CC}(\mathcal{E}(D))} KB\omega(G)\log_2|\mathcal{C}(G)| + d(\mathcal{T}_{G_R})KB\omega(G_R)\log_2|\mathcal{C}(G_R)| + \sqrt{\omega(G_R)KT\log T} \\ &= \widetilde{O}\big(K\max\{\frac{1}{\Delta^2}, \frac{1}{\varepsilon^2}\}\log n(d(\mathcal{T}_{G_R})\omega(G_R)\log_2|\mathcal{C}(G_R)| + \sum_{G\in\text{CC}(\mathcal{E}(D))} \omega(G)\log_2|\mathcal{C}(G)|) \\ &\quad + \sqrt{\omega(G_R)KT}\big) \end{aligned}$$

where $\widetilde{O}(\cdot)$ ignores poly-log terms non-regarding to n or clique sizes. □

A.4 Proof for Theorem 4

Proof. We use an example to show that, without Assumption 2, no algorithm can achieve worst-case regret better than $\widetilde{O}(\sqrt{nKT})$.

Consider $(K+1)$-nary variables $X_1, \ldots, X_n \in \{0, 1, \ldots, K\}$ and action set $\mathcal{A} = \{\text{do}(X_i = x) | x \in \{1, \ldots, K\}; i = 1, \ldots, n\}$. Note that $|\mathcal{A}| = nK$.

We construct below $nK + 1$ bandit instances such that Assumption 2 does not hold.

Bandit Instance 0**:**

- Causal structure $X_1 \to X_2 \to \cdots \to X_n$.
- Probabilities assigned: $\mathbb{P}(X_1 = 0) = \mathbb{P}(X_{i+1} = 0 | X_i) = 1, i = 1, \ldots, n-1$.[4]
- Reward generation: $R \sim N(0, 1)$.
- Note: for all nK action, conditional reward mean is 0.

Bandit Instance k**, where** $k = 1, \ldots, K$**:**

- Causal structure $X_1 \to X_2 \to \cdots \to X_n$.
- Probabilities assigned: $\mathbb{P}(X_1 = 0) = \mathbb{P}(X_{i+1} = 0 | X_i) = 1, i = 1, \ldots, n-1$.
- Reward generation: $R \sim N(\Delta \mathbb{1}_{\{X_1 = k\}}, 1)$.
- Note: only action $\text{do}(X_1 = k)$ has reward mean Δ, otherwise 0.

Bandit Instance $(j-1)K + k \sim jK$**, where** $j = 2, \ldots, n-1, k = 1, \ldots, K$**:**

- Causal structure $X_j \to X_1 \to \cdots \to X_{j-1} \to X_{j+1} \to \cdot \to X_n$.
- Probabilities assigned: $\mathbb{P}(X_j = 0) = \mathbb{P}(X_1 = 0 | X_j) = \mathbb{P}(X_{j+1} = 0 | X_{j-1}) = \mathbb{P}(X_{i+1} = 0 | X_i) = 1, i = 1, \ldots, j-2, j+1, \ldots, n-1$.
- Reward generation: $R \sim N(\Delta \mathbb{1}_{\{X_j = k\}}, 1)$.
- Note: only action $\text{do}(X_j = k)$ has reward mean Δ, otherwise 0.

Bandit Instance $(n-1)K + k \sim nK$**, where** $k = 1, \ldots, K$**:**

- Causal structure $X_n \to \cdots \to X_2 \to X_1$.
- Probabilities assigned: $\mathbb{P}(X_n = 0) = \mathbb{P}(X_{i-1} = 0 | X_i) = 1, i = 2, \ldots, n$.
- Reward generation: $R \sim N(\Delta \mathbb{1}_{\{X_n = k\}}, 1)$.
- Note: only action $\text{do}(X_n = k)$ has reward mean Δ, otherwise 0.

Take $\Delta = \frac{1}{4}\sqrt{\frac{nK}{T}}$, using the results in exercise 15.2 in Lattimore and Szepesvári (2018) we know that there exists one bandit intance ν in above such that

$$\mathbb{E}[R_T] \geq \frac{1}{8}\sqrt{nKT}, \tag{4}$$

where the expectation is taken over the entire randomness.

We present below lemmas in Lattimore and Szepesvári (2018) to describe the proof for above conclusion.

[4] $\mathbb{P}(X_{i+1} = 0 | X_i) = 1$ means no matter what value X_i is, the conditional probability $\mathbb{P}(X_{i+1} = 0 | X_i)$ is always 1. Similar expression is also used for constructing other bandit instances.

Lemma 2 (Divergence decomposition). *Let $\nu = (P_1, \dots, P_k)$ be the reward distributions associated with one k-armed bandit, and let $\nu' = (P'_1, \dots, P'_k)$ be the reward distributions associated with another k-armed bandit. Fix some policy π and let $\mathbb{P}_\nu = \mathbb{P}_{\nu\pi}$ and $\mathbb{P}_{\nu'} = \mathbb{P}_{\nu'\pi}$ be the probability measures on the bandit model induced by the n-round interconnection of π and ν (π' and ν'). Then*

$$\mathbf{KL}(\mathbb{P}_\nu, \mathbb{P}_{\nu'}) = \sum_{i=1}^{k} \mathbb{E}_\nu [T_i(n)] \, \mathbf{KL}(P_i, P'_i). \tag{5}$$

Lemma 3 (Pinsker inequality). *For measures P and Q on the same probability space $(\Omega, \mathcal{F})$ that*

$$\delta(P, Q) \triangleq \sup_{A \in \mathcal{F}} P(A) - Q(A) \leq \sqrt{\frac{1}{2} \mathbf{KL}(P, Q)}. \tag{6}$$

Lemma 4. *Let $(\Omega, \mathcal{F})$ be a measurable space and let $P, Q : \mathcal{F} \to [0, 1]$ be probability measures. Let $a < b$ and $X : \Omega \to [a, b]$ be a $\mathcal{F}$-measurable random variable, we have*

$$\left| \int_\Omega X(\omega) dP(\omega) - \int_\Omega X(\omega) dQ(\omega) \right| \leq (b - a) \delta(P, Q), \tag{7}$$

where $\delta(P, Q)$ is as defined in Lemma 3.

Define $\mu^{(l)} = \{\mu^{(l)}_{\mathrm{do}(X_1=1)}, \dots, \mu^{(l)}_{\mathrm{do}(X_1=K)}, \dots, \mu^{(l)}_{\mathrm{do}(X_n=1)}, \dots \mu^{(l)}_{\mathrm{do}(X_n=K)}\} \in \mathbb{R}^{nK}$ as the reward mean vector for the l-th bandit instance in above. By construction, only the l-th element of $\mu^{(l)}$ is Δ, and all other elements are zeros. For any policy π, we define $\mathbb{P}_l$ and $\mathbb{E}_l$ as the probability measure and expectation induced by bandit instance l within T rounds.

Denote the number of plays on arm i up to time T by $T_i(T)$, we have

$$\begin{aligned}
\mathbb{E}_i [T_i(T)] &\leq \mathbb{E}_0 [T_i(T)] + T\delta(\mathbb{P}_0, \mathbb{P}_i) \text{ by Lemma 4} \\
&\leq \mathbb{E}_0 [T_i(T)] + T\sqrt{\frac{1}{2} \mathbf{KL}(\mathbb{P}_0, \mathbb{P}_i)} \text{ by Lemma 3} \\
&= \mathbb{E}_0 [T_i(T)] + T\sqrt{\frac{1}{2} \cdot \frac{1}{2} \Delta^2 \mathbb{E}_0 [T_i(T)]} \text{ by Lemma 2} \\
&= \mathbb{E}_0 [T_i(T)] + \frac{1}{8}\sqrt{nKT \mathbb{E}_0 [T_i(T)]} \text{ by } \Delta = \frac{1}{4}\sqrt{\frac{nK}{T}}.
\end{aligned}$$

Sum over the left term in above, using the property $\sum_{i=1}^{nK} \mathbb{E}_0 [T_i(T)] = T$, we have

$$\begin{aligned}
\sum_{i=1}^{nK} \mathbb{E}_i [T_i(T)] &\leq T + \frac{\sqrt{nKT}}{8} \sum_{i=1}^{nK} \sqrt{\mathbb{E}_0 [T_i(T)]} \\
&\leq T + \frac{1}{8} nKT.
\end{aligned}$$

Let $R_i \triangleq R_T(\pi, \nu_i)$ (the regret of applying policy π on the $i-$th bandit instance up to time T), where ν_i refers to the i-th bandit instance in above construction.

$$\begin{aligned}
\sum_{i=1}^{nK} \mathbb{E}[R_i] &= \Delta \sum_{i=1}^{nK} (T - \mathbb{E}_i [T_i(T)]) \\
&\geq \frac{1}{4}\sqrt{\frac{nK}{T}} (nKT - T - \frac{1}{8} nKT) \geq \frac{nKT}{8}\sqrt{\frac{nK}{T}} = \frac{nK}{8}\sqrt{nKT}.
\end{aligned}$$

Thus, there exists one bandit instance i^*, such that $\mathbb{E}[R_{i^*}] \geq \frac{1}{8}\sqrt{nKT}$. □

A.5 Proof for Theorem 5

Proof. Consider $(K + 2)$-nary variables $X_1, \dots, X_n \in \{0, 1, \dots, K + 1\}$ and action set $\mathcal{A} = \{\mathrm{do}(X_i = x) \mid x \in \{2, \dots, K + 1\}; i = 1, \dots, n\}$. Note that $|\mathcal{A}| = nK$.

We construct below $nK + 1$ bandit instances such that Assumption 2 holds but Assumption 3 does not hold.

Bandit Instance 0**:**

- Causal structure $X_n \to X_{n-1} \to \cdots \to X_1$.
- Probabilities assigned: $\mathbb{P}(X_n = 0) = 1$; $\mathbb{P}(X_{i-1} = 0|X_i = 0) = \mathbb{P}(X_{i-1} = 1|X_i = 1) = 1$; $P(X_{i-1} \geq 2|X_i) = 0$ for any value of X_i. $(i = 2, \ldots, n)$
- Reward generation: $R \sim N(0, 1)$.
- Note: for all nK action, their reward mean is 0.

Bandit Instance k**, where** $k = 1, \ldots, K$**:**

- Causal structure $X_n \to X_{n-1} \to \cdots \to X_1$.
- Probabilities assigned: $\mathbb{P}(X_n = 0) = 1$; $\mathbb{P}(X_{i-1} = 0|X_i = 0) = \mathbb{P}(X_{i-1} = 1|X_i = 1) = 1$; $P(X_{i-1} \geq 2|X_i) = 0$ for any value of X_i. $(i = 2, \ldots, n)$
- Reward generation: $R \sim N(\Delta \mathbb{1}_{\{X_1 = k+1\}}, 1)$.
- Note: only action do$(X_1 = k + 1)$ has reward mean Δ, otherwise 0.

Bandit Instance $(j-1)K + k \sim jK$**, where** $j = 2, \ldots, n-1, k = 1, \ldots, K$**:**

- Causal structure $X_{j-1} \to \cdots \to X_1 \to X_n \to X_{n-1} \to \cdots \to X_j$.
- Probabilities assigned: $\mathbb{P}(X_{j-1} = 0) = 1$, for other directly connected variable pairs $X \to Y$ in the graph we have $\mathbb{P}(Y = 0|X = 0) = \mathbb{P}(Y = 1|X = 1) = 1$ and $\mathbb{P}(Y \geq 2|X) = 0$ for any value of X.
- Reward generation: $R \sim N(\Delta \mathbb{1}_{\{X_j = k+1\}}, 1)$.
- Note: only action do$(X_j = k + 1)$ has reward mean Δ, otherwise 0.

Bandit Instance $(n-1)K + k \sim nK$**, where** $k = 1, \ldots, K$**:**

- Causal structure $X_{n-1} \to \cdots \to X_1 \to X_n$.
- Probabilities assigned: $\mathbb{P}(X_{n-1} = 0) = 1$, for other directly connected variable pairs $X \to Y$ in the graph we have $\mathbb{P}(Y = 0|X = 0) = \mathbb{P}(Y = 1|X = 1) = 1$ and $\mathbb{P}(Y \geq 2|X) = 0$ for any value of X.
- Reward generation: $R \sim N(\Delta \mathbb{1}_{\{X_n = k+1\}}, 1)$.
- Note: only action do$(X_n = k + 1)$ has reward mean Δ, otherwise 0.

In above $nK + 1$ instances, we see that Assumption 2 holds since $P(Y = 1) = 0$ but $P(Y = 1|X = 1) = 1$ for all directly connected variable pairs $X \to Y$. However, Assumption 3 does not hold. For example in bandit instance 1, no matter how we intervene on $X_n, \ldots, X_2$, by probabilities construction $X_1 \geq 2$ can never happen, which means the expected reward is always zero unless we direcly intervene on X_1. Similarly, Assumption 3 does not hold for other bandit instances as well.

Follow the exact proof for Theorem 4, we know that there exists one bandit instance ν in above such that

$$\mathbb{E}[R_T(\pi, \nu)] = \Omega(\sqrt{nKT}), \tag{8}$$

for any policy π. □

B Algorithms

We present our algorithms in this section.

Algorithm 3 Find Sub-tree

Input: tree skeleton $\mathcal{T}_0$, K, B, ε, Δ, $\hat{R}$.
Initialize: $t \leftarrow 0, q_0(X) \leftarrow 1/n$ for all $X \in \mathcal{X}$, found $\leftarrow$ False (indicating whether an ancestor of X_R is found or not).
while found is False **do**
  Identify central node $v_c(t) \leftarrow$ Find Central Node$(\mathcal{T}_0, q_t(\cdot))$. //Algorithm 6 in Section B.
  Initialize the edge direction between Y and $v_c(t)$ as $Y \to v_c(t)$ by:
  direction$(Y) \leftarrow$ up, for all $Y \in N_{\mathcal{T}_0}(v_c(t))$.
  for $z \in [K]$ **do**
    Perform $a = \text{do}(v_c(t) = z)$ for B times,
    collect interventional samples $Y_1^a, \ldots, Y_B^a$ for $Y \in N_{\mathcal{T}_0}(v_c(t))$ and $R_1^a, \ldots, R_B^a$.
    Estimate interventional probabilities $\hat{P}(Y = y \mid a) \leftarrow \frac{1}{B}\sum_{b=1}^{B} \mathbb{1}_{\{Y_b^a = y\}}$ for $y \in [K]$.
    if $|\hat{R} - \frac{1}{B}\sum_{b=1}^{B} R_b^a| > \Delta/2$ **then**
      found $\leftarrow$ True. //by Assumption 3, $v_c(t)$ is an ancestor of X_R.
    end if
    for $Y \in N_{\mathcal{T}_0}(v_c(t))$ **do**
      if $|P(Y = y) - \hat{P}(Y = y \mid a)| > \varepsilon/2$ for some $y \in [K]$ **then**
        Update the edge direction between Y and $v_c(t)$ as $v_c(t) \to Y$ by:
        direction$(Y) \leftarrow$ down. //by Assumption 2, the edge direction is updated as $v_c(t) \to Y$.
      end if
    end for
  end for
  if found is True **then**
    return $\widetilde{\mathcal{T}}_0$ induced by root $v_c(t)$. //cut off upstream branches using direction$(\cdot)$ results.
  end if
  for $Y \in N_{\mathcal{T}_0}(v_c(t))$ **do**
    if direction$(Y) =$ down **then**
      $q_{t+1}(X) \leftarrow 0$, for $X \in B_{\mathcal{T}_0}^{v_c(t):Y} \cup \{v_c(t)\}$. //variables that cannot be an ancestor of X_R.
    else
      $q_{t+1}(X) \leftarrow 1$, for $X \in B_{\mathcal{T}_0}^{v_c(t):Y}$. //variables that might be an ancestor of X_R.
    end if
  end for
  normalize $q_{t+1}(\cdot)$, $t \leftarrow t + 1$.
end while

C Discussion on Multiple Reward Generating Variables

In this section, we discuss how to generalize Algorithm 1 to the setting where multiple reward generating variables exist.

In this setting, we will have to generalize Assumption 3 to hold on interventions on any ancestor of each of the direct causes. That is to say for any variable X that is an ancestor of certain direct cause of the reward, we have $|\mathbb{E}[R \mid \text{do}(X = x) - \mathbb{E}[R \mid \text{do}()]]| > \Delta$ for some x in the domain of X. Then our approach can be generalized to this setting by running multiple times. Specifically, in Algorithm 1, we don't stop after stage 2 finds a reward generating variable, say X_{R_1} because it is only one of the direct causes. By construction of stage 2 (Algorithm 4), we know except for X_{R_1} itself, none of its descendants is a direct cause of the reward (we always check children first). We can fix the values of all variables in the subtree $\mathcal{T}_1$ induced by X_{R_1} as the root. Then we re-run Algorithm 1 on the remaining graph (still a tree): original tree cut by subtree $\mathcal{T}_1$, and find a second direct cause of the reward. This procedure can be repeated until no further reward-generating variable can be found.

Above idea is a way to find multiple reward generating variables, but it's also possible to come up with more efficient ways. For example, instead of finding the direct causes one by one, it will be interesting to develop methods that can find several of them simultaneously. We think this setting itself is also interesting and worth studying as an independent work.

Algorithm 4 Find Key Node

Input: directed sub-tree $\widetilde{\mathcal{T}}_0$, K, B, Δ, $\hat{R}$.
Initialize: $t \leftarrow 0, q_0(X) \leftarrow 1/|\widetilde{\mathcal{T}}_0|$ for all $X \in V(\mathcal{T}_0)$.
while $q_t(X) > 0$ for more than one $X \in V(\widetilde{\mathcal{T}}_0)$ **do**
  Identify central node $v_c(t) \leftarrow$ Find Central Node ($\widetilde{\mathcal{T}}_0$, $q_t(\cdot)$). //Algorithm 6 in Section B.
  Initialize direction $\leftarrow$ up. //Indicating whether the true X_R is towards the upstream direction of $v_c(t)$ or downstream direction of $v_c(t)$.
  for $z \in [K]$ **do**
    Perform $a = \text{do}(v_c(t) = z)$ for B times, collect samples $R_1^a, \ldots, R_B^a$.
    if $|\hat{R} - \frac{1}{B}\sum_{b=1}^{B} R_b^a| > \Delta/2$ **then**
      direction $\leftarrow$ down. //by Assumption 3.
    end if
  end for
  if direction $=$ up **then**
    $q_{t+1}(X) \leftarrow 1$ for $X \in B_{\widetilde{\mathcal{T}}_0}^{v_c(t):\text{Pa}_{\widetilde{\mathcal{T}}_0}(v_c(t))}$. //variables that might be an ancestor of X_R.
    $q_{t+1}(v_c(t)) \leftarrow 0$. //$v_c(t)$ cannot be an ancestor of X_R if direction is up.
    for $Y \in \text{Ch}_{\widetilde{\mathcal{T}}_0}(v_c(t))$ **do**
      $q_{t+1}(X) \leftarrow 0$, for $X \in B_{\widetilde{\mathcal{T}}_0}^{v_c(t):Y}$. //variables that cannot be an ancestor of X_R.
    end for
  else
    Initialize RewardBranch $\leftarrow$ None. //indicating the branch pointing from $v_c(t)$ that contains the true X_R.
    for $Y \in \text{Ch}_{\widetilde{\mathcal{T}}_0}(v_c(t))$ **do**
      for $y \in \text{Dom}(Y)$ **do**
        Perform $a = \text{do}(Y = y)$ for B times, collect $R_1^a, \ldots, R_B^a$.
        if $|\hat{R} - \frac{1}{B}\sum_{b=1}^{B} R_b^a| > \Delta/2$ **then**
          RewardBranch $\leftarrow B_{\widetilde{\mathcal{T}}_0}^{v_c(t):Y}$. by Assumption 3.
        end if
      end for
    end for
    if RewardBranch is None. **then**
      return $v_c(t)$. //$v_c(t)$ is an ancestor of X_R but none of its children is, thus, $v_c(t)$ itself is X_R.
    end if
    $q_{t+1}(X) \leftarrow 0$ for $X \notin$ RewardBranch and $q_{t+1}(X) \leftarrow 1$ for $X \in$ RewardBranch. //only the variables in RewardBranch might be X_R.
  end if
  normalize $q_{t+1}(\cdot)$, $t \leftarrow t + 1$.
end while

Algorithm 5 CN-UCB for Causal Forest

Input: essential graph $\mathcal{E}(D)$, $K, \varepsilon, \Delta, B, T_2$, observational probabilities $P(\mathcal{X})$.
Perform do() for B times, collect $R_1, \ldots, R_B$, $\hat{R} \leftarrow \frac{1}{B}\sum_{b=1}^{B} R_b$.
for $\mathcal{T}_0$ in CC($\mathcal{E}(D)$) **do**
  Stage 1: Find a directed subtree that contains X_R: $\widetilde{\mathcal{T}}_0 \leftarrow$ Find Sub-tree($\mathcal{T}_0, K, B, \varepsilon, \Delta, \hat{R}$). //call Algorithm 3.
  if $\widetilde{\mathcal{T}}_0$ is not empty **then**
    Stage 2: Find the key node that generates the reward: $X_R \leftarrow$ Find Key Node($\widetilde{\mathcal{T}}_0, K, B, \Delta, \hat{R}$). //call Algorithm 4.
    Stage 3: Apply UCB algorithm on $\mathcal{A}_R = \{\text{do}(X_R = k) \mid k = 1, \ldots, K\}$ for T_2 rounds.
  end if
end for

Algorithm 6 Find Central Node

Input: Undirected tree $\mathcal{T}$ with some distribution q over the nodes $i = 1, \ldots, n$.

Choose a node v from $X_1, \ldots, X_n$. Find neighbors $N_{\mathcal{T}}(v)$.
while $\max_{j \in N_{\mathcal{T}}} q(B_{\mathcal{T}}^{v_c:X_j}) \geq 1/2$ **do**
  $v \leftarrow \text{argmax}_{j \in N_{\mathcal{T}}} q(B_{\mathcal{T}}^{v_c:X_j})$
end while

Output: Central node $v_c = v$.

Algorithm 7 Find Sub-junction-tree

Input: graph G, junction tree $\mathcal{T}_G, K, B, \varepsilon, \Delta, \hat{R}$.
Initialize: $t \leftarrow 0, q_0(C) \leftarrow 1/|\mathcal{T}_G|$ for all $C \in \mathcal{T}_G$, found $\leftarrow$ False (indicating whether an ancestor of X_R is found or not).
while found is False **do**
  Identify central node $C_c(t) \leftarrow$Find Central Node$(\mathcal{T}_G, q_t(\cdot))$. //call Algorithm 6.
  Initialize the edge direction between C_Y and $C_c(t)$ as $C_Y \to C_c(t)$ by: direction$(C_Y) \leftarrow$ up, for all $C_Y \in N_{\mathcal{T}_G}(C_c(t))$.
  Clique Intervention $(G, C_c(t), B)$ with collected interventional samples: $R_b^{\text{do}(Z=z)}$, $Y_b^{\text{do}(Z=z)}$ for $Z \in V(C_c(t)), z \in [K], Y \in V(G), b = 1, \ldots, B$. //call Algorithm 9.
  $\hat{P}(Y = y|\text{do}(Z = z)) \leftarrow \frac{1}{B} \sum_{b=1}^{B} \mathbb{1}_{\{Y_b^{\text{do}(Z=z)} = y\}}$ for $Z \in C_c(t), z \in [K]$, $Y \in V(G) \setminus \{Z\}, y \in [K]$.
  if $|\hat{R} - \frac{1}{B} \sum_{b=1}^{B} R_b^{\text{do}(Z=z)}| > \Delta/2$ for some $Z \in C_c(t), z \in [K]$ **then**
    found $\leftarrow$ True. //by Assumption 3.
  end if
  for $C_Y \in V(N_{\mathcal{T}_G}(C_c(t)))$ **do**
    if $\forall Z \in C_Y \cap C_c(t), \forall Y \in C_Y \setminus C_c(t), \exists z, y$, s.t. $|P(Y = y) - \hat{P}(Y = y \mid \text{do}(Z = z))| > \varepsilon/2$ **then**
      direction$(C_Y) \leftarrow$ down. //by Assumption 2 and Definition 8, the edge direction is updated as $C_c(t) \to C_Y$.
    end if
  end for
  if found is True **then**
    return sub-junction-tree $\widetilde{\mathcal{T}}_G$ induced by $\{C \mid C \notin B_{\mathcal{T}_G}^{C_c(t):C_Y}$, where direction$(C_Y) = \text{up}\}$
  end if
  for $C_Y \in N_{\mathcal{T}_G}(C_c(t))$ **do**
    if direction$(C_Y) =$ down **then**
      $q_{t+1}(X) \leftarrow 0$ for $X \in B_{\mathcal{T}_G}^{C_c(t):C_Y} \cup \{C_c(t)\}$. //cliques that cannot contain X_R.
    else
      $q_{t+1}(X) \leftarrow 1$ for $X \in B_{\mathcal{T}_G}^{C_c(t):C_Y}$. //cliques that may contain X_R.
    end if
  end for
  normalize $q_{t+1}(\cdot), t \leftarrow t + 1$.
end while

Algorithm 8 Find Key Clique

Input: graph G, directed sub-junction tree $\widetilde{\mathcal{T}}_G, K, B, \Delta, \hat{R}$.
Initialize: $t \leftarrow 0, q_0(C) \leftarrow 1/|\widetilde{\mathcal{T}}_G|$ for clique C in $\widetilde{\mathcal{T}}_G$.
while $q_t(C) > 0$ for more than one clique C in $\mathcal{T}_{G_0}$ **do**
  Identify central clique $C_c(t) \leftarrow$Find Central Node($\widetilde{\mathcal{T}}_G, q_t(\cdot)$). //call Algorithm 6.
  Initialize direction $\leftarrow$ up. //Indicating whether the true X_R is towards the upstream direction of $C_c(t)$ (not include $C_c(t)$) or downstream direction of $C_c(t)$ (include $C_c(t)$).
  Clique Intervention($G, C_c(t), B$) and collect reward data $R_1^{\text{do(Z=z)}}, \ldots, R_B^{\text{do(Z=z)}}$, for $Z \in C_c(t), z \in [K]$. //call Algorithm 9.
  if $|\hat{R} - \frac{1}{B}\sum_{b=1}^{B} R_b^{\text{do}(Z=z)}| > \Delta/2$ for some $Z \in C_c(t), z \in [K]$ **then**
    directon $\leftarrow$ down. //by Assumption 3.
  end if
  if direction $=$ up **then**
    $q_{t+1}(C) \leftarrow 0$, for clique $C \in B_{\widetilde{\mathcal{T}}_G}^{C_c(t):C_Y}$ if C_Y satisfies: $C_c(t) \to C_Y$ in $\widetilde{\mathcal{T}}_G$.
    $q_{t+1}(C) \leftarrow 1$, for the remaining cliques.
  else
    RewardBranch $\leftarrow$ None.
    for $C_Y \in \text{Ch}_{\widetilde{\mathcal{T}}_G}(C_c(t))$ **do**
      Clique Intervention(G, C_Y, B) and collect reward data $R_1^{\text{do(Y=y)}}, \ldots, R_B^{\text{do(Y=y)}}$, for $Y \in V(C_Y), y \in [K]$. //call Algorithm 9.
      if $|\hat{R} - \frac{1}{B}\sum_{b=1}^{B} R_b^{\text{do}(Y=y)}| > \Delta/2$ for some $Y \in V(C_Y), y \in [K]$ **then**
        RewardBranch $\leftarrow B_{\widetilde{\mathcal{T}}_G}^{C_c(t):C_Y}$.
        Break
      end if
    end for
    if RewardBranch is None **then**
      return $C_c(t)$. //One of the variables in $C_c(t)$ is an ancestor of X_R, but none of $C_c(t)$'s children contains an ancestor of X_R. Thus, $C_c(t)$ contains X_R.
    else
      $q_{t+1}(C) \leftarrow 1$, for $C \in$ RewardBranch. //only cliques in RewardBranch may contain X_R.
      $q_{t+1}(C) \leftarrow 0$, for $C \notin$ RewardBranch. //cliques not in RewardBranch do not contain X_R.
    end if
  end if
  normalize $q_{t+1}(\cdot)$, $t \leftarrow t+1$.
end while

Algorithm 9 Clique Intervention

Input: Graph G, Clique C, B.
for $Z \in V(C)$ **do**
  for $k = 1, \ldots, K$ **do**
    for $b = 1, \ldots, B$ **do**
      Perform intervention $\text{do}(Z = k)$.
      Collect interventional data for reward and other variables on the graph: $R_b^{\text{do}(Z=k)}$, $Y_b^{\text{do}(Z=k)}$ for $Y \in V(G)$.
    end for
  end for
end for