Graphical Models Meet Bandits: A Variational Thompson Sampling Approach

Tong Yu [1] **Branislav Kveton** [2] **Zheng Wen** [3] **Ruiyi Zhang** [4] **Ole J. Mengshoel** [1 5]

Abstract

We propose a novel framework for structured bandits, which we call an influence diagram bandit. Our framework uses a graphical model to capture complex statistical dependencies between actions, latent variables, and observations; and thus unifies and extends many existing models, such as combinatorial semi-bandits, cascading bandits, and low-rank bandits. We develop novel online learning algorithms that learn to act efficiently in our models. The key idea is to track a structured posterior distribution of model parameters, either exactly or approximately. To act, we sample model parameters from their posterior and then use the structure of the influence diagram to find the most optimistic action under the sampled parameters. We empirically evaluate our algorithms in three structured bandit problems, and show that they perform as well as or better than problem-specific state-of-the-art baselines.

1. Introduction

Structured multi-armed bandits, such as combinatorial semi-bandits (Gai et al., 2012; Chen et al., 2013; Kveton et al., 2015b), cascading bandits (Kveton et al., 2015a; Li et al., 2016), and low-rank bandits (Katariya et al., 2017; Bhargava et al., 2017; Jun et al., 2019; Lu et al., 2018; Zimmert & Seldin, 2018), have been extensively studied. Various learning algorithms have been developed, and many of them have provable regret bounds, good experimental results, or both. Despite such significant progress along this research line, the prior work still suffers from limitations.

A major limitation is that there is no unified framework or general learning algorithms for structured bandits. Most existing algorithms are tailored to a specific structured bandit, and new algorithms are necessary even when the modeling assumptions are slightly modified. For example, cascading bandits assume that if an item in a recommended list is examined but not clicked by a user, the user examines the next item in the list. `CascadeKL-UCB` algorithm for cascading bandits (Kveton et al., 2015a) relies heavily on this assumption. In practice though, there might be a small probability that the user skips some items in the list when browsing. If we take this into consideration, `CascadeKL-UCB` is no longer guaranteed to be sound and needs to be redesigned.

To enable more general algorithms, we propose a novel online learning framework of influence diagram bandits. The influence diagram (Howard & Matheson, 1984) is a graphical model, which generalizes Bayesian networks by adding decision and reward nodes. It can naturally represent structured stochastic decision problems, and elegantly model complex statistical dependencies between actions, latent variables, and observations. This paper presents a specific type of influence diagrams (Section 2), enabling many structured bandits, such as cascading bandits (Kveton et al., 2015a), combinatorial semi-bandits (Gai et al., 2012; Chen et al., 2013; Kveton et al., 2015b), and rank-1 bandits (Katariya et al., 2017), to be formulated as special cases of influence diagram bandits.

However, it is still non-trivial to efficiently learn to make the best decisions online with convergence guarantee, in an influence diagram with (i) *complex structure* and (ii) *exponentially many actions*. In this paper, we develop a Thompson sampling algorithm `idTS` and its approximations for influence diagram bandits. The key idea is to represent the model in a compact way and track a structured posterior distribution of model parameters. We sample model parameters from their posterior and then use the structure of the influence diagram to find the most optimistic action under the sampled parameters. In complex influence diagrams, latent variables naturally occur. In this case, the model posterior is intractable and exact posterior sampling is infeasible. To handle such problems, we propose variational Thompson sampling algorithms, `idTSvi` and `idTSinc`. The algorithms are both statistically and computationally efficient, as they use our compact model representation and the fact that the best action under a sampled model can be computed using dynamic programming. We further derive an upper bound on the regret of `idTS` under additional assumptions, and show that the regret only depends on the

[1]Carnegie Mellon University [2]Google Research [3]DeepMind [4]Duke University [5]Norwegian University of Science and Technology. Correspondence to: Tong Yu <worktongyu@gmail.com>.

Proceedings of the 37^{th} International Conference on Machine Learning, Online, PMLR 119, 2020.

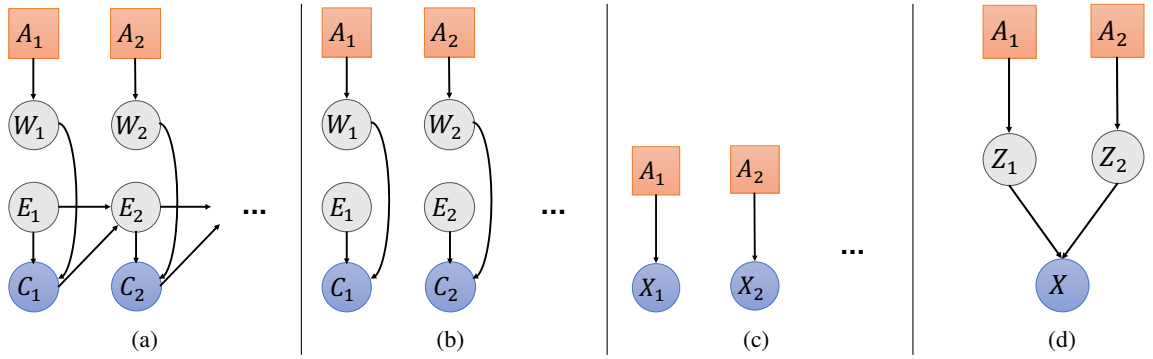


Figure 1. Examples of existing models, which are special cases of our framework: (a) cascade model, (b) position-based model, (c) semi-bandit, and (d) rank-1 bandit. See the details of the models in Section 3. The red nodes are decision nodes, the gray nodes are latent nodes, and the blue nodes are observed nodes. See formal definitions in Section 2.

model parameterization and is independent of the number of actions.

This paper makes four major contributions. First, we propose a novel framework called influence diagram bandit, which unifies and extends many existing structured bandits. Second, we develop algorithms to efficiently learn to make the best decisions under the influence diagrams with complex structures and exponentially many actions. We further derive an upper bound on the regret of our `idTS` algorithm. Third, by tracking a structured posterior distribution of model parameters, our algorithms naturally handle complex problems with latent variables. Finally, we validate our algorithms on three structured bandit problems. Empirically, the average cumulative reward of our algorithms is up to three times higher than the reward of the baseline algorithms.

2. Background

An *influence diagram* is a graphical model, which augments a Bayesian network by decision nodes and a reward function (Howard & Matheson, 1984). The structure of an influence diagram is determined by a *directed acyclic graph (DAG)* G. In our model, nodes in the influence diagram can be partially observed and are classified into three categories: *decision nodes* $A = (A_i)_{i\in\mathbb{N}}$, *latent nodes* $Z = (Z_i)_{i\in\mathbb{N}}$, and *observed nodes* $X = (X_i)_{i\in\mathbb{N}}$. Among these nodes, X are stochastic and observed, Z are stochastic and unobserved, and A are non-stochastic. Except for the decision nodes, each node corresponds to a random variable. Without loss of generality, we assume that all random variables are Bernoulli[1]. The decision nodes represent decisions that are under the full control of the decision maker. We further assume that each decision node has a single child and that each child of a decision node has a single parent. The edges in the diagram represent probabilistic dependencies between variables. The reward function $r(X, Z)$ is a deterministic function of the values of nodes X and Z. In an influence diagram, the *solution* to the decision problem is an instantiation of decision nodes that maximizes the reward.

[1]To handle non-binary categorical variables, we can extend our algorithms (Section 4) by replacing Beta with Dirichlet.

2.1. Influence Diagrams for Bandit Problems

The decision nodes affect how random variables X and Z are realized, which in turn determine the reward $r(X, Z)$. To explain this process, we introduce some notation. Let V be a node, $\mathsf{child}(V)$ be the set of its children, and $\mathsf{par}(V)$ be the set of its parents. For simplicity of exposition, each decision node A_i is a categorical variable that can take on L values $\mathcal{A} = \{1, \dots L\}$. Each value $a \in \mathcal{A}$ corresponds to a decision and is associated with a fixed Bernoulli mean $\mu_a \in [0, 1]$. The number of decision nodes is K. The *action* $a = (a_1, \dots, a_K) \in \mathcal{A}^K$ is a tuple of all K decisions.

The nodes in the influence diagram are instantiated by the following stochastic process. First, all decision nodes A_i are assigned decisions by the decision maker. Let $A_i = a_i$ for all $i \in [K]$. Then the value of each $\mathsf{child}(A_i)$ is drawn according to a_i. Specifically, it is sampled from a Bernoulli distribution with mean μ_{a_i}, $\mathsf{child}(A_i) \sim \mathrm{Ber}(\mu_{a_i})$. All remaining nodes are set as follows. For any node V that has not been set yet, if all nodes in $\mathsf{par}(V)$ have been instantiated, $V \sim \mathrm{Ber}(\mu)$, where $\mu \in [0, 1]$ depends only on the assigned values to $\mathsf{par}(V)$. Since the influence diagram is a DAG, this process is well defined and guaranteed to instantiate all nodes. The corresponding reward is $r(X, Z)$.

The model can be parameterized by a vector of Bernoulli means $\theta \in [0, 1]^{d+L}$. The last L entries of θ correspond to μ_a, one for each decision $a \in \mathcal{A}$. The first d entries parameterize the conditional distributions of all nodes that are not in $\cup_{i\in[K]}\mathsf{child}(A_i)$, directly affected by the decision nodes. We denote the joint probability distribution of X and Z conditioned on action $a \in \mathcal{A}^K$ by $P(X, Z \mid \theta, a)$.

2.2. Simple Example

We show a simple influence diagram in Figure 1d. The decisions nodes are $A = (A_1, A_2)$, the observed node is X, and the latent nodes are $Z = (Z_1, Z_2)$. After the decision node A_i is set to a_i, it determines the mean of Z_i, μ_{a_i}. Then, after both $Z_i \sim \mathrm{Ber}(\mu_{a_i})$ are drawn, X is drawn from a Bernoulli distribution with mean $P(X = 1 \mid Z_1, Z_2)$.

Since the number of decisions is L, this model has $L + 4$ parameters. The L parameters are $(\mu_a)_{a \in \mathcal{A}}$. The remaining 4 parameters are $P(X = 1 \mid Z_1, Z_2)$ for $Z_1, Z_2 \in \{0, 1\}$.

3. Influence Diagram Bandits

Consider an influence diagram with observed nodes X, latent nodes Z, and decision nodes A in Section 2. Recall that all random variables associated with these nodes are binary, and the model is parameterized by a vector of $d + L$ Bernoulli means $\theta_* \in [0, 1]^{d+L}$. The learning agent knows the structure of the influence diagram, but does not know the marginal and conditional distributions in it. That is, the agent does not know θ_*. Let $r(x, z)$ be the reward associated with $X = x$ and $Z = z$. To simplify exposition, let $r(a, \theta)$ be the expected reward of action a under model parameters $\theta \in [0, 1]^{d+L}$,

$$r(a, \theta) = \sum_{x,z} r(x, z) P(x, z \mid \theta, a).$$

At time t, the agent adaptively chooses action a_t based on the past observations. Then the binary values of the children of decision nodes A are generated from their respective Bernoulli distributions, which are specified by a_t. Analogously, the values of all other nodes are generated based on their respective marginal and conditional distributions. Let x_t and z_t denote the values of X and Z, respectively, at time t. At the end of time t, the agent observes x_t and receives stochastic reward $r(x_t, z_t)$.

The agent's policy is evaluated by its n-step expected cumulative regret

$$R(n, \theta_*) = \mathbb{E}\left[\sum_{t=1}^{n} R(a_t, \theta_*) | \theta_*\right],$$

where $R(a_t, \theta_*) = r(a^*, \theta_*) - r(a_t, \theta_*)$ is the instantaneous regret at time t, and

$$a^* = \arg\max_{a \in \mathcal{A}^K} r(a, \theta_*)$$

is the optimal action under true model parameters θ_*. For simplicity of exposition, we assume that a^* is unique. When we have a prior over θ_*, an alternative performance metric is the n-step Bayes regret, which is defined as

$$R_B(n) = \mathbb{E}\left[R(n, \theta_*)\right],$$

where the expectation is over θ_* under its prior.

We review several examples of prior works, which can be viewed as special cases of influence diagram bandits, below.

3.1. Online Learning to Rank in Click Models

The *cascade model* is a popular model in *learning to rank* (Chuklin et al., 2015), which has been studied extensively in the bandit setting, starting with Kveton et al. (2015a). The model describes how a user interacts with a list of items $a = (a_1, \ldots, a_K)$ at K positions. We visualize it in Figure 1a. For each position $k \in [K]$, the model has four nodes: decision node A_k, which is set to the item at position k, a_k; latent attraction node W_k, which is the attraction indicator of item a_k; latent examination node E_k, which indicates that position k is examined; and observed click node C_k, which indicates that item a_k is clicked. The attraction of item a_k is a Bernoulli random variable with mean μ_{a_k}. Therefore, $P(W_k = 1 \mid A_k = a_k) = \mu_{a_k}$, as in our model. The rest of the dynamics, that the item is clicked only if it is attractive and its position is examined, and that the examination of items stops upon a click, is encoded as

$$P(C_k = 1 \mid W_k, E_k) = W_k E_k\,,$$
$$P(E_k = 1 \mid C_{k-1}, E_{k-1}) = (1 - C_{k-1}) E_{k-1}\,.$$

The first position is always examined, and therefore we have $P(E_1 = 1) = 1$. If at least one item in the list is clicked, the reward is 1. Otherwise, the reward is 0. That is, $r(X, Z) = 1 - \prod_{k=1}^{K} (1 - C_k)$.

The *position-based model* (Chuklin et al., 2015) in Figure 1b is another popular click model, which was studied in the bandit setting by Lagrée et al. (2016). The difference from the cascade model is that the examination indicator of position k, E_k, is an independent random variable. Besides, in the position-based model, the reward is the total number of clicks, $r(X, Z) = \sum_{k=1}^{K} C_k$.

3.2. Combinatorial Semi-Bandits

The third example is a combinatorial semi-bandit (Gai et al., 2012; Chen et al., 2013; Kveton et al., 2015b; Wen et al., 2015). In this model, the agent chooses K items $a = (a_1, \ldots, a_K)$ and observes their rewards. We visualize this model in Figure 1c. For the k-th item, the model has two nodes: decision node A_k, which is set to the k-th chosen item a_k; and observed reward node X_k, which is the reward of item a_k. If the reward of item a_k is a Bernoulli random variable with mean μ_{a_k}, $P(X_k = 1 \mid A_k = a_k) = \mu_{a_k}$, as in our model. The reward is the sum of individual item rewards, $r(X, Z) = \sum_{k=1}^{K} X_k$.

3.3. Bernoulli Rank-1 Bandits

The last example is a Bernoulli rank-1 bandit (Katariya et al., 2017). In this model, the agent selects the row and

column of a rank-1 matrix, and observes a stochastic reward of the product of their latent factors. We visualize this model in Figure 1d and can represent it as follows. We have two decision nodes, A_1 for rows and A_2 for columns. The values of these nodes, a_1 and a_2, are the chosen rows and columns, respectively. We have two latent nodes, Z_1 for rows and Z_2 for columns. For each, $P(Z_k = 1 \mid A_k = a_k) = \mu_{a_k}$, where μ_{a_k} is the latent factor corresponding to choice a_k. Finally, we have one observed reward node X such that $P(X = 1 \mid Z_1, Z_2) = Z_1 Z_2$. The reward is $r(X, Z) = X$.

4. Algorithm

There are two major challenges in developing efficient online learning algorithm for influence diagram bandits. First, with exponentially many actions, it is challenging to develop a sample efficient algorithm to learn a generalizable model statistically efficiently. Second, it is computationally expensive to compute the most valuable action when instantiating the decision nodes in each step, given exponentially many combinations of options.

Many exploration strategies exist in the online setting, such as the ϵ-greedy policy (Sutton & Barto, 2018), `UCB1` (Auer et al., 2002), and Thompson sampling (Thompson, 1933). While we do not preclude that UCB-like algorithms can be developed for our problem, we argue that they are unnatural. Roughly speaking, the upper confidence bound (UCB) is the highest value of any action under any plausible model parameters, given the history. It is unclear how to solve this problem efficiently in influence diagrams. In contrast, Thompson sampling is more natural. The model parameters can be sampled from their posterior, and the problem of finding the most valuable action given fixed model parameters can be solved using dynamic programming (Tatman & Shachter, 1990).

4.1. Thompson Sampling

Thompson sampling (Thompson, 1933) is a popular online learning algorithm, which we adapt to our setting as follows. Let $\tilde{x} = (x_\ell)_{\ell=1}^{t-1}$, $\tilde{z} = (z_\ell)_{\ell=1}^{t-1}$, and $\tilde{a} = (a_\ell)_{\ell=1}^{t-1}$ be the values of observed nodes, latent nodes, and decision nodes, respectively, up to the end of time $t-1$. First, the algorithm samples model parameters $\theta_t \sim p_{t-1}$, where p_{t-1} is the posterior of θ_* at the end of time $t-1$. That is,

$$p_{t-1}(\theta) = P(\theta_* = \theta \mid \tilde{x}, \tilde{a})$$

for all θ. Second, the action at time t is chosen greedily with respect to θ_t,

$$a_t = \arg\max_{a \in \mathcal{A}^\kappa} r(a, \theta_t).$$

Finally, the agent observes x_t and receives reward $r(x_t, z_t)$. We call this algorithm `idTS`. Note that `idTS` is generally computationally intractable when latent nodes are present, due to the need to sample from the exact posterior.

4.2. Fully-Observable Case

In the fully-observable case, with no latent variables and Beta prior, `idTS` is computationally efficient. To see it, note that in this case p_{t-1} factors over model parameters, and is a product of Beta distributions. Based on the observed node values, we can update the Beta posterior for each model parameter in θ_* individually and computationally efficiently, since the Beta distribution is the conjugate prior of the Bernoulli distribution. One example of this case is the combinatorial semi-bandit in Section 3.2.

4.3. Partially-Observable Case

In complex influence diagram bandits, such as rank-1 bandits, latent variables are typically present. In such cases, exact sampling from p_{t-1} is usually computationally intractable, which limits the use of `idTS`. However, there are many computationally tractable approximations to sampling from p_{t-1}, such as variational inference and particle filtering (Bishop, 2006; Andrieu et al., 2003). In practice, the performance of particle filtering heavily depends on different settings of the algorithm (*e.g.*, number of particles and transition prior). Thus, we develop an approximate Thompson sampling algorithm, `idTSvi`, based on variational inference. We compare our algorithms to particle filtering later in Section 7.4.

To keep notation uncluttered, we omit $\tilde{a}$ below. Let $q(\tilde{z}, \theta)$ be a factored distribution over $(\tilde{z}, \theta)$. To approximate the posterior p_{t-1}, we approximate $P(\tilde{z}, \theta \mid \tilde{x})$ by minimizing its difference to $q(\tilde{z}, \theta)$. To achieve this, we decompose the following log marginal probability using

$$\begin{aligned}
\log P(\tilde{x}) &= \int_\theta \sum_{\tilde{z}} q(\tilde{z}, \theta) \log P(\tilde{x}) d\theta \\
&= \int_\theta \sum_{\tilde{z}} q(\tilde{z}, \theta) \log \frac{P(\tilde{z}, \theta \mid \tilde{x}) P(\tilde{x}) q(\tilde{z}, \theta)}{P(\tilde{z}, \theta \mid \tilde{x}) q(\tilde{z}, \theta)} d\theta \\
&= \int_\theta \sum_{\tilde{z}} q(\tilde{z}, \theta) \log \frac{P(\tilde{x}, \tilde{z}, \theta)}{q(\tilde{z}, \theta)} d\theta \\
&\quad + \int_\theta \sum_{\tilde{z}} q(\tilde{z}, \theta) \log \frac{q(\tilde{z}, \theta)}{P(\tilde{z}, \theta \mid \tilde{x})} d\theta.
\end{aligned}$$

We denote the first term by $\mathcal{L}(q)$. The second term is the KL divergence between $P(\tilde{z}, \theta \mid \tilde{x})$ and $q(\tilde{z}, \theta)$. Note $P(\tilde{x})$ is fixed and the KL divergence is non-negative. Therefore, we can minimize the KL divergence by maximizing $\mathcal{L}(q)$ with respect to q, to achieve better posterior approximations.

We maximize $\mathcal{L}(q)$ as follows. By the mean field approxi-

mation, let the approximate posterior factor as

$$q(\tilde{z}, \theta) = q(\theta)\Pi_{\ell=1}^{t-1} q_\ell(z_\ell), \tag{1}$$

where $q(\theta)$ is the probability of model parameters θ and $q_\ell(z_\ell)$ is the probability that the values of latent nodes at time ℓ are z_ℓ. Since θ_i is the mean of a Bernoulli variable, we factor $q(\theta)$ as $\Pi_{i=1}^{d+L}\theta_i^{\alpha_i}(1-\theta_i)^{\beta_i}$ and represent it as $d+L$ tuples $\{(\alpha_i, \beta_i)\}_{i=1}^{d+L}$. For any ℓ, $q_\ell(z_\ell)$ is a categorical distribution. From the definition of $P(x, z, \theta)$, we have

$$\log P(\tilde{x}, \tilde{z}, \theta) = \log P(\theta) + \sum_{\ell=1}^{t-1} \log P(x_\ell, z_\ell \mid \theta). \tag{2}$$

By combining (1) and (2) with the definition of $\mathcal{L}(q)$, we get that

$$\begin{aligned}
\mathcal{L}(q) =& \int_\theta \sum_{\tilde{z}} q(\tilde{z}, \theta) \log P(\tilde{x}, \tilde{z}, \theta) d\theta \\
&- \int_\theta \sum_{\tilde{z}} q(\tilde{z}, \theta) \log q(\tilde{z}, \theta) d\theta \\
=& \sum_{\ell=1}^{t-1} \int_\theta q(\theta) \sum_{z_\ell} q_\ell(z_\ell) \log P(x_\ell, z_\ell \mid \theta) d\theta \\
&+ \int_\theta q(\theta) \log P(\theta) d\theta - \sum_{\ell=1}^{t-1} \sum_{z_\ell} q_\ell(z_\ell) \log q_\ell(z_\ell) \\
&- \int_\theta q(\theta) \log q(\theta) d\theta
\end{aligned} \tag{3}$$

The above decomposition suggests the following EM-like algorithm (Dempster et al., 1977).

First, in the E-step, we fix $q(\theta)$. Then

$$\begin{aligned}
\mathcal{L}(q) =& \sum_{\ell}^{t-1} \sum_{z_\ell} q_\ell(z_\ell) \int_\theta q(\theta) \log P(x_\ell, z_\ell \mid \theta) d\theta \\
&- \sum_{\ell}^{t-1} \sum_{z_\ell} q_\ell(z_\ell) \log q_\ell(z_\ell) + C
\end{aligned}$$

for some constant C. By taking its derivative with respect to $q_\ell(z_\ell)$ and setting it to zero, $\mathcal{L}(q)$ is maximized by

$$q_\ell(z_\ell) \propto \exp\left[\int_\theta q(\theta) \log P(x_\ell, z_\ell \mid \theta) d\theta\right]. \tag{4}$$

Second, in the M-step, we fix all $q_\ell(z_\ell)$. Then, for some constant C,

$$\begin{aligned}
\mathcal{L}(q) =& \int_\theta q(\theta) \sum_{\ell=1}^{t-1} \sum_{z_\ell} q_\ell(z_\ell) \log P(x_\ell, z_\ell \mid \theta) d\theta \\
&+ \int_\theta q(\theta) \log P(\theta) d\theta - \int_\theta q(\theta) \log q(\theta) d\theta + C.
\end{aligned}$$

Algorithm 1 `idTSvi`: Influence diagram TS with variational inference.

1: **Input:** $\epsilon > 0$
2: Randomly initialize q
3: **for** $t = 1, \ldots, n$ **do**
4: Sample θ_t proportionally to $q(\theta_t)$
5: Take action $a_t = \arg\max_{a \in \mathcal{A}^\kappa} r(a, \theta_t)$
6: Observes x_t and receive reward $r(x_t, z_t)$
7: Randomly initialize q
8: Calculate $\mathcal{L}(q)$ using (3) and set $\mathcal{L}'(q) = -\infty$
9: **while** $\mathcal{L}(q) - \mathcal{L}'(q) \geq \epsilon$ **do**
10: Set $\mathcal{L}'(q) = \mathcal{L}(q)$
11: **for** $\ell = 1, \ldots, t$ **do**
12: Update $q_\ell(z_\ell)$ using (4), for all z_ℓ
13: **end for**
14: Update $q(\theta)$ using (5)
15: Update $\mathcal{L}(q)$ using (3)
16: **end while**
17: **end for**

By taking its derivative with respect to $q(\theta)$ and setting it to zero, $\mathcal{L}(q)$ is maximized by

$$q(\theta) \propto \exp\left[\log P(\theta) + \sum_{\ell=1}^{t-1} \sum_{z_\ell} q_\ell(z_\ell) \log P(x_\ell, z_\ell \mid \theta)\right]. \tag{5}$$

The above two steps, the E-step and M-step, are alternated until convergence. This is guaranteed under our model assumption.

The pseudocode of `idTSvi` is in Algorithm 1. In line 4, θ_t is sampled from its posterior. In line 5, the agent takes action a_t that maximizes the expected reward under model parameters θ_t. In line 6, the agent observes x_t and receives reward. From line 7 to line 16, we update the posterior of θ, by alternating the estimations of $q_\ell(z_\ell)$ and $q(\theta)$ until convergence. In line 12, we update $q_\ell(z_\ell)$ by the E-step. In line 14, we update $q(\theta)$ by the M-step.

4.4. Improving Computational Efficiency

To make `idTSvi` computationally efficient, we propose its incremental variant, `idTSinc`. The main problem in Algorithm 1 is that each $q_t(z_t)$ is re-estimated at all times from time t to time n. To reduce the computational complexity of Algorithm 1, we estimate $q_t(z_t)$ only at time t. That is, the "for" loop in line 11 is only run for $\ell = t$; and $q_\ell(z_\ell)$ for $\ell < t$ are reused from the past. The pseudocode of `idTSinc` is in Appendix C.

5. Regret Bound

In this section, we derive an upper bound on the n-step Bayes regret $R_B(n)$ for `idTS` in influence diagram bandits.

We introduce notations before proceeding. We say a node in an influence diagram bandit is *de facto* observed at time t if its realization is either observed or can be exactly inferred[2] at that time. Let $\theta_*^{(i)}$ denote the i-th element of θ_*, which parameterizes a (conditional) Bernoulli distribution at node j_i in the influence diagram. Note that if node j_i has parents $\mathsf{par}(j_i)$, then $\theta_*^{(i)}$ corresponds to a particular realization at nodes $\mathsf{par}(j_i)$. We define the event $E_t^{(i)}$ as the event that (1) both node j_i and its parents $\mathsf{par}(j_i)$ (if any) are *de facto* observed at time t, (2) the realization at $\mathsf{par}(j_i)$ corresponds to $\theta_*^{(i)}$, and (3) the realization at j_i is conditionally independently drawn from $\mathrm{Ber}\left(\theta_*^{(i)}\right)$. Note that under event $E_t^{(i)}$, the agent observes and knows that it observes a Bernoulli sample corresponding to $\theta_*^{(i)}$ at time t. To simplify exposition, if clear from context, we omit the subscript t of $E_t^{(i)}$.

Our assumptions are stated below.

Assumption 1 *The parameter index set $\{1, \ldots, d+L\}$ is partitioned into two disjoint subsets $\mathcal{I}^+$ and $\mathcal{I}^-$. For all $i \in \mathcal{I}^+$ (or $i \in \mathcal{I}^-$), $r(a, \theta)$ is weakly increasing (or weakly decreasing) in $\theta^{(i)}$ for any action a and probability measure $\theta \in [0,1]^{d+L}$.*

Assumption 2 *For any action a and any Bernoulli probability measures $\theta_1, \theta_2 \in [0,1]^{d+L}$, we have*

$$|r(a, \theta_1) - r(a, \theta_2)| \leq C \sum_{i=1}^{d+L} P(E^{(i)} | \theta_2, a) \left| \theta_1^{(i)} - \theta_2^{(i)} \right|,$$

where $C \geq 0$ is a constant.

Assumption 1 says that $r(a, \theta)$ is element-wise monotone (either weakly increasing or decreasing) in θ. Assumption 2 says that the expected reward difference is bounded by a weighted sum of the probability measure difference, with weights proportional to the observation probabilities. Note that the satisfaction of Assumption 2 depends on both the functional form of $r(X, Z)$ and the information feedback structure in the influence diagram.

Intuitively, both combinatorial semi-bandits and cascading bandits discussed in Section 3 satisfy Assumptions 1 and 2. The expected reward increases with θ in both models. Thus Assumption 1 holds. Assumption 2 holds in combinatorial semi-bandits, since all nodes are observed and the expected reward is linear in θ. Assumption 2 holds in cascading bandits due to Lemma 1 in Kveton et al. (2015a). The proof is in Appendix A.

Before we present our regret bound, we define the metric of *maximum expected observations*

$$O_{\max} = \max_a \mathbb{E}\left[\sum_{i=1}^{d+L} \mathbf{1}\left[E^{(i)}\right] \middle| a \right], \tag{6}$$

where the expectation is over both θ^* under the prior, and the random samples in the influence diagram under θ^* and a. Notice that for any action a, $\mathbb{E}\left[\sum_{i=1}^{d+L} \mathbf{1}\left[E^{(i)}\right] \middle| a \right]$ is the expected number of observations under a; hence, $O_{\max}$ is the maximum expected number of observations over all actions. Let $|X|$ and $|Z|$ be the number of observed nodes and latent nodes, respectively. Notice that by definition, $O_{\max} \leq |X| + |Z|$. Moreover, $O_{\max} \leq |X|$ if no latent variables are exactly inferred[3]; and $O_{\max} = |X|$ in the fully observable case.

Our main result is stated below.

Theorem 1 *Under Assumptions 1 and 2, if we apply exact Thompson sampling `idTS` to influence diagram bandits,*

$$R_B(n) \leq \mathcal{O}\left(C \sqrt{(L+d) O_{\max} n \log n} \right),$$

where $O_{\max}$ is defined in (6).

Please refer to Appendix B for the proof of Theorem 1. It is natural for the regret bound to be $\mathcal{O}(\sqrt{O_{\max}})$. One can see this by considering special cases. For example, in combinatorial semi-bandits, $O_{\max} = K$ and Kveton et al. (2015b) proved a $\mathcal{O}(\sqrt{K})$ regret bound. Intuitively, the $\mathcal{O}(\sqrt{K})$ term reflects the total reward magnitude in combinatorial semi-bandits.

We conclude this section by comparing our regret bound with existing bounds in special cases. In cascading bandits, we have $d = 0$ [4] and $C = 1$. Thus our regret bound is $\mathcal{O}(\sqrt{L O_{\max} n \log n})$. On the other hand, the regret bound in Kveton et al. (2015a) is $\mathcal{O}(\sqrt{LKn \log n})$. Since $O_{\max} \leq K$, our regret bound is at most $\mathcal{O}(\sqrt{\log n})$ larger. In combinatorial semi-bandits, we have $d = 0$, $O_{\max} = K$, and $C = 1$ (see Appendix A). Thus, our regret bound reduces to $R_B(n) \leq \mathcal{O}\left(\sqrt{LKn \log n}\right)$. On the other hand, Theorem 6 of Kveton et al. (2015b) derives a $\mathcal{O}\left(\sqrt{LKn \log n}\right)$ worst-case regret bound for a UCB-like algorithm for combinatorial semi-bandits. Our regret bound is only $\mathcal{O}\left(\sqrt{\log n}\right)$ larger.

[2] Exact inference is often possible when some conditional distributions are known and deterministic. For instance, in the cascade model (Section 3.1), the attractions of items above the click position can be exactly inferred.

[3] Note that a parent of an observed node might not be observed, thus in general $O_{\max} \leq |X|$.

[4] As is in Kveton et al. (2015a), we assume that the deterministic conditional distributions in cascading bandits are known to the learning agent, thus $d = 0$.

6. Related Work

In general, it is challenging to calculate the exact posterior distribution in Thompson sampling in complex problems. Urteaga & Wiggins (2018) used variational inference to approximate the posterior of arms by a mixture of Gaussians. The main difference in our work is that we focus on structured arms and rewards, where the rewards are correlated through latent variables. Theoretical analysis of approximate inference in Thompson sampling was done in Phan et al. (2019). By matching the minimal exploration rates of sub-optimal arms, Combes et al. (2017) solved a different class of structured bandit problems.

Gopalan et al. (2014) and Kawale et al. (2015) used particle filtering to approximate the posterior in Thompson sampling. Particle filtering is known to be consistent. However, in practice, its performance depends heavily on the number of particles. When the number of particles is small, particle filtering is computationally efficient but achieves poor approximation results. This issue can be alleviated by particle-based Bayesian sampling (Zhang et al., 2020). On the other hand, when the number of particles is large, particle filtering performs well but is computationally demanding.

Blundell et al. (2015) used variational inference to approximate the posterior in neural networks and incorporated it in Thompson sampling (Blundell et al., 2015). Haarnoja et al. (2017; 2018) learned an energy-based policy determined by rewards, which is approximated by minimizing the KL divergence between the optimal posterior and variational distribution. The intractable posterior of Thompson sampling in neural networks can be approximated in the last layer, which is treated as features in Bayesian linear regression (Riquelme et al., 2018; Liu et al., 2018). However, the uncertainty may be underestimated in bandit problems (Zhang et al., 2019) or zero uncertainty is propagated via Bellman error (Osband et al., 2018). This can be alleviated by particle-based Thompson sampling (Lu & Van Roy, 2017; Zhang et al., 2019). Follow-the-perturbed-leader exploration (Kveton et al., 2019c;a;b; Vaswani et al., 2020; Kveton et al., 2020) is an alternative to Thompson sampling that does not require posterior.

7. Experiments

In this section, we show that our algorithms can be applied beyond traditional models and *learn more general models*. Specifically, we compare our approaches to traditional bandit algorithms for the cascade model, position-based model, and rank-1 bandit. The performance of the algorithms is measured by their average cumulative reward. The *average cumulative reward* at time n is $\frac{1}{n}\sum_{t=1}^{n} r(x_t, z_t)$, where $r(x_t, z_t)$ is the reward received at time t. The rewards of various models are defined in Sections 3.1 to 3.3. We report the average results over 20 runs with standard errors.

Note that `idTSvi` and `idTSinc` are expected to perform worse than `idTS` since they are approximations. We now briefly discuss how to justify that `idTSvi` and `idTSinc` perform similarly to `idTS`. Recall that due to latent variables, it is computationally intractable to implement the exact Thompson sampling `idTS` in general influence diagram bandits. However, we can efficiently compute an upper bound on the average cumulative reward of `idTS` based on numerical experiments in a feedback-relaxed setting. Specifically, consider a feedback-relaxed setting where all latent nodes Z are observed. Note that `idTS` is computationally efficient in this new setting since it is fully observed. Moreover, due to more information feedback, `idTS` should perform better in this feedback-relaxed setting and hence provide an upper bound on the average cumulative reward of `idTS` in the original problem. In this section, we refer to `idTS` in this feedback-relaxed setting as `idTSfull`, to distinguish it from `idTS` in the original problem. Intuitively, if `idTSfull` and `idTSvi` perform similarly, we also expect `idTSvi` and `idTS` to perform similarly.

To speed up `idTSvi` and `idTSinc` in our experiments, we do not run the "while" loop in line 9 of Algorithm 1 until convergence. In `idTSvi`, we run it only once. In `idTSinc`, which is less stable in estimating $q_t(z_t)$ but more computationally efficient, we run it up to 30 times. We did not observe any significant drop in the quality of `idTSvi` and `idTSinc` policies when we used this approximation.

7.1. Cascade Model

Algorithms for the cascade model, such as `CascadeKL-UCB`, make strong assumptions. On the other hand, our algorithms make fewer assumptions and allow us to learn more general models. First, we experiment with the cascade model (Section 3.1). The number of items is $L = 20$ and the length of the list is $K = 2$. The attraction probability of item i is $\theta_i = i/20$. We refer to this problem as *cascade model* 1. Second, we experiment with a variant of the problem where the cascade assumption is violated, which we call *cascade model 2*. In cascade model 2, we modify the conditional dependencies of C_k and E_k as

$$P(C_k = 1 \mid W_k, E_k) = (1 - W_k)E_k\,,$$
$$P(E_k = 1 \mid C_{k-1}, E_{k-1}) = C_{k-1}E_{k-1}\,.$$

This means that an item is clicked only if it is not attractive and its position is examined, and an item is examined only if the previous item is examined and clicked.

Our results are reported in Figures 2a and 2b. We observe several trends. First, among all our algorithms, `idTSfull` and `idTSvi` perform the best. `idTSinc` performs the worst

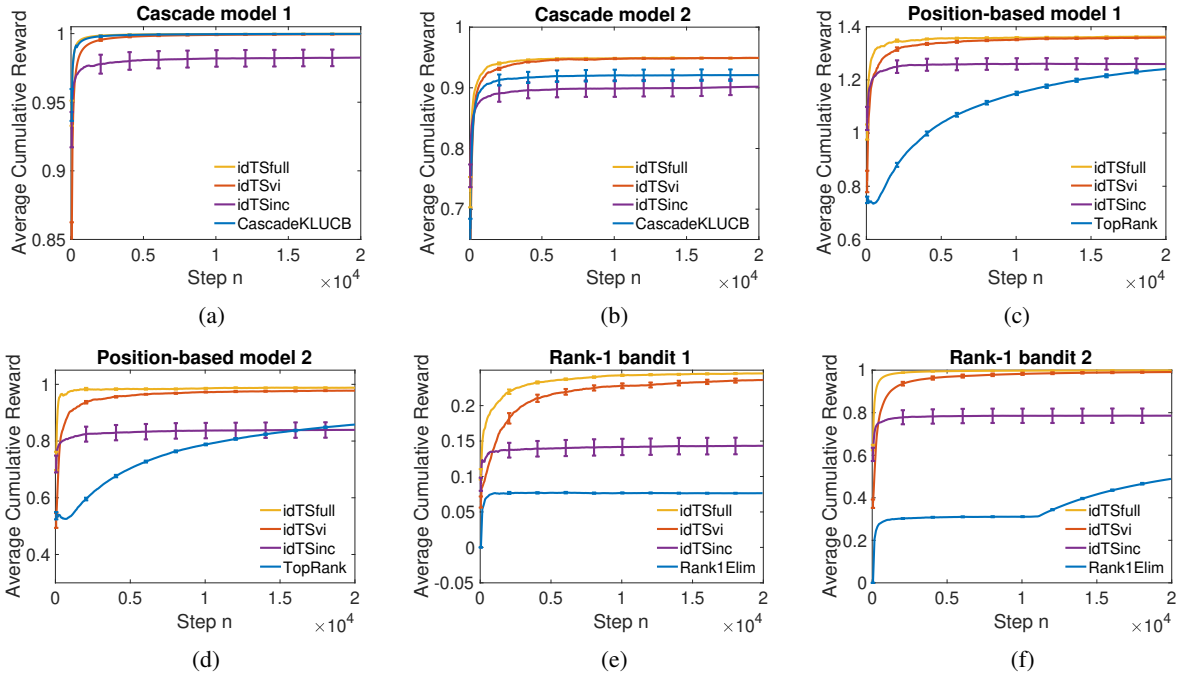


Figure 2. A comparison between different bandit algorithms in various structured bandit problems.

and its runs have a lot of variance, which is caused by an inaccurate estimation of $q(z)$. Second, `idTSinc` is much more computationally efficient than `idTSvi`. For each run in cascade model 1, `idTSvi` needs 1990.502 seconds on average, while its incremental version `idTSinc` only needs 212.772 seconds. Third, in Figure 2b, we observe a clear gap between `CascadeKL-UCB` and our algorithms, `idTSfull` and `idTSvi`. This is because the cascade assumption is violated, and thus `CascadeKL-UCB` cannot effectively leverage the problem structure. In contrast, our algorithms are general and still effectively identify the best action.

7.2. Position-Based Model

We also experiment with the position-based model, which is detailed in Section 3.1. The number of items is $L = 20$ and the length of the list is $K = 2$. The attraction probability of item i is $\theta_i = i/20$. We consider two variants of the problem. In *position-based model 1*, the examination probabilities of both positions are 0.7. In *position-based model 2*, the examination probabilities of the two positions are 0.8 and 0.2. The baseline is a state-of-the-art bandit algorithm for the position-based model, `TopRank` (Lattimore et al., 2018). Our results are reported in Figures 2c and 2d. We observe that `idTSfull` and `idTSvi` clearly outperform `TopRank`, while `idTSinc` achieves similar rewards compared to `TopRank`.

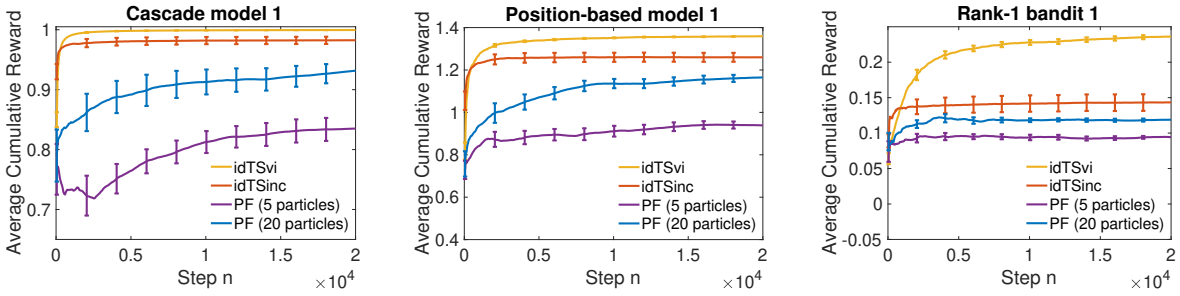
7.3. Rank-1 Bandit

We also evaluate our algorithms in a rank-1 bandit, which is detailed in Section 3.3. The underlying rank-1 matrix is UV^T, where $U \in [0, 1]^8$ and $V \in [0, 1]^{10}$. We consider two problems. In *rank-1 bandit 1*, $U_i = i/16$ and $V_i = i/20$. In *rank-1 bandit 2*, $U_i = i/8$ and $V_i = i/10$. The baseline is a state-of-the-art algorithm for the rank-1 bandit, `Rank1Elim` (Katariya et al., 2017). As shown in Figures 2e and 2f, our algorithms outperform `Rank1Elim`. In Figure 2f, `Rank1Elim` improves significantly after 10000 steps. The reason is that `Rank1Elim` is an elimination algorithm, and thus improves sharply at the end of each elimination stage. Nevertheless, a clear gap remains between our algorithms and `Rank1Elim`. The average cumulative reward of our algorithms is up to three times higher than that of `Rank1Elim`.

7.4. Comparison to Particle Filtering

In this section, we compare variational inference to particle filtering for posterior sampling in influence diagram bandits. As discussed in Section 4.3, our algorithms are based on variational inference, considering that the performance of particle filtering depends on hard to tune parameters, such as the number of particles and transition prior. We validate the advantages of variational inference in cascade model 1, position-based model 1, and rank-1 bandit 1.

We implement particle filtering as in Andrieu et al. (2003). Let m be the number of particles. At each time, the algorithm works as follows. First, we obtain m models (parti-



(a) (b) (c)

Figure 3. A comparison between variational inference and particle filtering (PF) with different numbers of particles, when approximating the posterior of Thompson sampling in influence diagram bandits.

cles) by sampling them from their Gaussian transition prior. Second, we set their importance weights based on their likelihoods. Similarly to `idTSfull`, we assume that the latent variables are observed. This simplifies the computation of the likelihood, and gives particle filtering an unfair advantage over `idTSvi` and `idTSinc`. Third, we choose the model with the highest weight, take the best action under that model, and observe its reward. Finally, we resample m particles proportionally to their importance weights. The new particles serve as the mean values of the Gaussian transition prior at the next time.

Our results are reported in Figure 3. First, we observe that variational inference based algorithms, `idTSvi` and `idTSinc`, perform clearly better than particle filtering. The performance of particle filtering is unstable and has a higher variance. Second, the performance of particle filtering is sensitive to the number of particles used. By increasing the number from 5 to 20, the performance of particle filtering improves. Third, the performance of particle filtering may drop over time, because sometimes particles with low likelihoods are sampled, although with a small probability.

To further show the advantage of our algorithms, we compare the computational cost of variational inference and particle filtering in Table 1. This experiment is in cascade model 1, and we report the run time (in seconds) and average cumulative reward at 20000 steps. The run time is measured on a computer with one 2.9 GHz Intel Core i7 processor and 16 GB memory. We observe that `idTSvi` has the longest run time, while `idTSinc` is much faster. As we increase the number of particles, the run time of particle filtering increases, since we need to evaluate the likelihood of more particles. At the same run time, `idTSinc` achieves a higher reward than particle filtering.

	Run time	Reward
`idTSvi`	1990.502 ± 19.702	0.999 ± 0.001
`idTSinc`	212.772 ± 2.204	0.983 ± 0.006
PF (5 particles)	117.856 ± 1.049	0.835 ± 0.019
PF (10 particles)	246.291 ± 3.327	0.878 ± 0.015
PF (15 particles)	357.284 ± 2.529	0.926 ± 0.013
PF (20 particles)	454.844 ± 3.054	0.932 ± 0.015

Table 1. The comparison of variational inference and particle filtering (PF) in cascade model 1. We report the run time (in seconds) and average cumulative reward at 20000 steps. The results are averaged over 20 runs.

8. Conclusions

Existing algorithms for structured bandits are tailored to specific models, and thus hard to extend. This paper overcomes this limitation by proposing a novel online learning framework of influence diagram bandits, which unifies and extends most existing structured bandits. We also develop efficient algorithms that learn to make the best decisions under influence diagrams with complex structures, latent variables, and exponentially many actions. We further derive an upper bound on the regret of our algorithm `idTS`. Finally, we conduct experiments with various structured bandits: cascading bandits, online learning to rank in the position-based model, and rank-1 bandits. The experiments demonstrate that our algorithms perform well and are general.

As discussed earlier, this paper focuses on influence diagram bandits with Bernoulli random variables. We believe that the Bernoulli assumption is without loss of generality, in the sense that by appropriately modifying some technical assumptions, our developed algorithms and analysis can be extended to more general cases with categorical or continuous random variables. We leave the rigorous derivations in these more general cases to future work.

References

Andrieu, C., De Freitas, N., Doucet, A., and Jordan, M. I. An introduction to MCMC for machine learning. *Machine Learning*, 50(1-2):5–43, 2003.

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

Bhargava, A., Ganti, R., and Nowak, R. Active positive semidefinite matrix completion: Algorithms, theory and applications. In *Artificial Intelligence and Statistics*, pp. 1349–1357, 2017.

Bishop, C. M. *Pattern Recognition and Machine Learning*. springer, 2006.

Blundell, C., Cornebise, J., Kavukcuoglu, K., and Wierstra, D. Weight uncertainty in neural network. In *International Conference on Machine Learning*, pp. 1613–1622, 2015.

Chen, W., Wang, Y., and Yuan, Y. Combinatorial multi-armed bandit: General framework, results and applications. In *Proceedings of the 30th International Conference on Machine Learning*, pp. 151–159, 2013.

Chuklin, A., Markov, I., and Rijke, M. d. Click models for web search. *Synthesis Lectures on Information Concepts, Retrieval, and Services*, 7(3):1–115, 2015.

Combes, R., Magureanu, S., and Proutiere, A. Minimal exploration in structured stochastic bandits. In *Advances in Neural Information Processing Systems*, pp. 1763–1771, 2017.

Dempster, A. P., Laird, N. M., and Rubin, D. B. Maximum likelihood from incomplete data via the em algorithm. *Journal of the Royal Statistical Society: Series B (Methodological)*, 39(1):1–22, 1977.

Gai, Y., Krishnamachari, B., and Jain, R. Combinatorial network optimization with unknown variables: Multi-armed bandits with linear rewards and individual observations. *IEEE/ACM Transactions on Networking*, 20(5): 1466–1478, 2012.

Gopalan, A., Mannor, S., and Mansour, Y. Thompson sampling for complex online problems. In *Proceedings of the 31st International Conference on Machine Learning*, pp. 100–108, 2014.

Haarnoja, T., Tang, H., Abbeel, P., and Levine, S. Reinforcement learning with deep energy-based policies. In *ICML*, 2017.

Haarnoja, T., Zhou, A., Abbeel, P., and Levine, S. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. In *ICML*, 2018.

Howard, R. A. and Matheson, J. E. Influence diagrams. *The Principles and Applications of Decision Analysis*, 1984.

Jun, K.-S., Willett, R., Wright, S., and Nowak, R. Bilinear bandits with low-rank structure. In *ICML*, 2019.

Katariya, S., Kveton, B., Szepesvari, C., Vernade, C., and Wen, Z. Stochastic rank-1 bandits. In *Artificial Intelligence and Statistics*, pp. 392–401, 2017.

Kawale, J., Bui, H., Kveton, B., Tran-Thanh, L., and Chawla, S. Efficient Thompson sampling for online matrix-factorization recommendation. In *Advances in Neural Information Processing Systems 28*, pp. 1297–1305, 2015.

Kveton, B., Szepesvari, C., Wen, Z., and Ashkan, A. Cascading bandits: Learning to rank in the cascade model. In *International Conference on Machine Learning*, pp. 767–776, 2015a.

Kveton, B., Wen, Z., Ashkan, A., and Szepesvari, C. Tight regret bounds for stochastic combinatorial semi-bandits. In *Proceedings of the Eighteenth International Conference on Artificial Intelligence and Statistics*, pp. 535–543, 2015b.

Kveton, B., Szepesvari, C., Ghavamzadeh, M., and Boutilier, C. Perturbed-history exploration in stochastic multi-armed bandits. In *Proceedings of the 28th International Joint Conference on Artificial Intelligence*, 2019a.

Kveton, B., Szepesvari, C., Ghavamzadeh, M., and Boutilier, C. Perturbed-history exploration in stochastic linear bandits. In *Proceedings of the 35th Conference on Uncertainty in Artificial Intelligence*, 2019b.

Kveton, B., Szepesvari, C., Vaswani, S., Wen, Z., Ghavamzadeh, M., and Lattimore, T. Garbage in, reward out: Bootstrapping exploration in multi-armed bandits. In *Proceedings of the 36th International Conference on Machine Learning*, pp. 3601–3610, 2019c.

Kveton, B., Zaheer, M., Szepesvari, C., Li, L., Ghavamzadeh, M., and Boutilier, C. Randomized exploration in generalized linear bandits. In *Proceedings of the 23rd International Conference on Artificial Intelligence and Statistics*, 2020.

Lagrée, P., Vernade, C., and Cappe, O. Multiple-play bandits in the position-based model. In *NIPS*, pp. 1597–1605, 2016.

Lattimore, T., Kveton, B., Li, S., and Szepesvari, C. Toprank: A practical algorithm for online stochastic ranking. In *Advances in Neural Information Processing Systems*, pp. 3945–3954, 2018.

Li, S., Wang, B., Zhang, S., and Chen, W. Contextual combinatorial cascading bandits. In *ICML*, volume 16, pp. 1245–1253, 2016.

Liu, B., Yu, T., Lane, I., and Mengshoel, O. Customized nonlinear bandits for online response selection in neural conversation models. In *Proceedings of the 32nd AAAI Conference on Artificial Intelligence*, pp. 5245–5252, 2018.

Lu, X. and Van Roy, B. Ensemble sampling. In *Advances in Neural Information Processing Systems*, pp. 3258–3266, 2017.

Lu, X., Wen, Z., and Kveton, B. Efficient online recommendation via low-rank ensemble sampling. In *Proceedings of the 12th ACM Conference on Recommender Systems*, pp. 460–464, 2018.

Osband, I., Aslanides, J., and Cassirer, A. Randomized prior functions for deep reinforcement learning. In *NIPS*, 2018.

Phan, M., Abbasi-Yadkori, Y., and Domke, J. Thompson sampling and approximate inference. In *NIPS*, 2019.

Riquelme, C., Tucker, G., and Snoek, J. Deep Bayesian bandits showdown: An empirical comparison of Bayesian deep networks for Thompson sampling. In *Proceedings of the 6th International Conference on Learning Representations*, 2018.

Russo, D. and Van Roy, B. Learning to optimize via posterior sampling. *Mathematics of Operations Research*, 39 (4):1221–1243, 2014.

Sutton, R. S. and Barto, A. G. *Reinforcement learning: An introduction*. MIT press, 2018.

Tatman, J. A. and Shachter, R. D. Dynamic programming and influence diagrams. *IEEE Transactions on Systems, Man, and Cybernetics*, 20(2):365–379, 1990.

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

Urteaga, I. and Wiggins, C. Variational inference for the multi-armed contextual bandit. In *International Conference on Artificial Intelligence and Statistics*, pp. 698–706, 2018.

Vaswani, S., Mehrabian, A., Durand, A., and Kveton, B. Old dog learns new tricks: Randomized UCB for bandit problems. In *Proceedings of the 23rd International Conference on Artificial Intelligence and Statistics*, 2020.

Wen, Z., Kveton, B., and Ashkan, A. Efficient learning in large-scale combinatorial semi-bandits. In *ICML*, pp. 1113–1122, 2015.

Zhang, J., Zhang, R., Carin, L., and Chen, C. Stochastic particle-optimization sampling and the non-asymptotic convergence theory. In *International Conference on Artificial Intelligence and Statistics*, pp. 1877–1887, 2020.

Zhang, R., Wen, Z., Chen, C., Fang, C., Yu, T., and Carin, L. Scalable Thompson sampling via optimal transport. In *AISTATS*, 2019.

Zimmert, J. and Seldin, Y. Factored bandits. In *Advances in Neural Information Processing Systems*, pp. 2835–2844, 2018.