Date of publication xxxx 00, 0000, date of current version xxxx 00, 0000.

Digital Object Identifier 10.1109/ACCESS.2017.DOI

Differential Privacy in Social Networks Using Multi-Armed Bandit

OLUSOLA T. ODEYOMI[1], (Member, IEEE),
[1]Department of Electrical Engineering and Computer Science, Wichita State University, Wichita, KS, 67260-0083 USA (e-mails: otodeyomi@shockers.wichita.edu

Corresponding author: Olusola T. Odeyomi (e-mail: otodeyomi@shockers.wichita.edu).

This work was supported in part by the Petroleum Technology Development Fund in Nigeria

ABSTRACT There has been an exponential growth over the years in the number of users connected to social networks. This has spurred research interest in social networks to ensure the privacy of users. From a theoretical standpoint, the social network is modeled as a directed graph network and interactions among agents in the graph network can be analyzed with non-Bayesian learning and online learning strategies - such as the multi-armed bandit. The goal of the agents is to learn the time-varying true state of the network through repeated cooperation among themselves. Recent work includes differential privacy in social network analysis to guarantee the privacy of shared information among the agents. However, the stochastic multi-armed bandit approach is used in these existing works which assume that the loss distribution is independent and identically distributed. This does not account for the arbitrariness of the time-varying true state in the social network. Therefore, this paper proposes a tougher but realistic setting that removes the restriction on the loss distribution. Two non-stochastic multi-armed bandit algorithms are proposed. The first algorithm uses the Laplace mechanism to guarantee differential privacy against a third-party intruder. The second algorithm uses the Laplace mechanism to guarantee differential privacy against both a third-party intruder and any spying agent in the network. The simulation results show that the agents' beliefs converge to the most dominant true state among the sequence of arbitrarily time-varying true states over the time horizon. The speed of convergence comes as a trade-off with privacy. Regret bounds are obtained for the proposed algorithms and compared to the non-private algorithm in the literature.

INDEX TERMS Non-Bayesian learning, diffusion learning, differential privacy, Laplace mechanism, multi-armed bandit, regret, online learning.

I. INTRODUCTION

THE social network has played an important role in our daily lives by providing a platform for social interactions. Social interactions are often among users having a common interest, i.e., friendship, colleagues at work, peer groups, etc. Social network users often collaborate to learn the truth about an event over time, through social interactions within their social connections. However, during such collaborations, it is important to protect shared vital information from third-party intruders, who are not members of the social connections. Also, it is pertinent for each user to share only information that is needed for social cooperation within its social connections, and protect sensitive information from other members of its social connections. For instance, it is common among young people to share insensitive photos on Flickr or other popular social media applications, to attract comments from friends and family members within their social connections. However, sensitive photos are stored in local photo applications on their mobile phones, with the hope that nobody can access and view them. Unfortunately, there are lots of privacy breaches in social networks [1].

A technique adopted in the past to provide privacy is to anonymize the datasets. However, research has shown that public information that is anonymized can be easily de-anonymized leading to sensitive information being exposed. For instance, in 2007, Netflix released anonymized datasets to researchers in the field of information retrieval for research purposes. After a year, the datasets were de-anonymized using public information from the Internet Movie Database (IMDb) which led to the re-identification of anonymized Netflix subscribers. This became a legal case of privacy breach with one of Netflix subscribers who claimed to have been adversely affected by the breach of privacy [2], [3]. Similarly, the medical records of the governor of Massachusetts were exposed by matching anonymized medical data with publicly available voter registration records [2]. The failure of anonymizing the datasets and the pressing need for researchers to provide a better technique to protect sensitive information led to differential privacy.

To address the issue of privacy in data analysis, Dwork et al. [2], [4], [5] proposed differential privacy which is a rigorous theoretical guarantee for privacy regardless of an intruder's prior knowledge about the databases. In particular, differential privacy guarantees that two databases that differ in only one record will output randomized results with identical probability distributions. Thus, the intruder cannot detect this difference in the databases whether it possesses some auxiliary information about the databases or analyzes the results. It is to be noted that the definition of privacy in [4], [5] is sufficient to guarantee the privacy of each agent's information against a third-party intruder, but it is insufficient to guarantee privacy when there is a curious spy within the social circle. Thus, the notion of local differential privacy was conceived and proposed by Duchi et. al [6]. Local differential privacy guarantees the privacy of information in a distributed system, where agents must cooperate, but do not trust each other.

Theoretical analyses of the social network are done by modeling the social network as a graph network [7], [8]. Social network users, referred to as agents, collaborate among themselves by a set of directed edges in the graph network. The graph network is strongly connected when each agent can communicate with every other agent. Non-Bayesian diffusion learning approaches are commonly used to analyze the interactions among these agents [9]. In these learning approaches, the goal is to learn the unknown true state of the graph network among a discrete set of states [10]–[13]. There are situations where the unknown true state is time-varying, such as the prediction of stock prices. Recent studies show that incorporating non-stochastic multi-armed bandit techniques into the non-Bayesian learning approaches can track this time-varying true state [14]–[16]. The non-stochastic multi-armed bandit is a variant of the online learning strategies that work well in sequential decision-making. To account for full privacy, private information shared by the agents during collaboration must be robust against third-party intruders and curious spies.

Most existing work on differential privacy that incorporates online learning strategies in social network assume that all agents receive full feedback over all possible actions [17]–[19]. This means that when an agent chooses a state as the true state from a set of states at a given round or time slot, the agent incurs the loss of its action - the chosen state, and also observes the losses of all unchosen states at the end of that round. However, in this paper, we consider a tougher setting where the agents can observe only the loss value of the state it chooses at each round. Also, no agent knows the loss incurred by other agents in each round. This setting is peculiar to the multi-armed bandit. It is to be noted that although no agent can observe the losses of other agents in a multi-armed bandit setting, there is still the possibility of privacy leakage to a spying agent during cooperation. Also, a third-party intruder is not restricted from observing the loss values of any agent if not protected. Few works incorporate multi-armed bandit technique with differential privacy in the social network [20], [21]. However, these works assume that the loss distribution is independent and identically distributed. Such assumption is impractical in the social network where the true state varies arbitrarily. Hence, this paper does not impose such an assumption on the loss distribution. We show that after a series of iterations using the proposed algorithms, the agents learn from the history of their past choices and make better decisions in tracking the time-varying true state with a privacy guarantee.

A. RESEARCH CONTRIBUTIONS

The contributions of this paper are as follows:

1) It models the social network as a graph network consisting of a set of strongly connected agents and a set of edges. The goal of the agents is to learn an arbitrarily time-varying true state of the network, with a privacy guarantee, against a third-party intruder and any spying agent.
2) It applies non-Bayesian learning, differential privacy, and non-stochastic multi-armed bandit seamlessly for the first time to achieve this goal.
3) Two non-stochastic multi-armed bandit algorithms are proposed. The first algorithm uses the Laplace mechanism to protect the incurred loss values of the agents from a third-party intruder, regardless of the computational power of this intruder. The second algorithm applies the Laplace mechanism to protect the incurred loss values of each agent from a third-party intruder, and also to protect the shared information of each agent when there is a spying agent in the neighborhood.
4) Simulation results are obtained to show that through cooperation over time, the agents' beliefs converge to the most dominant true state among the sequence of arbitrarily time-varying true states over the time horizon. The speed of convergence and the regret bounds of the proposed algorithms are compared with that of the non-private algorithm in the literature.

II. RELATED WORKS

There is some existing work in the literature that has addressed privacy concerns in graph networks. [22] introduced the concept of node-differential privacy and edge-differential privacy in graph networks. Node-differential privacy infers the removal or addition of one node and its incident edges in a graph network to generate another differentially private graph network. Node-differential privacy is a strong privacy guarantee difficult to achieve. On the other hand, edge-differential privacy infers the removal of an edge in the graph network to generate another differentially private graph. Edge-differential privacy is a weak differential privacy guarantee that is easily achievable. A variant of the edge-differential privacy is the k-edge differential privacy, where two nodes can differ by at most k number of edges. There have been various attempts by researchers to enforce differential privacy in graph networks through the intrinsic properties of the graph, such as its degree distribution [22], clustering coefficients [23], [24], eigenvalues and eigenvectors [25] and so on. Also, there have been attempts to enforce edge-differential privacy in graph generation, such as the Kronecker graph model [26], the dK-graph model [27], and the 2K-graph model [28].

In Bayesian inference, noise is added directly to the Bayesian updates to keep it differentially private. This means that noise is either added to the posterior parameters or their Fourier transform coefficients. This is based on the posterior sampling mechanism proposed in [29], [30]. The authors in [29] achieved differential privacy in non-parametric posterior without additional noise. There are other refinements of this mechanism such as in [31]. Also, the authors in [32] explored Monte-Carlo approaches to Bayesian inference based on posterior sampling mechanism. The authors in [33] applied probabilistic inference, by computing the posteriors in a noisy measurement model, to improve the utility of differentially private releases. Although differential privacy has been applied in Bayesian learning, it is yet to be extended to non-Bayesian learning.

In the online learning setting where data arrives sequentially, differentially private algorithms have been proposed with provable privacy guarantee as well as good regret bounds [34]. Recent work can be found in [19], [35]. In the multi-armed bandit, which is an online learning strategy, extensive work has been done to formulate ϵ-differentially private stochastic bandit algorithms from the classic non-private UCB algorithm [36]–[39]. These differentially private algorithms have nearly optimal regret bounds with provable privacy guarantees. In the non-stochastic multi-armed bandit, the authors in [40] and [41] have both proposed ϵ-differentially private algorithms with good regret bounds for non-private EXP3 and EXP2 algorithms respectively. The authors in [42] and [43] introduced the notion of local differential privacy both in stochastic and non-stochastic multi-armed bandit.

III. PRELIMINARIES

A. NETWORK MODEL

A graph network is denoted as $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, where $\mathcal{V} = \{1, \cdots, N\}$ represents a set of agents in the network with $|\mathcal{V}| = N$ and $\mathcal{E}$ represents the set of edges. A pair of non-negative scalar weights $\{a_{jk}, a_{kj}\} \in \mathcal{E}$ can be assigned to the edge joining agents $k \in \mathcal{V}$ and $j \in \mathcal{V}$. The network is defined as strongly connected if there exists a directed path in both ways connecting any two agents and at least a self-loop is present, i.e., $a_{kk} > 0$. There is the possibility of having $a_{jk} > 0$ and $a_{kj} = 0$. The neighborhood $\mathcal{N}_k$ of the agent k is the set of agents connected to k. Agent k is a member of its neighborhood. Also, the adjacency matrix of the graph can be defined as a square matrix whose elements represent the weights of the edges linking any two agents. The adjacency matrix is denoted as A. The adjacency matrix is left-stochastic when the sum of elements in each column is one i.e.,

$$a_{jk} \geq 0, \quad \sum_{j} a_{jk} = 1. \tag{1}$$

Strongly connected networks are left-stochastic and they have a spectral radius of one, i.e., their eigenvalues are always positive and bounded by one. They also obey the Perron-Frobenius theorem, and have a single eigenvalue at one, while other eigenvalues are strictly inside a unit disc [44].

B. DIFFUSION LEARNING

Diffusion learning starts by assigning a uniform prior belief to all agents in the network over each state. To illustrate mathematically, assume $\Theta = \{\theta_1, ..., \theta_M\}$ represents the set of all possible states that is detectable by a network, and assume $\theta_t^* \in \Theta$ represents the time-varying true state of the network that is unknown at time t. The prior belief of any agent k is given as $\mu_{k,0}(\theta) = \frac{1}{M}$ at time $t = 0$ over the state θ, where M is the cardinality of Θ. Each of the agents will update its belief at each time $t \geq 1$ by first observing a random observable signal. For agent k, its random observable signal can be denoted as $S_{k,t}$, and drawn from some known likelihood function $L_k(\cdot|\theta_t^*)$ that is dependent on the true state θ_t^*. The set of random observable signals $\{S_{k,t}\}_{t=1}^{t=T}$ belongs to a finite state space $\{\mathcal{Z}_{k,t}\}_{t=1}^{t=T}$, and it is independent over time and agents. These signals are not fully informed about the time-varying true state, thus, necessitating cooperation among the agents, i.e.,

$$S_{k,t} = \theta_t^* + n, \quad \forall k \in \mathcal{V}, t \leq T \tag{2}$$

where $n \sim \mathcal{N}(0, 1)$. The random observable signal is a noisy version of the underlying time-varying true state. Agent k computes the likelihood $L_k(S_{k,t}|\theta)$ over each state $\theta \in \Theta$ using this random observable signal as shown below:

$$L_k(S_{k,t}|\theta) = \frac{1}{\sqrt{2\pi\sigma_{k,t}^2}} \exp\left\{-(S_{k,t} - \theta)^2/2\sigma_{k,t}^2\right\} \tag{3}$$

where $\sigma_{k,t}^2$ represents the variance of agent k at time t. Then, the agent generates an intermediate belief using the Bayesian rule as follows:

$$\psi_{k,t}(\theta) = \frac{\mu_{k,t-1}(\theta) L_k\left(S_{k,t}|\theta\right)}{\sum_{\theta' \in \Theta} \mu_{k,t-1}\left(\theta'\right) L_k\left(S_{k,t}|\theta'\right)} \tag{4}$$

where $\psi_{k,t}(\theta)$ is the intermediate belief of the agent k at time t. Each agents can combine the weighted version of its intermediate belief with the weighted version of the intermediate beliefs of its neighbors over each state at each time t in a non-Bayesian fashion as shown below:

$$\mu_{k,t}(\theta) = \sum_{j \in \mathcal{N}_k} a_{jk} \psi_{j,t}(\theta) \tag{5}$$

TABLE 1. List of Notations

Notations	Meaning
$\mathcal{G}$	Graph network
$\mathcal{V}$	set of agents
$\mathcal{E}$	set of edges
$\mathcal{N}$	neighborhood
A	adjacency matrix
Θ	set of all possible states
θ_t^*	time-varying true state
θ	state
$\mu_{k,t}$	belief of agent k at time t
$L(\cdot\|\theta)$	a known Likelihood function dependent on θ
$S_{k,t}$	a random observable signal for the agent k at time t
$\mathcal{Z}_{k,t}$	a finite-state space for the agent k at time t
n	Gaussian noise
$\mathcal{N}(0,\sigma)$	Gaussian distribution with mean of 0 and variance of σ
$\psi_{k,t}(\theta)$	intermediate belief of agent k at time t over state θ
l	loss
$\mathbb{E}$	expectation
$\theta^\bullet$	best possible state
$\mathcal{A}$	randomized algorithm
P	probability
δ, ϵ	privacy parameters
t	time
T	time horizon
Reg	regret
Δl	sensitivity
$\|\| \cdot \|\|$	Manhattan norm
$\mathcal{M}$	private mechanism
$Lap(1/\epsilon)$	Laplace distribution
η	learning rate
γ	exploration parameter
N_L	Laplace noise
$p_{k,t}(\theta)$	probability of agent k at time t over state θ
$P_{k,t}(\theta)$	consensus probability of agent k at time t over state θ
$\mathbb{R}$	real numbers
exp	exponential
M	cadinality of Θ

C. MULTI-ARMED BANDIT PROBLEM

The multi-armed bandit is a game set between an agent and an adversary. The game setting is as follows: There is a set of states denoted by $\Theta = \{\theta_1, \cdots, \theta_M\}$. An oblivious adversary fixes the loss $l_t(\theta) \in [0,1]$ for all states before the start of the game. At the start of the game, an agent selects a state θ_t at each time t, and incur the loss $l_t(\theta_t) \in [0,1]$. The agent only observes its incurred loss $l_t(\theta_t)$ at every time t, but the agent is unaware of the losses of states not chosen at such time. For instance, for a binary loss game setting, when the chosen state of the agent at time t is the same as the true state at that time i.e., $\theta_t = \theta_t^*$, then the loss $l_t(\theta_t) = 0$, and if $\theta_t \neq \theta_t^*$, then $l_t(\theta_t) = 1$. The goal of the agent is to minimize its total incurred losses from time $t=1$ to time $t=T$ given as $\sum_{t=1}^{T} l_t(\theta_t)$. The performance of the agent is compared against an oracle that sticks to the best state $\theta^\bullet$ over the entire duration of the game. This performance metric is known as regret, which is defined as the difference between the total loss incurred by the agent and the total loss incurred by the oracle over the time horizon T.

$$Reg_T := \sum_{t=1}^{T} l_t(\theta_t) - \sum_{t=1}^{T} l_t(\theta^\bullet) \tag{6}$$

The above regret definition is deterministic. It is sometimes difficult to obtain deterministic regret. Thus, it is important to consider the expected regret. Expected regret necessitates the use of a randomized bandit algorithm. The expected regret is defined as

$$\mathbb{E}[Reg_T] := \mathbb{E}\left[\sum_{t=1}^{T} l_t(\theta_t) - \sum_{t=1}^{T} l_t(\theta^\bullet)\right] \tag{7}$$

where the expectation is taken over the randomness of the choice of the agent's actions and its incurred losses.

D. DIFFERENTIAL PRIVACY

In differential privacy, the goal is to ensure that no third party intruder can extract sensitive information from the output of a private mechanism (i.e., a randomized private bandit algorithm), even if there is a distortion of a loss value among the sequence of loss values incurred by an agent. It is formally defined in the non-stochastic multi-armed bandit setting as follows [40]–[42]:

Definition 1: A randomized bandit algorithm $\mathcal{A}$ is (ϵ, δ) differentially private at around t, if for all loss sequence $l_{1:t-1}$ and $l'_{1:t-1}$ that differs in at most one round for any subset $\zeta \subseteq \Theta$

$$P(\theta_t \in \zeta | l_{1:t-1}) \leq \delta + P(\theta_t \in \zeta | l'_{1:t-1}) exp(\epsilon) \tag{8}$$

where $l_{1:t-1} = l_1(\theta_1), \cdots, l_i(\theta_i), \cdots l_{t-1}(\theta_{t-1})$; $l'_{1:t-1} = l_1(\theta_1), \cdots, l_i(\theta'_i), \cdots, l_{t-1}(\theta_{t-1})$, $\theta', \theta \in \Theta$, and $i < t-1$; P is the probability distribution specified by the randomized private algorithm; ϵ and δ are parameters that define the privacy loss. When $\delta = 0$, then, the randomized algorithm is said to be ϵ-differentially private. Generally lower (ϵ, δ) indicates higher privacy and vice-versa. The intuition in Definition 1 is that even if there is a change in an incurred loss in just one round among the loss sequence from round 1 to round $t-1$, the agent will still choose the same state at round t. This means that no third-party intruder can infer any information about the loss value either by simply observing the chosen state incurred or by distorting a loss value, no matter the computational power of this third-party intruder. Hence, the agent's loss value at time t is kept private.

The privacy guarantee in Definition 1 is for one round, hence, it is said that the algorithm has instantaneous privacy parameters. To ensure privacy for all rounds, such that the algorithm has a cumulative privacy parameter, Definition 1 will be redefined as follows:

Definition 2: A randomized bandit algorithm $\mathcal{A}$ is (ϵ, δ) differentially private up to round t, if for the loss sequences $l_{1:t-1}$ and $l'_{1:t-1}$ that differ in at most one round, and for a subset $\zeta \subseteq \Theta$, we have

$$P(\theta_{1:t} \in \zeta | l_{1:t-1}) \leq \delta + P(\theta_{1:t} \in \zeta | l'_{1:t-1}) \exp(\epsilon) \quad (9)$$

where $\theta_{1:t} = \theta_1, ..., \theta_t$. The goal of the private bandit algorithm is to ensure that the cumulative privacy loss is low as possible, while still maintaining a low regret or high utility. This causes a trade-off challenge between privacy and utility. We refer readers to [42] for elaborate discussion on cumulative privacy parameter.

IV. DIFFERENTIALLY PRIVATE MULTI-ARMED BANDIT

The expected regret definition in equation (7) is for a single agent. In a social network, there are many agents. Thus, the game setting is reformulated as follows: An oblivious adversary fixes the loss $l_{k,t}(\theta) \in [0,1]$ for each agent k at each time t and overall states $\theta \in \Theta$ before the start of the game. At each time t, each agent k chooses a state θ_t and observes the loss $l_{k,t}(\theta_t) \in [0,1]$. Each agent does not know the loss value of the states it does not choose at each time, and also, it does not know the loss value of the states chosen by other agents. The goal of each agent is to minimize its regret over the time horizon of the game by incurring the possible minimum number of losses. The expected weighted regret for each agent is defined as the difference between the expected cumulative loss of the agent and the expected cumulative loss incurred by the oracle. The expected weighted regret for the agent k is given as:

$$\mathbb{E}[Reg_T] := \mathbb{E}_{\mathcal{F}_T}\left[\sum_{t=1}^{T}\sum_{\theta\in\Theta}\mu_{k,t}(\theta) l_{k,t}(\theta) - \sum_{t=1}^{T} l_{k,t}\left(\theta^{\bullet}\right)\right], \quad (10)$$

where $\mu_{k,t}(\theta)$ is the belief of the agent k over the state $\theta \in \Theta$; $\mathcal{F}_T = \sigma(S_{k,1}, ..., S_{k,T}, l_{k,1}, ..., l_{k,T}, \theta_1, .., \theta_T)$ is the filtration that represents the history of the agent over all observed random signals, states chosen, and incurred losses.

Although, each agent cannot observe the incurred loss values of other agents in the network; a third-party intruder will observe the exact incurred loss values of any agent if the loss values are not private. Hence, each agent uses a private randomized bandit algorithm to secure its loss values. Thus, Definition 2 is restated as follows:

Definition 3: A randomized bandit algorithm $\mathcal{A}_k$ for the agent k is (ϵ, δ)-differentially private up to round t, if for the loss sequences $l_{k,1:t-1}$ and $l'_{k,1:t-1}$ that differs in at most one round, and for a subset $\zeta \subseteq \Theta$, we have

$$P(\theta_{1:t} \in \zeta | l_{k,1:t-1}) \leq \delta + P(\theta_{1:t} \in \zeta | l'_{k,1:t-1}) \exp(\epsilon) \quad (11)$$

If $\delta = 0$, the private bandit algorithm is said to be ϵ-differentially private for the agent k.

To learn the time-varying true state, the agents are expected to cooperate. Such cooperation may leak vital information about each agent. For instance, from equation (5), each agent observes the intermediate beliefs of other agents in its neighborhood. Such observation may breach the privacy of the neighbors. Thus, the notion of local differential privacy, where each agent protects its vital information before sharing it with its neighbors is important. Local differential privacy is defined in Definition 4.

Definition 4 [42]: A randomized bandit algorithm $\mathcal{A}_k$ for the agent k is locally differentially private, if its input are generated through an (ϵ, δ)-differentially private mechanism $\mathcal{M}$.

Definition 4 infers that the input to the algorithm $\mathcal{A}_k$ undergoes pre-processing by a private mechanism $\mathcal{M}$. Using equation (5) for illustration only, the intermediate belief from each neighbor $j \in \mathcal{N}_k$ will first be pre-processed by a private mechanism, where it will undergo (ϵ, δ)-differential privacy before it is sent as input to the randomized private bandit algorithm $\mathcal{A}_k$ of the agent k for consensus, i.e.,

$$P_{\mathcal{M}}(\psi_{j,1:t} \in \beta | l_{k,1:t-1}) \leq \delta + P_{\mathcal{M}}(\psi_{j,1:t} \in \beta | l'_{k,1:t-1}) \exp(\epsilon) \quad (12)$$

where $\beta \subseteq \mathbb{R}$ and $P_{\mathcal{M}}$ is the probability distribution specified by the private mechanism $\mathcal{M}$. Then, the algorithm $\mathcal{A}_k$ is differentially private with respect to $l_{k,1:t-1}$ through post-processing as shown below:

$$P(\theta_{1:t} \in \zeta | \psi_{j,1:t}) \leq \delta + P(\theta_{1:t} \in \zeta | \psi'_{j,1:t}) \exp(\epsilon). \quad (13)$$

A variant of equation (5) discussed in detail in Section V involves the transfer of intermediate probabilities, instead of intermediate beliefs. Definition 3 protects the information of each agent from a third-party intruder, while Definition 4 protects the information of each agent from other spying agents, to avoid privacy breaches.

Definition 5: For any $l_{k,1:t}$ and $l'_{k,1:t}$ that differs in only one round, the L_1 sensitivity at the $t-th$ round is given as

$$\Delta l = max||l_{k,1:t} - l'_{k,1:t}||_1 = 1 \tag{14}$$

where $||\cdot||_1$ is the Manhattan norm. The maximum change that can occur in the loss sequence in any round is bounded by 1, because the loss values themselves are bounded by 1.

V. PROPOSED ALGORITHMS

The proposed algorithms are non-stochastic multi-armed bandit algorithms that involve a trade-off between exploration and exploitation. Algorithm 1 is proposed to secure the loss values of the agents against a third-party intruder outside the network. However, the agents in the network can share information among themselves with trust and do not need to protect shared information. This means that algorithm 1 is useful when all agents are cooperative, and there is no spy among the agents. Algorithm 2 is proposed to secure all information of an agent against both a third-party intruder and any spying neighboring agents seeking to know more information about the agent than what is necessary. The non-private algorithm of these proposed algorithms is found in [16]. The non-private version leaks information to a third-party intruder and a spying agent. For convenience, the non-private algorithm is shown in the Appendix as Algorithm 3. The input parameters of the proposed algorithms are the feedback graph, the exploration parameter γ, and the learning rate η. Each agent runs the proposed algorithms independently. At time $t = 0$, the belief $\mu_{k,t}(\theta)$ of the agent k is initialized over the states $\theta \in \Theta$. Let us focus on how the algorithms work starting with Algorithm 1. At each round $t \in \{1, \cdots, T\}$, the following steps are executed:

In Step 1, an intermediate probability $p_{k,t}(\theta)$ is computed. The intermediate probability is necessary to provide a trade-off between exploitation and exploration as commonly done in multi-armed bandits. The algorithm finds a balance for the agent k to either stick to its previous belief about the time-varying true state or explores its generated intermediate belief at time t. This balance is controlled by γ. *In algorithm 1, $p_{k,t}(\theta)$ does not undergo differential privacy.*

Step 2 involves the computation of the consensus probability $P_{k,t}(\theta)$. The consensus probability sums the weighted intermediate probabilities of the agent k and that of the other agents in the neighborhood of k. The consensus probability is computed because the agent k cannot accurately learn about the true state on its own and needs to cooperate with other agents in its neighborhood. This can be understood from the fact that the consensus probability $P_{k,t}(\theta)$ depends on the intermediate probability $p_{k,t}(\theta)$ according to Step 2. Similarly, to compute $p_{k,t}(\theta)$ in Step 1, the intermediate belief $\psi_{k,t}(\theta)$ of the agent k must have been computed using equation (4). Also, this intermediate belief depends on the likelihood $L_k(S_{k,t}|\theta)$. However, the signal $S_{k,t}$ in $L_k(S_{k,t}|\theta)$ is not fully informative about the true state as shown in equation (2), which necessitates cooperation among the agents.

In step 3, a state θ_t is drawn according to the consensus probability distribution $P_{k,t}$ and the loss $l_{k,t}(\theta_t) \in [0,1]$ is incurred. Step 3 is common to all multi-armed bandit algorithms. If the chosen state θ_t matches the exact true state θ^*_t, then the incurred loss at that round is 0, but if otherwise, the incurred loss is 1.

In step 4, Laplace noise is drawn from a Laplace distribution $Lap(1/\epsilon)$ (See Definition 6 and Corollary 1) is added to the incurred loss $l_{k,t}(\theta_t)$, to randomize it and make it noisy. The essence of adding random Laplace noise to the incurred loss is to prevent a third-party spy from knowing the exact loss value of the agent k. For the sake of analogy only, let us assume that the loss value is not randomized with Laplace noise, then a third-party spy will know if agent k predicts the true state correctly or not, by simply observing the loss value whether it is 0 or 1. However, after the addition of Laplace noise to the loss value, it becomes difficult for the third-party spy to know if agent k correctly predicted the true state or not. Practically, using stock predictions as an example, a third-party spy can accurately tell if an agent correctly predicts the stock price or not, if it sees the exact loss (or profit) of the agent. By randomizing this loss, it becomes difficult for the third-party spy to infer correct information about the stock trading of the agent. Thus, privacy is not breached. We can bound the amount of Laplace noise added to the incurred loss. The bounding is important to ensure that we do not lose the information we are trying to protect due to added unbounded noise. This will be explained further in step 5.

In step 5, the noisy incurred loss $l^N_{k,t}(\theta_t)$ is computed. This is the sum of the incurred loss over the chosen state $l_{k,t}(\theta_t)$ and the Laplace noise $N_{k,t}$ for the agent k at the time t. The added Laplace noise is bounded between $[-b, b]$ where $b \in \mathbb{R}$ as shown in Step 4. This bounding is necessary to regulate the amount of noise added to the loss value. Adding unbounded noise will slow down convergence to the most dominant true state. For instance, adding a Laplace noise value of 100 will be considered too large since the loss value is bounded between 0 and 1. Recall that there is a trade-off between privacy and utility. For rounds where the Laplace noise drawn from the Laplace distribution is outside the range $[-b, b]$, the algorithm uses $N_{k,t} = \frac{b}{2}$. Still, the third-party spy does not know the exact value of the true loss incurred even when the Laplace noise is deterministic (i.e., when we use $N_{k,t} = \frac{b}{2}$). This is because the number of rounds where Laplace noise is drawn from the Laplace distribution would exceed the range $[-b, b]$ is random and likely few for a carefully chosen value of b. This means that the third-party spy cannot accurately predict the rounds where a deterministic Laplace noise is used. Bounding the Laplace noise is common in algorithms that apply differential privacy [40], [43], [45].

In step 6, the noisy incurred loss is bounded to keep it within $[0,1]$. This is necessary because we want the algorithm to behave like traditional multi-armed bandit algorithms where the loss value is within $[0,1]$. For the sake of analogy, assume that $l_{k,t}(\theta_t)$ at round t is 1 and $N_{k,t} = 0.2$, then $l^N_{k,t}(\theta_t) = 1.2$. If we assume that the added Laplace noise is within the range $[-b, b]$, the noisy loss value still exceeds 1. Bounding at this stage is used specifically in multi-armed bandit algorithms where we desire that the loss is kept within $[0, 1]$ (refer to Algorithm 1 and Theorem 3.1 in [40]). Since $2b + 1 \geq l^N_{k,t}(\theta_t) + b$, the noisy loss value is kept within $[0, 1]$. Despite this bounding, the expected value of the noisy incurred loss gives the true incurred loss (i.e., $\mathbb{E}[l^N_{k,t}(\theta)] = l_{k,t}(\theta)$). A third-party intruder cannot infer any vital information from the noisy incurred loss because it is differentially private as shown in Lemma 1.

In Step 7, an estimated noisy loss $\hat{l}^N_{k,t}(\theta)$ is computed over all the states $\theta \in \Theta$ in order to update the belief $\mu_{k,t}(\theta)$ in Step 8. This computation is necessary because the agent k knows the true loss value of the state it chooses at a given round (i.e., $l_{k,t}(\theta_t)$) but it does not know the true loss value of the other unchosen states at that round. For instance, if at time t, the agent k chooses θ_1 as its true state from the set $\Theta = \{\theta_1, \cdots, \theta_M\}$ based on its computed consensus probability, then it observes the loss value of θ_1 as explained in Step 3. However, it does not know the true loss values of other unchosen states. This is a well-known characteristic of adversarial multi-armed bandit algorithms [40], [46]. This becomes a challenge because the algorithm needs to update the belief $\mu_{k,t}(\theta)$ over all the states using the loss values. A common technique used to overcome this challenge in multi-armed bandit will be to compute an unbiased estimate of the noisy loss $\hat{l}^N_{k,t}(\theta)$ over all the states. It is to be noted that computing such an estimated noisy loss still preserves privacy since the value of the true loss for each of the unchosen states remains unknown. This means that no third-party spy can know the true loss values of the unchosen states. The expectation over this estimated noisy loss gives the true noisy loss as shown below:

$$\mathbb{E}_{\mathcal{F}_t|\mathcal{F}_{t-1}}[\hat{l}^N_{k,t}(\theta)|\mathcal{F}_{t-1}] = \sum_{\Theta} \frac{l^N_{k,t}(\theta)}{P_{k,t}(\theta)} P_{k,t}(\theta_t)\mathbb{I}\{\theta_t = \theta\}.$$

$$= \frac{l^N_{k,t}(\theta)}{P_{k,t}(\theta)} P_{k,t}(\theta) = l^N_{k,t}(\theta). \tag{15}$$

In Step 8, the updated belief is computed using exponential weighting. The normalization in Step 8 is to ensure that the sum of the beliefs over all states $\theta \in \Theta$ is equal to 1. A round of iteration is complete and the algorithm starts all over from step 1 until it reaches the time horizon.

Algorithm 2 is slightly different from Algorithm 1 due to the addition of Laplace noise to the intermediate probability $p_{k,t}(\theta)$ before computing the consensus probability as shown from Steps 2 to 5. The explanation for bounding the Laplace noise is the same as discussed in Algorithm 1. Step 5 shows that agent k receives noisy intermediate probabilities from its neighbors (i.e., $p^N_{j,t}(\theta) \quad \forall j \in \mathcal{N}_k$), which it uses to compute its consensus probability. Each neighboring agent privatizes its intermediate probability over all the states before it is sent to the agent k for consensus. The essence of adding Laplace noise to the intermediate probability is to ensure that agent k does not infer any information from the noisy intermediate probabilities it receives from its neighbors. This mitigates the risk of agent k breaching the privacy of its neighbors. Laplace noise added to the intermediate probability is independent of all the states and overall the agents. Thus, the consensus probability computes the sum of the weighted noisy intermediate probabilities of all agents in the neighborhood at time t over all the states. Since the algorithm is run independently by each agent in the network, it means that no spying agent can access the information of its neighbors.

VI. THEORETICAL RESULTS

This section gives the theoretical results.

Definition 6: The Laplace distribution centered at zero with scale c has the probability distribution

$$Lap(x|c) = \frac{1}{2c} exp\left(\frac{-|x - 0|}{c}\right) \tag{16}$$

where x is a random variable. This is the standard definition of Laplace distribution. Laplace distribution centered at zero is known to be the symmetric version of an exponential function.

Corollary 1: For the Laplace mechanism used to generate the noisy loss $l^N_{k,t}(\theta)$ in Algorithms 1 and 2, the scale c is given as $\Delta l/\epsilon$. Hence,

$$Lap(l^N_{k,1:t-1}|\Delta l/\epsilon) = \frac{1}{2\Delta l/\epsilon} exp\left(\frac{-||l^N_{k,1:t-1} - 0||_1}{\Delta l/\epsilon}\right) \tag{17}$$

For ease of notation, $Lap(l^N_{k,1:t-1}|\Delta l/\epsilon)$ will be simply represented as $Lap(\Delta l/\epsilon)$. By applying Definition 5, the Laplace distribution for Algorithms 1 and 2 from which Laplace noise is drawn and added to the true loss is given as $Lap(1/\epsilon)$.

Lemma 1: The noisy loss sequence $l^N_{k,t-1}$ in Algorithm 1 and 2 preserves $(\epsilon, 0)$-differential privacy.

Proof: Let $l^N_{k,1:t} = \left(l^N_{k,1}(\theta) \cdots l^N_{k,\tau}(\theta) \cdots l^N_{k,t}(\theta)\right)$ and $l^{N'}_{k,1:t} = \left(l^N_{k,1}(\theta) \cdots l^N_{k,\tau}(\beta) \cdots l^N_{k,t}(\theta)\right)$ differ in one round with θ and $\beta \in \Theta$. Let the Laplace distribution be centered around an arbitrary loss sequence $l^{N''}_{k,1:t-1}$, then

$$\frac{P(\theta_{1:t} \in \zeta|l^N_{k,1:t-1})}{P(\theta_{1:t} \in \zeta|l^{N'}_{k,1:t-1})} =^{(a)} \frac{\exp(-\epsilon||l^N_{k,1:t-1} - l^{N''}_{k,1:t-1}||_1)}{\exp(-\epsilon||l^{N'}_{k,1:t-1} - l^{N''}_{k,1:t-1}||_1)}$$

$$= \exp(\epsilon||l^{N'}_{k,1:t-1} - l^{N''}_{k,1:t-1}||_1 - \epsilon||l^N_{k,1:t-1} - l^{N''}_{k,1:t-1}||_1)$$

$$\leq^{(b)} exp(\epsilon||l^N_{k,1:t-1} - l^{N'}_{k,1:t-1}||_1)$$

$$\leq^{(c)} \exp(\epsilon)$$

where in (a), we use the Laplace mechanism in Corollary 1, with $\Delta l = 1$ and $|| \cdot ||_1$ is the Manhattan norm; in (b), we use triangle inequality; and in (c), we use the fact that $||l^N_{k,1:t-1} - l^{N'}_{k,1:t-1}||_1 \leq 1$.

```
Algorithm 1: Private Multi-Armed Bandit Algorithm With Non-Spying Agents
Parameters: Feedback graph, learning rate $\eta > 0$.
$V$ is the set of strongly connected agents and $E$ is the set of edges.
Exploration parameter $\gamma \in (0, \frac{1}{2}]$
Initialize $\mu_{k,0}(\theta) = \frac{1}{M}$
Output: $\mu_{k,t}(\theta) \quad \forall\theta \in \Theta$
For each round $t \in \{1, \cdots, T\}$
Step 1:Compute $p_{k,t}(\theta) = (1-\gamma)\mu_{k,t-1}(\theta) + \gamma\psi_{k,t}(\theta)\ \forall\ \theta \in \Theta$
Step 2: Compute $P_{k,t}(\theta) = \sum_{j \in \mathcal{N}_k} a_{jk} p_{j,t}(\theta), \quad P_{k,t} = (P_{k,t}(\theta_1), ..., P_{k,t}(\theta_M))$
Step 3: Draw state $\theta_t \sim P_{k,t}$ and incur loss $l_{k,t}(\theta_t) \in [0,1]$
Step 4: Draw Laplace noise $N_{k,t} \sim Lap(1/\epsilon)$
    If $N_{k,t} \in [-b, b]$ for some fixed number $b \in \mathbb{R}$ then
        Use the exact value of $N_{k,t}$
    Else Use $N_{k,t} = \frac{b}{2}$
Step 5: Compute $l^N_{k,t}(\theta_t) = l_{k,t}(\theta_t) + N_{k,t}; \quad l^N_{k,t}(\theta_t) \in [-b, b+1]$
Step 6: Scale $l^N_{k,t}(\theta_t)$ to $[0,1]$ using $l^N_{k,t}(\theta_t) = \frac{l^N_{k,t}(\theta_t)+b}{2b+1}$
Step 7: Compute $\hat{l}^N_{k,t}(\theta) = \frac{l^N_{k,t}(\theta)}{P_{k,t}(\theta)}\mathbb{I}\{\theta = \theta_t\} \quad \forall\theta \in \Theta$
Step 8: Update
    $\mu_{k,t}(\theta) = \frac{\mu_{k,t-1}(\theta)\exp\left(-\eta\hat{l}^N_{k,t}(\theta)\right)}{\sum_{\theta' \in \Theta}\mu_{k,t-1}(\theta')\exp\left(-\eta\hat{l}^N_{k,t}(\theta')\right)} \quad \forall\theta \in \Theta$
End
```

Remark 1: The goal of Lemma 1 is to show that adding Laplace noise to the loss values preserves differential privacy. The proof of the Lemma is similar to the proof for $(\epsilon, 0)$-differential privacy in Theorem 3.1 [2]. However, both proofs differ in the sense that Lemma 1 applies to online learning algorithms where the parameter of interest is time-varying, while the proof in [2] is for offline learning where the parameter of interest does not vary with time. Since the noisy loss sequence is $(\epsilon, 0)$-differentially private up to round t as shown in Lemma 1, Algorithms 1 and 2 preserve differential privacy for the incurred losses. By similar analog, we can show that the noisy intermediate probabilities are $(\epsilon, 0)$-differentially private up to round t for Algorithm 2. Thus, Algorithm 2 preserves differential privacy both for the incurred losses and the intermediate probabilities.

Theorem 1 [Theorem 3.1 in [40]]: The expected regret bound for any (ϵ, δ)-differentially private algorithm wrapping a non-private base algorithm with scaled loss $l^N_{k,t}(\theta) = \frac{l^N_{k,t}(\theta)+b}{2b+1}$ is given as

$$\mathbb{E}[Reg_T] \leq \frac{2b}{\max a_{kk}}\mathbb{E}[Reg^{base}_T] + 2TM\exp(-\epsilon b) + \frac{\sqrt{32T}}{\epsilon} \tag{18}$$

where $\mathbb{E}[Reg^{base}_T]$ is the expected regret of the non-private algorithm and $\max a_{kk}$ is the largest self-loop weight in the adjacent matrix.

Remark 2: Corollary 1 implies that if the non-private version of any private algorithm is known, then the expected regret for the private algorithm is bounded by (18).

Corollary 2: Given that $\mathbb{E}[Reg^{base}_T] \leq O(\sqrt{\alpha T \ln M})$ with α as the graph independence number, and $b = \frac{\ln T}{\epsilon}$, the regret bound for Algorithm 1 is given as

$$\mathbb{E}[Reg_T] \leq \frac{2\ln T\sqrt{\alpha T \ln M}}{\epsilon \max a_{kk}} + 2M + \frac{\sqrt{32T}}{\epsilon} \tag{19}$$

Remark 3: Substituting the values of the expected regret for the non-private algorithm and the parameter b into Corollary 1 gives Corollary 2. The proof of the expected regret for the non-private base algorithm is shown in the Appendix. The proof is similar to the proof in [16] but with a slightly different regret definition. Refer to the Appendix for more explanation. The upper bound on the expected regret in Corollary 2 is given as $O(\sqrt{\alpha T}\ln M/\epsilon)$ with $T >> M$. This upper bound determines the rate at which the regret for algorithm 1 grows.

Theorem 2: Given that $\mathbb{E}[Reg^{base}_T] \leq O(\sqrt{\alpha T \ln M})$, $b = \frac{\ln T}{\epsilon}$ and $b' = 1$, the expected regret for Algorithm 2 is given as

$$\mathbb{E}[Reg_T] \leq \frac{2\ln T\sqrt{\alpha T \ln M}}{\epsilon} + 4M + \frac{2\sqrt{32T}}{\epsilon} \tag{20}$$

Remark 4: Since more Laplace noise is used in algorithm 2, it can be seen that the expected regret is worse than for algorithm 1. However, this comes with increased privacy, thus obeying the privacy-utility trade-off. The regret of Algorithm 2 grows at a rate of $O(\alpha^{1/2}T^{3/2}\ln M/\epsilon)$ with $T >> M$.

VII. SIMULATION RESULTS

For the simulation, we use the strongly connected network in Fig. 1 consisting of three agents with column-stochastic adjacency matrix as shown below:

$$A = \begin{bmatrix} 0.2 & 0.2 & 0.8 \\ 0.5 & 0.4 & 0.1 \\ 0.3 & 0.4 & 0.1 \end{bmatrix}$$

The goal of the agents is to track the arbitrarily time-varying true state θ^*_t of the network from the set $\Theta = \{\theta_1, \cdots, \theta_5\}$ at each time t. To achieve this, the agents must cooperate in a non-Bayesian fashion while still maintaining privacy. Algorithms 1 and 2 are used to help the agents track this time-varying true state. The parameters for algorithm

```
Algorithm 2: Private Multi-Armed Bandit Algorithm With Spying Agents
Parameters: Feedback graph, learning rate $\eta > 0$.
$V$ is the set of strongly connected agents and $E$ is the set of edges.
Exploration parameter $\gamma \in (0, \frac{1}{2}]$
Output: $\mu_{k,t}(\theta) \quad \forall \theta \in \Theta$
Initialize $\mu_{k,0}(\theta) = \frac{1}{M}$
For each round $t \in \{1, \cdots, T\}$
Step 1: Compute $p_{k,t}(\theta) = (1-\gamma)\mu_{k,t-1}(\theta) + \gamma\psi_{k,t}(\theta) \quad \forall\, \theta \in \Theta$
Step 2: Draw Laplace $N_{k,t} \sim Lap(1/\epsilon)$ and add to $p_{k,t}(\theta) \quad \forall \theta \in \Theta$
          If $N_{k,t} \in [-b', b']$ for some fixed number $b' \in \mathbb{R}$ then
             Use the exact value of $N_{k,t}$
          Else $N_{k,t} = \frac{b'}{2}$ .
Step 3: Scale $p_{k,t}(\theta)$ to $[0,1]$ using $p_{k,t}(\theta) = \frac{p_{k,t}(\theta)+b'}{2b'+1}$.
Step 4: Compute $p^N_{k,t}(\theta) = p_{k,t}(\theta) + N_{k,t}$; $p^N_{k,t}(\theta) \in [-b', b'+1]$
Step 5: Compute $P_{k,t}(\theta) = \sum_{j \in \mathcal{N}_k} a_{jk} p^N_{j,t}(\theta), \quad P_{k,t} = (P_{k,t}(\theta_1), ..., P_{k,t}(\theta_M))$
Step 6: Draw state $\theta_t \sim P_{k,t}$ and incur loss $l_{k,t}(\theta_t) \in [0,1]$
Step 7: Draw Laplace noise $N_{k,t} \sim Lap(\frac{1}{\epsilon})$
          If $N_{k,t} \in [-b, b]$ for some fixed number $b \in \mathbb{R}$ then
             Use the exact value of $N_{k,t}$
          Else $N_{k,t} = \frac{b}{2}$
Step 8: Compute $l^N_{k,t}(\theta_t) = l_{k,t}(\theta_t) + N_{k,t}; \quad l^N_{k,t}(\theta_t) \in [-b, b+1]$
Step 9: Scale $l^N_{k,t}(\theta_t)$ to $[0,1]$ using $l^N_{k,t}(\theta_t) = \frac{l^N_{k,t}(\theta_t)+b}{2b+1}$
Step 10: Compute $\hat{l}^N_{k,t}(\theta) = \frac{l^N_{k,t}(\theta)}{P_{k,t}(\theta)} \mathbb{I}\{\theta = \theta_t\} \quad \forall \theta \in \Theta$
Step 11: Update
             $\mu_{k,t}(\theta) = \frac{\mu_{k,t-1}(\theta)\exp\left(-\eta \hat{l}^N_{k,t}(\theta)\right)}{\sum_{\theta' \in \Theta} \mu_{k,t-1}(\theta')\exp\left(-\eta \hat{l}^N_{k,t}(\theta')\right)} \quad \forall \theta \in \Theta$
End
```

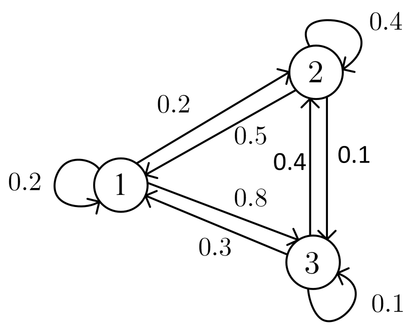

FIGURE 1. A strongly connected network consisting of three agents.

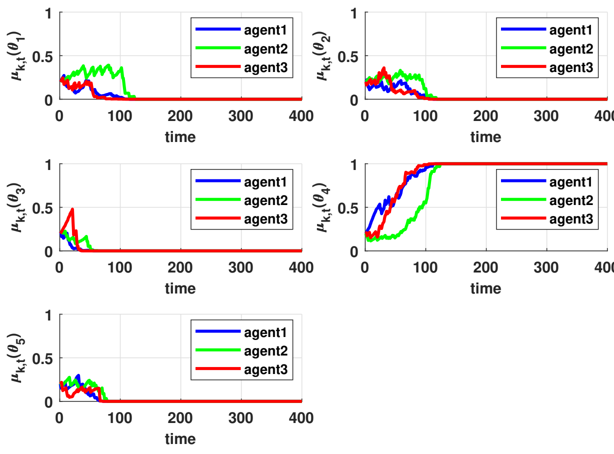

FIGURE 2. Convergence of agents' beliefs in the strongly connected network to θ_4 at $\eta = 0.1$ at the 1^{st} iteration using Algorithm 1.

1 are $\gamma = 0.1$, $\eta = 0.1$, $\epsilon = 0.1$, $T = 400$ and $b = \ln(400)/0.1$. The parameters for algorithm 2 are $\gamma = 0.1$, $\eta = 0.1$, $\epsilon = 0.1$, $T = 500$, $b = \ln(400)/0.1$ and $b' = 1$. The prior belief $\mu_{k,0}(\theta)$ of each agent k over each state is $\frac{1}{5}$. To compute the intermediate belief $\psi_{k,t}(\theta)$ in step 1 of both algorithms, the random observable signal of each agent $S_{k,t}(\theta)$ (which is a noisy version of the underlying time-varying true state), is first drawn from $\mathcal{N}(\theta^*_t, 1)$ and used to compute the likelihood $L_k(S_{k,t}|\theta)$ according to equation (3). Then, the intermediate probability can be computed as a trade-off between the previous belief $\mu_{k,t-1}(\theta)$ and the current intermediate belief $\psi_{k,t}(\theta)$. Laplace noise is not added to the intermediate probability in Algorithm 1, but Laplace noise is added to the intermediate probability in Algorithm 2. Also, Laplace noise is added to the incurred loss of each agent at each time in both algorithms. The simulation is repeated over 50 iterations. The simulation results for the graph network in Fig. 1 are shown from Figs. 2 - 5. The regret bounds comparison is shown in Fig. 6.

Fig. 2 shows the convergence of the agents' beliefs to the most dominant true state at the 1^{st} iteration, when Algorithm 1 is used. The most dominant true state appears to be the most stable state among the sequence of arbitrarily time-varying true states, i.e., the most frequently occurring state from $\theta^*_1, \cdots, \theta^*_T$. Illustrating fluctuating stock prices, stockbrokers

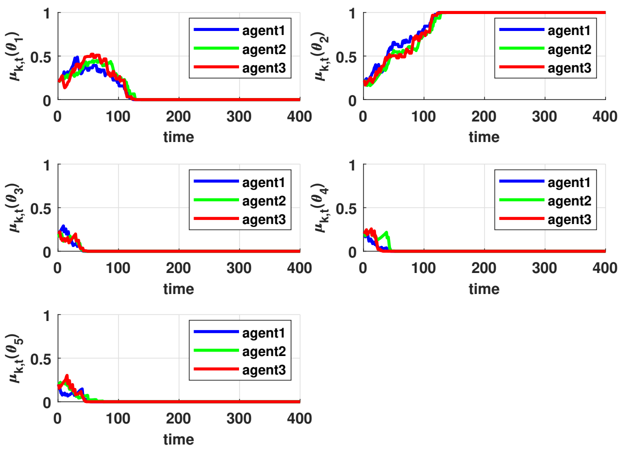

FIGURE 3. Convergence of agents' beliefs in the strongly connected network to θ_2 at $\eta = 0.1$ at the 50^{th} iteration using Algorithm 1.

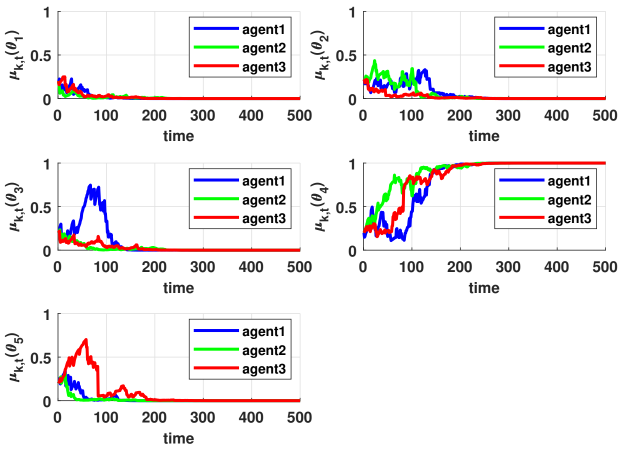

FIGURE 4. Convergence of agents' beliefs in the strongly connected network to θ_4 at $\eta = 0.1$ at the 1^{st} iteration using Algorithm 2.

are more likely to make predictions with a stock price that is the most frequently occurring. The beliefs of the agents are one at the most dominant true state and zero at every other state. Hence, from Fig. 2, the most dominant true state is θ_4, and the convergence time is $t = 127$.

Fig. 3 shows the convergence of the agents' beliefs to the most dominant true state at the 50^{th} iteration when Algorithm 1 is used. The most dominant true state is θ_2, and the convergence time is $t = 132$. The most dominant true state in Fig. 3 is different from the most dominant true state in Fig. 2, due to the randomness of the sequence $\theta_1^*, \cdots, \theta_T^*$ over the number of iterations.

Fig. 4 shows the convergence of the agents' beliefs to θ_4 at time $t = 251$ at the 1^{st} iteration. Here, Algorithm 2 is used. θ_4 is the most dominant true state for this iteration. Also, in Fig. 5, the agents beliefs' converge to θ_2 at time $t = 306$ at the 50^{th} iteration, when Algorithm 2 is used. θ_2 is the most dominant true state for this iteration. Again, the most dominant true state varies with the number of iterations due

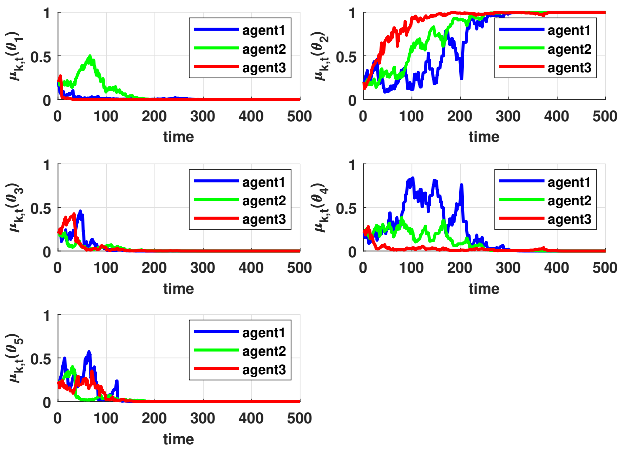

FIGURE 5. Convergence of agents' beliefs in the strongly connected network to θ_2 at $\eta = 0.1$ at the 50^{th} iteration using Algorithm 2.

to the randomness of the sequence of time-varying true states.

The speed of convergence for the agents in Fig. 2 and Fig. 3 tends to be faster than in Fig. 4 and Fig. 5. This is due to the privacy-utility trade-off discussed in Section III. Increasing privacy in Algorithm 2 led to much slower convergence compared to Algorithm 1. Careful analysis of the convergence of Algorithm 2 shows that it is much slower because much Laplace noise is added to the algorithm. Intuitively, Laplace noise is added independently to each of $p_{k,t}(\theta_1), \cdots, p_{k,t}(\theta_5)$ for the agent k. Then, Laplace noise from the intermediate probabilities of its two neighbors is added to the agent k, when computing the consensus probability. Again, another Laplace noise is added to the incurred loss. All these happen in a single round.

The speed of convergence for Algorithm 1 is nearly 23% slower on average when compared to the speed of convergence of the non-private algorithm in [16]. However, algorithm 2 is over 56% slower on average, than the non-private algorithm. The speed of convergence can be improved by increasing the learning rate η, or increasing the value of privacy parameter ϵ.

The regret bound for the non-private algorithm in [16] grows at a sublinear rate of $O(\sqrt{t}/t)$ with sublinearity defined as $lim_{t\to\infty}\frac{\mathbb{E}[Reg_t]}{t}$; similarly, the regret bounds for Algorithms 1 and 2 grow at the rate of $O(\sqrt{t}/t\epsilon)$ and $O(t^{3/2}/t\epsilon)$ respectively. Fig. 6 shows the regret bounds for the non-private algorithm, Algorithm 1 and Algorithm 2. It should be noted that by letting $\epsilon = 1$, which means privacy is lost completely, Algorithm 1 grows at the same rate as the non-private algorithm. From Fig. 6, the non-private algorithm has the least regret, while Algorithm 2 has the highest regret. However, the cost of additional regret incurred by using both Algorithm 1 and Algorithm 2 is a worthy trade-off to the privacy gains, especially in social networks where privacy is a challenge.

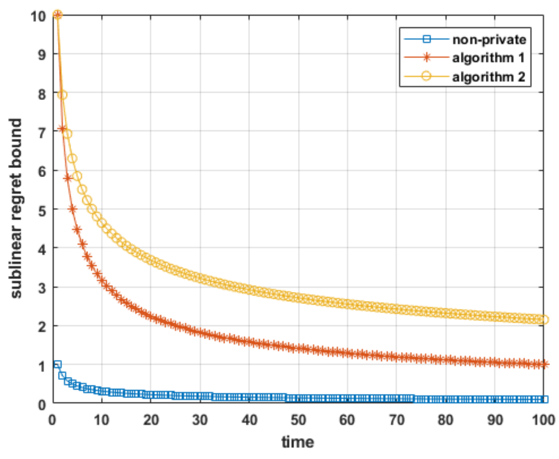

FIGURE 6. Regret bounds of the non-private algorithm, algorithm 1 and algorithm 2 with $\epsilon = 0.1$.

VIII. CONCLUSION

This paper addressed privacy concerns in the social network from a theoretical standpoint. The social network is modeled as a graph network consisting of a set of agents representing the social network users, and a set of edges representing the interactions among the agents. The goal of the agents is to learn an arbitrarily time-varying true state of the network with a privacy guarantee. To achieve this, this paper combined non-Bayesian learning, differential privacy, and multi-armed bandit seamlessly for the first time. Two algorithms were proposed. The first algorithm guaranteed differential privacy using the Laplace mechanism against a third-party intruder when there is no spy among the agents. The second algorithm also applied the Laplace mechanism to guarantee differential privacy against both a third-party intruder and a spying agent. The simulation results showed that continuous interactions among the agents using the proposed algorithms help the agents converge to this most dominant true state. The most dominant true state appears to be the most stable state from the sequence of arbitrarily time-varying true states over the time horizon. The speed of convergence for Algorithms 1 and 2 are compared with the speed of convergence for the non-private algorithm in the literature. The speed of convergence for Algorithm 2 is much slower than the speed of convergence for Algorithm 1, due to more Laplace noise added to Algorithm 2. However, increasing the learning rate or increasing the value of the privacy parameter will improve its speed of convergence. The regret bounds of the proposed algorithms are compared to the regret bound of the non-private algorithm in the literature.

This work can be extended to a graph network with weakly connected agents. The simulation results in this paper are prototypes of a large massive graph network. Thus, the simulation can be repeated for large graph networks, with more number agents, to represent a practical social network. Also, the simulation can be done with real datasets.

ACKNOWLEDGMENT

The author thanks Dr. Gergely Zaruba for his support of this manuscript. Also, the author thanks the anonymous reviewers of this manuscript.

.

APPENDIX A PROOF OF THE NON-PRIVATE BASE ALGORITHM

The non-private algorithm in [16] is repeated here for convenience as Algorithm 3.

The definition of regret in (10) is slightly different from the definition of regret in [16]. The $p_{k,t}(\theta)$ in the definition of the regret in [16] is replaced with $\mu_{k,t}(\theta)$ in (10). Hence, we will compute the regret bound. However, the proof shows that the upper bound on the regret remains unchanged. We start as follows:

Let $\mu_{k,t}(\theta) = \frac{w_{k,t}(\theta)}{W_{k,t}}$ and
$W_{k,t} = \sum_{\theta\in\Theta} w_{k,t-1}(\theta)\exp\left(-\eta\hat{l}_{k,t}(\theta)\right)$.

$$\frac{W_{k,t}}{W_{k,t-1}} = \frac{\sum_{\theta\in\Theta} w_{k,t-1}(\theta)\exp\left(-\eta\hat{l}_{k,t}(\theta)\right)}{W_{k,t-1}}$$
$$= \sum_{\theta\in\Theta}\mu_{k,t-1}(\theta)\exp\left(-\eta\hat{l}_{k,t}(\theta)\right)$$
$$\leq \sum_{\theta\in\Theta}\mu_{k,t-1}(\theta)\left(1-\eta\hat{l}_{k,t}(\theta)+\eta^2\hat{l}^2_{k,t}\right)$$

but $e^x \leq 1 + x + x^2$ for all $x \leq 1$. Therefore,

$$\leq 1-\eta\sum_{\theta\in\Theta}\mu_{k,t-1}(\theta)\hat{l}_{k,t}(\theta)+\eta^2\sum_{\theta\in\Theta}\mu_{k,t-1}(\theta)\hat{l}^2_{k,t}(\theta) \tag{21}$$

using $\sum_{\theta\in\Theta}\mu_{k,t-1}(\theta) \leq 1$ in (21).
Using $\ln(1-x) \leq -x$,

$$\ln\frac{W_{k,t}}{W_{k,t-1}} = \ln\Bigg(1-\eta\sum_{\theta\in\Theta}\mu_{k,t-1}(\theta)\hat{l}_{k,t}(\theta) + \eta^2\sum_{\theta\in\Theta}\mu_{k,t-1}(\theta)\hat{l}^2_{k,t}(\theta)\Bigg)$$
$$\leq -\eta\sum_{\theta\in\Theta}\mu_{k,t-1}(\theta)\hat{l}_{k,t}(\theta) + \eta^2\sum_{\theta\in\Theta}\mu_{k,t-1}(\theta)\hat{l}^2_{k,t}(\theta).$$

Sum over $t = 1, \cdots, T$

$$\ln\frac{W_{k,T}}{W_0} \leq \sum_{t=1}^{T}\sum_{\theta\in\Theta}\Bigg(-\eta\mu_{k,t-1}(\theta)\hat{l}_{k,t}(\theta)+ \eta^2\mu_{k,t-1}(\theta)\hat{l}^2_{k,t}(\theta)\Bigg). \tag{22}$$

Also, for any fixed $\theta_f \in \Theta$,

```
Algorithm 3: Non-Private Multi-Armed Algorithm for Strongly Connected Network
Parameters: Feedback graph, learning rate $\eta > 0$.
$V$ is the set of strongly connected agents and $E$ is the set of edges.
Exploration parameter $\gamma \in (0, \frac{1}{2}]$
Initialize $\mu_{k,0}(\theta) = \frac{1}{M}$
For each round $t \in \{1, \cdots, T\}$
Compute $p_{k,t}(\theta) = (1-\gamma)\mu_{k,t-1}(\theta) + \gamma\psi_{k,t}(\theta)\ \forall\ \theta \in \Theta$
Compute $P_{k,t}(\theta) = \sum_{j\in\mathcal{N}_k} a_{jk} p_{j,t}(\theta), \quad P_{k,t} = (P_{k,t}(\theta_1), ..., P_{k,t}(\theta_M))$
Draw state $\theta_t \sim P_{k,t}$ and incur loss $l_{k,t}(\theta_t) \in [0,1]$
Compute
    $\hat{l}_{k,t}(\theta) = \frac{l_{k,t}(\theta)}{P_{k,t}(\theta)} \mathbb{I}\{\theta = \theta_t\} \quad \forall\theta \in \Theta$
Update
    $\mu_{k,t}(\theta) = \frac{\mu_{k,t-1}(\theta)\exp\left(-\eta\hat{l}_{k,t}(\theta)\right)}{\sum_{\theta'\in\Theta}\mu_{k,t-1}(\theta')\exp\left(-\eta\hat{l}_{k,t}(\theta')\right)} \quad \forall\theta \in \Theta$
end
```

$$\ln\frac{W_{k,T}}{W_{k,0}} \geq \ln\frac{w_{k,T}(\theta_f)}{W_{k,0}} = -\eta\sum_{t=1}^{T}\hat{l}_{k,t}(\theta_f) - \ln W_{k,0}. \tag{23}$$

with $W_{k,0} = M$, and equating (22) and (23),

$$\sum_{t=1}^{T}\sum_{\theta\in\Theta}\left(-\eta\mu_{k,t-1}(\theta)\hat{l}_{k,t}(\theta) + \eta^2\mu_{k,t-1}(\theta)\hat{l}^2_{k,t}(\theta)\right) \geq -\eta\sum_{t=1}^{T}\hat{l}_{k,t}(\theta_f) - \ln M.$$

Hence,

$$\sum_{t=1}^{T}\sum_{\theta\in\Theta}\left(-\eta\mu_{k,t-1}(\theta)\hat{l}_{k,t}(\theta) + \eta^2\mu_{k,t-1}(\theta)\hat{l}^2_{k,t}(\theta)\right) \geq -\eta\min_{\theta_f\in\Theta}\sum_{t=1}^{T}\hat{l}_{k,t}(\theta_f) - \ln M.$$

Therefore,

$$\sum_{t=1}^{T}\sum_{\theta\in\Theta}\eta\mu_{k,t-1}(\theta)\hat{l}_{k,t}(\theta) - \eta\min_{\theta_f\in\Theta}\sum_{t=1}^{T}\hat{l}_{k,t}(\theta_f) \leq \sum_{t=1}^{T}\sum_{\theta\in\Theta}\eta^2\mu_{k,t-1}(\theta)\hat{l}^2_{k,t}(\theta) + \ln M.$$

Take conditional expectation,

$$\mathbb{E}_{\mathcal{F}_t/\mathcal{F}_{t-1}}\left[\sum_{t=1}^{T}\sum_{\theta\in\Theta}\mu_{k,t-1}(\theta)\hat{l}_{k,t}(\theta) - \min_{\theta_f\in\Theta}\sum_{t=1}^{T}\hat{l}_{k,t}(\theta_f)\middle|\mathcal{F}_{t-1}\right] \leq \mathbb{E}_{\mathcal{F}_t/\mathcal{F}_{t-1}}\left[\sum_{t=1}^{T}\sum_{\theta\in\Theta}\eta\mu_{k,t-1}(\theta)\hat{l}^2_{k,t}(\theta)\middle|\mathcal{F}_{t-1}\right] + \frac{\ln M}{\eta}.$$

Therefore,

$$\sum_{t=1}^{T}\sum_{\theta\in\Theta}\mu_{k,t-1}(\theta)\mathbb{E}_{\mathcal{F}_t/\mathcal{F}_{t-1}}\left[\hat{l}_{k,t}(\theta)\middle|\mathcal{F}_{t-1}\right] - \min_{\theta_f\in\Theta}\sum_{t=1}^{T}\mathbb{E}_{\mathcal{F}_t/\mathcal{F}_{t-1}}\left[\hat{l}_{k,t}(\theta_f|\mathcal{F}_{t-1})\right] \leq \sum_{t=1}^{T}\sum_{\theta\in\Theta}\eta\mu_{k,t-1}(\theta)\mathbb{E}_{\mathcal{F}_t/\mathcal{F}_{t-1}}\left[\hat{l}^2_{k,t}(\theta)\middle|\mathcal{F}_{t-1}\right] + \frac{\ln M}{\eta}.$$

Using the fact that the expectation of the estimated loss yields the true loss similar to (15),

$$\sum_{t=1}^{T}\sum_{\theta\in\Theta}\mu_{k,t-1}(\theta)l_{k,t}(\theta) - \sum_{t=1}^{T}l_{k,t}(\theta^\bullet) \leq \eta\sum_{t=1}^{T}\sum_{\theta\in\Theta}\mu_{k,t-1}(\theta)\frac{l^2_{k,t}(\theta)}{P_{k,t}(\theta)} + \frac{\ln M}{\eta}.$$

Hence,

$$\sum_{t=1}^{T}\sum_{\theta\in\Theta}\mu_{k,t-1}(\theta)l_{k,t}(\theta) - \sum_{t=1}^{T}l_{k,t}(\theta^\bullet) \leq \eta\sum_{t=1}^{T}\sum_{\theta\in\Theta}\mu_{k,t-1}(\theta)\frac{1}{P_{k,t}(\theta)} + \frac{\ln M}{\eta} \tag{24}$$

using $l_{k,t}(\theta) \leq 1$ in (24). From algorithm 3

$$p_{k,t}(\theta) \geq (1-\gamma)\mu_{k,t-1}(\theta) \geq \frac{1}{2}\mu_{k,t-1}(\theta).$$

Hence,

$$\sum_{t=1}^{T}\sum_{\theta\in\Theta}\mu_{k,t-1}(\theta)l_{k,t}(\theta) - \sum_{t=1}^{T}l_{k,t}(\theta^\bullet) \leq \eta\sum_{t=1}^{T}\sum_{\theta\in\Theta}p_{k,t}(\theta)\frac{2}{P_{k,t}(\theta)} + \frac{\ln M}{\eta}. \tag{25}$$

Applying Lemma 2 in [16] with $\varepsilon = \frac{\gamma}{M}$

$$\sum_{t=1}^{T}\sum_{\theta\in\Theta}\mu_{k,t-1}(\theta)l_{k,t}(\theta)-\sum_{t=1}^{T}l_{k,t}\left(\theta^{\bullet}\right) \le 8\alpha\eta T\ln\frac{4M^2}{\alpha\gamma}+\frac{\ln M}{\eta}. \tag{26}$$

Then,

$$\mathbb{E}_{\mathcal{F}_t}\left[\sum_{t=1}^{T}\sum_{\theta\in\Theta}\mu_{k,t}(\theta)l_{k,t}(\theta)-\sum_{t=1}^{T}l_{k,t}\left(\theta^{\bullet}\right)\right]\le 8\alpha\eta T\ln\frac{4M^2}{\alpha\gamma}+\frac{\ln M}{\eta}.$$

Choose

$$\gamma=\min\left\{8\alpha\eta,\frac{1}{2}\right\},\eta=\left(\frac{\ln M}{\alpha T}\right)^{1/2}.$$

Then, the upper bound is obtained as

$$O\left(\sqrt{\alpha T\ln M}\right).$$

The effect of the self-loop weight will reduce the regret as each agent will assign more weight to its probability in the computation of its consensus probability.

REFERENCES

[1] C. Zhang, J. Sun, X. Zhu, and Y. Fang, "Privacy and security for online social networks: challenges and opportunities," IEEE network, vol. 24, no. 4, pp. 13–18, 2010.

[2] C. Dwork, A. Roth et al., "The algorithmic foundations of differential privacy." Foundations and Trends in Theoretical Computer Science, vol. 9, no. 3-4, pp. 211–407, 2014.

[3] C. Task and C. Clifton, "A guide to differential privacy theory in social network analysis," in 2012 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining. IEEE, 2012, pp. 411–417.

[4] C. Dwork, "Differential privacy: A survey of results," in International conference on theory and applications of models of computation. Springer, 2008, pp. 1–19.

[5] C. Dwork, F. McSherry, K. Nissim, and A. Smith, "Calibrating noise to sensitivity in private data analysis," in Theory of cryptography conference. Springer, 2006, pp. 265–284.

[6] J. C. Duchi, M. I. Jordan, and M. J. Wainwright, "Local privacy and statistical minimax rates," in 2013 IEEE 54th Annual Symposium on Foundations of Computer Science. IEEE, 2013, pp. 429–438.

[7] U. Brandes, L. C. Freeman, and D. Wagner, Social networks, 2013.

[8] A. Anagnostopoulos, R. Kumar, and M. Mahdian, "Influence and correlation in social networks," in Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining, 2008, pp. 7–15.

[9] A. Jadbabaie, P. Molavi, A. Sandroni, and A. Tahbaz-Salehi, "Non-bayesian social learning," Games and Economic Behavior, vol. 76, no. 1, pp. 210–225, 2012.

[10] S. Shahrampour, S. Rakhlin, and A. Jadbabaie, "Online learning of dynamic parameters in social networks," in Advances in Neural Information Processing Systems, 2013.

[11] R. M. Frongillo, G. Schoenebeck, and O. Tamuz, "Social learning in a changing world," in International Workshop on Internet and Network Economics. Springer, 2011, pp. 146–157.

[12] G. Moscarini, M. Ottaviani, and L. Smith, "Social learning in a changing world," Economic Theory, vol. 11, no. 3, pp. 657–665, 1998.

[13] K. Dasaratha, B. Golub, and N. Hak, "Social learning in a dynamic environment," arXiv preprint arXiv:1801.02042, 2018.

[14] O. T. Odeyomi, "Learning the truth by weakly connected agents in social networks using multi-armed bandit," IEEE Access, vol. 8, pp. 202 090–202 099, 2020.

[15] O. T. Odeyomi, H. M. Kwon, and D. A. Murrell, "Time-varying truth prediction in social networks using online learning," in 2020 International Conference on Computing, Networking and Communications (ICNC). IEEE, 2020, pp. 171–175.

[16] O. T. Odeyomi, "Learning the truth in social networks using multi-armed bandit," IEEE Access, vol. 8, pp. 137 692–137 701, 2020.

[17] P. Zhou, K. Wang, J. Xu, and D. Wu, "Differentially-private and trustworthy online social multimedia big data retrieval in edge computing," IEEE Transactions on Multimedia, vol. 21, no. 3, pp. 539–554, 2018.

[18] P. Zhou, K. Wang, L. Guo, S. Gong, and B. Zheng, "A privacy-preserving distributed contextual federated online learning framework with big data support in social recommender systems," IEEE Transactions on Knowledge and Data Engineering, 2019.

[19] O. T. Odeyomi and G. Zaruba, "Differentially-private federated learning with long-term budget constraints using online lagrangian descent," in 2021 IEEE World AI IoT Congress (AIIoT). IEEE, 2021, pp. 0001–0006.

[20] H. Wang, Q. Zhao, Q. Wu, S. Chopra, A. Khaitan, and H. Wang, "Global and local differential privacy for collaborative bandits," in Fourteenth ACM Conference on Recommender Systems, 2020, pp. 150–159.

[21] S. Chen, Y. Tao, D. Yu, F. Li, B. Gong, and X. Cheng, "Privacy-preserving collaborative learning for multiarmed bandits in iot," IEEE Internet of Things Journal, vol. 8, no. 5, pp. 3276–3286, 2020.

[22] M. Hay, C. Li, G. Miklau, and D. Jensen, "Accurate estimation of the degree distribution of private networks," in 2009 Ninth IEEE International Conference on Data Mining. IEEE, 2009, pp. 169–178.

[23] V. Rastogi, M. Hay, G. Miklau, and D. Suciu, "Relationship privacy: output perturbation for queries with joins," in Proceedings of the twenty-eighth ACM SIGMOD-SIGACT-SIGART symposium on Principles of database systems, 2009, pp. 107–116.

[24] Y. Wang, X. Wu, J. Zhu, and Y. Xiang, "On learning cluster coefficient of private networks," Social network analysis and mining, vol. 3, no. 4, pp. 925–938, 2013.

[25] Y. Wang, X. Wu, and L. Wu, "Differential privacy preserving spectral graph analysis," in Pacific-Asia Conference on Knowledge Discovery and Data Mining. Springer, 2013, pp. 329–340.

[26] D. J. Mir and R. N. Wright, "A differentially private graph estimator," in 2009 IEEE International Conference on Data Mining Workshops. IEEE, 2009, pp. 122–129.

[27] P. Mahadevan, D. Krioukov, K. Fall, and A. Vahdat, "Systematic topology analysis and generation using degree correlations," ACM SIGCOMM Computer Communication Review, vol. 36, no. 4, pp. 135–146, 2006.

[28] A. Sala, X. Zhao, C. Wilson, H. Zheng, and B. Y. Zhao, "Sharing graphs using differentially private graph models," in Proceedings of the 2011 ACM SIGCOMM conference on Internet measurement conference, 2011, pp. 81–98.

[29] C. Dimitrakakis, B. Nelson, A. Mitrokotsa, and B. I. Rubinstein, "Robust and private bayesian inference," in International Conference on Algorithmic Learning Theory. Springer, 2014, pp. 291–305.

[30] C. Dimitrakakis, B. Nelson, Z. Zhang, A. Mitrokotsa, and B. Rubinstein, "Differential privacy in a bayesian setting through posterior sampling," arXiv preprint ArXiv:1306.1066, pp. 1–27, 2015.

[31] S. Zheng, "The differential privacy of bayesian inference," Ph.D. dissertation, 2015.

[32] Y.-X. Wang, S. Fienberg, and A. Smola, "Privacy for free: Posterior sampling and stochastic gradient monte carlo," in International Conference on Machine Learning, 2015, pp. 2493–2502.

[33] O. Williams and F. McSherry, "Probabilistic inference and differential privacy," Advances in Neural Information Processing Systems, vol. 23, pp. 2451–2459, 2010.

[34] P. Jain, P. Kothari, and A. Thakurta, "Differentially private online learning," in Conference on Learning Theory, 2012, pp. 24–1.

[35] J. D. Abernethy, Y. H. Jung, C. Lee, A. McMillan, and A. Tewari, "Online learning via the differential privacy lens," in Advances in Neural Information Processing Systems, 2019, pp. 8894–8904.

[36] N. Mishra and A. Thakurta, "(nearly) optimal differentially private stochastic multi-arm bandits," in Proceedings of the Thirty-First Conference on Uncertainty in Artificial Intelligence, 2015, pp. 592–601.

[37] A. Tossou and C. Dimitrakakis, "Algorithms for differentially private multi-armed bandits," arXiv preprint arXiv:1511.08681, 2015.

[38] T. Sajed and O. Sheffet, "An optimal private stochastic-mab algorithm based on an optimal private stopping rule," arXiv preprint arXiv:1905.09383, 2019.

[39] X. Chen, K. Zheng, Z. Zhou, Y. Yang, W. Chen, and L. Wang, "(locally) differentially private combinatorial semi-bandits," arXiv preprint arXiv:2006.00706, 2020.

[40] A. C. Tossou and C. Dimitrakakis, "Achieving privacy in the adversarial multi-armed bandit," arXiv preprint arXiv:1701.04222, 2017.

[41] N. Agarwal and K. Singh, "The price of differential privacy for online learning," arXiv preprint arXiv:1701.07953, 2017.

[42] D. Basu, C. Dimitrakakis, and A. Tossou, "Differential privacy for multi-armed bandits: What is it and what is its cost?" arXiv preprint arXiv:1905.12298, 2019.

[43] W. Ren, X. Zhou, J. Liu, and N. B. Shroff, "Multi-armed bandits with local differential privacy," arXiv preprint arXiv:2007.03121, 2020.

[44] A. H. Sayed, "Adaptation, learning, and optimization over networks," Foundations and Trends in Machine Learning, vol. 7, no. article, pp. 311–801, 2014.

[45] D. van der Hoeven, "User-specified local differential privacy in unconstrained adaptive online learning." in NeurIPS, 2019, pp. 14 080–14 089.

[46] P. Auer, N. Cesa-Bianchi, Y. Freund, and R. E. Schapire, "Gambling in a rigged casino: The adversarial multi-armed bandit problem," in Proceedings of IEEE 36th Annual Foundations of Computer Science. IEEE, 1995, pp. 322–331.