Decentralized Multi-Agent Linear Bandits with Safety Constraints

Sanae Amani,[1] **Christos Thrampoulidis** [2]
[1] University of California, Los Angeles
[2] University of British Columbia, Vancouver
samani@ucla.edu, cthrampo@ece.ubc.ca

Abstract

We study decentralized stochastic linear bandits, where a network of N agents acts cooperatively to efficiently solve a linear bandit-optimization problem over a d-dimensional space. For this problem, we propose DLUCB: a fully decentralized algorithm that minimizes the cumulative regret over the entire network. At each round of the algorithm each agent chooses its actions following an upper confidence bound (UCB) strategy and agents share information with their immediate neighbors through a carefully designed consensus procedure that repeats over cycles. Our analysis adjusts the duration of these communication cycles ensuring near-optimal regret performance $\mathcal{O}(d\log NT\sqrt{NT})$ at a communication rate of $\mathcal{O}(dN^2)$ per round. The structure of the network affects the regret performance via a small additive term – coined the regret of delay – that depends on the spectral gap of the underlying graph. Notably, our results apply to arbitrary network topologies without a requirement for a dedicated agent acting as a server. In consideration of situations with high communication cost, we propose RC-DLUCB: a modification of DLUCB with rare communication among agents. The new algorithm trades off regret performance for a significantly reduced total communication cost of $\mathcal{O}(d^3N^{2.5})$ over all T rounds. Finally, we show that our ideas extend naturally to the emerging, albeit more challenging, setting of safe bandits. For the recently studied problem of linear bandits with unknown linear safety constraints, we propose the first safe decentralized algorithm. Our study contributes towards applying bandit techniques in safety-critical distributed systems that repeatedly deal with unknown stochastic environments. We present numerical simulations for various network topologies that corroborate our theoretical findings.

1 Introduction

Linear stochastic bandits (LB) provide simple, yet commonly encountered, models for a variety of sequential decision-making problems under uncertainty. Specifically, LB generalizes the classical multi-armed bandit (MAB) problem of K arms that each yields reward sampled independently from an underlying distribution with unknown parameters, to a setting where the expected reward of each arm is a linear function that depends on the same unknown parameter vector (Dani, Hayes, and Kakade 2008; Abbasi-Yadkori, Pál, and Szepesvári 2011; Rusmevichientong and Tsitsiklis 2010). LBs have been successfully applied over the years in online advertising, recommendation services, resource allocation, etc. (Lattimore and Szepesvári 2018). More recently, researchers have explored the potentials of such algorithms in more complex systems, such as in robotics, wireless networks, the power grid, medical trials, e.g., (Li et al. 2013; Avner and Mannor 2019; Berkenkamp, Krause, and Schoellig 2016; Sui et al. 2018). A distinguishing feature of many of these –perhaps less conventional– bandit applications, is their *distributive* nature. For example, in sensor/wireless networks (Avner and Mannor 2019), a collaborative behavior is required for decision-makers/agents to select better actions as individuals, but each of them is only able to share information about the unknown environment with a subset of neighboring agents. While a distributed nature is inherent in certain systems, distributed solutions might also be preferred in broader settings, as they can lead to speed-ups of the learning process. This calls for extensions of the traditional bandit setting to networked systems. At the same time, in many of these applications the unknown system might be *safety-critical*, i.e., the algorithm's chosen actions need to satisfy certain constraints that, importantly, are often unknown. This leads to the challenge of balancing the goal of reward maximization with the restriction of playing safe actions. The past few years have seen a surge of research activity in these two areas: (i) distributed (Wang et al. 2019; Martínez-Rubio, Kanade, and Rebeschini 2019; Szörényi et al. 2013; Landgren, Srivastava, and Leonard 2016a); and (ii) safe bandits (Sui et al. 2015, 2018; Kazerouni et al. 2017; Amani, Alizadeh, and Thrampoulidis 2019; Moradipari et al. 2019; Khezeli and Bitar 2019; Pacchiano et al. 2020).

This paper contributes to the intersection of these two emerging lines of work. Concretely, we consider the problem of decentralized multi-agent linear bandits for a general (connected) network structure of N agents, who can only communicate messages with their immediate neighbors. For this, we propose and analyze the first fully-decentralized algorithm. We also present a communication-efficient version and discuss key trade-offs between regret, communication cost and graph structure. Finally, we present the first simultaneously distributed and safe bandit algorithm for a setting with unknown linear constraints.

Notation. We use lower-case letters for scalars, lower-case bold letters for vectors, and upper-case bold letters for matrices. The Euclidean-norm of $\mathbf{x}$ is denoted by $\|\mathbf{x}\|_2$. We denote the transpose of any column vector $\mathbf{x}$ by $\mathbf{x}^T$. For any vectors $\mathbf{x}$ and $\mathbf{y}$, we use $\langle \mathbf{x}, \mathbf{y} \rangle$ to denote their inner product. Let $\mathbf{A}$ be a positive definite $d \times d$ matrix and $\boldsymbol{\nu} \in \mathbb{R}^d$. The weighted 2-norm of $\boldsymbol{\nu}$ with respect to $\mathbf{A}$ is defined by $\|\boldsymbol{\nu}\|_{\mathbf{A}} = \sqrt{\boldsymbol{\nu}^T \mathbf{A} \boldsymbol{\nu}}$. For positive integers n and $m \leq n$, $[n]$ and $[m : n]$ denote the sets $\{1, 2, \ldots, n\}$ and $\{m, \ldots, n\}$, respectively. We use $\mathbf{1}$ and $\mathbf{e}_i$ to denote the vector of all 1's and the i-th standard basis vector, respectively.

1.1 Problem Formulation

Decentralized Linear Bandit. We consider a network of N agents and known convex compact decision set $\mathcal{D} \subset \mathbb{R}^d$ (our results can be easily extended to settings with time varying decision sets). Agents play actions synchronously. At each round t, each agent i chooses an action $\mathbf{x}_{i,t} \in \mathcal{D}$ and observes reward $y_{i,t} = \langle \boldsymbol{\theta}_*, \mathbf{x}_{i,t} \rangle + \eta_{i,t}$, where $\boldsymbol{\theta}_* \in \mathbb{R}^d$ is an unknown vector and $\eta_{i,t}$ is random additive noise.

Communication Model. The agents are represented by the nodes of an undirected and connected graph G. Each agent can send and receive messages only to and from its immediate neighbors. The topology of G is known to all agents via a communication matrix $\mathbf{P}$ (see Assumption 1).

Safety. The learning environment might be subject to unknown constraints that restrict the choice of actions. In this paper, we model the safety constraint by a linear function depending on an *unknown* vector $\boldsymbol{\mu}_* \in \mathbb{R}^d$ and a *known* constant $c \in \mathbb{R}$. Specifically, the chosen action $\mathbf{x}_{i,t}$ must satisfy $\langle \boldsymbol{\mu}_*, \mathbf{x}_{i,t} \rangle \leq c$, for all i and t, with high probability. We define the unknown safe set as $\mathcal{D}^{\mathrm{s}}(\boldsymbol{\mu}_*) := \{\mathbf{x} \in \mathcal{D} : \langle \boldsymbol{\mu}_*, \mathbf{x} \rangle \leq c\}$. After playing $\mathbf{x}_{i,t}$, agent i observes bandit-feedback measurements $z_{i,t} = \langle \boldsymbol{\mu}_*, \mathbf{x}_{i,t} \rangle + \zeta_{i,t}$. This type of safety constraint, but for single-agent settings, has been recently introduced and studied in (Amani, Alizadeh, and Thrampoulidis 2019; Pacchiano et al. 2020; Sui et al. 2015, 2018; Moradipari et al. 2019). See also (Kazerouni et al. 2017; Khezeli and Bitar 2019) for related notions of safety studied recently in the context of single-agent linear bandits.

Goal. Let T be the total number of rounds. We define the cumulative regret of the entire network as: $R_T := \sum_{t=1}^{T} \sum_{i=1}^{N} \langle \boldsymbol{\theta}_*, \mathbf{x}_* \rangle - \langle \boldsymbol{\theta}_*, \mathbf{x}_{i,t} \rangle$. The optimal action $\mathbf{x}_*$ is defined with respect to $\mathcal{D}$ and $\mathcal{D}^{\mathrm{s}}(\boldsymbol{\mu}_*)$ as $\arg\max_{\mathbf{x} \in \mathcal{D}} \langle \boldsymbol{\theta}_*, \mathbf{x} \rangle$ and $\arg\max_{\mathbf{x} \in \mathcal{D}^{\mathrm{s}}(\boldsymbol{\mu}_*)} \langle \boldsymbol{\theta}_*, \mathbf{x} \rangle$ in the original and safe settings, respectively. The goal is to minimize the cumulative regret, while each agent is allowed to share $poly(Nd)$ values per round to its neighbors. Specifically, we wish to achieve a regret close to that incurred by an optimal *centralized algorithm for NT rounds* (the total number of plays). In the presence of safety constraint, in addition to the aforementioned goals, agents' actions must also satisfy the safety constraint at each round.

1.2 Contributions

DLUCB. We propose a fully decentralized linear bandit algorithm (DLUCB), at each round of which, the agents simultaneously share information among each other and pick their next actions. We prove a regret bound that captures both the degree of selected actions' optimality and the inevitable delay in information-sharing due to the network structure. See Sec. 2.1 and 2.2. Compared to existing distributed LB algorithms, ours can be implemented (and remains valid) for any arbitrary (connected) network without requiring a peer-to-peer network structure or a master node. See Sec. 2.4.

RC-DLUCB. We propose a fully decentralized algorithm with rare communication (RC-DLUCB) to reduce the communication cost (total number of values communicated during the run of algorithm) for applications that are sensitive to high communication cost. See Sec. 2.3

Safe-DLUCB. We present and analyze the first fully decentralized algorithm for *safe* LBs with linear constraints. Our algorithm provably achieves regret of the same order (wrt. NT) as if no constraints were present. See Sec. 3 We complement our theoretical results with numerical simulations under various settings in Sec. 4.

1.3 Related Works

Decentralized Bandits. There are several recent works on decentralized/distributed stochastic MAB problems. In the context of the classical K-armed MAB, (Martínez-Rubio, Kanade, and Rebeschini 2019; Landgren, Srivastava, and Leonard 2016a,b) proposed decentralized algorithms for a network of N agents that can share information only with their immediate neighbors, while (Szörényi et al. 2013) studies the MAB problem on peer-to-peer networks. More recently, (Wang et al. 2019) focuses on communication efficiency and presented K-armed MAB algorithms with significantly lower communication overhead. In contrast to these, here, we study a LB model. The most closely related works on distributed/decentralized LB are (Wang et al. 2019) and (Korda, Szörényi, and Shuai 2016). In (Wang et al. 2019), the authors present a communication-efficient algorithm that operates under the coordination of a central server, such that every agent has instantaneous access to the full network information through the server. This model differs from the fully decentralized one considered here. In another closely related work, (Korda, Szörényi, and Shuai 2016) studies distributed LBs in peer-to-peer networks, where each agent can only send information to one other randomly chosen agent, not necessarily its neighbor, per round. A feature, in common with our algorithm, is the delayed use of bandit feedback, but the order of the delay differs between the two, owing to the different model. Please also see Sec. 2.4 for a more elaborate comparison. To recap, even-though motivated by the aforementioned works, our paper presents the first *fully decentralized algorithm* for the *multi-agent LB problem* on a general network topology, with communication between any two neighbors in the network. Furthermore, non of the above has studied the presence of safety constraints.

Safe Bandits. In a more general context, the notion of safe learning has many diverse definitions in the literature. Specifically, safety in bandit problems has itself received significant attention in recent years, e.g. (Sui et al. 2015, 2018; Kazerouni et al. 2017; Amani, Alizadeh, and Thram-

poulidis 2019; Moradipari et al. 2019; Khezeli and Bitar 2019; Pacchiano et al. 2020). To the best of our knowledge, all existing works on MAB/LB problems with safety constraints study a single-agent. As mentioned in Sec. 1.1, the multi-agent safe LB studied here is a canonical extension of the single-agent setting studied in (Amani, Alizadeh, and Thrampoulidis 2019; Moradipari et al. 2019; Pacchiano et al. 2020). Accordingly, our algorithm and analysis builds on ideas introduced in this prior work and extends them to multi-agent collaborative learning.

2 Decentralized Linear Algorithms

In this section, we present *Decentralized Linear Upper Confidence Bound* (DLUCB). Starting with a high-level description of the gossip communication protocol and of the benefits and challenges it brings to the problem in Sec. 2.1, we then explain DLUCB Algorithm 1 in Sec. 2.2. In Sec. 2.3 we present a communication-efficient version of DLUCB. Finally, in Sec. 2.4 we compare our algorithms to prior art. Throughout this section, we do *not* assume any safety constraints. Below, we introduce some necessary assumptions.

Assumption 1 (Communication Matrix)**.** *For an undirected connected graph G with N nodes, $\mathbf{P} \in \mathbb{R}^{N\times N}$ is a symmetric communication matrix if it satisfies the following three conditions: (i) $\mathbf{P}_{i,j} = 0$ if there is no connection between nodes i and j; (ii) the sum of each row and column of $\mathbf{P}$ is 1; (iii) the eigenvalues are real and their magnitude is less than 1, i.e., $1 = |\lambda_1| > |\lambda_2| \geq \ldots |\lambda_N| \geq 0$. We assume that agents have knowledge of communication matrix $\mathbf{P}$.*

We remark that $\mathbf{P}$ can be constructed with little global information about the graph, such as its adjacency matrix and the graph's maximal degree; see Sec. 4 for an explicit construction. Once $\mathbf{P}$ is known, the total number of agents N and the graph's spectral gap $1 - |\lambda_2|$ are also known. We show in Sec. 2.2 that the latter two parameters fully capture how the network structure affects the algorithm's regret.

Assumption 2 (Subgaussian Noise)**.** *For $i \in [N]$ and $t > 0$, $\eta_{i,t}$, $\zeta_{i,t}$ are zero-mean σ-subGaussian random variables.*

Assumption 3 (Boundedness)**.** *Without loss of generality, $\|\mathbf{x}\|_2 \leq 1$ for all $\mathbf{x} \in \mathcal{D}$, $\|\boldsymbol{\theta}_*\|_2 \leq 1$, and $\|\boldsymbol{\mu}_*\|_2 \leq 1$.*

2.1 Information-Sharing Protocol

DLCUB implements a UCB strategy. At the core of single-agent UCB algorithms, is the construction of a proper confidence set around the true parameter $\boldsymbol{\theta}_*$ using past actions and their observed rewards. In multi-agent settings, each agent $i \in [N]$ maintains their own confidence set $\mathcal{C}_{i,t}$ at every round t. To exploit the network structure and enjoy the benefits of collaborative learning, it is important that $\mathcal{C}_{i,t}$ is built using information about past actions of not only agent i itself, but also of agents $j \neq i \in [N]$. For simplicity, we consider first a centralized setting of perfect information-sharing among agents. Specifically, assume that at every round t, agent i knows the past chosen actions and their observed rewards by *all* other agents in the graph. Having gathered all this information, each agent i maintains knowledge of the following sufficient statistics during all rounds t:

$$\mathbf{A}_{*,t} = \lambda I + \sum_{\tau=1}^{t-1}\sum_{i=1}^{N} \mathbf{x}_{i,\tau}\mathbf{x}_{i,\tau}^T, \;\; \mathbf{b}_{*,t} = \sum_{\tau=1}^{t-1}\sum_{i=1}^{N} y_{i,\tau}\mathbf{x}_{i,\tau}. \quad (1)$$

Here, $\lambda \geq 1$ is a regularization parameter. Of course, in this idealized scenario, the confidence set constructed based on (1) is the same for every agent. In fact, it is the same as the confidence set that would be constructed by a single-agent that is allowed to choose N actions at every round. Here, we study a decentralized setting with imperfect information sharing. In particular, each agent i can only communicate with its immediate neighbors $j \in \mathcal{N}(i)$ at any time t. As such, it does *not* have direct access to the "perfect statistics" $\mathbf{A}_{*,t}$ and $\mathbf{b}_{*,t}$ in (1). Instead, it is confined to approximations of them, which we denote by $\mathbf{A}_{i,t}$ and $\mathbf{b}_{i,t}$. At worst-case, where no communication is used, $\mathbf{A}_{i,t} = \lambda I + \sum_{\tau=1}^{t-1} \mathbf{x}_{i,\tau}\mathbf{x}_{i,\tau}^T$ (similarly for $\mathbf{b}_{i,t}$). But, this is a very poor approximation of $\mathbf{A}_{*,t}$ (correspondingly, $\mathbf{b}_{*,t}$). Our goal is to construct a communication scheme that exploits exchange of information among agents to allow for drastically better approximations of (1). Towards this goal, our algorithm implements an appropriate *gossip protocol* to communicate each agent's past actions and observed rewards to the rest of the network (even beyond immediate neighbors). We describe the details of this protocol next.

Running Consensus. In order to share information about agents' past actions among the network, we rely on *running consensus*, e.g., (Lynch 1996; Xiao and Boyd 2004). The goal of running consensus is that after enough rounds of communication, each agent has an accurate estimate of the average (over all agents) of the initial values of each agent. Precisely, let $\boldsymbol{\nu}_0 \in \mathbb{R}^N$ be a vector, where each entry $\boldsymbol{\nu}_{0,i}, i \in [N]$ represents agent's i information at some initial round. Then, running consensus aims at providing an accurate estimate of the average $\frac{1}{N}\sum_{i\in[N]} \boldsymbol{\nu}_{0,i}$ at each agent. Note that encoding $\boldsymbol{\nu}_0 = X\mathbf{e}_j$, allows all agents to eventually get an estimate of the value $X = \sum_{i\in[N]} \boldsymbol{\nu}_{0,i}$ that was initially known only to agent j. To see how this is relevant to our setting recall (1) and focus at $t = 2$ for simplicity. At round $t = 2$, each agent j only knows $\mathbf{x}_{j,1}$ and estimation of $\mathbf{A}_{*,2} = \sum_{i=1}^{N} \mathbf{x}_{i,1}\mathbf{x}_{i,1}^T$ by agent j boils down to estimating each $\mathbf{x}_{i,1}, i \neq j$. In our previous example, let X be k-th entry of $\mathbf{x}_{i,1}$ for some $i \neq j$. By running consensus on $\boldsymbol{\nu}_0 = [\mathbf{x}_{j,1}]_k\mathbf{e}_j$ for $k \in [d]$, every agent eventually builds an accurate estimate of $[\mathbf{x}_{j,1}]_k$, the k-th entry of $\mathbf{x}_{j,1}$ that would otherwise only be known to j. It turns out that the communication matrix $\mathbf{P}$ defined in Assumption 1 plays a key role in reaching consensus. The details are standard in the rich related literature (Xiao and Boyd 2004; Lynch 1996). Here, we only give a brief explanation of the high-level principles. Roughly speaking, a consensus algorithm updates $\boldsymbol{\nu}_0$ by $\boldsymbol{\nu}_1 = \mathbf{P}\boldsymbol{\nu}_0$ and so on. Note that this operation respects the network structure since the updated value $\boldsymbol{\nu}_{1,j}$ is a weighted average of only $\boldsymbol{\nu}_{0,j}$ itself and neighbor-only values $\boldsymbol{\nu}_{0,i}, i \in \mathcal{N}(j)$. Thus, after S rounds, agent j has access to entry j of $\boldsymbol{\nu}_S = \mathbf{P}^S\boldsymbol{\nu}_0$. This is useful because $\mathbf{P}$ is well-known to satisfy the following mixing prop-

erty: $\lim_{S\to\infty}\mathbf{P}^S = \mathbf{1}\mathbf{1}^T/N$ (Xiao and Boyd 2004). Thus, $\lim_{S\to\infty}[\boldsymbol{\nu}_S]_j = \frac{1}{N}\sum_{i=1}^N \boldsymbol{\nu}_{0,i}, \forall j \in [N]$, as desired. Of course, in practice, the number S of communication rounds is finite, leading to an ϵ-approximation of the average.

Accelerated-Consensus. In this paper, we adapt *polynomial filtering* introduced in (Martínez-Rubio, Kanade, and Rebeschini 2019; Seaman et al. 2017) to speed up the mixing of information by following an approach whose convergence rate is faster than the standard multiplication method above. Specifically, after S communication rounds, instead of $\mathbf{P}^S$, agents compute and apply to the initial vector $\boldsymbol{\nu}_0$ an appropriate re-scaled *Chebyshev polynomial* $q_S(\mathbf{P})$ of degree S of the communication matrix. Recall that Chebyshev polynomials are defined recursively. It turns out that the Chebyshev polynomial of degree ℓ for a communication matrix $\mathbf{P}$ is also given by a recursive formula as follows: $q_{\ell+1}(\mathbf{P}) = \frac{2w_\ell}{|\lambda_2|w_{\ell+1}}\mathbf{P}q_\ell(\mathbf{P}) - \frac{w_{\ell-1}}{w_{\ell+1}}q_{\ell-1}(\mathbf{P})$, where $w_0 = 0, w_1 = 1/|\lambda_2|, w_{\ell+1} = 2w_\ell/|\lambda_2| - w_{\ell-1}, q_0(\mathbf{P}) = I$ and $q_1(\mathbf{P}) = \mathbf{P}$. Specifically, in a Chebyshev-accelerated gossip protocol (Martínez-Rubio, Kanade, and Rebeschini 2019), the agents update their estimates of the average of the initial vector's $\boldsymbol{\nu}_0$ entries as follows:

$$\boldsymbol{\nu}_{\ell+1} = (2w_\ell)/(|\lambda_2|w_{\ell+1})\mathbf{P}\boldsymbol{\nu}_\ell - (w_{\ell-1}/w_{\ell+1})\boldsymbol{\nu}_{\ell-1}. \quad (2)$$

Our algorithm DLUCB, presented later in Sec. 2.2, implements the Checyshev-accelerated gossip protocol outlined above; see (Martínez-Rubio, Kanade, and Rebeschini 2019) for a similar implementation only for the classical K-Armed MAB. Specifically, we summarize the accelerated communication step described in (2) with a function $\mathrm{Comm}(x_{\mathrm{now}}, x_{\mathrm{prev}}, \ell)$ with three inputs: (1) x_{now}, the quantity of interest that the agent wants to update at the current round; (2) x_{prev}, the estimated value for the same quantity of interest that the agent updated in the previous round (cf. $\boldsymbol{\nu}_{\ell-1}$ in (2)); (3) ℓ, the current communication round. Note that inputs here are scalars, however, matrices and vectors can also be passed as inputs, in which case Comm runs entrywise. For a detailed description of Comm please refer to Algorithm 3 in App. B. The accelerated consensus algorithm implemented in Comm guarantees fast mixing of information thanks to the following key property (Martínez-Rubio, Kanade, and Rebeschini 2019, Lem. 3): for $\epsilon \in (0,1)$ and any vector $\boldsymbol{\nu_0}$ in the N-dimensional simplex, it holds that

$$\|Nq_S(\mathbf{P})\boldsymbol{\nu_0} - \mathbf{1}\|_2 \leq \epsilon, \text{ provided } S = \frac{\log(2N/\epsilon)}{\sqrt{2\log(1/|\lambda_2|)}}. \quad (3)$$

In view of this, our algorithm properly calls Comm (see Algorithm 1) such that for every $i \in [N]$ and $t \in [T]$, the action $\mathbf{x}_{i,t}$ and corresponding reward $y_{i,t}$ are communicated within the network for S rounds. At round $t+S$, agent i has access to $a_{i,j}\mathbf{x}_{j,t}$ and $a_{i,j}y_{j,t}$ where $a_{i,j} = N[q_S(\mathbf{P})]_{i,j}$. Thanks to (3), $a_{i,j}$ is ϵ close to 1, thus, these are good approximations of the true $\mathbf{x}_{j,t}$ and $y_{j,t}$. Accordingly, at the beginning of round $t > S$, each agent i computes

$$\mathbf{A}_{i,t} := \lambda I + \sum_{\tau=1}^{t-S}\sum_{j=1}^{N} a_{i,j}^2 \mathbf{x}_{j,\tau}\mathbf{x}_{j,\tau}^T, \ \mathbf{b}_{i,t} := \sum_{\tau=1}^{t-S}\sum_{j=1}^{N} a_{i,j}^2 y_{j,\tau}\mathbf{x}_{j,\tau}, \quad (4)$$

which are agent i's approximations of the sufficient statistics $\mathbf{A}_{*,t-S+1}$ and $\mathbf{b}_{*,t-S+1}$ defined in (1). On the other hand, for rounds $1 \leq t \leq S$ (before any mixing has been completed), let $\mathbf{A}_{i,t} = \lambda I + \sum_{\tau=1}^{t-1}\mathbf{x}_{i,\tau}\mathbf{x}_{i,\tau}^T$ and $\mathbf{b}_{i,t} = \sum_{\tau=1}^{t-1} y_{i,\tau}\mathbf{x}_{i,\tau}$ for $i \in [N]$. With these, at the beginning of each round $t \in [T]$, agent i constructs the confidence set

$$\mathcal{C}_{i,t} := \{\boldsymbol{\nu} \in \mathbb{R}^d : \|\boldsymbol{\nu} - \hat{\boldsymbol{\theta}}_{i,t}\|_{\mathbf{A}_{i,t}} \leq \beta_t\}, \quad (5)$$

where $\hat{\boldsymbol{\theta}}_{i,t} = \mathbf{A}_{i,t}^{-1}\mathbf{b}_{i,t}$ and β_t is chosen as in Thm. 1 below to guarantee $\boldsymbol{\theta}_* \in \mathcal{C}_{i,t}$ with high probability.

Theorem 1 (Confidence sets)**.** *Let Assumptions 1, 2 and 3 hold. Fix $\epsilon \in (0,1)$ and S as in* (3)*. For $\delta \in (0,1)$, let $\beta_t := (1+\epsilon)\sigma\sqrt{d\log\left(\frac{2\lambda dN + 2N^2 t}{\lambda d\delta}\right)} + \lambda^{1/2}$. Then with probability at least $1-\delta$, for all $i \in [N]$ and $t \in [T]$ it holds that $\boldsymbol{\theta}_* \in \mathcal{C}_{i,t}$.*

The proof is mostly adapted from (Abbasi-Yadkori, Pál, and Szepesvári 2011, Thm. 2) with necessary modifications to account for the imperfect information; see App. A.1.

2.2 Decentralized Linear UCB

We now describe DLUCB Algorithm 1 (see App. A.3 for a more detailed version). Each agent runs DLUCB in a parallel/synchronized way. For concreteness, let us focus on agent $i \in [N]$. At every round t, the agent maintains the following first-in first-out (FIFO) queues of size at most S: $\mathcal{A}_{i,t}$, $\mathcal{B}_{i,t}$, $\mathcal{A}_{i,t-1}$, and $\mathcal{B}_{i,t-1}$. The queue $\mathcal{A}_{i,t}$ contains agent i's estimates of all actions played at rounds $[t-S : t-1]$. Concretely, its j-th member, denoted by $\mathcal{A}_{i,t}(j) \in \mathbb{R}^{N\times d}$, is a matrix whose k-th row is agent i's estimate of agent k's action played at round $t+j-S-1$. Similarly, we define $\mathcal{B}_{i,t}$ as the queue containing agent i's estimates of rewards observed at rounds $[t-S : t-1]$. At each round t, agent i sends every member of $\mathcal{A}_{i,t}$ and $\mathcal{B}_{i,t}$ (each entry of them) to its neighbors and at the same time it receives the corresponding values from them. The received values are used to update the information stored in $\mathcal{A}_{i,t}$ and $\mathcal{B}_{i,t}$. The update is implemented by the sub-routine Comm outlined in Sec. 2.1 and presented in detail in App. B.

At the beginning of rounds $t > S$ when, the information of rounds $[t-S]$ is mixed enough, agent i updates its estimates $\mathbf{A}_{i,t}$ and $\mathbf{b}_{i,t}$ of $\mathbf{A}_{*,t-S}$ and $\mathbf{b}_{*,t-S}$, respectively. Using these, it creates the confidence set $\mathcal{C}_{i,t}$ and runs the UCB decision rule of Line 11 to select an action. Next, agent i updates $\mathcal{A}_{i,t}$ and $\mathcal{B}_{i,t}$ in Lines 12 and 13, by eliminating the first elements (dequeuing) $\mathcal{A}_{i,t}(1)$ and $\mathcal{B}_{i,t}(1)$ of the queues $\mathcal{A}_{i,t}$ and $\mathcal{B}_{i,t}$ and adding the following elements at their end (enqueuing). At $\mathcal{A}_{i,t}$ it appends $\mathbf{X}_{i,t} \in \mathbb{R}^{N\times d}$, whose rows are all zero but its i-th row which is set to $\mathbf{x}_{i,t}^T$. Concurrently, at $\mathcal{B}_{i,t}$, it appends $\mathbf{y}_{i,t} \in \mathbb{R}^N$, whose elements are all zero but its i-th element which is set to $y_{i,t}$. Note that $\mathbf{X}_{i,t}$ (similarly, $\mathbf{y}_{i,t}$) contains agent i's estimates of actions at round t, and the zero rows will be updated with agent i's estimates of other agents' information at round t in future rounds. This is achieved via calling the consensus algorithm Comm in Lines 14 and 15, with which agent i communicates all the members of $\mathcal{A}_{i,t}$ and $\mathcal{B}_{i,t}$ with its neighbors.

Algorithm 1: DLUCB for Agent i

Input: $\mathcal{D}$, N, d, $|\lambda_2|$, ϵ, λ, δ, T

$S = \log(2N/\epsilon)/\sqrt{2\log(1/|\lambda_2|)}$

$\mathbf{A}_{i,1} = \lambda I$, $\mathbf{b}_{i,1} = \mathbf{0}$, $\mathcal{A}_{i,0} = \mathcal{A}_{i,1} = \mathcal{B}_{i,0} = \mathcal{B}_{i,1} = \emptyset$

for $t = 1, \ldots, S$ **do**

  Play $\mathbf{x}_{i,t} = \arg\max_{\mathbf{x}\in\mathcal{D}} \max_{\boldsymbol{\nu}\in\mathcal{C}_{i,t}} \langle \boldsymbol{\nu}, \mathbf{x}\rangle$ and observe $y_{i,t}$.

  $\mathcal{A}_{i,t}$.append$(\mathbf{X}_{i,t})$ and $\mathcal{B}_{i,t}$.append$(\mathbf{y}_{i,t})$

  $\mathcal{A}_{i,t+1} = \text{Comm}(\mathcal{A}_{i,t}, \mathcal{A}_{i,t-1}, [t])$

  $\mathcal{B}_{i,t+1} = \text{Comm}(\mathcal{B}_{i,t}, \mathcal{B}_{i,t-1}, [t])$ // Comm runs for each member of $\mathcal{A}_{i,t}$ and $\mathcal{B}_{i,t}$

  $\mathbf{A}_{i,t+1} = \mathbf{A}_{i,t} + \mathbf{x}_{i,t}\mathbf{x}_{i,t}^T$, $\mathbf{b}_{i,t+1} = \mathbf{b}_{i,t} + y_{i,t}\mathbf{x}_{i,t}$

for $t = S+1, \ldots, T$ **do**

  $\mathbf{A}_{i,t} = \mathbf{A}_{i,t-1} + N^2\mathcal{A}_{i,t}(1)^T\mathcal{A}_{i,t}(1)$, $\mathbf{b}_{i,t} = \mathbf{b}_{i,t-1} + N^2\mathcal{A}_{i,t}(1)^T\mathcal{B}_{i,t}(1)$

  Play $\mathbf{x}_{i,t} = \arg\max_{\mathbf{x}\in\mathcal{D}} \max_{\boldsymbol{\nu}\in\mathcal{C}_{i,t}} \langle \boldsymbol{\nu}, \mathbf{x}\rangle$ and observe $y_{i,t}$

  $\mathcal{A}_{i,t}$.remove$\big(\mathcal{A}_{i,t}(1)\big)$.append $\big(\mathbf{X}_{i,t}\big)$

  $\mathcal{B}_{i,t}$.remove $\big(\mathcal{B}_{i,t}(1)\big)$ .append $\big(\mathbf{y}_{i,t}\big)$

  $\mathcal{A}_{i,t+1} = \text{Comm}(\mathcal{A}_{i,t}, \mathcal{A}_{i,t-1}(2:S), [S])$

  $\mathcal{B}_{i,t+1} = \text{Comm}(\mathcal{B}_{i,t}, \mathcal{B}_{i,t-1}(2:S), [S])$

Regret analysis. There are two key challenges in the analysis of DLUCB compared to that of single-agent LUCB. First, information sharing is imperfect: the consensus algorithm mixes information for a finite number S of communication rounds resulting in ϵ-approximations of the desired quantities (cf. (3)). Second, agents can use this (imperfect) information to improve their actions only after an inevitable delay. To see what changes in the analysis of regret, consider the standard decomposition of agent i's instantaneous regret at round t: $r_{i,t} = \langle \boldsymbol{\theta}_*, \mathbf{x}_*\rangle - \langle \boldsymbol{\theta}_*, \mathbf{x}_{i,t}\rangle \le 2\beta_t \big\|\mathbf{x}_{i,t}\big\|_{\mathbf{A}_{i,t}^{-1}}$. Using Cauchy-Schwartz inequality, an upper bound on the cumulative regret $\sum_{t=1}^T \sum_{i=1}^N r_{i,t}$ can be obtained by bounding the following key term:

$$\sum_{t=1}^{T}\sum_{i=1}^{N} \big\|\mathbf{x}_{i,t}\big\|^2_{\mathbf{A}_{i,t}^{-1}}. \tag{6}$$

We do this in two steps, each addressing one of the above challenges. First, in Lemma 1, we address the influence of imperfect information by relating the $\mathbf{A}_{i,t}^{-1}$–norms in (6), with those in terms of their perfect information counterparts $\mathbf{A}_{*,t-S+1}^{-1}$. Hereafter, let $\mathbf{A}_{*,t} = \lambda I$ for $t = -S, \ldots, 0, 1$.

Lemma 1 (Influence of imperfect information)**.** *Fix any $\epsilon \in (0, 1/(4d+1))$ and choose S as in* (3)*. Then, for all $i \in [N]$, $t \in [T]$ it holds that $\big\|\mathbf{x}_{i,t}\big\|^2_{\mathbf{A}_{i,t}^{-1}} \le e\big\|\mathbf{x}_{i,t}\big\|^2_{\mathbf{A}_{*,t-S+1}^{-1}}$.*

The intuition behind the lemma comes from the discussion on the accelerated protocol in Sec. 2.1. Specifically, with sufficiently small communication-error ϵ (cf. (3)), $\mathbf{A}_{i,t}$ (cf. (4)) is a good approximation of $\mathbf{A}_{*,t-S+1}$ (cf. (1)). The lemma replaces the task of bounding (6) with that of bounding $\sum_{t=1}^T \sum_{i=1}^N \big\|\mathbf{x}_{i,t}\big\|^2_{\mathbf{A}_{*,t-S+1}^{-1}}$. Unfortunately, this remains challenging. Intuitively, the reason for this is the mismatch of information about past actions in the gram matrix $\mathbf{A}_{*,t-S+1}$ at time t, compared to the inclusion of all terms $\mathbf{x}_{i,\tau}$ up to time t in (6). Our idea is to relate $\big\|\mathbf{x}_{i,t}\big\|_{\mathbf{A}_{*,t-S+1}^{-1}}$ to $\big\|\mathbf{x}_{i,t}\big\|_{\mathbf{B}_{i,t}^{-1}}$, where $\mathbf{B}_{i,t} = \mathbf{A}_{*,t} + \sum_{j=1}^{i-1} \mathbf{x}_{j,t}\mathbf{x}_{j,t}^T$. This is possible thanks to the following lemma.

Lemma 2 (Influence of delays)**.** *Let S as in* (3)*. Then, $\big\|\mathbf{x}_{i,t}\big\|^2_{\mathbf{A}_{*,t-S+1}^{-1}} \le e\big\|\mathbf{x}_{i,t}\big\|^2_{\mathbf{B}_{i,t}^{-1}}$, is true for all pairs $(i,t) \in [N] \times [T]$ except for at most $\psi(\lambda, |\lambda_2|, \epsilon, d, N, T) := Sd\log\left(1 + \frac{NT}{d\lambda}\right)$ of them.*

Using Lemmas 1 and 2 allows controlling the regret of all actions, but at most ψ of them, using standard machinery in the analysis of UCB-type algorithms. The proofs of Lemmas 1 and 2 and technical details relating the results to a desired regret bound are deferred to App. A.2. The theorem below is our first main result and bounds the regret of DLUCB.

Theorem 2 (Regret of DLUCB)**.** *Fix $\epsilon \in (0, 1/(4d+1))$ and $\delta \in (0,1)$. Let Assumptions 1, 2, 3 hold, and S be chosen as in* (3)*. Then, with probability at least $1-\delta$, it holds that:* $R_T \le 2Sd\log\left(1 + \frac{NT}{d\lambda}\right) + 2e\beta_T\sqrt{2dNT\log\left(\lambda + \frac{NT}{d}\right)}$.

The regret bound has two additive terms: a small term $2\psi(\lambda, |\lambda_2|, \epsilon, d, N, T)$ (cf. Lemma 2), which we call *regret of delay*, and, a second main term that (notably) is of the same order as the regret of a centralized problem where communication is possible between any two nodes (see Table 1). Thm. 2 holds for small $\epsilon \le 1/(4d+1)$. In App. A.2, we also provide a general regret bound for arbitrary $\epsilon \in (0,1)$.

2.3 DLUCB with Rare Communication

As discussed in more detail in Sec. 2.4, DLUCB achieves order-wise optimal regret, but its communication cost scales as $\mathcal{O}(dN^2T)$, i.e., linearly with the horizon duration T (see Table 1). In this section, we present a modification tailored to communication settings that are sensitive to communication cost. The new algorithm – termed RC-DLUCB – is also a fully decentralized algorithm that trade-offs a slight increase in the regret performance, while guaranteeing a significantly reduced communication cost of $\mathcal{O}\big(d^3N^{2.5}\frac{\log(Nd)}{\log^{1/2}(1/|\lambda_2|)}\big)$ over the *entire* horizon $[T]$. Due to space limitations, we defer a detailed description (see Algorithm 4) and analysis (see Thms. 4 and 5) of RC-DLUCB in App. C. At a high-level, we design RC-DLUCB inspired by the Rarely Switching OFUL algorithm by (Abbasi-Yadkori, Pál, and Szepesvári 2011). In contrast to the Rarely Switching OFUL algorithm that is designed to save on computations in single-agent systems, RC-DLUCB incorporates a similar idea in our previous DLUCB to save on communication rounds. Specifically, compared to DLUCB where communication happens at each round, in RC-DLUCB agents continue selecting actions individually (i.e., with no communication), unless a certain condition is triggered by any one of them. Then, they all switch to a communication phase, in which they communicate the unmixed information they have gathered for a duration of S rounds. Roughly speaking, an agent triggers the

Algorithm	Regret	Communication
DLUCB	$\mathcal{O}(d\frac{\log(Nd)}{\log^{0.5}(1/\lvert\lambda_2\rvert)}\log(NT) + d\log(NT)\sqrt{NT})$	$\mathcal{O}(dN^2T\frac{\log(Nd)}{\log^{0.5}(1/\lvert\lambda_2\rvert)})$
RC-DLUCB	$\mathcal{O}(Nd^{1.5}\frac{\log(Nd)}{\log^{0.5}(1/\lvert\lambda_2\rvert)}\log^{1.5}(NT) + d\log^2(NT)\sqrt{NT})$	$\mathcal{O}(d^3N^{2.5}\frac{\log(Nd)}{\log^{0.5}(1/\lvert\lambda_2\rvert)})$
No Communication	$\mathcal{O}(dN\log(T)\sqrt{T})$	0
Centralized	$\mathcal{O}(d\log(NT)\sqrt{NT})$	$\mathcal{O}(dN^2T)$
DCB	$\mathcal{O}((dN\log(NT))^3 + \log(NT)\sqrt{NT})$	$\mathcal{O}(d^2NT\log(NT))$
DisLinUCB	$\mathcal{O}(\log^2(NT)\sqrt{NT})$	$\mathcal{O}(d^3N^{1.5})$

Table 1: Comparison of DLUCB and RC-DLUCB to baseline, as well as, to state-of-the-art. See Sec. 2.4 for details.

communication phase only once it has gathered enough new information compared to the last update by the rest of the network. This can be measured by keeping track of the variations in the corresponding gram matrix.

2.4 Regret-Communication Trade-offs and Comparison to State of the Art

In Table 1, we compare (in terms of regret and communication) DLUCB and RC-DLUCB to two baselines: (i) a 'No Communication' and (ii) a fully 'Centralized' algorithm, as well as, to the state of the art: (iii) DCB (Korda, Szörényi, and Shuai 2016) and (iv) DisLinUCB (Wang et al. 2019).

Baselines. In the absence of communication, each agent independently implements a single-agent LUCB (Abbasi-Yadkori, Pál, and Szepesvári 2011). This trivial 'No Communication' algorithm has zero communication cost and applies to any graph, but its regret scales linearly with the number of agents. At another extreme, a fully 'Centralized' algorithm assumes communication is possible between any two agents at every round. This achieves optimal regret $\tilde{O}(\sqrt{NT})$, which is a lower bound to the regret of any decentralized algorithm. However, it is only applicable in very limited network topologies, such as a star graph where the central node acts as a master node, or, a complete graph. Notably, DLUCB achieves order-wise optimal regret that is same as that of the 'Centralized' algorithm modulo a small additive regret-of-delay term.

DisLinUCB. In a motivating recent paper (Wang et al. 2019), the authors presented 'DisLinUCB' a communication algorithm that applies to multi-agent settings, in which agents can communicate with a master-node/server, by sending or receiving information to/from it with zero latency. Notably, DisLinUCB is shown to achieve order-optimal regret performance same as the 'Centralized' algorithm, but at a significantly lower communication cost that does *not* scale with T (see Table 1). In this paper, we do *not* assume presence of a master-node. In our setting, this can only be assumed in very limited cases: a star or a complete graph. Thus, compared to DisLinUCB, our DLUCB can be used for arbitrary network topologies with similar regret guarantees. However, DLUCB requires that communication be performed at each round. This discrepancy motivated us to introduce RC-LUCB, which has communication cost (slightly larger, but) comparable to that of DisLinUCB (see Table 1), while being applicable to general graphs. As a final note, as in RC-DLUCB, the reduced communication cost in DisLinUCB relies on the idea of the Rarely Switching OFUL algorithm of (Abbasi-Yadkori, Pál, and Szepesvári 2011).

DCB. In another closely related work (Korda, Szörényi, and Shuai 2016) presented DCB for decentralized linear bandits in *peer-to-peer networks*. Specifically, it is assumed in (Korda, Szörényi, and Shuai 2016) that at every round each agent communicates with only *one* other *randomly* chosen agent per round. Instead, we consider fixed network topologies where each agent can only communicate with its immediate neighbors at every round. Thus, the two algorithms are not directly comparable. Nevertheless, we remark that, similar to our setting, DCB also faces the challenge of controlling a delayed use of information, caused by requiring enough mixing of the communicated information among agents. A key difference is that the duration of delay is typically $\mathcal{O}(\log t)$ in DCB, while in DLUCB it is fixed to S, i.e., independent of the round t. This explains the significantly smaller first-term in the regret of DLUCB as compared to the first-term in the regret of DCB in Table 1.

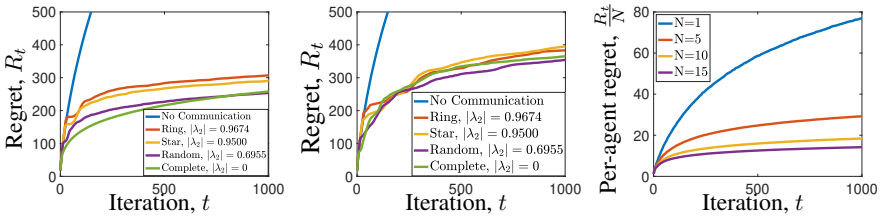
3 Safe Decentralized Linear Bandits

For the safe decentralized LB problem, we propose Safe-DLUCB, an extension of DLUCB to the safe setting and an extension of single-agent safe algorithms (Amani, Alizadeh, and Thrampoulidis 2019; Moradipari et al. 2019; Pacchiano et al. 2020) to multi-agent systems. Due to space limitations, we defer a detailed description of Safe-DLCUB to Algorithm 5 in App. D. Here, we give a high-level description of its main steps and present its regret guarantees. First, we need the following assumption and notation.

Assumption 4 (Non-empty safe set)**.** *A safe action $\mathbf{x}_0 \in \mathcal{D}$ and $c_0 := \langle \boldsymbol{\mu}_*, \mathbf{x}_0 \rangle < c$ are known to all agents. Also, $\langle \boldsymbol{\theta}_*, \mathbf{x}_0 \rangle \geq 0$.*

Define the normalized safe action $\tilde{\mathbf{x}}_0 := \frac{\mathbf{x}_0}{\|\mathbf{x}_0\|}$. For any $\mathbf{x} \in \mathbb{R}^d$, denote by $\mathbf{x}^o := \langle \mathbf{x}, \tilde{\mathbf{x}}_0 \rangle \tilde{\mathbf{x}}_0$ its projection on $\mathbf{x}_0$, and, by $\mathbf{x}^\perp := \mathbf{x} - \mathbf{x}^o$ its projection onto the orthogonal subspace.

In the presence of safety, the agents must act conservatively to ensure that the chosen actions $\mathbf{x}_{i,t}$ do not violate the safety constraint $\langle \boldsymbol{\mu}_*, \mathbf{x}_{i,t} \rangle \leq c$. To this end, agent i communicates, not only $\mathbf{x}_{i,t}$ and $y_{i,t}$, but also the bandit-feedback measurements $z_{i,t}$, following the communication protocol implemented by Comm (cf. Sec. 2.1). Once information is sufficiently mixed, it builds an additional confi-



(a) DLUCB (b) RC-DLUCB (c) DLUCB

Figure 1: Regret comparison.

dence set $\mathcal{E}_{i,t}$ that includes $\boldsymbol{\mu}_*^\perp$ with high probability (note that $\boldsymbol{\mu}_*^o$ is already known by Assumption 4). Please refer to App. D.1 for the details on constructing $\mathcal{E}_{i,t}$. Once $\mathcal{E}_{i,t}$ is constructed, agent i creates the following *safe* inner approximation of the true $\mathcal{D}^{\text{s}}(\boldsymbol{\mu}_*)$: $\mathcal{D}_{i,t}^{\text{s}} := \{\mathbf{x} \in \mathcal{D} : \frac{\langle \mathbf{x}^o, \tilde{\mathbf{x}}_0 \rangle}{\|\mathbf{x}_0\|} c_0 + \langle \hat{\boldsymbol{\mu}}_{i,t}^\perp, \mathbf{x}^\perp \rangle + \beta_t \|\mathbf{x}^\perp\|_{\mathbf{A}_{i,t}^{\perp,-1}} \leq c\}$. Specifically, Proposition 1 in App. D.1 guarantees for any $\delta \in (0,1)$ that for all $i \in [N]$, $t \in [T]$, all actions in $\mathcal{D}_{i,t}^{\text{s}}$ are safe with probability $1-\delta$. After constructing $\mathcal{D}_{i,t}^{\text{s}}$, agent i selects *safe* action $\mathbf{x}_{i,t} \in \mathcal{D}_{i,t}^{\text{s}}$ following a UCB decision rule:

$$\langle \tilde{\boldsymbol{\theta}}_{i,t}, \mathbf{x}_{i,t} \rangle = \max_{\mathbf{x} \in \mathcal{D}_{i,t}^{\text{s}}} \max_{\boldsymbol{\nu} \in \kappa_r \mathcal{C}_{i,t}} \langle \boldsymbol{\nu}, \mathbf{x} \rangle. \tag{7}$$

A subtle, but critical, point in (7) is that the inner maximization is over an appropriately *enlarged confidence set* $\kappa_r \mathcal{C}_{i,t}$. Specifically, compared to Lines 4 and 11 in Algorithm 1, we need here that $\kappa_r > 1$. Intuitively, this is required because the outer maximization in (7) is not over the entire $\mathcal{D}^{\text{s}}(\boldsymbol{\mu}_*)$, but only a subset of it. Thus, larger values of κ_r are needed to provide enough exploration to the algorithm so that the selected actions in $\mathcal{D}_{i,t}^{\text{s}}$ are -often enough- *optimistic*, i.e., $\langle \tilde{\boldsymbol{\theta}}_{i,t}, \mathbf{x}_{i,t} \rangle \geq \langle \boldsymbol{\theta}_*, \mathbf{x}_* \rangle$; see Lemma 5 in App. D.2 for the exact statement. We attribute the above idea that more aggressive exploration of that form is needed in the safe setting to (Moradipari et al. 2019), only they considered a Thompson-sampling scheme and a single agent. (Pacchiano et al. 2020) extended this idea to UCB algorithms, again in the single-agent setting (and for a slightly relaxed notion of safety). Here, we show that the idea extends to multi-agent systems and when incorporated to the framework of DLUCB leads to a safe decentralized algorithm with provable regret guarantees stated in the theorem below. See App. D for the proof.

Theorem 3 (Regret of Safe-DLUCB)**.** *Fix* $\delta \in (0, 0.5)$, $\kappa_r = \frac{2}{c - c_0} + 1$, $\epsilon \in (0, 1/(4d+1))$. *Let Assumptions 1, 2, 3, 4 hold, and* S *be chosen as in* (3)*. Then, with probability at least* $1 - 2\delta$*, it holds that:* $R_T \leq 2Sd \log(1 + \frac{NT}{d\lambda}) + 2e\kappa_r \beta_T \sqrt{2dNT \log(\lambda + \frac{NT}{d})}$.

The regret bound is of the same order as DLUCB regret bound, with only an additional factor κ_r in its second term.

4 Experiments

In this section, we evaluate our algorithms' performance on synthetic data. Since the UCB decision rule at line 11 of Algorithm 1 involves a generally non-convex optimization problem, we use a standard computationally tractable modification that replaces ℓ_2 with ℓ_1 norms in the definition of confidence set (5) (unless the decision set is finite); see (Dani, Hayes, and Kakade 2008). All results directly apply to this modified algorithm after only changing the radius β_t with $\beta_t \sqrt{d}$ (Dani, Hayes, and Kakade 2008, Sec. 3.4). All the results shown depict averages over 20 realizations, for which we have chosen $d = 5, \mathcal{D} = [-1, 1]^5$, $\lambda = 1$, and $\sigma = 0.1$. Moreover, $\boldsymbol{\theta}_*$ is drawn from $\mathcal{N}(0, I_5)$ and then normalized to unit norm. We compute the communication matrix as $\mathbf{P} = I - \frac{1}{\delta_{\max}+1} \mathbf{D}^{-1/2} \mathcal{L} \mathbf{D}^{-1/2}$, where $\delta_{\max}$ is the maximum degree of the graph and $\mathcal{L}$ is the graph Laplacian (see (Duchi, Agarwal, and Wainwright 2011) for details). In Figs. 1a and 1b, fixing $N = 20$, we evaluate the performance of DLUCB and RC-DLUCB on 4 different topologies: Ring, Star, Complete, and a Random Erdős–Rényi graph with parameter $p = 0.5$; see Fig. 2 in App. E for graphical illustrations of the graphs. We also compare them to the performance of No Communication (see Sec. 2.4). The plot verifies the sublinear growth for all graphs, the superiority over the setting of No Communication and the fact that smaller $|\lambda_2|$ leads to a smaller regret (regret of delay term in Thm. 2). A comparison between Figs. 1a and 1b, further confirms the slightly better regret performance of DLUCB compared to RC-DLUCB (but the latter has superior communication cost). Fig. 1c emphasizes the value of collaboration in speeding up the learning process. It depicts the per-agent regret of DLUCB on random graphs with $N = 5$, 10 and 15 nodes and compares their performance with the single-agent LUCB. Clearly, as the number of agents increases, each agent learns the environment faster as an individual.

5 Conclusion

In this paper, we proposed two fully decentralized LB algorithms: 1) DLUCB and 2) RC-DLUCB with small communication cost. We also proposed Safe-DLUCB to address the problem of safe LB in multi-agent settings. We derived near-optimal regret bounds for all the aforementioned algorithms that are applicable to arbitrary, but fixed networks. An interesting open problem is to design decentralized algorithms with provable guarantees for settings with time-varying networks. Also, extensions to nonlinear settings and other types of safety-constraints are important future directions.

Acknowledgments

This work is supported by the National Science Foundation under Grant Number (1934641). This work was completed while both authors were affiliated with UC, Santa Barbara.

Ethics Statement

Sequential decision making problems arise at every occasion that learners repeatedly interact with an unknown environment in an effort to maximize a certain notion of reward gained from interactions with this environment. Bandits often provide a simple form of this interaction and bandit optimization algorithms have been successfully applied over the years in online advertising, recommendation services, resource allocation, etc. (Lattimore and Szepesvári 2018). More recently, researchers have started exploring the potentials of bandit algorithms in physical systems, such as in robotics, wireless networks, the power grid and in medical trials. A distinguishing feature of many of these "new" applications is their *safety-critical* nature. Specifically, the algorithm's chosen actions need to satisfy certain system constraints. Importantly, the constraints are often unknown, which leads to the challenge of balancing the goal of reward maximization with the restriction of playing "safe actions". At the same time, many modern applications of bandit algorithms involve a networked set of distributed agents (e.g., wireless/sensor networks). This calls for extensions of the traditional bandit setting to networked systems. The past few years have seen a surge of research activity in these two areas: (i) safe, and (ii) distributed bandit optimization (Sui et al. 2015; Amani, Alizadeh, and Thrampoulidis 2019; Sui et al. 2018; Kazerouni et al. 2017; Moradipari et al. 2019; Korda, Szörényi, and Shuai 2016; Martínez-Rubio, Kanade, and Rebeschini 2019; Wang et al. 2019; Szörényi et al. 2013). This paper presents the first (simultaneously) safe and distributed bandit algorithm and contributes at the intersection of these two emerging lines of works. We study simplified linear models, but we believe that they already capture some relevant key problem features and challenges. Finally, while our study is theoretical we believe that it motivates further research in this field that can potentially guide practical implementations.

References

Abbasi-Yadkori, Y.; Pál, D.; and Szepesvári, C. 2011. Improved algorithms for linear stochastic bandits. In *Advances in Neural Information Processing Systems*, 2312–2320.

Amani, S.; Alizadeh, M.; and Thrampoulidis, C. 2019. Linear stochastic bandits under safety constraints. In *Advances in Neural Information Processing Systems*, 9252–9262.

Avner, O.; and Mannor, S. 2019. Multi-user communication networks: A coordinated multi-armed bandit approach. *IEEE/ACM Transactions on Networking* 27(6): 2192–2207.

Berkenkamp, F.; Krause, A.; and Schoellig, A. P. 2016. Bayesian optimization with safety constraints: safe and automatic parameter tuning in robotics. *arXiv preprint arXiv:1602.04450* .

Dani, V.; Hayes, T. P.; and Kakade, S. M. 2008. Stochastic linear optimization under bandit feedback .

Duchi, J. C.; Agarwal, A.; and Wainwright, M. J. 2011. Dual averaging for distributed optimization: Convergence analysis and network scaling. *IEEE Transactions on Automatic control* 57(3): 592–606.

Kazerouni, A.; Ghavamzadeh, M.; Yadkori, Y. A.; and Van Roy, B. 2017. Conservative contextual linear bandits. In *Advances in Neural Information Processing Systems*, 3910–3919.

Khezeli, K.; and Bitar, E. 2019. Safe Linear Stochastic Bandits. *arXiv preprint arXiv:1911.09501* .

Korda, N.; Szörényi, B.; and Shuai, L. 2016. Distributed clustering of linear bandits in peer to peer networks. In *Journal of machine learning research workshop and conference proceedings*, volume 48, 1301–1309. International Machine Learning Societ.

Landgren, P.; Srivastava, V.; and Leonard, N. E. 2016a. Distributed cooperative decision-making in multiarmed bandits: Frequentist and Bayesian algorithms. In *2016 IEEE 55th Conference on Decision and Control (CDC)*, 167–172. IEEE.

Landgren, P.; Srivastava, V.; and Leonard, N. E. 2016b. On distributed cooperative decision-making in multiarmed bandits. In *2016 European Control Conference (ECC)*, 243–248. IEEE.

Lattimore, T.; and Szepesvári, C. 2018. Bandit algorithms. *preprint* 28.

Li, S.; Hao, F.; Li, M.; and Kim, H.-C. 2013. Medicine rating prediction and recommendation in mobile social networks. In *International conference on grid and pervasive computing*, 216–223. Springer.

Lynch, N. A. 1996. *Distributed algorithms*. Elsevier.

Martínez-Rubio, D.; Kanade, V.; and Rebeschini, P. 2019. Decentralized Cooperative Stochastic Bandits. In *Advances in Neural Information Processing Systems*, 4531–4542.

Moradipari, A.; Amani, S.; Alizadeh, M.; and Thrampoulidis, C. 2019. Safe Linear Thompson Sampling. *arXiv preprint arXiv:1911.02156* .

Pacchiano, A.; Ghavamzadeh, M.; Bartlett, P.; and Jiang, H. 2020. Stochastic Bandits with Linear Constraints. *arXiv preprint arXiv:2006.10185* .

Rusmevichientong, P.; and Tsitsiklis, J. N. 2010. Linearly parameterized bandits. *Mathematics of Operations Research* 35(2): 395–411.

Seaman, K.; Bach, F.; Bubeck, S.; Lee, Y. T.; and Massoulié, L. 2017. Optimal algorithms for smooth and strongly convex distributed optimization in networks. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, 3027–3036. JMLR. org.

Sui, Y.; Gotovos, A.; Burdick, J. W.; and Krause, A. 2015. Safe exploration for optimization with Gaussian processes. *Proceedings of Machine Learning Research* 37: 997–1005.

Sui, Y.; Zhuang, V.; Burdick, J. W.; and Yue, Y. 2018. Stagewise safe bayesian optimization with gaussian processes. *arXiv preprint arXiv:1806.07555* .

Szörényi, B.; Busa-Fekete, R.; Hegedűs, I.; Ormándi, R.; Jelasity, M.; and Kégl, B. 2013. Gossip-based distributed stochastic bandit algorithms. In *Journal of Machine Learning Research Workshop and Conference Proceedings*, volume 2, 1056–1064. International Machine Learning Societ.

Wang, Y.; Hu, J.; Chen, X.; and Wang, L. 2019. Distributed Bandit Learning: How Much Communication is Needed to Achieve (Near) Optimal Regret. *arXiv preprint arXiv:1904.06309* .

Xiao, L.; and Boyd, S. 2004. Fast linear iterations for distributed averaging. *Systems & Control Letters* 53(1): 65–78.