Equilibrium Bandits: Learning Optimal Equilibria of Unknown Dynamics

Siddharth Chandak
Stanford University, Department of Electrical Engineering
USA
chandaks@stanford.edu

Ilai Bistritz
Stanford University, Department of Electrical Engineering
USA
bistritz@stanford.edu

Nicholas Bambos
Stanford University, Department of Electrical Engineering
USA
bambos@stanford.edu

ABSTRACT

Consider a decision-maker that can pick one out of K actions to control an unknown system, for T turns. The actions are interpreted as different configurations or policies. Holding the same action fixed, the system asymptotically converges to a unique equilibrium, as a function of this action. The dynamics of the system are unknown to the decision-maker, which can only observe a noisy reward at the end of every turn. The decision-maker wants to maximize its accumulated reward over the T turns. Learning what equilibria are better results in higher rewards, but waiting for the system to converge to equilibrium costs valuable time. Existing bandit algorithms, either stochastic or adversarial, achieve linear (trivial) regret for this problem. We present a novel algorithm, termed Upper Equilibrium Concentration Bound (UECB), that knows to switch an action quickly if it is not worth it to wait until the equilibrium is reached. This is enabled by employing 'convergence bounds' to determine how far the system is from equilibrium. We prove that UECB achieves a regret of $O(\log(T) + \tau_c \log(\tau_c) + \tau_c \log\log(T))$ for this "equilibrium bandit problem" where τ_c is the worst case approximate convergence time to equilibrium. We then show that both epidemic control and game control are special cases of equilibrium bandits, where $\tau_c \log \tau_c$ typically dominates the regret. We then test UECB numerically for both of these applications.

KEYWORDS

online learning; multiagent systems; game theory

ACM Reference Format:
Siddharth Chandak, Ilai Bistritz, and Nicholas Bambos. 2023. Equilibrium Bandits: Learning Optimal Equilibria of Unknown Dynamics. In *Proc. of the 22nd International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2023), London, United Kingdom, May 29 – June 2, 2023*, IFAAMAS, 9 pages.

1 INTRODUCTION

Many large-scale complex systems reach an equilibrium over time. Examples include epidemics, transportation, markets, and supply chains. With no planning, the global performance at this equilibrium can be poor. When a decision-maker can control some parameters of such a system, it can influence the equilibrium that the system converges to. Examples are changing the frequencies of subway lines, or the masking and isolation policies during an epidemic. However, a model for the dynamics of these large-scale complex systems is rarely available. Instead, the decision-maker can only observe the impact of its decisions in real time. It is infeasible to allow the system to converge to equilibrium under each policy since this will waste significant time on suboptimal policies. This introduces a learning problem of controlling such systems with 'bandit feedback' [18].

Motivated by this, we consider an agent that takes an action from a discrete set of actions at each timestep. There is an underlying system that evolves with time depending on the agent's action. At each timestep, the agent receives a noisy reward as a function of its action **and** the 'state' of the system. The key aspect of the underlying system is that if we fix the action, it would asymptotically converge to a unique equilibrium as a function of this action. However, waiting too long for the system to converge to a bad equilibrium is costly. We measure the regret of the agent as the difference between the reward at the optimal equilibrium (i.e., for the optimal action) and the accumulated reward of the agent. This introduces a new bandit problem which we name *'Equilibrium Bandits'*.

The reward process in equilibrium bandits is not i.i.d. over time. The expected reward approaches the expected reward at equilibrium and therefore has memory. Hence, stochastic bandit algorithms result in linear regret in T. On the other hand, adversarial bandit algorithms also result in linear regret.

In applications such as epidemics and transportation, convergence to equilibrium can take significant time. Therefore, we are interested in the dependence of the regret on the worst-case 'approximate convergence time' τ_c in addition to the horizon T. Alternatively, we can think of τ_c as T-dependent.

When estimating the reward at equilibrium, the distance of the current state from equilibrium creates 'equilibrium noise'. Hence, we need more consecutive 'arm pulls' to weaken the equilibrium noise at the last pull. This conflicts with the averaging required to weaken the i.i.d. reward noise, since averaging over longer periods would have to use states that are too far from equilibrium.

We present the Upper Equilibrium Concentration Bound (UECB) algorithm for equilibrium bandits that builds upon the basic intuition behind the Upper Confidence Bound (UCB) algorithm [18]. Our main innovation is employing 'convergence bounds' to determine the maximum possible reward the agent could get by waiting for the system to converge to equilibrium for a given action. A chosen action in UECB is played consecutively for a full "epoch". The epoch length increases with the number of epochs this action has been chosen in the past. Consequently, UECB spends time weakening the equilibrium noise only for promising actions. UECB balances between the i.i.d. noise and equilibrium noise by only using a fraction of samples from a given epoch for the reward estimation.

Proc. of the 22nd International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2023), A. Ricci, W. Yeoh, N. Agmon, B. An (eds.), May 29 – June 2, 2023, London, United Kingdom.

We show that UECB achieves $O(\log(T) + \tau_c \log(\tau_c) + \tau_c \log\log(T))$ expected regret. We also prove a lower bound that shows that UECB is optimal up to logarithmic factors in τ_c.

We detail two real-life problems that can be modeled as equilibrium bandits. The first example is epidemic control, where the agent is the government that chooses a policy that may include lockdowns, link closures, and enforcing masks [7]. These policies incur both an operational cost and a health cost by affecting the infection rate. We use the SIS model [2] to formalize the epidemic spread. Our second example is a continuous game where the system consists of multiple players trying to optimize their utility functions using gradient-based learning [20]. Specifically, we consider resource allocation games where the policymaker chooses the set of resources available to each player and wishes to maximize the sum of utilities of all players at equilibrium [16]. We then simulate these examples and show that UECB performs well based on minimal knowledge of the system that is readily available in practice.

1.1 Related Work

There has been plenty of research on multi-armed bandits in the last century, including stochastic bandits, contextual bandits, linear bandits, and adversarial bandits [18]. Special cases for each of these have been studied in great detail [4, 17, 24]. There are also works on bandits which deal with a system evolving with time. These often deal with a Markov-chain based stochastic evolution, e.g., restless bandits [34, 35].

Unlike restless bandits, in equilibrium bandits, an unperturbed system, where the chosen action is fixed, would asymptotically converge to equilibrium as a function of that action. Such behavior is typical to multiagent systems, specifically with many humans in the loop (e.g., epidemics and transportation). Furthermore, stochastic and adversarial bandit algorithms both give linear regret in equilibrium bandits. We propose the first algorithm, named UECB, that achieves sublinear regret for equilibrium bandits.

Equilibrium bandits can be thought of as a special case of non-stationary stochastic bandits [5], but applying this approach would result in a regret bound of $O(T^{\frac{2}{3}})$ at best (depending on the convergence rate to equilibrium) since it does not leverage the converging structure of equilibrium bandits.

Another field closely related to our work is that of reinforcement learning (RL) [31]. Classically, the RL problem is modeled using Markov decision processes and the aim of the agent is to choose the action at each step that maximizes its cumulative reward. There has been significant research in developing algorithms for RL [25, 32, 33]. In recent years, there has been progress in developing deep learning-based methods for complex problems such as multi-agent control, robotics, and games[10, 15, 29, 30]. Equilibrium bandits differ from the RL literature in one major aspect: the state in our case evolves in a non-stochastic, converging way. One could model our evolution as a deterministic Markov chain. However, RL algorithms are designed for general problems and typically assume an ergodic Markov chain [33] or the existence of an offline simulator that allows 'restarts' of the Markov chain [30]. These assumptions do not hold in our case, since our deterministic Markov chain is absorbing and our target is real-time learning which cannot be restarted. Furthermore, in equilibrium bandits, only the reward is observable whereas typical RL assumes that the state is observable as well.

Our work is related to the literature on control and intervention in games, where a manager can tune some parameters in the reward functions of the players [3, 6, 13, 22, 23, 27]. While our dynamics do not have to stem from a game, games are a key example of a system that converges to equilibrium. From this point of view, our work is the first to provide regret guarantees while learning to control an unknown game.

2 PROBLEM FORMULATION

Consider an agent that chooses an action a_t at each time t from the action set $\mathcal{A} = \{1, \dots, K\}$. The action controls an underlying system that evolves with time and affects the agent's reward. Let z_t be the state of the system at time t. We assume that z_t lies in a bounded and closed set $\mathcal{Z} \subset \mathbb{R}^d$. Then we define the 'evolution' function $g : \mathcal{A} \times \mathcal{Z} \to \mathcal{Z}$ and the 'reward' function $f : \mathcal{A} \times \mathcal{Z} \to \mathbb{R}$. We assume that $f(a, z)$ is Lipschitz continuous with parameter $L > 0$, as a function of z, for all actions a. Without loss of generality, we make the following two assumptions to simplify the notation: $f(a, z)$ is bounded in $[0, 1]$ for all a, z and $\mathcal{Z} \subseteq \mathcal{B}_{0.5}(\|\cdot\|)$ where $\mathcal{B}_{0.5}(\|\cdot\|) = \{z \in \mathbb{R}^d \mid \|z\| \leq 0.5\}$. Here $\|\cdot\|$ denotes any compatible norm on $\mathbb{R}^d$. The function g determines the next state of the system based on the current state and the action taken, i.e., $z_{t+1} = g(a_t; z_t)$. The function f determines the agent's expected reward, i.e., $x_t = f(a_t; z_t)$. The noisy reward y_t observed by the agent is given by $y_t = x_t + \eta_t$ where η_t is i.i.d. subgaussian noise with parameter σ, i.e., $\mathbb{E}[\eta_t] = 0$ and $\mathbb{E}[\exp(\alpha\eta_t)] \leq \exp(\alpha^2\sigma^2/2)$ for all $\alpha \in \mathbb{R}$. Rewards are typically noisy since the effectiveness of a policy cannot be deduced accurately and is often based on stochastic data.

Motivated by applications such as epidemic control and game control, we make the following assumption on the evolution function:

Assumption 1. *The function $g(\cdot;\cdot)$ satisfies the following conditions:*

(a) For each action a, consider the iteration $z_{t+1} = g(a; z_t)$ for $t > 0$. There exists a unique equilibrium corresponding to action a, i.e., a z_a^* such that $z_a^* = g(a; z_a^*)$. Furthermore, this equilibrium is a stable point, i.e., $\lim_{t\uparrow\infty} g^{(t)}(a; z) = z_a^*$ for all $z \in \mathcal{Z}$.

(b) When action a is played, the distance of the state z_t from z_a^* decreases. Formally,

$$\|g(a; z) - z_a^*\| \leq c(a; z)\|z - z_a^*\|, \ \forall\ a \in \mathcal{A}, z \in \mathcal{Z}, \tag{1}$$

where $0 < c(a; z) < 1$. We assume that $c(a; z)$ is bounded away from 1, i.e., $\exists\ \tau_c \geq 1$ s.t. $c(a; z) < e^{-\frac{1}{\tau_c}}$ for all a and z.

Part (a) of the above assumption implies that if the agent keeps the action a fixed, the system will asymptotically converge to the equilibrium state z_a^*. In addition, we define $x_a^* := f(a, z_a^*)$ as the equilibrium reward for action a.

The τ_c in part (b) is the "approximate convergence time to equilibrium", i.e., the timesteps required until the distance of the state from equilibrium is at most a $1/e$ factor of its initial value.

A basic class of functions g satisfying the above assumption are contraction mappings. A contraction mapping has a unique fixed point which is the unique equilibrium point required for part (a). Similarly, the contraction factor is the constant $c(a; z)$. Contraction mappings are commonly found in solutions of many ODEs such as

Newton's method [26] and for popular policy evaluation schemes such as temporal difference learning [9]. In section 5, we detail two applications that result in non-contractive mappings but still satisfy the above assumption.

We now return to the problem and the objective of the agent. The agent takes an action a_t at each time t. The agent does not observe the underlying state z_t and only observes a noisy version of the reward $f(a_t; z_t)$ at each time t. The optimal action a^* is defined as $a^* := \arg\max_a f(a; z_a^*)$, i.e., the action with the highest equilibrium reward. For simplicity, we assume that this optimal action is unique, i.e., $f(a, z_a^*) < f(a^*, z_{a^*}^*)$ for all $a \neq a^*$. Nevertheless, our analysis follows with minor modifications for the case of multiple optimal actions, using the same algorithm. Define the suboptimality gap for each action as $\Delta_a = x_{a^*}^* - x_a^*$. The regret till time T is $R(T) := \sum_{t=1}^{T} \left(f(a^*, z_{a^*}^*) - y_t \right)$. We want to design an algorithm for the agent that minimizes the expected cumulative regret:

$$\mathbb{E}[R(T)] = \max_a \sum_{t=1}^{T} f(a; z_a^*) - \mathbb{E}\left[\sum_{t=1}^{T} f(a_t; z_t) \right]$$

where the expectation is with respect to the stochastic noise. With multiple equilibria, we cannot guarantee to which one the system would converge. We can then redefine the above regret such that $z_{a^*}^*$ is the worst equilibrium for action a, with no modifications needed in our analysis or algorithm.

With this regret, a good algorithm would find the optimal action as quickly as possible and then commit to it to allow the system to converge to the corresponding equilibrium. This objective is inspired by applications such as epidemics and transportation (studied in detail in section 5). In an epidemic, the government is the agent that has to choose the best policy to control the spread. Then, the different policies (e.g., lockdown, masks) are the actions and the underlying state is the fraction of infected individuals. The cost then takes into account the health costs (e.g., deaths and complications) and the operational cost (e.g., treatment and economic implications). This example makes it clear why we wish to maximize the expected cumulative reward and why we wish to commit to the optimal action as quickly as possible.

Our contribution is the novel UECB algorithm, which is presented in the next section. Our main result proves a regret bound for UECB, which we show is optimal up to logarithmic factors.

Theorem 1. *For an equilibrium bandit instance satisfying Assumption 1, with a Lipschitz continuous reward function bounded in* $[0, 1]$ *and noise* η_t *that is subgaussian with parameter* σ*, the expected cumulative regret of the UECB algorithm (Algorithm 1) is bounded as:*

$$\begin{aligned} \mathbb{E}[R(T)] \quad = \quad & O\Bigg(\sum_{a \in \mathcal{A}, a \neq a^*} \frac{\sigma^2 \log(T)}{\Delta_a} + \tau_c \log\left(\tau_c \log\left(\frac{1}{\Delta_a} \right) \right) \\ & + \tau_c \log\left(\frac{\sigma^2 \log(T)}{\Delta_a^2} \right) \Bigg), \end{aligned} \tag{2}$$

where $\Delta_a = x_{a^*}^* - x_a^*$ *is the suboptimality gap for action a.*

The dominant term in (2) depends on $\frac{T}{\tau_c}$ which quantifies how many times we can afford to converge to equilibrium, and is application dependent. For example, $\mathbb{E}[R(T)] = O(\sqrt{T}\log(T))$ for $\tau_c = O(\sqrt{T})$, and $\mathbb{E}[R(T)] = O(\log(T)\log\log(T))$ for $\tau_c = O(\log(T))$.

Stochastic bandits can be viewed as a special case of equilibrium bandits where the convergence to equilibrium is instantaneous. Equilibrium bandits is a more challenging problem since the rewards are no longer i.i.d. over time. Algorithms like UCB treat the rewards as independent over time and do not account for how far the system is from the equilibrium. Consequently, the UCB algorithm has linear regret for equilibrium bandits [8, Theorem 3].

Equilibrium bandits are a special case of adversarial bandits, where any sequence of rewards is allowed. However, adversarial regret bounds are significantly weaker than our regret bound since they compare to the best action in hindsight. In contrast, our regret resembles the more demanding regret of stochastic bandits, which compares to the "absolute" optimal action. Therefore, adversarial bandit algorithms also achieve linear regret for equilibrium bandits.

In our notation, the regret for the adversarial problem would be

$$\mathbb{E}[R_{adv}(T)] = \max_a \sum_{t=1}^{T} f(a; z_t) - \mathbb{E}\left[\sum_{t=1}^{T} f(a_t; z_t) \right].$$

The adversarial regret looks at the state sequence $\{z_t\}_{t=1}^T$ as given and ignores the fact that our action sequence $\{a_t\}_{t=1}^T$ impacted the state sequence. In the epidemic control example, this would mean that the adversarial regret tries to find the best action given the number of infections over time, as though these numbers could not be avoided by an agent who would have taken better actions.

3 UECB ALGORITHM

In this section, we present our novel UECB algorithm designed for equilibrium bandits. To provide intuition, we start by analyzing the simpler case where the rewards are not noisy.

3.1 The Noiseless Case

Consider the special case where the rewards are not noisy, i.e., $\eta_t = 0$ a.s. for all t. Hence, the agent directly observes $y_t = x_t = f(a_t; z_t)$. Since there is no noise, it should be possible to find the optimal action in bounded time which results in an expected regret of $O(1)$.

A naive algorithm would pick each action consecutively a sufficiently large number of times (denoted by t_{try}), to allow the system to approach the equilibrium corresponding to this action. The agent will then know the reward at equilibrium for each action with arbitrarily low error, and can then commit to the optimal action. This naive algorithm is actually the default choice in many real-life scenarios. This algorithm can achieve sublinear regret if t_{try} is above a threshold, which depends on τ_c and the suboptimality gap. Since the suboptimality gap is unknown to the agent, the naive algorithm achieves linear regret in general.

Instead, we propose the "Upper Equilibrium Confidence Bound" (UECB) algorithm for the noiseless case based on 'convergence bounds'. Suppose the state of the system at time t is z_t and action a is taken for ℓ timesteps consecutively after that. Then, using Assumption 1, we have

$$\|z_{t+\ell} - z_a^*\| \leq e^{-\frac{\ell}{\tau_c}} \|z_t - z^*\|.$$

Using the assumption that $\mathcal{Z} \subseteq \mathcal{B}_{0.5}(\|\cdot\|)$, we have $\|z_t - z^*\| \leq 1$. Without this assumption, the only modification required would be to replace L with $2L \max_{z \in \mathcal{Z}} \|z\|$ henceforth in the paper. Now

using the Lipschitz property of the reward function, we deduce that

$$f(a; z_{t+\ell}) - Le^{-\frac{\ell}{\tau_c}} \leq f(a; z_a^*) \leq f(a; z_{t+\ell}) + Le^{-\frac{\ell}{\tau_c}}. \quad (3)$$

We assume that the agent knows τ_c, which is a worst-case bound on the actual convergence time. Given this knowledge, $f(a; z_{t+\ell}) + Le^{-\ell/\tau_c}$ is the maximum possible reward action a can yield at the equilibrium point corresponding to a. As we demonstrate in the applications of Section 5, knowing a bound on τ_c is significantly easier than knowing the system parameters.

In practice, the agent can often observe more than just the reward (e.g., the state z_t or the convergence rate $c(a; z_t)$). With more knowledge, the above bounds can be tightened without affecting our analysis or the UECB algorithm that uses them as input.

It is necessary to play an action consecutively for some time to get an accurate enough estimation of the reward at equilibrium given that action. However, we do not want to always wait for the system to converge as it might waste precious time and incur significant regret. The idea behind the UECB algorithm (Algorithm 1) is to play actions that seem to lead to good equilibria for an increasing number of turns to allow it to reach closer to convergence. This along with the above-mentioned convergence bound serves as the basic intuition behind UECB. Instead of switching an action at every timestep, UECB chooses an action to be played consecutively over a full 'epoch'. The epoch length increases with the number of epochs the chosen action has been played before.

Let $m_{a,n}$ denote the number of epochs action a has been chosen for till the end of epoch n, and let t_n denote the total number of timesteps till the end of epoch n. Additionally, let $t_{a,n}$ denote the number of timesteps action a has been played till the end of epoch n. Let the action in the next epoch a_{n+1} be chosen as

$$a_{n+1} = \arg\max_{a \in \mathcal{A}} UECB_{a,n}$$

where $UECB_{a,n}$ is defined below. The length of epoch $n+1$ is chosen as $\ell_{n+1} := 2\rho_2 \exp(\rho_1(m_{a_{n+1},n}+1))$, where ρ_1, ρ_2 are positive parameters, as explained after Theorem 2. The agent plays this action for the complete epoch and observes the reward obtained at the last timestep of that epoch $x_{t_n+\ell_{n+1}} = x_{t_{n+1}}$, denoted by $\hat{x}_{a,n+1}$. Finally at end of epoch $n+1$, $UECB_{a_{n+1},n+1}$ is updated as follows:

$$UECB_{a_{n+1},n+1} = \hat{x}_{a,n+1} + Le^{-\frac{\ell_{n+1}}{\tau_c}}$$

and $UECB_{a,n+1} = UECB_{a,n}$ for all other actions $a \neq a_{n+1}$.

We now give a bound on the maximum number of times the UECB algorithm chooses a non-optimal action and a bound on the maximum possible regret in the noiseless case.

Theorem 2. *Let $\eta_t = 0$, a.s. for all t (i.e., no noise). Then for an equilibrium bandit instance satisfying Assumption 1, with a Lipschitz continuous reward function bounded in $[0,1]$,*

(a) Algorithm 1 chooses a suboptimal action, i.e., action other than a^*, only for a finite number of turns $\hat{T}$ where

$$\hat{T} = O\left(\sum_{a \in \mathcal{A}, a \neq a^*} \tau_c \log\left(\frac{1}{\Delta_a}\right)\right).$$

(b) For all $t > 0$,

$$R(t) = O\left(\sum_{a \in \mathcal{A}, a \neq a^*} \Delta_a \tau_c \log\left(\frac{1}{\Delta_a}\right) + \tau_c \log\left(\tau_c \log\left(\frac{1}{\Delta_a}\right)\right)\right).$$

Part (a) of Theorem 2 is based on the maximum number of times a suboptimal action may need to be played consecutively to differentiate it from the optimal action. This gives the maximum number of epochs that may be required for each suboptimal action and hence the number of steps required in the worst case. The first term in the regret bound is obtained by simply multiplying the suboptimality gap for each action. The second term stems from the maximum number of times UECB switches between actions. Switching to a new action resets the convergence of the system to a new equilibrium, which incurs $O(\tau_c)$ regret per switch.

We can construct scenarios where any algorithm that achieves sublinear regret would have to play each suboptimal action a at least $\Omega(\tau_c \log(1/\Delta_a))$ times to distinguish it from the optimal action. To see that, consider converging reward sequences that are identical for all actions for the first $\tau_c \log(1/\Delta_a)$ turns and start differing only after. This implies a worst-case lower bound of $\Omega(\tau_c \Delta_a \log(1/\Delta_a))$ for the noiseless case [8, Theorem 4].

The exponential increase in the epoch lengths is chosen to obtain bound of the form $O(\tau_c \log(1/\Delta_a))$ in Theorem 2 part (a). Any increasing sequence of epoch lengths will give a finite regret but will not have a better bound orderwise. For example, linearly increasing epoch lengths i.e., $\ell_n \propto m_{a_n,n}$, yield a bound of $O((\tau_c \log(1/\Delta_a))^2)$. On the other hand, even if epoch lengths grew faster than exponential, e.g., $\ell_n \propto \exp(\exp(m_{a_n,n}))$, we would still obtain a bound of $O(\tau_c \log(1/\Delta_a))$. Epoch lengths that increase too fast do not do well in practice since they waste precious time on suboptimal actions.

3.2 The Noisy Case

We now consider the general case where $\eta_t \neq 0$. The UECB algorithm given for the noiseless case cannot be used here as it only considers the final reward observed, which can be very noisy. To deal with the noise, it is necessary to average multiple observations. The estimated expected reward corresponding to the equilibrium point of an action has two kinds of errors - due to the i.i.d. noise and due to the distance from equilibrium (i.e., "equilibrium noise"). Averaging creates a trade-off between the two errors. For example, averaging over all rewards observed for an action reduces the i.i.d. noise but increases the equilibrium noise since early rewards were earned far from equilibrium. Similarly, considering only the last reward has low equilibrium noise but high i.i.d. noise.

Hence we propose the UECB algorithm for the noisy case inspired by the popular UCB (Upper Confidence Bound) algorithm [18]. We have the same epoch-based structure as before, i.e., $\ell_{n+1} := 2\rho_2 \exp\left(\rho_1(m_{a_{n+1},n}+1)\right)$. For action a_n, define

$$\hat{x}_{a_n,n} = \frac{2}{\ell_n} \sum_{t=t_{n-1}+\frac{\ell_n}{2}+1}^{t_n} y_t,$$

i.e., the average of the second half of the last epoch corresponding to that action. For other actions $a \neq a_n$, $\hat{x}_{a,n} = \hat{x}_{a,n-1}$. Also, for all actions a, define n_a as the last epoch action a was played before the end of epoch n. Then ℓ_{n_a} is the length of the last epoch action a

Algorithm 1: UECB Algorithm

Initialization: Let $m_{a,0} = 0$ for all $a \in \mathcal{A}$.
Input: Constant $\tau_c \geq 1, L > 0$ and parameters $\rho_1, \rho_2 > 0$.
For epochs $n = 1$ **to** K **do**
(1) Play action $a_n = n$ for $\ell_n = 2\rho_2 e^{\rho_1}$ timesteps from $t = t_{n-1} + 1$ to $t = t_n = t_{n-1} + \ell_n$.
(2) $\hat{x}_{a,n}, m_{a,n}, UECB_{a,n} = UPDATE(y_{t_{n-1}+1}, \ldots, y_{t_n})$
For epochs $n \geq K + 1$ **do**
(1) Choose action $a_n = \arg\max_{a \in \mathcal{A}} UECB_{a,n-1}$.
(2) Play action a_n for $\ell_n = 2\rho_2 \exp(\rho_1(m_{a_n,n-1} + 1))$ timesteps from $t = t_{n-1} + 1$ to $t = t_n = t_{n-1} + \ell_n$.
(3) $\hat{x}_{a,n}, m_{a,n}, UECB_{a,n} = UPDATE(y_{t_{n-1}+1}, \ldots, y_{t_n})$
End
function $UPDATE(y_{t_{n-1}+1}, \ldots, y_{t_n})$
(1) $m_{a_n,n} = m_{a_n,n-1} + 1$ and $m_{a,n} = m_{a,n-1}$ for $a \neq a_n$.
(2) if (noiseless):
(a) $\hat{x}_{a_n,n} = y_{t_n}$ and $\hat{x}_{a,n} = \hat{x}_{a,n-1}$ for $a \neq a_n$.
(b) $UECB_{a,n} = \hat{x}_{a,n} + Le^{-\frac{1}{\tau_c}\ell_{n_a}}$ for $a \in \mathcal{A}$.
(3) if (noisy):
(a) $\hat{x}_{a_n,n} = \frac{2}{\ell_n} \sum_{t=t_{n-1}+\frac{\ell_n}{2}+1}^{t_n} y_t$ and $\hat{x}_{a,n} = \hat{x}_{a,n-1}$ for $a \neq a_n$.
(b) For all actions $a \in \mathcal{A}$,

$$UECB_{a,n} = \hat{x}_{a,n} + \frac{2}{\ell_{n_a}} \frac{L \exp(-\frac{1}{\tau_c}(1 + \frac{\ell_{n_a}}{2}))}{1 - \exp(-\frac{1}{\tau_c})} + \sqrt{\frac{4\sigma^2}{\ell_{n_a}} \log\left(\frac{2}{\delta_n}\right)},$$

where $\delta_n = 1/t_n^3$.

was played before the end of epoch n, i.e., $\ell_{n_a} = 2\rho_2 \exp(\rho_1 m_{a,n})$. Then for all actions, we define:

$$UECB_{a,n} := \hat{x}_{a,n} + \frac{2}{\ell_{n_a}} \frac{L \exp(-\frac{1}{\tau_c}(1 + \frac{\ell_{n_a}}{2}))}{1 - \exp(-\frac{1}{\tau_c})} + \sqrt{\frac{4\sigma^2}{\ell_{n_a}} \log\left(\frac{2}{\delta_n}\right)},$$

where $\delta_n := \frac{1}{t_n^3}$. Similar to the intuition behind the UCB algorithm, this $UECB_{a,n}$ is defined to ensure that $UECB_{a,n} \geq x_a^*$ with probability of at least $1 - \delta_n$. This is proved in Lemma 1. Finally, as before, the action for the next epoch is chosen as follows:

$$a_{n+1} = \arg\max_{a \in \mathcal{A}} UECB_{a,n}.$$

Theorem 1 gives a regret bound on the UECB algorithm. Next, we make a few comments on the algorithm and its bound:

- The first term in the regret bound, $O(\log(T)/\Delta_a)$, appears also in the regret bound for the UCB algorithm in stochastic multi-armed bandits and stems from the noisy observations that both scenarios share. The second term in the regret bound, $O(\tau_c \log(\tau_c))$, also appears in the regret bound for the noiseless case given in Theorem 2. This second term dominates the first term if convergence to equilibrium takes significant time which is the case in applications.
- [8] gives a lower bound of $\Omega(\log(T)/\Delta_a + \tau_c \Delta_a \log(1/\Delta_a))$ for equilibrium bandits. The first and second terms in this lower bound are obtained using the lower bounds for stochastic and noiseless equilibrium bandits, respectively, both of which are special cases of equilibrium bandits. Hence, UECB is order-wise optimal in T and Δ_a while being optimal up to logarithmic factors in τ_c.
- For each action, UECB only uses the rewards observed during the current epoch. It also uses only the latter half of that epoch. We chose the 'half' fraction arbitrarily for simplicity and any other constant fraction yields the same order of magnitude dependencies. Another possible modification is to employ a weighted average over all the past rewards [12].

4 REGRET ANALYSIS

In this section, we explain our proof strategy by breaking the proof of Theorem 1 into lemmas. While the proof generally follows that of UCB [18], significant modifications are needed due to the converging nature of the rewards, and the epoch-based structure. In particular, UECB has to balance the tradeoff between the i.i.d. noise and the "equilibrium noise", which measures the distance to equilibrium. This tradeoff is unique to our problem.

The first lemma gives a probabilistic bound on $|\hat{x}_{a,n} - x_a^*|$ and motivates the definition of $UECB_{a,n}$. This gives a probabilistic upper bound on how far away our estimate of the equilibrium reward is. The difference between $\hat{x}_{a,n}$ and x_a^* has two terms - one due to the i.i.d. noise and the other due to "equilibrium noise".

Lemma 1. *The UECB algorithm maintains the following statements:*
(a) The inequality

$$|\hat{x}_{a,n} - x_a^*| \leq \sqrt{\frac{4\sigma^2}{\ell_{n_a}} \log\left(\frac{2}{\delta_n}\right)} + \frac{2}{\ell_{n_a}} \frac{L \exp(-\frac{1}{\tau_c}(1 + \frac{\ell_{n_a}}{2}))}{1 - \exp(-\frac{1}{\tau_c})},$$

holds with probability of at least $1 - \delta_n$.
(b) $UECB_{a,n} \geq x_a^*$ *with probability of at least* $1 - \delta_n$.
(c) Define

$$\ell_{a,n}^{(1)} := \frac{64\sigma^2}{\Delta_a^2} \log\left(\frac{2}{\delta_n}\right) \quad \text{and} \quad \ell_{a,n}^{(2)} := 2\tau_c \log\left(\frac{8L}{\Delta_a}\right).$$

Then given that $\ell_{n_a} \geq \ell_{a,n}^{(1)}$ *and* $\ell_{n_a} \geq \ell_{a,n}^{(2)}$ *we have* $\hat{x}_{a,n} \leq x_a^* + \frac{\Delta_a}{2}$ *with probability greater than* $1 - \delta_n$.

Part (b) and (c) above are direct implications of part (a) and the definition of $UECB_{a,n}$. The conditions on ℓ_{n_a} in part (c) can be easily translated to conditions on $m_{a,n}$, i.e., the number of epochs during which a has been played. Let the corresponding number of epochs be $m_{a,n}^{(1)}$ and $m_{a,n}^{(2)}$ respectively, i.e., $\ell_{a,n}^{(i)} = 2\rho_2 \exp\left(\rho_1 m_{a,n}^{(i)}\right)$ for $i = 1, 2$. Here $m_{a,n}^{(1)}$ is the number of epochs required for the noise term to be sufficiently small, and $m_{a,n}^{(2)}$ is the number of epochs required for the second term, stemming from the convergence time, to be sufficiently small. A large $m_{a,n}$ implies that a was played consecutively for a higher number of turns which reduces the error due to i.i.d. noise and brings the system closer to equilibrium.

The next lemma shows that with high probability, the $UECB$ algorithm identifies suboptimal arms given that they have been played for a sufficiently large number of times. It also gives an upper bound on the probability of playing a suboptimal action given that the action has been played a sufficient number of times.

Lemma 2. *At the end of epoch n, if a suboptimal arm* $a \neq a^*$ *has been played for enough epochs, such that* $\ell_{n_a} \geq \max\{\ell_{a,n}^{(1)}, \ell_{a,n}^{(2)}\}$ *(defined*

in Lemma 1) then $UECB_{a,n} \leq UECB_{a^*,n}$ with probability at least $1 - 2\delta_n$. Therefore, under these conditions, the probability that the UECB algorithm plays action a in the $(n+1)^{th}$ epoch is bounded by:

$$P\left(a_{n+1} = a \mid \ell_{n_a} \geq \ell_{a,n}^{(1)}, \ell_{n_a} \geq \ell_{a,n}^{(2)}\right) \leq 2\delta_n.$$

Note that the expected instantaneous loss at time t when action a is taken can be split as follows:

$$x_{a^*}^* - x_t = (x_{a^*}^* - x_a^*) + (x_a^* - x_t). \tag{4}$$

The first term in (4) is the difference in the rewards at equilibrium between the optimal arm and a suboptimal arm. We first bound the regret corresponding to the first term, which depends on the number of times an action is taken multiplied by the suboptimality gap. The next lemma bounds the expected number of times *(and not epochs)* a suboptimal action is played.

Lemma 3. *For any suboptimal arm $a \neq a^*$, the expected number of timesteps the UECB algorithm plays a can be bounded as follows:*

$$\mathbb{E}\left[\sum_{k=1}^{n} \ell_k I\{a_k = a\}\right] = O\left(\frac{\sigma^2 \log(t_n)}{\Delta_a^2} + \tau_c \log\left(\frac{1}{\Delta_a}\right)\right)$$

where $I\{\cdot\}$ is 1 when $\{\cdot\}$ is true and 0 otherwise (i.e., an indicator).

The second term in (4) denotes the convergence error, or "equilibrium noise". The regret accumulated due to this term can be shown to be $O(\tau_c)$ times the number of arm switches, which is in turn bounded by the number of epochs where suboptimal actions were played. The next lemma bounds this number.

Lemma 4. *For any suboptimal arm $a \neq a^*$, the expected number of epochs the UECB algorithm chooses a can be bounded as follows:*

$$\mathbb{E}\left[\sum_{k=1}^{n} I\{a_k = a\}\right] = O\left(\log\left(\frac{\sigma^2 \log(t_n)}{\Delta_a^2}\right) + \log\left(\tau_c \log\left(\frac{1}{\Delta_a}\right)\right)\right).$$

This gives rise to the final term in the bound in Theorem 1. Combining Lemma 3 with Lemma 4 gives us the following final lemma which gives a regret bound at the end of each epoch.

Lemma 5. *Let t_n denote the time at the end of epoch n, then the UECB algorithm maintains:*

$$\begin{aligned}\mathbb{E}[R(t_n)] \quad = \quad & O\Bigg(\sum_{a \in \mathcal{A}, a \neq a^*} \frac{\sigma^2 \log(t_n)}{\Delta_a} + \tau_c \log\left(\tau_c \log\left(\frac{1}{\Delta_a}\right)\right) \\ & + \tau_c \log\left(\frac{\sigma^2 \log(t_n)}{\Delta_a^2}\right)\Bigg).\end{aligned}$$

The lemma above proves our UECB regret bound, but only at timesteps that are at the end of some epoch. To complete the proof of Theorem 1, we just need to prove a similar regret bound for all T. To that end, define τ as the time at which the last epoch ended, i.e., the ongoing epoch started at $t = \tau + 1$. Then we divide $[1, T]$ into two intervals: $[1, \tau]$ and $[\tau + 1, T]$. The lemma above gives a bound for the regret accumulated during $[1, \tau]$. The result for Theorem 1 is obtained by showing that the regret achieved during $[\tau + 1, T]$ is bounded by a constant times the regret achieved during $[1, \tau]$.

5 APPLICATIONS

In this section, we detail two real-life problems that the equilibrium bandits framework can model. The agent is a policymaker who learns the policy that maximizes the collective good. However, the impact of any policy cannot be seen instantly as society interacts in a game-theoretic manner and converges to an equilibrium.

5.1 SIS Epidemic Model

We consider the Susceptible-Infectious-Susceptible (SIS) model of epidemics [2, 21] as the underlining unknown dynamics to be controlled by the decision-maker. In this model, a susceptible individual becomes infected with some probability after contacting an infected individual and remains infected for a random period of time. Once the individual recovers, they return to the susceptible class since the disease does not provide any long-lasting immunity. Examples for diseases that follow the SIS model are influenza, meningitis, and tuberculosis [14].

Specifically, we consider the networked SIS model with a graph with M nodes and a symmetric weighted adjacency matrix A. These nodes can represent communities, cities, or countries. The weight A_{ij} in A is the contact probability between nodes i and j. Let $I(i,t)$ be the fraction of infected individuals in node i at time t. Then the discretized differential equation is given by:

$$I(i, t+\Delta t) = I(i,t) + \left(\beta(1 - I(i,t)) \sum_{j=1}^{M} A_{ij} I(j,t) - \gamma I(i,t)\right)\Delta t. \tag{5}$$

Here $0 < \Delta t \ll 1$ is the stepsize for discretization which is assumed to be sufficiently small. $\beta > 0$ is the infection rate and $\gamma > 0$ is the recovery rate. Let $I(t)$ be the M-dimensional vector with the elements $I(i,t)$, then equation (5) can be written as

$$\begin{aligned} I(t + \Delta t) \quad &= \quad h(\beta, \gamma, A, \Delta t; I(t)) \\ &:= \quad I(t) + (\beta(\mathbb{1}_M - \mathrm{diag}(I(t)))AI(t) - \gamma I(t))\,\Delta t, \end{aligned} \tag{6}$$

where $\mathbb{1}_M$ is the identity matrix of dimension M and $\mathrm{diag}(I(t))$ is a diagonal matrix with the elements of the vector $I(t)$.

The decision-maker is the government or the policy-maker. Examples of actions can be the enforcement of masks, advertisements to increase awareness, and different types of lockdowns, e.g., shutting down schools or offices. These actions change the contact patterns between individuals and the rate of infection, which dictate the adjacency matrix and infection rate for this action, denoted by A_a and β_a, respectively, for action a. Δt is the stepsize for the discretization and is typically much smaller than the time step at which the infection rates or the rewards are actually observed, which can range from a few days to a few weeks. Let $z_t = I(t)$ be the state of the system at time t. Then, for action a,

$$z_{t+1} = g(a; z_t) = h^{(1/\Delta t)}(\beta_a, \gamma, A_a, \Delta t; z_t).$$

The cost function $-f(a; \cdot)$ can be a combination of the operation cost of the policies and the health damages due to the disease.

The government does not know the functions f and g. In particular, it is unlikely that the government can estimate the matrix A_a for each action in a large-scale setting. In addition, depending on the resolution, the government may or may not know the fraction of infected individuals in each node, (e.g., neighborhoods as opposed

to individuals). Fortunately, as explained next, UECB only requires weak bounds on the problem parameters to perform very well.

Let λ_a^{max} be the maximal eigenvalue of A_a. We assume that $\beta_a \lambda_a^{max} > \gamma_a$ for all actions. This assumption implies that a non-zero stable equilibrium point exists for the iteration given by (6) for all actions. Let this equilibrium be I_a^*. The zero vector (i.e., no infections) is an unstable equilibrium in this case. If this assumption were false, i.e., $\beta_a \lambda_a^{max} \leq \gamma_a$, then there exists only one equilibrium point given by the zero vector which is also stable. We only consider actions that satisfy our assumption because, unfortunately, there are often no policies that completely eradicate the epidemic. Next, we show that this g satisfies Assumption 1):

Proposition 1. *For each action a, let I_a^* be the non-zero stable equilibrium of the iteration in (6), where β_a, γ and A_a satisfy $\beta_a \lambda_a^{max} > \gamma$ for the maximal eigenvalue of A_a, λ_a^{max}. If $I(t) > 0$, then*

$$\|I(t+\Delta t) - I_a^*\|_1 \leq \max_{1 \leq i \leq M} \left(1 - \beta_a \Delta t \sum_{j=1}^{M} A_{a_{ij}} I(j,t)\right) \|I(t) - I_a^*\|_1, \tag{7}$$

where $\|\cdot\|_1$ denotes the ℓ_1 norm.

We just need a bound on β_a and on the elements of matrix A_a for all actions a to employ the bound in (7). If the policymaker does not know the number of infections in each node (but only the "reward"), then they just need a bound on the number of infections. This bound is easy to deduce due to our assumption that $\beta_a \lambda_a^{max} > \gamma$. Let $\beta_a \sum_{j=1}^{M} A_{a_{ij}} I(j,t) > \alpha$ for all nodes i and all actions. Then,

$$\begin{aligned}\|z_{t+1} - z_a^*\|_1 &\leq \left(\max_{1 \leq i \leq M} \left(1 - \beta_a \Delta t \sum_{j=1}^{M} A_{a_{ij}} I(j,t)\right)\right)^{1/\Delta t} \|z_t - z_a^*\|_1 \\ &\lessapprox e^{-\alpha} \|z_t - z_a^*\|_1.\end{aligned}$$

Then $c(a;z)$ as defined in equation (1) is approximately equal to $e^{-\alpha}$, which becomes accurate as Δt goes to 0. If the number of infections in each node is known, then better bounds can be obtained.

5.2 Strongly Monotone Game

Our second example is deterministic gradient-based learning in continuous games [20]. The underlying system in this example is a set of M players. Each player $i \in \mathcal{M} = \{1, \ldots, M\}$ has their own decision variable $z_{i,t} \in \mathcal{Z}_i \subset \mathbb{R}^d$ at time t. Here each $\mathcal{Z}_i$ is a convex and compact set. The state variable of the system is just a concatenation of all the decision variables, i.e., $z_t = (z_{1,t}, \ldots, z_{M,t}) \in \mathcal{Z}_1 \times \ldots \times \mathcal{Z}_M \subset \mathbb{R}^{dM}$. Each player has their own utility function, which depends on the decision variables of all the players and is parameterized by the action taken by the agent. At time t, if the action taken by the agent is a_t, then the utility function for player i is given by $u_i(a_t; z_t)$. Each player seeks to maximize their utility function and can only control their own decision variable. We assume that all players have access to the decision variables of each player and their own utility function, but not to the utility functions of other players. Then at each time t, each agent updates their decision variables as follows:

$$z_{i,t+1} = z_{i,t} + \alpha h_i(a_t; z_t), \tag{8}$$

where $h_i(a; z_t) = \frac{\partial u_i(a;z_t)}{\partial z_{i,t}}$. We have considered a constant step-size α for simplicity; the following results can easily be generalized to a decreasing step-size. The agent observes noisy rewards based on its own reward function: $f(a_t; z_t)$.

We make certain assumptions on the underlying game to ensure that the assumptions for our bandit problem are satisfied. To that end, define $H(a;z) = (h_1(a;z), \ldots, h_M(a;z))$, i.e., the concatenation of all gradients. Then we assume that

$$\langle z' - z, H(a;z') - H(a;z) \rangle \leq -\lambda_a \|z' - z\|_2^2, \ \forall a \in \mathcal{A}.$$

This implies that for all actions, $\mathcal{G}_a = (\mathcal{M}, \{\mathcal{Z}_i\}, \{u_i(a;\cdot)\})$ is a strongly monotone game with parameter $\lambda_a > 0$. Then for each action a, $\mathcal{G}_a$ has a unique pure Nash equilibrium z_a^* [28] which acts as the equilibrium corresponding to that action. For an action a, the Nash equilibrium z^* is defined as the decision profile which satisfies: $u_i(a; z_i^*, z_{-i}^*) \geq u_i(a; z_i, z_{-i}^*)$ for all $z_i \in \mathcal{Z}_i$ and for all $i \in \mathcal{M}$. Note that z_{-i} denotes the decision variables for all players except i. This ensures that part (a) of Assumption 1 is satisfied. We also assume that, for all a, $H(a,.)$ is β_a-Lipschitz continuous:

$$\|H(a;z) - H(a;z')\|_2 \leq \beta_a \|z - z'\|_2.$$

The next proposition shows that such games satisfy Assumption 1:

Proposition 2. *The iterates given by (8), with action $a_t = a$, satisfy:*

$$\|z_{t+1} - z_a^*\|_2 \leq \sqrt{1 - 2\lambda_a \alpha + \alpha^2 \beta_a^2} \|z_t - z_a^*\|_2.$$

For a sufficiently small step-size ($\alpha \leq \frac{2\lambda_a}{\beta_a^2}$), this proposition shows that the distance of the state z_t from z_a^* decreases when action a is taken. A sufficiently small step-size can be avoided with a sequence of decreasing step-sizes, which instead would imply that the system satisfies Assumption 1 from some t_0 onwards.

As a concrete example, consider a resource allocation game [1, 6] where there are d resources and each player's decision variable $z_i = (z_i^1, \ldots, z_i^d)$ denotes how much to use of each resource. The utility function of player i depends on the value it assigns to each resource and the price of each resource. Here, action a by the policymaker means that each player i has access only to the subset of resources $\mathcal{R}_i(a) \subseteq \{1, \ldots, d\}$, so $z_i^\ell = 0$ for $\ell \notin \mathcal{R}_i(a)$. The reward function for the agent is the sum of utilities of the players $f(a;z) = \sum_{i=1}^{M} u_i(a;z)$.

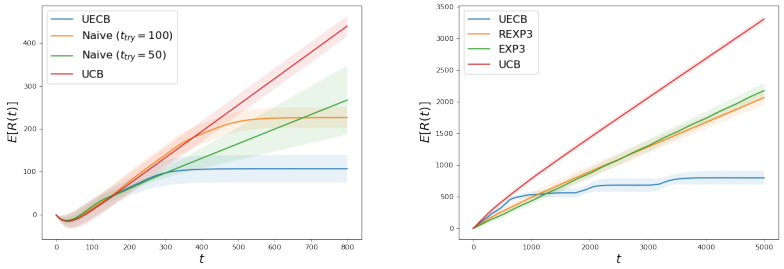
6 SIMULATIONS

In this section, we simulate the applications from Section 5. Each curve is the average of 100 random realizations, and has been plotted along with the standard deviation region. The randomness in the noiseless case stems from the random initializations.

6.1 SIS Epidemic

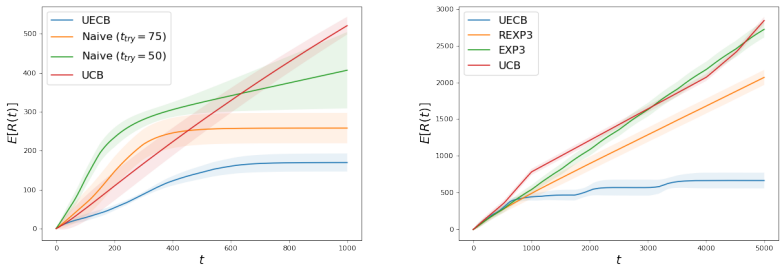
We simulate a system with $K = 4$ actions and $M = 10$ nodes. For each action a, we generate a random sparse symmetrical matrix A_a. We use $\gamma = 0.01$ for all actions. The values for β_a for the 4 actions are $0.011, 0.012, 0.013$ and 0.014 respectively [14]. We use the **cost** function $f(a;z) := w_{0,a} + w_a^T z$,, where $w_a \in (0,1]^M$ is the health cost vector and $w_{0,a} \in (0,1]$ is the operational cost which only depends on the action. Clearly, f is Lipschitz with $L = 1$.

To implement UECB, we only assume that $\beta_a > 0.05$ for all actions and that the sum of each row of A is at least 1, while the



(a) Noiseless **(b) Noisy**

Figure 1: Simulation plots for SIS epidemic control in the (a) noiseless, and (b) noisy cases



(a) Noiseless **(b) Noisy**

Figure 2: Simulation plots for resource allocation game in the (a) noiseless, and (b) noisy cases.

actual values are unknown and in $[3, 5]$. Additionally, the infected fraction for each node is unknown and only the cost is known.

Figure 1a compares the performance of UECB for the noiseless case with a naive algorithm where each action is played for t_{try} consecutive timesteps in the beginning (see subsection 3.1). Then, the naive algorithm plays the arm that had the best reward at the end of its epoch for the rest of the timesteps. As expected, our UECB algorithm outperforms the naive algorithm for both small and large t_{try}. For small t_{try}, we do not give enough time for the system to converge which then commits to a suboptimal action, yielding linear regret. For large t_{try}, the system gets close to equilibrium for each action, but wastes time on suboptimal actions. This gives an $O(1)$ regret, but it is still worse than that of UECB. For the noisy case, Figure 1b shows that UECB achieves sublinear regret over time while UCB, EXP3 and REXP3 [5] do not.

6.2 Resource Allocation Game

We consider electricity grids as our resource allocation game [11, 19], with $d = 10$ resources and $M = 1000$ players. The utility function for each player i is defined as

$$u_i(z) = \sum_{\ell=1}^{d} \left(\gamma_{i,\ell} \log(1 + z_i^\ell) - \zeta_{i,\ell} z_i^\ell s_\ell \right),$$

where $s_\ell = \sum_{i=1}^{M} z_i^\ell$. $\gamma_{i,\ell}$ and $\zeta_{i,\ell}$ are chosen uniformly at random in $[0.8, 1]$. It can be easily verified that this function satisfies our assumptions. For each action, the subsets of resources that can be chosen by each player are generated randomly.

We again assume very little knowledge about the system. The agent knows α, but only uses bounds on λ_a and β_a. Here, we assume that the agent can observe the current state, since monitoring which player picked what resource is natural in practice. The results, given in Figure 2, are similar to those from the SIS epidemic scenario.

7 CONCLUSIONS

In this paper, we presented equilibrium bandits, a new bandit problem, designed to deal with systems that converge to equilibrium over time. The agent can control some parameters of this system that dictate the resulting equilibrium. While the agent only observes the real-time impact of their actions, their aim is to find the set of parameters that give the best performance at equilibrium.

We proposed Upper Equilibrium Concentration Bound (UECB), a new algorithm for equilibrium bandits that assumes very little about the system. The key innovation of UECB is the use of 'convergence bounds' which bound how far the system is from the equilibrium at any given point. We proved regret bounds for UECB which are optimal up to logarithmic factors. We showed that two applications, epidemic control and resource allocation games, fall under the framework of equilibrium bandits. We simulated UECB to confirm the theoretical performance guarantees for these applications.

By introducing a new bandit model, our work opens up many new research avenues. An important extension is to be able to learn the evolution system parameters (i.e., τ_c and L) on the fly, instead of using fixed worst-case bounds on these parameters. Another significant extension is to systems that evolve stochastically, which would allow the equilibrium bandits framework to include reinforcement learning algorithms and stochastic gradient-based games.

REFERENCES

[1] Shipra Agrawal, Morteza Zadimoghaddam, and Vahab Mirrokni. 2018. Proportional allocation: Simple, distributed, and diverse matching with high entropy. In *International Conference on Machine Learning*. PMLR, 99–108.

[2] Linda J.S. Allen. 1994. Some discrete-time SI, SIR, and SIS epidemic models. *Mathematical Biosciences* 124, 1 (1994), 83–105. https://doi.org/10.1016/0025-5564(94)90025-6

[3] Tansu Alpcan and Lacra Pavel. 2009. Nash equilibrium design and optimization. In *Game Theory for Networks, 2009. GameNets' 09. International Conference on.*

[4] Peter Auer, Nicolo Cesa-Bianchi, Yoav Freund, and Robert E Schapire. 2002. The nonstochastic multiarmed bandit problem. *SIAM journal on computing* 32, 1 (2002), 48–77.

[5] Omar Besbes, Yonatan Gur, and Assaf Zeevi. 2014. Stochastic multi-armed-bandit problem with non-stationary rewards. *Advances in neural information processing systems* 27 (2014).

[6] Ilai Bistritz and Nicholas Bambos. 2021. Online learning for load balancing of unknown monotone resource allocation games. In *International Conference on Machine Learning*. PMLR, 968–979.

[7] Ilai Bistritz, Dor Kahana, Nicholas Bambos, Irad Ben-Gal, and Dan Yamin. 2019. Controlling contact network topology to prevent measles outbreaks. In *2019 IEEE Global Communications Conference (GLOBECOM)*. IEEE, 1–6.

[8] Siddharth Chandak, Ilai Bistritz, and Nicholas Bambos. 2023. Equilibrium Bandits: Learning Optimal Equilibria of Unknown Dynamics. https://doi.org/10.48550/ARXIV.2302.13653

[9] Siddharth Chandak, Vivek S. Borkar, and Parth Dodhia. 2022. Concentration of Contractive Stochastic Approximation and Reinforcement Learning. *Stochastic Systems* 12, 4 (2022), 411–430.

[10] Lili Chen, Kevin Lu, Aravind Rajeswaran, Kimin Lee, Aditya Grover, Misha Laskin, Pieter Abbeel, Aravind Srinivas, and Igor Mordatch. 2021. Decision transformer: Reinforcement learning via sequence modeling. *Advances in neural information processing systems* 34 (2021), 15084–15097.

[11] Ruilong Deng, Zaiyue Yang, Mo-Yuen Chow, and Jiming Chen. 2015. A survey on demand response in smart grids: Mathematical models and approaches. *IEEE Transactions on Industrial Informatics* 11, 3 (2015), 570–582.

[12] Aurélien Garivier and Eric Moulines. 2008. On Upper-Confidence Bound Policies for Non-Stationary Bandit Problems. https://doi.org/10.48550/arxiv.0805.3415

[13] Sergio Grammatico. 2017. Dynamic control of agents playing aggregative games with coupling constraints. *IEEE Trans. Automat. Control* 62, 9 (2017), 4537–4548.

[14] A. Gray, D. Greenhalgh, L. Hu, X. Mao, and J. Pan. 2011. A Stochastic Differential Equation SIS Epidemic Model. *SIAM J. Appl. Math.* 71, 3 (2011), 876–902. https://doi.org/10.1137/10081856X

[15] Jens Kober, J Andrew Bagnell, and Jan Peters. 2013. Reinforcement learning in robotics: A survey. *The International Journal of Robotics Research* 32, 11 (2013), 1238–1274.

[16] Yannis A Korilis, Aurel A Lazar, and Ariel Orda. 1999. Avoiding the Braess paradox in non-cooperative networks. *Journal of Applied Probability* 36, 1 (1999), 211–222.

[17] John Langford and Tong Zhang. 2007. The Epoch-Greedy Algorithm for Multi-armed Bandits with Side Information. In *Advances in Neural Information Processing Systems*, J. Platt, D. Koller, Y. Singer, and S. Roweis (Eds.), Vol. 20. Curran Associates, Inc.

[18] Tor Lattimore and Csaba Szepesvári. 2020. *Bandit algorithms*. Cambridge University Press.

[19] Jinghuan Ma, Jun Deng, Lingyang Song, and Zhu Han. 2014. Incentive mechanism for demand side management in smart grid using auction. *IEEE Transactions on Smart Grid* 5, 3 (2014), 1379–1388.

[20] Eric Mazumdar, Lillian J Ratliff, and S Shankar Sastry. 2020. On gradient-based learning in continuous games. *SIAM Journal on Mathematics of Data Science* 2, 1 (2020), 103–131.

[21] Wenjun Mei, Shadi Mohagheghi, Sandro Zampieri, and Francesco Bullo. 2017. On the Dynamics of Deterministic Epidemic Propagation over Networks. https://doi.org/10.48550/ARXIV.1701.03137

[22] David Mguni, Joel Jennings, Sergio Valcarcel Macua, Emilio Sison, Sofia Ceppi, and Enrique Munoz De Cote. 2019. Coordinating the crowd: Inducing desirable equilibria in non-cooperative systems. *arXiv preprint arXiv:1901.10923* (2019).

[23] Francesca Parise and Asuman E Ozdaglar. 2020. Analysis and Interventions in Large Network Games. *Available at SSRN 3692826* (2020).

[24] Vianney Perchet and Philippe Rigollet. 2013. The multi-armed bandit problem with covariates. *The Annals of Statistics* 41, 2 (2013), 693–721.

[25] Jan Peters and Stefan Schaal. 2008. Reinforcement learning of motor skills with policy gradients. *Neural networks* 21, 4 (2008), 682–697.

[26] Murray H. Protter and Charles B. Morrey. 1991. *Contraction Mappings, Newton's Method, and Differential Equations.* Springer New York, New York, NY, 329–340.

[27] Lillian J Ratliff and Tanner Fiez. 2020. Adaptive incentive design. *IEEE Trans. Automat. Control* 66, 8 (2020), 3871–3878.

[28] J. B. Rosen. 1965. Existence and Uniqueness of Equilibrium Points for Concave N-Person Games. *Econometrica* 33, 3 (1965), 520–534.

[29] Julian Schrittwieser, Ioannis Antonoglou, Thomas Hubert, Karen Simonyan, Laurent Sifre, Simon Schmitt, Arthur Guez, Edward Lockhart, Demis Hassabis, Thore Graepel, et al. 2020. Mastering atari, go, chess and shogi by planning with a learned model. *Nature* 588, 7839 (2020), 604–609.

[30] David Silver, Julian Schrittwieser, Karen Simonyan, Ioannis Antonoglou, Aja Huang, Arthur Guez, Thomas Hubert, Lucas Baker, Matthew Lai, Adrian Bolton, et al. 2017. Mastering the game of go without human knowledge. *nature* 550, 7676 (2017), 354–359.

[31] Richard S Sutton and Andrew G Barto. 2018. *Reinforcement learning: An introduction*. MIT press.

[32] J.N. Tsitsiklis and B. Van Roy. 1997. An analysis of temporal-difference learning with function approximation. *IEEE Trans. Automat. Control* 42, 5 (1997), 674–690. https://doi.org/10.1109/9.580874

[33] Christopher JCH Watkins and Peter Dayan. 1992. Q-learning. *Machine learning* 8, 3 (1992), 279–292.

[34] P. Whittle. 1981. Arm-Acquiring Bandits. *The Annals of Probability* 9, 2 (1981), 284–292.

[35] P. Whittle. 1988. Restless bandits: activity allocation in a changing world. *Journal of Applied Probability* 25, A (1988), 287–298. https://doi.org/10.2307/3214163