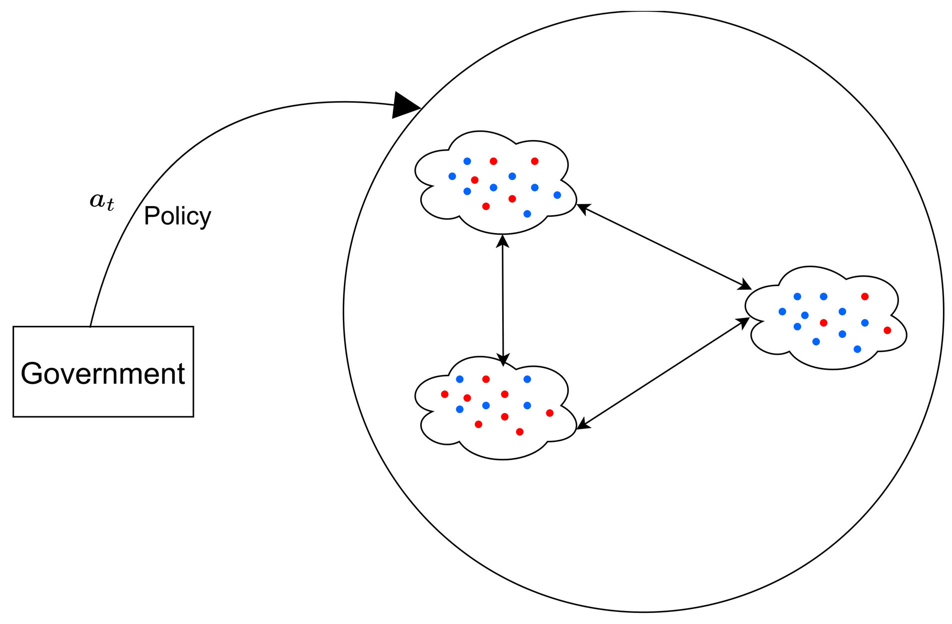
Equilibrium Bandits: Learning Optimal Equilibria of Unknown Dynamics

Siddharth Chandak, Ilai Bistritz, Nicholas Bambos

`chandaks, bistritz, bambos@stanford.edu`

Epidemic Control



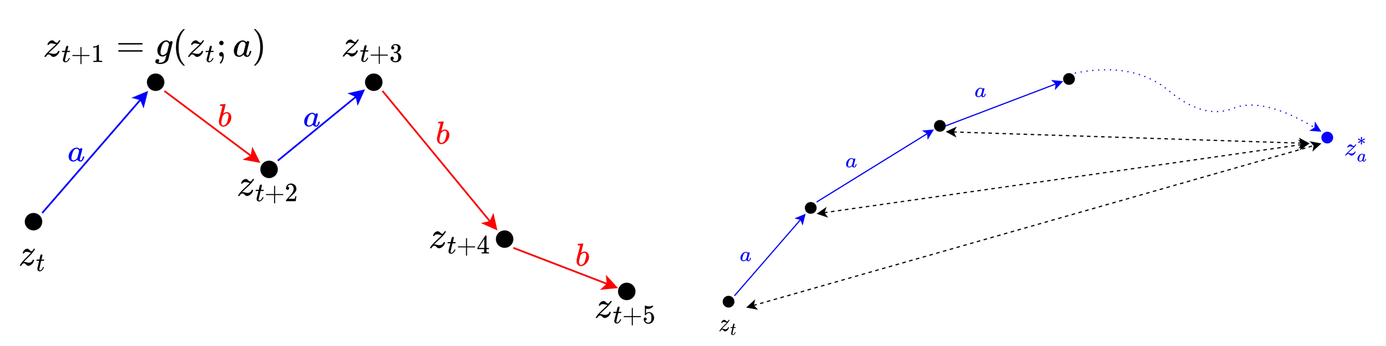
- How should the government control a new epidemic?
- Hard to model the epidemic and population interaction
- Multiple policies:
  - → e.g., lockdown, mask enforcement, advertising for awareness
  - → Each has their own operational cost
- Care about the equilibrium infection rate of each policy:
  - → Need to enact it consecutively for a "large number of time-steps"

Equilibrium Bandits: System Evolution

- Agent takes action $a_t \in \{1, \dots, K\}$ at each time $t = 0, 1, 2, \dots$
- $\vec{z_t}$: System State
  - → Evolution Function: $z_{t+1} = g(z_t; a_t)$
  - → Each action a has their equilibrium point z_a^*
  - → Converges if action is fixed, i.e., $\lim_{t\uparrow\infty} g^{(t)}(z; a) = z_a^*$
  - → Distance from equilibrium decreases when action a is played, i.e.,

$$||g(z, a) - z_a^*|| \le \exp\left(-\frac{1}{\tau_c}\right) ||z - z_a^*||$$

  - → τ_c: *approximate convergence time to equilibrium*



Equilibrium Bandits: Rewards & Regret

- $f(z_t; a_t)$: Reward Function
- Agent receives noisy rewards
- Optimal action a^*: action with maximum reward at equilibrium

$$a^* = \arg\max_a f(z_a^*, a)$$

- Regret:

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

  - → Difference w.r.t. what the optimal action achieves at equilibrium

UECB Algorithm: Key Steps

- **Convergence Bound:** To get a bound on how well an action can perform at equilibrium
  - → Suppose action a is player consecutively ℓ times (from t to $t + \ell$):

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

- **Epochs of Increasing Length:** To give *promising* actions more consecutive time-steps to converge
  - → Lengths of epochs increased as an action is chosen more times
  - → If action a has been played for m epochs, then length of $(m+1)^{th}$ epoch is e^{m+1} time-steps
- **Noise Averaging:** To average-out noise while eliminating equilibrium bias
  - → If action a is played for ℓ consecutive steps in an epoch, take average of last $\ell/2$ observed rewards

Guarantees

Theorem: Regret Bound

For any instance of equilibrium bandits, the regret achieved by UECB algorithm is bounded as:

$$\mathbb{E}[R(T)] = \mathcal{O}\left(\sum_{a \neq a^*} \underbrace{\frac{\log(T)}{\Delta_a}}_{\substack{\text{Stochastic}\\\text{Bandits}}} + \underbrace{\tau_c \log\left(\tau_c \log\left(\frac{1}{\Delta_a}\right)\right) + \tau_c \log(\log(T))}_{\text{Convergence Time}}\right),$$

where Δ_a is the suboptimality gap for arm a defined w.r.t. equilibrium rewards.

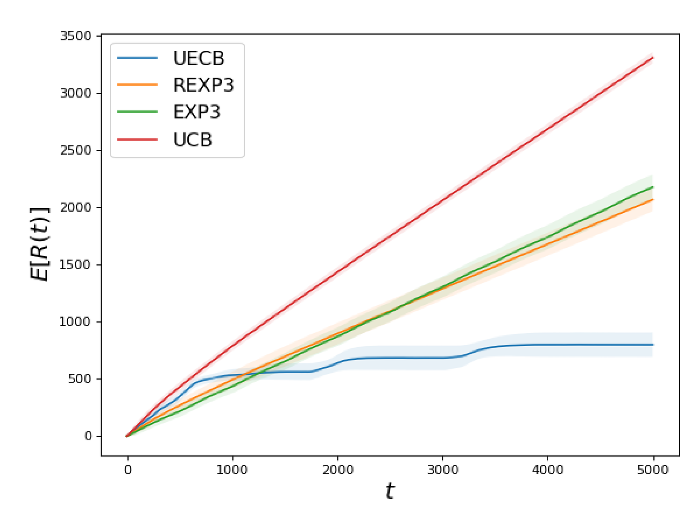
Theorem: Lower Bound

There exist instances of equilibrium bandits where for all 'good' algorithms

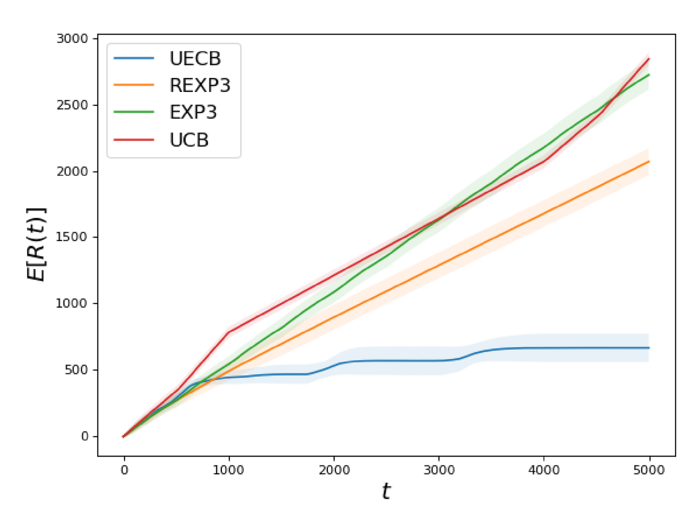
$$\mathbb{E}[R(T)] = \Omega\left(\frac{\log(T)}{\Delta_a} + \tau_c \Delta_a \log\left(\frac{1}{\Delta_a}\right)\right).$$

- UECB is optimal in T, Δ_a, and optimal upto logarithmic factors in τ_c
- Lower bound obtained using an instance where arms cannot be distinguished for the first $\sim\tau_c$ steps

Numerical Experiments



SIS Epidemic Control



Strongly Monotone Games

- Strongly Monotone Games:
  - → Game designer tries to optimize global objective by controlling game parameters
  - → Players optimize local utility using gradient ascent
  - → On fixing parameters, players eventually converge to Nash equilibrium
- UECB achieves logarithmic regret while standard algorithms such as UCB and EXP3 achieve linear regret

Upper Equilibrium Concentration Bound (UECB)

UECB Algorithm

For epoch $n = 1, 2, \dots$

(1) Play action $a_n = \arg\max_a \text{UECB}_a$ for $\ell_n = \exp(m_a + 1)$ time-steps

(2) Estimate:

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

(3) Update UECB:

$$\text{UECB}_{a,n} = \hat{x}_{a,n} + \underbrace{\frac{c_1}{\ell_n/2} \exp\left(-\frac{\ell_n}{2\tau_c}\right)}_{\text{Equilibrium Bias}} + \underbrace{\sqrt{\frac{c_2\sigma^2}{\ell_n/2} \log(2t_n^3)}}_{\text{Noise Averaging } (\sim \text{UCB})}$$

End

- Algorithm inspired by UCB
- An additional term obtained using convergence bound

Funding

- Koret Foundation grant for Smart Cities and Digital Living 2030
- 3Com Corporation Stanford Graduate Fellowship