Combinatorial Bandits with Linear Constraints: Beyond Knapsacks and Fairness

Qingsong Liu, Weihang Xu, Siwei Wang, Zhixuan Fang

IIIS, Tsinghua University

NeurIPS 2022

Motivation

- Multi-Armed Bandit (MAB):
  - A fundamental online learning model
  - Exploration-exploitation trade-off
  - Goal: maximize the accumulated reward

- Combinatorial Multi-Armed Bandit (C-MAB)
  - A more general framework
  - Base arms, super arm
  - Goal: identify the optimal super-arm which maximize the sum of rewards of its containing base arms
  - Applications: wireless scheduling, crowdsourcing

Motivation

However, in real world,

- The agent usually subjects to some operational constraints
  - Knapsacks constraints: the process terminates when the total resource budget has been used-up
    - Limited inventory in dynamic pricing
    - Network resource allocation
  - Fairness constraints: the frequency of an arm can be taken must exceed a threshold
    - Wireless scheduling with QoS guarantees
    - Fairness-aware ad recommendation or federated-learning systems
  - Hybrid or multi-type constraints
    - Information gathering in IoT systems
    - Energy dispatching in power systems

Combinatorial bandits with linear constraints

- N base arms, reward realization vector $\boldsymbol{f}(t)$, mean reward vector $\boldsymbol{\mu}$
- t-th round action/decision $\boldsymbol{a}(t) \in \{\boldsymbol{a} | \boldsymbol{a} \in \{0,1\}^N, ||\boldsymbol{a}||_1 \leq m\}$
- Feedback model: semi-bandit feedback
- (Instantaneous) reward: $R_t = \sum_{i=1}^{N} f_i(t) a_i(t)$

Combinatorial bandits with linear constraints

- N base arms, reward realization vector $\boldsymbol{f}(t)$, mean reward vector $\boldsymbol{\mu}$
- t-th round action/decision $\boldsymbol{a}(t) \in \{\boldsymbol{a} | \boldsymbol{a} \in \{0,1\}^N, ||\boldsymbol{a}||_1 \leq m\}$
- Feedback model: semi-bandit feedback
- (Instantaneous) reward: $R_t = \sum_{i=1}^{N} f_i(t) a_i(t)$
- Constraints: $\boldsymbol{g}(\boldsymbol{a}(t)) = [g_1(\boldsymbol{a}(t)), g_2(\boldsymbol{a}(t)), \ldots, g_N(\boldsymbol{a}(t))]^T$

Combinatorial bandits with linear constraints

- Goal:

$$\max \sum_{t=1}^{T} R_t \,, \qquad \text{s.t.} \sum_{t=1}^{T} \boldsymbol{g}(\boldsymbol{a}(t)) \le \mathbf{0}$$

- Performance metric: regret and constraint violations

$$\text{Regret}(T) = \text{OPT}(T) - \text{E}\left[\sum_{t=1}^{T} R_t\right], \qquad \text{Vio}(T) = \sum_{t=1}^{T} \boldsymbol{g}(\boldsymbol{a}(t))$$

- Remark
  - First to study the combinatorial multi-armed bandits with long-term constraints
  - Generalization of several prominent lines of prior work, including unconstrained bandits, bandits with fairness constraints, bandits with knapsacks (BwK), etc

Algorithm: UCB-LP

- Observation:
  - No super-arm is optimal across all rounds, but there exists an optimal sampling distribution over super-arms, i.e., optimal stationary randomized policy
  - When $\boldsymbol{\mu}$ is known, the optimal stationary randomized policy can be characterized as

$$\max_{\boldsymbol{x} \in R^N} \langle \boldsymbol{\mu}, \boldsymbol{x} \rangle, \quad \text{s.t. } \boldsymbol{g}(\boldsymbol{x}) \leq \boldsymbol{0}, \;\; \boldsymbol{0} \leq \boldsymbol{x} \leq \boldsymbol{1}, \; ||\boldsymbol{x}||_1 \leq m.$$

Algorithm: UCB-LP

- UCB-LP
  - UCB estimate computation
  - LP solving to obtain an optimistic probabilistic selection vector
  - Constructing a sampling probability distribution over super-arms

Algorithm 1 UCB-LP

Initialization: $\mathcal{A} = \{\boldsymbol{x} | \boldsymbol{x} \in \{0,1\}^N, ||\boldsymbol{x}||_1 \leq m\}$
for round $t = 1, \ldots T-1$ **do**
  Compute UCBs: $\hat{\mu}_i(t) = \min\{\overline{\mu}_i(t) + \sqrt{\frac{2 \ln t}{h_i(t)}}, 1\}, \ \forall i.$
  Solve optimization problem (4) and obtain $\boldsymbol{x}(t)$.
  Construct a distribution $\pi_t(\cdot)$ over $\mathcal{A}$ such that $\mathrm{E}_{\pi_t}[\boldsymbol{a}(t)] = \boldsymbol{x}(t)$, and sample $\boldsymbol{a}(t) \sim \pi_t$.
  Pull the arms according to the action vector $\boldsymbol{a}(t)$.
  Update the statistics: $h_i(t+1), \ \overline{\mu}_i(t+1), \ \forall i.$
end for

Performance guarantee of UCB-LP

- General case: $\boldsymbol{g}(\cdot)$ is generally linear

$$\mathrm{Regret}(T) \leq O\left(\frac{mN\log T}{\Delta_{\min}}\right), \qquad \mathrm{E}[\mathrm{Vio}(T)] \leq 0.$$

Performance guarantee of UCB-LP

- General case: $\boldsymbol{g}(\cdot)$ is generally linear

$$\mathrm{Regret}(T) \leq O\left(\frac{mN\log T}{\Delta_{\min}}\right), \qquad \mathrm{E}[\mathrm{Vio}(T)] \leq 0.$$

- Comparison with prior related works
  - Better (optimal) dependence on $\Delta_{\min}$ and N, combinatorial setting
  - Valid for all linear constraints, while theirs is only valid for specific kind of constraints, either knapsacks constraints, or fairness constraints
  - Without requiring any assumptions or knowledge of some parameters of the problem instance a prior

Performance guarantee of UCB-LP

- General case: $\boldsymbol{g}(\cdot)$ is generally linear

$$\text{Regret}(T) \leq O\left(\frac{mN\log T}{\Delta_{\min}}\right), \qquad \text{E}[\text{Vio}(T)] \leq 0.$$

- Constant (better) regret guarantee for special case: fairness constraints

$$\text{Regret}(T) \leq O\left(\frac{mN^2}{\Delta_{\min}^2}\right), \qquad \text{E}[\text{Vio}(T)] \leq 0.$$

UCB-PLLP: an efficient version of UCB-LP

Main idea

- (partial) Lagrangian tranformation

$$\max_{\boldsymbol{x}\in R^N} \langle \widehat{\boldsymbol{\mu}}(t), \boldsymbol{x}\rangle - \boldsymbol{\lambda}_{\boldsymbol{t}}^{\boldsymbol{T}} \boldsymbol{g}(\boldsymbol{x}), \quad \text{s.t.} \;\; \boldsymbol{0} \le \boldsymbol{x} \le \boldsymbol{1}, \;\; ||\boldsymbol{x}||_1 \le m.$$

$$\Updownarrow$$

$$\max_{\boldsymbol{a}} \left\langle \widehat{\boldsymbol{\mu}}(t) - \boldsymbol{\lambda}_{\boldsymbol{t}}^{\boldsymbol{T}} \nabla \boldsymbol{g}(\boldsymbol{a}(t-1)), \boldsymbol{a} \right\rangle, \quad \text{s.t.} \;\; \boldsymbol{a} \in \{0,1\}^N, \;\; ||\boldsymbol{a}||_1 \le m.$$

- Virtual queue technique incorporated with "pessimistic" mechanism

$$\boldsymbol{Q}(t) = [\boldsymbol{Q}(t-1) + \boldsymbol{g}\big(\boldsymbol{a}(t-1)\big) + \epsilon_t \cdot \boldsymbol{I}]^+$$

$$\boldsymbol{\lambda}_t = \alpha_t \boldsymbol{Q}(t)$$

UCB-PLLP: an efficient version of UCB-LP

Algorithm 2 UCB-PLLP

1: **Initialization:** $\mathcal{A} = \{\boldsymbol{x} | \boldsymbol{x} \in \{0,1\}^N, ||\boldsymbol{x}||_1 \leq m\}$
2: **for** round $t = 1, \ldots T-1$ **do**
3: Compute UCBs: $\hat{\mu}_i(t) = \min\{\overline{\mu}_i(t) + \sqrt{\frac{2\ln t}{h_i(t)}}, 1\}, \ \forall i.$
4: Update the primal iterate: $\boldsymbol{a}(t) = \arg\max_{\boldsymbol{a}\in\mathcal{A}} \left\langle \hat{\boldsymbol{\mu}}(t) - \alpha_t \sum_{k=1}^{K} \nabla g_k(\boldsymbol{a}(t-1)) Q_k(t), \boldsymbol{a} \right\rangle$
5: Play arm i and receive $f_i(t)$ if $a_i(t) = 1$.
6: Update the virtual queues:
7: $\boldsymbol{Q}(t+1) = [\boldsymbol{Q}(t) + \boldsymbol{g}(\boldsymbol{a}(t)) + \epsilon_t \boldsymbol{I}]^+.$
8: Update the statistics: $h_i(t+1), \ \overline{\mu}_i(t+1), \ \forall i.$
9: **end for**

Set $\epsilon_t = O\left(\frac{\delta}{\sqrt{t}}\right)$ and $\alpha_t = O\left(\frac{N}{\delta\sqrt{t}}\right)$, then UCB-PLLP achieves:

$$\text{Regret}(T) \leq \tilde{O}\left(m\sqrt{T}\right), \qquad \mathbb{P}[\text{Vio}(T) > \mathbf{0}] \leq O(e^{-\delta\sqrt{T}}).$$

Q & A

Thanks for Your Attention!