Poster: Multi-agent Combinatorial Bandits with Moving Arms

Zhiming Huang Bingshan Hu Jianping Pan

Department of Computer Science, University of Victoria, Victoria BC, Canada

Abstract

- We study a distributed stochastic multi-armed bandit problem that can address many real-world problems.
- We propose an efficient algorithm called *multi-agent combinatorial upper confidence bound (MACUCB)* with provable performance guarantees and low communication overhead.
- Furthermore, we perform extensive experiments to show the effectiveness of the proposed algorithm.

Introduction

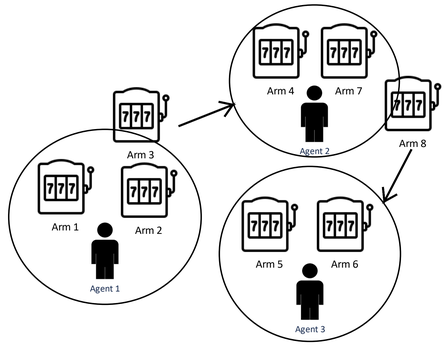


Figure 1: An example of MACMAB-M

We study an innovative distributed multi-armed bandit problem called *multi-agent combinatorial bandits with moving arms (MACMAB-M)* involving an agent set $\mathbb{K} := \{1, 2, \ldots, K\}$ and an arm set $\mathbb{N} := \{1, 2, \ldots, N\}$.

- Time is slotted into T rounds: $t = 1, 2, \ldots, T$.
- Each arm can randomly move across each agent over the time, resulting in the variance of the available arms $A_{t,k} \subseteq \mathbb{N}$ to agent $k \in \mathbb{K}$, and, apparently, we have $\bigcup_{k \in \mathbb{K}} A_{t,k} = \mathbb{N}$.
- Each agent k needs to play a combination of arms $S_{t,k} \subseteq A_{t,k}$ subject to some certain constraints, and observes the random reward of each played arm $\theta_t(n), \forall n \in S_{t,k}$ in each round t.
- For each arm n, $\theta_t(n)$ is drawn from a fixed but unknown distribution, and is *independent and identically distributed (i.i.d.)* over the time.

As the arm reward distributions are unknown a priori, an algorithm needs to balance the tradeoff between exploring the distribution of arms and exploiting the already learned knowledge to gain more rewards, which results in *regret* compared with the optimal algorithm that knew the arm distribution in advance.

Motivations: 1. Task assignment for multiple crowdsourcing platforms such as Amazon Mechanical Turk, Amazon Flex and Testlio, where workers can move across platforms. 2. Traffic scheduling problem in wireless networks with multiple access points (APs), where clients can move across APs. 3. Caching in wireless base stations, where users can move across different base stations.

The MACUCB Algorithm

The multi-agent combinatorial upper confidence bound MACUCB algorithm is an extension to UCB [1], where each agent plays a combination of available arms with the highest UCB estimate, as defined in (1).

$$U_{t,k}(n) = \hat{\theta}_{t-1}(n) + \sqrt{\frac{1.5 \ln(t-1)}{O_{t-1}(n)}}, \forall n \in A_{t,k}, \tag{1}$$

where $\hat{\theta}_t(n)$ is the empirical mean (or sample-mean) reward and $O_t(n)$ is the total number of times that arm n has been played, by the end of round t.

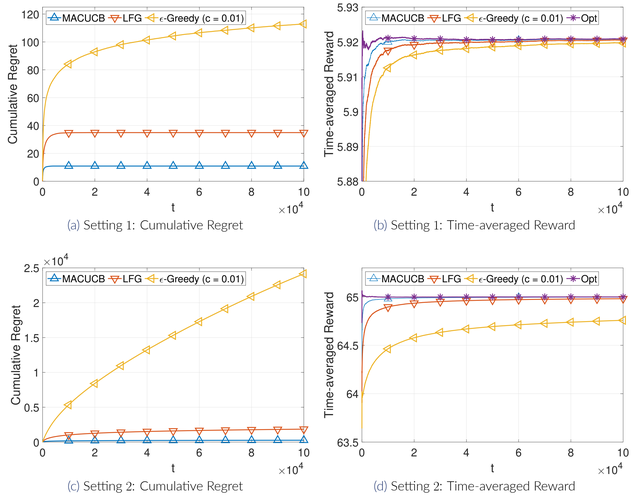
To save communication overhead, instead of communicating all past information such as the random rewards and the trajectory of any moving arm, we only transfer a tuple consisting of $\hat{\theta}_t(n)$ and $O_t(n)$. The whole process for each agent k in each round t is shown as follows:

1. Observe incoming arms and their statistic information, i.e., $\hat{\theta}_t(n)$ and $O_t(n)$.
2. Calculate UCB estimate for each arm in $A_{t,k}$.
3. Play a combination of arms with the highest UCB estimate.
4. Update statistic information for the played arms.
5. Transfer statistic information for each outgoing arm.

MACUCB has a T-round cumulative regret bounded by $\frac{96NM^{\frac{4}{3}} \log T}{\Delta_{\min}} + \frac{\pi^2}{3} NM$, and has time, space, and message complexities for each agent to be $O(N \log N)$, $O(N)$ and $O(N)$,respectively, in each round for N arms.

Experiments

We compared MACUCB with the ϵ-Greedy algorithm [1] which is an oft-used algorithm for reinforcement learning, and the LFG algorithm [2], which can be regarded as the MACUCB algorithm without communication between agents in the following two bandit settings.



(a) Setting 1: Cumulative Regret (b) Setting 1: Time-averaged Reward

(c) Setting 2: Cumulative Regret (d) Setting 2: Time-averaged Reward

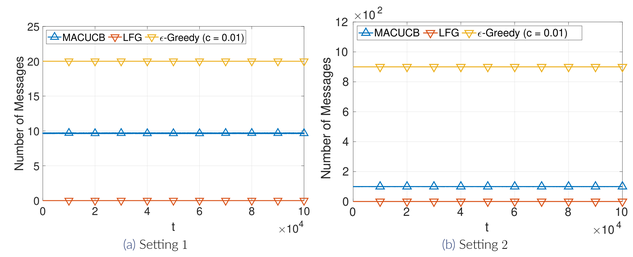
Figure 2: The evaluation results for both settings.

- **Setting 1:** The total number of agents is $K = 3$, the total number of arms is $N = 10$ and the maximum number of arms can be played in each round is $M = 3$. The rewards of arms in each round are i.i.d. drawn from Bernoulli distributions with mean rewards $0.95, 0.9, 0.85, 0.8, 0.75, 0.7, 0.65, 0.6, 0.55, 0.5$.
- **Setting 2:** We have $K = 10$, $N = 100$, and $M = 10$. The reward of each arm also follows a Bernoulli distribution with a randomly generated mean value.

Cumulative Regret and Time-averaged Reward

We can see that in both settings, the cumulative regret of MACUCB is the lowest compared to that of LFG and ϵ-Greedy ($c = 0.01$), as shown in Figs. 2a and 2c. Regarding the time-averaged reward, all considered algorithms can converge to the optimal solution (Opt), but MACUCB is the quickest for both settings, as shown in Fig. 2b and 2d. Note that in Fig. 2c, MACUCB still has a learning curve (i.e., the cumulative regret increases at the beginning) but is too small to be observed clearly when comparing with other algorithms.

Communication Overhead and Time Complexity



(a) Setting 1 (b) Setting 2

Figure 3: The communication overhead for the algorithms.

The results of communication overhead for both settings are shown in Fig. 3, where LFG has 0 messages as it does not communicate, and we can see that MACUCB is efficient in terms of the number of messages, with 10 and 100 messages per round in Settings 1 and 2, respectively, which corresponds to the analyzed communication overhead $O(N)$.

Table 1: The time consumption for each agent over T rounds

Algorithm	MACUCB	LFG	ϵ-Greedy
Setting 1	0.344 s	0.345 s	0.335 s
Setting 2	3.934 s	3.773 s	3.779 s

Although MACUCB needs to spend a bit more time to process incoming and outgoing arms, all the algorithms have the similar performance for both settings in a platform of a desktop computer with Intel Core i7-3770 and 16 GB RAM, as it can be proved that their time complexity is $O(N \log N)$ for each agent.

References

[1] P. Auer, N. Cesa-Bianchi, and P. Fischer, ``Finite-time Analysis of the Multi-armed Bandit Problem,'' *Machine Learning*, vol. 47, no. 2--3, pp. 235--256, 2002.

[2] F. Li, J. Liu, and B. Ji, ``Combinatorial Sleeping Bandits with Fairness Constraints,'' *IEEE Transactions on Network Science and Engineering (TNSE)*, vol. 7, no. 3, pp. 1799--1813, 2019.