PAC Algorithms for the Infinitely-Many Armed Problem with Multiple Pools

Yahel David YAHELD@TX.TECHNION.AC.IL
Nahum Shimkin SHIMKIN@EE.TECHNION.AC.IL
Faculty of Electrical Engineering
Technion, Haifa 32000, Israel

Editor:

Abstract

We consider a multi-pool version of the infinitely-many armed bandit problem, where a learning agent is faced with several large pools of items, and interested in finding the best item overall. At each time step the agent chooses a pool, and obtains a random item whose value is precisely revealed. The obtained values within each pool are assumed to be i.i.d., with an unknown probability distribution that generally differs among the pools. Under the PAC framework, we provide lower bounds on the sample complexity of any (ϵ, δ)-correct algorithm, and propose an algorithm that attains this bound up to logarithmic factors. We compare the performance of this multi-pool algorithm to the variant in which the pools are not distinguishable by the agent and are chosen randomly at each stage. Interestingly, when the maximal values of the pools happen to be similar, the latter approach may provide better performance.

Keywords: Multi-armed bandits, pure exploration, max k-armed bandit.

1. Introduction

We consider the problem of finding the best item from a large set of items, or arms, which are arranged in separate pools. The value distributions of the items in each pool is unknown, but may well be different across pools. A learning agent samples the pools sequentially, where at each time step it chooses some pool, obtains a random element from that pool, and observes its numerical value.

The goal of the agent is to quickly return an (ϵ, δ)-correct arm, namely an arm whose value is ϵ-close to the overall best arm with a probability larger than $1 - \delta$. Specifically, we wish to minimize the sample complexity, namely the expected number of samples observed by the learning algorithm before it terminates. Our model assumes that the value of each newly sampled arm is an independent sample from a pool-dependent probability distribution. We further assume that the probability distribution function of each pool is continuous, and has a density which is bounded from below, with a known lower bound.

The scenario considered here is relevant when a single item needs to be selected from among several clustered sets. These may be parts that come from different manufacturers or produced by different processes, employees that are refereed by different employment agencies, finding the best match to certain genetic characteristics in different populations, or choosing the best channel among different frequency bands in a cognitive radio wireless

system. We note that our model considers the pools as being initially indistinguishable, in the sense that no prior knowledge is presumed regarding their relative merit.

The model considered here is related to the so-call infinitely-many armed bandit problem, studied by Berry et al. (1997); Teytaud et al. (2007); Wang et al. (2009); Chakrabarti et al. (2009); Bonald and Proutiere (2013); David and Shimkin (2014). These works consider the case of a combined pool, focusing on online learning with the regret criterion. In most of these works the observed values are stochastic, so that repeated sampling of each observed arm is generally required to learn its value. Here, we assume that the values of the arms are fixed and precisely revealed once sampled, which enables us to focus on the choice of pool as the main decision issue.

For the classical Multi-Armed Bandit (MAB) problem, algorithms that find the best arm (in terms of its expected reward) in the PAC sense were presented by Even-Dar et al. (2002); Audibert et al. (2010); Gabillon et al. (2012). For the same problem, a lower bound on the sample complexity was presented by Mannor and Tsitsiklis (2004); Audibert et al. (2010). Our model can be viewed as analogous to this MAB model by considering, respectively, the pools as the arms, and the item values in our model as the stochastic rewards in the MAB problem. The essential difference is in the objective, which in our case is to find and retain the item (i.e., sample) with the highest value.

A version of the MAB problem which is analogue to our model (in which the objective is to maximaize the largest obtained reward) is known as the max k-armed bandit problem, introduced by Cicirello and Smith (2005). The max k-armed bandit model can be viewed as our model by considering arms as pools and rewards as item values. For distribution functions in a specific family, Streeter and Smith (2006) provided an algorithm and a corresponded upper bound on the sample complexity of order $\frac{-\ln(\delta)}{\epsilon^2}$. Carpentier and Valko (2014) considered a similar model in which the number of samples (n) is given and the goal is to minimize the difference between the expected largest sampled reward obtained by a learning agent to that obtained by sampling only the best arm. They proposed an algorithm and showed that the difference is of the order of $\frac{1}{n^\alpha}$, where $\alpha < 1$ and decreases as the distribution functions are further from a specific functional family.

In this paper, we propose an algorithm for returning an (ϵ, δ)-correct arm, and show that its sample complexity is upper bounded by $O(\frac{-\ln(\delta)-\ln(\epsilon)}{\epsilon})$. Furthermore, we present a lower bound on the sample complexity of every (ϵ, δ)-correct algorithm and show that our algorithm attains that lower bound up to a logarithmic term, when at least one pool is ϵ-close to the best pool. Our result, improves on the bound of Streeter and Smith (2006) by a factor of ϵ^{-1}. To compare with the work of Carpentier and Valko (2014), we note that for $\delta = n^{-2}$, the difference between the expected largest sampled value obtained by our algorithm and the best possible value is of order of $\frac{\ln(n)}{n}$.

The paper proceeds as follows. In the next section we present our model. In Section 3 we provide a lower bound on the sample complexity of every (ϵ, δ)-correct algorithm. In Section 4 we present an (ϵ, δ)-correct algorithm. In Section 5, we consider for comparison the combined-pool variant of the problem, where the agent does not distinguish between pools and samples from them uniformly at random. We close the paper with some concluding remarks. Certain proofs are differed to the Appendix due to space limitations.

2. Model Definition

We consider a set of pools, denoted K. When the learning agent chooses $k \in K$, an arm from that pool is obtained and its value (a real number in the interval $[0,1]$) is revealed. The values of arms in pool $k \in K$ are independent and identically distributed with a distribution function (CDF) $F_k(\mu)$, and corresponding density function (pdf) $f_k(\mu)$. Let $\text{supp}(f_k)$ denote the support of f_k, and let $\mu_k^* = \sup\{\mu | \mu \in \text{supp}(f_k)\}$ denote its maximal value. The largest value among all of the pools is denoted by $\mu_*^* = \max_{k \in K} \mu_k^*$.

Throughout the paper, we shall make the following assumption.

Assumption 1 *For each $k \in K$, $\text{supp}(f_k)$ is a single interval in $[0,1]$, and $f_k(\mu) \geq a$ for $\mu \in \text{supp}(f_k)$, for some known constant $a > 0$.*

The lower bound on the the density f_k ensures that the maximal value of each pool can be approximated by observing a finite number of samples. We note that more refined bounds, in the style of the tail function bounds in the papers of Wang et al. (2009) or David and Shimkin (2014), can be used instead, provided that such prior knowledge exists.

An algorithm for this model selects a pool to sample from at each time step, based on the observed history so far (i.e., the previously selected pools and observed values). We shall require an algorithm to terminate after a random number T of samples, which is finite with probability 1, and return a value V which is the maximal value of all the arms observed by T. An algorithm is said to be (ϵ, δ)-*correct* if

$$P\left(V > \mu_*^* - \epsilon\right) > 1 - \delta\,.$$

The expected running time $E[T]$ of the algorithm is called *sample complexity*, which we wish to minimize.

3. Lower Bound

The following result sets a lower bound on the sample complexity of any (ϵ, δ)-correct algorithm

Theorem 1 *Suppose $a \leq \frac{1}{2}$, $\mu_*^* \leq 1 - \epsilon_0$, and let $\epsilon \in (0, \epsilon_0)$, $\delta \in (0, \frac{1}{8})$, where $\epsilon_0 < 1$. Let k^* stands for an optimal pool, namely, $\mu_{k^*}^* = \mu_*^*$. Then, for every (ϵ, δ)-correct algorithm, it holds that*

$$E[T] \geq \sum_{k \in K \setminus \{k^*\}} \frac{1}{8a\left(\epsilon + \mu_*^* - \mu_k^*\right)} \ln\left(\frac{3}{16\delta}\right). \tag{1}$$

The proof is provided in Appendix A and proceeds by showing that if an algorithm is (ϵ, δ)-correct and its sample complexity is lower than a certain threshold for some set of value distributions, then this algorithm cannot be (ϵ, δ)-correct for some related value distributions.

4. Algorithm

Here we provide an (ϵ, δ)-correct algorithm. The algorithm starts by sampling once from each pool. Then, it repeatedly calculates an index for each pool which can be interpreted

Algorithm 1 Multi Pool Algorithm

1: **Input:** Constants $\delta > 0$, $\epsilon > 0$ and $L = 6\ln\left(|K|\left(1+\frac{-\ln(\delta)}{a\epsilon}\right)\right)$.
2: **Initialization:** Counters $C(i) = 1\,\forall i \in K$.
3: Sample one arm from every pool.
4: Compute $Y^i_{C(i)} = V^i_{C(i)} + \epsilon^{UB}(C(i))$
where $V^i_{C(i)}$ is the largest value observed from pool i and $\epsilon^{UB}(C(i)) = \frac{L-\ln(\delta)}{aC(i)}$,
and set $i^* \in \arg\max_{i\in K} Y^i_{C(i)}$.
5: If $\epsilon^{UB}(C(i^*)) < \epsilon$, return the best sampled arm.
Else, sample one arm from pool i^*, set $C(i^*) = C(i^*) + 1$ and return to step 4.

as a certain upper bound on the maximal value of this pool, and samples one arm from the pool with the largest index. The algorithm terminates when the number of samples from the pool with the largest index is above a certain threshold. This idea is similar to that in the UCB1 Algorithm provided by Auer et al. (2002).

Theorem 2 *Under Assumption 1, for $L \geq 10$, Algorithm 1 is (ϵ,δ)-correct with a sample complexity of*

$$E[T] \leq \sum_{k\in K} \frac{L-\ln(\delta)}{a\max\left(\epsilon, \mu^*_* - \mu^*_k\right)} + |K| + 1,$$

where $L = 6\ln\left(|K|\left(1+\frac{-\ln(\delta)}{a\epsilon}\right)\right)$ as defined in the algorithm.

Note that L is logarithmic in $|K|$, that $\frac{2}{\epsilon+\mu^*_*-\mu^*_k} \geq \frac{1}{\max\left(\epsilon,\mu^*_*-\mu^*_k\right)}$ and that $\sum_{k\in K\setminus\{k^*\}} \frac{1}{\epsilon+\mu^*_*-\mu^*_k} \geq \sum_{k\in K} \frac{1}{\epsilon+\mu^*_*-\mu^*_k} - \frac{1}{\epsilon}$. Hence, for cases in which there are more than one ϵ-optimal pool (pools for which $\mu^*_*-\mu^*_k \leq \epsilon$), the upper bound on the sample complexity is of the same order as the lower bound in Theorem 1, up to a logarithmic factor in $|K|$.

To establish Theorem 2, we first bound the probability of the event under which the upper bound of the best pool is below the maximal value. Then, we bound the largest number of samples after which the algorithm terminates under the assumption that the upper bound of the best pool is above the maximal value.

Proof First we denote the time step of the algorithm by t, and the value of the counter $C(i)$ at time step t by $C^t(i)$. Recall that T stands for the random final time step. By the condition in step 5 of the algorithm, for every pool $k \in K$, it follows that,

$$C^T(k) \leq \lfloor \frac{L-\ln(\delta)}{a\epsilon} \rfloor + 1. \tag{2}$$

Note that by the fact that for $x \geq 6$ it follows that $\frac{d6\ln(x)}{dx} \leq 1$, and by the fact that for $x_0 = \exp\left(1\frac{2}{3}\right)$ it follows that $x_0 > 6\ln(x_0) = 10$ it is obtained that

$$L' \triangleq |K|\left(\frac{-\ln(\delta)}{a\epsilon}+1\right) > 6\ln\left(|K|\left(\frac{-\ln(\delta)}{a\epsilon}+1\right)\right) = L,$$

for $L \geq 10$. So, by the fact that $T = \sum_{i\in K} C^T(i)$, for $L \geq 10$ it follows that

$$T \leq |K|\left(\frac{L-\ln(\delta)}{a\epsilon}+1\right) < |K|\left(\frac{L'-\ln(\delta)}{a\epsilon}+1\right) \leq L'^2 = e^{\frac{L}{3}}. \tag{3}$$

Now, we begin with proving the (ϵ, δ)-correctness property of the algorithm. Recall that for every pool $k \in K$ the values are distributed according to the c.d.f. $F_k(\mu)$. Let assume w.l.o.g. that $\mu_1^* = \mu_*^*$. Then, for $N > 0$ and by the fact that $(1-\epsilon)^{\frac{1}{\epsilon}} \leq e^{-1}$ for every $\epsilon \in (0, 1]$, for $\epsilon^{UB}(N) = \frac{L - \ln(\delta)}{Na}$ it follows that

$$P\left(V_N^1 < \mu_*^* - \epsilon^{UB}(N)\right) = \left(F\left(\mu_*^* - \epsilon^{UB}(N)\right)\right)^N \leq \left(1 - a\epsilon^{UB}(N)\right)^N \leq \delta e^{-L}, \tag{4}$$

where V_N^k is the largest value observed from pool $k \in K$ after this pool has been sampled for N times. Hence, at every time step t, by the definition of $Y_{C^t(1)}^1$ and Equations (3) and (4), by applying the union bound, it follows that

$$P\left(Y_{C^t(1)}^1 < \mu_*^*\right) \leq P\left(V_{C^t(1)}^1 < \mu_*^* - \epsilon^{UB}(C^t(1))\right) \leq \sum_{t=1}^{\exp\left(\frac{L}{3}\right)} P\left(V_N^1 < \mu_*^* - \epsilon^{UB}(N)\right) \leq \delta e^{-\frac{2L}{3}}. \tag{5}$$

Since by the condition in step 5, it is obtained that when the algorithm stops

$$V_{C^t(i^*)}^{i^*} > Y_{C^t(i^*)}^{i^*} - \epsilon,$$

and by the fact that for every time step $Y_{C^t(i^*)}^{i^*} \geq Y_{C^t(1)}^1$, it follows by Equation (5) that

$$P\left(V_{C^t(i^*)}^{i^*} \leq \mu_*^* - \epsilon\right) \leq P\left(Y_{C^t(1)}^1 < \mu_*^*\right) \leq \delta e^{-\frac{2L}{3}}.$$

Therefore, it follows that the algorithm returns an arm greater than $\mu_*^* - \epsilon$ with a probability larger than $1 - \delta$. So, it is (ϵ, δ)-correct.

For proving the bound on the expected sample complexity of the algorithm we define the following sets:

$$M(\epsilon) = \{l \in K | \mu_*^* - \mu_l^* < \epsilon\}, \quad N(\epsilon) = \{l \in K | \mu_*^* - \mu_l^* \geq \epsilon\}.$$

As before, we assume w.l.o.g. that $\mu_1^* = \mu_*^*$. For the case in which

$$E_1 \triangleq \bigcap_{1 \leq t < T} \left\{Y_{C^t(1)}^1 \geq \mu_*^*\right\},$$

occurs, since $V_{C^t(k)}^k \leq \mu_k^*$ for every $k \in K$, and every time step, it follows that the necessary condition for sampling from pool k, $Y_{C^k(1)}^k \geq Y_{C^t(1)}^1$, occurs only when the event

$$E_2(t) \triangleq \left\{\mu_k^* + \epsilon^{UB}\left(C^t(k)\right) \geq \mu_*^*\right\},$$

occurs. But

$$E_2(t) \subseteq \left\{C^t(i) \leq \frac{L - \ln(\delta)}{a\left(\mu_*^* - \mu_k^*\right)}\right\}.$$

Therefore, it is obtained that

$$C^T(i) \leq \lfloor \frac{L - \ln(\delta)}{a\left(\mu_*^* - \mu_k^*\right)} \rfloor + 1. \tag{6}$$

By using the bound in Equation (2) for the pools in the set $M(\epsilon)$, the bound in Equation (6) for the pools in the set $N(\epsilon)$ and the bound in Equation (3), it is obtained that

$$E[T] \leq (1 - P(E_1))\, e^{\frac{L}{3}} + P(E_1)\, \Phi(\epsilon), \tag{7}$$

where

$$\Phi(\epsilon) \triangleq \left(\sum_{k \in N(\epsilon)} \left(\lfloor \frac{L - \ln(\delta)}{a\left(\mu_*^* - \mu_k^*\right)} \rfloor + 1 \right) + \sum_{k \in M(\epsilon)} \left(\lfloor \frac{L - \ln(\delta)}{a\epsilon} \rfloor + 1 \right) \right).$$

In addition, by Equation (5), the bound in Equation (3) and by applying the union bound, it follows that

$$P(E_1) \geq 1 - \sum_{t=1}^{T} P\left(Y_{C^t(1)}^1 < \mu_*^* \right) \geq 1 - \delta e^{-\frac{2L}{3}} e^{\frac{L}{3}} = 1 - \delta e^{-\frac{L}{3}}.$$

So,

$$1 - P(E_1) \leq \delta e^{-\frac{L}{3}}. \tag{8}$$

Furthermore, by the definitions of the sets $N(\epsilon)$ and $M(\epsilon)$, it can be obtained that

$$\Phi(\epsilon) \leq \sum_{k \in K} \lfloor \frac{L - \ln(\delta)}{a \max\left(\epsilon, \mu_*^* - \mu_k^*\right)} \rfloor + 1. \tag{9}$$

Therefore, by Equation (7), (8) and (9) the bound on the sample complexity is obtained. ■

5. Comparison with the Combined Pool Model

In this section, we examine the improvement in the sample complexity obtained by utilizing the multi pool property (the ability to choose from which pool to sample at each time step) compared to a model in which all the pools are unified into a combined pool, so that the sample is effectively obtained from a random pool. In the combined pool model, when the agent samples from this combined pool, a certain pool (among the multi pool) is chosen uniformly and an arm is sampled from this pool. We denote the pdf and the CDF of the combined pool as $f(\mu)$ and $F(\mu)$ respectively, with $f = \frac{1}{|K|} \sum_{k \in K} f_k$. By our Assumption 1 $f(\mu) \geq \frac{a}{|K|}$, and the corresponding maximal value is μ_*^*.

We next provide a lower bound on the sample complexity and an (ϵ, δ)-correct algorithm that attains the same order of this bound for the combined pool model. (Note that the lower bound in Theorem 1 is meaningless for $|K| = 1$.) Then, we discuss which approach (multi pool or combined pool) is better for different model parameters, and provide examples that illustrate these cases.

5.1 Lower Bound

The following Theorem provides a lower bound on the sample complexity for the combined pool model.

Algorithm 2 Combined Pool Algorithm

1: **Input:** Constants $\delta > 0$, $\epsilon > 0$.
2: Sample $\lceil \frac{-\ln(\delta)|K|}{a\epsilon} \rceil + 1$ arms from the pool.
3: Return the best sampled arm.

Theorem 3 (combined pool) *Suppose $a \leq \frac{1}{2}$, $\mu_*^* \leq 1 - \epsilon_0$, and let $\epsilon \in (0, \epsilon_0)$, $\delta \in (0, \frac{2}{5})$, where $\epsilon_0 < 1$. For every (ϵ, δ)-correct algorithm, it holds that*

$$E[T] \geq \frac{|K|}{8a\epsilon} \ln\left(\frac{3}{5\delta}\right). \tag{10}$$

The proof is provided in Appendix B and is based on a similar idea to that of Theorem 1.

5.2 Algorithm

In Algorithm 2 a certain number of arms is sampled, and the algorithm chooses the best one among them. In the following Theorem we provide a bound on the sample complexity achieved by Algorithm 2.

Theorem 4 *Under Assumption 1, Algorithm 2 is (ϵ, δ)-correct, with a sample complexity bound of*

$$E[T] \leq \frac{|K| \ln(\delta^{-1})}{a\epsilon} + 2.$$

The proof is provided in Appendix C. Note that the upper bound on the sample complexity is of the same order as the lower bound in Theorem 3.

5.3 Comparison and Examples

To find when the multi-pool algorithm is helpful, we can compare the *upper* bound on the sample complexity provided in Theorem 2 for Algorithm 1 (multi-pool) with the *lower* bound for the combined pool model in Theorem 3. If the upper bound for the multi-pool case is smaller than the lower bound for the combined model, then the former provably performs better; otherwise, the result may be indicative but not conclusive.

We discuss two cases, corresponding to different parameter configurations, where in the first the multi-pool approach provably provides better performance, while in the second case the combined pool approach is preferable (in terms of upper bounds).

Case 1: Suppose first that pool 1 is best: $\mu_1^* = \mu_*^*$, while all the other pools fall short significantly compared to the required accuracy ϵ: $\mu_k^* \ll \mu_*^* - \epsilon$ for $k \neq 1$. In this case $\frac{1}{\epsilon} \gg \frac{1}{\max(\epsilon, \mu_*^* - \mu_k^*)}$ for $k \neq 1$. Hence, the upper bound on sample complexity of Algorithm 1 (multi-pool) will be smaller than the lower bound for the combined pool model in Theorem 3. We now provide a numerical example which illustrates this case.

Example 1 (Case 1) *Let $|K| = 10^4$, $\mu_1^* = 0.9$, $\mu_k^* = 0.1$ $\forall k \neq 1$, and $a = 0.01$. Then for $\epsilon = 10^{-4}$ and $\delta = 10^{-3}$, the sample complexity bound for Algorithm 1 is 3.52×10^8, while the lower bound for the combined pool is 7.99×10^9 (and the upper bound for Algorithm 2 is 6.9×10^{10}).*

Case 2: Consider next the opposite case, where there are many optimal arms and few that are worse: say $\mu_1^* \ll \mu_*^* - \epsilon$, while $\mu_k^* = \mu_*^*$ for all $k \neq 1$. In this case $\frac{1}{\epsilon} = \frac{1}{\max\left(\epsilon, \mu_*^* - \mu_k^*\right)}$ for $k \neq 1$. Since there is a logarithmic-in-$|K|$ multiplicative factor in the upper bound on the sample complexity of Algorithm 1 (multi-pool), this bound will be larger than the lower bound for the combined pool model in Theorem 3. The following numerical example illustrates case 2.

Example 2 (Case 2) *Let $|K|$, a, δ and ϵ remain the same as in Example 1, and let $\mu_1^* = 0.1$ and $\mu_k^* = 0.9$ $\forall k \neq 1$. Then, the sample complexity of Algorithm 1 is 1.56×10^{12}. This is larger than the sample complexity of Algorithm 2, which is 6.9×10^{10}.*

The above observations support the following intuitive understanding. When there are few good (close to optimal) pools, the multi-pool approach allows to focus on these pools and reduces wasted sampling from the inferior arms. This, however, incurs some overhead in the proposed Algorithm 1, in order to ensure that neglected arms are indeed inferior. Hence, when there are many good arms, the combined-pool algorithm might exhibit better performance.

6. Conclusion

This paper introduces a multi-pool version of the infinitely-many armed bandit model, where the best arm is sought under the PAC criterion. We have developed corresponding lower and upper bounds on the sample complexity, which are essentially the same order up to logarithmic terms in the number of pools.

These results were compared to the combined pool model, where the learning algorithm effectively combines the different pools into one. While the multi-pool algorithm usually performs better, in some cases, in particular when most pools are optimal, the combined pool algorithm may provide better performance. It still remains to be shown whether an algorithm that provides the performance benefits of both approaches may be devised.

Another direction for future work concerns the relaxation or generalization of our Assumption 1, which requires a known lower bound on the density functions of the values over their support.

References

Jean-Yves Audibert, Sébastien Bubeck, and Rémi Munos. Best arm identification in multi-armed bandits. In *Conference on Learning Theory*, pages 41–53, 2010.

Peter Auer, Nicol Cesa-Bianchi, and Paul Fischer. Finite-time analysis of the multiarmed bandit problem. *Machine Learning*, 47:235–256, 2002.

Donald A Berry, Robert W Chen, Alan Zame, David C Heath, and Larry A Shepp. Bandit problems with infinitely many arms. *The Annals of Statistics*, pages 2103–2116, 1997.

Thomas Bonald and Alexandre Proutiere. Two-target algorithms for infinite-armed bandits with Bernoulli rewards. In *Advances in Neural Information Processing Systems*, pages 2184–2192. 2013.

Alexandra Carpentier and Michal Valko. Extreme bandits. In *Advances in Neural Information Processing Systems*, pages 1089–1097. 2014.

Deepayan Chakrabarti, Ravi Kumar, Filip Radlinski, and Eli Upfal. Mortal multi-armed bandits. In *Advances in Neural Information Processing Systems*, pages 273–280. 2009.

Vincent A. Cicirello and Stephen F. Smith. The max K-armed bandit: A new model of exploration applied to search heuristic selection. In *Proceedings of the National Conference on Artificial Intelligence*, pages 1355–1361, 2005.

Yahel David and Nahum Shimkin. Infinitely many-armed bandits with unknown value distribution. In *Machine Learning and Knowledge Discovery in Databases*, pages 307–322. 2014.

Eyal Even-Dar, Shie Mannor, and Yishay Mansour. PAC bounds for multi-armed bandit and markov decision processes. In *Conference on Learning Theory*, pages 255–270, 2002.

Victor Gabillon, Mohammad Ghavamzadeh, and Alessandro Lazaric. Best arm identification: A unified approach to fixed budget and fixed confidence. In *Advances in Neural Information Processing Systems*, pages 3212–3220. 2012.

Shie Mannor and John N. Tsitsiklis. The sample complexity of exploration in the multi-armed bandit problem. *Journal of Machine Learning Research*, 5:623–648, 2004.

Matthew J. Streeter and Stephen F. Smith. An asymptotically optimal algorithm for the max k-armed bandit problem. In *Proceedings of the National Conference on Artificial Intelligence*, pages 135–142, 2006.

Olivier Teytaud, Sylvain Gelly, and Michèle Sebag. Anytime many-armed bandits. In *CAP*, Grenoble, France, 2007.

Yizao Wang, Jean-Yves Audibert, and Rémi Munos. Algorithms for infinitely many-armed bandits. In *Advances in Neural Information Processing Systems*, pages 1729–1736, 2009.

7. Appendix A

Proof [Theorem 1] First, define the following set of hypotheses $\{H_0, H_1, \ldots, H_{|K|}\}$:

$$H_0: \quad f_k^{H_0}(\mu) = f_k(\mu) \quad \forall k \in K,$$

and, for every $k = 1, \ldots, |K|$

$$\begin{aligned} H_k: \quad & f_k^{H_k}(\mu) = \max\left(\gamma_k f_k(\mu), a\right) \mathbf{1}_{\mathrm{supp}(f_k)}(\mu) + a\mathbf{1}_{\left[\mu_k^*, \mu_*^* + \epsilon\right]}(\mu), \\ & f_l^{H_k}(\mu) = f_l(\mu), \quad l \neq k, \end{aligned}$$

where $\mathbf{1}_A$ stand for the indicator function of the set A, and γ_k is chosen such that $\int_0^1 f_k^{H_k}(\mu)d\mu = 1$ (hence $0 < \gamma_k < 1$). To further bound γ_k, note that

$$f_k^{H_k}(\mu) \leq \gamma_k f_k(\mu) + a(1-\gamma_k)\mathbf{1}_{\mathrm{supp}(f_k)}(\mu) + a\mathbf{1}_{\left[\mu_k^*, \mu_*^* + \epsilon\right]}(\mu)\,,$$

so that

$$1 = \int_0^1 f_k^{H_k}(\mu)d\mu \leq \gamma_k + a(1-\gamma_k)D_k + a\left(\mu_*^* + \epsilon - \mu_k^*\right),$$

where $D_k = \int_0^1 \mathbf{1}_{\text{supp}(f_k)}(\mu)d\mu \leq 1$ is the size of the support of f_k. Therefore,

$$1 - \frac{a\left(\mu_*^* + \epsilon - \mu_k^*\right)}{1 - aD_k} \leq \gamma_k \leq 1.$$

If hypothesis H_k ($k \neq 0$) is true, then $\mu_k^* \geq \mu_l^* + \epsilon$ for all $l \neq k$, hence the algorithm should provide a value from pool k with probability larger than $1-\delta$. We use E_k and P_k to denote the expectation and probability, respectively, under the algorithm being considered and hypothesis H_k. Further, for every $k \in K$ let

$$t_k = \frac{1}{8a\left(\epsilon + \mu_*^* - \mu_k^*\right)} \ln\left(\frac{3}{16\delta}\right),$$

and let T_k stands for the number of samples from pool k.

Suppose now that our algorithm is (ϵ, δ)-correct under H_0, and that $E_0[T_k] \leq t_k$ for some $k \in K$. We will show that this algorithm cannot be (ϵ, δ)-correct under hypothesis H_k. Therefore, an (ϵ, δ)-correct algorithm must have $E_0[T_k] > t_k$ for all $k \in K$.

Define the following events:

- $A_k = \{T_k \leq 4t_k\}$. It easily follows from $4t_k\left(1 - P_0(A_k)\right) \leq E_0[T_k]$ that if $E_0[T_k] \leq t_k$, then $P_0(A_k) \geq \frac{3}{4}$.
- Let B_k stand for the event under which the returned sample at termination is from pool k, and B_k^C for its complement. Since $P_0\left(B_{k'}\right) > \frac{1}{2}$ can hold for one pool at most, it follows that $P_0\left(B_k^C\right) > \frac{1}{2}$ for every $k \in K \setminus \{k'\}$ for some k'.
- Let C_k to be the event under which all the samples obtained from pool k are on the interval $[0, \mu_k^*]$. Clearly, $P_0(C_k) = 1$.

Define now the intersection event $S_k = A_k \cap B_k^C \cap C_k$. We have just shown that for every $k \in K \setminus \{k'\}$ it holds that $P_0(A_k) \geq \frac{3}{4}$, $P_0(B_k^C) > \frac{1}{2}$ and $P_0(C_k) = 1$, from which it follows that $P_0\left(S_k\right) > \frac{1}{4}$ for $k \neq k'$. Further, observe that for every history h_N of N samples for which the event C_k holds, it holds that $\frac{dP_k}{dP_0}(h_N) \geq (\gamma_k)^N$. We therefore obtain the following inequalities,

$$\begin{aligned} P_k\left(B_k^C\right) \geq P_k\left(S_k\right) = E_0\left[\frac{dP_k}{dP_0} I\left(S_k\right)\right] &\geq \gamma_k^{-4t_k} P_0\left(I\left(S_k\right)\right) \\ > \frac{1}{4}\gamma_l^{-4t_k} \geq \frac{3}{16} e^{-\frac{1}{2(1-a)}\ln\frac{3}{16\delta}} &\geq \delta, \end{aligned}$$

where in the last inequality we used the facts that $(1-\epsilon)^{\frac{1}{\epsilon}} \geq e^{-1}$ and that $D_k \leq 1$.

We found that if an algorithm is (ϵ, δ)-correct under hypothesis H_0 and $E_0[T_k] \leq t_k$ for some $k \neq k'$, then, under hypothesis H_k this algorithm returns a sample from a pool *other* than k with probability of δ or more, hence the algorithm is not (ϵ, δ)-correct. Therefore, any (ϵ, δ)-correct algorithm must satisfy $E_0[T_k] > t_k$ for all of pools except possibly for one

(namely, for the one k' for which $P_0\left(B_{k'}^C\right) \leq \frac{1}{2}$). Hence, since $t_{k'} \leq t_{k^*}$ (where k^* is such a pool for which $\mu_{k^*}^* = \mu_*^*$) the lower bound is obtained. ■

8. Appendix B

Proof [Theorem 3] First , we define the following hypotheses:

$$H_0: \quad f^{H_0}(\mu) = f(\mu),$$

and

$$H_1: \quad f^{H_1}(\mu) = \max\left(\gamma f(\mu), a\right) \mathbf{1}_{\mathrm{supp}(f)}(\mu) + \frac{a}{|K|}\mathbf{1}_{[\mu_*^*, \mu_*^*+\epsilon]}(\mu),$$

where, as in the proof of Theorem 1, $\mathbf{1}_A$ stand for the indicator function of the set A, and γ is chosen such that $\int_0^1 f^{H_1}(\mu)d\mu = 1$ (hence $0 < \gamma < 1$). To further bound γ, note that

$$f^{H_1}(\mu) \leq \gamma f(\mu) + a(1-\gamma)\mathbf{1}_{\mathrm{supp}(f)}(\mu) + \frac{a}{|K|}\mathbf{1}_{[\mu_*^*, \mu_*^*+\epsilon]}(\mu)\,,$$

so that

$$1 = \int_0^1 f^{H_1}(\mu)d\mu \leq \gamma + a(1-\gamma)D + \frac{a\epsilon}{|K|},$$

where $D = \int_0^1 \mathbf{1}_{\mathrm{supp}(f)}(\mu)d\mu \leq 1$ is the size of the support of f. Therefore,

$$1 - \frac{a\epsilon}{|K|\left(1-aD\right)} \leq \gamma \leq 1.$$

If hypothesis H_1 is true, the algorithm should provide a value greater than μ_*^*. We use E_l and P_l (where $l \in \{0,1\}$) to denote the expectation and probability respectively, under the algorithm being considered and under hypothesis H_l. Now, let

$$t = \frac{|K|}{8a\epsilon}\ln\left(\frac{3}{5\delta}\right),$$

and recall that T stands for the total number of samples from the pool.

Now, we assume we run an algorithm which is (ϵ,δ)-correct under H_0 and that $E_0[T] \leq t$ for this algorithm. We will show that this algorithm cannot be (ϵ,δ)-correct under hypothesis H_1. Therefore, an (ϵ,δ)-correct algorithm must have $E_0[T] > t$.

Define the following events:

- $A = \{T \leq 4t\}$. By the same consideration as in the proof of Theorem 1 (for the events $\{A_k\}_{k\in K}$), it follows that if $E_0[T] \leq t$, then $P_0(A) \geq \frac{3}{4}$.
- Let B stand for the event under which the chosen sample is smaller or equal to μ_*^*, and B^C for its complementary. Clearly, $P_0\left(B\right) = 1$.
- We define the event C to be the event under which all the samples obtained from the pool are on the interval $[0, \mu_*^*]$. Clearly, $P_0(C) = 1$.

Define now the intersection event $S = A \cap B^C \cap C$. We have shown that $P_0(A) \geq \frac{3}{4}$, $P_0(B) = 1$ and $P_0(C) = 1$, from which it is obtained that $P_0(S) \geq \frac{3}{4}$. In addition, since for every history h_N of N samples, for which the event C holds, it is obtained that $\frac{dP_1}{dP_0}(h_N) \geq \gamma^N$, we have the following,

$$\begin{aligned} P_1(B) \geq P_1(S) &= E_0\left[\frac{dP_1}{dP_0} I(S)\right] \geq \gamma^{-4t} P_0(I(S)) \\ &\geq \frac{3}{4}\gamma^{-4t} \geq \frac{3}{4} e^{-\frac{1}{2(1-a)} \ln \frac{3}{5\delta}} \geq \delta, \end{aligned}$$

where in the last inequality we used the facts that $(1-\epsilon)^{\frac{1}{\epsilon}} \geq e^{-1}$ and that $D \leq 1$.

We found that if an algorithm is (ϵ, δ)-correct under hypothesis H_0 and $E_0[T] \leq t$, then, under hypothesis H_1 this algorithm returns a sample that is smaller by at least ϵ than the maximal value with a probability of δ or more, hence the algorithm is not (ϵ, δ)-correct. Therefore, any (ϵ, δ)-correct algorithm, must satisfy $E_0[T] > t$. Hence the lower bound is obtained. ■

9. Appendix C

Proof [Theorem 4] Since sampling from the combined pool consists of choosing one pool out of the $|K|$ pools (with equal probability), and then, sampling from this pool, it follows that $f(\mu) \geq \frac{a}{|K|}$. So, $F(\mu_*^* - \epsilon) \leq \left(1 - \frac{a\epsilon}{|K|}\right)$. Also, we note that $(1-\epsilon)^{\frac{1}{\epsilon}} \leq e^{-1}$ for every $\epsilon \in (0, 1]$. Therefore, for $N = \lceil \frac{-\ln(\delta)|K|}{a\epsilon} \rceil + 1$

$$P\left(V_N^1 < \mu_*^* - \epsilon\right) = \left(F(\mu_*^* - \epsilon)\right)^N \leq \left(1 - \frac{a\epsilon}{|K|}\right)^N < \delta, \tag{11}$$

where V_N^1 is the largest value observed among the first N samples. Hence, the algorithm is (ϵ, δ)-correct. The bound on the sample complexity is immediate from the definition of the algorithm. ■