6 The Explore-then-Commit Algorithm

With the background on concentration out of the way, we are ready to present the first bandit policy of the book. The policy is simple. First it explores each action a fixed number of times and then exploits by playing the arm that appeared best after exploration.

For this chapter, as well as Chapters 7 to 9, we assume that all bandit instances are in $\mathcal{E}_{\text{SG}}^K(1)$, which means the reward distribution for all arms is 1-subgaussian.

The focus on subgaussian distributions is mainly for simplicity. Many of the techniques in the chapters that follow can be applied to other stochastic bandits such as those listed in Table 4.1. The key difference is that new concentration analysis is required that exploits the different assumptions. The Bernoulli case is covered in Chapter 10 where other situations are discussed along with references to the literature. As this stage, what is important to keep in mind is that larger environment classes correspond to harder bandit problems and we expect to see the effect of this in the strength of results. Notice that the subgaussian assumption restricts the subgaussian constant to $\sigma = 1$, which saves us from endlessly writing σ. All results hold for other subgaussian constants by scaling the rewards (see Lemma 5.2). Two points are obscured by this simplification.

1 All the algorithms that follow rely on the knowledge of σ.

2 It may happen sometimes that P_i is subgaussian for all arms, but with a different subgaussian constant for each arm. Algorithms are easily adapted to this situation if the subgaussian constants are known, a process we leave to the readers in Exercise 7.2. The situation where σ^2 is not known is more complicated and is discussed in Chapter 7 (and see Exercise 7.8).

We now describe the **explore-then-commit** (ETC) policy, which is characterized by the number of times it explores each arm, denoted by a natural number m. Because there are K actions, the algorithm will explore for mK rounds before choosing a single action for the remaining rounds. In order to define this policy formally we need a little more notation. Let $\hat{\mu}_i(t)$ be the average

reward received from arm i after round t, which is written formally as

$$\hat{\mu}_i(t) = \frac{1}{T_i(t)} \sum_{s=1}^{t} \mathbb{I}\{A_s = i\} X_s \,,$$

where $T_i(t) = \sum_{s=1}^{t} \mathbb{I}\{A_s = i\}$ is the number of times action i has been played after round t. The explore-then-commit policy is given in Algorithm 1 below.

1: **Input** $m \in \mathbb{N}$.
2: In round t choose action

$$A_t = \begin{cases} i \,, & \text{if } (t \bmod K) + 1 = i \text{ and } t \leq mK \,; \\ \operatorname{argmax}_i \hat{\mu}_i(mK) \,, & t > mK \,. \end{cases}$$

(ties in the argmax are broken arbitrarily)

Algorithm 1: Explore-then-commit policy

The formal definition of the explore-then-commit policy leads rather immediately to the analysis of its regret.

THEOREM 6.1 *The expected regret of the ETC policy is bounded by,*

$$R_n \leq \left(m \wedge \left\lceil \frac{n}{K} \right\rceil\right) \sum_{i=1}^{K} \Delta_i + (n - mK)^+ \sum_{i=1}^{K} \Delta_i \exp\left(-\frac{m\Delta_i^2}{4}\right) .$$

Proof By the decomposition given in Lemma 4.2 the regret can be written as

$$R_n = \sum_{i=1}^{K} \Delta_i \mathbb{E}\left[T_i(n)\right] .$$

In the first mK rounds the policy is completely deterministic, choosing each action exactly m times. Subsequently it chooses a single action maximizing the average reward during exploration. Thus,

$$\begin{aligned} \mathbb{E}\left[T_i(n)\right] &\leq m \wedge \left\lceil \frac{n}{K} \right\rceil + (n - mK)^+ \mathbb{P}\left(A_{mK+1} = i\right) \\ &\leq m \wedge \left\lceil \frac{n}{K} \right\rceil + (n - mK)^+ \mathbb{P}\left(\hat{\mu}_i(mK) \geq \max_{j \neq i} \hat{\mu}_j(mK)\right) . \end{aligned} \tag{6.1}$$

Now we need to bound the probability in the second term above. For the sake of simplifying the presentation, assume without loss of generality that arm one is optimal so that $\mu_1 = \mu^* = \max_i \mu_i$. Of course the learner does not know this. Then

$$\begin{aligned} \mathbb{P}\left(\hat{\mu}_i(mK) \geq \max_{j \neq i} \hat{\mu}_j(mK)\right) &\leq \mathbb{P}\left(\hat{\mu}_i(mK) \geq \hat{\mu}_1(mK)\right) \\ &= \mathbb{P}\left(\hat{\mu}_i(mK) - \mu_i - (\hat{\mu}_1(mK) - \mu_1) \geq \Delta_i\right) . \end{aligned}$$

The next step is to check that $\hat{\mu}_i(mK) - \mu_i - (\hat{\mu}_1(mK) - \mu_1)$ is $\sqrt{2/m}$-subgaussian,

which by the properties of subgaussian random variables follows from the definitions of $(\hat{\mu}_j)_j$ and the algorithm. Hence by Corollary 5.1,

$$\mathbb{P}\left(\hat{\mu}_i(mK) - \mu_i - \hat{\mu}_1(mK) + \mu_1 \geq \Delta_i\right) \leq \exp\left(-\frac{m\Delta_i^2}{4}\right) .$$

Substituting the above display into Eq. (6.1) and the regret decomposition shows that

$$R_n \leq \left(m \wedge \left\lceil \frac{n}{K} \right\rceil\right) \sum_{i=1}^{K} \Delta_i + (n - mK)^+ \sum_{i=1}^{K} \Delta_i \exp\left(-\frac{m\Delta_i^2}{4}\right) . \qquad \square$$

The above bound illustrates the fundamental challenge faced by the learner, which is the trade-off between exploration and exploitation. If m is large, then the policy explores for too long and the first term will be eventually too large. On the other hand, if m is too small, then the probability that the algorithm commits to the wrong arm will grow and the second term becomes too large. The question is how to choose m? Assume that $K = 2$ and that the first arm is optimal so that $\Delta_1 = 0$ and abbreviate $\Delta = \Delta_2$. Then the above display simplifies to

$$R_n \leq m\Delta + (n - 2m)^+ \Delta \exp\left(-\frac{m\Delta^2}{4}\right) \leq m\Delta + n\Delta \exp\left(-\frac{m\Delta^2}{4}\right) . \tag{6.2}$$

Provided that n is reasonably large the quantity on the right-hand side of the above display is minimized (up to a possible rounding error) by

$$m = \max\left\{0, \left\lceil \frac{4}{\Delta^2} \log\left(\frac{n\Delta^2}{4}\right) \right\rceil\right\} \tag{6.3}$$

and for this choice and any n the regret is bounded by

$$R_n \leq \Delta + \frac{4}{\Delta}\left(1 + \max\left\{0, \log\left(\frac{n\Delta^2}{4}\right)\right\}\right) . \tag{6.4}$$

Notice that Δ appears in the denominator of the regret bound, which means that as Δ becomes very small the regret bound grows unboundedly. Is that reasonable and why is it happening? The explanation is simply that in the second inequality of (6.2) we have over-enthusiastically upper bounded $(n - mK)^+ \leq n$ and $m \wedge \lceil n/K \rceil \leq m$, regardless of the value of n. To overcome this weakness we simply notice that $R_n = \Delta \mathbb{E}[T_2(n)] \leq n\Delta$. Taking the minimum of this and the bound shown in (6.4),

$$R_n \leq \min\left\{n\Delta, \Delta + \frac{4}{\Delta}\left(1 + \log\left(\frac{n\Delta^2}{4}\right)\right)\right\} . \tag{6.5}$$

We leave it as an exercise to the reader to check that $R_n = O(\sqrt{n})$ regardless of the value of Δ. Bounds of this nature are called **worst-case**, **problem free** or **problem independent** (see Eq. (4.2) or Eq. (4.3)). The reason is that the bound depends only on the distributional assumption and not on the specific bandit. Sometimes bounds of this type are also called gap-free as they do not

depend on Δ. In contrast, bounds that depend on the suboptimality gaps are called **gap/problem/distribution/instance dependent**.

Later you will see that the bound in (6.4) is very close to optimal in a number of ways. There is one big caveat, however. The optimal choice of m that defines the policy and leads to this bound depends on both the suboptimality gap and the horizon. While sometimes the horizon might be known in advance, it is practically never reasonable to assume knowledge of the suboptimality gaps. So is there a reasonable way to choose m that does not depend on the unknown gap, but may depend on n? It turns out that the there is a choice for which $R_n = O(n^{2/3})$ regardless of the value of Δ. The need to know the suboptimality gap can be overcome by allowing m to be data-dependent. That is, the learner chooses each arm alternately until it decides based on its observations to commit to a single arm for the remaining rounds. We return to this point briefly in the notes at the end of the section, but will not be cover the details because it turns out there are other algorithms that are superior in practice and do not even need to know the horizon.

So how does this play out on actual data? To examine this question we plot the expected regret of ETC when playing a two-armed Gaussian bandit with means $\mu_1 = 0$ and $\mu_2 = -\Delta$. The horizon is set to $n = 1000$ and $\Delta > 0$ is varied between 0 and 1. The plot shows three curves:

(a) The theoretical upper bound given in Eq. (6.5).
(b) The regret of the ETC algorithm with m set as suggested in Eq. (6.3).
(c) The regret of the ETC algorithm with the optimal m, which may be calculated numerically using the assumption that the noise is exactly Gaussian.

Each data point is the average of 10^4 simulations, which means that error bars are so small that they are not visible (and hence no attempt is made to show them). The figure shows the actual performance of the ETC algorithm roughly tracks the theory and the optimally tuned version provides a modest improvement in performance. It is important to emphasize that the optimally tuned algorithm is 'cheating' in an even stronger way than the choice based on (6.3) because the calculation of the optimal m was based on a Gaussian assumption, while the choice given in Eq. (6.3) only relied on the noise being subgaussian.

6.1 Notes

1 An algorithm is called **anytime** if it does not require advance knowledge of the horizon n. As discussed, explore-then-commit is not anytime because the choice of commitment time depends on the horizon. This limitation can be

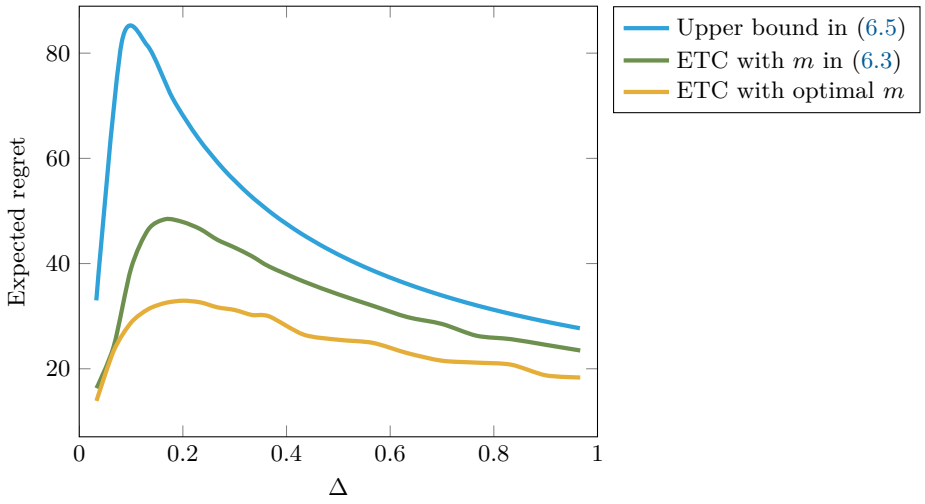

Figure 6.1 Expected regret of ETC strategies

addressed in a general way by using the **doubling trick**, which is a simple way to convert a horizon-dependent algorithm into an **anytime algorithm** that does not depend on the horizon. We explain in Exercise 6.6.

2 By allowing the exploration time m to be a data-dependent random variable it is possible to recover near-optimal regret without knowing the suboptimality gaps. For examples where this is done, see the articles by Auer and Ortner [2010] and Garivier et al. [2016b] and Exercise 6.5.

6.2 Bibliographical remarks

Explore-then-commit has a long history. Robbins [1952] considered 'certainty equivalence with forcing', which chooses the arm with the largest sample mean except at a fixed set of times $T_k \subset \mathbb{N}$ when arm k is chosen for $k \in [K]$. By choosing the set of times carefully it is shown that this policy enjoys sublinear regret. While ETC performs all the exploration at the beginning, Robbins's policy spreads the exploration over time. This is advantageous if the horizon is not known, but disadvantageous otherwise. Anscombe [1963] considered exploration and commitment in the context of medical trials or other experimental setups. He already largely solves the problem in the Gaussian case and highlights many of the important considerations. Besides this, the article is beautifully written and well worth reading. Strategies based on exploration and commitment are simple to implement and analyze. They can also generalize well to more complex settings. For example, Langford and Zhang [2008] considers this style of policy under the name 'epoch-greedy' for contextual bandits (the idea of exploring then exploiting in epochs, or intervals, is essentially what Robbins [1952] suggested).

We'll return to contextual bandits in Chapter 18. Abbasi-Yadkori et al. [2009] (see also Abbasi-Yadkori 2009b) and Rusmevichientong and Tsitsiklis [2010] consider ETC-style policies under the respective names of 'forced exploration' and 'phased exploration and greedy exploitation' (PEGE) in the context of linear bandits (which we shall meet in Chapter 19). Other names include 'forced sampling', 'explore-first', 'explore-then-exploit'. Garivier et al. [2016b] have shown that ETC policies are necessarily suboptimal in the limit of infinite data in a way that is made precise in Chapter 16. As mentioned earlier, ε-greedy is a close relative of ETC and can be viewed as the randomized version of Robbin's algorithm. One may ask why you would want to randomize? The best answer is simplicity: The policy only depends on the exploration parameter ε rather than some complicated schedule. This can be useful in complicated settings like reinforcement learning where many instances of the same algorithm are acting simultaneously and communication is costly. In Chapter 11 we'll see the role of randomization when the bandit itself is allowed to react to the actions of the learner in a malicious way. The history of ε-greedy is unclear, but it is a popular and widely used and known algorithm in reinforcement learning [Sutton and Barto, 1998]. Auer et al. [2002a] analyze the regret of ε-greedy with slowly decreasing ε (see Exercise 6.7). There are other kinds of randomized exploration as well, including Thompson sampling [1933] and Boltzmann exploration analyzed recently by Cesa-Bianchi et al. [2017].

6.3 Exercises

6.1 In the proof of Theorem 6.1 we wrote: "The next step is to check that $\hat{\mu}_i(mK) - \mu_i - (\hat{\mu}_1(mK) - \mu_1)$ is $\sqrt{2/m}$-subgaussian, which by the properties of subgaussian random variables follows from the definitions of $\{\hat{\mu}_j\}_j$ and the algorithm." Prove this in a fully rigorous manner. You can use the interaction protocol, the assumption that rewards are 1-subgaussian, the definition of $\{\hat{\mu}_j\}_j$ and the definition of ETC. In particular, prove that $\hat{\mu}_j(mK)$ is the sample mean of m i.i.d. random variables chosen from P_j. Note that this is *not* the definition of $\hat{\mu}_j(mK)$. Further, the interaction protocol only specifies that $X_t \sim P_{A_t}$, independently of $(A_1, X_1, \dots, A_{t-1}, X_{t-1})$ given A_t.

6.2 Fix $\delta \in (0,1)$. Modify the ETC algorithm to depend on δ and prove a bound on the pseudo-regret $\bar{R}_n = n\mu^* - \sum_{t=1}^n \mu_{A_t}$ of ETC that holds with probability $1-\delta$.

6.3 Fix $\delta \in (0,1)$. Prove a bound on the random regret $\hat{R}_n = n\mu^* - \sum_{t=1}^n X_t$ of ETC that holds with probability $1-\delta$. Compare this to the bound derived for the pseudo-regret in the previous exercise. What can you conclude?

6.4 In this exercise we investigate the empirical behavior of the Explore-Then-Commit algorithm on a two-armed Gaussian bandit with means $\mu_1 = 0$ and

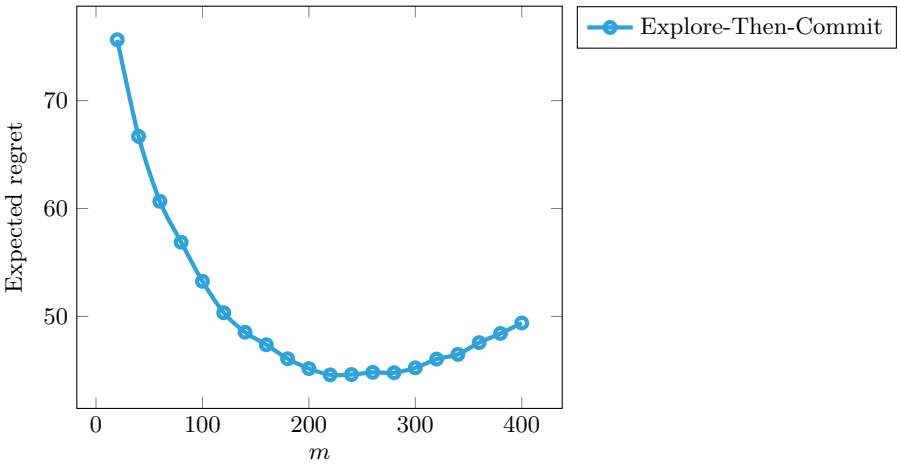

Figure 6.2 Expected regret for Explore-Then-Commit over 10^5 trials on a Gaussian bandit with means $\mu_1 = 0, \mu_2 = -1/10$

$\mu_2 = -\Delta$. Let

$$\bar{R}_n = \sum_{t=1}^{n} \Delta_{A_t} ,$$

which is chosen so that $R_n = \mathbb{E}[\bar{R}_n]$. Complete the following:

(a) Using programming language of your choice, write a function that accepts an integer n and $\Delta > 0$ and returns the value of m that *exactly* minimizes the expected regret.

(b) Reproduce Fig. 6.1, which shows the expected regret of the ETC algorithm for different choices of m as a function of Δ.

(c) Now fix $\Delta = 1/10$ and plot the expected regret as a function of m with $n = 2000$. Your plot should resemble Fig. 6.2.

(d) Plot the variance $\mathbb{V}[\bar{R}_n]$ as a function of m for the same bandit as above. Your plot should resemble Fig. 6.3.

(e) Explain the shape of the curves you observed in Parts (b), (c) and (d) and reconcile what you see with the theoretical results.

(f) Think, experiment and plot. Is it justified to plot $\mathbb{V}[\bar{R}_n]$ as a summary of how $\bar{R}_n$ is distributed? Explain your thinking.

6.5 In this question we investigate how far we can push the ETC algorithm. Assume for the purpose of this exercise that ETC interacts with a 2-armed 1-subgaussian bandit with means $\mu_1, \mu_2 \in \mathbb{R}$ such that $\Delta = |\mu_1 - \mu_2|$.

(a) Find a choice of m that depends only on the horizon n and *not* Δ such the

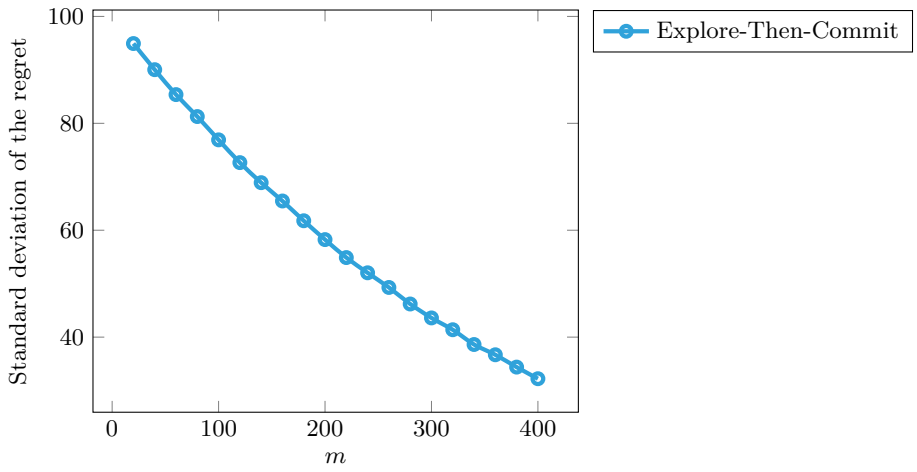

Figure 6.3 Standard deviation of the regret for ETC over 10^5 trials on a Gaussian bandit with means $\mu_1 = 0, \mu_2 = -1/10$

regret of Algorithm 1 is bounded by

$$R_n \leq C\left(\Delta + n^{2/3}\right),$$

where $C > 0$ is a universal constant.

(b) Now suppose the commitment time is allowed to be data-dependent, which means the algorithm explores each arm alternately until some condition is met and then commits to a single arm for the remainder. Design a condition such that the regret of the resulting algorithm can be bounded by

$$R_n \leq C\left(\Delta + \frac{\log n}{\Delta}\right), \tag{6.6}$$

where C is a universal constant. Your condition should only depend on the observed rewards and the time horizon. It should *not* depend on μ_1, μ_2 or Δ.

(c) Show that any algorithm for which (6.6) holds also satisfies $R_n \leq C(\Delta + \sqrt{n \log(n)})$ for suitably chosen universal constant $C > 0$.

(d) As for part (b), but now the objective is to design a condition such that the regret of the resulting algorithm is bounded by

$$R_n \leq C\left(\Delta + \frac{\log \max\left\{e, n\Delta^2\right\})}{\Delta}\right), \tag{6.7}$$

(e) Show that any algorithm for which (6.7) holds also satisfies $R_n \leq C(\Delta + \sqrt{n})$ for suitably chosen universal constant $C > 0$.

For Part (a) start from $R_n \leq m\Delta + n\Delta \exp(-m\Delta^2/2)$ and assume that the second term here dominates the first term. Find Δ maximizing the resulting

regret upper bound. Based on this propose m and verify that the starting assumption has been met by your choice regardless the values of n and Δ. For Part (b) think about the simplest stopping policy and then make it 'robust' by using confidence intervals. Tune the failure probability. For Part (c) note that the regret can never be larger than $n\Delta$.

In the later parts of Exercise 6.5 we allowed the commitment time to be data-dependent. This makes it a random variable and so it should be capitalized to M instead of m. In the language of formal probability, the random variable $T = 2M$ is called a stopping time with respect to the filtration $(\mathcal{F}_t)_{t\geq 1}$ where $\mathcal{F}_t = \sigma(A_1, X_1, \ldots, A_t, X_t)$. We'll introduce the formal definition of a stopping time in a later chapter, but for now we mention briefly that it means that the event $\{T = t\}$ is $\mathcal{F}_t$-measurable for each t. The curious reader might like to think about what this definition means. (**Hint:** The algorithm cannot commit on the basis of information it does not have!)

6.6 Let $\mathcal{E}$ be an arbitrary class of bandits (for example, $\mathcal{E} = \mathcal{E}_{\mathrm{SG}}^K$). Suppose you are given a policy $\mathcal{A}$ designed for $\mathcal{E}$ that accepts the horizon n as a parameter and has a regret guarantee of

$$R_n \leq f_n(\nu)\,, \qquad \forall \nu \in \mathcal{E}\,,$$

where $f_n : \mathcal{E} \to [0, \infty)$ is a sequence of functions. The purpose of this exercise is to analyze a meta-algorithm based on the so-called **doubling trick** that converts a policy depending on the horizon to a policy with similar guarantees that does not. Let $n_1 < n_2 < n_3 < \cdots$ be a fixed sequence of integers and consider the policy that runs $\mathcal{A}$ with horizon n_1 until round $t = \max\{n, n_1\}$. Then restarts the algorithm with horizon n_2 until $t = \max\{n, n_1 + n_2\}$. Then restarts again with horizon n_3 until $t = \max\{n, n_1 + n_2 + n_3\}$ and so-on.

(a) Let $\ell_{\max} = \min\{\ell : \sum_{i=1}^{\ell} n_i \geq n\}$. Prove that the regret of the meta-algorithm is at most

$$R_n \leq \sum_{\ell=1}^{\ell_{\max}} f_{n_\ell}(\nu)\,.$$

(b) Suppose that $f_n(\nu) \leq \sqrt{n}$. Show that if $n_\ell = 2^{\ell-1}$, then the regret of the meta-algorithm is at most

$$R_n \leq C\sqrt{n}\,,$$

where $C > 0$ is a carefully chosen universal constant.

(c) Suppose that $f_n(\nu) = g(\nu)\log(n)$ for some function $g : \mathcal{E} \to [0, \infty)$. What is the regret of the meta-algorithm if $n_\ell = 2^{\ell-1}$? Can you find a better choice of $(n_\ell)_\ell$?

(d) In light of this idea, should we bother trying to design algorithms that do not depend on the horizon? Are there any disadvantages to using the doubling trick? If so, what are they?

6.7 For this exercise assume the rewards are 1-subgaussian and there are $K \geq 2$ arms. The ε-greedy algorithm depends on a sequence of parameters $\varepsilon_1, \varepsilon_2, \ldots$. First it chooses each arm once and subsequently chooses $A_t = \operatorname{argmax}_i \hat{\mu}_i(t-1)$ with probability $1 - \varepsilon_t$ and otherwise chooses an arm uniformly at random.

(a) Prove that if $\varepsilon_t = \varepsilon > 0$, then $\lim_{n\to\infty} \frac{R_n}{n} = \frac{\varepsilon}{K} \sum_{i=1}^{K} \Delta_i$.

(b) Let $\Delta_{\min} = \min\{\Delta_i : \Delta_i > 0\}$ and let $\varepsilon_t = \min\left\{1, \frac{CK}{t\Delta_{\min}^2}\right\}$ where $C > 0$ is a sufficiently large universal constant. Prove that there exists a universal $C' > 0$ such that

$$R_n \leq C' \sum_{i=1}^{K} \left(\Delta_i + \frac{\Delta_i}{\Delta_{\min}^2} \log \max\left\{ e, \frac{n\Delta_{\min}^2}{K} \right\} \right).$$

6.8 A simple way to generalize the ETC policy to multiple arms and overcome the problem of tuning the commitment time is to use an **elimination algorithm**. The algorithm operates in phases and maintains an **active set** of arms that could be optimal. In the ℓth phase the algorithm aims to eliminate from the active set all arms i for which $\Delta_i \geq 2^{-\ell}$.

Input: K and sequences $(\tau_\ell)_\ell$
$A_1 = \{1, 2, \ldots, K\}$
for $\ell = 1, 2, 3, \ldots$ **do**
    Choose $i \in A_\ell$ each arm τ_ℓ times
    Let $\hat{\mu}_{i,\ell}$ be the average reward for arm i from this phase
    Update active set:

$$A_{\ell+1} = \left\{ i : \hat{\mu}_{i,\ell} + 2^{-\ell} \geq \max_{j \in A_\ell} \hat{\mu}_{j,\ell} \right\}$$

end for

Without loss of generality, assume that arm 1 is an optimal arm.

(a) Show that for any $\ell \geq 1$,

$$\mathbb{P}\left(1 \notin A_{\ell+1}, 1 \in A_\ell\right) \leq K \exp\left(-\frac{\tau_\ell 2^{-2\ell}}{4}\right)$$

(b) Show that if $i \in [K]$ and $\ell \geq 1$ are such that $\Delta_i \geq 2^{-\ell}$, then

$$\mathbb{P}\left(i \in A_{\ell+1},\ 1 \in A_\ell,\ i \in A_\ell\right) \leq \exp\left(-\frac{\tau_\ell (\Delta_i - 2^{-\ell})^2}{4}\right)$$

(c) Let $\ell_i = \min\left\{\ell \geq 1 : 2^{-\ell} \leq \Delta_i/2\right\}$. Choose τ_ℓ in such a way that $\mathbb{P}\left(\exists \ell : 1 \notin A_\ell\right) \leq 1/n$ and $\mathbb{P}\left(i \in A_{\ell_i+1}\right) \leq 1/n$.

(d) Show that your algorithm has regret at most

$$R_n \leq C \sum_{i:\Delta_i>0} \left(\Delta_i + \frac{1}{\Delta_i}\log(n)\right) ,$$

where $C > 0$ is a carefully chosen universal constant.

(e) Modify your choice of τ_ℓ (if necessary) to derive a regret bound

$$R_n \leq C \sum_{i:\Delta_i>0} \left(\Delta_i + \frac{1}{\Delta_i}\log\max\left\{e, n\Delta_i^2\right\}\right) .$$

6.9 For this exercise assume the rewards are 1-subgaussian. The ε-greedy algorithm depends on a sequence of parameters $\varepsilon_1, \varepsilon_2, \ldots$. First it chooses each arm once and subsequently chooses $A_t = \operatorname{argmax}_i \hat{\mu}_i(t-1)$ with probability $1-\varepsilon_t$ and otherwise an arm uniformly at random.

(a) Prove that if $\varepsilon_t = \varepsilon > 0$, then $\lim_{n\to\infty} \frac{R_n}{n} = \frac{\varepsilon}{K}\sum_{i=1}^{K} \Delta_i$.

(b) Let $\Delta_{\min} = \min\left\{\Delta_i : \Delta_i > 0\right\}$ and let $\varepsilon_t = \min\left\{1, \frac{CK}{t\Delta_{\min}^2}\right\}$ where $C > 0$ is a sufficiently large universal constant. Prove that there exists a universal $C' > 0$ such that

$$R_n \leq C' \sum_{i=1}^{K} \left(\Delta_i + \frac{\Delta_i}{\Delta_{\min}^2}\log\max\left\{e, \frac{n\Delta_{\min}^2}{K}\right\}\right) .$$