A Farewell to Arms: Sequential Reward Maximization on a Budget with a Giving Up Option

P Sharoff
Department of Computer Science
University of Victoria
sharoff@uvic.ca

Nishant A. Mehta
Department of Computer Science
University of Victoria
nmehta@uvic.ca

Ravi Ganti
Google
gmravi2003@gmail.com

Abstract

We consider a sequential decision-making problem where an agent can take one action at a time and each action has a stochastic temporal extent, i.e., a new action cannot be taken until the previous one is finished. Upon completion, the chosen action yields a stochastic reward. The agent seeks to maximize its cumulative reward over a finite time budget, with the option of "giving up" on a current action — hence forfeiting any reward – in order to choose another action. We cast this problem as a variant of the stochastic multi-armed bandits problem with stochastic consumption of resource. For this problem, we first establish that the optimal arm is the one that maximizes the ratio of the expected reward of the arm to the expected waiting time before the agent sees the reward due to pulling that arm. Using a novel upper confidence bound on this ratio, we then introduce an upper confidence based-algorithm, Wait-UCB, for which we establish logarithmic, problem-dependent regret bound which has an improved dependence on problem parameters compared to previous works. Simulations on various problem configurations comparing Wait-UCB against the state-of-the-art algorithms are also presented.

1 Introduction

In online learning, regret is not always about taking the wrong action. In the real world, an action can take some unknown time to return its reward, and, due to resources being tied up, a learning agent may be unable to take its next action until the previous action is completed. The learner is thus presented with a choice: should it wait until its current action returns a reward or instead bid farewell to the chosen action after a selected waiting time has transpired in order to take a different action.

Consider, for example, a model of crowdsourcing where an employer submits tasks to a crowdsourcing platform which has a large pool of workers. The employer wishes to have as many tasks as possible completed under a fixed time budget. However, for a given type of task, the efficiency of workers can vary significantly. While some omniscient actor could dispatch tasks only to fast workers, a typical employer would not know the efficiency of different workers on the task at hand. If a randomly selected worker is fast, it can be worthwhile to wait for the worker to complete a job; however, if a worker takes too long to complete a task, it can be advantageous for the employer to terminate the assignment and reassign the task to a new worker. How long should the employer wait for an assigned task to be completed before reassigning the task to a new worker? This problem fits into a new multi-armed bandit framework, *Farewell to Arms* (F2A), that introduces the idea of "giving up" on an action (arm) that takes too long to return a reward; for a given arm, we view choosing the waiting time itself as pulling a kind of "micro-arm". As we explain in Section 2.2, other applications include hyperparameter tuning, repeated second-price auctions with participation costs, and computational advertising.

Informally, the F2A framework is a variant of the classical stochastic K-armed bandit problem framework, but with an added twist of a stochastic resource consumption. For simplicity, consider first the case of the F2A framework with only a single arm. Upon pulling the single arm the learning agent gets some stochastic reward, not instantaneously, but only after a stochastic

Proceedings of the 23rd International Conference on Artificial Intelligence and Statistics (AISTATS) 2020, Palermo, Italy. PMLR: Volume 108. Copyright 2020 by the author(s).

delay[1], say τ. There could be cases where the delay is extremely large. To incorporate this notion, the learning agent is not only required to choose an arm, but is also required to commit to a waiting time j. The learning agent receives a reward if $j \geq \tau$, i.e., the agent is willing to wait until the reward arrives, and gets a reward of 0 otherwise. Hence, the F2A framework can be thought of as a 2-level stochastic multi-armed bandit framework with K "macro-arms" and (say) D "micro-arms" for each of the K macro-arms. The micro-arms capture the willingness of the learning agent to wait and hence bound the total amount of time that can be consumed by a pull, while the macro-arms capture the goodness of a certain arm.

An F2A problem can be cast into the more general bandits with knapsack (BwK) framework (Badanidiyuru et al., 2013) by considering the latter with only a single resource. However, whereas Badanidiyuru et al. (2013) established worst-case regret bounds for the BwK problem, in this work we establish (logarithmic) problem-dependent regret bounds. Such logarithmic problem-dependent regret bounds tend to be sharper and better capture problem complexity in the standard stochastic multi-armed bandit case when the gap between the best and the second best arm is large. The pioneering work of Flajolet and Jaillet (2017) established, for their algorithm UCB-Simplex, logarithmic problem-dependent regret bounds (growing logarithmically in the number of rounds) that also apply to F2A problems. However, the new algorithm we develop — Wait-UCB — is based on a rather different upper confidence bound than the one used by UCB-Simplex; moreover, the regret bounds we prove for Wait-UCB are often better than those of Flajolet and Jaillet (2017) and also those derived by Xia et al. (2016) for their algorithm Budget-UCB. Furthermore, we also show that Wait-UCB has better empirical performance than UCB-Simplex and Budget-UCB (Xia et al., 2016). Many important problems (as discussed in Section 2.2) fall in the more specialized F2A framework and hence this framework, despite being a special case of the BwK framework, demands a specialized treatment and better algorithms.

Our core contributions are as follows:

- We introduce the Farewell to Arms framework.
- We show in Section 4 that, due to the stochastic consumption of resources in an F2A problem, the right quality measure for an arm is the ratio of expected reward to expected waiting time.
- We derive in Section 5 a novel upper confidence bound for the ratio of mean reward to mean waiting time; using this bound, we design Wait-UCB, a new upper confidence-style algorithm.
- We establish (Section 6) a logarithmic problem-dependent regret guarantee for Wait-UCB which is never worse than $\mathcal{O}((D^3/\Delta)\log T)$, where Δ is the gap between the aforementioned ratio for the best arm to the best suboptimal arm. In important regimes for F2A problems (like when mean waiting times for most arms are small), our bound can be D times smaller than the regret bounds for UCB-Simplex (Flajolet and Jaillet, 2017) and Budget-UCB (Xia et al., 2016).
- We provide (Section 7) a detailed experimental study of Wait-UCB, including comparisons to UCB-Simplex and Budget-UCB which suggest that Wait-UCB fares better for F2A problems.

[1]The stochastic rewards and delays can be dependent.

2 The Farewell to Arms framework

We now formally introduce the Farewell to Arms framework and then show how it captures several important applications.

2.1 A Farewell to Arms game

In the F2A framework, there is a hierarchical set of arms with K macro-arms at the top level and, for each macro-arm, D micro-arms at the next level. We will index macro-arms with $k \in [K] := \{1, 2, \ldots, K\}$ and micro-arms with $j \in [D]$. Associated with each macro-arm k, there is a joint distribution Q_k over $[0,1] \times [D]$, for a space of rewards $[0,1]$[2] and a space of delays $[D]$. A game in the F2A framework lasts for T time units and proceeds in epochs. In each epoch $s = 1, 2, \ldots,$

1. The learning agent plays a macro/micro-arm pair $i_s := (k, j) \in [K] \times [D]$.
2. Independently of the learning agent's choice, the stochastic environment draws a potential reward of $V_{k,s} \in [0,1]$ and a delay of $\tau_{k,s} \in [D]$ from distribution Q_k.[3]
3. The agent collects rewards $r_{i_s}^{(s)} := V_{k,s} \cdot \mathbf{1}\left[\tau_{k,s} \leq j\right]$ and consumes $c_{i_s}^{(s)} := \min\{\tau_{k,s}, j\}$ units of time.

Epochs can be of variable length in the F2A framework. This is in contrast to the standard multi-armed

[2]We assume rewards are bounded, in which case it is without loss of generality that we can and will assume that they fall in the unit interval.

[3]We consider a finite number of delay values. The same ideas with minor technical changes can be applied for the case when the number of delay values is finite but the set of delays is not necessarily equal to $[D]$.

bandit framework, where each epoch has unit length. Hence, given a total time budget of T units in the F2A framework there can be a variable number of epochs, whereas in the multi-armed bandit framework there are exactly T epochs.

The goal of the learning algorithm is to maximize reward within the fixed time budget of T. We will study the *pseudo-regret* of the learning algorithm against the best constant policy: the pseudo-regret of a learning algorithm that plays the macro/micro-arm pair sequence $i_1, i_2, \ldots$ is

$$R_T := \max_{(k,j)\in[K]\times[D]} \mathsf{E}\left[\sum_{s=1}^{L_{k,j}} r_{k,j}^{(s)}\right] - \mathsf{E}\left[\sum_{s=1}^{L} r_{i_s}^{(s)}\right]; \quad (1)$$

where $L_{k,j}$ and L are the random stopping times when playing the macro/micro-arm pair sequence $((k,j),(k,j),\ldots,))$ and $(i_1, i_2, \ldots)$ respectively. Intuitively, it seems that an optimal (constant) policy is one that maximizes the average reward obtained per unit time. In Theorem 1 we show that this is indeed true and that the *ratio estimator* $\frac{\mathsf{E}[r]}{\mathsf{E}[c]}$, where r is the reward and c is the amount of resource consumed when a certain macro/micro-arm pair is pulled, is indeed the right estimator to optimize for.

2.2 Applications of the F2A framework

In addition to the crowdsourcing example mentioned earlier, many other applications fit into the F2A framework. We now present a few of them.

Repeated second price auctions. In a repeated, sealed, second price auction (Weed et al., 2016) with participation costs (Gal et al., 2007; Stegeman, 1996; McAfee and McMillan, 1987; Samuelson, 1985), the goal is to maximize the expected cumulative reward given a budget of B dollars. In each round s of the auction, a bidder pays a flat (positive) participation cost of c dollars and selects a bid $i_s \in \{b_1, b_2, \ldots, b_D\}$ along with the other competing bidders; the bidder wins the auction if their bid was the highest. If the bidders wins the auction, they get a reward of V_s and consume a budget of $c + M_s$ dollars, where M_s is the second highest bid. If the bidder loses the auction, their reward is 0 but they consume c dollars of their budget. The game ends once the budget is no longer positive. Under appropriate stochastic assumptions on the items and bids — namely, that the items are drawn i.i.d. (so that the utilities V_s are i.i.d.) and that the highest bids M_s among the other bidders also are i.i.d.[4] — this problem can be cast into the F2A framework where there is a single macro-arm and the D micro-arms are the values of the bids.

[4]We allow V_s and M_s to be dependent.

k-fold cross-validation. Consider the problem of performing hyperparameter selection via k-fold cross-validation (CV). In k-fold CV, we are required to run a learning algorithm to convergence a very large number of times. For each of a typically large number of hyperparameter configurations, we need to run the learning algorithm on each of k subsets of the training data. Each execution can take a different amount of time to converge, and with random initialization the runtime and also the model learned by the algorithm are stochastic. Suppose that we have a fixed time budget and view the quality of the model learned as potential reward (high quality solution meaning a large reward); then a natural goal is to find a set of hyper-parameters that are near-optimal. A natural formulation of this problem is to cast it as a pure-exploration multi-armed bandit problem (Li et al., 2017). In this paper, we cast the k-fold CV problem as a regret minimization problem, inspired by a similar regret minimization approaches used in bandit convex optimization (Agarwal et al., 2011). In practice, k-fold CV is done with an implicit time budget constraint, where long-running experiments are terminated, receiving a reward of 0, and the experiments are re-started from a different parameter configuration. This practical consideration means that we want to maximize the total cumulative reward under a given total time budget and hence makes this problem fit well into the F2A framework: the macro-arms are the various parameter configurations, and the micro-arms are the amount of time/computational resources we are willing to allocate for different experiments.

Computational advertising. In computational advertising, a publisher wants to show an ad from an inventory of ads. When a user sees a published ad, the user might be interested in it but might not click on the ad immediately. In such cases, there is an economic incentive for the publisher to display the ad for multiple time periods and wait for a response from the user rather than switch to a different ad immediately. However, at the same time the publisher (learning agent) would like to minimize their regret of not showing the best ad. This problem can be cast in the F2A framework, where the macro-arms are the various ads and the micro-arms are the different durations for which the publisher is willing to wait.

3 Related work

On the surface, F2A problems bear close similarity to the problem of learning the optimal waiting time. In the latter problem, studied in detail by Lattimore et al. (2014), in each of a fixed number of rounds a learning agent selects a waiting time. If a stochastic delay exceeds this waiting time, the agent suffers a loss equal to the waiting time plus a fixed cost; otherwise, the agent suffers the stochastic delay itself plus a different fixed cost. While learning an optimal waiting time is a common thread between the work of Lattimore et al. (2014) and our work, there are three key differences: in an F2A problem, *(i)* the time spent affects a budget; *(ii)* there is separate collection of stochastic reward (which is not present at all in the work of Lattimore et al. (2014)); and *(iii)* the game has a random stopping time that depends on all the actions taken, making exploration more challenging. Consequently, the optimal waiting time differs considerably in our setting, instead depending on a ratio of means.

The F2A framework is however very related to the bandits with knapsacks (BwK) setting (Badanidiyuru et al., 2013) (see also the earlier work of Tran-Thanh et al. (2012) and Ding et al. (2013)). A BwK problem is a generalization of the classical multi-armed bandit problem (Lai and Robbins, 1985) in which the learning agent has finite quantities of a number of resources, and each pull of an arm stochastically consumes each resource while also yielding some stochastic reward (the reward and resource consumptions in each round can be dependent). The game ends once any resource is exhausted. The classical multi-armed bandit problem is recovered by taking time as the single resource, which deterministically decreases by 1 when any arm is pulled (the game ends when the finite time budget is exhausted). F2A problems also can be cast in the BwK setting, now by taking time as a single resource which is consumed stochastically. However, the full BwK setting is so general that the algorithms developed for this setting, and the type of regret guarantees given, differ substantially from the type of guarantees we seek here. In particular, we seek problem-dependent bounds with regret growing logarithmically with the size of the budget. Such bounds previously were obtained by Ding et al. (2013), Xia et al. (2016) and Flajolet and Jaillet (2017), but, as we explain in Section 6, in the case of F2A problems our results can be better in important regimes.

Finally, we mention in passing that there also are weak connections to two other settings. In bandits with delayed feedback (Joulani et al., 2013), feedback from arms can be delayed, but the learning agent can still pull an arm in every round; this difference is critical. In bandits with lock-up periods (Komiyama et al., 2013), feedback is not delayed and an arm can be pulled in each round, but the learning agent experiences "lock-up" periods during which it must constantly pull the same arm. This is similar to our waiting period, but an important difference is that at the end of a waiting period in our setting, the learning agent only receives one reward (possibly equal to zero), whereas in the lock-up period setting, a reward is received in each round.

4 A ratio estimator for F2A problems

In this section, we find a suitable metric that captures both the reward and resource aspects of F2A problems. An upper confidence bound for this metric will be key in designing the Wait-UCB algorithm in Section 5. For notational convenience, in this section we only consider the case of one macro-arm, so $K = k = 1$ (but of course with multiple micro-arms).

Given a finite time budget and full knowledge of the data-generating distribution, which constant policy maximizes expected cumulative reward? As pulls can have stochastic extent, intuitively this policy should always pull the micro-arm that maximizes the ratio of expected reward to expected waiting time. Our first main result gives formal backing to this intuition.

Theorem 1 *Let $(V_1, \tau_1), (V_2, \tau_2), \ldots$ be i.i.d. according to a joint distribution Q over $[0,1] \times [D]$. Consider the constant policy which pulls micro-arm j in each epoch, so that in a given epoch s this arm consumes (i.e., waits for) $c_j^{(s)} := \min\{j, \tau_s\}$ units of time and collects reward $r_j^{(s)} := \mathbf{1}[j \geq \tau_s] \cdot V_s$. Then the total cumulative reward collected by this constant policy under a time budget of T is equal to*

$$\mathsf{E}\left[\sum_{s=1}^{L} r_j^{(s)}\right] = T \cdot \frac{\mathsf{E}[r_j^{(1)}]}{\mathsf{E}[c_j^{(1)}]} + A_j$$

for some constant $A_j \in [0, j]$, where $L = \max\{n : \sum_{s=1}^{n} c_j^{(s)} \leq T\}$ is the last epoch before the game ends.

From the above theorem, it is clear that to maximize expected cumulative reward, we should devise an estimator for the expected per-round reward of each micro-arm. Let us introduce notation for the quantities we wish to estimate. In the following, let $(V, \tau) \sim Q_1$. For $j \in [D]$, define the expected per-round reward

$$g_{1,j} := \frac{\mathsf{E}\left[\mathbf{1}[j \geq \tau] \cdot V\right]}{\mathsf{E}\left[\min\{j, \tau\}\right]}. \tag{2}$$

A natural estimator of $g_{1,j}$ is

$$\hat{g}_{1,j}(s) := \frac{\sum_{m=1}^{s} \mathbf{1}[i_m = (1,j)] \cdot \mathbf{1}[j \geq \tau_m] \cdot V_m}{\sum_{m=1}^{s} \mathbf{1}[i_m = (1,j)] \cdot \min\{j, \tau_m\}}. \tag{3}$$

The above expression records the total reward received from pulls of arm-pair $(1, j)$ divided by the total rounds spent during these pulls. Comparing to Theorem 1, we can see that the numerator of (3) is an unbiased estimator of the numerator of Theorem 1, and the same relationship holds between the respective denominators. However, the full estimator above is not an unbiased estimator of the expected per-round reward $g_{1,j}$, as is readily observed from Jensen's inequality. It is easy to see that (3) can easily be generalized to the case of multiple macro-arms as follows:

$$\hat{g}_{k,j}(s) := \frac{\sum_{m=1}^{s} \mathbf{1}\left[i_m = (k,j)\right] \cdot \mathbf{1}\left[j \geq \tau_{k,m}\right] \cdot V_m}{\sum_{m=1}^{s} \mathbf{1}\left[i_m = (k,j)\right] \cdot \min\{j, \tau_{k,m}\}}. \quad (4)$$

5 Wait-UCB

Our approach to solving F2A problems is to develop an upper confidence bound-style algorithm based on the reward per round estimate from (4). To achieve this, we develop a concentration inequality for how much higher the mean expected per-round reward may exceed this estimate. The following lemma gives us the concentration inequality for our quantity of interest.

Lemma 1 *Let X, Y be (possibly dependent) random variables with joint distribution P. Consider a sample $(X_1, Y_1), \ldots, (X_n, Y_n)$ of independent copies of $(X, Y) \sim P$. Assume that X takes values in $[0, 1]$ and Y takes values in $[1, B]$. Define $\mu_Y := \mathsf{E}[Y]$ and let $\hat{X}$ denote the sample mean of $X_1, \ldots, X_n$ (likewise for $\hat{Y}$ and $Y_1, \ldots, Y_n$). For any choice of $\delta \in [0, 1]$, we have with probability at least $1 - \delta$ over the sample,*

$$\left| \frac{\hat{X}}{\hat{Y}} - \frac{\mathsf{E}[X]}{\mathsf{E}[Y]} \right| \leq \sqrt{\frac{(B-1)\log\frac{4}{\delta}}{2n}} + \frac{2(B-1)\log\frac{4}{\delta}}{3\mu_Y n} + \sqrt{\frac{\log\frac{4}{\delta}}{2n}}.$$

A proof of this lemma can be found in Appendix B.

With Lemmas 1 and (4) in hand, we now introduce the new algorithm, Algorithm 1. Since the arms correspond to waiting times, we call this algorithm "Wait-UCB".[5] From Lemma 1 and our later results in Section 6.1, for any arm pair (k, j) and non-negative integers s and n, the upper deviation around $\hat{g}_{k,j}(s)$ is given by

$$a_{k,j}(s) = \alpha_j \frac{\log s}{N_{k,j}(s)} + \beta_j \sqrt{\frac{\log s}{N_{k,j}(s)}},$$

where $\alpha_j := \frac{8(j-1)}{3}$ and $\beta_j := \sqrt{2}(\sqrt{j-1} + 1)$ are constants independent of the macro-arms and $N_{k,j}(s)$ is the number of pulls of arm pair (k, j) at the end of epoch s.

Let us summarize the main idea behind the algorithm. Before an arm is pulled at least once, all the upper confidence bounds are initialized to infinity. This forces us to explore each arm at least once. After this phase, the arm that has the highest upper confidence bound is played. As an arm is played often, from the concentration inequality given in Lemma 1, our estimate of the ratio of reward to resource consumed by the arm becomes sharper and we converge to the optimal arm.

[5]Also, "wait" is a homophone for "weight": the upper deviation terms are weighted by the waiting-time-dependent quantities α_j and β_j introduced below.

Algorithm 1: Wait-UCB Algorithm

```
Input: time budget $T$, maximum waiting time $D$
$\alpha_j \leftarrow \frac{8(j-1)}{3}, \beta_j \leftarrow \sqrt{2}(\sqrt{j-1}+1)$ for all $j \in [D]$
$N_{k,j}(0) = 0$ for all $(k,j) \in [K] \times [D]$
$s = 1$
while $T > 0$ do
    Pull $i_s \leftarrow \operatorname{argmax}_{(k,j)\in[K]\times[D]} \hat{g}_{k,j}(s-1) + a_{k,j}(s-1)$
     // see (4) for $\hat{g}_{k,j}(s-1)$
    $T \leftarrow T - \min\{j_s, \tau_s\}$, where $i_s = (k_s, j_s)$
    $N_{(k,j)}(s) \leftarrow N_{(k,j)}(s-1) + \mathbf{1}[i_s = (k,j)]\ \forall (k,j)$
    $s \leftarrow s + 1$
```

6 Expected regret of Wait-UCB

We begin bounding the regret of Wait-UCB by bounding the number of times a suboptimal arm is pulled.

6.1 Bound on expected number of pulls

Lemma 2 *For any sub-optimal arm pair (k, j): $\Delta_{k,j} := g_{k^*,j^*} - g_{k,j} > 0$, the expected number of pulls in L epochs is given by*

$$\mathsf{E}[N_{k,j}(L)] \leq \log(T) \left(\frac{\beta_j + \sqrt{\beta_j^2 + 2\alpha_j \Delta_{k,j}}}{\Delta_{k,j}} \right)^2 + \frac{4\pi^2}{3}.$$

A proof of this lemma can be found in Appendix C.

Lemma 3 *Let $l = \log(T) \left(\frac{\beta_j + \sqrt{\beta_j^2 + 2\alpha_j \Delta_{k,j}}}{\Delta_{k,j}} \right)^2$. Then for any epoch s such that $l \leq s \leq L$,*

$$\Pr\left(|g_{k,j} - \hat{g}_{k,j}(s)| \geq \epsilon\right) \leq 4s^{-4},$$

where $\epsilon = \alpha_j \frac{\log s}{N_{k,j}(s)} + \beta_j \sqrt{\frac{\log s}{N_{k,j}(s)}}$.

The proof of the above lemma (see Appendix D) is based on tuning the values of α_j and β_j in the algorithm. It follows by a heavy sequence of algebraic steps and is omitted from the main text for brevity.

6.2 Regret bound

Theorem 2 *Wait-UCB's pseudo-regret is at most*

$$\sum_{(k,j)\mid\Delta_{k,j}>0} \mu_{k,j}^{(c)} \left(\frac{\left(\beta_j + \sqrt{\beta_j^2 + 2\Delta_{k,j}\alpha_j}\right)^2 \log T}{\Delta_{k,j}} + \mathcal{O}(1) \right),$$

where we define the mean waiting time $\mu^{(c)}_{k,j} := \mathsf{E}[c^{(1)}_{k,j}]$.

The $\mathcal{O}(1)$ only hides moderate constants and scales as $\Delta_{k,j} \leq 1$. The leading term (involving $\log T$) in the above bound is less explicit due to the notation; a coarser version of the leading term is

$$\mathcal{O}\left(\sum_{(k,j)\mid\Delta_{k,j}>0} \mu^{(c)}_{k,j}\left(\frac{j\log T}{\Delta_{k,j}}\right)\right). \tag{5}$$

In comparing this result to previous regret bounds of Flajolet and Jaillet (2017), Xia et al. (2016), and Ding et al. (2013) for UCB-Simplex, Budget-UCB, and UCB-BV1 respectively, we focus on the case of a single macro-arm ($K = 1$), as this is enough to capture the "waiting" aspect of the problem. Let us assume that all suboptimal arms have gap lower bounded by $\Delta > 0$. Then since all the j are at most D, the leading term (5) in our regret bound is of order at most

$$\left(\frac{D^2}{\Delta}\bar{\mu}^{(c)}\right)\log T, \tag{WAIT}$$

where we define $\bar{\mu}^{(c)} := \frac{1}{D}\sum_j \mu^{(c)}_{1,j}$. Now, in the worst case (when mean waiting times are high), this term becomes $(D^3/\Delta)\log T$. However, for easier problems where the mean waiting time for most arms is much smaller than D, the term improves to $(D^2/\Delta)\log T$. Before comparing this result to the regret bounds of Flajolet and Jaillet (2017), Xia et al. (2016), and Ding et al. (2013), it is important to note that those works have stochastic resource consumptions lying in $[0, 1]$. Therefore, to view the F2A framework in their setting, we rescale our consumptions from $[D]$ to $\{\frac{1}{D}, \frac{2}{D}, \dots, 1\}$. Consequently, for each gap $\Delta_{1,j}$ in our paper, the corresponding gap in their paper will be scaled up by D.

With this conversion in mind, we turn our attention to Corollary 1 of Ding et al. (2013). After some unpacking, one can see that in an F2A problem, the leading term of their regret bound is of order

$$\left(D^4 + \frac{D^3}{\Delta} + \frac{D^2}{\Delta^2}\right)\frac{\mu^{(r)}_{1,j^*}}{\mu^{(c)}_{1,j^*}}\log T. \tag{DQZL}$$

In the situation where the optimal arm's mean waiting time $\mu^{(c)}_{1,j^*} \in [1, D]$ is small, their bound is noticeably worse: the first term is quartic in D, the second term matches our worst-case bound, and the third term is quadratic in $(1/\Delta)$. Yet, when the mean waiting time for the optimal arm is large, their bound becomes closer to the behavior of our bound were $\bar{\mu}^{(c)}$ to be small. In either case, their bound grows as $\frac{1}{\Delta^2}$.

Next, we compare our regret bound to a regret bound of Flajolet and Jaillet (2017) for UCB-Simplex. After converting their notation to ours, their Theorem 1 gives regret that is of order

$$\left(\frac{D^2}{\Delta}\sum_j \frac{1}{\mu^{(c)}_{1,j}}\right)\log T + \mathcal{O}(D^3). \tag{FJ}$$

The leading terms in (FJ) versus (WAIT) are close but there is a important distinction. Both leading terms contain a common factor of $\frac{D^2}{\Delta}$. However, (WAIT) has the average of the mean waiting times, while (FJ) has the sum of the reciprocal mean waiting times. When the mean waiting times are small, the average is smaller. When the mean waiting times are larger, the sum of reciprocals is smaller. Each quantity has a range of $[1, D]$. However, in any case, the constant term in (FJ) can be of order D^3, whereas the constant term from Theorem 2 (not shown but visible in the proof) is *always* $\mathcal{O}(D^2)$. A similar comparison can also be made for Budget-UCB from Theorem 3 of Xia et al. (2016) which gives the same logarithmic leading term as in (FJ) when the budget is sufficiently large.

We posit that the reason for our regret bound's improvement over the other bounds (in some regimes) is that our analysis is quite different: we directly form an upper confidence bound for the ratio estimator, and this gives us an opportunity to leverage a Bernstein-style improvement (from Bernstein's inequality). It would be interesting to somehow combine the style of analysis used in this work and the style from one of the other works to get a bound that dominates in all regimes. However, based on the experimental results in Section 7, it might be that our regret bounds for the *existing* Wait-UCB algorithm could be improved.

7 Experiments

Our experiments focus on testing Wait-UCB in three different scenarios in the F2A framework. In all our experiments, we compare Wait-UCB to UCB-Simplex (Flajolet and Jaillet, 2017) and Budget-UCB (Xia et al., 2016); recall that these algorithms also admit logarithmic regret bounds in the BwK setting. Also, UCB-BV1 (Ding et al., 2013) was considered in all experiments, but it was later excluded as the other algorithms performed better. The delay τ_k and potential reward V_k are chosen such that they have a moderate minimum gap Δ. The delay distribution τ_k over $[D]$ for each experiment is given as a bar graph above the cumulative regret figures. Each experiment in this section is an average over 10 independent runs, and we use the same time budget of $T = 10^7$ rounds for all the experiments. The pseudo-regret in each experiment is calculated as

$$R_t = t \cdot g_{k^*,j^*} - \sum_{s=1}^{L_t} r^{(s)}_{i_s},$$

where L_t is the algorithm's last epoch for budget t.

7.1 One macro-arm and several micro-arms

We start by having just 1 macro arm with deterministic potential reward $V_{k,s} = 1$ and $D = 10$ micro-arms configured with τ_k such that the optimal arm falls in different intervals of $[D]$, allowing us to understand the algorithms' behavior for various τ_k.

In the first experiment, we chose τ_k such that the delay doubles with the rewards obtained (see the bar graph in Figure 1a). For instance, if an algorithm chooses to wait for only 1 round, it consumes less resources and so can play for more epochs; on the flip side, if it decides to wait slightly longer than two rounds, it has twice the chance of getting the reward. This delay distribution induces a minimal gap of $\Delta = 0.042$. The learning algorithm must navigate a tight trade-off to select the optimal waiting time. In the next experiment, we chose the delay τ_k such that the optimal arm lies in the middle of $[D]$ incurring a moderate gap of $\Delta = 0.124$. The final experiment in this section stems from the observation that α_j becomes zero for arm $j = 1$, and so to study the behavior of the algorithm when the optimal arm is 1, an appropriate delay τ_k that incurs a moderate gap of $\Delta = 0.166$ is chosen.

Figure 1 shows the performance of Wait-UCB for the configurations of delay distribution that were discussed before. These experiments demonstrate a few interesting insights. From Figures 1a and 1b, we can see that for the first two experiments, Wait-UCB performs much better than UCB Simplex and Budget-UCB. We believe that the main reason for this comes from the underlying principle on which UCB algorithms are developed, i.e., Optimism in the Face of Uncertainty. In our F2A framework, the uncertainty in getting a reward decreases if we choose to wait for longer time. This behavior is very well captured by Wait-UCB in the construction of the confidence radius (which grows with j), resulting in relatively quick convergence towards the best arm. Also, observe that α_j becomes 0 for arm $j = 1$, and in the last experiment (Figure 1c), Wait-UCB performs worse than the other algorithms; however, it still enjoys logarithmic regret.

7.2 Several macro-arms and one micro-arm

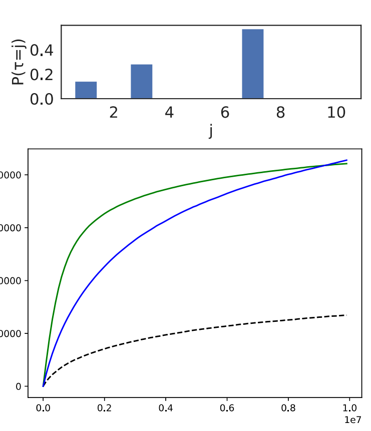
By having only one micro-arm ($D = 1$) and several macro-arms ($K > 1$), our problem reduces exactly to the standard MAB setting. Now, the algorithms simply need to pull the arm with highest V_k. A similar observation can be found in the setting when $D \geq 1$ and the delay distribution happens to be $P(\tau = 1) = 1$, i.e., any arm $j \in [D]$ will be the optimal arm as the epochs are completed in unit time (so the waiting time becomes insignificant here). Now, observe that the above two setups appear to be the same but are different in the perspective of algorithms which compute confidence radii for all the (k, j) pairs and select one among them. We decided to test the algorithms in the latter scenario with the following experimental setup. We took $K = 3$ and $D = 5$ with deterministic $\tau_k = 1$ and V_k being a Bernoulli random variable with success probabilities 0.5, 0.7, and 1 respectively.

Note that for this case, our upper confidence bound cleanly reduces to the upper confidence bound of UCB1 (Auer et al., 2002) for the standard stochastic multi-armed bandit problem, which is $a_{k,1}(s) = \sqrt{\frac{2 \log s}{N_{k,1}(s)}}$. Figure 2a shows the results for this setting. It is evident from the figure that Wait-UCB performs much better than UCB-Simplex and Budget-UCB. We think this might be mainly due to the fact that the exploration term in Wait-UCB scales at most by the number of micro-arms j whereas, for UCB Simplex, the exploration term scales with the total number of Macro-Micro arm pairs $k \cdot D$. Due to this large exploratory factor, it takes longer for UCB-Simplex to converge.

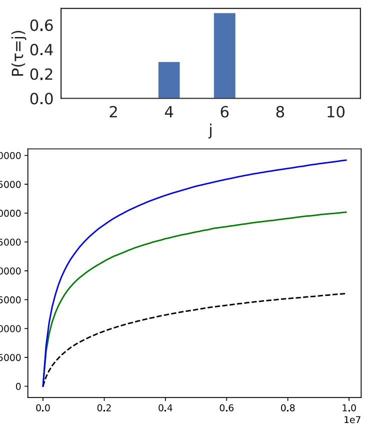
7.3 Several macro- and micro-arms

This setting is used in several of the real world applications discussed in Section 2.2. We performed a synthetic experiment inspired from computational advertising. We have a set of ads belonging to $K = 3$ categories with V_k their corresponding private utility for an impression (click) and $D = 5$ denotes the maximum waiting time for the ad before switching it. The delay distribution τ_k captures the time of impression of a user towards category k, which in a way reflects their interest on that category k. Note that showing a relevant ad affects the time of impression, thereby establishing an implicit connection between reward and delay. Now, the goal of the learning algorithm is to show the ad that best fits the interest of the user and learn the optimal waiting time for their impression.

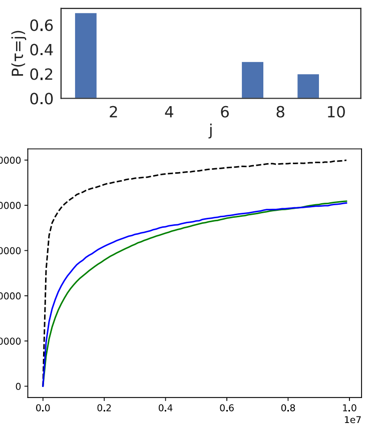
We consider two cases for our experiments. In Case I, we set V_k to be a Bernoulli random variable with success probabilities 0.7, 1, and 0.5 respectively. In Case II, we keep the same delay τ_k which represent the same user's interest but only change the V_k i.e., the publisher now receives different private utilities for different ad categories. The new Bernoulli random variable with success probabilities for the same categories are 1, 0.5, and 0.7 respectively. Figures 2b and 2c shows the learning behavior of Wait-UCB in these scenarios and the bar graphs on top represent the τ_k distribution for each of the 3 categories.



(a) Doubling Scenario: $j^* \gtrsim [D/2]$

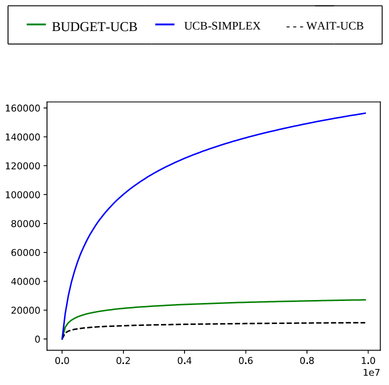


(b) $j^* \approx [D/2]$

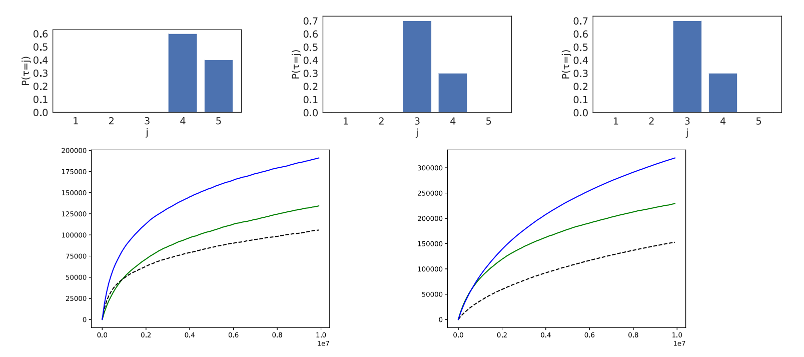


(c) $j^* = 1$

Figure 1: Cumulative Regret of Wait-UCB for $D = 10, K = 1, V_k = 1$



(a) for $D = 10, K = 1, V_k = 1$



(b) Case I: $D = 5, K = 3$

(c) Case II: $D = 5, K = 3$

Figure 2: Cumulative Regret of Wait-UCB

8 Conclusion and Future work

In this work, we introduced the Farewell to Arms framework of multi-armed bandit problems and presented a new algorithm, Wait-UCB, along with a logarithmic problem-dependent regret bound of order $O((D^3/\Delta)\log T)$. In the case of a single macro-arm and when most micro-arms have low mean waiting times (much smaller than D), our regret bound improves to $O((D^2/\Delta)\log T)$; in this regime, the leading $(\log T)$ term of our bound for Wait-UCB is smaller by a factor of D than the bounds of Ding et al. (2013), Xia et al. (2016), and Flajolet and Jaillet (2017) for UCB-BV1, Budget-UCB, and UCB-Simplex respectively. However, in the opposite regime, the leading term (but not the constant term) of the bound for UCB-Simplex and Budget-UCB can be smaller than ours. Yet, our experiments show that Wait-UCB empirically outperforms UCB-Simplex and Budget-UCB in a number of settings. Notably, in the standard bandit setting with only one arm, our algorithm collapses to be equal to a standard UCB algorithm, while this is not the case for the other algorithms. Indeed, our algorithm well-outperforms UCB-Simplex and Budget-UCB even in this simple setting.

In closing, we mention a few exciting directions for future work. First, it would be interesting (but challenging) to take into account that an F2A game potentially can have more feedback than mere bandit feedback. For example, if a learning agent decides to wait for 5 rounds, the learning agent also gains feedback for any shorter waiting time. This feedback could readily be used by Wait-UCB, but proving improved regret bounds in light of this feedback is highly challenging. A second, further improvement would be to completely utilize the structure of the F2A game, which also includes conditional feedback. For instance, observe that if the learning algorithm decides to wait for $j \geq \tau$ rounds (for delay τ), then the algorithm knows τ and hence also gets feedback for all the longer waiting times $j+1, \ldots, D$. Yet, if it does not wait for long enough (i.e., $j < \tau$), then this feedback will not be available.

Acknowledgements

We thank Vamsi Potluru for lengthy discussions on this work at an early stage. This work was supported by NSERC Discovery Grant RGPIN-2018-03942.

References

Agarwal, A., Foster, D. P., Hsu, D. J., Kakade, S. M., and Rakhlin, A. (2011). Stochastic convex optimization with bandit feedback. In *Advances in Neural Information Processing Systems*, pages 1035–1043.

Auer, P., Cesa-Bianchi, N., and Fischer, P. (2002). Finite-time analysis of the multiarmed bandit problem. *Machine Learning*, 47(2):235–256.

Badanidiyuru, A., Kleinberg, R., and Slivkins, A. (2013). Bandits with knapsacks. In *2013 IEEE 54th Annual Symposium on Foundations of Computer Science*, pages 207–216. IEEE.

Bernstein, S. (1934). Teoriia veroiatnostei [The theory of probabilities]. *Moskva–Leningrad: Gosudarstvennoe Tekhniko-Teoreticheskoe Izdatelstvo.[2nd augmented ed. The 3rd ed., of the same year, is identical. The 4th ed., augmented, appeared in 1946.].*

Blackwell, D. (1946). On an equation of Wald. *The Annals of Mathematical Statistics*, 17(1):84–87.

Ding, W., Qin, T., Zhang, X.-D., and Liu, T.-Y. (2013). Multi-armed bandit with budget constraint and variable costs. In *Twenty-Seventh AAAI Conference on Artificial Intelligence*.

Flajolet, A. and Jaillet, P. (2017). Logarithmic regret bounds for bandits with knapsacks. *arXiv preprint arXiv:1510.01800v4*.

Gal, S., Landsberger, M., and Nemirovski, A. (2007). Participation in auctions. *Games and Economic Behavior*, 60(1):75–103.

Grimmett, G., Grimmett, G. R., and Stirzaker, D. (2001). *Probability and random processes*. Oxford university press.

Joulani, P., Gyorgy, A., and Szepesvári, C. (2013). Online learning under delayed feedback. In *International Conference on Machine Learning*, pages 1453–1461.

Komiyama, J., Sato, I., and Nakagawa, H. (2013). Multi-armed bandit problem with lock-up periods. In *Asian Conference on Machine Learning*, pages 100–115.

Lai, T. L. and Robbins, H. (1985). Asymptotically efficient adaptive allocation rules. *Advances in Applied Mathematics*, 6(1):4–22.

Lattimore, T., György, A., and Szepesvári, C. (2014). On learning the optimal waiting time. In *International Conference on Algorithmic Learning Theory*, pages 200–214. Springer.

Li, L., Jamieson, K., DeSalvo, G., Rostamizadeh, A., and Talwalkar, A. (2017). Hyperband: A novel bandit-based approach to hyperparameter optimization. *The Journal of Machine Learning Research*, 18(1):6765–6816.

McAfee, R. P. and McMillan, J. (1987). Auctions with entry. *Economics Letters*, 23(4):343–347.

Samuelson, W. F. (1985). Competitive bidding with entry costs. *Economics letters*, 17(1-2):53–57.

Stegeman, M. (1996). Participation costs and efficient auctions. *Journal of Economic Theory*, 71(1):228–259.

Tran-Thanh, L., Chapman, A., Rogers, A., and Jennings, N. R. (2012). Knapsack based optimal policies for budget–limited multi–armed bandits. In *Twenty-Sixth AAAI Conference on Artificial Intelligence*.

Weed, J., Perchet, V., and Rigollet, P. (2016). Online learning in repeated auctions. In *Conference on Learning Theory*, pages 1562–1583.

Xia, Y., Ding, W., Zhang, X.-D., Yu, N., and Qin, T. (2016). Budgeted bandit problems with continuous random costs. In *Asian Conference on Machine Learning*, pages 317–332.