Forced-Exploration Based Algorithms for Playing in Stochastic Linear Bandits*

Yasin Abbasi-Yadkori
Department of Computing Science
University of Alberta

András Antos
MTA
SZTAKI

Csaba Szepesvári
Department of Computing Science
University of Alberta

Abstract

We study stochastic linear payoff bandit problems and give a simple, computationally efficient algorithm whose regret, under certain regularity assumptions on the action set, is $O(d\sqrt{T})$, where d is the dimensionality of the action space and T is the time-horizon. However, this result is problem dependent and not a minimax bound. We show that our algorithm is able to achieve lower regret bounds when we have sparsity in the problem. Our experimental results support our upper bound and show that the algorithm proposed might be competitive with alternative algorithms when the payoffs of the actions are correlated and when the parameters vector is sparse.

1 INTRODUCTION

In a stochastic bandit problem a decision maker repeatedly chooses actions from some action set $\mathbb{A}$ and receives a random payoff from the unknown distribution associated with the chosen action, the goal being to maximize the total expected payoff received, or equivalently to minimize the total expected regret, which is the expected loss compared to the oracle who plays the action with the maximal expected payoff in each round.

In many practical problems the number of arms is very large (if not infinite), but often some prior knowledge is available about the possible interdependence of the payoffs associated with the arms. For example, one may suspect that the payoffs of certain arms are close to each other as in the paper by Pandey et al. (2007). In this paper we consider such bandit problems with large or infinite action sets.

Let $r(a)$ denote the mean payoff when action $a \in \mathbb{A}$ is chosen by the decision maker ($r : \mathbb{A} \to \mathbb{R}$). One special case of bandits with dependent arms is when $\mathbb{A}$ is a subset of a Euclidean space, say $\mathbb{R}^d$, and r is linear: $r(a) = \theta_*^{\mathrm{T}} a$. This case was considered first by Auer (2002). He has given an algorithm based on the upper-confidence bound idea and has shown that in the case of finite action sets the (expected) regret of this algorithm at time T is bounded by $\tilde{O}((\log|\mathbb{A}|)^{3/2}\mathrm{poly}(d)\sqrt{T})$.[1]

More recently, Dani et al. (2008) has shown that another algorithm based on the upper-confidence bound idea (which was also proposed, but not analyzed by Auer (2002)) satisfies a regret bound of $O(d\sqrt{T}\log^{3/2} T)$. This algorithm, which is called the Confidence Ellipsoid Algorithm, at time step t constructs a confidence ellipsoid C_t around the current least-squares estimate of the parameter based on the observations available and then takes the action

$$A_t = \operatorname*{argmax}_{a\in\mathbb{A}} \max_{\theta\in C_t} \theta^{\mathrm{T}} a. \qquad (1)$$

Dani et al. (2008) discuss the efficiency of the Confidence Ellipsoid Algorithm and point out that the optimization problem (1) is NP-hard and so the Confidence Ellipsoid Algorithm is not practical for large values of d (see Appendix A.1 for some experimental results). Dani et al. (2008) also propose another algorithm that is computationally more efficient, but they only show a regret of $O(d^{3/2}\sqrt{T}\log^{3/2} T)$ for this algorithm.

In a recent and independent work, Rusmevichientong and Tsitsiklis (2009) propose an efficient algorithm based on the Forced Exploration method and show that their algorithm achieves a $O(d\sqrt{T})$ regret when the action set is strongly convex. In this paper we provide a similar and independently developed algorithm and show that it satisfies a regret bound of $O(d\sqrt{T})$ (the exact conditions will be given in Section 2). We show that the strong convexity assumption is not actually needed and $O(d\sqrt{T})$ bound holds in more general settings such as a polytope.

The forced exploration algorithm (in both papers) has the advantage of being (relatively) efficient and simple: In each time step the algorithm decides about if it should "explore or exploit" based on a fixed schedule. Up to time step T our algorithm will use $O(d\sqrt{T})$ exploration steps. When exploring, it selects an action from the action set according to some fixed distribution and updates its estimate of the parameter vector based

*This paper is based on the MS thesis of the first author.

[1] By convention, for the nonnegative sequences (a_n), (b_n) converging to $+\infty$, $a_n = \tilde{O}(b_n)$ means that $\limsup_{n\to\infty} a_n/(b_n \log(b_n)) < +\infty$.

on the received feedback. When exploiting it selects an action from $\mathbb{A}$ that maximizes $\theta_t^{\mathrm{T}} a$, where θ_t is the most recent parameter estimate. Note that after the exploiting steps the parameter is not updated.

In Section 3 we show how the basic algorithm can be modified to exploit the sparsity of the problem. When the parameters vector θ_* is p-sparse, we can improve the regret bound to $O(p\sqrt{T})$ by tuning the exploration rate. Our experiment on an ad allocation problem, where parameters vector has a sparse representation, supports this theoretical result.

Dani et al. (2008) prove a $\Omega(d\sqrt{T})$ lower bound for the stochastic linear bandit problem. This lower bound analysis is a bit technical. We have a short natural proof for the same lower bound that is not included in this abstract due to the lack of space.

2 THE PROBLEM AND THE ALGORITHM

Let $\mathbb{A} \subset \mathbb{R}^d$ be the set of actions available to the decision maker. We assume that $\mathbb{A}$ is a bounded, convex set.[2] Let A_t be the action chosen by the decision maker at time step t ($t = 1, 2, \ldots$). In a *stochastic bandit problem* upon testing action A_t the decision maker receives a real-valued random reward, R_t such that $\mathbb{E}[R_t | A_t, R_{t-1}, A_{t-1}, \ldots, R_1, A_1] = \mathbb{E}[R_t | A_t] = r(A_t)$ for some fixed function $r : \mathbb{A} \to \mathbb{R}$, where it is assumed that the decision maker can base her decisions on past information only. In other words, the reward at time t can be written in the form $R_t = r(A_t) + Z_t$, where (Z_t) is a martingale difference sequence. We will assume that (Z_t) is bounded with probability one. More specifically, without the loss of generality, we shall assume that $|Z_t| \leq 1$ with probability one. In a stochastic *parametric* bandit problem the reward function r belongs to a parametric family of functions: $r(a) = r(a; \theta_*)$ for some *parameter* $\theta_* \in \Theta \subset \mathbb{R}^d$ (i.e., with a slight abuse of notation $r : \mathbb{A} \times \Theta \to \mathbb{R}$). In a stochastic *linear* bandit problem $r(\cdot; \cdot)$ assumes a linear form: $r(a; \theta) = \theta^{\mathrm{T}} a$, where now $\theta \in \mathbb{R}^d$. In what follows we focus on linear bandit problems.

The algorithm, that we call FEL (Forced Exploration for Linear bandit problems), is shown in Table 1. The algorithm has two parameters: the increasing sequence (f_t^*) and P. The sequence (f_t^*) is assumed to be strictly increasing sublinear sequence converging to infinity (in particular, we will use $f_t^* = cd\sqrt{t}$ with some $c > 0$) and f_t^* specifies how many exploration steps are desirable in the first t time steps. The other parameter, P, denotes a probability distribution over $\mathbb{A}$. For the sake of simplicity, we assume that P is chosen such that if $A \sim P(\cdot)$ then the dispersion matrix $C = \mathbb{E}\left[AA^{\mathrm{T}}\right]$ is non-singular.

[2]The convexity of $\mathbb{A}$ can be assumed without the loss generality as will be explained later. Boundedness could be replaced by a technical condition at the price of considerable difficulties. Hence, for the sake of simplicity, we assume that $\mathbb{A}$ is bounded.

Let $C_0 := \mathbb{I}$, $y_0 := 0$, $\theta_0 := 0$, $f_0 := 0$
$\{C_0 \in \mathbb{R}^{d \times d}$, and $y_0, \theta_0 \in \mathbb{R}^d\}$
for $t := 1, 2, \ldots$ **do**
  if $f_{t-1} < f_t^*$ **then**
    {Exploration:}
    {Draw a random action from $\mathbb{A}$ according to distribution P}
    $A_t \sim P$
    Take A_t and receive payoff Y_t
    $C_t := C_{t-1} + A_t A_t^T$
    $y_t := y_{t-1} + Y_t A_t$
    $\theta_t := (\mathbb{I} + C_t)^{-1} y_t$
    $f_t := f_{t-1} + 1$
  else
    {Exploitation:}
    $A_t := \mathrm{argmax}_{a \in \mathbb{A}} \theta_{t-1}^T a$
    Take A_t and receive payoff Y_t
    $C_t := C_{t-1}$, $y_t := y_{t-1}$, $\theta_t := \theta_{t-1}$, $f_t := f_{t-1}$
  end if
end for

Table 1: FEL algorithm for stochastic linear bandit problems. Note that $\mathbb{I}$ denotes the identity matrix, making the algorithm estimate the unknown parameter using ridge regression. Note that f_t, unlike C_t and θ_t, is not random.

2.1 UPPER BOUND

The analysis of the regret of the algorithm will be done in a few steps. First, we provide bounds on the error of the parameter estimate. Next, assuming that the growth rate of loss function, ℓ introduced earlier can be bounded by a strictly quadratic function in a neighborhood of θ_*, we give a bound on the regret of the algorithm. This is followed by identifying a number of cases when the loss function indeed enjoys this property. We make the following assumptions:

Assumption A1 The noise sequence (Z_t) satisfies $\mathbb{E}[Z_t | A_t] = 0$, no matter how the action sequence (A_t) is chosen based on the past payoffs. Further, $|Z_t| \leq 1$ holds with probability one.

Assumption A2 The reward function is uniformly bounded and in particular $\|r\|_\infty \leq 1$.

The main idea of our regret bound is as follows: Let $a(\theta) = \mathrm{argmax}_{a \in \mathbb{A}} \theta^{\mathrm{T}} a$ and consider the loss function

$$\ell(\theta) = \theta_*^{\mathrm{T}} a_* - \theta_*^{\mathrm{T}} a(\theta), \tag{2}$$

where $a_* = a(\theta_*)$ is the best action. Note that this loss function gives the immediate expected regret of an algorithm that plays the best looking action given the parameter estimate θ. Clearly, $\ell(\theta_*) = 0$. Further, under various conditions, $\nabla \ell(\theta_*) = 0$. Hence, the function ℓ can be bounded from above by a quadratic surface around θ_*. This means that the price of an error ε in estimating θ in terms of the regret is ε^2. This in turn

allows one to work with relatively imprecise estimates of the parameter without incurring huge losses.

Assumption A3 The regret function, as defined by (2), satisfies

$$\ell(\theta) \le c \left\|\theta - \theta_*\right\|_2^2 + c' \left\|\theta - \theta_*\right\|_2^3,$$

for some $c, c' > 0$. In Section 2.2 we will show that this condition holds for a wide selection of choices of $\mathbb{A}$ and θ_*.

Assumption A4 The probability distribution P is such that if $A \sim P(.)$ then the matrix $\mathbb{E}\left[AA^T\right]$ is non-singular.

Assumption A5 There exists $B > 0$ such that for any $a \in \mathbb{A}$, $\left\|a\right\|_2 \le B$.

The following theorem shows that the expected value of the cumulative regret can be bounded by $O(d\sqrt{T})$:

Theorem 2.1 *Let Assumptions A1-A5 hold. Then the expected regret of FEL with $f_t^* = d\sqrt{t}$ up to time T satisfies*

$$\mathbb{E}\left[L(T)\right] \le O(d\sqrt{T}).$$

The proof of the above theorem and the exact constants can be found in Abbasi-Yadkori (2009).

Remark 2.2 *The above result is a problem dependent bound and not a uniform bound over all problems. The uniform rate for algorithms of this type have a $T^{2/3}$ dependence on the horizon. The implication is that for restricted classes (e.g. sphere) the algorithm might be competitive, but the larger the class is, the more tuning can be expected.*

2.2 RESULTS FOR VARIOUS ACTION SETS

In this section we consider some cases when the action set $\mathbb{A}$ is such that the loss function will satisfy the growth assumption A3.

2.2.1 Strictly convex action sets

Let us assume that the action set is the 0-level set of some strictly convex, sufficiently smooth function, $c : \mathbb{R}^d \to \mathbb{R}$:

$$\mathbb{A} = \{a \in \mathbb{R}^d \,:\, c(a) \le 0\,\}. \tag{3}$$

Note that $a(\theta)$ is the solution of the following constrained, parametric linear optimization problem:

$$\begin{aligned} -\theta^T a &\to \min \\ \text{s.t. } & c(a) \le 0. \end{aligned} \tag{4}$$

Since the linear objective function is unbounded on $\mathbb{R}^d$, the solution always lies on the boundary of the set $\mathbb{A}$, i.e., $c(a(\theta)) = 0$ holds for any θ. We make the following assumption:

Assumption A6 The function c is four-times differentiable in a neighborhood of θ_* and if $\lambda(\theta)$ denotes the the Langrange multiplier underlying the solution of the optimization problem (4) when the parameter is θ then $\lambda(\theta)$ is differentiable in the same neighborhood[3].

Then with the help of the Karush-Kuhn-Tucker (KKT) Theorem and the Implicit Function Theorem we get the following result (Abbasi-Yadkori, 2009):

Theorem 2.3 *Assume that the action set is given by* (3)*, where c is a function that is strictly convex. Further, let Assumption A6 hold. Then the regret function r satisfies Assumption A3.*

Note that if $\mathbb{A}$ is a sphere, or more generally an ellipsoid then $\mathbb{A}$ will satisfy the conditions of this theorem. In particular, when $\mathbb{A}$ is the unit sphere and the length of θ_* is one, $r(\theta) = \left\|(\theta/\left\|\theta\right\|_2) - \theta_*\right\|_2^2$, which can be used to show that in Assumption A3 in this case $c = 1$ can be chosen to be independent of d.

We suspect that the above result holds for very general action sets. In particular, it is not very difficult to see that the statement continues to hold when the set is described by a number of convex constraints which are all active in a neighborhood of a_*, such as in the example constructed for the lower bound proof in (Dani et al., 2008). One may believe based on the proof of this result that smoothness of $a(\cdot)$ is important. In the next section, we will look at the case when $a(\cdot)$ can have jumps, showing that smoothness is not essential. However, it remains for future work to fully characterize the cases when the subquadratic growth of r holds.

2.2.2 Polytopes

In this section we assume that the action set $\mathbb{A}$ is a polytope (an intersection of a finite number of half-spaces). In this case without the loss of generality one can define function $a(\cdot)$ such that its range is the vertex set of the polytope. Then $r(\theta_*)$ becomes a piecewise constant function. We want to show that it is constant in a small neighborhood of θ_*. Let F be the unique i-face of the polygon for the largest $i = 0, 1, \ldots, d-1$ that contains $a(\theta_*)$ and which is perpendicular to θ_*. (If there is no such i-face with $i \ge 1$ then we take F to be the vertex $a(\theta_*)$.) By an elementary argument it follows that there exist a neighborhood U of θ_* such that if $\theta \in U$ then $a(\theta)$ is on F. Since F is perpendicular to θ_*, for any $a, a' \in F$, $\theta_*^T a = \theta_*^T a'$. Hence, we have the following result:

Theorem 2.4 *Assume that $\mathbb{A}$ is a polytope. Then the regret function r is zero in a neighborhood of θ_* and thus satisfies Assumption A3.*

Note that if the action set is a polytope, the forcing schedule can be changed to e.g. $f_t = c\log^2(t)$, which by an argument similar to the above one, but

[3]Note that the differentiability of λ could be proven using arguments like in Chapter 12 of Nocedal and Wright (2006). These proofs are considerably technical and go beyond the scope of this abstract.

which exploits that the regret function is constant in a small neighborhood of θ_*, gives a regret bound of order $O(c \log^2 T)$ (the probability of choosing a suboptimal vertex in the exploitation step will decay at least as fast $1/T$, while the exploration steps contribute to the $\log^2 T$ growth of the regret). Note that in order to optimize the scaling of the regret with the dimension d, one should choose c to be proportional to $\sqrt{d}$. Clearly, with this approach any regret slightly faster than $\log(t)$ can be achieved at the price of increasing the transient period.

2.3 General Linear Payoff

Now, assume that the reward function takes the form: $h(a;\theta) = \theta^T \phi(a)$, i.e., the reward function takes the form of a general linear function (the function is linear in the parameters, but not in the actions). Here $\phi : \mathbb{A} \to \mathbb{R}^d$ and now the action space does not need to be a subset of a Euclidean space (or it could be a subset of a Euclidean space of dimension, say $s \neq d$). This case is interesting from the point of view of practical applications where the expected relatedness of the actions can be expressed with the help of some features ϕ.

In order to make a connection to the linear payoff case, assume that the decision maker chooses a random action A from some distribution $\mathcal{P}$. Then the expected immediate reward of this random action is $\mathbb{E}[h(A;\theta)] = \theta^T \mathbb{E}[a]$. If $\mathcal{P} = \sum_k p_k \delta_{a_k}$ then $\mathbb{E}[h(A;\theta)] = \theta^T \sum_k p_k \phi(a_k)$. Hence, if one defines $\tilde{\mathbb{A}}$ as the convex hull of $\mathbb{A}$ then we can view the problem as one defined with action set $\tilde{\mathbb{A}}$ and with a linear reward function $\bar{h}(\tilde{a};\theta) = \theta^T \tilde{a}$. Thus, we can apply Algorithm 1 to this problem. Note that the optimization problem $\operatorname{argmax}_{\tilde{a} \in \tilde{\mathbb{A}}} \theta^T \tilde{a}$ will have solutions on the boundary of $\tilde{\mathbb{A}}$. This means that in the exploitation steps, the algorithm can always select a non-randomized action.

If the action set $\mathbb{A}$ is finite, $\tilde{\mathbb{A}}$ becomes a polytope in which case the associated regret function satisfies Assumption A3. More generally, if this growth condition is satisfied, the algorithm's regret will be of order $d\sqrt{T}$ in time T, where d is the dimension of the *parameter space*. Thus, the dimension (or cardinality) of the action space does not play a direct role in the regret of the algorithm (as expected). Of course, these remarks apply to any linear bandit algorithm whose regret can be bounded in terms of the dimension of the parameter space.

3 EXPLOITING SPARSITY

In this section, we show that the FEL Algorithm is able to exploit the sparsity in the problem.

Lets collect the actions that we have tried in a matrix A_t and the corresponding rewards in a vector R_t (A_t has dimension $t \times d$, R_t has dimension $n \times 1$) and solve

$$\min_{\theta} \|\theta\|_1 \text{ s.t.} \quad (5)$$
$$\|A_t \theta - R_t\|_2 \leq K,$$

where K is the assumed bound on the noise. This means we use (5) instead of the ridge regression estimate in each exploitation step of the FEL Algorithm. Typical results in compressive sensing literature is as follows (Candes and Plan, 2007): Assume that θ_* has p nonzero entries out of d entries, where $p \ll d$. Then when the rows of A_t are selected say from a Gaussian and $t \sim p \log(p)$ then the solution to (5) will be "close" to θ_*:

$$\|\theta - \theta_*\|_2^2 \leq C K^2,$$

with some $C > 0$.

It does not seem too difficult to generalize this result to the case when we make $t = Mp \log(p)$ measurements and to get

$$\|\theta - \theta_*\|_2^2 \leq C \frac{K^2}{M}.$$

Since $M = t/(p \log p)$, we would then get that with t measurements, if θ_* is p-sparse, the solution to the above problem satisfies

$$\|\theta - \theta_*\|_2^2 \leq C p \frac{K^2}{t}.$$

As a consequence, one can follow the argument in Section 2.2 of (Abbasi-Yadkori, 2009) and show that with the exploration rate of $f_t^* = p\sqrt{t}$, our cumulative regret will scale with $O(p\sqrt{t})$ instead of $O(d\sqrt{t})$. Our experiments in Appendix A.2 support this result. Note that p is a tuning parameter here and its automatic tuning is an open question.

References

Abbasi-Yadkori, Y. (2009). Forced-exploration based algorithms for playing in bandits with large action sets. Master's thesis, Department of Computing Science, University of Alberta.

Auer, P. (2002). Using confidence bounds for exploitation-exploration trade-offs. *Journal of Machine Learning Research*, 3:397–422.

Auer, P., Cesa-Bianchi, N., and Fischer, P. (2002). Finite time analysis of the multiarmed bandit problem. *Machine Learning*, 47(2-3):235–256.

Candes, E. J. and Plan, Y. (2007). Near-ideal model selection by l1 minimization. *To appear in Annals of Statistics*.

Dani, V., Hayes, T., and Kakade, S. (2008). Stochastic linear optimization under bandit feedback. *COLT-2008*, pages 355–366.

Nocedal, J. and Wright, S. (2006). *Numerical Optimization*. Springer.

Pandey, S., Chakrabarti, D., and Agarwal, D. (2007). Multi-armed bandit problems with dependent arms. In *ICML-2007*, pages 721–728.

Rusmevichientong, P. and Tsitsiklis, J. N. (2009). Linearly parameterized bandits. (submitted).

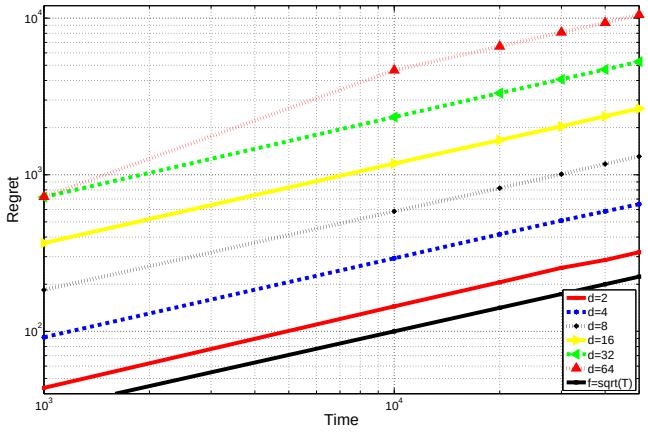


Figure 1: The total regret of FEL-U as a function of the time. For more explanation see the text.

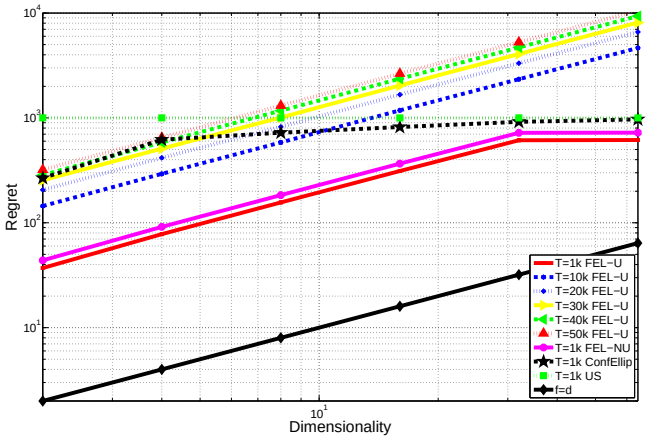


Figure 2: The total regret as a function of the dimensionality of the parameter vector. For more explanation see the text.

A EXPERIMENTS

This section has two parts. First we confirm that the regret of FEL is in the order of $O(d\sqrt{T})$ where d is the size of the parameter vector and T is time. Then we apply FEL to the ad allocation problem and show that it outperforms Confidence Ellipsoid of Dani et al. (2008) and algorithms proposed by Pandey et al. (2007) for this problem.

A.1 Scaling with d and T

Assume that d is even. Let $\mathbb{A}_d$ be the Cartesian product of $d/2$ circles, $\mathbb{A}_d = \{(a_1, \ldots, a_d) : \ a_1^2 + a_2^2 = a_3^2 + a_4^2 = \cdots = a_{d-1}^2 + a_d^2 = 1\}$. In this section, we empirically show that the regret of FEL on this problem scales according to $O(d\sqrt{T})$.

We run FEL for 1000 timesteps with $d = [2, 4, 8, 16, 32, 64]$ and repeat this experiment for 5 times. The value of the optimal action is equal to 1 in all experiments. The zero-mean noise is normally distributed with standard deviation equal to 0.1 (this noise does not satisfy the boundedness assumptions of our results, but our results in fact can be extended to this case). Figures 1 and 2 confirm that the regret of FEL scales linearly with $\sqrt{T}$ and d (black line, labeled as $r = d$, shows the linear behavior).

In Section 2, we analyzed FEL when it uses only exploration information to estimate the parameter vector. In Figure 2, *FEL-U* refers to FEL when it uses all information, *FEL-NU* refers to FEL when it uses only the information gathered during exploration steps and *ConfEllip* refers to the Confidence Ellipsoid Algorithm. As we can see in Figure 2, the performance of *FEL-U* and *FEL-NU* are almost identical in this problem. We have multiplied *FEL-U* with a small constant to make them distinguishable. So we only report the results for *FEL-U*, which gives slightly better results. The results for *FEL-NU* are reported only for $T = 1000$ and denoted by $T = 1000$, *FEL-NU* in Figure 2.

We have two interesting observations in Figure 2: 1) for $T = 1000$, the regret of FEL becomes almost constant for big values of d. Generally, for this class of problems it holds that (not shown here) if t is fixed and $d \to \infty$, then the regret becomes ct with some $c > 0$ that depends on the problem class. 2) The regret of *ConfEllip* is almost constant with respect to d. Actually *ConfEllip* is still in its transient mode and its regret is ct. The regret of the uniform sampling, referred to as *US* , is also included in Figure 2. We observe that the regret of *ConfEllip* is converging to the regret of *US* as $d \to \infty$. Based on our observations (again not shown here), it takes a very long time for Confidence Ellipsoid to exit from its transient mode in this problem (something in the order of $T = 500,000$).

A.2 The Ad Allocation Problem

In the ad allocation problem, a website is provided with a number of ads. At each time step, the website chooses an ad to display. The website gets paid by each user-click. The objective is to maximize the number of user-clicks.

Since the number of ads is usually very large, we can not directly apply a K-armed bandit algorithm such as the popular UCB1 of Auer et al. (2002) to this problem. Fortunately, there are correlations between the ads and we exploit this property to achieve better performance.

The problem setting, borrowed from Pandey et al. (2007) is as follows: We have C clusters indexed by $i \in \{1, \ldots, C\}$. Let us denote the optimal cluster (cluster containing the optimal action) by i_{opt} (we refer to ads as actions). Each suboptimal cluster contains N actions. The optimal cluster contains N_{opt} actions. The outcome of an action is 0 or 1 and is distributed according to a Bernoulli distribution. Let μ_{opt} be the expected payoff of the optimal action in the optimal cluster, μ^s be the payoff of the best action among other clusters, μ_i be the payoff of the best action in cluster i, and $\mu_{i,j}$ be the payoff of the jth action of cluster i. Define $\Delta = \mu_{opt} - \mu^s$ as the cluster separation. The cohesiveness of cluster i is defined as follows:

$$\delta_i = \frac{1}{N} \sum_{j=1}^{N} (\mu_i - \mu_{i,j}).$$

Further we let δ_{opt} denote the cohesiveness of the optimal cluster.

Following Pandey et al. (2007), we use $C = 10$ and $N = 10$ (i.e., the total number of actions is 100). The configuration is shown in Figure 3. For the suboptimal clusters, we use $\mu_i = 0.5$ and $\delta_i = 0.1$. The default values of the parameters of the optimal cluster are $N_{opt} = 10$, $\delta_{opt} = 0.3$, and $\mu_{opt} = 0.63$. The time horizon is 12000 and we repeat each experiment 200 times.

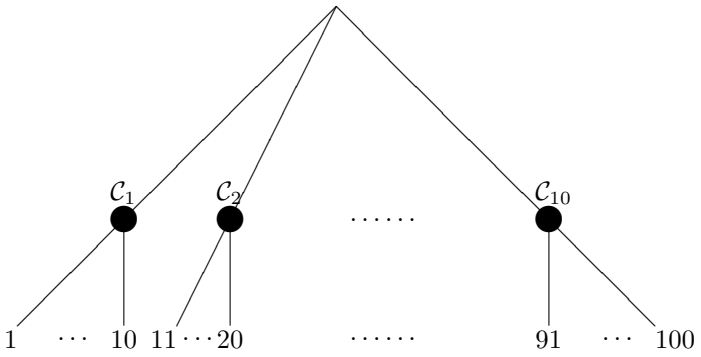


Figure 3: Each cluster contains 10 actions.

Pandey et al. (2007) simply put ads in clusters and use a two-stage UCB1 algorithm. In the first stage, they choose the cluster and in the second stage, they choose the ad. itself. Pandey et al. (2007) show that this method substantially outperforms UCB1.

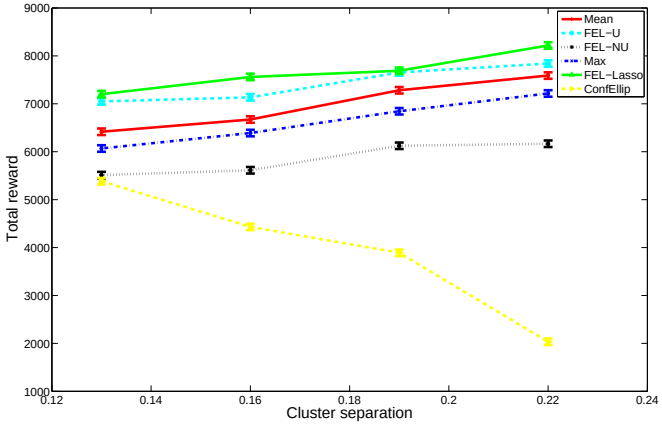
We use one basis for each cluster and one for each action. For example, if we had 10 clusters and 10 actions per cluster, we will have 110 bases (so $d = 110$). Denote by $\mathcal{C}_j \subset \mathbb{A}$ the set of actions belonging to cluster j ($1 \le j \le 10$) and let $\mathbb{A} = \{1, 2, \ldots, 100\}$. Then the feature vector for each action is a vector of length 110 with two ones and 108 zeros: $\phi_i(a) = \mathbb{I}_{\{i=a, i\le 100\}} + \mathbb{I}_{\{a\in\mathcal{C}_{i-100}, i>100\}}$. Like UCB1, we start running the algorithm by taking each action once. We expect 10+(a small number) of the elements of the parameter vector be around 1 and the rest of them be almost 0. Hence, using the results of Section 3, we execute FEL with the exploration rate of $\approx 10\sqrt{t}$.

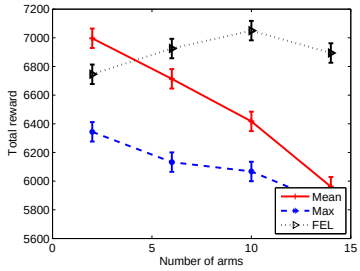
Figure 4 summarizes our findings. Algorithms *Mean* and *Max* are introduced by Pandey et al. (2007). *FEL-U* refers to FEL when we are using all information and FEL-NU refers to FEL when we use only exploration information. *FEL-Lasso* refers to the FEL Algorithm when it uses a Lasso-like penalty term in estimating the parameter vector. Finally, *ConfEllip* refers to the Confidence Ellipsoid Algorithm of Dani et al. (2008). Due to time constraints, we repeated each experiment of *ConfEllip* for 5 times and each experiment of *FEL-Lasso* for only one time.

Figure 4 compares the total reward of these algorithms as (a) the cluster separation changes, (b) the number of actions in the optimal cluster changes, and (c) $(1 - \delta_{opt})$ changes. As Figure 4 confirms, *FEL-U* outperforms both algorithms proposed by Pandey et al. (2007) in all three experiments under almost all conditions tested. The numbers that we are reporting in these figures for *Mean* and *Max* are slightly different (lower) than those that we see in (Pandey et al., 2007). We suspect that this might be due to different implementations of the underlying UCB1 algorithm.

Further, by comparing the performance of *FEL-U* and *FEL-NU*, we observe that there is a huge advantage in using the information gathered during the exploitation phases.

Another interesting observation is the poor performance of *ConfEllip*. We explain this observation by noting that we have 110 parameters and 100 actions. So it is expected that *ConfEllip* can't outperform simple UCB1 in this problem. The numbers that Pandey et al. (2007) report for UCB1 are higher that our results for *ConfEllip*. Having said that, one strength of the FEL Algorithm comes from the implicit regularization that is happening because C_t is initialized with





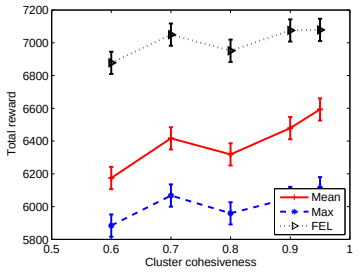


Figure 4: Total reward as a function of (a) Cluster separation (Δ); (b) Number of actions in the optimal cluster (N_{opt}); (c) Optimal cluster cohesiveness $(1 - \delta_{opt})$. 95 percent confidence bands are provided. *Features* is outperforming both algorithms proposed by Pandey et al. (2007) in all three domains under almost all conditions tested.

$\mathbb{I}$.

Finally, we observe that *FEL-Lasso* outperforms all alternative algorithms. We attribute this performance to the fact that in this problem, the L_1 norm of the parameter vector is small (because only a few coefficients are significantly different from zero). This situation is particularly suitable for a method like Lasso.