# Efficient algorithms for online decision problems

Kalai & and Vempala, 2005

*Paper review by* Kartik Bharadwaj CS20S020

April 16, 2021

## 1 Introduction

In online decision problem, one has to make a sequence of decisions without knowledge of the future. At each time step, we have to pay a cost based on the decision made and state observed. The paper proposes Follow-the-perturbed-leader (FPL), a randomized Follow-the-leader (FTL) algorithm. Moreover, the paper shows that FPL style algorithm extends naturally to a large class of structured online problem and are more efficient than modern exponential weighting algorithms such as Weighted Majority.

## 2 Problem Formulation

In the online decision problem, a series of decisions $d_1, d_2, \ldots$, are made at each time step from a possibly infinite set $\mathcal{D} \subset \mathbb{R}^n$. At time step $t$, state $s_t \in \mathcal{S} \subset \mathbb{R}^n$ is observed. Hence, the cost of making decision $d_t$ is $d_t \cdot s_t$, thereby, giving us the total cost is $\sum d_t \cdot s_t$. Therefore, the goal is:

$$\textbf{Goal} : \min \left( \sum d_t \cdot s_t - \min_{d \in \mathcal{D}} \sum d \cdot s_t \right)$$

Since costs are additive, let $\mathbf{M} : \mathbb{R}^n \to \mathcal{D}$, be a function of total state vectors that computes the best single decision in hindsight. Hence:

$$\mathbf{M}(s) = \arg\min_{d \in \mathcal{D}} d \cdot s$$

## 3 Approach used & Regret bounds

We know FTL is a deterministic algorithm that suffers regret $\mathcal{O}(T)$ for linear adversary. This means any deterministic algorithm can be forced to suffer $\mathcal{O}(T)$ regret. Given $n$ experts, the best expert incurs cost of at most $\frac{t}{n}$.

To mitigate this issue, the paper proposes to incorporate randomness in the decisions we make. On each period $t$, two ways of adding perturbations in Follow-the-perturbed-leader approach are as follows:

1. **FPL**($\epsilon$) or additive perturbation:

   $\rightarrow$ $\mathbf{p}_t \sim$ Uniform$([0, \frac{1}{\epsilon}]^N)$

   $\rightarrow$ $\mathbf{d}_t = \arg\min_{d \in \mathcal{D}} \sum_{i=1}^{t-1} \mathbf{d} \cdot \mathbf{s}_i + \mathbf{d} \cdot \mathbf{p}_t$
   $= \mathbf{M}(\mathbf{s_{1:t-1}} + \mathbf{p_t})$

2. **FPL***($\epsilon$) or multiplicative perturbation:

   $\rightarrow$ $\mathbf{p}_t \sim$ Laplacian with density $f(\mathbf{x}) = \frac{\epsilon}{2} e^{-\epsilon ||\mathbf{x}||_1}$.

   $\rightarrow$ $\mathbf{d}_t = \arg\min_{d \in \mathcal{D}} \sum_{i=1}^{t-1} \mathbf{d} \cdot \mathbf{s}_i + \mathbf{d} \cdot \mathbf{p}_t$
   $= \mathbf{M}(\mathbf{s_{1:t-1}} + \mathbf{p_t})$

## 4 Analysis

In all regret bounds being discussed in the paper, the authors are working with an oblivious adversary that does not adapt to learner's randomness. Hence, adding perturbation at time step $t = 0$ will suffice. Also, the authors assume loss functions to be linear.

The whole point of adding randomness is that it makes FTL almost same as BTL. In other words, adding perturbations makes the algorithm less predictable for the adversary. The expected difference between FPL and BPL is the number of times the chosen leader switches. For, FPL, this number is relative low. Note that FTL performed poorly because it switched leaders way too often.

The larger we set epsilon, the smaller $\mathcal{O}(\frac{\log n}{\epsilon})$, but the larger $\epsilon T$ becomes. The more randomness we added in the beginning, the smaller the RAT term is, and the less effect any individual loss function is going to have. However, if there is a large amount of randomness, BTPL is going to be far from BTL, leading to a worse choice seem better. This accounts for the $\frac{D}{\epsilon}$ term. Also, it will take the longer to adapt to a setting where one expert is clearly better than others.

Another interesting question which arises is the use of fresh randomness each period. If we did not use fresh randomness, i.e. our probability vector is the same for each time period $t$, an adaptive adversary could figure out what our perturbations and give us large regret by choosing cost vectors based on our previous decisions. Re-randomizing each period makes FPL have low regret against adaptive adversaries as well.

# 5 Contributions

## 5.1 Tree Update problem

The author has applied a version of the FPL algorithm to different online problems. For instance, in the *tree update problem*, one maintains a binary search tree over $n$ items in the face of an unknown sequence of look-ups to these items. For each lookup, the cost is the number of comparisons necessary to find the item, which, essentially, is its depth in the tree. If we maintain frequency counts for each item in the tree, and then before each access find the best tree based on these frequencies and perturbations, it would take $\mathcal{O}(n^2)$ time. For each lookup, these many computations and tree rotations is extreme for an online algorithm.

The authors, however, proposes Follow-the-lazy-leader (FLL), a version of FPL algorithm, which only needs $\mathcal{O}(\sqrt{T})$ updates. The only disadvantage of the lazy algorithms is that they only work against oblivious adversaries. Unlike FPL, which finds the optimal decision in hindsight every period, FLL takes advantage of the fact that we can correlate our perturbations for every period. FLL chooses those perturbations for which $s_{1:t-1} + p_t = s_{1:t} + p_{t+1}$. For such a scenario, we needn't compute $M(s_{1:t} + p_{t+1})$ as we will get the same result. The authors give an intuitive geometric proof on the probability of switching leaders to support their claims. This makes FLL equivalent to FPL in terms of expected cost.

## 5.2 Experts problem

FPL can be used to solve problems where number of experts is exponential in size of input. However, Hedge cannot handle infinitely many actions. FPL allows solving the experts problem in polynomial time. For instance, one can model the online shortest path as an experts problem and use FPL to get optimal bounds. Another example is that of tree-update problem we have seen above. This makes FPL computationally efficient.

# 6 Conclusion & Future work

While FPL (and its versions) give optimal regret bounds for specific online decision problems, these algorithms don't generalize among different noise distributions. Thus, they fail to provide intuitions on what core properties of the distributions lead to their optimal regret bounds. Future line of research could include analysis of techniques which generalize on the properties of distributions.

Moreover, there seems to be some inherent relationship between FTRL and FPL, which the paper hasn't discussed but would be an interesting line of research. Also, recent trends in ML research show relationship between FPL and differential privacy. This is another follow-up research arena one can explore.

Finally, the paper could have been better structured. For instance, Section 1 talks about problem formulation and its applicability in different online decision problems while proofs are in Section 4 and Section 5. Incorporating some structure into presenting the paper would have made it more readable.

## 7 References


1. Kalai & Vempala, 2005
2. Analysis of Perturbation Techniques in Online Learning, Thesis, Chansoo Lee
3. CS6781 Lecture 14 notes