Lecture 6. Prediction with Expert Advice

Advanced Optimization (Fall 2023)

Peng Zhao

zhaop@lamda.nju.edu.cn

Nanjing University

Outline

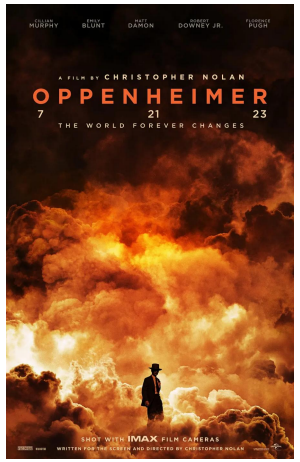
- Problem Setup
- Algorithms
- Connection to Online Convex Optimization

Motivation

- Consider that we are making predictions based on external experts.

A Chinese Odyssey Part Two - Cinderella

豆 9.2/10 | RT 87% | IMDb 7.8/10



Oppenheimer

豆 8.8/10 | RT 93% | IMDb 8.5/10

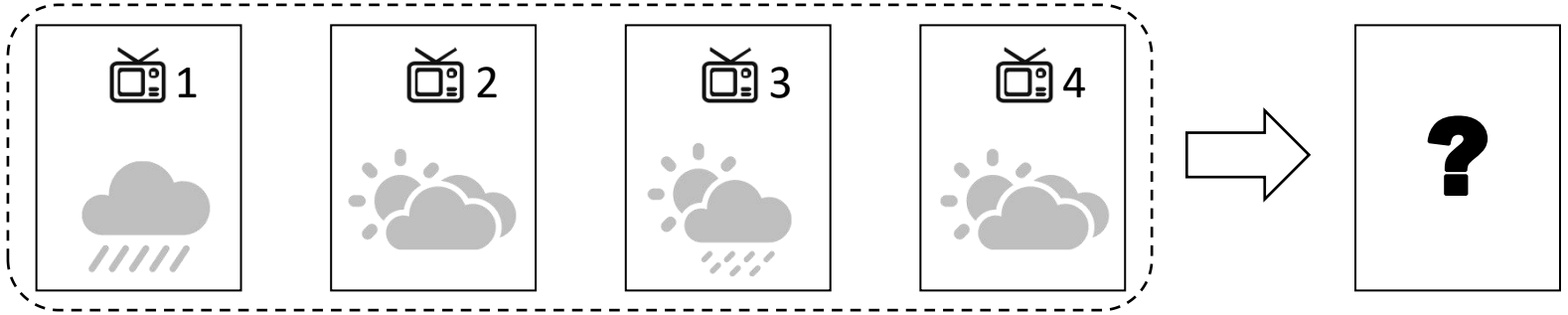
Titanic

豆 9.5/10 | RT 88% | IMDb 7.9/10

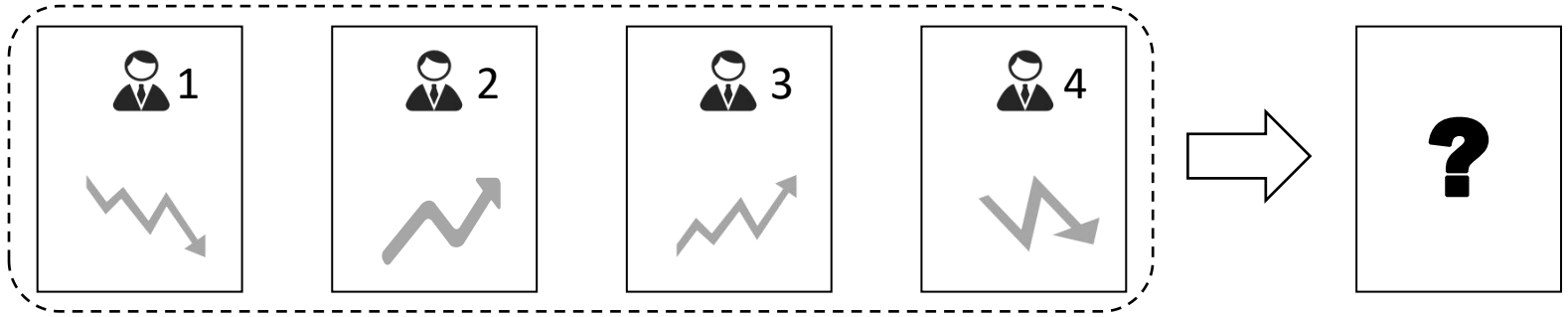
Prediction with Expert Advice

- Other examples include

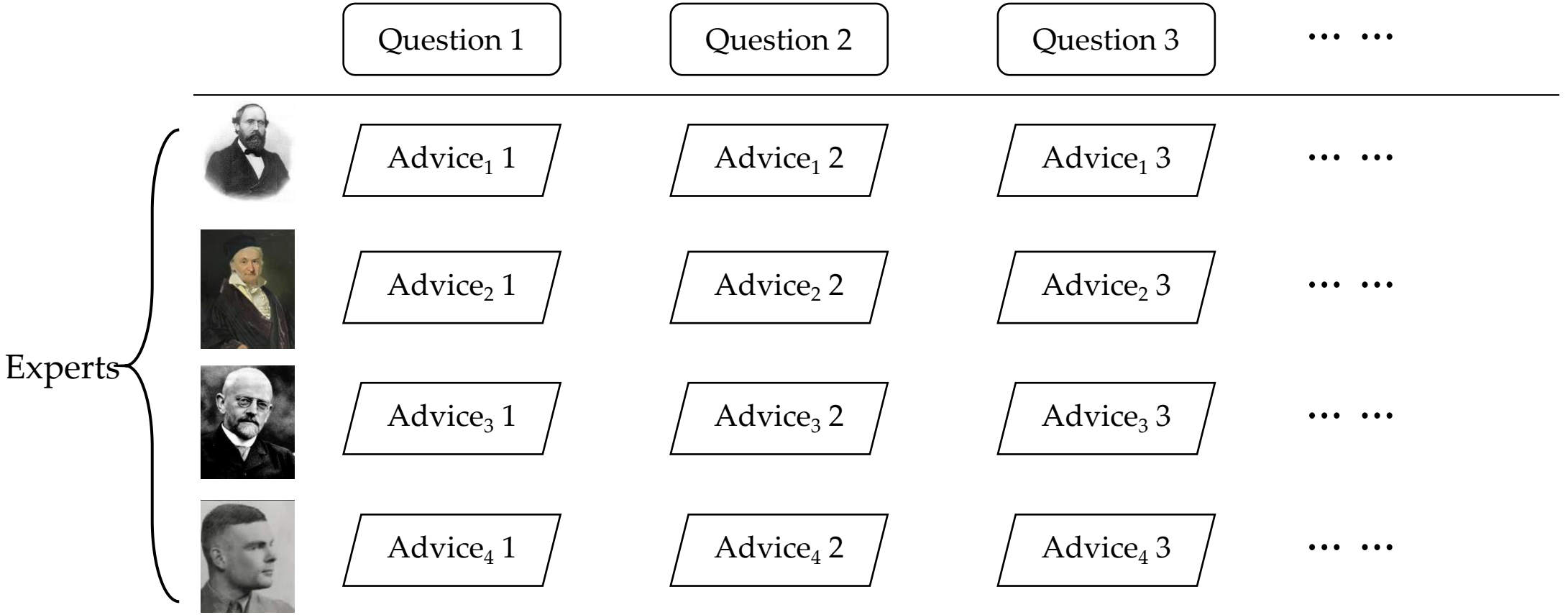
Weather report:



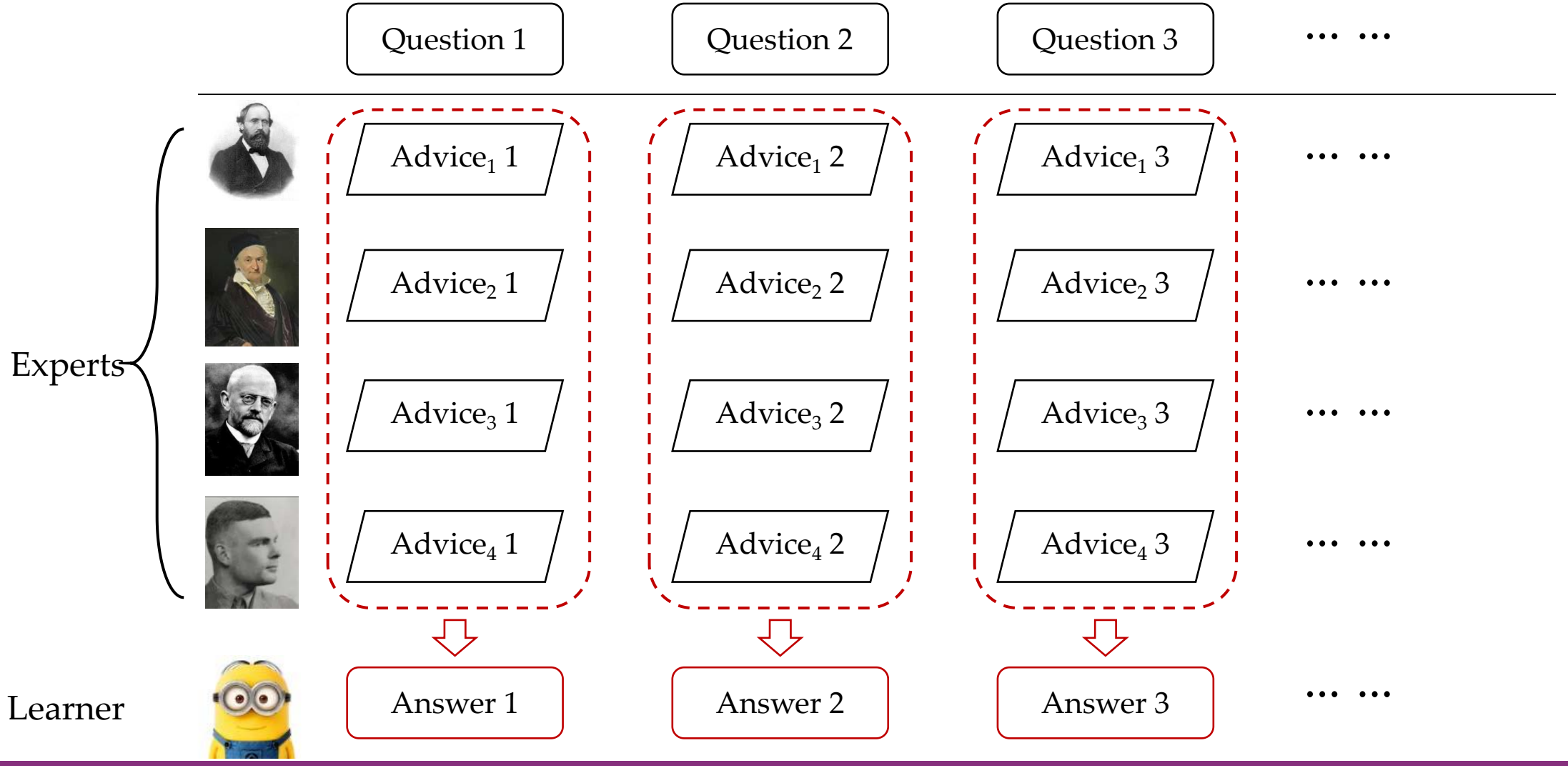
Stock prediction:



PEA Problem Setup



PEA Problem Setup



PEA: Formulization

- The online learner (player) aims to make the prediction based by combining N experts' advice.

At each round $t = 1, 2, \cdots$

(1) the player first picks a weight $\boldsymbol{p}_t$ from a simplex Δ_N;

(2) and simultaneously environments pick a loss vector $\boldsymbol{\ell}_t \in \mathbb{R}^N$;

(3) the player suffers loss $f_t(\boldsymbol{p}_t) \triangleq \langle \boldsymbol{p}_t, \boldsymbol{\ell}_t \rangle$, observes $\boldsymbol{\ell}_t$ and updates the model.

The feasible domain is the $(N-1)$-dim simplex $\Delta_N = \left\{ \boldsymbol{p} \in \mathbb{R}^N \mid p_i \geq 0, \sum_{i=1}^N p_i = 1 \right\}$.

We typically assume that $0 \leq \ell_{t,i} \leq 1$ holds for all $t \in [T]$ and $i \in [N]$.

PEA: Formulization

- The online learner (player) aims to make the prediction based by combining N experts' advice.

At each round $t = 1, 2, \cdots$

(1) the player first picks a weight $\boldsymbol{p}_t$ from a simplex Δ_N;

(2) and simultaneously environments pick a loss vector $\boldsymbol{\ell}_t \in \mathbb{R}^N$;

(3) the player suffers loss $f_t(\boldsymbol{p}_t) \triangleq \langle \boldsymbol{p}_t, \boldsymbol{\ell}_t \rangle$, observes $\boldsymbol{\ell}_t$ and updates the model.

- The goal is to minimize the regret with respect to the ***best expert***:

$$\text{Regret}_T \triangleq \sum_{t=1}^{T} \langle \boldsymbol{p}_t, \boldsymbol{\ell}_t \rangle - \min_{\boldsymbol{p} \in \Delta_N} \sum_{t=1}^{T} \langle \boldsymbol{p}, \boldsymbol{\ell}_t \rangle = \sum_{t=1}^{T} \langle \boldsymbol{p}_t, \boldsymbol{\ell}_t \rangle - \min_{i \in [N]} \sum_{t=1}^{T} \ell_{t,i}$$

A Natural Solution

- **Follow the Leader (FTL)**

  Select the expert that ***performs best so far***, specifically,

$$\boldsymbol{p}_t^{\text{FTL}} = \arg\min_{\boldsymbol{p}\in\Delta_N} \langle \boldsymbol{p}, L_{t-1}\rangle = \operatorname*{argmin}_{i\in[N]} L_{t-1,i}$$

  where $L_{t-1} \in \mathbb{R}^N$ is the cumulative loss vector with $L_{t-1,i} \triangleq \sum_{s=1}^{t-1} \ell_{s,i}$.

$\ell_{1,1} = 0.49$ ⇨ $\ell_{2,1} = 1$ ⇨ $\ell_{3,1} = 0$ ⋯ ⋯

$\ell_{1,2} = 0.51$ ⇨ $\ell_{2,2} = 0$ ⇨ $\ell_{3,2} = 1$ ⋯ ⋯

$$\text{Reg}_T = \sum_{t=1}^{T} \langle \boldsymbol{p}_t, \boldsymbol{\ell}_t\rangle - \min_{i\in[N]} \sum_{t=1}^{T} \ell_{t,i}$$

$$= T - \frac{T}{2} = \mathcal{O}(T)$$

FTL achieves ***linear regret*** in the worst case!

A Natural Solution

- **Follow the Leader (FTL)**

  Select the expert that ***performs best so far***, specifically,

$$\boldsymbol{p}_t^{\text{FTL}} = \arg\min_{\boldsymbol{p}\in\Delta_N} \langle \boldsymbol{p}, \boldsymbol{L}_{t-1} \rangle = \operatorname*{argmin}_{i\in[N]} L_{t-1,i}$$

  where $\boldsymbol{L}_{t-1} \in \mathbb{R}^N$ is the cumulative loss vector with $L_{t-1,i} \triangleq \sum_{s=1}^{t-1} \ell_{s,i}$.

⇨ Pitfall: decision is actually a one-hot vector, which can be very ***unstable***.

⇨ Replacing the 'max' operation in FTL by ***'softmax'***.

Hedge: Algorithm

- Hedge: replacing the 'max' operation in FTL by ***'softmax'***.

At each round $t = 1, 2, \cdots$

(1) compute $\boldsymbol{p}_t \in \Delta_N$ such that $\boldsymbol{p}_{t,i} \propto \exp\left(-\eta L_{t-1,i}\right)$ for $i \in [N]$

(2) the player submits $\boldsymbol{p}_t$, suffers loss $\langle \boldsymbol{p}_t, \boldsymbol{\ell}_t \rangle$, and observes loss $\boldsymbol{\ell}_t \in \mathbb{R}^N$

(3) update $\boldsymbol{L}_t = \boldsymbol{L}_{t-1} + \boldsymbol{\ell}_t$

FTL update

$$\boldsymbol{p}_t^{\text{FTL}} = \arg\max_{\boldsymbol{p} \in \Delta_N} \langle \boldsymbol{p}, -\boldsymbol{L}_{t-1} \rangle$$

Hedge update

$$p_{t,i} \propto \exp\left(-\eta L_{t-1,i}\right), \forall i \in [N]$$

Lazy and Greedy Updates

- Hedge algorithm

lazy update

$$p_{t+1,i} \propto \exp\left(-\eta L_{t,i}\right), \forall i \in [N] \qquad L_{t,i} = \sum_{s=1}^{t} \ell_{s,i}, \ \forall i \in [N]$$

- Another equivalent update (when the learning rate η is ***fixed***)

greedy update

$$p_{t+1,i} \propto p_{t,i} \exp\left(-\eta \ell_{t,i}\right), \forall i \in [N]$$

where we set the uniform initialization as $p_{0,i} = 1/N, \forall i \in [N]$.

⇨ But the two updates can be ***significantly different*** ***when learning rate is changing***.

Hedge: Regret Bound

Theorem 1. *Suppose that* $\forall t \in [T]$ *and* $i \in [N], 0 \leq \ell_{t,i} \leq 1$*, then Hedge with learning rate* η *guarantees*

$$\mathrm{Regret}_T \leq \frac{\ln N}{\eta} + \eta T = \mathcal{O}\big(\sqrt{T \log N}\big),$$

where the last equality is by setting η *optimally as* $\sqrt{(\ln N)/T}$*.*

Proof. We present a 'potential-based' proof here, where the potential is defined as

$$\Phi_t \triangleq \frac{1}{\eta} \ln \left(\sum_{i=1}^{N} \exp\left(-\eta L_{t,i}\right) \right).$$

Proof of Hedge Regret Bound

Proof.

$$\Phi_t - \Phi_{t-1} = \frac{1}{\eta} \ln \left(\frac{\sum_{i=1}^N \exp\left(-\eta L_{t,i}\right)}{\sum_{i=1}^N \exp\left(-\eta L_{t-1,i}\right)} \right) \qquad \Phi_t \triangleq \frac{1}{\eta} \ln \left(\sum_{i=1}^N \exp\left(-\eta L_{t,i}\right) \right)$$

$$= \frac{1}{\eta} \ln \left(\sum_{i=1}^N \left(\frac{\exp\left(-\eta L_{t-1,i}\right)}{\sum_{i=1}^N \exp\left(-\eta L_{t-1,i}\right)} \exp(-\eta \ell_{t,i}) \right) \right)$$

$$= \frac{1}{\eta} \ln \left(\sum_{i=1}^N p_{t,i} \exp\left(-\eta \ell_{t,i}\right) \right) \qquad \text{(update step of } \boldsymbol{p}_t\text{)}$$

$$\leq \frac{1}{\eta} \ln \left(\sum_{i=1}^N p_{t,i} \left(1 - \eta \ell_{t,i} + \eta^2 \ell_{t,i}^2\right) \right) \qquad (\forall x \geq 0, e^{-x} \leq 1 - x + x^2)$$

$$= \frac{1}{\eta} \ln \left(1 - \eta \left\langle \boldsymbol{p}_t, \boldsymbol{\ell}_t \right\rangle + \eta^2 \sum_{i=1}^N p_{t,i} \ell_{t,i}^2 \right)$$

Proof of Hedge Regret Bound

Proof. $$\Phi_t - \Phi_{t-1} = \frac{1}{\eta} \ln \left(\frac{\sum_{i=1}^N \exp\left(-\eta L_{t,i}\right)}{\sum_{i=1}^N \exp\left(-\eta L_{t-1,i}\right)} \right)$$

$$\leq - \langle \boldsymbol{p}_t, \boldsymbol{\ell}_t \rangle + \eta \sum_{i=1}^N p_{t,i} \ell_{t,i}^2 \qquad (\ln(1+x) \leq x)$$

Summing over t, we have

$$\sum_{t=1}^T \langle \boldsymbol{p}_t, \boldsymbol{\ell}_t \rangle \leq \Phi_0 - \Phi_T + \eta \sum_{t=1}^T \sum_{i=1}^N p_{t,i} \ell_{t,i}^2 \qquad \Phi_t \triangleq \frac{1}{\eta} \ln \left(\sum_{i=1}^N \exp\left(-\eta L_{t,i}\right) \right)$$

$$\leq \frac{\ln N}{\eta} - \frac{1}{\eta} \ln \left(\exp\left(-\eta L_{T,i^\star}\right) \right) + \eta \sum_{t=1}^T \sum_{i=1}^N p_{t,i} \ell_{t,i}^2$$

$$\leq \frac{\ln N}{\eta} + L_{T,i^\star} + \eta \sum_{t=1}^T \sum_{i=1}^N p_{t,i} \ell_{t,i}^2$$

Proof of Hedge Regret Bound

Proof.

$$\sum_{t=1}^{T} \langle \boldsymbol{p}_t, \boldsymbol{\ell}_t \rangle \le \frac{\ln N}{\eta} + L_{T,i^\star} + \eta \sum_{t=1}^{T} \sum_{i=1}^{N} p_{t,i} \ell_{t,i}^2$$

Rearranging the term gives

$$\begin{aligned} \sum_{t=1}^{T} \langle \boldsymbol{p}_t, \boldsymbol{\ell}_t \rangle - L_{T,i^\star} &\le \frac{\ln N}{\eta} + \eta \sum_{t=1}^{T} \sum_{i=1}^{N} p_{t,i} \ell_{t,i}^2 \\ &\le \frac{\ln N}{\eta} + \eta T \qquad (\ell_{t,i} \le 1) \end{aligned}$$

Thus, setting $\eta = \sqrt{\ln N / T}$ yields

$$\mathrm{Regret}_T \le \frac{\ln N}{\eta} + \eta T = 2\sqrt{T \ln N}.$$

□

Lower bound of PEA

- As above, we have proved the regret bound for Hedge:

$$\mathrm{Regret}_T \leq 2\sqrt{T \ln N}$$

- A natural question: can we further improve the bound?

Theorem 2 (Lower Bound of PEA). *For any algorithm $\mathcal{A}$, we have that*

$$\sup_{T,N} \max_{\ell_1,\ldots,\ell_T} \frac{\mathrm{Regret}_T}{\sqrt{T \ln N}} \geq \frac{1}{\sqrt{2}}.$$

Hedge achieves minimax optimal regret (up to a constant of $2\sqrt{2}$) for PEA.

Lower bound of PEA

Theorem 2 (Lower Bound of PEA). *For any algorithm $\mathcal{A}$, we have that*

$$\sup_{T,N} \max_{\boldsymbol{\ell}_1,\ldots,\boldsymbol{\ell}_T} \frac{\mathrm{Regret}_T}{\sqrt{T \ln N}} \geq \frac{1}{\sqrt{2}}.$$

Proof. We construct the 'hard' instance by randomization. Let $\mathcal{D}$ be the uniform distribution over $\{0, 1\}$. We have

$$\begin{aligned}
\max_{\boldsymbol{\ell}_1,\ldots,\boldsymbol{\ell}_T} \mathrm{Regret}_T &\geq \mathbb{E}_{\boldsymbol{\ell}_1,\ldots,\boldsymbol{\ell}_T \overset{\text{i.i.d.}}{\sim} \mathcal{D}^N} [\mathrm{Regret}_T] \\
&\qquad \text{(conditional expectation decomposition)} \\
&= \sum_{t=1}^{T} \mathbb{E}_{\boldsymbol{\ell}_1,\ldots,\boldsymbol{\ell}_{t-1}} \mathbb{E}_{\boldsymbol{\ell}_t} [\langle p_t, \boldsymbol{\ell}_t \rangle \mid \boldsymbol{\ell}_{t-1}, \ldots, \boldsymbol{\ell}_1] - \mathbb{E}_{\boldsymbol{\ell}_1,\ldots,\boldsymbol{\ell}_T} \left[\min_{i\in[N]} \sum_{t=1}^{T} \ell_{t,i} \right] \\
&= \sum_{t=1}^{T} \mathbb{E}_{\boldsymbol{\ell}_1,\ldots,\boldsymbol{\ell}_{t-1}} \langle p_t, \mathbb{E}_{\boldsymbol{\ell}_t} [\boldsymbol{\ell}_t \mid \boldsymbol{\ell}_{t-1}, \ldots, \boldsymbol{\ell}_1] \rangle - \mathbb{E}_{\boldsymbol{\ell}_1,\ldots,\boldsymbol{\ell}_T} \left[\min_{i\in[N]} \sum_{t=1}^{T} \ell_{t,i} \right]
\end{aligned}$$

Lower bound of PEA

Theorem 2 (Lower Bound of PEA). *For any algorithm $\mathcal{A}$, we have that*

$$\sup_{T,N} \max_{\boldsymbol{\ell}_1,\ldots,\boldsymbol{\ell}_T} \frac{\mathrm{Regret}_T}{\sqrt{T \ln N}} \geq \frac{1}{\sqrt{2}}.$$

Proof. $$\max_{\boldsymbol{\ell}_1,\ldots,\boldsymbol{\ell}_T} \mathrm{Regret}_T \geq \sum_{t=1}^{T} \mathbb{E}_{\boldsymbol{\ell}_1,\ldots,\boldsymbol{\ell}_{t-1}} \left\langle \boldsymbol{p}_t, \mathbb{E}_{\boldsymbol{\ell}_t} \left[\boldsymbol{\ell}_t \mid \boldsymbol{\ell}_{t-1}, \ldots, \boldsymbol{\ell}_1 \right] \right\rangle - \mathbb{E}_{\boldsymbol{\ell}_1,\ldots,\boldsymbol{\ell}_T} \left[\min_{i\in[N]} \sum_{t=1}^{T} \ell_{t,i} \right]$$

$$= T/2 - \mathbb{E}_{\boldsymbol{\ell}_1,\ldots,\boldsymbol{\ell}_T} \left[\min_{i\in[N]} \sum_{t=1}^{T} \ell_{t,i} \right] = \mathbb{E}_{\boldsymbol{\ell}_1,\ldots,\boldsymbol{\ell}_T} \left[\max_{i\in[N]} \sum_{t=1}^{T} \left(\frac{1}{2} - \ell_{t,i} \right) \right]$$

$$= \frac{1}{2} \mathbb{E}_{\sigma_1,\ldots,\sigma_T} \left[\max_{i\in[N]} \sum_{t=1}^{T} \sigma_{t,i} \right],$$

($\boldsymbol{\ell}_t \overset{\text{i.i.d.}}{\sim} \mathcal{D}$ with $\mathcal{D}$ be the uniform distribution over $\{0,1\}$)

(σ_t for $i \in [N], t \in [T]$ are i.i.d. Rademacher random variables)

Lower bound of PEA

Theorem 2 (Lower Bound of PEA). *For any algorithm $\mathcal{A}$, we have that*

$$\sup_{T,N} \max_{\ell_1,\ldots,\ell_T} \frac{\mathrm{Regret}_T}{\sqrt{T \ln N}} \geq \frac{1}{\sqrt{2}}.$$

Proof.

$$\max_{\ell_1,\ldots,\ell_T} \mathrm{Regret}_T \geq \frac{1}{2} \mathbb{E}_{\sigma_1,\ldots,\sigma_T} \left[\max_{i \in [N]} \sum_{t=1}^{T} \sigma_{t,i} \right]$$

($\sigma_{t,i}$ for $i \in [N], t \in [T]$ are i.i.d. Rademacher random variables)

Using the result from probability theory (*Prediction, Learning, and Games*, Chapter 3.7) of Rademacher variables,

$$\Rightarrow \quad \lim_{T \to \infty} \lim_{N \to \infty} \frac{\mathbb{E}_{\sigma_1,\ldots,\sigma_T} \left[\max_{i \in [N]} \sum_{t=1}^{T} \sigma_{t,i} \right]}{\sqrt{T \ln N}} = \sqrt{2}. \qquad \square$$

Upper Bound and Lower Bound

Theorem 1. *Suppose that* $\forall t \in [T]$ *and* $i \in [N], 0 \leq \boldsymbol{\ell}_{t,i} \leq 1$*, then Hedge with learning rate* η *guarantees*

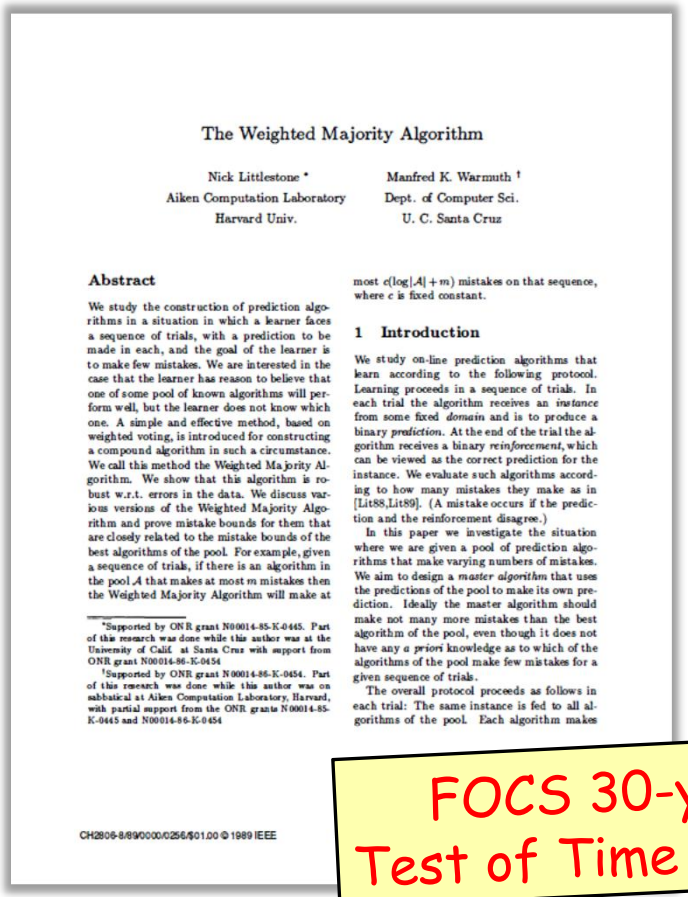
$$\mathrm{Regret}_T \leq \frac{\ln N}{\eta} + \eta T = \mathcal{O}\big(\sqrt{T \log N}\big),$$

where the last equality is by setting η *optimally as* $\sqrt{(\ln N)/T}$*.*

Theorem 2 (Lower Bound of PEA)**.** *For any algorithm* $\mathcal{A}$*, we have that*

$$\sup_{T,N} \max_{\boldsymbol{\ell}_1,\ldots,\boldsymbol{\ell}_T} \frac{\mathrm{Regret}_T}{\sqrt{T \ln N}} \geq \frac{1}{\sqrt{2}}.$$

Prediction with Expert Advice: history bits



The Weighted Majority Algorithm

Nick Littlestone * — Aiken Computation Laboratory, Harvard Univ.

Manfred K. Warmuth † — Dept. of Computer Sci., U. C. Santa Cruz

Abstract

We study the construction of prediction algorithms in a situation in which a learner faces a sequence of trials, with a prediction to be made in each, and the goal of the learner is to make few mistakes. We are interested in the case that the learner has reason to believe that one of some pool of known algorithms will perform well, but the learner does not know which one. A simple and effective method, based on weighted voting, is introduced for constructing a compound algorithm in such a circumstance. We call this method the Weighted Majority Algorithm. We show that this algorithm is robust w.r.t. errors in the data. We discuss various versions of the Weighted Majority Algorithm and prove mistake bounds for them that are closely related to the mistake bounds of the best algorithms of the pool. For example, given a sequence of trials, if there is an algorithm in the pool $\mathcal{A}$ that makes at most m mistakes then the Weighted Majority Algorithm will make at most $c(\log|\mathcal{A}|+m)$ mistakes on that sequence, where c is fixed constant.

*Supported by ONR grant N00014-85-K-0445. Part of this research was done while this author was at the University of Calif. at Santa Cruz with support from ONR grant N00014-86-K-0454

†Supported by ONR grant N00014-86-K-0454. Part of this research was done while this author was on sabbatical at Aiken Computation Laboratory, Harvard, with partial support from the ONR grants N00014-85-K-0445 and N00014-86-K-0454

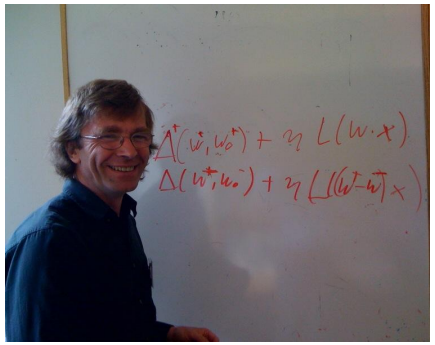
CH2805-8/89/0000/0256/$01.00 © 1989 IEEE

1 Introduction

We study on-line prediction algorithms that learn according to the following protocol. Learning proceeds in a sequence of trials. In each trial the algorithm receives an *instance* from some fixed *domain* and is to produce a binary *prediction*. At the end of the trial the algorithm receives a binary *reinforcement*, which can be viewed as the correct prediction for the instance. We evaluate such algorithms according to how many mistakes they make as in [Lit88,Lit89]. (A mistake occurs if the prediction and the reinforcement disagree.)

In this paper we investigate the situation where we are given a pool of prediction algorithms that make varying numbers of mistakes. We aim to design a *master algorithm* that uses the predictions of the pool to make its own prediction. Ideally the master algorithm should make not many more mistakes than the best algorithm of the pool, even though it does not have any *a priori* knowledge as to which of the algorithms of the pool make few mistakes for a given sequence of trials.

The overall protocol proceeds as follows in each trial: The same instance is fed to all algorithms of the pool. Each algorithm makes

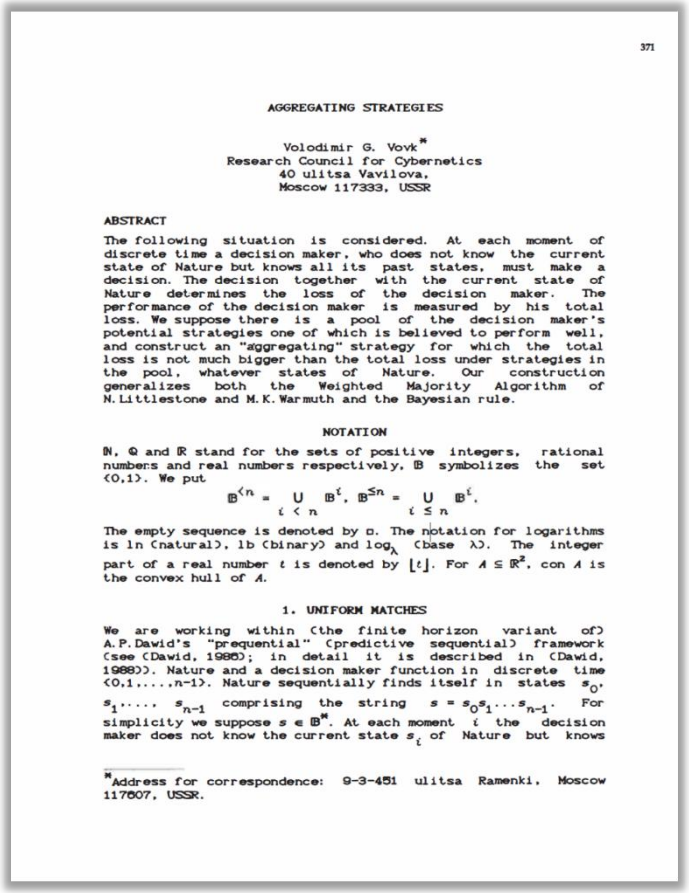


Manfred Warmuth

UC Santa Cruz

FOCS 30-year Test of Time Award!

Nick Littlestone and Manfred K. Warmuth.
"The Weighted Majority Algorithm." FOCS 1989: 256-261.



371

AGGREGATING STRATEGIES

Volodimir G. Vovk*
Research Council for Cybernetics
40 ulitsa Vavilova,
Moscow 117333, USSR

ABSTRACT

The following situation is considered. At each moment of discrete time a decision maker, who does not know the current state of Nature but knows all its past states, must make a decision. The decision together with the current state of Nature determines the loss of the decision maker. The performance of the decision maker is measured by his total loss. We suppose there is a pool of the decision maker's potential strategies one of which is believed to perform well, and construct an "aggregating" strategy for which the total loss is not much bigger than the total loss under strategies in the pool, whatever states of Nature. Our construction generalizes both the Weighted Majority Algorithm of N. Littlestone and M. K. Warmuth and the Bayesian rule.

NOTATION

$\mathbb{N}$, $\mathbb{Q}$ and $\mathbb{R}$ stand for the sets of positive integers, rational numbers and real numbers respectively, $\mathbb{B}$ symbolizes the set $\langle 0,1\rangle$. We put

$$\mathbb{B}^{<n} = \bigcup_{i<n} \mathbb{B}^i, \quad \mathbb{B}^{\le n} = \bigcup_{i\le n} \mathbb{B}^i.$$

The empty sequence is denoted by $\Box$. The notation for logarithms is ln (natural), lb (binary) and $\log_\lambda$ (base λ). The integer part of a real number t is denoted by $\lfloor t\rfloor$. For $A \subseteq \mathbb{R}^2$, con A is the convex hull of A.

1. UNIFORM MATCHES

We are working within (the finite horizon variant of) A. P. Dawid's "prequential" (predictive sequential) framework (see (Dawid, 1985); in detail it is described in (Dawid, 1988)). Nature and a decision maker function in discrete time $\langle 0,1,\dots,n-1\rangle$. Nature sequentially finds itself in states $s_0, s_1, \dots, s_{n-1}$ comprising the string $s = s_0 s_1 \dots s_{n-1}$. For simplicity we suppose $s \in \mathbb{B}^n$. At each moment i the decision maker does not know the current state s_i of Nature but knows

*Address for correspondence: 9-3-451 ulitsa Ramenki, Moscow 117607, USSR.

Volodimir G. Vovk. "Aggregating Strategies."
COLT 1990: 371-383.

Volodimir G. Vovk
Royal Holloway,
University of London

Prediction with Expert Advice: history bits

Yoav Freund

Robert Schapire

Goldel Prize 2003

This paper introduced AdaBoost, an adaptive algorithm to improve the accuracy of hypotheses in machine learning. The algorithm demonstrated novel possibilities in analyzing data and is a permanent contribution to science even beyond computer science.

JOURNAL OF COMPUTER AND SYSTEM SCIENCES **55**, 119–139 (1997)
ARTICLE NO. SS971504

A Decision-Theoretic Generalization of On-Line Learning and an Application to Boosting*

Yoav Freund and Robert E. Schapire†

AT&T Labs, 180 Park Avenue, Florham Park, New Jersey 07932

Received December 19, 1996

In the first part of the paper we consider the problem of dynamically apportioning resources among a set of options in a worst-case on-line framework. The model we study can be interpreted as a broad, abstract extension of the well-studied on-line prediction model to a general decision-theoretic setting. We show that the multiplicative weight-update Littlestone–Warmuth rule can be adapted to this model, yielding bounds that are slightly weaker in some cases, but applicable to a considerably more general class of learning problems. We show how the resulting learning algorithm can be applied to a variety of problems, including gambling, multiple-outcome prediction, repeated games, and prediction of points in $\mathbb{R}^n$. In the second part of the paper we apply the multiplicative weight-update technique to derive a new boosting algorithm. This boosting algorithm does not require any prior knowledge about the performance of the weak learning algorithm. We also study generalizations of the new boosting algorithm to the problem of learning functions whose range, rather than being binary, is an arbitrary finite set or a bounded segment of the real line.

converting a "weak" PAC learning algorithm that performs just slightly better than random guessing into one with arbitrarily high accuracy.

We formalize our *on-line allocation model* as follows. The allocation agent A has N options or *strategies* to choose from; we number these using the integers $1, \ldots, N$. At each time step $t = 1, 2, \ldots, T$, the allocator A decides on a distribution $\mathbf{p}^t$ over the strategies; that is $p_i^t \geqslant 0$ is the amount allocated to strategy i, and $\sum_{i=1}^N p_i^t = 1$. Each strategy i then suffers some *loss* ℓ_i^t which is determined by the (possibly adversarial) "environment." The loss suffered by A is then $\sum_{i=1}^n p_i^t \ell_i^t = \mathbf{p}^t \cdot \boldsymbol{\ell}^t$, i.e., the average loss of the strategies with respect to A's chosen allocation rule. We call this loss function the *mixture loss*.

In this paper, we always assume that the loss suffered by any strategy is bounded so that, without loss of generality, $\ell_i^t \in [0, 1]$. Besides this condition, we make no assumptions

Reference: Y. Freund and R. Schapire. A Decision-Theoretic Generalization of On-Line Learning and an Application to Boosting. JCSS 1997.

PEA vs. OCO

Prediction with Expert Advice

At each round $t = 1, 2, \cdots$

(1) the player first picks a weight $\boldsymbol{p}_t$ from a simplex Δ_N;

(2) and simultaneously environments pick an loss vector $\boldsymbol{\ell}_t \in \mathbb{R}^N$;

(3) the player suffers loss $f_t(\boldsymbol{p}_t) \triangleq \langle \boldsymbol{p}_t, \boldsymbol{\ell}_t \rangle$, observes $\boldsymbol{\ell}_t$ and updates the model.

require domain to be a simplex $\mathcal{X} = \Delta_N$ ⇧ linear loss $f_t(\mathbf{x}) \triangleq \langle \mathbf{x}, \boldsymbol{\ell}_t \rangle$

PEA is a ***special case*** of OCO!

Online Convex Optimization

At each round $t = 1, 2, \cdots$

(1) the player first picks a model $\mathbf{x}_t \in \mathcal{X}$;

(2) and simultaneously environments pick an online function $f_t : \mathcal{X} \to \mathbb{R}$;

(3) the player suffers loss $f_t(\mathbf{x}_t)$, observes f_t and updates the model.

Deploying OGD to PEA

- PEA is a special case of OCO:

  Why not directly deploy OGD (proposed in last lecture) to address PEA?

Theorem 4 (Regret bound for OGD). *Under Assumption 1, 2 and 3, online gradient descent (OGD) with step sizes* $\eta_t = \frac{D}{G\sqrt{t}}$ *for* $t \in [T]$ *guarantees:*

$$\text{Regret}_T = \sum_{t=1}^{T} f_t(\mathbf{x}_t) - \min_{\mathbf{x}\in\mathcal{X}} \sum_{t=1}^{T} f_t(\mathbf{x}) \leq \frac{3}{2} GD\sqrt{T}.$$

Regret guarantee: $D = \max_{\mathbf{x},\mathbf{y}\in\Delta_N} \|\mathbf{x}-\mathbf{y}\|_2 = \sqrt{2}$ $\quad G = \max_{\boldsymbol{\ell}_t\in\mathbb{R}^N} \|\boldsymbol{\ell}_t\|_2 = \sqrt{N}$

$$\Longrightarrow \text{Regret}_T = \sum_{t=1}^{T} \langle \boldsymbol{p}_t, \boldsymbol{\ell}_t \rangle - \min_{\boldsymbol{p}\in\Delta_N} \sum_{t=1}^{T} \langle \boldsymbol{p}, \boldsymbol{\ell}_t \rangle \leq \mathcal{O}(\sqrt{TN})$$

Deploying OGD to PEA

- OGD for PEA Problem:

$$D = \max_{\mathbf{x},\mathbf{y}\in\Delta_N} \|\mathbf{x}-\mathbf{y}\|_2 = \sqrt{2} \qquad G = \max_{\boldsymbol{\ell}_t\in\mathbb{R}^N} \|\boldsymbol{\ell}_t\|_2 = \sqrt{N}$$

$$\Rightarrow \quad \text{Regret}_T = \sum_{t=1}^{T} \langle \boldsymbol{p}_t, \boldsymbol{\ell}_t \rangle - \min_{\boldsymbol{p}\in\Delta_N} \sum_{t=1}^{T} \langle \boldsymbol{p}, \boldsymbol{\ell}_t \rangle \le \mathcal{O}(\sqrt{TN})$$

- A natural question: is the $\mathcal{O}(\sqrt{TN})$ regret bound tight enough?
  - recall that the lower bound of PEA is $\Omega(\sqrt{T\log N})$
  - OGD is not optimal with respect to N (number of experts)

Deploying OGD to PEA

- PEA is a special case of OCO:

  Why not directly deploy OGD (proposed in last lecture) to address PEA?

Theorem 4 (Regret bound for OGD). *Under Assumption 1, 2 and 3, online gradient descent (OGD) with step sizes* $\eta_t = \frac{D}{G\sqrt{t}}$ *for* $t \in [T]$ *guarantees:*

$$\text{Regret}_T = \sum_{t=1}^{T} f_t(\mathbf{x}_t) - \min_{\mathbf{x}\in\mathcal{X}} \sum_{t=1}^{T} f_t(\mathbf{x}) \leq \frac{3}{2} GD\sqrt{T}.$$

Regret guarantee: $D = \max_{\mathbf{x},\mathbf{y}\in\Delta_N} \|\mathbf{x}-\mathbf{y}\|_2 = \sqrt{2}$ $\qquad$ $G = \max_{\boldsymbol{\ell}_t\in\mathbb{R}^N} \|\boldsymbol{\ell}_t\|_2 = \sqrt{N}$

$$\Longrightarrow \quad \text{Regret}_T = \sum_{t=1}^{T} \langle \boldsymbol{p}_t, \boldsymbol{\ell}_t \rangle - \min_{\boldsymbol{p}\in\Delta_N} \sum_{t=1}^{T} \langle \boldsymbol{p}, \boldsymbol{\ell}_t \rangle \leq \mathcal{O}(\sqrt{TN})$$

Why OGD Fails for PEA

- PEA has a special structure whereas general OCO doesn't have.

Convex Problem	**PEA Problem**
Domain: convex set $\mathcal{X}$	Domain: simplex $\mathcal{X} = \Delta_N$
Online function: convex function f_t	Online function: linear $f_t(\boldsymbol{p}) \triangleq \langle \boldsymbol{p}, \boldsymbol{\ell}_t \rangle$
Lower Bound: $\Omega(GD\sqrt{T})$	Lower Bound: $\Omega(\sqrt{T \log N})$

Why OGD Fails for PEA

- Remember that for the general OCO, we linearized the function to analyze the first gradient descent lemma:

$$\begin{aligned}
\|\mathbf{x}_{t+1} - \mathbf{x}^\star\|^2 &= \|\Pi_{\mathcal{X}}[\mathbf{x}_t - \eta_t \nabla f(\mathbf{x}_t)] - \mathbf{x}^\star\|^2 \quad \text{(GD)} \\
&\leq \|\mathbf{x}_t - \eta_t \nabla f(\mathbf{x}_t) - \mathbf{x}^\star\|^2 \quad \text{(Pythagoras Theorem)} \\
&= \|\mathbf{x}_t - \mathbf{x}^\star\|^2 - 2\eta_t \langle \nabla f(\mathbf{x}_t), \mathbf{x}_t - \mathbf{x}^\star \rangle + \eta_t^2 \|\nabla f(\mathbf{x}_t)\|^2 \\
&\leq \|\mathbf{x}_t - \mathbf{x}^\star\|^2 - 2\eta_t (f(\mathbf{x}_t) - f^\star) + \eta_t^2 \|\nabla f(\mathbf{x}_t)\|^2 \\
&\quad \text{(convexity: } f(\mathbf{x}_t) - f^\star = f(\mathbf{x}_t) - f(\mathbf{x}^\star) \leq \langle \nabla f(\mathbf{x}_t), \mathbf{x}_t - \mathbf{x}^\star \rangle\text{)}
\end{aligned}$$

- So, linearized loss is not the essence, but the ***simplex domain*** of the PEA problem is worthy specifically considering.

Why OGD Fails for PEA?

- Recall that for general OCO, we update the model as follows:

General Online Convex Optimization

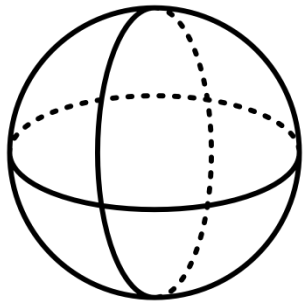
OGD:

$$\mathbf{x}_{t+1} = \Pi_{\mathcal{X}} \left[\mathbf{x}_t - \eta_t \nabla f(\mathbf{x}_t) \right]$$

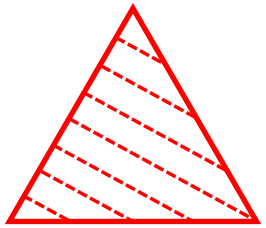
Proximal type update:

$$\mathbf{x}_{t+1} = \arg\min_{\mathbf{x} \in \mathcal{X}} \left\{ \langle \mathbf{x}, \eta_t \nabla f_t(\mathbf{x}_t) \rangle + \frac{1}{2} \|\mathbf{x} - \mathbf{x}_t\|_2^2 \right\}$$

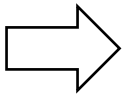
- In PEA, is it proper to use 2-norm (ball) to measure distance?

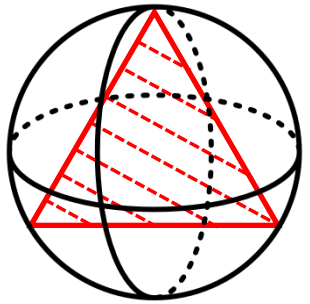


Ball



Simplex





A ball is too pessimistic (loose) to measure a simplex!

Proximal View

- Recall that for general OCO, we update the model as follows:

General Online Convex Optimization

OGD:

$$\mathbf{x}_{t+1} = \Pi_{\mathcal{X}} \left[\mathbf{x}_t - \eta_t \nabla f(\mathbf{x}_t) \right]$$

Proximal type update:

$$\mathbf{x}_{t+1} = \arg\min_{\mathbf{x} \in \mathcal{X}} \left\{ \langle \mathbf{x}, \eta_t \nabla f_t(\mathbf{x}_t) \rangle + \frac{1}{2} \left\| \mathbf{x} - \mathbf{x}_t \right\|_2^2 \right\}$$

- In PEA, is it proper to use 2-norm (ball) to measure distance?

⇨ We need to find an alternative distance measure for the ***special structure*** in PEA.

Proximal View

⇨ We need to find an alternative distance measure for the ***special structure*** in PEA.

- Intuitively, for Euclidean space, 2-norm is the most natural measure:

$$\|\mathbf{x} - \mathbf{y}\|_2^2$$

- For PEA problem
  - the decision can be viewed as a distribution within the simplex
  - for two distributions P and Q, KL divergence is a natural measure:

$$\mathrm{KL}(P\|Q) = \sum_{x \in \mathcal{X}} P(x) \log \left(\frac{P(x)}{Q(x)} \right)$$

Reinvent Hedge Algorithm

Theorem 3. *Consider* $f_t(\boldsymbol{p}) = \langle \boldsymbol{p}, \boldsymbol{\ell}_t \rangle$. *An online learning algorithm that updates the model following*

$$\boldsymbol{p}_{t+1} = \arg\min_{\boldsymbol{p} \in \Delta_N} \left\{ \eta \langle \boldsymbol{p}, \nabla f_t(\boldsymbol{p}_t) \rangle + \mathrm{KL}(\boldsymbol{p} \| \boldsymbol{p}_t) \right\}$$

is equal to Hedge update, i.e.,

$$p_{t+1,i} \propto p_{t,i} \exp\left(-\eta \ell_{t,i}\right) \text{ for all } i \in [N].$$

Proof.

$$\begin{aligned} \boldsymbol{p}_{t+1} &= \arg\min_{\boldsymbol{p} \in \Delta_N} \eta \langle \boldsymbol{p}, \nabla f_t(\boldsymbol{p}_t) \rangle + \mathrm{KL}(\boldsymbol{p} \| \boldsymbol{p}_t) \\ &= \arg\min_{\boldsymbol{p} \in \Delta_N} \underbrace{\eta \langle \boldsymbol{p}, \nabla f_t(\boldsymbol{p}_t) \rangle - \sum_{i=1}^{N} p_i \ln\left(\frac{p_{t,i}}{p_i}\right)}_{F(\boldsymbol{p})} \qquad \text{(definition of KL divergence)} \end{aligned}$$

□

Reinvent Hedge Algorithm

- Proximal update rule for OGD:

$$\mathbf{x}_{t+1} = \arg\min_{\mathbf{x}\in\mathcal{X}} \left\{ \eta_t \langle \mathbf{x}, \nabla f_t(\mathbf{x}_t) \rangle + \frac{1}{2} \|\mathbf{x} - \mathbf{x}_t\|_2^2 \right\}$$

- Proximal update rule for Hedge:

$$\mathbf{x}_{t+1} = \arg\min_{\mathbf{x}\in\mathcal{X}} \left\{ \eta_t \langle \mathbf{x}, \nabla f_t(\mathbf{x}_t) \rangle + \mathrm{KL}(\mathbf{x} \| \mathbf{x}_t) \right\}$$

- More possibility: changing the distance measure to a more general form using ***Bregman divergence***

$$\mathbf{x}_{t+1} = \arg\min_{\mathbf{x}\in\mathcal{X}} \left\{ \eta_t \langle \mathbf{x}, \nabla f_t(\mathbf{x}_t) \rangle + \mathcal{D}_\psi(\mathbf{x}, \mathbf{x}_t) \right\}$$

Bregman Divergence

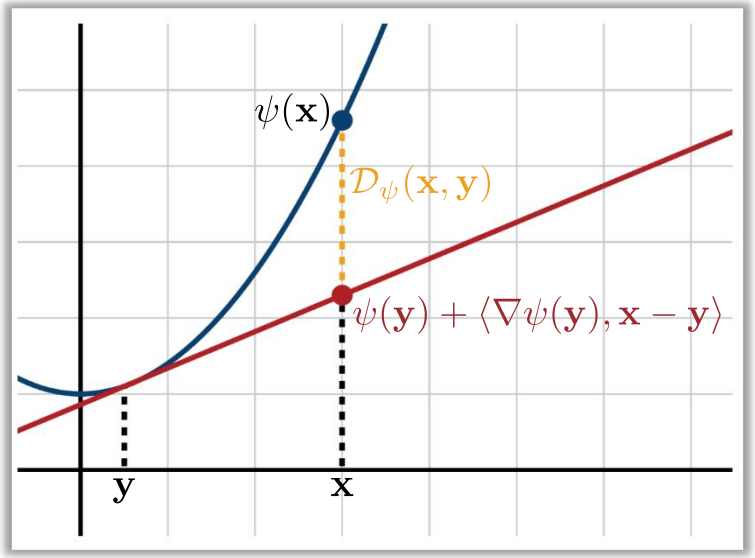
Definition 1 (Bregman Divergence). Let ψ be a strongly convex and differentiable function over a convex set $\mathcal{X}$, then for any $\mathbf{x}, \mathbf{y} \in \mathcal{X}$, the bregman divergence $\mathcal{D}_\psi$ associated to ψ is defined as

$$\mathcal{D}_\psi(\mathbf{x}, \mathbf{y}) = \psi(\mathbf{x}) - \psi(\mathbf{y}) - \langle \nabla\psi(\mathbf{y}), \mathbf{x} - \mathbf{y} \rangle.$$

- Bregman divergence measures the difference of a function and its linear approximation.
- Using second-order Taylor expansion, we know

$$\mathcal{D}_\psi(\mathbf{x}, \mathbf{y}) = \frac{1}{2}\|\mathbf{x} - \mathbf{y}\|^2_{\nabla^2\psi(\boldsymbol{\xi})}$$

for some $\boldsymbol{\xi} \in [\mathbf{x}, \mathbf{y}]$.



Bregman Divergence

Definition 1 (Bregman Divergence). Let ψ be a strongly convex and differentiable function over a convex set $\mathcal{X}$, then for any $\mathbf{x}, \mathbf{y} \in \mathcal{X}$, the bregman divergence $\mathcal{D}_\psi$ associated to ψ is defined as

$$\mathcal{D}_\psi(\mathbf{x}, \mathbf{y}) = \psi(\mathbf{x}) - \psi(\mathbf{y}) - \langle \nabla\psi(\mathbf{y}), \mathbf{x} - \mathbf{y} \rangle.$$

Table 1: Choice of $\psi(\cdot)$ and the corresponding Bregman divergence.

	$\psi(\mathbf{x})$	$\mathcal{D}_\psi(\mathbf{x}, \mathbf{y})$
Squared L_2-distance	$\|\mathbf{x}\|_2^2$	$\|\mathbf{x} - \mathbf{y}\|_2^2$
Mahalanobis distance	$\|\mathbf{x}\|_A^2$	$\|\mathbf{x} - \mathbf{y}\|_A^2$
Negative entropy	$\sum_i x_i \log x_i$	$\mathrm{KL}(\mathbf{x}\|\mathbf{y})$

Online Mirror Descent

Online Mirror Descent

At each round $t = 1, 2, \cdots$

$$\mathbf{x}_{t+1} = \arg\min_{\mathbf{x} \in \mathcal{X}} \left\{ \eta_t \langle \mathbf{x}, \nabla f_t(\mathbf{x}_t) \rangle + \mathcal{D}_\psi(\mathbf{x}, \mathbf{x}_t) \right\}$$

where $\mathcal{D}_\psi(\mathbf{x}, \mathbf{y}) = \psi(\mathbf{x}) - \psi(\mathbf{y}) - \langle \nabla \psi(\mathbf{y}), \mathbf{x} - \mathbf{y} \rangle$ is the Bregman divergence.

- $\psi(\cdot)$ is a required to be strongly convex and differentiable over a convex set $\mathcal{X}$.
- Strong convexity of ψ will ensure the uniqueness of the minimization problem, and actually we further need some analytical assumptions (see later mirror map defintion) to ensure the solutions' feasibility in domain $\mathcal{X}$.

Online Mirror Descent

- So we can unify OGD and Hedge from the same view of OMD.

$$\mathbf{x}_{t+1} = \arg\min_{\mathbf{x}\in\mathcal{X}} \left\{ \eta_t \langle \mathbf{x}, \nabla f_t(\mathbf{x}_t) \rangle + \mathcal{D}_\psi(\mathbf{x}, \mathbf{x}_t) \right\}$$

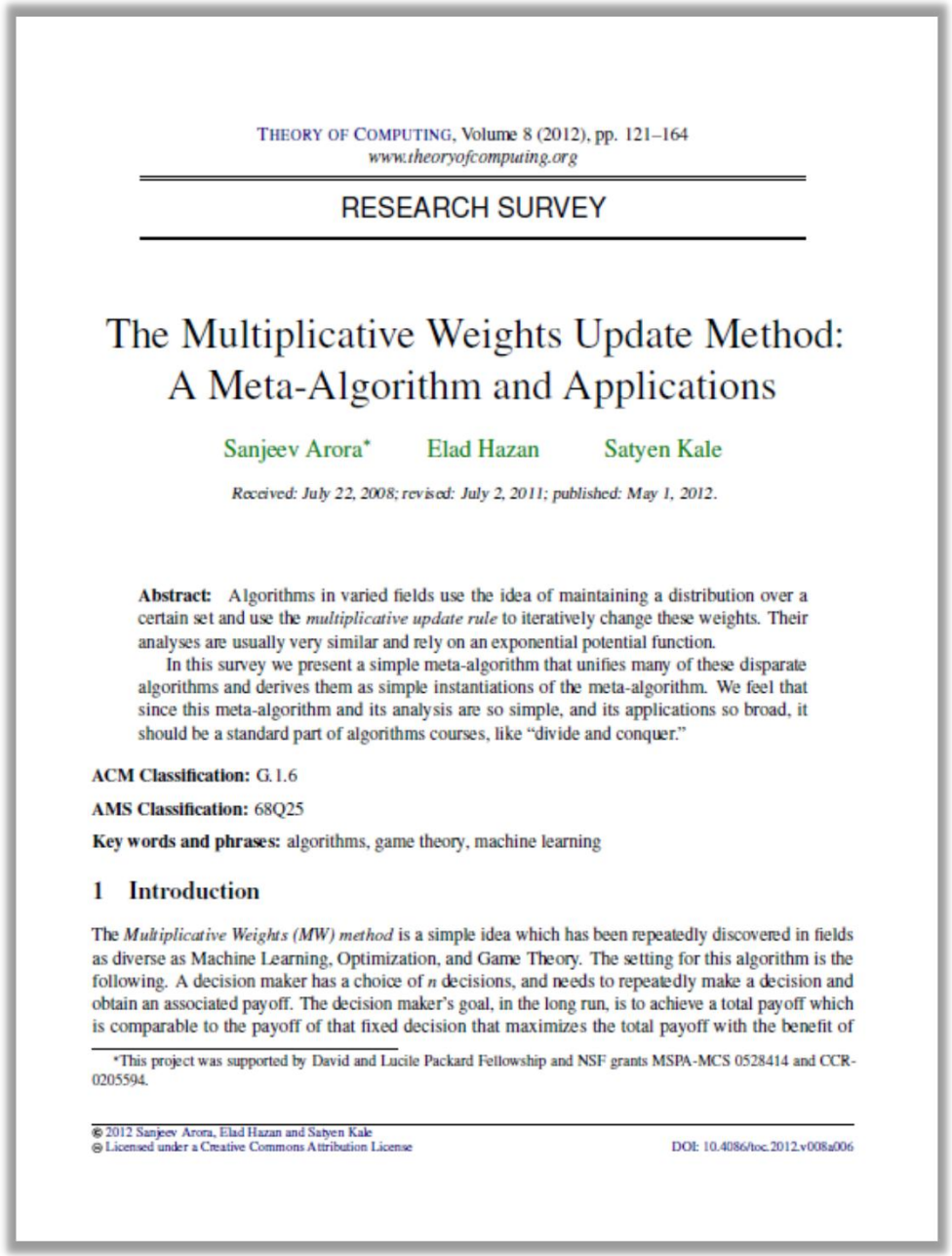
Algo.	OMD/proximal form	$\psi(\cdot)$	η_t	Regret_T
OGD	$\mathbf{x}_{t+1} = \arg\min_{\mathbf{x}\in\mathcal{X}} \left\{ \eta_t \langle \mathbf{x}, \nabla f_t(\mathbf{x}_t) \rangle + \frac{1}{2} \|\mathbf{x} - \mathbf{x}_t\|_2^2 \right\}$	$\|\mathbf{x}\|_2^2$	$\frac{1}{\sqrt{t}}$	$\mathcal{O}(\sqrt{T})$
Hedge	$\mathbf{x}_{t+1} = \arg\min_{\mathbf{x}\in\Delta_N} \left\{ \eta_t \langle \mathbf{x}, \nabla f_t(\mathbf{x}_t) \rangle + \text{KL}(\mathbf{x} \| \mathbf{x}_t) \right\}$	$\sum_{i=1}^N x_i \log x_i$	$\sqrt{\frac{\ln N}{T}}$	$\mathcal{O}(\sqrt{T \log N})$

⇨ We will give the (unified) regret analysis in the next lecture.

Why is PEA useful?

- Prediction with Expert Advice is essentially a **meta-algorithm** for combining different experts, and the "expert" can be interpreted as any learning model with a particular kind of expertise.

- It is used in a variety of algorithmic design.



THEORY OF COMPUTING, Volume 8 (2012), pp. 121–164
www.theoryofcomputing.org

RESEARCH SURVEY

The Multiplicative Weights Update Method: A Meta-Algorithm and Applications

Sanjeev Arora* Elad Hazan Satyen Kale

Received: July 22, 2008; revised: July 2, 2011; published: May 1, 2012.

Abstract: Algorithms in varied fields use the idea of maintaining a distribution over a certain set and use the *multiplicative update rule* to iteratively change these weights. Their analyses are usually very similar and rely on an exponential potential function.

In this survey we present a simple meta-algorithm that unifies many of these disparate algorithms and derives them as simple instantiations of the meta-algorithm. We feel that since this meta-algorithm and its analysis are so simple, and its applications so broad, it should be a standard part of algorithms courses, like "divide and conquer."

ACM Classification: G.1.6

AMS Classification: 68Q25

Key words and phrases: algorithms, game theory, machine learning

1 Introduction

The *Multiplicative Weights (MW) method* is a simple idea which has been repeatedly discovered in fields as diverse as Machine Learning, Optimization, and Game Theory. The setting for this algorithm is the following. A decision maker has a choice of n decisions, and needs to repeatedly make a decision and obtain an associated payoff. The decision maker's goal, in the long run, is to achieve a total payoff which is comparable to the payoff of that fixed decision that maximizes the total payoff with the benefit of

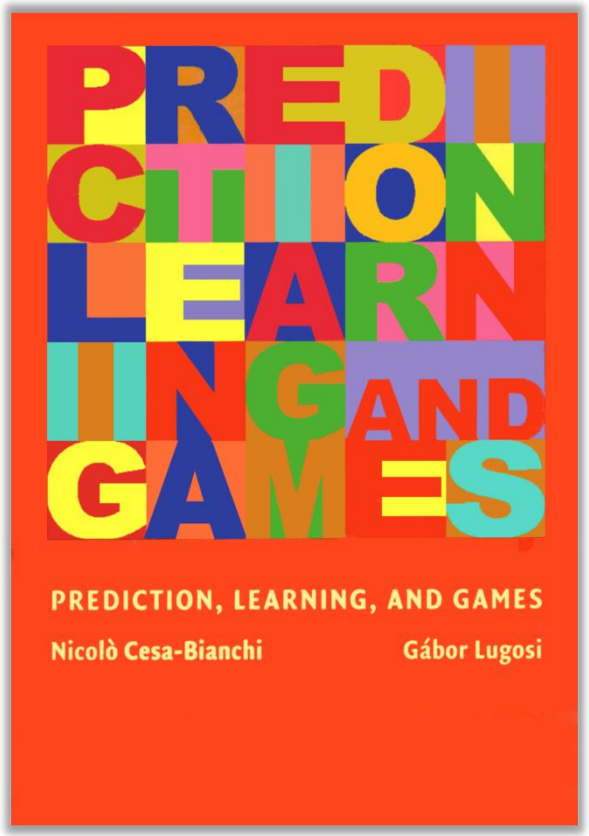
*This project was supported by David and Lucile Packard Fellowship and NSF grants MSPA-MCS 0528414 and CCR-0205594.

DOI: 10.4086/toc.2012.v008a006

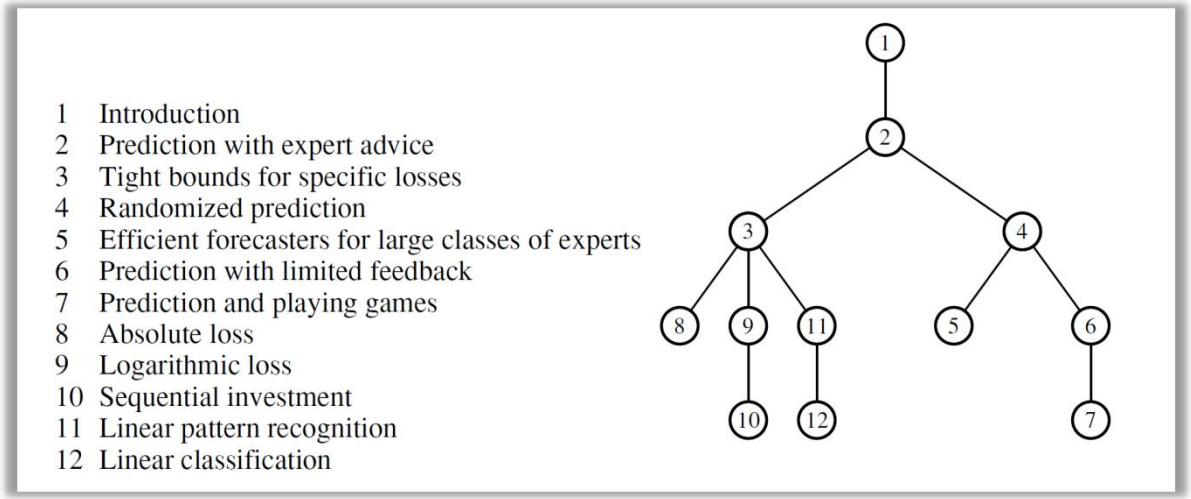
- Applications
  - Learning a linear classifier: the Winnow algorithm
  - Solving zero-sum games approximately
  - Plotkin, Shmoys, Tardos framework for packing/covering LPs
  - Approximating multicommodity flow problems
  - O(log n)-approximation for many NP-hard problems
  - Learning theory and boosting
  - Hard-core sets and the XOR Lemma
  - Hannan's algorithm and multiplicative weights
  - Online convex optimization
  - Other applications
  - Design of competitive online algorithms

The multiplicative weights update method: a meta-algorithm and applications. S Arora, E Hazan, S Kale. Theory of Computing, 2012

More Results on PEA



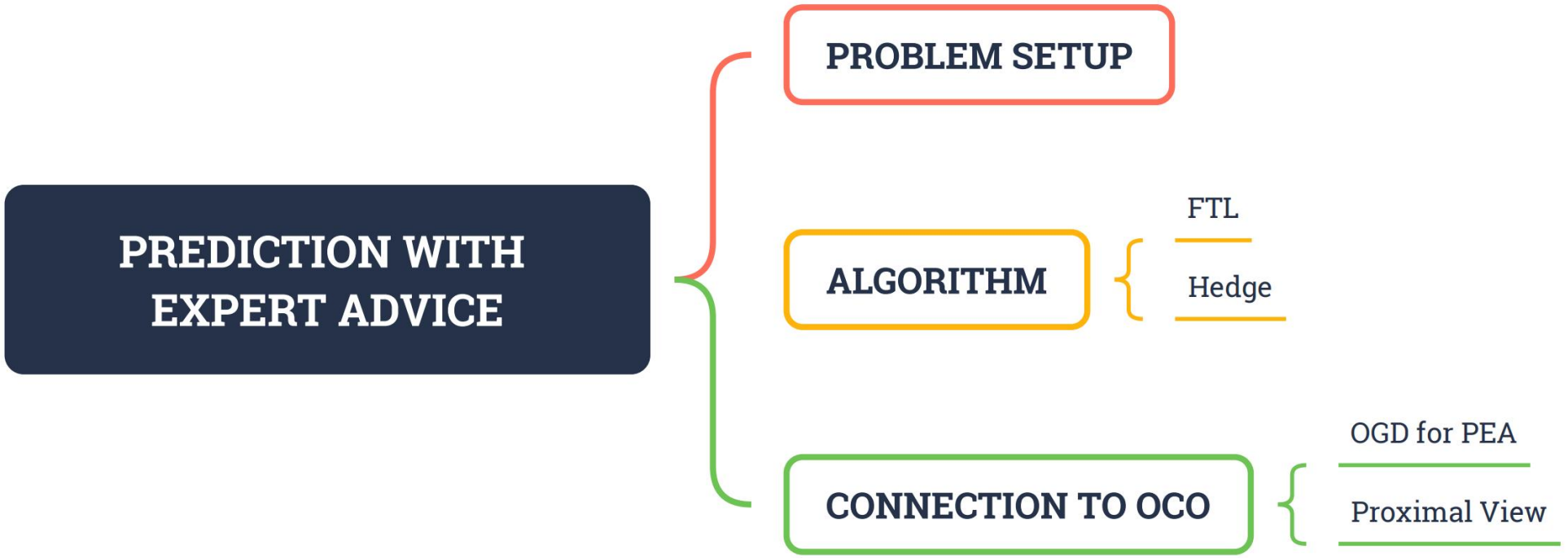
Prediction, Learning and Games.
Nicolò Cesa-Bianchi and Gabor Lugosi.
Cambridge University Press, 2006.



Nicolò Cesa-Bianchi

Gabor Lugosi

Summary



Q & A

Thanks!