CS170

SECTION 10

Multiplicative Weights, Reductions

Multiplicative Weights

Setup: - Have n experts, each giving you advice each day.
- At the end of the day, you find out how right they were (each expert $i(t)$ on day t has a loss $f^t_{i(t)} \in [0, 1]$). A lower loss means the expert gave good advice.

expert	day 1 am	day 1 pm	day 2 am	day 2 pm	....	day T am	day T pm
1	A	0.5	B	0.6			f_1^T
2	B	0.3	A	0.2			f_2^T
3	A	0.5	C	0.4			f_3^T
⋮							
n	D	1	F	0.1	....		f_n^T
	↑ opinion	↑ loss f_i^t for expert i on day t	↑ opinion	↑ loss		↑ opinion	↑ loss

Total loss: $\sum_{t=1}^{T} f^t_{i(t)}$ [sum of the loss of the expert you chose each day (could be different each day) over all days].

We want to minimize total loss.

Best we can do is minimize regret (how different our choices were than the experts who made the best decisions overall).

- Can only know regret in hindsight, but it is still effective as a metric.

Regret: $\sum_{t=1}^{T} f^t_{i(t)} - \min_{i=1,\dots,n} \sum_{t=1}^{T} f_i^t$

Our total loss — Loss of the best expert.

Note 1: If we don't choose our expert probabalistically (choose a fixed expert each day or according to a fixed pattern), it is always possible to adversarially cope up with losses that result in horrible total loss.

Example: Take the majority opinion each day!
If we assign a loss of 1 to the majority experts and 0 to minority experts, then our regret will be T (very bad).

We defeat adversarial losses by using randomness!
We assign each expert a trust value x_i^t for each day, and choose expert i with probability x_i^t for day t.

Note: $\sum_{i=1}^{n} x_i^t = 1$.

New regret: $\sum_{t=1}^{T} \sum_{i=1}^{n} x_i^t \cdot f_i^t - \min_{i=1,\dots,n} \sum_{t=1}^{T} f_i^t$

Expected total loss

(Deterministic) loss of best expert.

Note 2: Why can't we minimize loss with respect to the best expert for each day, rather than the best expert overall?

A: We cannot hope for a selection strategy that competes with each day's best expert.

For each t, adversarially set $\begin{cases} \text{loss } 0 \text{ for expert with least probability on day } t \\ \text{loss } 1 \text{ for all other experts} \end{cases}$

This gives $\mathbb{E}\left[\left(\sum_{t=1}^{T} f_{i(t)}^t\right)\right] = \sum_{t=1}^{T} \sum_{i=1}^{n} p_i^t f_i^t \geq \sum_{t=1}^{T} \left(1 - \frac{1}{n}\right) = \frac{n-1}{n} \cdot T$

$\sum_{t=1}^{T} \min_{i \in [n]} f_i^t = \sum_{t=1}^{T} 0 = 0$

largest value for the smallest probability

difference is $\frac{n-1}{n} \cdot T$ (it's large)

MWU(Multiplicative Weight Update) algorithm

[High - level idea : If an expert makes a mistake , trust them less, but don't shut them out entirely .
The bigger the mistake , the less you trust them in future days] .

- given a parameter $\varepsilon \in [0, 1]$.
- each expert i has a weight $w_i^{(t)}$ for day t.
- On day 0, all weights $w_i^{(0)} = 1$.
    - for each day :
        - Choose an expert

REDUCTIONS

When a problem A reduces to a problem B ($A \to B$),

{ if we know how to solve B, we can use it as a $\boxed{\text{blackbox}}$ to solve problem A.

Note: Your TA probably will not cover all the problems. This is totally fine, the discussion worksheets are not designed to be finished in an hour. They are deliberately made long so they can serve as a resource you can use to practice, reinforce, and build upon concepts discussed in lecture, readings, and the homework.

1 Multiplicative Weights Intro

Multiplicative Weights

This is an online algorithm, in which you take into account the advice of n experts. Every day you get more information on how good every expert is until the last day T.

Let's first define some terminology:

- $x_i^{(t)}$ = proportion that you 'trust' expert i on day t
- $l_i^{(t)}$ = loss you would incur on day i if you invested everything into expert i
- total regret: $R_T = \sum_{t=1}^{T} \sum_{i=1}^{n} x_i^{(t)} l_i^{(t)} - \min_{i=1,\dots,n} \sum_{t=1}^{T} l_i^{(t)}$

$\forall i \in [1, n]$ and $\forall t \in [1, T]$, the multiplicative update is as follows:

$$w_i^{(0)} = 1$$

$$w_i^{(t)} = w_i^{(t-1)} (1 - \epsilon)^{l_i^{(t-1)}}$$

$$x_i^{(t)} = \frac{w_i^{(t)}}{\sum_{i=1}^{n} w_i^{(t)}}$$

If $\epsilon \in (0, 1/2]$, and $l_i^{(t)} \in [0, 1]$, we get the following bound on total regret:

$$R_T \leq \epsilon T + \frac{\ln(n)}{\epsilon}$$

Let's play around with some of these questions. For this problem, we will be running the randomized multiplicative weights algorithm with two experts. Consider every subpart of this problem distinct from the others.

(a) Let's say we believe the best expert will have cost 20, we run the algorithm for 100 days, and epsilon is $\frac{1}{2}$. What is the maximum value that the total loss incurred by the algorithm can be?

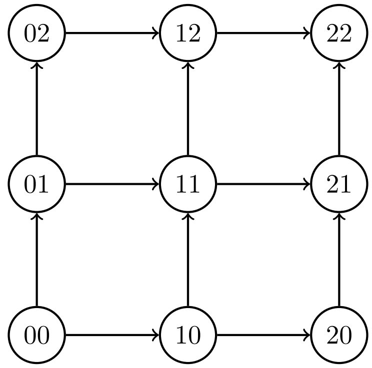
(b) What value of ϵ should we choose to minimize the total regret, given that we run the algorithm for 25 days?

(c) We run the randomized multiplicative weights algorithm with two experts. In all of the first 140 days, Expert 1 has cost 0 and Expert 2 has cost 1. If we chose $\epsilon = 0.01$, on the 141st day with what probability will we play Expert 1? (Hint: You can assume that $0.99^{70} = \frac{1}{2}$)

2 Multiplicative Weights

Consider the following simplified map of Berkeley. Due to traffic, the time it takes to traverse a given path can change each day. Specifically, the length of each edge in the network is a number between

$[0,1]$ that changes each day. The travel time for a path on a given day is the sum of the edges along the path.



For T days, both Max and Vinay drive from node 00 to node 22.

To cope with the unpredictability of traffic, Vinay builds a time machine and travels forward in time to determine the traffic on each edge on every day. Using this information, Vinay picks the path that has the smallest total travel time over T days, and uses the same path each day.

Max wants to use the multiplicative weights update algorithm to pick a path each day. In particular, Max wants to ensure that the difference between his expected total travel time over T days and Vinay's total travel time is at most $T/10000$. Assume that Max finds out the lengths of all the edges in the network, even those he did not drive on, at the end of each day.

(a) How many experts should Max use in the multiplicative weights algorithm?

(b) What are the experts?

(c) Given the weights maintained by the algorithm, how does Max pick a route on any given day?

(d) The regret bound for multiplicative weights is as follows:

Theorem. Assuming that all losses for the n experts are in the range $[0,4]$, the worst possible regret of the multiplicative weights algorithm run for T steps is

$$R_T \leq 8\sqrt{T \ln n}$$

Use the regret bound to show that expected total travel time of Max is not more than $T/10000$ worse than that of Vinay for large enough T.

Reduction: Suppose we have an algorithm to solve problem A, how to use it to solve problem B?

This has been and will continue to be a recurring theme of the class. Examples so far include

- Use SCC to solve 2SAT.
- Use LP to solve max flow.
- Use max flow to solve mincut.
- Use max flow to solve maximum bipartite matching.

In each case, we would transform the instance of problem B we want to solve into an instance of problem A that we can solve. Importantly, the transformation is efficient, say, in polynomial time.

Conceptually, a efficient reduction means that problem B is no harder than A. On the other hand, if we somehow know that B cannot be solved efficiently, we cannot hope that A can be solved efficiently.

To show that the reduction works, you need to prove (1) if there is a solution for an instance of problem A, there must be a solution to the transformed instance of problem B and (2) if there is a solution to the transformed instance of B, there must be a solution in the corresponding instance of problem A.

3 Vertex Cover to Set Cover

In the minimum vertex cover problem, we are given an undirected unweighted graph $G = (V, E)$ and asked to find the smallest set $U \subseteq V$ that "covers" the set of edges E. In other words, we want to find the smallest set U such that for each $(u, v) \in E$, either u or v is in U.

Now recall the definition of the minimum set cover problem: Given a set U of elements and a collection $S_1, \ . \ . \ . \ , S_m$ of subsets of U, the problem asks for the smallest collection of these sets whose union equals U.

Give an efficient reduction from the minimum vertex cover problem to the minimum set cover problem.

4 Maximum Spanning Tree

In this class, we have been talking about minimum spanning tree. What about maximum spanning tree? Can you use the minimum spanning tree algorithms we learned, Prim's and Kruskal's, as blackbox to find maximum spanning tree? Assume the graph is undirected and with positive edge weights.

LECTURE #17

CS 170

Spring 2021

Last time:

zero-sum games

row player and column player can be associated to dual LPs

value of the game is the optimum value of these LPs

(an example of duality)

Today:

experts problem

multiplicative weight updates

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."

n experts (people whose advice you take or not)

T days (may or may not be known a priori)

Each day $t \in [T]$, you choose an expert $i(t) \in [n]$ (according to some rule), and incur a loss $f^t_{i(t)} \in [0,1]$. (Losses are bounded.)

you chose to "follow" the prediction of expert $i(t)$ an incurred a corresponding loss

expert	day 1 am	day 1 pm	day 2 am	day 2 pm	day 3 am	day 3 pm	...	day T am	day T pm
1		f^1_1		f^2_1		f^3_1	...		f^T_1
2		f^1_2		f^2_2		f^3_2	...		f^T_2
3		f^1_3		f^2_3		f^3_3	...		f^T_3
$\vdots$		$\vdots$		$\vdots$		$\vdots$	...		$\vdots$
n		f^1_n		f^2_n		f^3_n	...		f^T_n
(possibly probabilistic) choices:	$i(1)$		$i(2)$		$i(3)$			$i(T)$	

a day's losses are adversarially set at beginning of day but revealed after our choice

$\Rightarrow$ total loss $= \sum_{t=1}^{T} f^t_{i(t)}$

(it is a random variable if the choices are probabilistic)

Ex: select stocks based on opinions of n other people

$\boxed{\text{GOAL: minimize (expected) total loss}}$

Observation: all experts could be idiots so we cannot expect to design a selection strategy that always achieves small total loss

We refine the goal as follows:

minimize the (expected) regret $R = \mathbb{E}\left[\left(\sum_{t=1}^{T} f_{i(t)}^{t}\right)\right] - \left(\min_{i\in[n]} \sum_{t=1}^{T} f_i^t\right)$

(across all adversalial losses)

no need to consider a distribution here bc there is a best expert

That is, we minimize the total loss wrt best expert in hindsight.

Why not minimize $\mathbb{E}\left[\left(\sum_{t=1}^{T} f_{i(t)}^{t}\right)\right] - \left(\sum_{t=1}^{T} \min_{i\in[n]} f_i^t\right)$?

A: We cannot hope for a selection strategy that competes with each day's best expert.

For each t, adversarially set $\begin{cases} \text{loss } 0 \text{ for expert with least probability on day } t \\ \text{loss } 1 \text{ for all other experts} \end{cases}$

This gives $\mathbb{E}\left[\left(\sum_{t=1}^{T} f_{i(t)}^{t}\right)\right] = \sum_{t=1}^{T} \sum_{i=1}^{n} p_i^t f_i^t \geq \sum_{t=1}^{T} \left(1 - \frac{1}{n}\right) = \frac{n-1}{n} \cdot T$

$\sum_{t=1}^{T} \min_{i\in[n]} f_i^t = \sum_{t=1}^{T} 0 = 0$

largest value for the smallest probability

difference is $\frac{n-1}{n} \cdot T$ (it's large)

Q: how to pick expert each day?

temporary simplification: binary losses $f_i^t \in \{\underset{\text{right}}{0}, \underset{\text{wrong}}{1}\}$

Try #1: always pick expert 1 ($i(t)=1\ \forall t$, regardless of losses)

the losses could be

$$\begin{array}{c|cccc} & 1 & 2 & \cdots & T \\ \hline 1 & 1 & 1 & \cdots & 1 \\ 2 & 0 & 0 & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ n & 0 & 0 & \cdots & 0 \end{array}$$

which leads to $R \geq T$

Try #2: choose majority opinion (this is well-defined for binary predictions)

the losses could be

$$\begin{array}{c|cccc} & 1 & 2 & \cdots & T \\ \hline 1 & 0 & 0 & \cdots & 0 \\ 2 & 1 & 1 & \cdots & 1 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ n & 1 & 1 & \cdots & 1 \end{array}$$

which leads to $R \geq T$

Try #3: choose expert at random (intuition is to use randomness to defeat adversarial losses)

the (expected regret) is

$[F_i := \sum_{t=1}^T f_i^t]$

$$\sum_{t=1}^T \left(\sum_{i=1}^n \frac{1}{n} \cdot f_i^t\right) - \left(\min_{i \in [n]} \sum_{t=1}^T f_i^t\right) = \underbrace{\sum_{i=1}^n \frac{1}{n} \cdot F_i}_{\text{average}} - \underbrace{\min_{i\in[n]} F_i}_{\text{minimum}} \leq \frac{n-1}{n} \cdot T$$

The upper bound is tight (eg. $F_1 = 0, F_2 = T, \ldots, F_n = T$ as for the bad weights in Try #2)

Try #4: choose best expert so far (intuition is to take the past into account)
[& break ties lexicographically]

While losses can be adversarially chosen so that every day's best expert so far does poorly, this makes experts overall worse, reducing regret.

But can still arrange losses so that

total loss = T
best expert's loss = T/n

$$R = \frac{n-1}{n} \cdot T \quad \text{(still large)}$$

Here is the example with $n=3$:

expert	day 1 am	day 1 pm	day 1 tot	day 2 am	day 2 pm	day 2 tot	day 3 am	day 3 pm	day 3 tot	day 4 am	day 4 pm	day 4 tot	...
1		1	1		0	1		0	1		1	2	
2		0	0		1	1		0	1		0	1	
3		0	0		0	0		1	1		0	1	
choices:	1			2			3			1			...

The chosen expert has loss 1 each day $\Rightarrow$ total loss is T.

Each expert has loss 1 once every n days, and loss 0 all other days $\Rightarrow$ best expert's loss is T/n.

Note: the example can be tweaked to avoid abusing the tie-breaking rule by relying on fractional losses

Try #5: choose expert according to a weighted majority (this is well-defined for binary predictions)

Fix a parameter ε.

- initialization: set weights $w_1^0, w_2^0, \dots, w_n^0$ to 1
- expert choice at time t:

$$E_A^t = \{ i \mid \text{expert } i \text{ predicts A on am of day } t \}$$
$$E_B^t = \{ i \mid \text{expert } i \text{ predicts B on am of day } t \}$$

if $\sum_{i \in E_A^t} w_i^t \geq \sum_{i \in E_B^t} w_i^t$ then predict A; else predict B

- update: $w_i^{t+1} := w_i^t \cdot (1-\varepsilon)^{f_i^t}$

theorem: $\forall \varepsilon > 0$, $WM(\varepsilon)$ achieves the following guarantee

$$\left(\sum_{i=1}^{T} f_{i(t)}^t \right) \leq 2(1+\varepsilon) \left(\min_{i \in [n]} \sum_{t=1}^{T} f_i^t \right) + \frac{2 \ln(n)}{\varepsilon}$$

The theorem gives a multiplicative guarantee.

But we can do even better!

Multiplicative Weight Updates

(uses past losses and randomness)

⊛ in some references the update is

$$w_i^{t+1} := w_i^t \cdot (1-\varepsilon)^{f_i^t}$$

for which a similar analysis applies

Fix a parameter ε.

- initialization: set weights $w_1^0, w_2^0, \dots, w_n^0$ to 1
- expert choice at time t: choose $i \in [n]$ w.p. $p_i^t := \frac{w_i^t}{\sum_{j=1}^n w_j^t}$
- update: $w_i^{t+1} := w_i^t \cdot (1 - \varepsilon f_i^t)$ ⊛

theorem: $\forall \varepsilon \in (0, \frac{1}{2}]$ MWU(ε) achieves the following (expected) regret:

$$\left(\sum_{t=1}^T \langle p^t, f^t \rangle\right) - \left(\min_{i \in [n]} \sum_{t=1}^T f_i^t\right) \leq \varepsilon \cdot T + \frac{\ln(n)}{\varepsilon}$$

if losses are in $[a,b]$ then the bound becomes $(b-a) \cdot \left(\varepsilon T + \frac{\ln(n)}{\varepsilon}\right)$ by re-scaling

- The guarantee is better than an approx factor of ≈ 2 bc loss of best expert could grow with T.
- The regret per day tends to $\varepsilon + o(1)$ as $T \to \infty$.
- If we know T in advance then we can set $\varepsilon = \sqrt{\ln(n)/T}$ so that $R \leq 2\sqrt{T \ln(n)}$, and the regret per day is $2\sqrt{\frac{\ln(n)}{T}}$, which tends to 0 very quickly.

Proof is based on the potential function $\boxed{\Phi^t := \sum_{i=1}^n w_i^t}$

① **claim:** $\Phi^T \leq n \cdot e^{-\varepsilon \sum_{t=1}^T \langle p^t, f^t \rangle}$

$$\Phi^{t+1} = \sum_{i=1}^n w_i^{t+1} = \sum_{i=1}^n w_i^t \cdot (1 - \varepsilon f_i^t)$$
$$= \sum_{i=1}^n (p_i^t \Phi^t) \cdot (1 - \varepsilon f_i^t) = \Phi^t \sum_{i=1}^n p_i^t \cdot (1 - \varepsilon f_i^t)$$
$$= \Phi^t \cdot \left(\sum_{i=1}^n p_i^t - \varepsilon \sum_{i=1}^n p_i^t f_i^t \right)$$
$$= \Phi^t \cdot (1 - \varepsilon \langle p^t, f^t \rangle)$$
$$\leq \Phi^t \, e^{-\varepsilon \langle p^t, f^t \rangle}$$

if there is an expected loss then the potential drops accordingly

Hence $\Phi^T \leq \Phi^0 \prod_{t=1}^T e^{-\varepsilon \langle p^t, f^t \rangle} = n \cdot e^{-\varepsilon \sum_{t=1}^T \langle p^t, f^t \rangle}$

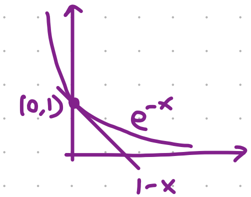
② **claim:** $\forall i \in [n] \quad \Phi^T \geq e^{-\varepsilon \sum_{t=1}^T f_i^t - \varepsilon^2 \sum_{t=1}^T (f_i^t)^2}$

if there is a good expert then Φ^T cannot be too small

$$\Phi^T \geq w_i^T = \prod_{t=1}^T (1 - \varepsilon f_i^t) \geq \prod_{t=1}^T e^{-\varepsilon f_i^t - \varepsilon^2 (f_i^t)^2} = e^{-\varepsilon \sum_{t=1}^T f_i^t - \varepsilon^2 \sum_{t=1}^T (f_i^t)^2}$$

Two inequalities:

ⓐ $1 - x \leq e^{-x}$



ⓑ $1 - x \geq e^{-x-x^2}$ for $x \leq \frac{1}{2}$

By Taylor expansion:

$$\ln(1-x) = -x - \frac{x^2}{2} - \frac{x^3}{3} \ldots$$
$$\geq -x - x^2 \text{ for } x \leq \frac{1}{2}$$

Proof is based on the potential function $\Phi^t := \sum_{i=1}^n w_i^t$

analysis holds even if losses $f^t = (f_i^t)_{i\in[n]}$ are adversarially chosen based on prior losses $f^1, \dots, f^{t-1}$, prior choices $i(1), \dots, i(t-1)$, and selection probabilities (all we need is that f^t is indep of randomness to choose $i(t)$)

① **claim:** $\Phi^T \leq n \cdot e^{-\varepsilon \sum_{t=1}^T \langle p^t, f^t \rangle}$

② **claim:** $\forall i \in [n]$, $\Phi^T \geq e^{-\varepsilon \sum_{t=1}^T f_i^t - \varepsilon^2 \sum_{t=1}^T (f_i^t)^2}$

Combining ① and ② we get that $\forall i \in [n]$:

$$n \cdot e^{-\varepsilon \sum_{t=1}^T \langle p^t, f^t \rangle} \geq \Phi^T \geq e^{-\varepsilon \sum_{t=1}^T f_i^t - \varepsilon^2 \sum_{t=1}^T (f_i^t)^2}$$

take $\ln(\cdot)$

$$\ln(n) - \varepsilon \sum_{t=1}^T \langle p^t, f^t \rangle \geq -\varepsilon \sum_{t=1}^T f_i^t - \varepsilon^2 \sum_{t=1}^T (f_i^t)^2$$

shuffle terms

$$\ln(n) + \varepsilon^2 \sum_{t=1}^T (f_i^t)^2 \geq \varepsilon \left(\sum_{t=1}^T \langle p^t, f^t \rangle - \sum_{t=1}^T f_i^t \right)$$

As the inequality holds $\forall i \in [n]$ we deduce that:

$$\ln(n) + \varepsilon^2 \sum_{t=1}^T (f_i^t)^2 \geq \varepsilon \left(\sum_{t=1}^T \langle p^t, f^t \rangle - \min_{i \in [n]} \sum_{t=1}^T f_i^t \right) = \varepsilon \cdot R$$

As losses are bounded in $[0,1]$ we have $\sum_{t=1}^T (f_i^t)^2 \leq T$. Hence

$$\ln(n) + \varepsilon^2 \cdot T \geq \varepsilon R \Rightarrow \boxed{R \leq \varepsilon \cdot T + \frac{\ln(n)}{\varepsilon}}$$