

# CS 49/149: 21st Century Algorithms (Fall 2018): Lecture 1

Date: 13th September, 2018

Topic: The Experts Problem

Scribe: Deeparnab Chakrabarty

*Disclaimer: These notes have not gone through scrutiny and in all probability contain errors. Please email errors to [deeparnab@dartmouth.edu](mailto:deeparnab@dartmouth.edu).*

---

## 1 The Experts Problem

Suppose you want to predict if it is going to rain or not. Unfortunately, you have no idea of meteorology. But you have  $m$  friends who are *experts* (that is, they have a PhD; it doesn't mean they are infallible) who are willing to tell you their opinions. Your goal is to use their opinions to make a good prediction.

Let's formalize. We denote  $e_i(t) \in \{-1, 1\}$  to be expert  $i$ 's prediction of whether it will rain or not on the  $t$ th day. The index  $i$  ranges from 1 to  $m$ . You observe these  $e_i(t)$ 's, and then you need to make a prediction  $a(t) \in \{-1, 1\}$ . Then the  $t$ th day unfolds, and you get to see  $r(t) \in \{-1, 1\}$  whether it actually rained or not. If  $a(t)$  doesn't match  $r(t)$ , say you pay a dollar – this is your *loss*  $\ell(t)$  on day  $t$ . Thus, formally,  $\ell(t) = (1 - a(t) \cdot r(t))/2$ . We want to devise a strategy, an algorithm, to make the total loss incurred,  $\text{loss} := \sum_{t=1}^T \ell(t)$ , as small as possible. This is the experts problem.

Note that this is not your usual computation problem with a fixed input and output. It has a flavor of a game that one is playing with nature, and in a sense nature (who is responsible for  $r(t)$ ) gets to “play second”. That is, our algorithm's decision has to be made under uncertainty of the outcome. Whenever this happens, it is not clear what “the best algorithm” even means. What is a good benchmark to compare against?

Well, since our algorithm is deriving information from the experts, perhaps we should compare our losses with those of the  $m$  experts. In particular, at time  $t$  define  $\ell_i(t)$  to be 1 if expert  $i$  makes a mistake at time  $i$ , and let  $\text{loss}_i := \sum_{t=1}^T \ell_i(t)$ . But our experts can have differing qualities – which one should we choose our benchmark as? Well, it is clear that the “best expert” (the one with the smallest  $\text{loss}_i$ ) is the most stringent among these benchmarks; let us see if we can design an algorithm which is as good as the best expert.

**The case of the perfect expert.** Let's start with the case there is one “perfect expert”. That is, there is some  $i^*$  with  $\text{loss}_{i^*} = 0$ . If we knew who this perfect expert was, then our strategy is clear – predict whatever  $i^*$  is predicting. Can we quickly recognize this expert? The answer is yes, and the algorithm is simple. It maintains a candidate set  $A$  of active experts; any expert not in  $A$  has made some mistake in the past and thus cannot be the perfect expert. Initially  $A = [m]$ , and we wish to whittle this set down fast. The MAJORITY algorithm is this: always predict what the majority of the experts in  $A$  are predicting (if  $|A|$  is even, break ties arbitrarily). Every time our algorithm makes a mistake, that is, everytime loss increments by +1, we know that at least  $|A|/2$  experts are whittled out. Thus, the maximum number of mistakes MAJORITY makes is  $\log_2 m$ .

**Theorem 1.** In the case of the perfect expert, the MAJORITY algorithm finds this expert making  $\log_2 m$  mistakes.



**Question:** Can any algorithm always find the perfect expert making  $\ll \log_2 m$  queries?

However, perfect experts are mythical. What if we were only promised an expert who is correct 99% of the time? How would MAJORITY perform? Well, not too well as defined since the near-perfect expert could make a mistake on the first day along with the majority and be immediately whittled out. How should we fix this?

Idea 1: Suppose we knew we were playing for  $T$  days. Keep taking MAJORITY till some expert makes  $> 0.01T$  mistakes and then whittle him/her out. Two issues: one, it needs to know that some expert was as good as 99% (how would we know that?). More seriously, it's not a good algorithm – find an example where this algorithm fails badly.

Idea 2: When an expert makes a mistake, instead of whittling them out, just decrease their “importance”. More precisely, every expert has an importance or a weight  $w_i(t)$ . Initially, all the weights are 1; all experts are equal in our eyes. Each day, we go with the *weighted* majority. More precisely, we compute  $\sum_{i=1}^m w_i(t)e_i(t)$  and predict its sign (with 0 considered “positive”). Upon receiving the truth, that is,  $r(t)$ , we *update* the weights as follows: we penalize every expert which makes a mistake at time  $t$  by halving their weight. The full algorithm is described below.

#### WEIGHTED MAJORITY

- Maintain weights  $w_i(\cdot)$  for each expert with  $w_i(1) = 1$  for all  $i$ .
- On days  $t = 1, \dots, T$ :
  - We receive  $e_i(t)$  from each  $i$ .
  - For each prediction  $\{+1, -1\}$  we calculate the total *weight* of experts predicting it, and go with whichever is larger.
  - Then we receive  $r(t)$ . For every  $i$  with  $e_i(t) \neq r(t)$ , we set

$$w_i(t+1) = w_i(t)/2 \tag{1}$$

**Theorem 2.** If there exists an expert which makes at most  $k$  mistakes, then the WEIGHTED MAJORITY algorithm makes at most  $2.41k + O(\log m)$  mistakes.

**Corollary 1.** If there is an expert who is correct at least 99% of the time, then if one plays for  $T \gg \log m$  days, the WEIGHTED MAJORITY is correct  $\geq 96\%$  of the time.

*Proof.* (of Theorem.) The analysis is similar in spirit to the analysis of the MAJORITY algorithm – we focus on the times  $t$  at which the algorithm makes a mistake. Consider such a time  $t$  when  $a(t) \neq r(t)$ . By the algorithm’s design we get to see at this  $t$ ,

$$\sum_{i:e_i(t) \neq r(t)} w_i(t) \geq \sum_{i:e_i(t) = r(t)} w_i(t) \tag{2}$$

Now note that for all experts in the LHS, their weight  $w_i(t+1) = w_i(t)/2$  while for all experts in the right,  $w_i(t+1) = w_i(t)$ . Therefore, the *total weight* scales down.

More precisely, let's define for any  $t$ ,

$$Z(t) := \sum_{i=1}^m w_i(t)$$

For any time  $t$  at which WEIGHTED MAJORITY makes a mistake, (2) implies  $\sum_{i:e_i(t)=r(t)} w_i(t) \leq Z(t)/2$ . Furthermore, we get

$$\begin{aligned} Z(t+1) &= \sum_{i:e_i(t) \neq r(t)} w_i(t+1) + \sum_{i:e_i(t)=r(t)} w_i(t+1) \\ &= \sum_{i:e_i(t) \neq r(t)} w_i(t)/2 + \sum_{i:e_i(t)=r(t)} w_i(t) \\ &= Z(t)/2 + \frac{1}{2} \sum_{i:e_i(t)=r(t)} w_i(t) \\ &\leq 3Z(t)/4 \end{aligned}$$

The above was for the times when our algorithm makes a mistake. What about the times when it doesn't? Well, since the weights never *increase*, we have the trivial inequality  $Z(t+1) \leq Z(t)$ . Thus, if our algorithm makes  $\ell$  mistakes (that is,  $\ell = \text{loss}$ ), after  $T$  rounds we get

$$Z(T) \leq (3/4)^\ell \cdot Z(1) = (3/4)^\ell \cdot m \quad (3)$$

So if our algorithm makes a lot of mistakes, the *potential*  $Z$  falls rapidly. Why is this at all useful in comparing with the best expert's loss? This is the second insight: if there is some expert making only  $k$  mistakes, then his weight at the end is *at least*  $(1/2)^k$ . Even if all the other experts are duds making tons and tons of mistakes, this expert forces

$$Z(T) \geq (0.5)^k \quad (4)$$

Now rest is arithmetic. From (3) and (4) we get

$$(1/2)^k \leq m \cdot (3/4)^\ell$$

Taking logs base 2 and swapping signs, we get

$$k \geq -\log_2 m + \ell \log_2(4/3)$$

and then changing sides, we get

$$\ell \leq \frac{1}{\log_2(4/3)} \cdot (k + \log_2 m)$$

completing the proof. □

Can we do better? Well surely what was special about "halving". What if instead we scaled down by some other factor  $(1 - \eta)$ . That is, consider WEIGHTED MAJORITY where (5) is replaced by

$$w_i(t+1) = w_i(t) \cdot (1 - \eta)$$

**Theorem 3.** If there exists an expert which makes at most  $k$  mistakes, then the WEIGHTED MAJORITY algorithm with any parameter  $0 \leq \eta \leq \frac{1}{2}$  makes at most  $(2 + \eta)k + O\left(\frac{\log m}{\eta}\right)$  mistakes.

*Proof.* The proof is similar to the one above with a little more arithmetic jugglery. The inequality corresponding to (3) becomes (check this!)

$$Z(T) \leq \left(1 - \frac{\eta}{2}\right)^\ell \cdot m$$

Note that when  $\eta = 1/2$ , we get (3). Similarly, (4) becomes (check this!)

$$Z(T) \geq (1 - \eta)^k$$

Therefore, together we get

$$(1 - \eta)^k \leq m \cdot (1 - \eta/2)^\ell$$

which in turn implies

$$k \ln(1 - \eta) \leq \ln m + \ell \ln(1 - \eta/2)$$


Finally, we use the fact<sup>1</sup> that for any  $|x| \leq \frac{1}{2}$ , we have  $-(x + x^2) \leq \ln(1 - x) \leq -x$  giving us

$$-k(\eta + \eta^2) \leq \ln m - \ell\eta/2$$

whenever  $\eta \leq \frac{1}{2}$ . Moving things around we get

$$\ell \leq 2(1 + \eta)k + \frac{2 \ln m}{\eta}$$

□

So we got the 2.41 down to “arbitrarily” close to 2. Can we do better? Turns out that deterministic algorithms can’t do better. 

**Question:** Show that given any deterministic algorithm can’t get a better than factor 2 approximation. More precisely, for any strategy and for any  $T$ , show a collection of expert answers and “truths” such that the best expert makes  $\leq \delta$  mistakes but the algorithm makes  $2\delta$  mistakes. In fact, this is true even with just two experts.

## 1.1 Getting arbitrarily close with randomization

Suppose now are algorithm is allowed to toss coins. That is, the quantity  $a(t)$  is a random variable which  $-1$  with a certain probability and  $+1$  with the remainder. The *expected* number of mistakes made by the algorithm at time  $t$  is therefore the *probability* with which  $a(t) \neq r(t)$ . The total expected loss is the sum of these expectations (recall, linearity of expectation).

What would be the “natural” algorithm if we allowed randomization? Note that in the WEIGHTED MAJORITY if 50.001% of the weight voted for  $+1$  and 49.999% voted for  $-1$ , the algorithm went with  $+1$ . Even with this slim lead. The “fairer” randomized algorithm that suggests itself is that

<sup>1</sup>Just plot the functions and see. One can also prove this analytically.

we should say +1 with probability 50.001% and −1 with the remainder probability. Turns out, this algorithm leads us arbitrarily close to the best expert!

A more convenient way of looking at the RANDOMIZED WEIGHTED MAJORITY algorithm is by thinking of picking a random expert and going with their decision. That is, given the weights we pick an expert  $i$  proportional to  $w_i(t)$ , and then predict  $a(t) = e_i(t)$ . Observe that this is the same fair algorithm described above.

#### RANDOMIZED WEIGHTED MAJORITY

- Maintain weights  $w_i()$  for each expert with  $w_i(1) = 1$  for all  $i$ .
- On days  $t = 1, \dots, T$ :
  - Select expert  $i$  with probability  $\propto w_i(t)$ .
  - Predict  $a(t) = e_i(t)$ .
  - Then we receive  $r(t)$ . For every  $i$  with  $e_i(t) \neq r(t)$ , we set

$$w_i(t+1) = w_i(t) \cdot (1 - \eta) \quad (5)$$

where  $\eta$  is a parameter between 0 and 1/2.

**Theorem 4.** If there is an expert making at most  $k$  mistakes, then the *expected* number of mistakes made by the RANDOMIZED WEIGHTED MAJORITY algorithm is

$$\mathbf{Exp}[\text{loss}] \leq (1 + \eta) \cdot k + \frac{O(\log m)}{\eta}$$

*Proof.* The proof idea is similar to the deterministic proof at a high level. As before, let  $Z(t) := \sum_{i=1}^m w_i(t)$ . Note we have (4) as is; to repeat

$$Z(T) \geq (1 - \eta)^k \geq e^{-k(\eta + \eta^2)} \quad (6)$$

We have used the old inequality  $\ln(1 - x) \geq -x - x^2$  for all  $|x| \leq \frac{1}{2}$ , above.

Now the greedy rule is no longer true. Let  $\text{loss}_t$  be the random variable indicating whether  $a(t) \neq r(t)$ . Note that,

$$\mathbf{Exp}[\text{loss}_t] = \sum_{i:r(t) \neq e_i(t)} \Pr[i \text{ selected}] = \frac{1}{Z(t)} \sum_{i:r(t) \neq e_i(t)} w_i(t) \quad (7)$$

Now, note that the difference in the ‘potential’ is

$$Z(t+1) - Z(t) = -\eta \sum_{i:r(t) \neq e_i(t)} w_i(t) = -\eta Z(t) \mathbf{Exp}[\text{loss}_t]$$

giving us

$$Z(t+1) \leq Z(t) \cdot (1 - \eta \mathbf{Exp}[\text{loss}_t]) \leq Z(t) \cdot \exp(-\eta \mathbf{Exp}[\text{loss}_t])$$

We have used  $1 + x \leq e^x$  for all  $x$  above. Why? Because it allows us to “telescope”. We get

$$Z(T) \leq m \cdot \exp\left(-\eta \sum_{t=1}^T \mathbf{Exp}[\text{loss}_t]\right) = m \cdot \exp(-\eta \cdot \mathbf{Exp}[\text{loss}]) \quad (8)$$

Taking natural logs on (8) and (6), gives

$$-(\eta + \eta^2)k \leq \ln m - \eta \mathbf{Exp}[\text{loss}]$$

The theorem follows by “moving stuff around”.

□