Learning from Experts

Expanding your Algorithmic Toolbox

Paul Beaujean[†]

[†]IMT/OLN/NMP/TRM

Operations Research Tuesdays · March 17, 2015

Table of Contents

The Algorithm Designer

Our usual toolkit

- Linear Optimization
- Dynamic Programming
- Greedy Algorithms
- Metaheuristics...
- Classification
- Clustering

Today

General algorithmic technique applicable to optimization, machine learning, and online decision making.

Learning from Experts

How can I apply this framework?

$$\mathcal{A} = \{a \mid \ell_a^t \in [-1, 1]\}$$

- a can be:
  - a strategy
  - an algorithm
  - a prediction
- t is a round (discrete time)

Notation

Action a must have:

- bounded loss: ℓ_a^t
- bounded gain: g_a^t

Gain, Loss, Regret

- Loss incurred by action a over a certain number of rounds:

$$L_a = \sum_t \ell_a^t$$

- Loss incurred by the best action in hindsight:

$$L_* = \min_{a \in \mathcal{A}} L_a$$

A surprising result

For a certain algorithm with expected loss L (or gain G):

Loss bound

$$L \leq L_a + \epsilon \left| L_a \right| + \frac{\ln |\mathcal{A}|}{\epsilon}, \ \forall a \in \mathcal{A}$$

Gain guarantee

$$G \geq G_a - \epsilon \left| G_a \right| - \frac{\ln |\mathcal{A}|}{\epsilon}, \ \forall a \in \mathcal{A}$$

Multiplicative weights

Compounding loss!

Initialization

$$\lambda_a^1 = 1, \ \forall a \in \mathcal{A}$$

Sequential Updates

- Choose action a with probability $p_a^t = \lambda_a^t / \Phi^t$
- Compound loss

$$\lambda_a^{t+1} = \lambda_a^t (1 - \epsilon \, \ell_a^t), \ \forall a \in \mathcal{A}$$

where $\Phi^t = \sum_a \lambda_a^t$

Multiplicative weights algorithm

Expected loss per round

$$L^t = \mathbb{E}(\ell^t) = \sum_a p_a^t \ell_a^t$$

Expected loss

$$L = \sum_t L^t$$

Let's prove it!

Φ: a potential function argument

Comparing Φ^t and L^t

$$\Phi^t = \sum_a \lambda_a^t$$

$$\Phi^{t+1} = \Phi^t - \epsilon \sum_a \lambda_a^t \ell_a^t$$

Recall that $\lambda_a^t = \Phi^t p_a^t$

$$\Phi^{t+1} = \Phi^t - \epsilon\ \Phi^t \sum_a p_a^t \ell_a^t$$

$$\Phi^{t+1} = \Phi^t(1 - \epsilon L^t)$$

Taylor expansion

$$\Phi^{t+1} \leq \Phi^t \mathrm{e}^{-\epsilon L^t}$$

An upper bound on the potential

By simple induction

Given

$$\Phi^1 = \sum_a 1 = |\mathcal{A}|$$

and

$$\Phi^{t+1} \leq \Phi^t \mathrm{e}^{-\epsilon L^t}$$

Let L be the expected loss after T rounds:

$$\Phi^{T+1} \leq |\mathcal{A}| \mathrm{e}^{-\epsilon L}$$

A lower bound on the compound loss

Definition of the compound loss

For each action a,

- Initialization

$$\lambda_a^1 = 1$$

- Recurrence

$$\lambda_a^{t+1} = \lambda_a^t (1 - \epsilon \; \ell_a^t)$$

$$\lambda_a^{T+1} = \prod_t^T (1 - \epsilon \; \ell_a^t)$$

More Taylor expansions

$$(1+\epsilon)^{-x} \leq (1 - \epsilon x), \;\; x \in [-1, 0]$$

$$(1-\epsilon)^{x} \leq (1 - \epsilon x), \;\; x \in [0, 1]$$

A lower bound on the compound loss

More Taylor expansions

$$(1+\epsilon)^{-x} \leq (1-\epsilon x),\ x \in [-1,0]$$
$$(1-\epsilon)^{x} \leq (1-\epsilon x),\ x \in [0,1]$$

Lower bounding λ

$$(1+\epsilon)^{-\sum_{<0} \ell_a^t}(1-\epsilon)^{\sum_{\geq 0} \ell_a^t} \leq \lambda_a^{T+1}$$

Note

If we assume that $\epsilon < \frac{1}{2}$ then $\lambda_a^t > 0$

The potential is higher than each compound loss

More Taylor expansions

Since for each action a we have $\lambda_a^t > 0$

$$\lambda_a^{T+1} \leq \sum_a \lambda_a^{T+1} = \Phi^{T+1}$$

Putting it all together

$$(1+\epsilon)^{-\sum_{<0} \ell_a^t}(1-\epsilon)^{\sum_{\geq 0} \ell_a^t} \leq \lambda_a^{T+1} \tag{1}$$

$$\lambda_a^{T+1} \leq \Phi^{T+1} \tag{2}$$

$$\Phi^{T+1} \leq |\mathcal{A}|\mathrm{e}^{-\epsilon L} \tag{3}$$

$$(1+\epsilon)^{-\sum_{<0} \ell_a^t}(1-\epsilon)^{\sum_{\geq 0} \ell_a^t} \leq |\mathcal{A}|\mathrm{e}^{-\epsilon L}$$

Wrapping it up

Bounding L

$$(1+\epsilon)^{-\sum_{<0} \ell_a^t}(1-\epsilon)^{\sum_{\geq 0} \ell_a^t} \leq |\mathcal{A}|\mathrm{e}^{-\epsilon L}$$

Taking the ln gives us:

$$-\sum_{<0} \ell_a^t \ln(1+\epsilon) + \sum_{\geq 0} \ell_a^t \ln(1-\epsilon) \leq \ln|\mathcal{A}| - \epsilon L$$

$$L \leq \sum_{<0} \ell_a^t \frac{\ln(1+\epsilon)}{\epsilon} - \sum_{\geq 0} \ell_a^t \frac{\ln(1-\epsilon)}{\epsilon} + \frac{\ln|\mathcal{A}|}{\epsilon}$$

Our first second-order Taylor expansions

$$\ln(1+\epsilon) \geq \epsilon - \epsilon^2$$

$$-\ln(1-\epsilon) \leq \epsilon + \epsilon^2$$

Wrapping it up

Using $\frac{\ln(1+\epsilon)}{\epsilon} \geq 1-\epsilon$ and $-\frac{\ln(1-\epsilon)}{\epsilon} \leq 1+\epsilon$ we get:

Done!

$$L \leq \frac{1}{\epsilon} \sum_{<0} \ell_a^t (1-\epsilon) + \sum_{\geq 0} \ell_a^t (1+\epsilon) + \frac{\ln |\mathcal{A}|}{\epsilon}$$

$$L \leq \sum_t \ell_a^t + \epsilon \underbrace{(-\sum_{<0} \ell_a^t + \sum_{\geq 0} \ell_a^t)}_{\sum_t |\ell_a^t|} + \frac{\ln |\mathcal{A}|}{\epsilon}$$

With $|L_a| = \sum_t |\ell_a^t|$, we obtain:

$$L \leq L_a + \epsilon\, |L_a| + \frac{\ln |\mathcal{A}|}{\epsilon},\ \forall a \in \mathcal{A}$$

Regret

What does the bound on L mean?

$$L \leq L_a + \epsilon\, |L_a| + \frac{\ln |\mathcal{A}|}{\epsilon},\ \forall a \in \mathcal{A}$$

In particular, with regard to the best action in hindsight:

$$L \leq L_* + \epsilon\, |L_*| + \frac{\ln |\mathcal{A}|}{\epsilon}$$

Average expected regret

Over T rounds, we define the average expected regret by:

$$R = \frac{L - L_*}{T}$$

Not feeling much regret

Regret bound

Rewriting $|L_*|$ as $||\ell_{a^*}||_1$, we get:

$$L - L_* \leq \epsilon \; ||\ell_{a^*}||_1 + \frac{\ln |\mathcal{A}|}{\epsilon}$$

The trivial norm bound $||\ell_{a^*}||_1 \leq T$ gives:

$$L - L_* \leq \epsilon T + \frac{\ln |\mathcal{A}|}{\epsilon}$$

$$R \leq \epsilon + \frac{\ln |\mathcal{A}|}{\epsilon T}$$

How long until I feel no regret?

$$T = \frac{\ln |\mathcal{A}|}{\epsilon^2} \implies R \leq 2\epsilon$$

Applications

Wide variety of domains

- Stock market prediction
- Online Convex Optimization
- Boosting (Ensemble Learning)
- Approximate Linear Optimization
- Multicommodity flow problems $\leftarrow$ today
- and many more! competitive online algorithms, SDP, approximation algorithms for NP-hard problems, ...

Applicable yes but efficient?

Analysis of the algorithm is tight.
Given the same assumptions, no algorithm produces better bounds.

The multicommodity flow problem

Given a capacitated graph together with a set of origin-destination pairs (representing a source and a target).

Problem

Find the maximum amount of flow between origin-destination pairs that satisfies capacity constraints

Complexity

- If the flow is integral, NP-hard.
- If the flow is fractional, can be cast as a large linear program so it is $\in$ P. $\leftarrow$ today
- If there is only one origin-destination pair, reduces to the simple maximum flow.

The maximum multicommodity flow as a LP

A path formulation

$$(\Pi)\begin{cases}\max & \displaystyle\sum_{p\in\mathcal{P}} f_p \\ \text{s.t.} & \displaystyle\sum_{p\ni a} f_p \leq c_a,\ \forall a \in A\ (l_a) \\ & f \in \mathbb{R}_+^{|\mathcal{P}|}\end{cases}$$

$$(\Delta)\begin{cases}\min & \displaystyle\sum_{a\in A} c_a l_a \\ \text{s.t.} & \displaystyle\sum_{a\in p} l_a \geq 1,\ \forall p \in \mathcal{P}\ (f_p) \\ & l \in \mathbb{R}_+^{|A|}\end{cases}$$

Learning from expert arcs

Using the gain guarantee

$$G \geq G_a - \epsilon \, |G_a| - \frac{\ln |\mathcal{A}|}{\epsilon}, \ \forall a \in \mathcal{A}$$

Mapping concepts

actions	$\mathcal{A}$	$\rightarrow$	A	arcs
gain	g_a	$\rightarrow$	f_a/c_a	network utilization
compound gain	l_a	$\rightarrow$	l_a	length (compound congestion)

Bounding network utilization

Can f_a/c_a be bounded?

Yes!

If the flow routed at each round does not exceed the minimum capacity of the path.

We can apply the multiplicative weights algorithm

- A: finite set of arcs
- $g_a = f_a/c_a \in [0, 1]$

A primal-dual algorithm based on multiplicative weights

Initialization

- For each arc $a \in A$, set length $l_a^1 = 1$

At round t

- Compute the shortest path p^t relative to the length l^t among all paths in $\mathcal{P}$ and choose all arcs a on that path.
- The gain on each arc is given relative to the flow that saturates p^t:

$$f^t = \min_{a \in p^*} c_a$$

$$g_a^t = \begin{cases} f^t / c_a & \text{if } a \in p^t \\ 0 & \text{otherwise} \end{cases}$$

- Compound gain

$$l_a^{t+1} = l_a^t (1 + \epsilon \, g_a^t)$$

Analysis

Relating G_* and the maximum network utilization

Since $g_a^t \in [0, 1]$:

$$G \geq (1 - \epsilon) G_* - \frac{\ln |A|}{\epsilon}$$

with

$$G_* = \max_{a \in A} \sum_t f_a^t / c_a$$

Which means that G_* is the value of the edge with maximum utilization in the network.

Analysis

Relating G and the approximation

$$G \geq (1-\epsilon)G_* - \frac{\ln |A|}{\epsilon}$$

We set the probabilities $p_a^t = l_a^t/\Phi^t$ a posteriori, which gives:

$$G = \sum_t \sum_{a \in A} g_a^t \frac{l_a^t}{\Phi^t}$$

$$G = \sum_t \sum_{a \in p^t} g_a^t \frac{l_a^t}{\Phi^t}$$

$$G = \sum_t f^t \frac{\sum_{a \in p^t} l_a^t / c_a}{\Phi^t}$$

Upper bounded by the optimal flow

Relevant facts

Abstracting away t, we have:

$$\frac{\sum_{a \in p} l_a / c_a}{\Phi}$$

Consider the following:

- p^* is a shortest path w.r.t. l_a / c_a
- $F^* = \sum_q f_q^*$ is an optimal (feasible) flow

Relating G and the approximation

$$\frac{\sum_{a \in p^*} l_a / c_a}{\sum_a l_a} \leq \frac{\sum_{a \in p^*} l_a / c_a}{\sum_a l_a \sum_{q \ni a} \frac{f_q^*}{c_a}} = \frac{\sum_{a \in p^*} l_a / c_a}{\sum_q f_q^* \sum_{a \in q} l_a / c_a}$$

Upper bounded by the optimal flow

Relating G and the approximation

$$\frac{1}{\sum_q f_q^*} \cdot \frac{\sum_{a \in p^*} l_a/c_a}{\sum_{a \in q} l_a/c_a} = \frac{1}{F^*} \cdot \frac{\sum_{a \in p^*} l_a/c_a}{\sum_{a \in q} l_a/c_a} \leq \frac{1}{F^*}$$

which gives the following bound:

$$\frac{\sum_{a \in p^t} l_a^t/c_a}{\Phi^t} \leq \frac{1}{F^*}, \ \forall t$$

Upper bounded by the optimal flow

Relating G and the approximation

If we call $F = \sum_t f^t$ the flow given by our algorithm:

$$G = \sum_t f^t \frac{\sum_{a \in p^t} l_a^t / c_a}{\Phi^t} \leq \sum_t f^t \, \frac{1}{F^*} = \frac{F}{F^*}$$

$$\frac{F}{F^*} \geq G \geq (1 - \epsilon) G_* - \frac{\ln |A|}{\epsilon}$$

Gain guarantee $\rightarrow$ Approximation guarantee

Scaling

Now we can scale down our flow by G_* to obtain $\hat{F} = F/G_*$:

$$\frac{\hat{F}}{F^*} \geq (1-\epsilon) - \frac{\ln|A|}{\epsilon C}$$

Approximation guarantee

$$G_* = \frac{\ln|A|}{\epsilon^2}$$

implies that

$$\hat{F} \geq (1-2\epsilon)F^*$$

our algorithm returns a solution with arbitrary precision.

In polynomial time?

FPTAS

An algorithm returning a solution with arbitrary precision ϵ

- in polynomial time w.r.t the instance size
- in polynomial time w.r.t the precision ϵ^{-1}

How many iterations?

- Each edge can receive congesting flow at most $O(\ln |A| \ \epsilon^{-2})$ times.
- $|A|$ edges in total

$$O(m \log m \ \epsilon^{-2}) \text{ iterations.}$$

Cost per iteration?

The cost of finding k shortest paths where k is the number of origin-destination pairs:

$$O(km)$$

Conclusion

Total runtime

$$O(km^2 \log m \ \epsilon^{-2})$$

Application of the multiplicative weights

Practical benefits:

- simple implementation (16 lines of Python)
- very fast for $\epsilon \approx 10^{-1}$
- simple analysis
- adaptable to any convex setting!

Any questions?

Thank you!

Applications

Wide variety of domains

- Stock market prediction $\leftarrow$ today?
- ...

Setting

- $\mathcal{A}$: financial experts predicting {up, down}
- $\ell_a^t = \begin{cases} 1 & \text{if expert } a \text{ got it wrong at round } t \\ 0 & \text{otherwise} \end{cases}$
- L, L_a, L_*: (expected) number of mistakes

Result

$$L \leq (1+\epsilon)L_* + \frac{\ln |\mathcal{A}|}{\epsilon}$$

Also available in $[-\rho, \rho]$!

Weaker bound on the loss

$$\ell_a^t \in [-\rho, \rho]$$

How long until I feel no regret?

Roughly, to achieve arbitrary regret of ϵ:

$$O(\rho^2 \ln |\mathcal{A}| \ \epsilon^{-2})\text{iterations}$$

and your runtime per iteration might depend on the width too!

Bibliography

Sanjeev Arora, Elad Hazan and Satyen Kale
"The Multiplicative Weights Update Method:
A Meta-Algorithm and Applications".
Theory of Computing 2012.