Algorithms and Uncertainty, Summer 2020	Lecture 22 (5 pages)
Max-Flow via Experts	
Thomas Kesselheim	Last Update: July 13, 2020

Today, we will get to know another very surprising application of the experts framework. We will use it to solve the Maximum-Flow Problem. Our algorithm will be slow but it follows the same pattern that the fastest known algorithms for this problem use.

1 Max-Flow Problem

We are given a graph $G = (V, E)$ with edge capacities $(c_e)_{e \in E}$ and a dedicated source node $s \in V$ and sink node $t \in V$. Let $\mathcal{P}$ be the set of all paths from s to t. Our goal is to assign flow values $(x_P)_{P \in \mathcal{P}}$ to the s-t-paths such that $x_P \geq 0$ for all P, no edge has more flow than its capacity, i.e., $\sum_{P: e \in P} x_P \leq c_e$ for all $e \in E$, and $\sum_{P \in \mathcal{P}} x_P$ is maximized.

This problem can also be stated as a linear program as follows.

$$\begin{aligned} \text{maximize} \quad & \sum_{P \in \mathcal{P}} x_P \\ \text{subject to} \quad & \sum_{P: e \in P} x_P \leq c_e && \text{for all } e \in E \\ & x_P \geq 0 && \text{for all } P \in \mathcal{P} \end{aligned}$$

2 Recap: No-Regret Learning

Let us quickly recap the framework of no-regret learning. We rephrase it slightly to better fit our needs for today. There are m actions (experts) we can choose from in every step. There is a sequence of initially unknown *gain vectors* $g^{(1)}, \ldots, g^{(T)}$. Choosing action i in step t gives gain $g_i^{(t)} \in [0, 1]$. In step t, the algorithm first chooses a probability vector $y^{(t)}$, then it incurs gain $g_{\text{Alg}}^{(t)} = \sum_{i=1}^{m} y_i^{(t)} g_i^{(t)}$ and gets to know the entire vector $g^{(t)}$. To avoid any confusion with the paths, we call the probability vector $y^{(t)}$ today.

The *regret* of the algorithm is defined as

$$\text{Regret}^{(T)} = G_{\max}^{(T)} - G_{\text{Alg}}^{(T)} ,$$

where $G_{\max}^{(T)} = \max_i \sum_{t=1}^{T} g_i^{(t)}$ and $G_{\text{Alg}}^{(T)} = \sum_{t=1}^{T} g_{\text{Alg}}^{(t)} = \sum_{t=1}^{T} \sum_{i=1}^{m} y_i^{(t)} g_i^{(t)}$.

The Multiplicative Weights algorithm guarantees

$$G_{\text{Alg}}^{(T)} \geq (1 - \eta) G_{\max}^{(T)} - \frac{\ln m}{\eta} .$$

So, $\text{Regret}^{(T)} \leq \eta G_{\max}^{(T)} + \frac{\ln m}{\eta}$.

3 Algorithm Intuition

We design an algorithm based on the experts framework. It is, indeed, more or less the same algorithm that was proposed by Garg and Könemann, although they actually do not talk about regret. The algorithm actually works, just as it is, for multi-commodity flow.

The idea behind the algorithm is simple but maybe not intuitive. Like many other flow algorithms, we choose shortest paths from s to t and route as much flow along these edges as possible. The Edmonds-Karp algorithm chooses a path that minimizes the number of edges and then changes the network to a residual network. Our algorithm is different: It changes the lengths of the edges. At this point, the experts algorithm comes into play: We let it define the edge lengths. This is done by considering each edge as an expert and the probability that it puts on an expert as the respective edge length.

4 Flows and Edge Lengths

There is an important connection between edge lengths and flows, which we state in the following necessary condition for the existence of a flow.[1]

Lemma 22.1. *If there is a flow of value F^*, then for all choices of edge lengths $(y_e)_{e\in E}$ with $\sum_{e\in E} y_e = 1$ there is a path P such that $\sum_{e\in P} \frac{y_e}{c_e} \leq \frac{1}{F^*}$.*

Proof. For any feasible LP solution x we have

$$\sum_{P:e\in P} \frac{1}{c_e} x_P \leq 1 \qquad \text{for all } e \in E \ .$$

This also implies

$$\sum_{e\in E} y_e \sum_{P:e\in P} \frac{1}{c_e} x_P \leq \sum_{e\in E} y_e = 1 \ .$$

We can also reorder the left-hand side to

$$\sum_{e\in E} y_e \sum_{P:e\in P} \frac{1}{c_e} x_P = \sum_{P\in\mathcal{P}} \left(\sum_{e\in P} \frac{y_e}{c_e} \right) x_P \ .$$

Now suppose that $\sum_{e\in P} \frac{y_e}{c_e} > \frac{1}{F^*}$ for all paths P. Then this immediately implies that also

$$\sum_{P\in\mathcal{P}} x_P < F^* \sum_{P\in\mathcal{P}} \left(\sum_{e\in P} \frac{y_e}{c_e} \right) x_P \leq F^* \ .$$

So, there could be no flow of value F^*. □

Note that finding such a path algorithmically is easy: Just compute the shortest path.

Our algorithm uses Lemma 22.1 as follows. An experts algorithm sets edge lengths (which are expressed through a probability distribution over edges). This way, it tries to "convince" us that there is no flow of value F^*. We then compute the shortest path and this way show that the edge lengths are no counterexample. The expert algorithm's gain is the length of the path that we found and used. Then, the experts algorithm updates the edge lengths based on which path we chose.

[1] Actually, it is not only necessary but also sufficient. More on this later.

5 Algorithm

We now formally define the algorithm. We use an arbitrary experts algorithm as a subroutine.

- For $t = 1, \ldots, T$
  - Get probability distribution $y^{(t)}$ from the experts algorithm.
  - Compute $P^{(t)}$ as the shortest path with edge lengths $\frac{y_e^{(t)}}{c_e}$
  - Let $c^{(t)} = \min_{e \in P^{(t)}} c_e$
  - Let $(x_P^{(t)})_{P \in \mathcal{P}}$ be a vector such that $x_{P^{(t)}} = c^{(t)}$ and $x_P = 0$ for $P \neq P^{(t)}$.
  - Return $g^{(t)}$ back to the experts algorithm, where

$$g_e^{(t)} = \begin{cases} \frac{c^{(t)}}{c_e} & \text{if } e \in P^{(t)} \\ 0 & \text{otherwise} \end{cases}$$

- Compute $\bar{x} = \sum_{t=1}^T x^{(t)}$, $G_{\max}^{(T)} = \max_{e \in E} \sum_{t=1}^T g_e^{(t)}$
- Return $x = \frac{1}{G_{\max}^{(T)}} \bar{x}$

Interestingly, using any no-regret algorithm, this algorithm always computes a $1-\epsilon$-approximate flow if the number of iterations, T, is chosen large enough.

Lemma 22.2. *The algorithm computes a feasible flow x.*

Proof. Note that for every edge $e \in E$

$$\sum_{t=1}^T g_e^{(t)} = \sum_{t: e \in P^{(t)}} \frac{c^{(t)}}{c_e} = \frac{1}{c_e} \sum_{t: e \in P^{(t)}} c^{(t)} = \frac{1}{c_e} \sum_{t=1}^T \sum_{P: e \in P} x_P^{(t)} = \frac{1}{c_e} \sum_{P: e \in P} \bar{x}_P \quad .$$

This also means

$$G_{\max}^{(T)} = \max_{e \in E} \frac{1}{c_e} \sum_{P: e \in P} \bar{x}_P \quad .$$

So $G_{\max}^{(T)}$ is exactly the maximum factor by which $\bar{x}$ exceeds an edge capacity. Therefore, it is clear that the flow x is feasible. □

Lemma 22.3. *The flow x has value at least $F^*(1 - \frac{1}{G_{\max}^{(T)}} \mathrm{Regret}^{(T)})$, where F^* is the value of an optimal flow.*

Proof. First, we will argue that the expert algorithm's gain cannot be high because we always choose the shortest path with respect to the current weights. This lets us upper-bound the algorithm's gain in terms of the flow that we compute. Afterwards, we use the no-regret property of the experts algorithm, which gives us a lower bound on its gain.

Consider any step t. The expert algorithm's gain in this step is given by

$$g_{\mathrm{Alg}}^{(t)} = \sum_{e \in E} y_e^{(t)} g_e^{(t)} = \sum_{e \in P^{(t)}} y_e^{(t)} \frac{c^{(t)}}{c_e} = c^{(t)} \sum_{e \in P^{(t)}} \frac{y_e^{(t)}}{c_e} \quad .$$

Recall that $P^{(t)}$ was is a shortest path with respect to edge lengths $\left(\frac{y_e^{(t)}}{c_e}\right)_{e \in E}$. So, by Lemma 22.1,

$$\sum_{e \in P^{(t)}} \frac{y_e^{(t)}}{c_e} \leq \frac{1}{F^*} \quad ,$$

meaning that $g_{\text{Alg}}^{(t)} \leq \frac{c^{(t)}}{F^*}$. Using that

$$\sum_{t=1}^{T} c^{(t)} = \sum_{P \in \mathcal{P}} \bar{x}_P \quad ,$$

we observe that

$$G_{\text{Alg}}^{(T)} = \sum_{t=1}^{T} g_{\text{Alg}}^{(t)} \leq \frac{1}{F^*} \sum_{P \in \mathcal{P}} \bar{x}_P = \frac{G_{\max}^{(T)}}{F^*} \sum_{P \in \mathcal{P}} x_P \quad .$$

Recall that $\text{Regret}^{(T)} = G_{\max}^{(T)} - G_{\text{Alg}}^{(T)}$. So, this gives us

$$\frac{G_{\max}^{(T)}}{F^*} \sum_{P \in \mathcal{P}} x_P \geq G_{\max}^{(T)} - \text{Regret}^{(T)}$$

and so

$$\sum_{P \in \mathcal{P}} x_P \geq F^* \left(1 - \frac{\text{Regret}^{(T)}}{G_{\max}^{(T)}}\right) \quad . \qquad \square$$

Note that this bound only is meaningful if $G_{\max}^{(T)}$ is large. Fortunately, this is true in our case.

Lemma 22.4. *The gain vectors $g^{(1)}, \ldots, g^{(T)}$ generated by the algorithm fulfill*

$$G_{\max}^{(T)} \geq \frac{T}{m} \quad .$$

Proof. Observe that in each step t there is an edge e such that $g_e^{(t)} = 1$, therefore

$$G_{\max}^{(T)} = \max_{e \in E} \sum_{t=1}^{T} g_e^{(t)} \geq \frac{1}{m} \sum_{e \in E} \sum_{t=1}^{T} g_e^{(t)} \geq \frac{T}{m} \quad . \qquad \square$$

If we combine these lemmas, then as long as we use a no-regret algorithm, that is, $\text{Regret}^{(T)} = o(T)$, then the flow value approaches F^* asymptotically for larger and larger T.

6 Guarantee with Multiplicative Weights

Let us now derive a quantitative bound if we use Multiplicative Weights. It actually pays off to be a little careful and to not just use the $O(\sqrt{T \log m})$ regret guarantee. Recall that the regret guarantee in case of m experts is

$$\text{Regret}^{(T)} \leq \eta G_{\max}^{(T)} + \frac{\ln m}{\eta} \quad ,$$

so the above guarantee becomes

$$\sum_{P \in \mathcal{P}} x_P \geq F^* \left(1 - \eta - \frac{1}{G_{\max}^{(T)}} \frac{\ln m}{\eta} \right) \geq F^* \left(1 - \eta - \frac{m}{T} \frac{\ln m}{\eta} \right) \quad .$$

If we choose $\eta = \frac{\epsilon}{2}$ and $T = \frac{2}{\epsilon} m \ln m$, then $\sum_{P \in \mathcal{P}} x_P \geq F^*(1 - \epsilon)$.

Theorem 22.5. *With Multiplicative Weights, the algorithm computes a $(1 - \epsilon)$-approximate flow using $\frac{1}{2\epsilon} m \ln m$ shortest-path computations. Its overall running time is $O(\frac{1}{\epsilon} m^2 \ln m)$.*

7 What is really happening?

One may wonder: Why does this work? As often, the answer is simple and complicated at the same time: It is because of strong LP duality. The dual of the flow LP (in the path formulation from above) is:

$$\begin{aligned} \text{minimize} \quad & \sum_{e \in E} c_e z_e & \\ \text{subject to} \quad & \sum_{e \in P} z_e \geq 1 & \text{for all } P \in \mathcal{P} \\ & z_e \geq 0 & \text{for all } e \in E \end{aligned}$$

The necessary condition stated in Lemma 22.1 is indeed nothing but weak duality for this LP (define $z_e = \frac{y_e}{c_e F^*}$).

Strong duality tells us that the condition in Lemma 22.1 is actually also sufficient and our algorithm is a constructive proof of strong duality. It tries to find a solution to the primal and the dual LP by iteratively adapting the primal and dual solution in a way similar to the algorithm for online set cover that we saw earlier.

The pair of a primal and a dual solution can be understood as an equilibrium of a game. This is what we will talk about next time.

References

- Naveen Garg, Jochen Könemann: Faster and Simpler Algorithms for Multicommodity Flow and Other Fractional Packing Problems. FOCS 1998

- Sanjeev Arora, Elad Hazan, Satyen Kale: The Multiplicative Weights Update Method: a Meta-Algorithm and Applications. Theory of Computing 8(1): 121-164 (2012): Survey on Multiplicative Weights Technique including this algorithm and others