Algorithms and Uncertainty, Winter 2025/26 Lecture 24 (5 pages)

Max-Flow via Experts

Thomas Kesselheim

Last Update: January 26, 2026

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 as the fastest known algorithms for this problem.

1 Max-Flow Problem and Algorithmic Approaches

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.

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

Note that an alternate, maybe more common but equivalent formulation asks us to assign flow values to all edges such that flow conservation is fulfilled.

We will 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: We define lengths of each edge via the experts setting as follows. Let each edge correspond to one expert. The experts algorithm puts some probability mass on each edge. Based on this probabilities, the length of each edge is just the probability mass divided by the capacity of this edge. In the ongoing process, we adapt the lengths with respect to the changes in probability that the experts algorithm tells us to do. In each round, our algorithm computes a shortest path with respect to the edge lengths and increases the flow along this path.

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

$$\mathrm{Regret}^{(T)} = G^{(T)}_{\max} - G^{(T)}_{\mathrm{Alg}} \enspace ,$$

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

The Multiplicative Weights algorithm guarantees

$$G^{(T)}_{\mathrm{Alg}} \geq (1-\eta) G^{(T)}_{\max} - \frac{\ln m}{\eta} \enspace .$$

So, $\mathrm{Regret}^{(T)} \leq \eta G^{(T)}_{\max} + \frac{\ln m}{\eta}$.

3 Flows and Edge Lengths

In our algorithm, we will define edge lengths and find shortest paths with respect to these edge lengths. LP duality gives a connection between flows and edge lengths. However, we need a somewhat unusual formulation. This is why we reprove it from scratch.

Lemma 24.1. *If there is a flow of value F^*, then for all choices of edge weights $(y_e)_{e \in E}$ 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. Let x be the flow of value F^*. We can scale x to $x' := \frac{1}{F^*} x$. Now x' is a (not necessarily feasible) flow of value 1, that is,

$$\sum_{P \in \mathcal{P}} x'_P = 1 \enspace .$$

Furthermore, for each edge e, we have

$$\sum_{P : e \in P} \frac{1}{c_e} x'_P = \frac{1}{F^*} \sum_{P : e \in P} \frac{1}{c_e} x_P \leq \frac{1}{F^*} \enspace .$$

This inequality has the interpretation that x' uses at most a $\frac{1}{F^*}$-fraction of each edge's capacity. As the edge weights sum up to 1, this implies

$$\sum_{e \in E} y_e \sum_{P : e \in P} \frac{1}{c_e} x'_P \leq \frac{1}{F^*} \enspace .$$

Note that because $\sum_{P \in \mathcal{P}} x'_P = 1$ we have

$$\sum_{e \in E} y_e \sum_{P : e \in P} \frac{1}{c_e} x'_P = \sum_{P \in \mathcal{P}} x'_P \sum_{e \in P} \frac{y_e}{c_e} \geq \min_{P \in \mathcal{P}} \sum_{e \in P} \frac{y_e}{c_e} \enspace .$$

So, in combination

$$\min_{P \in \mathcal{P}} \sum_{e \in P} \frac{y_e}{c_e} \leq \frac{1}{F^*} \enspace .$$

□

Observe that in Lemma 24.1 the constraint on $(y_e)_{e \in E}$ is exactly the kind of output we will be getting from an experts algorithm if each edge is an expert.

4 The Algorithm

Our algorithm uses Lemma 24.1 as follows. An experts algorithm chooses $(y_e)_{e\in E}$ with $\sum_{e\in E} y_e = 1$. By this choice of edge weights, the experts algorithm tries to "convince" us that there is no flow of value F^*. We then compute the shortest path with respect to edge lengths $\frac{y_e}{c_e}$. By Lemma 24.1, if there is a flow of value F^*, this path will be of length at most $\frac{1}{F^*}$. The expert algorithm's gain is the length of the path that we found and used. Then, the experts algorithm updates the edge weights based on which path we chose.

Formally, the algorithm is defined as follows.

- For $t = 1, \dots, T$
  - 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 $x_{P^{(t)}}^{(t)} = c^{(t)}$ and $x_P^{(t)} = 0$ 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 $x = \frac{1}{G_{\max}^{(T)}} \bar{x}$

We will show that the flow x is feasible and a good approximation if T is large. To this end, we will understand $\bar{x}$ in two ways: By how much are edge capacities exceeded and what's the value of the flow.

5 Capacities

Lemma 24.2. *The maximum factor by which $\bar{x}$ exceeds an edge capacity is exactly $G_{\max}^{(T)}$.*

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

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

This also means

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

So $G_{\max}^{(T)}$ is exactly the maximum factor by which $\bar{x}$ exceeds an edge capacity. □

6 Flow Value

Lemma 24.3. *The flow $\bar{x}$ has value*

$$\sum_{P \in \mathcal{P}} \bar{x}_P = \sum_{t=1}^{T} c^{(t)} \geq F^* \cdot G_{\mathrm{Alg}}^{(T)} \ .$$

Proof. By the definition of the algorithm, the value of the flow $\bar{x}$ is given by $\sum_{P \in \mathcal{P}} \bar{x}_P = \sum_{t=1}^{T} c^{(t)}$. In the remainder, we will show that $c^{(t)} \geq F^* \cdot g_{\mathrm{Alg}}^{(t)}$. Then taking the sum over all t implies the claim.

In order to show that $c^{(t)} \geq F^* \cdot g_{\mathrm{Alg}}^{(t)}$, we need to upper-bound $g_{\mathrm{Alg}}^{(t)}$, that is, the algorithm's gain in the t-th step. To this end, we use that we always choose the shortest path with respect to the current weights and Lemma 24.1 ensures that this path is short.

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

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

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 24.1,

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

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

7 Approximation Guarantee

Note that our results so far show that x is indeed a feasible flow of value $\frac{G_{\mathrm{Alg}}}{G_{\max}}$. Recall that $G_{\mathrm{Alg}} = G_{\max} - \mathrm{Regret}^{(T)}$, so we have the following bound regarding the approximation guarantee.

Corollary 24.4. *The flow x is feasible and has value at least $F^*(1 - \frac{1}{G_{\max}^{(T)}} \mathrm{Regret}^{(T)})$, where F^* is the value of an optimal flow.*

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

Lemma 24.5. *The gain vectors $g^{(1)}, \ldots, g^{(T)}$ generated by the algorithm fulfill*

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

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

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

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

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

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

so the above guarantee becomes

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

If we choose $\eta = \frac{\epsilon}{2}$ and $T = \frac{4}{\epsilon^2} m \ln m$, then $\sum_{P \in \mathcal{P}} x_P \geq F^*(1 - \epsilon)$.

Theorem 24.6. *With Multiplicative Weights, the algorithm computes a $(1-\epsilon)$-approximate flow using $\frac{4}{\epsilon^2} m \ln m$ shortest-path computations. Its overall running time is $O(\frac{1}{\epsilon^2} m^2 \ln m)$.*

8 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. Lemma 24.1 is indeed not only necessary but also sufficient for the existence of a flow of value F^*. Our algorithm can be seen as 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