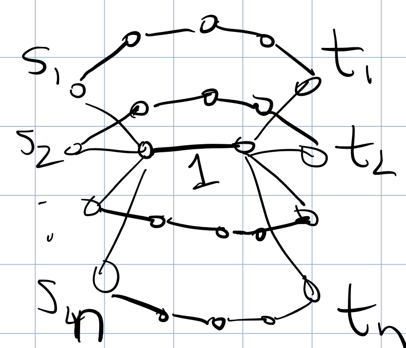
3 Nov 2021

Multiplicative Weights Method

A method for reducing certain optimization problems with constraints to iteratively running greedy algorithms to solve simpler (often unconstrained) problems.



Detour: Can we design a deterministic algorithm with the same approx guarantee as the randomized rounding algorithm from last time?

Yes, using 1 method of conditional expectations.

Reminder: The probability of overloading edge (u,v) beyond $(1+\varepsilon)\cdot c(u,v)$ was bounded above, using Markov's ineq. by the quantity

$$\mathbb{E}\left[e^{t\cdot \mathrm{load}(u,v)}\right]\cdot e^{-(1+\varepsilon)\,t\,r\,c(u,v)}$$

where $t = \ln(1+\varepsilon)$, $e^t = 1+\varepsilon$, so the above quantity is

$$\mathbb{E}\left[(1+\varepsilon)^{\mathrm{load}(u,v)}\right]\cdot (1+\varepsilon)^{-(1+\varepsilon)\,r\,c(u,v)}.$$

The expected # of overloaded edges is

$$\sum_{(u,v)\in E} \mathbb{E}\left[(1+\varepsilon)^{\mathrm{load}(u,v)}\right]\cdot(1+\varepsilon)^{-(1+\varepsilon)\,r\,c(u,v)}$$

$$= \boxed{\sum_{(u,v)\in E}\left(\prod_{i=1}^{k}\mathbb{E}\left[(1+\varepsilon)^{\mathbb{1}[(u,v)\in P_i]}\right]\right)\cdot(1+\varepsilon)^{-(1+\varepsilon)\,r\,c(u,v)}}.$$

The proof showed this monstrosity is $< \frac{1}{2}$ when $\forall\,(u,v)$ $\quad c(u,v) > 3\ln(2m)/\varepsilon^2$.

$\Phi(f_1, f_2, \ldots, f_k)$

f_i = prob distrib over paths P_i from s_i to t_i.

$\Phi(f_1, \ldots, f_k)$ is a linear function of f_i, for any index i.

$$\Phi(f_1, \ldots, f_k) = \sum_{(u,v) \in E} \underbrace{(1+\varepsilon)^{-(1+\varepsilon) r\, c(u,v)} \left[\prod_{\substack{j=1 \\ j \neq i}}^{k} \left[1 + \varepsilon f_j(u,v) \right] \right]}_{K_i(u,v)} \left(1 + \varepsilon f_i(u,v) \right)$$

Choosing unit flow f_i to minimize RHS amounts to solving a min-cost path problem from s_i to t_i with edge costs $K_i(u,v)$.

ALGORITHM.

1. Solve LP to compute $f_1, \ldots, f_k$ that minimize r, as in first step of rand rounding algorithm.
2. for $i = 1, \ldots, k$:
   - Compute or recompute edge costs $K_i(u,v)$ as above.
   - run Dijkstra to find min-cost P_i.
   - update $(f_1, \ldots, f_k)$ by replacing f_i with P_i.

Detour: Prediction with expert advice.

You need to make a series of predictions.

There are K "experts" who advise you on each prediction.

If you follow advice of expert i at time t you gain $u(i,t)$.

You have to choose which one to follow knowing $\{u(j,s) \mid s<t, j\in[K]\}$ but you don't know $u(i,t)$.

Goal: perform almost as well as best expert.

Theorem: $\forall \varepsilon > 0$ There is an algorithm for choosing (a random) expert i_t at each time t, s.t. $\forall$ seq's $u(\cdot,\cdot)$

$$\mathbb{E}\left[\sum_{t=1}^{T} u(i_t,t)\right] \geq (1-\varepsilon)\cdot \max_i \left(\sum_{t=1}^{T} u(i,t)\right) - O\left(\frac{\log K}{\varepsilon}\right)$$