Lecture 26 12/05/2025

Online Algorithms

Simple example.

Stairs vs Elevators problem

Time to take stairs S

Time to take elevator L

$$S > L .$$

How long should one wait for the elevator? If one knows the distribution of the arrival time the elevator then can compute various quantities. We will work with worst case model which has merits and cons. Adversary decides.

Is there a good deterministic strategy?

Let $T = S - L$.

For a time $t \geq 0$ let $OPT(t)$ be the optimum total time if elevator arrives at t and t was known in advance.

<u>Claim</u>: $OPT(t) = t + L$ if $t \leq T$

and $OPT(t) = S$ if $t > T$.

For an alg A let $A(t)$ be the total time if elevator comes at time t.

For an algorithm A its competitive ratio is

$$c(A) \sup_{t} \frac{A(t)}{OPT(t)}$$

Any deterministic algorithm for the problem can be characterized by a single parameter a which is the time it will wait before taking the stairs (it will take elevator if it comes before a).

Let $Wait_a$ be the algorithm.

What is its competitive ratio?

$Wait_0$ is always to take stairs.

If $t = 0^+$ then $Wait_0$ takes S time and $OPT(0^+) = L$ so competitive ratio is $\frac{S}{L}$ which can be very large.

One can see that the worst case for $Wait_a$ is if elevator comes at a^+

in which case competitive ratio is

$$\frac{a+S}{\min(a+L, S)}$$

Not hard to see that $a = S-L = T$ is the minimizing value which is intuitive. Does not make sense to wait $> T$.

Competitive ratio is $\dfrac{S-L+S}{S} = 2 - \dfrac{L}{S}$.

Theorem: The optimum competitive ratio for the problem is $2 - \frac{L}{S}$.

What about the case we allow randomization to the algorithm? Can we improve the ratio?

Need to be careful about: the model. We will consider oblivious adversary that cannot see the randomness of the algorithm. Alternatively we let the adversary know the algorithm and it picks the worst case input σ and then we allow algorithm to randomize and define competitive ratio as

$$\max/\sup_{\sigma} \frac{E[A(\sigma)]}{OPT(\sigma)}$$

Since $A(\sigma)$ is a random variable.

For the elevator problem one can view a randomized algorithm as picking the waiting time according to a distribution. What distribution?

Turns out that the optimal distribution is to pick $w \in [0,T]$ according to the density function

$$p(t) = \frac{1}{(e-1)T} e^{\frac{t}{T}}.$$

Note that $\int_0^T p(t)\,dt = 1.$

<u>Theorem</u>: The competitive ratio of the randomized algorithm that picks waiting time according to

$$p(t) = \frac{1}{(e-1)T} e^{\frac{t}{T}} \text{ for } t \text{ in } [0,T]$$

and $p(t) = 0$ for $t > T$.

is $\frac{e}{e-1} \approx 1.58$.

<u>Proof Sketch:</u> Fix any time $a \in [0, T]$. Suppose elevator arrives at time a. Then

$OPT(a) = a + L$.

What about the algorithm?

Its total time is $a+L$ if the waiting time it picks is $\geq a$.

Otherwise it is $t + S$ where $t < a$.

Thus
$$E[A \mid a] = \int_0^a (S+t)\, p(t)\, dt + \left(\int_a^T p(t)\, dt \right) [a+L].$$

The worst case happens when $a = T$ as in the deterministic case. ~~in~~

Then we have

$$E[A(T)] = S + \int_0^T t \cdot p(t)\, dt$$

$$= S + \int_0^T \frac{t}{(e-1)T} e^{\frac{t}{T}}\, dt.$$

$$= S + \frac{1}{e-1}\left[t e^{\frac{t}{T}} - T e^{\frac{t}{T}} \right]_0^T$$

$$S + \frac{1}{e-1} T.$$

While $OPT(T) = S.$

Hence competitive ratio is

$S+$

$$\frac{S + \frac{1}{e-1} T}{S}$$

$$= \frac{e}{e-1} - \frac{1}{e-1} \cdot \frac{L}{S}.$$

$\square$.

Ratio tends to $\frac{e}{e-1}$ as $\frac{L}{S} \to 0$.

<u>Lower bound</u>: How do we prove that above ratio is optimal. In general how do we prove lower bounds on randomized algorithms?

Typically we use what is called Yao's lemma who recognized early on that one can use Von-Neumann min-max characterization of two player games gives a way to prove lower bounds.

Instead of explaining the general set-up we illustrate it in the context of this simple problem.

To prove a lower bound on randomized algorithms we focus on a lower bound on "deterministic" algorithms against inputs drawn from a distribution. The game is as follows. Come up with a probability distribution on inputs, say $\mathcal{D}$. Now, given knowledge of $\mathcal{D}$ what is the best deterministic algorithm? Suppose no deterministic algorithm, even with knowledge of $\mathcal{D}$, can do better than c. Then c is a lower bound on randomized algs.

For our problem we design/pick a distribution on the arrival time of the elevator.

Suppose we pick the distribution e^{-t} for the elevator arrival time. Note that we are allowing arbitrary large times but that is ok.

To simplify analysis we will set $L \to 0$ and $S \to 1$.

Then what is expected value of OPT

$$OPT = \int_0^\infty \min(1, t)\, p(t)\, dt$$

$$= \int_0^1 t e^{-t} dt + \int_1^\infty 1 e^{-t} dt$$

$$= 1 - e^{-1}.$$

Fix any deterministic algorithm.
It's a threshold algorithm with $a \in [0, 1]$.
Its expected cost

$$E[A(a)] = \int_0^a t e^{-t} dt + \int_a^\infty (a+1) e^{-t} dt$$

$$= 1.$$

Doesn't depend on the threshold!

Thus the competitive ratio

$$\geq \frac{1}{1-\frac{1}{e}} \geq \frac{e-1}{e}.$$

$\square$.

Ski Rental:

closely related problem but in some sense a discrete problem. More well known.

Set up: costs b \$s to buy skiis and \$1 to rent/day $b > 1$.

Every day morning one has to decide whether to rent or buy if not already bought.

Adversary decides when to break one's leg and finish the ski season.

What is the best strategy to minimize total cost?

Deterministic: rent for $\lfloor b \rfloor$ days and then buy.

Can show 2-competitive and optimum for deterministic.

Can obtain $\left(1-\frac{1}{e}\right)$ for randomized which is also optimum.

Somewhat more technical than elevator/stairs problem because of discrete days.

Yao's min-max principle proof. of useful direction.

inputs

Algs

	I_1	I_2			I_n
A_1					
A_2	c_{ij}				
A_m					

Consider a discrete setting where we enumerate all inputs of particular size and all deterministic algs.

Say m algs and n inputs.

p a distribution over inputs

q a distribution over algorithm.

Let q^* be an opt rand alg. corresponding to choosing

$$\max_{I_j} \frac{\sum_{i=1}^{m} q_i^* A_i(I_j)}{OPT(I_j)} = c^*.$$

$$\geq \sum_{j=1}^{n} p_j \frac{\sum_{i=1}^{m} q_i^* A_i(I_j)}{OPT(I_j)} \quad \forall p$$

$$= \sum_{j=1}^{n} p_j \frac{\sum_{i=1}^{m} q_i^* A_i(I_j)}{OPT(I_j)} \quad \forall p$$

$$= \sum_{i=1}^{m} q_i^* \sum_{j=1}^{n} \frac{p_j A_i(I_j)}{OPT(I_j)} \quad \forall p$$

$$= \sum_{i=1}^{m} q_i^* \sum_{j=1}^{n} p_j \frac{A_i(I_j)}{OPT(I_j)} \quad \forall p$$

n

$$\geq \min_{A_i} \sum_{j=1}^{n} p_j \frac{A_i(I_j)}{OPT(I_j)} \quad \forall p.$$

$$\geq \max_{p} \min_{A_i} \sum_{j=1}^{n} p_j \frac{A_i(I_j)}{OPT(I_j)}.$$

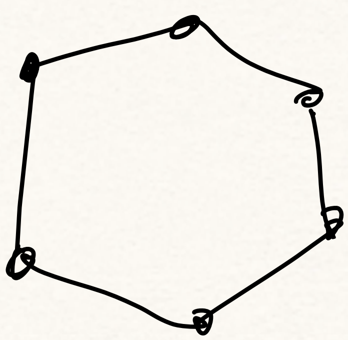
$\square$

Probabilistic Tree Embeddings

Let $G=(V,E)$ be a graph.
Want to approximate distances in G by a tree. Why? Trees are simple.

Can one approximate distances by a single spanning tree?

C_n n cycle.



For any spanning tree of C_n $\exists$ an edge $e=uv$ s.t.

$d_T(u,v) = n-1$

Can we use randomization to improve this?

Yes!

Theorem: $\exists$ a probability distribution p over spanning trees $ST(G)$ of G such that $\forall\, u,v \in V$

$$\mathop{E}_{T\sim p}\left[d_T(u,v)\right] \le O(\log n \log\log n)\, d_G(u,v).$$

Moreover one can sample a tree from p efficiently.

Many applications to algorithms.

The bound $O(\log n \log\log n)$ can be

improved to $O(\log n)$ if G is a complete graph that corresponds to a metric space. This suffices in many applications.

Lower bound is $\Omega(\log n)$ even for planar graphs. .