Online Computation, Ex 2.

David Ponarovsky

February 2, 2023

ex1. Find a simple description of the work-function algorithm in the case of uniform metric space.

Solution. Recall that in the work-function algorithm, we weigh the configurations by the price one has paid for serving all the requests and end at those configurations, combing the distance between them and the current configuration. In the paging problem, the configurations are the content of the cache stack. Denote by m and k the hard disk size and the number of servers. Enumarate each of the valid stack states by $Q = q_0, q_1, q_2, \dots q_M$ where $M = \binom{m}{k}$. In addition, define a weight function $w : Q \to \mathbb{R}$ to be Hamming distance between pair of configurations. Finally, define a $M \times m$ table $W_{i,j}$ to store the optimal work that has to be done while serving $\sigma_0, \sigma_1, \sigma_2, \dots \sigma_j$ requests and ending in configuration q_i.

Cliam. In the uniform case, if $v_1, v_2, v_3, .., v_l$ is an configuration path that servers $\sigma_1, \sigma_2..\sigma_l$ and ends at configuration v_l such that v_l deffers from v_{l-1} by $t > 1$ elements, then there is also an optimal path $v_1, u_2..u_{l-1}v_l$ which also serves the requests and v_l differs from u_{l-1} by at most $t-1$ elements.

Proof. As v_l differs from v_{l-1} by at least two elements, there must be an element $\tilde{\sigma} \neq \sigma_{l-1}$ belongs to v_{l-1}, By the assumption that the algorithm always holds a full cache, there must be an element $\sigma^\star \in v_l / v_{l-1}$ (that could be σ_l but doesn't has to). Consider the configuration $u_{l-1} = v_{l-1} \cup \sigma^\star / \tilde{\sigma}$. Since $v_1 v_2, .., v_{l-1}, v_l$ is optimal, $v_1, v_2.., v_{l-2}u_{l-1}$ must also be optimal, and cost at most 1 more than the $v_1 v_2, .., v_{l-1}$, clearly we have that $v_1, v_2, ..u_{l-1}v_l$ is also optimal and we done.

By induction argument, we obtain that in the uniform matric, the case is enough to consider configurations paths in which any adjacent configuration pair differ by at most one element. So our work-function algorithm will take advantage of that fact to compute W.

```
q_0 ← initial state
for each new request σ_j do
    for i ∈ [m] and q_i ≠ q_0 do
        if σ_j ∈ q_i then
            for (σ*, σ̃) such that σ* ∈ q_i and σ̃ ∉ q_i do
                q_{i'} ← q_i/σ* ∪ σ̃
                W_{i,j} ← min{W_{i,j}, W_{i',j-1} + 1}
            end
            W_{i,j} ← min{W_{i,j}, W_{i,j-1}}
        end
        else
            W_{i,j} ← ∞
        end
    end
    q_0 ← arg min_{i:σ_j ∈ q_i} {W_{i,j-1} + Ham(q_i, q_0)}
    Set q_0 as the current state, evict and serve if
     needed.
end
```

Algorithm 1: Work-function-Algo for paging.

ex2. Consider the following 3-point metric space, $w(a,b) = 1$ and $w(\cdot, c) = M$. The initial configuration is $\{b, c\}$ (2 servers). Show that randomized competitive ratio, for some value of M is $> H_2 = 1 + \frac{1}{2}$.

Solution. Define the following distribution:

$$\tilde{\sigma} = \begin{cases} (ab)^{\frac{M}{3}} & \text{w.p } \frac{1}{2} \\ (ab)^{\frac{M^{100}}{3}} & \text{w.p } \frac{1}{2} \end{cases}$$

Using Yao's principle, it's enough to show that any deterministic algorithm is H_2 competitive in expectation against that specific distribution. First, notice that knowing what is the exactly drawn σ, fixes an optimal strategy which is one of the following: moving the server initialized at a between a, b points alternately or choosing first the server that is located in c into a in the second scenario. Putting down, we obtain that:

$$\mathbf{E}\left[c_{\text{base}}(\sigma) : \sigma \sim \tilde{\sigma}\right] = \frac{1}{2}\left(\frac{M}{3} + M\right) = \frac{4}{6}M$$

Meanwhile, by the fact that reading any prefix of requests series at length less than $\frac{M}{3}$ doesn't expose any information about the drawn input which wasn't known at the initialized moment, it follows by indistinguishable arguments that the best a randomized algorithm can do is to guess. (Formal proof is a reduction from an algorithm that decides at an arbitrary step to one that decides at the first turn). Furthermore, we

could assume that any of the deterministic algorithms which we test $\tilde{\sigma}$ against are aware they compete against drawn from $\tilde{\sigma}$, (As any other algorithm could only play worse than them). To conclude, we will test $\tilde{\sigma}$ against deterministic algorithms, which have to decide at the first turn in what cases there are; If they succeed, they pay OPT; Otherwise, they pay OPT $+M$ at least.

So the excpection of such algorithm is at least: $\frac{1}{2}$ OPT + $\frac{1}{2}$ (OPT $+M$). Namly, OPT + $\frac{1}{2}M \Rightarrow$ the comppetive ratio is greater than $1\frac{1}{2}$ and that is what we exactly want to prove.

ex3. Show that randomized marking algortihm cannot be c-competitive against the adaptive online adversary, for $c = o(k)$.

Solution. Assume by contrdiction that there is a constant $c > 1$, and a randomizded algorithm which is an c-competitive in the adadptive online setting. According to the theorem shown in class, If there exists an α competitive algorithm for an online problem in the non-adaptive setting, and in addition, there exists a β competitive algorithm for the same problem against an adaptive online adversary. Then it follows the existence of algorithm $\alpha\beta$ competitive against an offline adaptive adversary. Combining the fact that randomized can't help against such an adversary, we obtain that the deterministic competitive ratio is lower than $\alpha\beta$. As we know that a k-lowerbound for the deterministic regime and also a $\log k$ solution using randomization against a non-adaptive adversary, we obtain that

$$\begin{aligned} &\alpha\beta \geq k \\ \Rightarrow &\frac{k}{c}\log k \geq k \end{aligned}$$

But for any $k \leq \log 2^c$ we obtain the oppsite direction. This means that there is a range of valid k that obtains a better ratio than the lower bound. And that is a contradiction.

ex4 - Ski Rental. At each step, the adversary decides either to continue or stop. Stop terminating the game. If it continues, the online algorithm decides either to rent or buy. Rent costs 1. Buy costs $M > 1$. Design a primal-dual randomized online ski-rental algorithm with a better than 2 competitive ratio.

Solution. . Let's start by formulating an integer LP for the Ski-Rental problem. Denote by m the days' number, and associate a variable x, indicating whether the algorithm decides to buy. Also, let's associate a variable ξ_j for each day, indicating if the algorithm pays for rent. In each turn, the solution must satisfy the restrictions $\xi_j + x \geq 1$. The cost which we would like to minimize is $M \cdot x + \sum_j \xi_j$. So, in overall, we get that LP is:

$$\begin{aligned} &\min Mx + \sum_j \xi_j \\ &\text{s.b } x + \xi_j \geq 1 \Leftrightarrow \\ &\begin{bmatrix} 1 & 1 & 0 & 0 & \cdot \\ 1 & 0 & 1 & 0 & \cdot \\ 1 & 0 & 0 & 1 & \cdot \\ 1 & 0 & 0 & 0 & \cdot \\ \cdot & \cdot & \cdot & \cdot & \cdot \end{bmatrix} \begin{bmatrix} x \\ \xi_1 \\ \xi_2 \\ \xi_3 \\ \cdot \end{bmatrix} \geq \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \\ \cdot \end{bmatrix} \end{aligned}$$

So the dual program is

$$\begin{aligned} &\max \sum_j z_j \\ &\text{subject to} \\ &\begin{bmatrix} 1 & 1 & 1 & 1 & \cdot \\ 1 & 0 & 0 & 0 & \cdot \\ 0 & 1 & 0 & 0 & \cdot \\ 0 & 0 & 1 & 0 & \cdot \\ \cdot & \cdot & \cdot & \cdot & \cdot \end{bmatrix} \begin{bmatrix} z_1 \\ z_2 \\ z_3 \\ z_4 \\ \cdot \end{bmatrix} \leq \begin{bmatrix} M \\ 1 \\ 1 \\ 1 \\ \cdot \end{bmatrix} \end{aligned}$$

So we would like to have an algortihm that gurntess that the current solution (fraction LP version) for one of the program is valid but sull serves as a good approximation to the other. Consider the following algorithm:

for *each new day* j **do**
  if $x < 1$ **then**
    $\xi_j \Leftarrow 1 - x$
    $x \Leftarrow \left(1 + \frac{1}{M}\right) x + \frac{1}{(c-1)M}$
    $z_j \Leftarrow 1$
  end
end
$\tilde{z_j} \leftarrow z_j / \left(\sum z_j\right)$

Algorithm 2: Ski-Rental

By the fact that the algorithm halt only after $x \geq 1$ it follows that x is a valid solution for the fraction primal program, and therefore is also an upper bound for dual program.

Also Notice that sacling $\tilde{z_i} \leftarrow z_i \cdot M / \left(\sum_i z_i\right)$, is a vaild soultion for primal program. Denote by k the last day in which $x < 1$. Namly $z_j = 0$ for any $j > k$, Then we have that : $x < (1 + 1/M) \cdot 1 + 1/\left((c-1)M\right)$ So picking $c = 2 \Rightarrow x < 1 + \frac{2}{M}$. Now As in each itearion we add at most :

$$\begin{aligned} &\Delta = x_{t+1} - x_t \sim \frac{1}{M}x_t + \frac{1}{M} \Rightarrow \frac{k}{M} < 1 + \frac{2}{M} \\ &\Rightarrow k \leq M\left(1 + \frac{2}{M}\right) \Rightarrow \sum z_j \leq M\left(1 + \frac{2}{M}\right) \end{aligned}$$

So $\tilde{z_j}$ are valid soultion to the dual fraction program such is $\left(1 + \frac{2}{M}\right)$ approximate. As the scaling made by a constant factor, one could ensure to hold the current approximate soultion in each iteration. So it left to show how one can transfrom the fractional version into randomized algorithm that need to choose etiher rent or buy. We will do dat by flip a coin in each iteartion according to the coin result.

ex5. Prove Yao's minimax principle.

$$
\begin{aligned}
&\forall \text{rand. } \tilde{\text{alg}}\ \exists\ \sigma \\
&\quad \mathbf{E}\left[c_{\text{alg}}(\sigma) : \text{alg} \sim \tilde{\text{alg}}\right] \geq c \cdot c_{\text{base}}(\sigma) \\
&\Leftrightarrow \exists \text{ rand. } \tilde{\sigma} \forall \text{alg} \\
&\quad \mathbf{E}\left[c_{\text{alg}}(\sigma) : \sigma \sim \tilde{\sigma}\right] \geq c\mathbf{E}\left[c_{\text{base}}(\sigma) : \sigma \sim \tilde{\sigma}\right]
\end{aligned}
$$

Solution. First direction, assume through contradiction that there exists a deterministic algorithm such that for all distributions $\tilde{\sigma}$:

$$
\mathbf{E}\left[c_{\text{alg}}(\sigma) : \sigma \sim \tilde{\sigma}\right] < c\mathbf{E}\left[c_{\text{base}}(\sigma) : \sigma \sim \tilde{\sigma}\right]
$$

And that holds, in paritcular, for distribution $\tilde{\sigma}$ which suportted by a single σ. Hence, because any deterministic algorithm is also a randomized algorithm, set it to be $\tilde{\text{alg}}$, and that immediately yields a contradiction. It is left to show the second direction. By the monotonic property of random variables, we have that for any distribution $\tilde{\sigma}$:

$$
\begin{aligned}
&\mathbf{E}\left[\mathbf{E}\left[c_{\text{alg}}(\sigma) : \sigma \sim \tilde{\sigma}\right] : \text{alg} \sim \tilde{\text{alg}}\right] \\
&\quad \geq c \cdot \mathbf{E}\left[\mathbf{E}\left[c_{\text{base}}(\sigma) : \sigma \sim \tilde{\sigma}\right] : \text{alg} \sim \tilde{\text{alg}}\right] \\
&\mathbf{E}\left[\mathbf{E}\left[c_{\text{alg}}(\sigma) : \text{alg} \sim \tilde{\text{alg}}\right] : \sigma \sim \tilde{\sigma}\right] \\
&\quad \geq c \cdot \mathbf{E}\left[\mathbf{E}\left[c_{\text{base}}(\sigma) : \text{alg} \sim \tilde{\text{alg}}\right] : \sigma \sim \tilde{\sigma}\right] \\
&\mathbf{E}\left[\mathbf{E}\left[c_{\text{alg}}(\sigma) : \text{alg} \sim \tilde{\text{alg}}\right] : \sigma \sim \tilde{\sigma}\right] \\
&\quad \geq c \cdot \mathbf{E}\left[c_{\text{base}}(\sigma) : \sigma \sim \tilde{\sigma}\right]
\end{aligned}
$$

And by the fact that inequality of exception between random variables follows an existence of atomic event on which the inequality holds, we obtain that there must exists at least a single σ such that:

$$
\mathbf{E}\left[\mathbf{E}\left[c_{\text{alg}}(\sigma) : \text{alg} \sim \tilde{\text{alg}}\right] : \sigma \sim \tilde{\sigma}\right] \geq c \cdot \mathbf{E}\left[c_{\text{base}}(\sigma) : \sigma \sim \tilde{\sigma}\right]
$$

And that ends the proof.