Problem Set 6

Collaborated with: David Kawashima, Steven Brotz

Used one (1) late token.

Problem 1

If we know that our original link cost functions $c_e(.)$ are optimal, then by definition we know that the total cost is minimized with these cost functions. More specifically, in our original game, we know that for each edge e in our original game with flow x, the change in total cost brought by the x^{th} user choosing edge e is:

$$xC_e(x) - (x-1)c_e(x-1)$$

Now, given that we are using an optimal strategy for the original game, we know that for each edge e, the change in total cost of the x^{th} user choosing edge e has to be less than or equal to the change in total cost of the xth user choosing any other edge e' that has flow x' (or otherwise, the total cost would not be minimized and by contradiction, we would not have an optimal strategy). Hence, for each edge e, and all other edges $e' \neq e$, we can write:

$$xc_e(x) - (x-1)c_e(x-1) \leq (x'+1)c_{e'}(x'+1) - (x')c_{e'}(x')$$

Now, notice that the LHS of the above inequality is precisely $c_e^* x$ while the RHS is $c_{e'}^*(x'+1)$. In the context of our game with new cost functions, this means that for each edge e, each user of edge e will incur a cost $c_e^* x$ which is always less than or equal to $c_{e'}^*(x'+1)$, the cost of any other edge e' if the user were to use e' instead of e. Clearly, due to the inequality, any user of any edge e will have no incentive to switch to any other edge e', so in our new game, setting the new cost function to be $c_e^*(x)$ is an equilibrium.

Problem 2

Part 1

a. Yes, it does resemble the definition of a Nash equilibrium. Recall, that a feasible flow f is an *equilibrium flow* for the nonatomic game if, for each commodity $i \in I$, and every pair of paths $p, p' \in P_i$, with $f_p > 0$,

$$c_p(f) \leq c_{p'}(f).$$

This essentially means that the f, by being an equilibrium flow allocates the transport of commodities so that the the path with the minimum cost is used to transport each commodity. This precisely relates to Nash equilibrium; each commodity has its cost minimized with equilibrium flow f, so there is no reason/incentive for a commodities to follow allocations of any other flow. We can think of the players as infinitesimals of a flow. Based on Nash equilibrium, we would have that no part of a flow would have any incentive to choose an alternative path. Thus, we have that the paths chosen by this flow must cost the same while paths not chosen by the flow cannot be any better. Worst case scenario, even if paths p and p' are both used, it would imply that $c_p(f) = c_{p'}(f)$ and so, no player will consider changing path. This resembles the definition of equilibrium flow in nonatomic routing games.

b. As mentioned in the hint, we define our potential function as $\Phi(f) = \sum_{e \in E} h_e(f_e)$, where $h_e(x) = \int_0^x c_e(y)dy$. Because c_e is continuous, we know that h_e is both continuously differentiable and convex. Also notice that $\Delta\Phi$ represents change in cost of flow $(\Phi(f) - \Phi(f'))$ represents the difference in flow costs between flow f and flow f'. Thus, if we find the minima of Φ, we can find the equilibrium flow f^E that achieves the minima of Φ such that the cost of using f_E is no worse than using any other flow (an infiteisimal amount of flow will have no incentive to change paths). Now, based on lecture 10 notes, we know that with this $\Phi(f)$ as our potential function and its derivative being change in flow costs, we have ourselves a potential game. Knowing that in any potential game, there exists an equilibrium which is simply the minima of Φ. Now, to show that a minima exists, we simply have to show that Φ admits at least one minimum. From the formulation of the problem, we know that the set of feasible flows is bounded (any f_p is bounded by 0 and r_i) and closed (any $f_p \in [0, r_i]$) and thus compact. Furthermore, we are given that cost functions are continuous. Thus, by the extreme value theorem, we know that our potential function admits at least one minimum. Hence, we have shown that an equilibrium always exists and that there does exist a feasible flow f^E where the potential function is minimized. We note the similarity between the Φ described in class and the potential function used for nonatomic games here; we simply replace the summation with an integral.

Part 2

a. To maximize $\frac{rc(r)}{xc(x)+(r-x)c(r)}$ over x, we minimize the denominator portion, $xc(x) + (r - x)c(r)$. Given that $\mathcal{C}$ is the class of affine cost functions, we get that:

$$\begin{aligned} & xc(x) + (r - x)c(r) \\ =& x(ax + b) + (r - x)(ar + b) \\ =& ar^2 - arx + ax^2 + br \end{aligned}$$

Differentiating with respect to x and equating to 0, we get:

$$\begin{aligned} 0 =& \frac{\partial}{\partial x}(ar^2 - arx + ax^2 + br) \\ 0 =& 2ax - ar \\ x =& \frac{r}{2} \end{aligned}$$

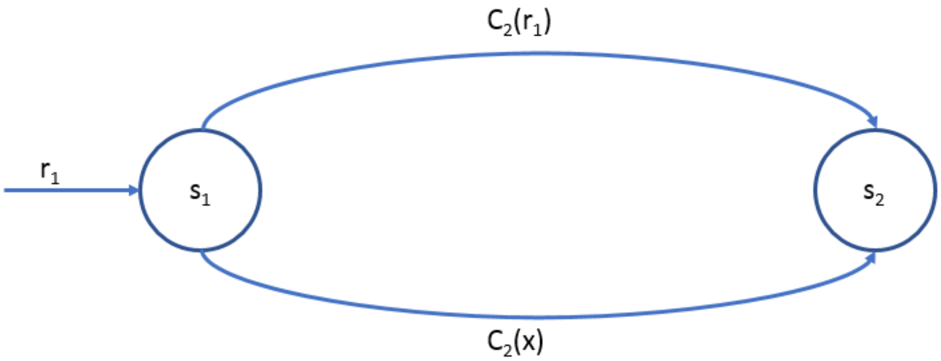
Plugging in for x for $\alpha(\mathcal{C})$, we get:

$$\begin{aligned}\alpha(\mathcal{C}) &= \sup_{c\in\mathcal{C}} \frac{rc(r)}{\frac{r}{2}c(\frac{r}{2}) + \frac{r}{2}c(r)} \\ &= \sup_{c\in\mathcal{C}} \frac{ar^2+br}{\frac{r}{2}(\frac{ar}{2}+b) + \frac{r}{2}(ar+b)} \\ &= \sup_{c\in\mathcal{C}} \frac{ar^2+br}{\frac{ar^2}{4}+\frac{br}{2}+\frac{ar^2}{2}+\frac{br}{2}} \\ &= \sup_{c\in\mathcal{C}} \frac{ar+b}{\frac{3ar}{4}+b}\end{aligned}$$

We are taking the supremum, so we let $b = 0$, and thus we get that $\alpha(\mathcal{C}) = \frac{4}{3}$ as desired.

b. This is quite simple to illustrate. We note that one equilibrium is if flow volume is split between the two halves and nobody takes the u to v edge. In this case, the equilibrium flow equals $\frac{3}{2}$. Note that this is also the optimal case; an even split at node s guarantees that nobody is incentivized to take the $u \to v$ path. Another equilibrium flow is if all incoming volume takes the route $s \to u \to v \to d$ for another equilibrium flow of 2. Calculating the PoA, we get that $\frac{2}{3/2} = \frac{4}{3}$. Hence, we have shown a game with affine cost functions that achieves the bound of part a.

c. We construct a nonatomic routing game by giving the bottom edge a cost function $c_2 \in \mathcal{C}$ and giving the top edge a cost function $c_1 \in \mathcal{C}$ that is everywhere equal to $c_2(r_1)$ where r_1 is the total flow volume (which in this case is just due to a single commodity). We provide a diagram of our game below:



Clearly, by construction, we have an equilibrium where all of the r_1 load goes to the bottom edge. Because the top edge has cost $c_2(r_1)$, even if all of r_1 goes to the bottom edge, they will each incur cost $c_2(r_1)$ which is no worse than the case where any of the load goes to the top edge which will also incur cost $c_2(r_1)$. In this equilibrium, if all the load goes to the bottom edge, we have a total cost of $r_1c_2(r_1)$.

Clearly, the above approach is not optimal. To see why, let's just consider the case where one unit of load decides to take the top edge. In this case, our total cost would be $c_2(r_1) + (r_1 - 1)c_2(r_1 - 1)$. Because c_2 is guaranteed to be non-decreasing we know that $c_2(r_1) + (r_1 - 1)c_2(r_1 - 1) \leq c_2(r_1) + (r_1 - 1)c_2(r) = r_1c_2(r_1)$.

To actually get the optimal cost, we simply have to find the x (the amount of load we should route through the bottom edge) such that the quantity $(r_1 - x)c_2(r_1) + xc_2(x)$ is minimized. More specifically, the optimal cost can be written as :

$$\min_{0 \leq x \leq r_1} ((r_1 - x)c_2(r_1) + xc_2(x))$$

Thus, to get the PoA, we substitute the above expression for the optimal cost, and given that we have an equilibrium case where the total cost is $r_1c_2(r_1)$, we get that the PoA is:

$$\frac{r_1c_2(r_1)}{\min_{0 \leq x \leq r_1}((r_1 - x)c_2(r_1) + xc_2(x))}$$

which, because the min is the denominator, we can rewrite as:

$$\max_{0 \leq x \leq r_1} \frac{r_1c_2(r_1)}{(r_1 - x)c_2(r_1) + xc_2(x)}$$

Now, because we fix our functions to be $c(x)$ and $c(r)$ and r_1 to be r, we have an instance where the price of anarchy is at least $\frac{rc(r)}{(r-x)c(r)+xc(x)}$ (given that we route $x < r$ load to the bottom edge) as desired.

Part 3

Yes, this bound is tight. For every m, we can devise an allocation of loads such that the PoA $\leq 2 - \frac{2}{m+1}$ where m is the number of servers. The setup is as follows:

- We will have $n = 2m$ jobs. For each $i \in [1, m]$, we will have two jobs with such load. For instance, if we have $m = 2$ servers, then we will have $n = 2m = 4$ jobs: 2 jobs with load 1 and 2 jobs with load 2. If we had $m = 3$ servers, then we will have 2 jobs with load 1, 2 jobs with load 2, and 2 jobs with load 3. Note that the total load in each of these cases is equal to $\sum_{i=1}^{m} 2i = m(m + 1)$.

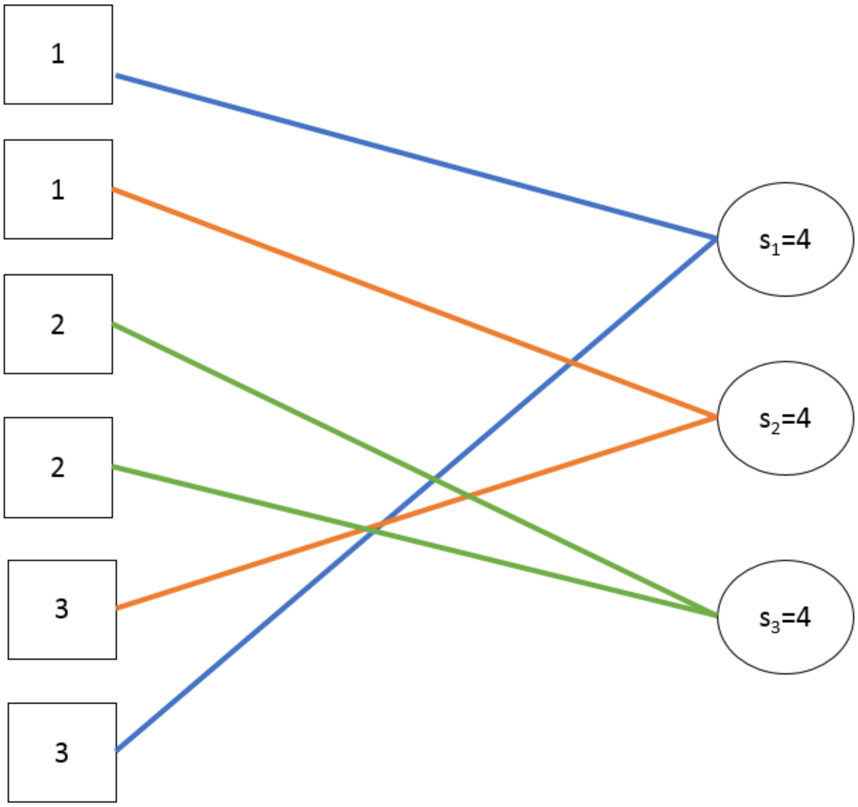
- It is quite obvious to see that our optimal solution arises from each of the m servers getting an equal load of $(m + 1)$. Thus, we have to show that we can always find an equilibrium where one of the servers has load $(2 - \frac{2}{m+1})(m + 1) = 2m$ to achieve the desired PoA.

  - To do so, given that we have m servers, we route the traffic of the two jobs with load m to server m such that server m now contains load $2m$. We then route traffic of the remaining $(n - 2)$ jobs so that each of the remaining $(m - 1)$ servers each have equal loads of $\frac{m(m+1)-2m}{m-1} = \frac{m^2-m}{m-1} = m$.

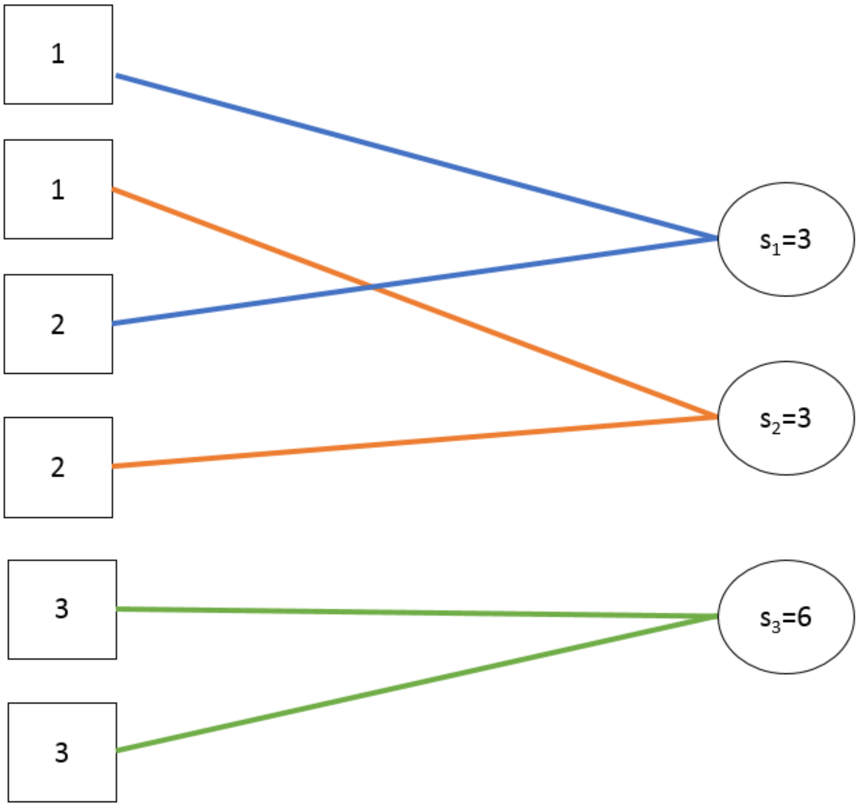
  - It's quite simple to see that we will always be able to pair up each of the $(n-2)$ jobs with one another so that each pair can go to the same server to give that server load m. With $n-2 = 2m-2$ jobs, we can always pair up two jobs to go to each of the $(m-1)$ servers so that each of the $m-1$ servers has load m. In the case where m is even, we know that the jobs with loads 1 and $(m-1)$ can pair up, the jobs with loads 2 and $(m-2)$ can pair up, ..., all the way up to the two jobs with load $m/2$ pairing up to go to the same server. Similarly, if m is odd, we know that the jobs with loads 1 and $(m-1)$ can pair up, the jobs with loads 2 and $(m-2)$ can pair up, ..., all the way up to the jobs with loads $(\lfloor m/2 \rfloor)$ and $(\lceil m/2 \rceil)$ pairing up to go to the same server.

Now, we know that this allocation of jobs to servers yields the desired tight PoA bound because server m has the maximum load of $2m$ and compared to the optimal where each server has load $m+1$, we have that $\frac{2m}{m+1} = (2 - \frac{2}{m+1})$. It's also important to realize that our allocation is an equilibrium. Because of the way we route jobs to servers, we know that server m consists of the loads of two jobs, each with load m. There is no incentive for any of these two jobs to route their traffic elsewhere because the other $(m-1)$ servers each have load m, so moving to any other server would yield the same maximum load as simply staying at server m. For any of the other servers that each have load m, there is clearly no incentive for any of the jobs at these servers to switch. Thus, we have shown that we can achieve the tight bound for each m.

Just for clarity, we illustrate the case for $m = 3$. We have $n = 2m = 6$ jobs (squares) and $m = 3$ servers (circles). First, we look at the optimal allocation, where each server gets a load of $m + 1 = 4$.



Now, we show an equilibrium with maximum load $= 2m = 6$.



$\frac{6}{4} = \frac{3}{2} = 2 - \frac{2}{4}$, so for $m = 3$, we have shown the tightness of the PoA bound. As described above, this generalizes for all valid m.