# Networked Life: Homework 6

**Guo Ziqi - 1000905**
**Zhao Juan - 1000918**
**Zhang Hao - 1000899**

1. **Answer:**

Table 1: Steps of the Bellman-Ford Algorithm

| Bidder | Day 1 | Day 2 | Day 3 | Day 4 | Day 5 |
|---|---|---|---|---|---|
| 1 | (1) 7 | (3) 9.5 | (5) 10→11 | | (8) 27.45 |
| 2 | | (2) 9.25 | | (6) 13.65 | |
| 3 | | | (4) 9.75→11.25 | (7) 13.9 | |
| Ask price | 7.25 | 9.75 | 11.5 | 14.15 | 27.7 |

2. **Answer:**

   According to the bidding progression, bidder 1 wins the bid and pays \$27.45.

3. **Answer:**

   (a) If $b_1 > b_2$, Alice wins the slot with clickthrough rate of 500 per hour. Her payoff is $\$(r - b_2)$ per click, or equivalently, $\$500(r - b_2)$ per hour.

   If $b_1 < b_2$, Alice wins the slot with clickthrough rate of 300 per hour. Alice's payoff is then $\$r$ per click, or equivalently, $\$300r$ per hour.

   (b) From the payoffs above, we can equate the two payoffs and find out that when $2/5r > b_2$, getting the first slot will yield higher payoff for Alice; otherwise, getting the second slot will yield higher payoff for Alice. Therefore, the dominant strategy for Alice is to bid $2/5r$, so that in both cases Alice can always maximize her payoff.

4. **Answer:**

   (a) The only scenario that will yield a positive payoff for Bidder 1 is that he wins both seats. However, assuming Bidder 2 is rational, he will always have the choice to give up one seat and bid \$10 on the other one. So with a \$15 total valuation of the two seats, Bidder 1 would never be able to win both seats.

   In this case, getting only one seat will yield a negative payoff for Bidder 1, as a single seat is worth nothing to him. As the best scenario results in a payoff of \$0, not bidding on any of the seats is the dominant strategy for Bidder 1.

   (b) In this case, Bidder 1 just needs to always bid on the package consisting of both seats. Regardless of whether Bidder 2 bids on individual seats or package, Bidder 1 just needs to top Bidder 2's total bid by minimum increment.

Therefore, Bidder 1 instead can win the bid if package bidding is used. The price charged is $\$(12+m)$ at most, assuming $m$ is the minimum increment to top a previous bid. His payoff is $\$(3-m)$.

5. **Answer:**

The price charged to the winners Bidder 1 and 4 are represented by $p_1$ and $p_4$:

$$M* = (1,1),(4,2)$$

$$V* = 10+9 = 19$$

$$p_1 = (7+8)-9 = 6$$

$$p_4 = (10+8)-10 = 8$$

6. **Answer:**

Vector $\pi[k]$ will exhibit periodic behavior as $k$ becomes large. In particular, $\pi[0] = [1/2,1/2,0,0]^T, \pi[1] = [1/2,0,0,1/2]^T, \pi[2] = [0,0,1/2,1/2]^T, \pi[3] = [0,1/2,1/2,0]^T$, and then the pattern repeats itself.

Therefore, there is no solution for steady-state probabilities $\pi^*$ such that $\pi^{*T} = \pi^{*T}H$.

7. **Answer:**

When $\theta = 0.1$, $\pi_1 = 0.211, \pi_2 = 0.205, \pi_3 = 0.200, \pi_4 = 0.193, \pi_5 = 0.190$;
When $\theta = 0.3$, $\pi_1 = 0.236, \pi_2 = 0.221, \pi_3 = 0.196, \pi_4 = 0.177, \pi_5 = 0.170$;
When $\theta = 0.5$, $\pi_1 = 0.270, \pi_2 = 0.249, \pi_3 = 0.183, \pi_4 = 0.153, \pi_5 = 0.145$;
When $\theta = 0.85$, $\pi_1 = 0.385, \pi_2 = 0.369, \pi_3 = 0.102, \pi_4 = 0.073, \pi_5 = 0.071$;

As $\theta$ increases, the difference of probabilities grows larger. It is more and more obvious that $\pi_1$ and $\pi_2$ have larger significance. This trend is because $\theta$ indicates how much we value the original distribution of transition probabilities. So as $\theta$ becomes larger, the probabilities at equilibrium are mostly determined by the structural connectivity of the graphs.

8. **Answer:**

(a) $[\pi_A^*, \pi_B^*]^T = [0.827, 0.173]^T$

(b) $[\pi_1^*, \pi_2^*]^T = [0.5, 0.5]^T$
$[\pi_3^*, \pi_4^*, \pi_5^*]^T = [0.416, 0.168, 0.416]^T$

(c) The approximate $\pi^*$ calculated by the given equation is $[0.413, 0.413, 0.072, 0.029, 0.072]^T$.

The time complexity of this approximate method is less than that of the directly computing $\pi^*$. No matter if we use inversion or matrices or iterative method, expensive matrix operations like inversion or multiplication is involved. As the matrix dimension $n$ gets larger, computational load would increase exponentially. Splitting the whole matrix into smaller matrices would make computaiton less costly. In this case, updating each iteration would cost around $5^2 = 25$ multiplications for the original matrix. But once we use the approximation, only $3^2+2^2+2^2 = 17$ multiplications are needed.