CS 598RM: Algorithmic Game Theory, Spring 2017 Practice Exam Solutions

1. Answer the following.

- Agents 1 and 2 are bargaining over how to split a dollar. Each agent simultaneously demands share he would like to have, s_1 and s_2, where $0 \leq s_1, s_2 \leq 1$. If $s_1 + s_2 \leq 1$, then the agents receive the shares they named; if $s_1 + s_2 \leq 1$, then both agents receive zero. What is the set of pure strategy equilibria of this game?

  Answer: It is easy to see that if player 1 demands a share x, the best response for player 2 is to demand $1 - x$. By symmetry, player 1 is also happy playing strategy x in response to $1 - x$, and hence this is a pure NE. This is true for every $0 \leq x \leq 1$, so the set of pure NE is $\{(x, 1 - x) \mid 0 \leq x \leq 1\}$.

- A two player game represented by matrices (A, B) is called constant sum if $A(i, j) + B(i, j) = c$, $\forall i, j$ where $c \in \mathbb{R}$. Is the following statement True or False: The set of Nash equilibria of this game is a convex set.

  Answer: The statement is true. The reason is as follows. For any constant-sum games, the objective of player 2, of maximizing $x^T By$ given the strategy x of player 1, is equivalent to that of minimizing $x^T Ay$, just as in the special case of zero-sum games. As a consequence, the set of Nash equilibria can be captured by a linear program and its dual, and by the property of LP solutions, the set of solutions is convex.

c. In a stable roommate problem, there is a set of m dorm rooms each of which can accommodate two students, and a set n of students where $m > n/2$. Each student has a total ordering on all the other students (non-bipartite), and prefers to have a roommate then to live alone. Given an assignment of students to rooms, a pair of students (s_1, s_2) is unstable if they both prefer each other more than their current assignment. An assignment is called stable if there is no unstable pair. Construct an example that has no stable assignment.

  [Hint: Think of a three node graph.]

  Answer: Consider 3 students A, B, C. A prefers B over C; B prefers C over A, C prefers A over B. If 2 students (or more) are kept alone, the assignment is obviously unstable as both of them prefer each other over being alone. In any assignment which pairs two of them, and keeps the third alone, one of the students in the matched pair prefers the third student, and the third prefers him over being alone, hence, this is unstable as well.

- Consider a single item auction where highest bidder wins but pays third highest bid. Show that this auction is not truthful.

  Answer: To show that it is not truthful, it suffices to show once case, where bidding truthfully is not a dominant strategy for a player. Consider 3 players with valuations $v_1 > v_2 > v_3$. If players 1 and 3 are bidding truthfully, player 2 gets a utility of 0 by bidding truthfully, whereas if it bids more than v_1, thus being untruthful, it wins the auction, paying v_3, and getting a strictly positive utility of $v_2 - v_3$. Thus, truthfully bidding is not a dominant strategy.

- Compute the virtual valuation function for the uniform distribution on $[0, a]$ with $a > 0$.

  Answer: For the uniform distribution, the probability distribution function is $f(v) = 1/a$ in $[0, a]$. Consequently, the cumulative distribution function can be easily seen to be $F(v) = v/a$ in $[0, a]$. By definition, the virtual valuation function is

$$\phi(v) = v - \frac{1 - F(v)}{f(v)} = v - \frac{1 - v/a}{1/a} = 2v - a$$

2. Consider a load balancing game with n jobs and m machines. Each job is a player who chooses a machine to run on, and trying to minimize its completion time. Job j has size p_j, and any jobs can choose any of the m machines. Let $r_i(x)$ be the time needed by machine i to process the total load (sum of sizes of assigned jobs) of x. Assume $r_i(x) = x$ for all machines. A machine releases a job only after finishing all of its jobs, i.e., if the set of jobs that choose machine i is $S \subseteq \{1, \ldots, n\}$, then completion time of job $j \in S$ is $\sum_{j \in S} p_j$.

- Is this a potential game?

  Answer: No, this is not a potential game. To prove contradiction, suppose there exists a potential function ϕ, in a simple game involving 2 jobs of unequal sizes and 2 machines. Thus, both players have only 2 pure strategies. Let s_{ij} denote the joint strategy in which job 1 is on machine i, and job 2 is on machine j. Let $t_i(s)$ denote the completion time of job i when the joint strategy is s. Consider the joint strategy s_{ij}. If job 1 switches to machine i', the change in its respective costs(completion times) is $t_1(s_{i'j}) - t_1(s_{ij})$ which must equal $\phi(s_{i'j}) - \phi(s_{ij})$. Thus, if both jobs are on machine 1, and job 1 switches to machine 2, we get,

$$\begin{aligned}\phi(s_{21}) - \phi(s_{11}) &= t_1(s_{21}) - t_1(s_{11}) \\ &= p_1 - (p_1 + p_2) \qquad (1)\end{aligned}$$

  similarly, if job 2 also switches to machine 2, we can write,

$$\begin{aligned}\phi(s_{22}) - \phi(s_{21}) &= t_2(s_{22}) - t_2(s_{21}) \\ &= (p_1 + p_2) - p_2 \qquad (2)\end{aligned}$$

  Adding (1) and (2) gives,

$$\phi(s_{22}) - \phi(s_{11}) = p_1 - p_2 \qquad (3)$$

  Now, if starting from the joint strategy s_{11}, if we reverse the order and make job 2 switch to machine 2 first, followed by job 1 switching to machine 2, we can write equations as above to get,

$$\phi(s_{22}) - \phi(s_{11}) = p_2 - p_1 \qquad (4)$$

  Thus, (3) and (4) imply that the jobs have equal sizes, a contradiction. Thus, we have used the fact that in a potential game, if the change in costs/payoffs via a unilateral deviation is summed over a sequence of deviations, this sum must be independent of the order(permutation) of the deviations.

- Suppose the social welfare is given by the maximum completion time. Show that the Price of Anarchy is upper-bounded by 2.

  Answer: Consider any equilibrium. Let the social welfare at this equilibrium be x. That is, a machine i (say), has a load of x, which is maximum among all the machines. We will prove the maximum completion time is at least $x/2$ in any other joint-strategy, which will give us the desired upper bound on PoA. Consider two cases:

  Case I: i has only one job
  Thus, the only job on i, say job j, has a size x. In any joint-strategy, and in particular the one with optimum social welfare, the machine which has j has a total load of at least x. So, its completion time, and consequently, the maximum completion time too, must be at least x.

  Case II: i has at least two jobs
  As the total load on i is x, there must be a job on i, say job j, of size y that is at most $x/2$. Since we have an equilibrium, job j does not want a different machine, which means that every other machine must have a load of at least $x - y$, hence, at least $x/2$. Thus, the total sum of sizes is at least $x + (n-1)x/2$. In any joint strategy, when this total load is distributed across machines, there must be at least one machine with a load of more than $x/2$, and thus, the maximum completion time is at least $x/2$.

3. Recall the knapsack auction where each bidder i has a publicly known size w_i and a private valuation v_i. Consider a variant of a knapsack auction in which we have two knapsacks, with known capacities W_1 and W_2. Feasible sets of this single-parameter setting now correspond to subset S of bidders that can be partitioned into sets S_1 and S_2 satisfying $\sum_{i \in S_j} \leq W_j$ for $j = 1, 2$. We assume that $w_i \leq \min\{W_1, W_2\}, \ \forall i$.

Consider the allocation rule that first uses the single-knapsack greedy allocation rule (sort jobs in decreasing order of $\frac{b_i}{w_i}$ and allocate until the knapsack is full, where b_i is the bid of agent i) to pack the first knapsack, and then uses it again on the remaining bidders to pack the second knapsack. Does this algorithm define a monotone allocation rule? Give either a proof of this fact or an explicit counter example.

Answer: Yes, the allocation rule is monotone.

Proof. Suppose agent i starts bidding $b'_i > b_i$. Then clearly, $\frac{b'_i}{w_i} > \frac{b_i}{w_i}$. Since no one else's bid has changed, i's her ranking in the ordering will only increase. Let $\boldsymbol{b} = (b_i, \boldsymbol{b}_{-i})$ and $\boldsymbol{b}' = (b'_i, \boldsymbol{b}_{-i})$. There can be these cases:

Case I: $x_i(\boldsymbol{b})$ = Bin 1
Suppose at $\boldsymbol{b}$ when i was considered, already $S \subseteq N$ of agents were allocated to Bin 1. Then at $\boldsymbol{b}'$, if $S' \subseteq N$ is allocated when i is considered then clearly, $S' \subseteq S$. Therefore Bin 1 will have space to accommodate i.

Case II: $x_i(\boldsymbol{b})$ = Bin 2
Suppose at $\boldsymbol{b}$ when i was considered, already $S_1, S_2 \subseteq N$ of agents were allocated to Bin 1 and Bin 2. At $\boldsymbol{b}'$, i is earlier in the ranking. Therefore, it may get considered while filling Bin 1, i.e., if $S'_1 \subseteq N$ in Bin 1 when i is considered then clearly, $S'_1 \subseteq S_1$. If it can not be fit into Bin 1, then

done. Otherwise, we have $S_1' = S_1$, and among the remaining agents argument for the first case applies to Bin 2.

Case III: $x_i(\boldsymbol{b})$**= None**
In this case i can only get better.

There can also be a proof by contradiction.

4. A graph is called two-edge connected, if removing any single edge does not disconnected the graph. Consider a *Reverse (procurement) auction*: We have an undirected two-edge connected network (graph), where each link (edge) is owned by a different agent. To the agent of an edge e, providing that link costs c_e. The auctioneer wants to buy a path from node s to node t. For any such path bought, the social-cost is defined as the sum of the costs of the edges on the path. Equivalent to maximizing the social-welfare in the standard auction, a VCG mechanism in a reverse auction aims to minimize the social-cost. Similarly, the price paid to an agent is nothing but the difference in the social cost of other players in her absence minus the social-cost of other players in her presence. Show that the VCG mechanism for this problem, i.e., deciding which links to buy and what to pay to the agents of the bought links, can be computed in polynomial time.

Answer: Let the given network be a graph (V, E), and let the set of agents be S. The mechanism gets reported costs of the links from the corresponding agents. The allocation rule in VCG computes the solution which maximizes the social surplus. Here, it is equivalent to minimizing the social cost c^*, which is nothing but the total (reported) cost of the edges bought. Thus, computing this is equivalent to computing a shortest-length path p^* with the reported edge-costs as the weights of the edges. Next, to decide the payments, VCG computes '(maximum social welfare without i) - (social welfare of agents in $S \setminus \{i\}$ with i)'. In this case, it translates to '(minimum social cost without i) - (social cost of agents in $S \setminus \{i\}$ for p^*)'. The latter is simply the social cost for p^* minus i's reported cost. Hence, it only needs to compute the minimum social cost when an agent (i.e., the corresponding edge) is excluded. For this, we can simply remove the edge, and compute the shortest-length path in the resultant graph (Since the graph is 2-edge connected, removing an edge still keeps it connected). Thus, computing the payment for each agent is one more shortest-length path computation. Thus, the whole mechanism can be implemented via $|S|+1$ shortest-length path problems which is at most $O(|E|)$ shortest-length path problems. Finally each shortest-length path problem can be computed by, say, Dijkstra's algorithm in $O(|V| \log |V| + |E|)$, thus, polynomial time suffices in total.

5. Consider the variant of stable matching problem, where the preference list can be incomplete, i.e., a woman (or a man) can exclude some men (or women) whom they does not want to be matched with.

- Extend the definition of stable matching for this case.

  Answer: Let E be a given matching. Think of agents unmatched in E as self-matched, i.e., matched to their self. We call a pair (w^*, m^*) unstable if the all of the following happens: (i)

they are in each others preference list. (ii) if $(w^*, v) \in E$ then either $v = w^*$ or $m^* >_{w^*} v$. (iii) if $(v, m^*) \in E$ then either $v = m^*$ or $w^* >_{m^*} v$. A matching is called stable if it has no unstable pair.

This can also be thought of as usual definition after modifying the preferences as follows: For any woman w, append her preference by w, and then add the men not in the preference in arbitrary order. Similarly for men.

- Show that all stable matching are of the same size.

  Answer: Suppose not. Let W and M denote the set of women and men in the system. Let E_1 and E_2 be two stable matching (no self matching here), such that $|E_1| < |E_2|$. Consider edges from E_1 and E_2 that are not common. There exists an odd length augmenting path among these with end-points $w^* \in W$ and $m^* \in M$. Let the path be $w^* - m_1 - w_1 - \cdots - m_k - w_k m^*$, i.e., containing $2k+1$ edges. The edges alternate between E_1 and E_2, where $(w^*, m_1), (w_1, m_2), \ldots, (w_k, m^*)$ are in E_2 and $(m_1, w_1), \ldots (m_k, w_k)$ are in E_1.

  Since E_2 is stable, we have that w^* prefers to match with m_1 than to be alone, and since E_1 is stable m_1 prefers w_1 to w^*. Now if w_1 prefers m_1 than m_2 then (w_1, m_1) form unstable pair in E_1, therefore w_1 prefers m_2 to m_1. Repeating this reasoning, we get that $\forall i$, w_i prefers her matching in E_2 than in E_1. This causes unstable pair (w_k, m^*) in E_1. A contradiction to E_1 being stable.

- Extend the deferred acceptance algorithm (*proposal* algorithm) for this case.

  Answer: First self-match all the men, and then run women proposing algorithm. A man accepts a proposal from a woman only if she is in her list, else he rejects.