CSE 516: Homework 1

Due: February 6, 2014

Note: Please keep in mind the collaboration policy as specified in the course syllabus. If you discuss questions with others you **must** write their names on your submission, and if you use any outside resources you **must** reference them. Please submit your writeup, but not your code. However, we may ask you to email us your code, and if you do not then you will not receive any credit. Finally, keep in mind that homework (in hardcopy) is due **at the beginning of lecture.** Note that several of the questions are quite open-ended. You will be graded in part on the quality of your analysis and in part on the quality of your writeup, so please write your answers up carefully. There are five questions on three pages.

1. (60 points) **The Assignment Problem:** In this problem you will explore the properties of the assignment problem and compare two different ways of solving it. We have broken it up into discrete tasks that build on each other.

   (a) (20 points) **Implementation:** Implement the auction algorithm in the language of your choice, using a representation that you deem appropriate. Also, make sure you have a general method for encoding assignment problems as linear programs using the linear programming modeling language of your choice (we recommend GLPK). Test these on small instances to make sure that you get correct answers. Once you're sure that your algorithms are working, solve the following instance of the assignment problem with ten agents and ten objects (agents are rows, objects are columns):

	O1	O2	O3	O4	O5	O6	O7	O8	O9	O10
A1	12	45	58	66	56	51	11	17	40	47
A2	17	99	45	22	21	40	39	24	56	21
A3	84	20	75	10	58	39	39	50	93	23
A4	45	7	2	66	71	75	8	96	5	0
A5	12	87	65	84	74	65	24	55	66	62
A6	72	28	46	97	59	31	36	18	71	0
A7	12	69	80	68	16	36	78	86	97	51
A8	97	66	94	80	27	21	74	59	96	74
A9	77	22	87	83	63	17	31	49	58	65
A10	74	6	47	92	4	47	86	11	75	98

   Report the actual assignment and the total value of that assignment.

   (b) (20 points) **Random problems and assignment values:** Now implement a way of generating a random assignment problem given two parameters, n and M, where n is the number of agents (there should be an equal number of objects), and the value of each assignment is an integer sampled uniformly at random between 0 and $M - 1$

(or 1 and M if you prefer). You should figure out a way to feed this problem to both the auction algorithm solver and the LP solver (if you are using GLPK, the best way to do this is to generate a separate data file, while keeping the model file to specify the common parts of the model). Now, using either solution technique, compute the per-agent average value of assignments as you increase n, in powers of 2, from 2 to 256. Average at least 1000 runs for each case of n. Plot the results and include this plot in your writeup. Explain why you see the pattern you see, backing up your claim with any specific evidence that you may want to gather from the instances you generate or the execution of your code.

(c) (20 points) **Timing:** Now, holding the number of agents constant at 256, change M, this time going up in orders of 10, from 10 to 100 to 1000 and all the way up to 10^7. Solve each generated instance using both the auction algorithm approach and the linear programming approach, keeping track of how long each solver takes on average to solve an instance as a function of M. You should run the code for at least 100 instances for any particular M to get a stable estimate. Plot the results for each of the two approaches. Why do you see the results you see? Are they what you expected from the worst case bounds on time discussed in class? Again, you should back up your claims with any specific evidence you deem appropriate from the instances you generate or the execution of your code.

2. (10 points) **Different objectives in stable matching:** Consider a standard stable matching problem with complete preference lists for both men and women. I want to find the stable matching that minimizes the average rank in the preference list of each agent's assigned partner. For example, if there are three men and three women and a stable matching assigns partners ranked 1, 2, and 2 (in their own preference lists) to the three men, and 1, 3, and 3 to the three women, then the average rank is 2. Write down a linear program (using the same conventions as the basic stable matching LP discussed in lecture) for achieving this objective.

3. (5 points) **Manipulation through truncation:** Suppose agents do not have to submit complete preference lists. An incomplete list indicates that an agent prefers to be single than matched with someone not on his or her list. Consider a matching market with two men and two women, so that m_1 has preferences $w_1 > w_2$, m_2 has preferences $w_2 > w_1$, w_1 has preferences $m_2 > m_1$ and w_2 has preferences $m_1 > m_2$. Suppose the mechanism to be run will be Gale-Shapley with men proposing, and that w_1 knows everyone's preferences, and that all the other agents will submit their true preferences. What preference list should she submit, and why?

4. (10 points) **Manipulation through permutation:** Consider a matching market with five men and five women with the following true preferences:

Men

1	2	3	4	5	1
2	3	4	5	1	2
3	5	1	4	2	3
4	3	1	2	4	5
5	1	5	2	3	4

Women

1	1	2	3	5	4
2	2	1	4	5	3
3	3	2	5	1	4
4	4	5	1	2	3
5	5	1	2	3	4

Suppose the mechanism is Gale-Shapley with men proposing and that all agents must submit complete preference lists. What should w_1 submit as her preference list if she knows that all the other agents will be truthful? Explain how you got your answer. (Any reasonable approach is OK, it doesn't necessarily have to scale.)

5. (15 points) **Uniqueness:** Prove that if the man optimal stable matching and the woman optimal stable matching are the same for a given instance of the stable matching problem, there is only one stable matching. (You may want to start by showing that, in a man optimal matching, each woman ends up with the lowest ranked man she could in any stable matching.)