Computational Microeconomics: Game Theory, Social Choice, and Mechanism Design

Homework Assignment 2 (due October 19 before or at the beginning of class)

Please read the rules for assignments on the course web page (http://www.cs.duke.edu/courses/fall18/compsci590.2/).
Use Piazza (preferred) or directly contact Harsh (harsh.parikh@duke.edu), Hanrui (hrzhang@cs.duke.edu), or Vince (conitzer@cs.duke.edu) with any questions.
Use Sakai to turn in the assignment.

1. (Properties of voting rules.)

Alice likes to analyze the outcomes of elections; specifically, she is interested in the different outcomes that different voting rules produce on the same votes. To do so, she executes many different rules on the same set of votes, a painstaking process. She likes knowing about properties of voting rules that ease her task. For example, she likes to know which voting rules satisfy the Condorcet criterion, so that if there is a Condorcet winner, she immediately knows that that will be the winner for those rules, without having to go through the trouble of executing each rule individually.

Recently, Alice has become interested in the phenomenon of votes "cancelling out." Let us say that a set[1] S of votes *cancels out with respect to voting rule* r if for **every** set T of votes, the winner[2] that r produces for T is the same as the winner that r produces for $S \cup T$. For example, the set of votes $\{a \succ b \succ c, b \succ a \succ c, c \succ a \succ b\}$ cancels out with respect to the plurality rule: each candidate is ranked first once in this set of votes, so it has no net effect on the outcome of the election. The same set does not cancel out with respect to Borda, though, because from these votes, a gets 4 points, b gets 3, and c gets 2, which may affect the outcome of the election. Alice likes to know when a set of votes cancels out with respect to a rule, so that she can just ignore these votes, easing her computation of the winner.

Define a pair of *opposite votes* to be a pair of votes with completely opposite rankings of the candidates, i.e. the votes can be written as $c_1 \succ c_2 \succ \ldots \succ c_m$ and $c_m \succ c_{m-1} \succ \ldots \succ c_1$. Let us say that a voting rule r satisfies the *Opposites Cancel Out (OCO)* criterion if every pair of opposite votes cancels out with respect to r.

[1] Technically, a multiset, since the same vote may occur multiple times.
[2] ... or set of winners if there are ties.

1a. (12 points) From among the (reasonable[3]) voting rules discussed in class, give 3 voting rules that satisfy the OCO criterion, and 3 that do not (and say which ones are which!).

Define a *cycle* of votes to be a set of votes that can be written as $c_1 \succ c_2 \succ \ldots \succ c_m, c_2 \succ c_3 \succ \ldots c_m \succ c_1, c_3 \succ c_4 \succ \ldots \succ c_m \succ c_1 \succ c_2, \ldots, c_m \succ c_1 \succ c_2 \succ \ldots \succ c_{m-1}$. Let us say that a voting rule r satisfies the *Cycles Cancel Out (CCO)* criterion if every cycle cancels out with respect to r.

1b. (12 points) From among the (reasonable) voting rules discussed in class, give 3 voting rules that satisfy the CCO criterion, and 3 that do not.

Define a pair of *opposite cycles* of votes to be a cycle, plus all the opposite votes of votes in that cycle. Note that these opposite votes themselves constitute a cycle, the opposite of which is the original cycle. Let us say that a voting rule r satisfies the *Opposite Cycles Cancel Out (OCCO)* criterion if every pair of opposite cycles cancels out with respect to r.

1c. (12 points) From among the (reasonable) voting rules discussed in class, give 5 voting rules that satisfy the OCCO criterion, and 1 that does not.

1d. (14 points) Criterion C_1 is *stronger* than criterion C_2 if every rule that satisfies C_1 also satisfies C_2. Two criteria are *incomparable* if neither is stronger than the other. For every pair of criteria among OCO, CCO, and OCCO, say which one is stronger (or that they are incomparable).

2. (A multi-unit auction with externalities.)

We are running a multi-unit auction for badminton rackets in the town Externa, where nobody owns one yet and we are the only supplier. Of course, being the only person to own a badminton racket is no fun; bidders care about which other bidders win rackets as well. In such a setting, where bidders care about what other bidders win, we say that there are *externalities*. Let us assume that each agent is awarded at most one racket, and that shuttlecocks and nets are freely available. In the most general bidding language for this setting, each bidder would specify, for every subset of the agents, what her value would be if exactly the agents in that subset won rackets. This is impractical because there are exponentially many subsets. Instead, we will consider more restricted bidding languages.

Let us suppose that it is commonly known which agents live close enough to each other that they could play badminton together. This can be represented as a graph, which has an edge between two agents if and only if they live close enough to each other to play together.

In the first bidding language, every agent i submits a single value v_i. The semantics of this are as follows. If the agent does not win a racket, her utility is 0 regardless of who else wins a racket. If she does win a racket, her value is v times the number of her neighbors that also win a racket.

[3]E.g., not dictatorial rules, rules for which there is a candidate that can't possibly win, randomized rules, etc. Also, approval cannot be one of the rules because it is not based on rankings. If you use Cup, Cup only satisfies a criterion if it satisfies it for every way of pairing the candidates.

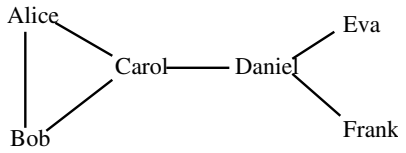

Figure 1: Externa's proximity graph.

Suppose we receive the following bids:

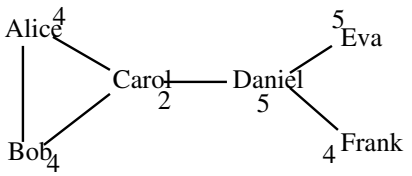

Figure 2: Graph with bids. The number next to an agent is that agent's bid.

Suppose we have three rackets for sale. One valid (but not optimal) allocation would be to give rackets to Carol, Daniel, and Eva. Carol would get a (reported) utility of 2, Daniel would get 10 ($2 \cdot 5$, because two of Daniel's neighbors have rackets), and Eva 5, for a total of 17.

2a. (12 points) Give the optimal allocation, as well as the VCG (Clarke) payment for each agent.

2b. (13 points) In general (general graphs, bids, numbers of rackets), is the problem of finding the optimal allocation solvable in polynomial time, or NP-hard? (Hint: think about the Clique problem (which is almost the same as the Independent Set problem).)

One year has passed, and we have returned to Externa. Everyone's rackets have broken (we are not in the business of selling high-quality rackets here) and they need new ones. However, the people in the town were not entirely happy with our previous system. Specifically, it turned out that each agent only ever played with (at most) a single other agent, so that multiplying the value by the number of neighbors with rackets really made no sense. Also, agents have realized that they would receive different utilities for playing with different agents.

In the new system, we must not only decide on who receives rackets, but (for the agents who win rackets) we must also decide on the pairing, i.e.,who plays with whom. Each agent can be paired with at most one other agent. Each agent i submits a value v_{ij} for every one of her neighbors j; agent i receives v_{ij} if she is paired with j (and both win rackets), and 0 otherwise.

Suppose we receive the following bids:

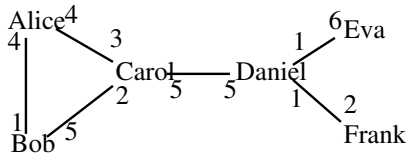

Figure 3: Graph with bids. Each number is the value that the closer agent on the edge has for playing with the further agent on the edge.

Suppose we have four rackets for sale. One valid (but not optimal) outcome would be to pair Alice and Bob, and Daniel and Eva (and give them all rackets), for a total utility of $4 + 1 + 1 + 6 = 12$.

2c. (12 points) Give the optimal outcome (pairing and allocation), as well as the VCG (Clarke) payment for each agent.

2d. (13 points) In general (general graphs, bids, numbers of rackets), is the problem of finding the optimal outcome solvable in polynomial time, or NP-hard? (Hint: think about the Maximum-Weighted-Matching problem. Keep in mind that the number of rackets is limited, though.)