Homework #5

Deadline: Tuesday, 19 May 2020, 18:00

You may submit this assignment either on your own or in a group of two students.

Question 1 (10 marks)

Recall the program presented in class to automatically prove the Gibbard-Satterthwaite Theorem for the special case of $n = 2$ voters and $m = 3$ alternatives. The purpose of this exercise is to explore some further applications of this implementation.

For $n = 2$ and $m = 3$, how many different resolute voting rules are there that are strategyproof? Answer this question by building on the program presented in class. Then provide a clear description and a suitable classification of these rules. (For instance, some of them will be dictatorships.) For this second part of the exercise, you may either extend our program further or you may resort to purely theoretical means. Either way, do not submit a program file for this exercise, but rather document what you have done within the PDF you submit (possibly showing relevant code snippets).

Question 2 (10 marks)

Recall that the Duggan-Schwartz Theorem establishes the impossibility of designing a (possibly irresolute) voting rule that, simultaneously, is (i) nonimposed, (ii) immune to manipulation by optimistic voters, (iii) immune to manipulation by pessimistic voters, and (iv) strongly nondictatorial. Prove this theorem for the special case of $n = 2$ voters and $m = 3$ alternatives using the SAT solving technique.

Reuse anything you find helpful from the program for the verification of the "base case" of the Gibbard-Satterthwaite Theorem presented in class (but clearly indicate which code you have copied, and whether you have altered that code or left it unchanged).

Hints: This is a difficult exercise, although modelling the requirement of the voting rule being strongly nondictatorial is relatively straightforward. So start with that. Modelling the two strategyproofness axioms requires some careful thinking, but you should end up with a fairly simple implementation as well. The main challenge is modelling nonimposition, which most immediately corresponds to a conjunction of disjunctions of conjunctions of literals. Translating this into CNF is impractical: the resulting formula would be huge (a conjunction of almost half a quintillion clauses of length 36). But you can use this trick: Introduce auxiliary variables $q_{\boldsymbol{R},x}$ with the intended meaning that in profile $\boldsymbol{R}$ alternative x is the *only* winner. Then express nonimposition with the help of these auxiliary variables, and fix their meaning by adding clauses that together enforce $q_{\boldsymbol{R},x} \leftrightarrow p_{\boldsymbol{R},x} \wedge \neg p_{\boldsymbol{R},y} \wedge \neg p_{\boldsymbol{R},z}$ for all profiles $\boldsymbol{R}$ and (distinct) alternatives x, y, and z.

To help us understand your solution, for every axiom you implement, please report the number of clauses this axiom corresponds to.

Besides proving the theorem, also demonstrate that for each of the four axioms featuring in the theorem it is possible to design a voting rule that satisfies the other three axioms (again, for the special case of $n = 2$ and $m = 3$). Report *how many* such voting rules there are for each of those four cases. Keep in mind that this corresponds to very demanding queries for the SAT solver, so you may not be able to obtain an answer in a reasonable amount of time. If one of the relevant queries does not return an answer within 15 minutes, please simply report this timeout instead of the relevant number of voting rules.

Submission: Besides the usual PDF with your solutions, also submit your program (if at all possible, this should be a single unzipped Python file with a modest but reasonable degree of in-file documentation). In principle, we intend to understand and grade your contribution on the basis of your PDF alone and will consult your program only in case we want to check a specific detail or verify one of your claims. So you probably will want to include some relevant code snippets in your PDF as well.

Question 3 (10 marks)

The International Conference on Autonomous Agents and Multiagent Systems (AAMAS) is one of the main conferences where work on computational social choice gets presented. Thanks to the coronavirus, the 2020 edition of the conference, which was supposed to take place in New Zealand, will be held as an online event. The good news is that attendance is free and open to everyone. Watch a talk corresponding to a full paper in the main conference programme on a topic that is broadly related to the course and write a report. Consult the paper corresponding to the talk where necessary, but try to get as much as possible out of the talk itself. If you are submitting in a group of two, then report on two talks. Each report should be around two pages long and it should cover the following points:

- Title and full names of all authors of the paper. Direct links to paper and talk.
- Write a short summary of the paper (or the part of the paper discussed in the talk), mentioning the research area, the question or problem addressed, the methodology employed, and the results obtained. Write at most half a page of text for this part.
- Report on the history of the paper. Questions you might consider include: Does the paper answer a question raised in an earlier paper? Have the same authors collaborated before, maybe on the same topic? Is the paper part of a larger project?
- Paraphrase what you consider to be the most interesting contribution discussed in the talk. This might be a single theorem proved or a single experiment conducted.
- Identify a weakness of the paper. In principle, this could be a technical mistake, but more likely it will be an unrealistic assumption or similar.
- Sketch a possible direction for future work one might take to build on the work reported on in the paper (and that is not already mentioned in the paper itself).

Try to be led by the talk rather than the paper, formulate things in your own words rather than the words of the speaker or the authors of the paper, and wherever applicable use the terminology of the course (which might be subtly different from the speaker's terminology).

Hint: Navigating the technical programme of a large conference is not easy, so set aside some time to identify talks you might find interesting and that are relevant to this assignment.