1 Class Projects

The goal of the final project is to develop a deep understanding of an important open research area, and, to the extent possible, to work on an open research problem. You may also review an existing area of the literature, providing a careful and critical comparison of different approaches to computational mechanism design. Projects may be theoretical or experimental, theoretical projects must be individual, experimental projects can be in pairs. A list of suggested topics for projects are provided here, but you are absolutely encouraged to propose a topic of your own for approval.

Project proposals are due Thursday, April 4, to Arthur Cram in Maxwell-Dworkin 133, by noon.

Your proposal must:

• Provide a high-level description of your project.
• Provide motivation for your proposed work. (what will we know afterwards that we don’t know now, and why do we care)
• Place your work in the context of the papers that we read in the course.
• Convince me that your ideas are well enough focused, and that you know what the first few steps will be.
• Let me know what things can I help with to get you started?

We will meet the morning of Friday, April 5, to discuss your project choices (I’ll leave a sign-up sheet for 15 minute slots outside my office).
Project presentations will be during reading week, and your final project report is due the last day of reading week, Wednesday May 15.

2 Project Suggestions

Note: Many of these projects could be either theoretical or experimental. In addition, in some of these problems there might be a place where you want to solve combinatorial optimization problems (e.g. winner-determination problems). Expect to be able to use off-the-shelf integer programming software (CPLEX) for this purpose, that you can call from either Java or C++. We will have this installed somewhere within the DEAS computing environment by the end of Spring break.

- **shortest-path problems.** Propose and analyze the incentive and computational properties of mechanisms for extensions of the Nisan-Ronen shortest-path problem, for example: (i) allowing agents to have costs for more than one edge; (ii) providing group-strategyproofness; (iii) introducing non-linear cost functions; (iv) multiple sender and receivers; (v) large messages, that can congest any single path; etc.

- **maximal-in-range.** Consider the tractable algorithms for the combinatorial allocation problem in the de Vries & Vohra (dVV) survey paper. Which of them can be used to build truthful approximate VCG mechanisms using Nisan& Ronen’s maximal-in-range concept? First, you might ask which are maximal-in-range with no assumptions on agent preferences, but for a bidding language suitably restricted to make the dVV algorithm apply. Then, you might ask what assumptions must be made about agent preferences to make the algorithms maximal-in-range, or whether simple extensions can make any of the algorithms maximal-in-range.

- **maximal-in-range.** Apply a maximal-in-range framework to design a fast VCG mechanism for an interesting multi-agent algorithmic problem from computer science. Perhaps compare its performance empirically with an optimal VCG solution (in terms of computation and economic efficiency).

- **incremental preference elicitation.** Consider a tractable and truthful VCG based setting, in which a suitable bidding language is provided, suitable assumptions about agent preferences are made, and a maximal-in-range algorithm is provided. Can incremental preference
elicitation be folded into this mechanism, i.e. can the algorithm to
determine the outcome of the VCG work with partial information and
just ask additional information from the agents as necessary. How
might this incremental information elicitation change the incentives of
the mechanism. Can any residual approximation in the information
provided by an agent be used to bound the accuracy of the solution
computed by the mechanism.

- **partition/maximal-in-range.** Given a particular distribution of agent
valuations, can you design an automated search method to select the
optimal partition (of a particular size) to maximize average case effi-
ciency? (Restrict agents to submit bids on the field constructed from
the partition, and use a maximal-in-range allocation algorithm).

- **partition/maximal-in-range.** Given a particular distribution of agent
valuations, perform empirical tests to compare communication vs. eco-
nomic efficiency for different partition refinements (e.g. from a par-
tition with one big bundle, all the way down towards partitions with
mostly single items).

- **winner-determination.** Develop integer programming formulations that
capture the structure of some of the tractable special-cases of winner-
determination in the de Vries & Vohra paper, and use CPLEX to in-
vestigate whether formulation matters (comparing compute time with
standard formulations). [A paper accepted to AAAI this year demon-
strated that CPLEX on-top of a logic based language performs well].

- **approximate strategyproofness.** Connect bounds on the worst-case
(or average-case) approximation to maximal-in-range in a VCG-based
mechanism to bounds on the worst-case (or average-case) gains to
agents from non-truth-revelation. Perhaps also discuss this in the
context of Nisan&Ronen’s “appeal functions”, from the second half
of the Computationally Feasible VCG Mechanism paper (that was not
discussed in class).

- **randomized mechanism design.** Empirical analysis on the best choice of
$\beta$ in Nisan&Ronen’s weighted-VCG mechanism for the minimal make-
span problem; perhaps on well-motivated problem distributions (e.g.
from a distributed systems application domain).

- **approximate mechanism design.** Apply a similar VCG-based method-
ology to that adopted in the Nisan&Ronen minimal make-span prob-
lem to provide a truthful but approximate solution to a problem in which a utilitarian (or social-welfare maximizing outcome) is not optimal.

- **randomized strategyproofness.** Apply the concept of Universal Domiance (again, Nisan&Ronen) to construct better mechanisms from a class of mechanisms, e.g. from a family of greedy-algorithms in the Lehmann et al. single-minded bidder paper, or a family of partition-based maximal-in-range VCG based mechanisms from the Monderer et al. paper.

- **randomized strategyproofness.** Weaken the concept of Universal Domiance to ex ante dominance (with respect to the randomized decision of the mechanism), such that a truthful strategy dominates all other strategies for the agent, and all strategies of other agents, but only in expectation with respect to the randomization within the mechanism. Investigate whether this weaker solution concept helps to improve the performance of a mechanism, for example in the minimal make-span problem. Basically, with this weaker notion it will sometimes be acceptable to allow a non-truthful strategy to have a higher utility for an agent ex post.

- **that reasonableness claim.** Rework Nisan&Ronen’s formulation of reasonableness to be able to make a contrary statement like, “many truthful, non-optimal, and reasonable mechanisms exist”, or (better?), “for any non-reasonable and truthful mechanism, \( M \), I can generate a reasonable and truthful mechanism, \( M' \), that is more optimal” (but without being fully optimal).

- **investigate a recent Ronen paper relating maximal in range to bidding languages.** Ronen has a recent paper in which he makes arguments about preference languages that are not suitably expressive, vis-a-vie a particular VCG-based mechanism and maximal-in-range. Look at this paper, relate to the maximal-in-range paper we read in class, and relate to the Monderer et al. bundling equilibrium paper.

- **bounded-rational Nash equilibria.** Investigate definitions of bounded-rational Nash equilibria in the literature, compare and critique, perhaps provide alternatives, and discuss in the context of mechanism design.
• **using bounded-rationality.** Use bounded-rationality as a “hammer” to escape some impossibility results in the mechanism design literature [e.g. efficiency and budget-balance], or at least to demonstrate that we can design mechanisms in which there is a wide plateau around truthfulness that is no better than truthfulness, and for which agents must be quite “bold” to manipulate. [I can point you to a recent paper of Parkes, Kalagnanam & Eso on combinatorial exchange design with some of these ideas].

• **voting mechanisms.** Survey the current literature on the difficulty of manipulating voting mechanisms, which has made arguments about the bounded-rationality of the voting populace (!) [again, I can point you in the right direction, and there are also a couple of very recent AAAI papers on this topic].

• **single-minded.** Design a family of maximal-in-range VCG-based fast mechanisms for single-minded bidders. Compare performance to Lehmann et al.’s (non-VCG based) greedy mechanisms.

• **many-minded.** Design a group-strategyproof mechanism for many single-minded bidders that extends to strategyproofness for bidders that may themselves be “multi-minded” (you know what I mean). [perhaps this would use a cross-monotonic cost-sharing method?]

• **hard valuation problems.** Develop a theoretical model to formalize why iterative mechanisms would be expected to have better computational and economic properties than direct-revelation mechanisms in settings where agents have hard valuation problems. [i.e. attack the revelation principle!]

• **anytime truthfulness.** Design direct-revelation mechanisms with an anytime truthfulness property, such that truth-revelation is a dominant strategy for every agent whenever the computation of the center is actually stopped (for example because the auction has to clear within an hour, and that is too short a time to solve the optimal winner-determination problem, but we want to use as much of that compute time usefully as possible).

• **sequential truthfulness.** Design mechanisms to provide strategyproofness in multi-period auctions, in which an auctioneer must sell goods every period and agents arrive into the system and must decide in which periods to bid, and how much. One nice property that we might
like is that an agent’s dominant strategy is to go ahead and bid truthfully as soon as it arrives, whatever the future competition from other agents is in later periods. [this is also related to interesting recent work on on-line clearing algorithms, that try to provide worst-case guarantees with respect to the best possible decisions with hindsight, and work on revenue-maximization in yield management problems such as those faced by airlines.]

- **median mechanisms.** Consider generalizations of Moulin’s median mechanism with phantom peaks to multiple dimensions. Perhaps bound the space available to agents for manipulation, based on bids submitted by other agents (i.e. how far can the median move?). Another type of project could consider an interesting application of the median mechanisms to a relevant problem in distributed systems.

- **iterative mechanism design.** Design fast versions of iterative combinatorial auctions (e.g. iBundle) for special preference structures, e.g. some of those from the de Vries& Vohra paper. What assumptions about agent preferences lead to bids in each round of the auction with a structure that makes the winner-determination problem easy to solve? Similarly, for a particular set of assumptions about agent preferences, do ask prices always have a particular structure (perhaps a structure similar to agent preferences)?

- **ascending auctions.** In ascending auctions it is often necessary for the mechanism to solve a winner-determination problem in each round. Agents’ bids typically change only slowly over the course of an auction, and optimal provisional allocations also change only slowly. Design algorithms for this sequential setting, that leverage computation done in previous rounds to speed up computation in the current round. Perhaps it is also OK to make mistakes in some rounds, so long as all mistakes are eventually caught? [As an example, Parkes& Ungar investigate some “flip-flop” heuristics in their AAAI’00 combinatorial auction paper.]

- **proxy bidding agents.** Design (and test?) algorithms for proxy bidding agents in ascending combinatorial auctions, that sit in-between the auction and the real auction participants. The proxy agents receive incremental information about user preferences, and play myopic best-response at current prices whenever there is enough information. Otherwise, the proxy agents elicit additional information about user
preferences, and check this information is consistent with current information. [the key will be to use good data structures to check consistency, I think perhaps Simple Temporal Networks.]

- **proxy bidding agents.** Assuming that proxy agents are able to check the consistency of preference information, investigate the ability of proxy agents to limit collusion between participants in ascending-price Vickrey auctions [by forcing preferences to be consistent across the course of an auction]. Reducing the number of times that a user can provide information reduces the opportunities for collusion, but increases average preference elicitation costs (and making an interesting tradeoff).

- **qualitative & truthful mechanisms.** Qualitative reasoning asks agents for approximate information about preferences (e.g. dark colors are preferred to light colors, faster is better than slower), and then asks for more accurate information as required. Design mechanisms that are incentive-compatible and support qualitative preference elicitation, with relevant refinements from users requested as more accuracy is required.

- **ascending auctions.** Design an asynchronous version of iBundle, in which prices are increased based on a subset of bids from all agents (you’ll be able to use a similar method to that in iBEA). Compare the average preference elicitation with and without asynchronous price updates, on interesting problem distributions, and also compare the effect of price discrimination of preference elicitation costs across auction variations iBundle(2), iBundle(d), and iBundle(3) (with no price-discrimination prices, dynamic price-discrimination, and full price-discrimination respectively). [we should be able to port my current iBundle code to run locally]

- **distributed mechanism design.** Design distributed VCG-based mechanisms for system resource allocation and coordination problems, for example web caching, load balancing, optimal information aggregation from multiple information sources, peer-to-peer systems, etc.

- **combinatorial exchanges.** Empirical work computing equilibria of a combinatorial exchange mechanism. Essentially the idea is to write an automated search algorithm to compute restricted Nash equilibrium in an exchange mechanism, restricted in the sense that only particular classes of agent strategies are considered. [we should be able to port my current combinatorial exchange code to run locally]
• multiattribute auctions. Run experiments to compare preference elicitation costs in iterative and direct-revelation multiattribute auctions. [we have a very simple prototype to perform basic measurements, but have only performed less than a day of tests.]