

Computational Aspects of Preference Aggregation

Vincent Conitzer

CMU-CS-06-145

July 2006

School of Computer Science
Computer Science Department
Carnegie Mellon University
Pittsburgh, PA

Thesis committee:

Tuomas Sandholm, Chair

Avrim Blum

Tom Mitchell

Craig Boutilier (University of Toronto)

Christos Papadimitriou (University of California, Berkeley)

*Submitted in partial fulfillment of the requirements
for the Degree of Doctor of Philosophy.*

Copyright © 2006 Vincent Conitzer

This research was sponsored by the National Science Foundation under grant nos. IIS-0234694, IIS-0427858, IIS-0234695, and IIS-0121678, as well as a Sloan Fellowship awarded to Tuomas Sandholm, and an IBM Ph.D. Fellowship. The views and conclusions contained in this document are those of the author and should not be interpreted as representing the official policies, either expressed or implied, of any sponsoring institution, the U.S. government or any other entity.

Keywords: Artificial Intelligence, Multiagent Systems, Electronic Commerce, Game Theory, Mechanism Design, Auctions and Exchanges, Voting.

Abstract

In a *preference aggregation setting*, a group of agents must jointly make a decision, based on the individual agents' privately known preferences. To do so, the agents need some protocol that will elicit this information from them, and make the decision. Examples include voting protocols, auctions, and exchanges. *Mechanism design* is the study of designing preference aggregation protocols in such a way that they work well in the face of strategic (self-interested) agents. In most real-world settings, mechanism design is confronted with various computational issues. One is the complexity of *executing* the mechanism. Particularly in *expressive* preference aggregation settings (such as combinatorial auctions), many mechanisms become hard to execute. Another is the complexity of *designing* the mechanism. When general mechanisms do not apply to, or are suboptimal for, the setting at hand, a custom mechanism needs to be designed for it, which is a nontrivial problem that is best solved by computer (*automated mechanism design*). Finally, there is the complexity of *participating* in the mechanism. In complex settings, agents with limited computational capabilities (*bounded agents*) will not necessarily be able to act optimally, which should be taken into account in the mechanism design process.

My thesis statement is that **we can employ the study of computational aspects of the mechanism design process to significantly improve the generated mechanisms in a hierarchy of ways**, leading to better outcomes (and a more efficient process). The dissertation outlines this hierarchy, and illustrates and addresses representative issues at various levels of the hierarchy with new results. It also serves as a significant step towards a longer-term research goal: realizing a mechanism design approach that addresses all of these issues simultaneously, comprehensively, and optimally, in settings with real-world complexity.

Acknowledgements

At times our own light goes out and is rekindled by a spark from another person. Each of us has cause to think with deep gratitude of those who have lighted the flame within us.

Albert Schweitzer

First and foremost, I want to thank my advisor, Tuomas Sandholm. Tuomas has been a great advisor. Over the course of countless research meetings, he has taught me how to take a loose idea or result and push it to its limits by discovering variations and generalizations, implications and applications. He has also taught me to think of the big picture and present my work accordingly. In addition, Tuomas' advice has extended beyond research to teaching, career planning, and even personal issues; and he has always been willing to take extra time out of his busy schedule when important or unusual issues came up. Meanwhile, he worked behind the scenes to make sure that I did not have to worry about funding or administrative problems, and could focus on research. Finally, and perhaps most importantly, Tuomas has been endlessly motivating and encouraging. Tuomas brings his infective energy and enthusiasm to everything he does, and it is amazing how he manages to stay involved in so many different threads.

I also especially want to thank the other members of my committee, Avrim Blum, Craig Boutilier, Tom Mitchell, and Christos Papadimitriou, for their valuable feedback on this dissertation. Few people are fortunate enough to have such a fantastic committee.

Special thanks go to Barbara Grosz and Avi Pfeffer for sparking my initial interest in AI and putting me on the way to graduate school, allowing me to spend the last five years to think about more interesting problems than how to spend a large salary. Barbara, Avi, and David Parkes have since made sure that I continue to feel a part of Harvard's computer science department by including me in various events at conferences.

Working at CombineNet has taught me much about how my work relates and applies to practice, and I am grateful for all the interactions that I have had with the great people there, including Bryan Bailey, Egon Balas, Craig Boutilier, Michael Concordia, Andrew Fuqua, Andrew Gilpin, Sam Hoda, David Levine, Paul Martyn, Jim McKenzie, George Nemhauser, David Parkes, Rob Shields, Yuri Smirnov, Brian Smith, and Subhash Suri. I also want to thank Andrew Davenport, Jayant Kalagnanam, and everyone else that I interacted with at IBM Research for a wonderful and productive summer there. I am also very grateful for the IBM Ph.D. Fellowship that supported me this past year (as well as for all the NSF funding that funded the remainder of my studies).

Back at CMU, I want to thank Andrew Gilpin for organizing the Game Theory Discussion Group for multiple years, which has been a great forum for us to discuss and explore our research. I have also benefited greatly from many technical discussions with Kate Larson, Andrew Gilpin, Anton Likhodedov, David Abraham, Rob Shields, Benoît Hudson, Alex Nareyek, Paolo Santi, Felix Brandt, Marty Zinkevich, Rudolf Müller, Nina Balcan, Michael Benisch, Michael Bowling, Shuchi Chawla, Liz Crawford, George Davis, Jon Derryberry, Nikesh Garera, Daniel Golovin, Jason Hartline, Sam Hoda, Sham Kakade, Zhijian Lim, Daniel Neill, XiaoFeng Wang, Andrew Moore, Roni Rosenfeld, Manuela Veloso, and many others.

I am also indebted to many other researchers in distributed AI, multiagent systems, electronic commerce, and economics with whom I have had valuable discussions. While the following list is undoubtedly incomplete (not only due to my failing memory, but also due to the anonymity of the review process), special thanks go out to Alon Altman, Moshe Babaioff, Liad Blumrosen, Giro Cavallo, Raj Dash, Edith Elkind, Boi Faltings, Kobi Gal, Rica Gonen, Georg Gottlob, Amy Greenwald, Barbara Grosz, Joe Halpern, Edith Hemaspaandra, Nathanaël Hyafil, Sam Jeong, Atsushi Iwasaki, Adam Juda, Radu Jurca, Gal Kaminka, Michael Kearns, Sebastián Lahaie, Jérôme Lang, Ron Lavi, Kevin Leyton-Brown, Victor Lesser, Michael Littman, Bob McGrew, Ahuva Mu'alem, Eugene Nudelman, Naoki Ohta, David Parkes, David Pennock, Adrian Petcu, Avi Pfeffer, Ryan Porter, Ariel Procaccia, Juan Antonio Rodríguez-Aguilar, Amir Ronen, Jeff Rosenschein, Michael Rothkopf, Rahul Savani, Jeffrey Shneidman, Grant Schoenebeck, Peter Stone, Moshe Tennenholtz, Rakesh Vohra, Bernhard von Stengel, Éva Tardos, Eugene Vorobeychik, Michael Wellman, Makoto Yokoo, and Shlomo Zilberstein.

Fortunately, my years in Pittsburgh were not all work and no play. I especially want to thank Atul, Kevin, Dave, Anand, Kristen, Lucian, Monica, Vahe, Jernej, Stefan, Bianca, Naz, Nikesh, Leo, Mugizi, and Paul for being such good friends here for many years. I will also miss everyone from Real Mellon Soccer, but am comforted to know that you are left in Aaron's capable hands (and feet).

I would not be who I am today without my mom, my dad, Jaap, Marischa, Ruben, and Jessica. I want to thank my family for their unconditional love, and am sad to always be living so far away from them. Perhaps technology will eventually make the distance less significant.

Finally, I want to thank Christina for her love and support, especially in this last hectic year. While I was working on such things as proving the #P-hardness of manipulating randomly and incrementally strategy-proofed preference aggregation mechanisms, she took care of less essential details such as food, clothing, and finding a place to live next year. More importantly, she makes my life more enjoyable. Christina, I love you and hope that you did not miss me too much while I was working on this dissertation.

Contents

1	Introduction	13
1.1	A hierarchy of uses for computation in preference aggregation	14
1.2	The hierarchy's nodes illustrated by example	14
1.2.1	Node (1): Outcome optimization	16
1.2.2	Node (2): Mechanism design	16
1.2.3	Node (3a): Automated mechanism design	18
1.2.4	Node (3b): Mechanism design for bounded agents	18
1.2.5	Node (4): Automated mechanism design for bounded agents	20
1.3	How to use the hierarchy	20
1.3.1	Interpretation	20
1.3.2	Complexity of node vs. complexity of setting	20
1.3.3	Are the shallow nodes outdated?	21
1.4	Orthogonal research directions	23
1.5	Outline	24
2	Expressive Preference Aggregation Settings	27
2.1	Voting over alternatives (rank aggregation)	28
2.2	Allocation of tasks and resources	32
2.3	Donations to (charitable) causes	35
2.3.1	Definitions	36
2.3.2	A simplified bidding language	37
2.3.3	Avoiding indirect payments	38
2.4	Public goods and externalities	39
2.5	Summary	41
3	Outcome Optimization	43
3.1	A preprocessing technique for computing Slater rankings	43
3.1.1	Definitions	44
3.1.2	Sets of similar candidates	45
3.1.3	Hierarchical pairwise election graphs can be solved in linear time	47
3.1.4	An algorithm for detecting sets of similar candidates	49
3.1.5	Experimental results	51
3.1.6	NP-hardness of the Slater problem	53

3.1.7	Extension to the Kemeny rule	56
3.2	Combinatorial auctions with structured item graphs	57
3.2.1	Item graphs	60
3.2.2	Clearing with bounded treewidth item graphs	60
3.2.3	An algorithm for constructing a valid item tree	62
3.2.4	Constructing the item graph with the fewest edges is hard	64
3.2.5	Applications	66
3.2.6	Bids on multiple connected sets	69
3.3	Expressive preference aggregation for donations to charities	73
3.3.1	Hardness of clearing the market	73
3.3.2	Mixed integer programming formulation	76
3.3.3	Why one cannot do much better than linear programming	77
3.3.4	Quasilinear bids	79
3.4	Expressive preference aggregation in settings with externalities	82
3.4.1	Hardness with positive and negative externalities	83
3.4.2	Hardness with only negative externalities	84
3.4.3	An algorithm for the case of only negative externalities and one variable per agent	85
3.4.4	Maximizing social welfare remains hard	90
3.4.5	Hardness with only two agents	91
3.4.6	A special case that can be solved to optimality using linear programming	92
3.5	Summary	92
4	Mechanism Design	95
4.1	Basic concepts	95
4.2	Vickrey-Clarke-Groves mechanisms	98
4.3	Other possibility results	100
4.4	Impossibility results	101
4.5	Summary	102
5	Difficulties for Classical Mechanism Design	103
5.1	VCG failures in combinatorial auctions and exchanges	104
5.1.1	Combinatorial (forward) auctions	105
5.1.2	Combinatorial reverse auctions	107
5.1.3	Combinatorial forward (or reverse) auctions without free disposal	112
5.1.4	Combinatorial exchanges	117
5.2	Mechanism design for donations to charities	118
5.2.1	Strategic bids under the first-price mechanism	118
5.2.2	Mechanism design in the quasilinear setting	119
5.2.3	Impossibility of efficiency	120
5.3	Summary	122

6	Automated Mechanism Design	123
6.1	The computational problem	125
6.2	A tiny example: Divorce settlement	126
6.2.1	A benevolent arbitrator	126
6.2.2	A benevolent arbitrator that uses payments	127
6.2.3	An arbitrator that attempts to maximize the payments extracted	128
6.3	Complexity of designing deterministic mechanisms	129
6.4	Linear and mixed integer programming approaches	135
6.5	Initial applications	137
6.5.1	Optimal auctions	137
6.5.2	Public goods problems	140
6.6	Scalability experiments	143
6.7	An algorithm for single-agent settings	145
6.7.1	Application: One-on-one bartering	146
6.7.2	Search over subsets of outcomes	147
6.7.3	A heuristic and its admissibility	148
6.7.4	The algorithm	149
6.7.5	Individual rationality	151
6.7.6	Experimental results	152
6.8	Structured outcomes and preferences	156
6.8.1	Example: Multi-item auctions	157
6.8.2	Complexity	157
6.8.3	A pseudopolynomial-time algorithm for a single agent	163
6.8.4	A polynomial-time algorithm for randomized mechanisms	164
6.9	Summary	165
7	Game-Theoretic Foundations of Mechanism Design	167
7.1	Normal-form games	167
7.1.1	Minimax strategies	168
7.1.2	Dominance and iterated dominance	168
7.1.3	Nash equilibrium	170
7.2	Bayesian games	171
7.3	Revelation principle	173
7.4	Summary	175
8	Mechanism Design for Bounded Agents	177
8.1	A failure of the revelation principle with bounded agents	178
8.2	Tweaking voting protocols to make manipulation hard	181
8.2.1	Definitions	182
8.2.2	NP-hardness when scheduling precedes voting	183
8.2.3	#P-hardness when voting precedes scheduling	187
8.2.4	PSPACE-hardness when scheduling and voting are interleaved	190
8.3	Hardness of manipulating elections with few candidates	193
8.3.1	Manipulating an election	194

8.3.2	Algorithm for individually manipulating the STV rule	196
8.3.3	Complexity of weighted coalitional manipulation with few candidates . . .	200
8.3.4	Effect of uncertainty about others' votes	211
8.4	Nonexistence of usually-hard-to-manipulate voting rules	216
8.4.1	Definitions	217
8.4.2	Impossibility result	219
8.4.3	Arguing directly for Property 1	220
8.4.4	Arguing experimentally for Property 1	221
8.4.5	Can the impossibility be circumvented?	223
8.5	Summary	227
9	Computing Game-Theoretic Solutions	231
9.1	Dominance and iterated dominance	232
9.1.1	Dominance (not iterated)	233
9.1.2	Iterated dominance	234
9.1.3	(Iterated) dominance using mixed strategies with small supports	239
9.1.4	(Iterated) dominance in Bayesian games	243
9.2	Nash equilibrium	247
9.2.1	Equilibria with certain properties in normal-form games	247
9.2.2	Inapproximability results	253
9.2.3	Counting the number of equilibria in normal-form games	256
9.2.4	Pure-strategy Bayes-Nash equilibria	256
9.3	A generalized eliminability criterion	258
9.3.1	A motivating example	259
9.3.2	Definition of the eliminability criterion	260
9.3.3	The spectrum of strength	261
9.3.4	Applying the new eliminability criterion can be computationally hard . . .	263
9.3.5	An alternative, equivalent definition of the eliminability criterion	266
9.3.6	A mixed integer programming approach	267
9.3.7	Iterated elimination	268
9.4	Summary	271
10	Automated Mechanism Design for Bounded Agents	273
10.1	Incrementally making mechanisms more strategy-proof	274
10.1.1	Definitions	275
10.1.2	Our approach and techniques	275
10.1.3	Instantiating the methodology	277
10.1.4	Computing the outcomes of the mechanism	285
10.1.5	Computational hardness of manipulation	286
10.2	Summary	288

11 Conclusions and Future Research	289
11.1 Contributions	289
11.2 Future research	291
11.2.1 Node (1): Outcome optimization	292
11.2.2 Node (2): Mechanism design	293
11.2.3 Node (3a): Automated mechanism design	294
11.2.4 Node (3b): Mechanism design for bounded agents	295
11.2.5 Node (4): Automated mechanism design for bounded agents	296

