computing in mechanism design

1 Introduction

Computational issues in mechanism design are important, but have received insufficient research interest until recently. Limited computing hinders mechanism design in several ways, and presents deep strategic interactions between computing and incentives. On the bright side, novel algorithms and increasing computing power have enabled better mechanisms. Perhaps most interestingly, limited computing of the agents can be used as a tool to implement mechanisms that would not be implementable among computationally unlimited agents. This article briefly reviews some of the key ideas, with the goal of alerting the reader to the importance of these issues and hopefully spurring future research.

I will discuss computing by the *centre*, such as an auction server or vote aggregator, in Section 2. Then, in Section 3, I will address the *agents*' computing, be they human or software.

2 Computing by the centre

Computing by the centre plays significant roles in mechanism design. In the following three subsections I will review three prominent directions.

2.1 Executing expressive mechanisms

As algorithms have advanced drastically and computing power has increased, it has become feasible to field mechanisms that were previously impractical. The most famous example is a *combinatorial auction* (*CA*). In a CA, there are multiple distinguishable items for sale, and the bidders can submit bids on self-selected packages of the items. (Sometimes each bidder is also allowed to submit exclusivity constraints of different forms among his bids.) This increase in the expressiveness of the bids drastically reduces the strategic complexity that bidders face. For one, it removes the exposure problems that bidders face when they have preferences over packages but in traditional auctions are allowed to submit bids on individual items only.

CAs shift the computational burden from the bidders to the centre. There is an associated gain because the centre has all the information in hand to optimize while in traditional auctions the bidders only have estimated projected (probabilistic) information about how others will bid. Thus CAs yield more efficient allocations.

On the downside, the centre's task of determining the winners in a CA (deciding which bids to accept so as to maximize the sum of the accepted bids' prices subject to not selling any item to more than one bid) is a complex combinatorial optimization problem, even without exclusivity constraints among bids. Three main approaches have been studied for solving it.

1. *Optimal winner determination using some form of tree search*. For a review, see Sandholm (2006). The advantage is that the bidding language is not restricted and the optimal solution is found. The downside is that no optimal winner determination algorithm can run in polynomial time in the size of the problem instance in the worst case, because the problem is $\mathcal{NP}$-complete (Rothkopf, Pekěc and Harstad, 1998). ($\mathcal{NP}$-complete problems are problems for which the fastest known algorithms take exponential time in the size of the problem instance in the worst case. $\mathcal{P}$ is the class of easy problems solvable in polynomial time. The statement of winner determination not being solvable in polynomial time in the worst case relies on the usual assumption $\mathcal{P} \neq \mathcal{NP}$. This is an open question in complexity theory, but is widely believed to be true. If false, that would have sweeping implications throughout computer science.)

2. *Approximate winner determination.* The advantage is that many approximation algorithms run in polynomial time in the size of the instance even in the worst case. For reviews of such algorithms, see Sandholm (2002a) and Lehmann, Müller and Sandholm (2006). (Other suboptimal algorithms do not have such time guarantees, such as local search, stochastic local search, simulated annealing, genetic algorithms, and tabu search.) The downside is that the solution is sometimes far from optimal: no such algorithm can always find a solution that is within a factor

$$\min\left\{\#bids^{1-\varepsilon}, \#items^{\frac{1}{2}-\varepsilon}\right\} \qquad (1)$$

of optimal (Sandholm, 2002a). (This assumes $\mathcal{LPP} \neq \mathcal{NP}$. It is widely believed that these two complexity classes are indeed unequal.) For example, with just nine items for sale, no such algorithm can extract even 33 per cent of the available revenue from the bids in the worst case. With 81 items, that drops to 11 per cent.
3. *Restricting the bidding language* so much that optimal (within the restricted language) winner determination can be conducted in worst-case polynomial time. For a review, see Müller (2006). For example, if each package bid is only allowed to include at most two items, then winners can be determined in worst-case polynomial time (Rothkopf, Pekěc and Harstad, 1998). The downside is that bidders have to shoehorn their preferences into a restricted bidding language; this gives rise to similar problems as in non-combinatorial mechanisms for multi-item auctions: exposure problems, need to speculate how others will bid, inefficient allocation, and so on.

Truthful bidding can be made a dominant strategy by applying the *Vickrey–Clarke–Groves* (*VCG*) *mechanism* to a CA. Such incentive compatibility removes strategic complexity of the bidders. The mechanism works as follows. The optimal allocation is used, but the bidders do not pay their winning bids. Instead each bidder pays the amount of value he takes away from the others by taking some of the items. This value is measured as the difference between the others' winning bids' prices and what the others' winning bids' prices would have been had the agent not submitted any bids. This mechanism can be executed by determining the winners once overall, and once for each agent removed in turn. (This may be accomplishable with less computing. For example, in certain network auctions it can be done in the same asymptotic complexity as one winner determination – Hershberger and Suri, 2001.)

Very few canonical CAs have found their way to practice. However, auctions with richer bid expressiveness forms (that are more natural in the given application and more concise) and that support expressiveness also by the bid taker have made a major breakthrough into practice (Sandholm, 2007; Bichler et al., 2006). This is sometimes called *expressive commerce* to distinguish it from vanilla CAs. The widest area of application is currently industrial sourcing. Tens of billions of dollars worth of materials, transportation, and services are being sourced annually using such mechanisms, yielding billions of dollars in efficiency improvements. The bidders' expressiveness forms include different forms of flexible package bids, conditional discounts, discount schedules, side constraints (such as capacity constraints), and often hundreds of cost drivers (for example, fixed costs, variable costs, trans-shipment costs, and costs associated with changes). The item specifications can also be left partially open, and the bidders can specify some of the item attributes (delivery date, insurance terms, and so on.) in alternate ways. The bid taker also specifies preferences and constraints. Winner determination then not only decides who wins what, but also automatically configures the items. In some of these events it also configures the supply chain several levels deep as a side effect. On the high end, such an auction can have tens of thousands of items (multiple units of each), millions of bids, and hundreds of thousands of side constraints. Expressive mechanisms have also been designed for settings beyond auctions, such as combinatorial exchanges, charity donations, and settings with externalities.

Basically all of the fielded expressive auctions use the simple pay-your-winning-bids pricing rule. There are numerous important reasons why few, if any, use the VCG mechanism. It can lead to low revenue. It is vulnerable to collusion. Bidders would not tell the truth because they do not want to reveal their cost structures which the auctioneer could exploit the next time the auction is conducted, and so on (Sandholm, 2000; Rothkopf, 2007).

Basically all of the fielded expressive auctions use tree search for winner determination. In practice, modern tree search algorithms for the problem scale to the large and winners can be determined optimally. If winner determination were not done optimally in a CA, the VCG mechanism can lose its truth-dominance property (Sandholm, 2002b). In fact, any truthful suboptimal VCG-based mechanism for CAs is unreasonable in the sense that it sometimes does not allocate an item to a bidder even if he is the only bidder whose bids assign non-zero value to that item (Nisan and Ronen, 2000).

2.2 Algorithmic mechanism design

Motivated by the worry that some instances of *NP*-hard problems may not be solvable within reasonable time, a common research direction in theory of computing is approximation algorithms. They trade off solution quality for a guarantee that even in the worst case, the algorithm runs in polynomial time in the size of the input.

Analogously, Nisan and Ronen (2001) proposed *algorithmic mechanism design*: designing approximately optimal mechanisms that take the centre a polynomial

number of computing steps even in the worst case. However, this is more difficult than designing approximately optimal algorithms because the mechanism has to motivate the agents to tell the truth.

Lehmann, O'Callaghan and Shoham (2002) studied this for CAs with single-minded bidders (each bidder being only interested in one specific package of items). They present a fast greedy algorithm that guarantees a solution within a factor $\sqrt{\#items}$ of optimal. They show that the algorithm is not incentive compatible with VCG pricing, but is with their custom pricing scheme. They also identify sufficient conditions for any (approximate) mechanism to be incentive compatible (see also Kfir-Dahav, Monderer and Tennenholtz, 2000). There has been substantial follow-on work on subclasses of single-minded CAs.

Lavi and Swamy (2005) developed a technique for a range of packing problems with which any k-approximation algorithm (that is, algorithm that guarantees that the solution is within a factor k of optimal) that also bounds the integrality gap of the linear programming (LP) relaxation of the problem by k can be used to construct a k-approximation mechanism. The LP solution, scaled down by k, can be represented as a convex combination of integer solutions, and viewing this convex combination as specifying a probability distribution over integer solutions begets a VCG-based randomized mechanism that is truthful in expectation. For CAs with general valuations, this yields an $O(\sqrt{\#items})$-approximate mechanism.

In a different direction, several mechanisms have been proposed where the agents can help the centre find better outcomes. This is done either by giving the agents the information to do the centre's computing (Banks, Ledyard and Porter, 1989; Land, Powell and Steinberg, 2006; Parkes and Shneidman, 2004), or by allowing the agents to change what they told the mechanism based on the mechanism's output and potentially also based on what other agents told the mechanism (Nisan and Ronen, 2000). In VCG-based mechanisms, an agent benefits from lying only if the lie causes the mechanism to find an outcome that is better overall.

2.3 Automated mechanism design

Conitzer and Sandholm (2002) proposed the idea of *automated mechanism design*: having a computer, rather than a human, design the mechanism. Because human effort is eliminated, this enables custom design of mechanisms for every setting. The setting can be described by the agents' (discretized) type spaces, the designer's prior over types, the desired notion of incentive compatibility (for example, dominant strategies vs. Bayes-Nash implementation), the desired notion of participation constraints (for example, *ex interim*, *ex post*, or none), whether payments are allowed, and whether the mechanism is allowed to use randomization.) This can yield better mechanisms for previously studied settings because the mechanism is designed for the specific setting rather than a class of settings. It can also be used for settings not previously studied in mechanism design.

For almost all natural (linear) objectives, all variants of the design problem are $\mathcal{NP}$-complete if the mechanism is not allowed to use randomization, but randomized mechanisms can be constructed for all these settings in polynomial time using linear programming. Custom algorithms have been developed for some problems in each of these two categories. (Even the latter category warrants research. While the linear programme is polynomial in the size of the input, the input itself can be exponential in the number of agents.) Structured representations of the problem can also make the design process drastically faster.

Beyond the general setting, automated mechanism design has been applied to specific settings, such as creating revenue-maximizing CAs (without the need to discretize types) (Likhodedov and Sandholm, 2005) (a recognized problem that eludes analytical characterization; even the two-item case is open), reputation systems (Jurca and Faltings, 2006), safe exchange mechanisms (Sandholm and Ferrandon, 2000), and supply chain settings (Vorobeychik, Kiekintveld and Wellman, 2006). Automated mechanism design software has recently also been adopted by several mechanism design theoreticians to speed up their research.

It turns out that even *multistage mechanisms* can be designed automatically (Sandholm, Conitzer and Boutilier, 2007). Furthermore, automated mechanism design has been applied to the design of *online mechanisms* (Hajiaghayi, Kleinberg and Sandholm, 2007), that is, mechanisms that execute while the world changes – for example, agents enter and exit the system.

3 Computing by the agents

I will now move to discussing computing by the agents.

3.1 Mechanisms that are hard to manipulate

This section demonstrates that one can use the fact that agents are computationally limited to achieve things that are not achievable via any mechanism among perfectly rational agents.

A seminal negative result, the *Gibbard–Satterthwaite theorem*, states that if there are three or more candidates, then in any non-dictatorial voting scheme there are candidate rankings of the other voters, and preferences of the agent, under which the agent is better off voting manipulatively than truthfully. One avenue around this impossibility is to construct desirable general non-dictatorial voting protocols under which *finding* a beneficial manipulation is prohibitively hard computationally.

There are two natural alternative goals of manipulation. In *constructive manipulation*, the manipulator tries to find an order of candidates that he can reveal so that his favourite candidate wins. In *destructive manipulation*,

the manipulator tries to find an order of candidates that he can reveal so that his hated candidate does not win. These are special cases of the utility-theoretic notion of improving one's utility, so the hardness results, discussed below, carry over to the usual utility-theoretic setting.

Unfortunately, finding a constructive manipulation is easy (in $\mathcal{P}$) for the *plurality*, *Borda*, and *maximin* voting rules (Bartholdi, Tovey and Trick, 1989), which are commonly used. On the bright side, constructive manipulation of the *single transferable vote* (*STV*) protocol is $\mathcal{NP}$-hard (Bartholdi and Orlin, 1991) (as is manipulation of the *second order Copeland* protocol (Bartholdi, Tovey and Trick, 1989), but that hardness is driven solely by the tie-breaking rule). Even better, there is a systematic methodology for slightly tweaking voting protocols that are easy to manipulate, so that they become hard to manipulate (Conitzer and Sandholm, 2003). Specifically, before the original protocol is executed, one pairwise elimination round is executed among the candidates, and only the winning candidates survive to the original protocol. This makes the protocols $\mathcal{NP}$-hard, $\#\mathcal{P}$-hard ($\#\mathcal{P}$-hard problems are at least as hard as counting the number of solutions to a problem in $\mathcal{P}$), or even $\mathcal{PSPACE}$-hard ($\mathcal{PSPACE}$-hard problems are at least as hard as any problem that can be solved using a polynomial amount of memory) to manipulate constructively, depending on whether the schedule of the pre-round is determined before the votes are collected, randomly after the votes are collected, or the scheduling and the vote collecting are carefully interleaved, respectively.

All of the hardness results of the previous paragraph rely on both the number of voters and the number of candidates growing. The number of candidates can be large in some domains, for example when voting over task or resource allocations. However, in other elections – such as presidential elections – the number of candidates is small. If the number of candidates is a constant, both constructive and destructive manipulation are easy (in $\mathcal{P}$), regardless of the number of voters (Conitzer, Sandholm and Lang, 2007). This holds even if the voters are weighted, or if a coalition of voters tries to manipulate. On the bright side, when a coalition of weighted voters tries to manipulate, complexity can arise even for a constant number of candidates, see Tables 1 and 2. Another lesson from that table is that randomizing over instantiations of the mechanism (such as schedules of a *cup*) can be used to make manipulation hard.

As usual in computer science, all the results mentioned above are worst-case hardness. Unfortunately, under weak assumptions on the preference distribution and voting rule, most instances of any voting rule are easy to manipulate (Conitzer and Sandholm, 2006).

All of the hardness results discussed above hold even if the manipulators know the non-manipulators' votes exactly. Under weak assumptions, if weighted coalitional manipulation with complete information about the others' votes is hard in some voting protocol, then individual unweighted manipulation is hard when there is uncertainty about the others' votes (Conitzer, Sandholm and Lang, 2007).

3.2 Non-truth-promoting mechanisms

A challenging issue is that even if it is prohibitively hard to find a beneficial manipulation, the agents might not tell the truth. For example, an agent might take a chance that he will do better with a lie. The following result

Table 2 *Complexity of destructive weighted coalitional manipulation*

Number of candidates:	2	≥3
STV	$\mathcal{P}$	$\mathcal{NP}$-complete
Plurality with runoff	$\mathcal{P}$	$\mathcal{NP}$-complete
Randomized cup	$\mathcal{P}$	?
Borda	$\mathcal{P}$	$\mathcal{P}$
Veto	$\mathcal{P}$	$\mathcal{P}$
Copeland	$\mathcal{P}$	$\mathcal{P}$
Maximin	$\mathcal{P}$	$\mathcal{P}$
Cup	$\mathcal{P}$	$\mathcal{P}$
Plurality	$\mathcal{P}$	$\mathcal{P}$

Source: Conitzer, Sandholm and Lang (2007)

Table 1 *Complexity of constructive weighted coalitional manipulation*

Number of candidates:	2	3	4, 5, 6	≥7
Borda	$\mathcal{P}$	$\mathcal{NP}$-complete	$\mathcal{NP}$-complete	$\mathcal{NP}$-complete
Veto	$\mathcal{P}$	$\mathcal{NP}$-complete	$\mathcal{NP}$-complete	$\mathcal{NP}$-complete
STV	$\mathcal{P}$	$\mathcal{NP}$-complete	$\mathcal{NP}$-complete	$\mathcal{NP}$-complete
Plurality with runoff	$\mathcal{P}$	$\mathcal{NP}$-complete	$\mathcal{NP}$-complete	$\mathcal{NP}$-complete
Copeland	$\mathcal{P}$	$\mathcal{P}$	$\mathcal{NP}$-complete	$\mathcal{NP}$-complete
Maximin	$\mathcal{P}$	$\mathcal{P}$	$\mathcal{NP}$-complete	$\mathcal{NP}$-complete
Randomized cup	$\mathcal{P}$	$\mathcal{P}$	$\mathcal{P}$	$\mathcal{NP}$-complete
Cup	$\mathcal{P}$	$\mathcal{P}$	$\mathcal{P}$	$\mathcal{P}$
Plurality	$\mathcal{P}$	$\mathcal{P}$	$\mathcal{P}$	$\mathcal{P}$

shows that, nevertheless, mechanism design can be improved by making the agents face complexity. (This is one reason why computational issues can render the *revelation principle* inapplicable. One of the things the principle says is that for any non-truth-promoting mechanism it is possible to construct an incentive-compatible mechanism that is at least as good. The theorem below challenges this.)

Theorem 1 (Conitzer and Sandholm, 2004) *Suppose the center is trying to maximize social welfare, and neither payments nor randomization is allowed. Then, even with just two agents (one of whom does not even report a type, so dominant strategy implementation and Bayes-Nash implementation coincide), there exists a family of preference aggregation settings such that:*

- *the execution of any optimal incentive-compatible mechanism is $\mathcal{NP}$-complete for the center, and*
- *there exists a non-incentive-compatible mechanism which (1) requires the centre to carry out only polynomial computation, and (2) makes finding any beneficial insincere revelation $\mathcal{NP}$-complete for the type-reporting agent. Additionally, if the type-reporting agent manages to find a beneficial insincere revelation, or no beneficial insincere revelation exists, the social welfare of the outcome is identical to the social welfare that would be produced by any optimal incentive-compatible mechanism. Finally, if the type-reporting agent does not manage to find a beneficial insincere revelation where one exists, the* ***social welfare of the outcome is strictly greater than the social welfare that would be produced by any optimal incentive-compatible mechanism****.*

An analogous theorem holds if, instead of counting computational steps, we count calls to a commonly accessible oracle which, when supplied with an agent, that agent's type, and an outcome, returns a utility value for that agent.

3.3 Preference (valuation) determination via computing or information acquisition

In many (auction) settings, even determining one's valuation for an item (or a bundle of items) is complex. For example when bidding for trucking lanes (tasks), this involves solving two $\mathcal{NP}$-complete local planning problems: the vehicle routing problem with the new lanes of the bundle and the problem without them (Sandholm, 1993). The difference in the costs of those two local plans is the cost (valuation) of taking on the new lanes.

In these types of settings, the *revelation principle* applies only in a trivial way: the agents report their data and optimization models to the centre, and the centre does the computation for them. It stands to reason that in many applications the centre would not want to take on that burden, in which case such extreme direct mechanisms are not an option. Therefore, I will now focus on mechanisms where the agents report valuations to the centre, as in traditional auctions.

Bidders usually have limited computing and time, so they cannot exactly evaluate all (or even any) bundles – at least not without cost. This leads to a host of interesting issues where computing and incentives are intimately intertwined.

For example, in a one-object auction, should a bidder evaluate the object if there is a cost to doing so? It turns out that the Vickrey auction loses its dominant-strategy property: whether or not the bidder should pay the evaluation cost depends on the other bidders' valuations (Sandholm, 2000).

If a bidder has the opportunity to *approximate his valuation to different degrees*, how much computing time should the bidder spend on refining its valuation? If there are multiple items for sale, how much computing time should the bidder allocate on different bundles? A bidder may even allocate some computing time to evaluate other bidders' valuations so as to be able to bid more strategically; this is called *strategic computing*.

To answer these questions, Larson and Sandholm (2001) developed a deliberation control method called a *performance profile tree* for projecting how an anytime algorithm (that is, an algorithm that has an answer available at any time, but where the quality of the answer improves the more computing time is allocated to the algorithm) will change the valuation if additional computing is allocated toward refining (or improving) it. This deliberation control method applies to any anytime algorithm. Unlike earlier deliberation control methods for anytime algorithms, the performance profile tree is a *fully normative model of bounded rationality*: it takes into account all the information that an agent can use to make its deliberation control decisions. This is necessary in the game-theoretic context; otherwise a strategic agent could take into account some information that the model does not.

Using this deliberation control method, the auction can be modelled as a game where the agents' strategy spaces include computing actions. At every point, each agent can decide on which bundle to allocate its next step of computing as a function of the agent's computing results so far (and in open-cry auction format also the others' bids observed so far). At every point, the agent can also decide to submit bids. One can then solve this for equilibrium: each agent's (deliberation and bidding) strategy is a best-response to the others' strategies. This is called *deliberation equilibrium*.

This notion, and the performance profile tree, apply not only to computational actions but also to information gathering actions for determining valuations. (In contrast, most of the literature on information acquisition in auctions does not take into account that valuations can be determined to different degrees and that an

Table 3 *Can strategic computing occur in deliberation equilibrium? The most interesting results are in bold. As a benchmark from classical auction theory, the table also shows whether or not perfectly rational agents, that can determine their valuations instantly without cost, would benefit from considering each others' valuations when deciding how to bid*

	Auction mechanism	Speculation by perfectly rational agents?	Strategic computing?	
			Limited computing	**Costly computing**
Single item	First price	yes	yes	yes
	Dutch	yes	yes	yes
	English	no	no	yes
	Vickrey	no	no	yes
Multiple items	First price	yes	yes	yes
	VCG	no	yes	yes

agent may want to invest effort to determine others' valuations as well – even in private-value settings.)

Table 3 shows in which settings strategic computing can and cannot occur in deliberation equilibrium. This depends on the auction mechanism. Interestingly, it also depends on whether the agent has limited computing (for example, owning a desktop computer that the agent can use until the auction's deadline) or costly computing (for example, being able to buy any amount of supercomputer time where each cycle comes at a cost).

The notion of deliberation equilibrium can also be used as the basis for designing new mechanisms which hopefully work well among agents whose computing is costly or limited. Unfortunately, there is an impossibility (Larson and Sandholm, 2005): there exists no mechanism that is *sensitive* (the outcome is affected by each agent's strategy), *preference formation independent* (does not do the computations for the agents; the agents report valuations), *non-misleading* (no agent acts in a way that causes others to believe his true type has zero probability), and *deliberation-proof* (no strategic computing occurs in equilibrium, that is, agents compute only on their own problems). Current work involves designing mechanisms that take part in preference formation in limited ways: for example, agents report their performance profile trees to the centre, which then coordinates the deliberations incrementally as agents report deliberation results. Current research also includes designing mechanisms where strategic computing occurs but its wastefulness is limited.

3.3.1 Preference elicitation by the centre

To reduce the agents' preference determination effort, Conen and Sandholm (2001) proposed a framework where the centre (also known as *elicitor*) explicitly elicits preference information from the agents incrementally on an as-needed basis by posing queries to the agents. The centre thereby builds a model of the agents' preferences, and decides what to ask, and from which agent, based on this model. Usually the process can be terminated with the provably correct outcome while requiring only a small portion of the agents' preferences to be determined. Multistage mechanisms can yield up to exponential savings in preference determination and communication effort the agents need to go through compared to single-stage mechanisms (Conitzer and Sandholm, 2004).

The explicit preference elicitation framework was originally proposed for CAs (but the approach has since been used for other settings as well, such as voting). For general valuations, an exponential number of bits in the number of items for sale has to be communicated in the worst case no matter what queries are used (Nisan and Segal, 2006). However, experimentally only a small fraction of the preference information needs to be elicited before the provably optimal solution is found. Furthermore, for valuations that have certain types of structure, even the worst-case number of queries needed is small. Research has also been done on the relative power of different query types.

If enough information is elicited to also determine the VCG payments, and these are the payments charged to the bidders, answering the elicitor's queries truthfully is an *ex post* equilibrium (a strengthening of Nash equilibrium that does not rely on priors). (This assumes there is no explicit cost or limit to valuation determination; mechanisms have also been designed for settings where there is an explicit cost. (Larson, 2006)) This holds even if the agents are allowed to answer queries that the elicitor did not ask (for example, queries that are easy for the agent to answer and which the agent thinks will significantly advance the elicitation process). We thus have a *pull–push mechanism* where both the centre and the agents guide the preference revelation (and thus also the preference determination/refinement by the agents). For a review, see Sandholm and Boutilier (2006). Ascending (combinatorial) auctions are an earlier special case, and have limited power compared to the general framework (Blumrosen and Nisan, 2005).

Preference elicitation can sometimes be computationally complex for the centre. It can be complex to intelligently decide what to ask next, and from whom. It can

also be complex to determine whether enough information has been elicited to determine the optimal outcome. Even if the elicitor knows that enough has been elicited, it can be complex to determine the outcome – for example, allocation of items to bidders in some CAs.

3.4 Distributed (centre-free) mechanisms

Computer scientists often have a preference for distributed applications that do not have any centralized coordination point (centre). Depending on the application, the reasons for this preference may include avoiding a single vulnerable point of failure, distributing the computing effort (for computational efficiency or because the data is inherently distributed), and enhancing privacy. The preference carries over from traditional computer science applications to different forms of negotiation systems – for example, see Sandholm (1993) for an early distributed automated negotiation system for software agents.

Feigenbaum et al. (2005) have studied lowest-cost inter-domain routing on the Internet, modifying a distributed protocol so that the agents (routing domains) are motivated to report their true costs and the solution is found with minimal message passing. For a review of some other research topics in this space, see Feigenbaum and Shenker (2002).

One can go further by taking into account the fact that agents might not choose to follow the prescribed protocol. They may cheat not only on information-revelation actions, but also on message-passing and computational actions. Despite computation actions not being observable by others, an agent can be motivated to compute as prescribed by tasking at least one other agent with the same computation, and comparing the results (Sandholm et al., 1999). Careful problem partitioning can also be used to achieve the same outcome without redundancy by only requiring agents to perform computing and message passing tasks that are in their own interest (Parkes and Shneidman, 2004). Shneidman and Parkes (2004) propose a general proof technique and instantiate it to provide a non-manipulable protocol for inter-domain routing. Monderer and Tennenholtz (1999) develop protocols for one-item auctions executed among agents on a communication network. The protocols motivate the agents to correctly reveal preferences and communicate. For the setting where agents with private utility functions have to agree on variable assignments subject to side constraints (for example, meeting scheduling), Petcu, Faltings and Parkes (2006) developed a VCG-based distributed optimization protocol that finds the social welfare maximizing allocation and each agent is motivated to follow the protocol in terms of all three types of action. The only centralized party needed is a bank that can extract payments from the agents.

Cryptography is a powerful tool for achieving privacy when trying to execute a mechanism in a distributed way without a centre, using private communication channels among the agents. Consider first the setting with passive adversaries, that is, agents that faithfully execute the specified distributed communication protocol, but who try to infer (at least something about) some agents' private information.

- If agents are computationally limited – for example, they are assumed to be unable to factor large numbers – then arbitrary functions can be computed while guaranteeing that each agent maintains his privacy (except, of course, to the extent that the answer of the computation says something about the inputs) (Goldreich, Micali and Wigderson, 1987). Thus the desire for privacy does not constrain what social choice functions can be implemented.
- In contrast, only very limited social choice functions can be computed privately among computationally unlimited agents. For example, when there are just two alternatives, every monotonic, non-dictatorial social choice function that can be privately computed is constant (Brandt and Sandholm, 2005). With special structure in the preferences, this impossibility can sometimes be avoided. For example, with the standard model of quasi-linear utility, first-price auctions can be implemented privately; second-price (Vickrey) auctions with more than two bidders cannot (Brandt and Sandholm, 2004).

A more general model is that of active adversaries who can execute the distributed communication protocol unfaithfully in a coordinated way. A more game-theoretic model is that of rational adversaries that are not passive, but not malicious either. For a brief overview of such work, see COMPUTER SCIENCE AND GAME THEORY.

TUOMAS SANDHOLM

This work was funded by the National Science Foundation under ITR grant IIS-0427858, and a Sloan Foundation Fellowship. I thank Felix Brandt, Christina Fong, JoeHalpern, and David Parkes for helpful comments.

Bibliography

Banks, J.S., Ledyard, J. and Porter, D. 1989. Allocating uncertain and unresponsive resources: an experimental approach. *RAND Journal of Economics* 20, 1–25.

Bartholdi, J., III and Orlin, J. 1991. Single transferable vote resists strategic voting. *Social Choice and Welfare* 8, 341–54.

Bartholdi, J., III, Tovey, C. and Trick, M. 1989. The computational difficulty of manipulating an election. *Social Choice and Welfare* 6, 227–41.

Bichler, M., Davenport, A., Hohner, G. and Kalagnanam, J. 2006. Industrial procurement auctions. In Cramton, Shoham and Steinberg (2006).

Blumrosen, L. and Nisan, N. 2005. On the computational power of iterative auctions. *Proceedings of the ACM Conference on Electronic Commerce*, 29–43.

Brandt, F. and Sandholm, T. 2004. (Im)possibility of unconditionally privacy-preserving auctions. *Proceedings of the International Conference on Autonomous Agents and Multi-Agent Systems*, 810–17.

Brandt, F. and Sandholm, T. 2005. Unconditional privacy in social choice. *Proceedings of the Conference on Theoretical Aspects of Rationality and Knowledge*, 207–18.

Conen, W. and Sandholm, T. 2001. Preference elicitation in combinatorial auctions: extended abstract. *Proceedings of the ACM Conference on Electronic Commerce*, 256–9. More detailed description of algorithmic aspects in *Proceedings of the IJCAI-01 Workshop on Economic Agents, Models, and Mechanisms*, 71–80.

Conitzer, V. and Sandholm, T. 2002. Complexity of mechanism design. *Proceedings of the Conference on Uncertainty in Artificial Intelligence*, 103–10.

Conitzer, V. and Sandholm, T. 2003. Universal voting protocol tweaks to make manipulation hard. *Proceedings of the International Joint Conference on Artificial Intelligence*, 781–8.

Conitzer, V. and Sandholm, T. 2004. Computational criticisms of the revelation principle. *Conference on Logic and the Foundations of Game and Decision Theory*. Earlier versions: AMEC-03, EC-04.

Conitzer, V. and Sandholm, T. 2006. Nonexistence of voting rules that are usually hard to manipulate. *Proceedings of the National Conference on Artificial Intelligence*.

Conitzer, V., Sandholm, T. and Lang, J. 2007. When are elections with few candidates hard to manipulate? *Journal of the ACM* 54(3), Article 14.

Cramton, P., Shoham, Y. and Steinberg, R. (eds.) 2006. *Combinatorial Auctions*. Cambridge, MA: MIT Press.

Feigenbaum, J., Papadimitriou, C., Sami, R. and Shenker, S. 2005. A BGP-based mechanism for lowest cost routing. *Distributed Computing* 18, 61–72.

Feigenbaum, J. and Shenker, S. 2002. Distributed algorithmic mechanism design: recent results and future directions. *Proceedings of the International Workshop on Discrete Algorithms and Methods for Mobile Computing and Communications*, 1–13.

Goldreich, O., Micali, S. and Wigderson, A. 1987. How to play any mental game or a completeness theorem for protocols with honest majority. *Proceedings of the Symposium on Theory of Computing*, 218–29.

Hajiaghayi, M.T., Kleinberg, R. and Sandholm, T. 2007. Automated online mechanism design and prophet inequalities. *Proceedings of the National Conference on Artificial Intelligence*.

Hershberger, J. and Suri, S. 2001. Vickrey prices and shortest paths: what is an edge worth? *Proceedings of the Symposium on Foundations of Computer Science*, 252–9.

Jurca, R. and Faltings, B. 2006. Minimum payments that reward honest reputation feedback. *Proceedings of the ACM Conference on Electronic Commerce*, 190–9.

Kfir-Dahav, N., Monderer, D. and Tennenholtz, M. 2000. Mechanism design for resource bounded agents. *Proceedings of the International Conference on Multi-Agent Systems*, 309–15.

Land, A., Powell, S. and Steinberg, R. 2006. PAUSE: a computationally tractable combinatorial auction. In Cramton, Shoham and Steinberg (2006).

Larson, K. 2006. Reducing costly information acquisition in auctions. *Proceedings of the Autonomous Agents and Multi-Agent Systems*, 1167–74.

Larson, K. and Sandholm, T. 2001. Costly valuation computation in auctions. *Proceedings of the Theoretical Aspects of Rationality and Knowledge*, 169–182.

Larson, K. and Sandholm, T. 2005. Mechanism design and deliberative agents. *Proceedings of the International Conference on Autonomous Agents and Multi-Agent Systems*, 650–6.

Lavi, R. and Swamy, C. 2005. Truthful and near-optimal mechanism design via linear programming. *Proceedings of the Symposium on Foundations of Computer Science*, 595–604.

Lehmann, D., Müller, R. and Sandholm, T. 2006. The winner determination problem. In Cramton, Shoham and Steinberg (2006).

Lehmann, D., O'Callaghan, L.I. and Shoham, Y. 2002. Truth revelation in rapid, approximately efficient combinatorial auctions. *Journal of the ACM* 49, 577–602.

Likhodedov, A. and Sandholm, T. 2005. Approximating revenue-maximizing combinatorial auctions. *Proceedings of the National Conference on Artificial Intelligence*, 267–74.

Monderer, D. and Tennenholtz, M. 1999. Distributed games: from mechanisms to protocols. *Proceedings of the National Conference on Artificial Intelligence*, 32–7.

Müller, R. 2006. Tractable cases of the winner determination problem. In Cramton, Shoham and Steinberg (2006).

Nisan, N. and Ronen, A. 2000. Computationally feasible VCG mechanisms. *Proceedings of the ACM Conference on Electronic Commerce*, 242–52.

Nisan, N. and Ronen, A. 2001. Algorithmic mechanism design. *Games and Economic Behavior* 35, 166–96.

Nisan, N. and Segal, I. 2006. The communication requirements of efficient allocations and supporting prices. *Journal of Economic Theory* 129, 192–224.

Parkes, D. and Shneidman, J. 2004. Distributed implementations of generalized Vickrey–Clarke–Groves auctions. *Proceedings of the International Conference on Autonomous Agents and Multi-Agent Systems*, 261–8.

Petcu, A., Faltings, B. and Parkes, D. 2006. MDPOP: faithful distributed implementation of efficient social choice problems. *Proceedings of the International Conference on Autonomous Agents and Multi-Agent Systems*, 1397–404.

Rothkopf, M. 2007. Thirteen reasons why the Vickrey–Clarke–Groves process is not practical. *Operations Research* 55, 191–7.

Rothkopf, M., Pekěc, A. and Harstad, R. 1998. Computationally manageable combinatorial auctions. *Management Science* 44, 1131–47.

Sandholm, T. 1993. An implementation of the contract net protocol based on marginal cost calculations. *Proceedings of the National Conference on Artificial Intelligence*, 256–62.

Sandholm, T. 2000. Issues in computational Vickrey auctions. *International Journal of Electronic Commerce* 4, 107–29. Early version in ICMAS-96.

Sandholm, T. 2002a. Algorithm for optimal winner determination in combinatorial auctions. *Artificial Intelligence* 135, 1–54. Earlier versions: ICE-98 keynote, Washington U. tech report WUCS-99-01 Jan. 1999, IJCAI-99.

Sandholm, T. 2002b. eMediator: a next generation electronic commerce server. *Computational Intelligence* 18, 656–76. Earlier versions: Washington U. tech report WU-CS-99-02 Jan. 1999, AAAI-99 Workshop on AI in Ecommerce, AGENTS-00.

Sandholm, T. 2006. Optimal winner determination algorithms. In Cramton, Shoham and Steinberg (2006).

Sandholm, T. 2007. Expressive commerce and its application to sourcing: how we conducted $35 billion of generalized combinatorial auctions. *AI Magazine* 28(3), 45–58.

Sandholm, T. and Boutilier, C. 2006. Preference elicitation in combinatorial auctions. In Cramton, Shoham and Steinberg (2006).

Sandholm, T., Conitzer, V. and Boutilier, C. 2007. Automated design of multistage mechanisms. *Proceedings of the International Joint Conference on Artificial Intelligence*, 1500–6.

Sandholm, T. and Ferrandon, V. 2000. Safe exchange planner. *Proceedings of the International Conference on Multi-Agent Systems*, 255–62.

Sandholm, T., Larson, K., Andersson, M., Shehory, O. et al. 1999. Coalition structure generation with worst case guarantees. *Artificial Intelligence* 111, 209–38.

Shneidman, J. and Parkes, D.C. 2004. Specification faithfulness in networks with rational nodes. *Proceedings of the ACM Symposium on Principles of Distributed Computing*, 88–97.

Vorobeychik, Y., Kiekintveld, C. and Wellman, M. 2006. Empirical mechanism design: methods, with application to a supply chain scenario. *Proceedings of the ACM Conference on Electronic Commerce*, 306–15.