Computational Social Choice

Ulle Endriss
Institute for Logic, Language and Computation
University of Amsterdam

Computational Social Choice

Social choice theory studies mechanisms for *collective decision making*, such as voting procedures or fair division protocols.

Computational social choice adds a computational perspective to this, and also explores the use of concepts from social choice in computing.

This Talk

Three examples showing that computer science can offer a useful new perspective on problems in collective decision making:

- Computational Barriers against Manipulation in Voting
- Compact Representation of Preferences in Combinatorial Domains
- Computing Fair and Efficient Allocations of Goods to Agents

Problem: Vote Manipulation

Suppose the *plurality rule* (as in most real-world situations) is used to decide the outcome of an election: the candidate receiving the highest number of votes wins.

Assume the preferences of the people in, say, Florida are as follows:

$$
\begin{array}{ll}
49\%: & \text{Bush} \succ \text{Gore} \succ \text{Nader} \\
20\%: & \text{Gore} \succ \text{Nader} \succ \text{Bush} \\
20\%: & \text{Gore} \succ \text{Bush} \succ \text{Nader} \\
11\%: & \text{Nader} \succ \text{Gore} \succ \text{Bush}
\end{array}
$$

So even if nobody is cheating, Bush will win in a plurality contest.

Issue: In a pairwise competition, Gore would have defeated anyone.

Issue II: It would have been in the interest of the Nader supporters to *manipulate*, *i.e.* to misrepresent their preferences (and vote for Gore).

Approach: Make Manipulation Intractable

By the Gibbard-Satterthwaite Theorem, *any* voting rule for choosing between ≥ 3 candidates can be manipulated (unless it is dictatorial).

Idea: So it's always *possible* to manipulate, but maybe it's *difficult*! Tools from *complexity theory* can be used to make this idea precise.

- For the *plurality rule* this does *not* work: if I know all other ballots and want c to win, it is *easy* to compute my best strategy.
- But for *single transferable vote* it does work. Bartholdi and Orlin showed that manipulation of STV is *NP-complete*.

A. Gibbard. Manipulation of Voting Schemes: A General Result. *Econometrica*, 41(4):587–601, 1973.

M.A. Satterthwaite. Strategy-proofness and Arrow's Conditions. *Journal of Economic Theory*, 10:187–217, 1975.

J.J. Bartholdi III and J.B. Orlin. Single Transferable Vote Resists Strategic Voting. *Social Choice and Welfare*, 8(4):341–354, 1991.

Problem: Huge Numbers of Alternatives

The alternatives often have a *combinatorial structure:* they are characterised by a tuple of variables ranging over a finite domain.

- allocate n indivisible goods to m agents: m^n alternatives
- elect a committee of size k, from n candidates: $\binom{n}{k}$ alternatives

Just *representing* and communicating the *preferences* of the agents can become a non-trivial problem (and that's not the only problem).

People in AI have long worked on *knowledge representation*, so there is a lot of expertise in this community . . .

Approach: Compact Representation of Preferences

We need languages that can represent preferences in a compact way. One type of language are so-called *weighted propositional formulas*. Utility is computed as the sum of the weights of the formulas satisfied.

Example: $\{(a, 3), (b \vee c \vee d, 4), (b \wedge \neg c, 2)\}$ defines this utility function:

$$
\begin{aligned}
u(\emptyset) &= 0 \\
u(a) &= 3 \\
u(b) &= 6 \\
u(c) &= 4 \\
u(d) &= 4
\end{aligned}
\qquad
\begin{aligned}
u(ab) &= 9 \\
u(ac) &= 7 \\
u(ad) &= 7 \\
u(bc) &= 4 \\
u(bd) &= 6 \\
u(cd) &= 4
\end{aligned}
\qquad
\begin{aligned}
u(abc) &= 7 \\
u(abd) &= 9 \\
u(acd) &= 7 \\
u(bcd) &= 4 \\
u(abcd) &= 7
\end{aligned}
$$

Questions: *Expressivity*, relative *succinctness*, computational *complexity*?; how to use this for preference *aggregation*?; ...

J. Uckelman and U. Endriss. *Preference Representation with Weighted Goals: Expressivity, Succinctness, Complexity*. Proc. AiPref-2007.

Problem: Finding Socially Optimal Allocations

Scenario: Group of *agents* and set of indivisible *goods*. Agents have *preferences* over bundles of goods. What would be a good *allocation*?

Welfare economics and social choice theory give various definitions for what is *socially optimal* (e.g., utilitarianism *vs*. egalitarianism).

Problem: How do we find (compute) a socially optimal allocation?

Solution I: Computer scientists have developed powerful algorithms for computing socially optimal solutions in a *centralised* manner (integer programming, constraint satisfaction, heuristic search techniques).

Also interesting are *distributed* approaches . . .

Approach: Convergence in Distributed Negotiation

We have studied several variations of the following model:

- Preferences: agents have arbitrary quasi-linear utility functions.
- Agents will accept all deals that benefit them (and only those).
- No structural restrictions on deals (≥ 2 agents possible etc).
- Side payments are possible and agents have "enough" money.

Then a known result states that *any sequence of deals* will eventually converge to an allocation that has *maximal utilitarian social welfare*.

Questions: What structural restrictions on *deals* work for which types of *preferences*?; other notions of social optimality (*fairness*)?; how many deals before termination (*communication complexity*)?

U. Endriss, N. Maudet, F. Sadri, and F. Toni. Negotiating Socially Optimal Allocations of Resources. *Journal of Artif. Intelligence Research*, 25:315–348, 2006.

Conclusion

- Computational social choice combines ideas from mathematical economics and computer science in new and fruitful ways.
- For further information, have a look at our "*Short Introduction to Computational Social Choice*".
- At the ILLC (Plantage Muidergracht 24), we have a seminar on Computational Social Choice with talks once or twice a month:

  `http://www.illc.uva.nl/~ulle/seminar/`

Y. Chevaleyre, U. Endriss, J. Lang, and N. Maudet. *A Short Introduction to Computational Social Choice*. Proc. SOFSEM-2007.