Collective Decision Making in Combinatorial Domains

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 protocols for fair division.

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

- Part of wider trend of interdisciplinary research at the interface of *mathematical economics* (social choice, game and decision theory) and *computer science* (and artificial intelligence and logic);
- with an active research community, witness e.g. the COMSOC workshops in Amsterdam (2006) and Liverpool (2008).

I will first give *three examples* of research in COMSOC, and then get into the main topic for today (another such example) . . .

Complexity as a Barrier against Manipulation

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 X 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*.

Recent work in COMSOC has expanded on this idea:

- NP is a worst-case notion. What about average complexity?
- Also: complexity of winner determination, control, bribery ...

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

Logical Modelling of Social Choice Mechanisms

Logic has long been used to *formally specify* computer systems, facilitating formal or even *automatic verification* of various properties. Can we apply this methodology also to social choice mechanisms?

Recall *Arrow's Theorem:* any preference aggregation mechanism for ≥ 3 alternatives that satisfies *unanimity* and *IIA* must be *dictatorial*.

Recent work by Lin and Tang shows that the "base case" of 2 agents and 3 alternatives can be fully modelled in propositional logic:

- Automated theorem provers can verify $\textsc{Arrow}(2, 3)$ to be correct in < 1 second — that's $(3!)^{3! \times 3!} \approx 10^{28}$ SWFs to check
- Opens up opportunities for quick sanity checks of hypotheses regarding new possibility and impossibility theorems.

F. Lin and P. Tang. *Computer-aided Proofs of Arrow's and other Impossibility Theorems*. Proc. AAAI-2008.

Multiagent Systems and Fair Division

Multiagent Systems is an important research area in Computer Science:

- A multiagent system is a system consisting of several autonomous software agents that interact with each other to further their own interests (competition) or in pursuit of a joint goal (cooperation).
- Classical Artificial Intelligence addresses (specific aspects of) single agents, MAS focusses on the interaction between different agents.

Problems studied include *multiagent resource allocation*. Social Choice Theory and Welfare Economics have contributed new ideas:

- Look into alternative definitions of what makes a "good" allocation: Rawlsian egalitarianism, envy-freeness, ...
- Now "computational" treatment of these issues: algorithms, formal analysis of protocols, complexity studies, ...

U. Endriss, N. Maudet, F. Sadri, and F. Toni. Negotiating Socially Optimal Allocations of Resources. *Journal of Artificial Intelligence Research*, 2006.

Talk Overview: Part II

Now for the main topic of the talk:

Collective Decision Making in Combinatorial Domains

We will cover:

- What is the problem?
- Languages for compactly modelling preferences
- Examples for technical results
- Applications to collective decision making
- Conclusion

Combinatorial Domains

Many social choice problems have a *combinatorial structure:*

- Elect a *committee* of k members from amongst n candidates.
- During a *referendum* (in Switzerland, California, places like that), voters may be asked to vote on n different propositions.
- Find a good *allocation* of n indivisible goods to agents.

Seemingly small problems generate huge numbers of alternatives:

- Number of 3-member committees from 10 candidates: $\binom{10}{3} = 120$ (i.e. $120! \approx 6.7 \times 10^{198}$ possible rankings)
- Allocating 10 goods to 5 agents: $5^{10} = 9765625$ allocations and $2^{10} = 1024$ bundles for each agent to think about

We need good *languages* for representing preferences!

Preference Representation Languages

The following are relevant questions to consider when we have to choose a preference representation language:

- *Cognitive relevance:* How close is a given language to the way in which humans would express their preferences?
- *Elicitation friendliness:* How difficult is it to elicit the preferences of an agent so as to represent them in the chosen language?
- *Expressive power:* Can the chosen language encode all the preference structures we are interested in?
- *Succinctness:* Is the representation of (typical) structures succinct? Is one language more succinct than the other?
- *Complexity:* What is the computational complexity of related problems, such as comparing two alternatives?

Combinatorial Domains

A *combinatorial domain* is a Cartesian product $\mathcal{D} = D_1 \times \cdots \times D_n$ of n finite domains. We want to represent *utility functions* over $\mathcal{D}$.

Focus on *allocation problems:* set $\mathcal{G}$ of indivisible goods; each agent has utility function $u : 2^{\mathcal{G}} \rightarrow \mathbb{R}$, mapping bundles of goods to the reals.

That is, here each D_i is a *binary domain*, and $n = |\mathcal{G}|$.

Explicit Representation

The *explicit form* of representing a utility function u consists of a table listing for every bundle $S \subseteq \mathcal{G}$ the utility $u(S)$.

By convention, table entries with $u(S) = 0$ may be omitted.

- the explicit form is *fully expressive:* any utility function $u : 2^{\mathcal{G}} \to \mathbb{R}$ may be so described
- the explicit form is *not succinct:* it may require up to 2^n entries

Even very simple utility functions may require exponential space: e.g. the function $u : S \mapsto |S|$ mapping bundles to their cardinality.

Weighted Propositional Formulas

A compact representation language for modelling utility functions over products of binary domains —

Notation: finite set of propositional letters PS; propositional language $\mathcal{L}_{PS}$ over PS to describe requirements, e.g.:

$$p, \quad \neg p, \quad p \wedge q, \quad p \vee q$$

A *goalbase* is a set $G = \{(\varphi_i, \alpha_i)\}_i$ of pairs, each consisting of a (consistent) propositional formula $\varphi_i \in \mathcal{L}_{PS}$ and a real number α_i. The utility function u_G generated by G is defined by

$$u_G(M) \quad = \quad \sum\{\alpha_i \mid (\varphi_i, \alpha_i) \in G \text{ and } M \models \varphi_i\}$$

for all models $M \in 2^{PS}$. G is called the *generator* of u_G.

Example: $\{(p \vee q \vee r, 7), (p \wedge q, -2), (\neg s, 1)\}$

A Family of Languages

By imposing different restrictions on formulas and/or weights we can design different representation languages.

Regarding *formulas*, we may consider restrictions such as:

- *positive* formulas (no occurrence of $\neg$)
- *clauses* and *cubes* (disjunctions and conjunctions of literals)
- k*-formulas* (formulas of length $\leq k$), e.g. 1-formulas = literals
- combinations of the above, e.g. k-pcubes

Regarding *weights*, interesting restrictions would be $\mathbb{R}^+$ or $\{0, 1\}$.

If $H \subseteq \mathcal{L}_{PS}$ is a restriction on formulas and $H' \subseteq \mathbb{R}$ a restriction on weights, then $\mathcal{L}(H, H')$ is the language conforming to H and H'.

Properties

We are interested in the following types of questions:

- Are there restrictions on goalbases such that the utility functions they generate enjoy natural structural properties?
- Are some goalbase languages more succinct than others?
- What is the complexity of reasoning about preferences expressed in a given language?

Y. Chevaleyre, U. Endriss, and J. Lang. *Expressive Power of Weighted Propositional Formulas for Cardinal Preference Modelling*. Proc. KR-2006.

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

Expressive Power

An example for a language that is fully expressive:

Theorem 1 (Expressivity of pcubes) *$\mathcal{L}(pcubes, all)$, the language of positive cubes, can express all utility functions.*

<u>Proof sketch:</u> Show how to build a goalbase for any given function u: (1) $\top$ must get weight $u(\emptyset)$. (2) Weights of longer formulas are uniquely determined by the weights of their subformulas. ✓

In fact, $\mathcal{L}(pcubes, all)$ has a *unique* way of representing any given u.

$\mathcal{L}(cubes, all)$, for example, is also fully expressive, but not unique:
$\{(p \wedge q, 5), (p \wedge \neg q, 5), (\neg p \wedge q, 3), (\neg p \wedge \neg q, 3)\} \equiv \{(\top, 3), (p, 2)\}$

Expressive Power: Modular Functions

A function $u : 2^{PS} \to \mathbb{R}$ is *modular* if for all $M_1, M_2 \subseteq 2^{PS}$ we have:

$$u(M_1 \cup M_2) = u(M_1) + u(M_2) - u(M_1 \cap M_2)$$

Here's a nice characterisation of the modular functions:

Theorem 2 (Expressivity of literals) *$\mathcal{L}(literals, all)$, the language of literals, can express all modular utility functions, and only those.*

Relative Succinctness

If two languages can express the same class of utility functions, which should we use? An important criterion is *succinctness*.

Let $\mathcal{L}$ and $\mathcal{L}'$ be two languages that can define all utility functions belonging to some class $\mathcal{U}$.

We say that *$\mathcal{L}'$ is at least as succinct as $\mathcal{L}$* over $\mathcal{U}$ if there exist a mapping $f : \mathcal{L} \to \mathcal{L}'$ and a *polynomial* function p such that for all expressions $G \in \mathcal{L}$ with the corresponding function $u_G \in \mathcal{U}$:

- $G \equiv f(G)$ (they both represent the same function); and
- $size(f(G)) \leq p(size(G))$ (polysize reduction).

Explicit Form v. Positive Cubes

Both the explicit form and $\mathcal{L}(pcubes, all)$, the language of positive cubes, are fully expressive. Which is more succinct?

Theorem 3 (Explicit form and positve cubes) *The explicit form and $\mathcal{L}(pcubes, all)$ are incomparable in terms of succinctness.*

Proof: These functions prove the mutual lack of a polysize reduction:

- $u_1(X) = |X|$: requires n weighted 1-pcubes (*linear*); but $2^n - 1$ non-zero values in the explicit form (*exponential*). ✓
- $u_2(X) = 1$ for $|X| = 1$ and $u_2(X) = 0$ otherwise: requires n non-zero values in the explicit form (*linear*); but $2^n - 1$ pcubes (*exponential*) — all cubes of length k need weight $(-1)^{k+1} \times k$. ✓

But: *interesting* functions usually more succinct in $\mathcal{L}(pcubes, all)$

The Efficiency of Negation

If we allow *negation* in our language, we can do better than either one of the two languages considered before:

Theorem 4 (Cubes and pcubes) *The language $\mathcal{L}(cubes, all)$ is strictly more succinct than the language $\mathcal{L}(pcubes, all)$.*

Theorem 5 (Cubes and explicit form) *The language $\mathcal{L}(cubes, all)$ is strictly more succinct than the explicit form.*

Computational Complexity

Other interesting questions concern the complexity of reasoning about preferences. Consider the following decision problem:

> $\textsc{MaxUtil}(H, H')$
> **Instance:** Goalbase $G \in \mathcal{L}(H, H')$ and $K \in \mathbb{Z}$
> **Question:** Is there an $M \in 2^{PS}$ such that $u_G(M) \geq K$?

Complexity results include:

- $\textsc{MaxUtil}(\mathit{all}, \mathit{all})$ is *NP-complete*.
- Even $\textsc{MaxUtil}(\mathit{2\text{-}pcubes}, \mathit{all})$ is *NP-complete*.
- But $\textsc{MaxUtil}(\mathit{pos}, \mathit{pos})$ and $\textsc{MaxUtil}(\mathit{literals}, \mathit{all})$ are *easy*.

Also interesting: What is the complexity of finding an allocation that maximises *utilitarian* or *egalitarian social welfare* (for language X)?

Application: Distributed Negotiation

Scenario: indivisible goods; agents with valuation functions

Goal: Want to design negotiation protocols for agents with good properties, ideally fast convergence to a socially optimal state.

Preference representation is one of several parameters in the model. Explicit modelling of the language has several advantages:

- Can guide *elicitation* of preferences from agents.
- Can characterise special *classes of preferences* that avoid impossibilities, allow for simpler protocols, etc.
- Permits *complexity* analysis.

Y. Chevaleyre, U. Endriss, S. Estivie, and N. Maudet. Multiagent Resource Allocation in k-additive Domains. *Annals of Operations Research*, 2008.

Application: Combinatorial Auctions

Combinatorial Auction: auction for simultaneously selling several items (with complements and substitutes)

Example: CA for a pair of shoes v. 2 auctions for one shoe each

Bidding is the process of communicating one's preferences to the auctioneer (truthfully or otherwise). Can use *goalbase languages!*

Winner determination is the problem faced by the auctioneer to decide which goods to award to which bidder.

- Winner determination is known to be *NP-hard*.
- *Heuristic-guided search* (AI technique) can often give optimal solution in reasonable time.

J. Uckelman and U. Endriss. *Winner Determination in Combinatorial Auctions with Logic-based Bidding Languages*. Proc. AAMAS-2008.

Conclusion

- Combinatorial explosion ⇒ number of alternatives can get huge ⇒ collective choice mechanisms need to be adapted
- Logic-based languages are good candidates for modelling preferences in combinatorial domains.
- Wider research area: *computational social choice*
- *Papers* are on my website (including surveys below), and you are welcome to attend our *seminar* (Plantage Muidergracht 24):

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.

Y. Chevaleyre, U. Endriss, J. Lang, and N. Maudet. Preference Handling in Combinatorial Domains: From AI to Social Choice. *AI Magazine*, Winter 2008.