Computational Social Choice 2021

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

[http://www.illc.uva.nl/~ulle/teaching/comsoc/2021/]

Plan for Today

We are going to briefly review a few additional research topics in COMSOC related to voting, with a focus on computational issues.

Our main topic will be *voting in combinatorial domains*.

The Paradox of Multiple Elections

Suppose 13 voters are asked to each vote *yes* or *no* on three issues:

- 3 voters each vote for YNN, NYN, NNY.
- 1 voter each votes for YYY, YYN, YNY, NYY.
- No voter votes for NNN.

If we use the *simple majority* rule *issue-by-issue*, then NNN wins, because on each issue 7 out of 13 vote *no*.

This is an instance of the *paradox of multiple elections:* the winning combination received not a single vote!

S.J. Brams, D.M. Kilgour, and W.S. Zwicker. The Paradox of Multiple Elections. *Social Choice and Welfare*, 15(2):211–236, 1998.

Voting in Combinatorial Domains

Suppose we need to decide on ℓ different *issues* (today: binary issues). What should *ballots* look like? What *aggregation rule* should we use?

Several approaches come to mind:

- Approach 1: vote on combinations (vectors in $\{0,1\}^\ell$) directly, using a common voting rule such as Plurality or Borda
- Approach 2: preselect a small number of admissible combinations
- Approach 3: elicit everyone's most preferred combination only, but use a sophisticated aggregation rule (e.g., $\min\max\mathrm{Hamming}$)
- Approach 4: vote sequentially on issues (to avoid the paradox)
- Approach 5: vote on compactly represented preferences ($\hookrightarrow$ next)

Exercise: *Can you identify obvious shortcomings of these approaches?*

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

Voting with Compactly Represented Preferences

Idea: Ask voters to report their ballots using a compact preference *representation language* and apply your favourite voting rule to the succinctly encoded ballots received.

Lang (2004) calls this approach *combinatorial vote*.

Discussion: A promising approach, but not too much is known to date about what would be good choices for preference representation languages or voting rules, or what algorithms to use to compute the winners. Also, complexity can be expected to be very high.

J. Lang. Logical Preference Representation and Combinatorial Vote. *Annals of Mathematics and Artificial Intelligence*, 2004.

The Language of Prioritised Goals

Associate issues with propositional variables (so: models = outcomes).

Then use propositional formulas to express *goals* and use *numbers* to indicate their importance. This induces a weak order:

> Suppose (φ, k_1) has higher *priority* than (ψ, k_2) if $k_1 > k_2$.
>
> Under the *lexicographic* form of aggregation, we would prefer model M to M' if there exists a k such that for all $j > k$ both M and M' satisfy *the same* number of goals of rank j, and M satisfies *more* goals of rank k.

Other forms of aggregation are possible as well.

J. Lang. Logical Preference Representation and Combinatorial Vote. *Annals of Mathematics and Artificial Intelligence*, 2004.

Example

Use the language of *prioritised goals* (1 has higher priority than 0) with *lexicographic aggregation* together with the *Borda rule:*

- Voter 1: $\{X{:}1,\ Y{:}0\}$ induces order $xy \succ_1 x\bar{y} \succ_1 \bar{x}y \succ_1 \bar{x}\bar{y}$
- Voter 2: $\{X \vee \neg Y{:}0\}$ induces order $x\bar{y} \sim_2 xy \sim_2 \bar{x}\bar{y} \succ_2 \bar{x}y$
- Voter 3: $\{\neg X{:}0,\ Y{:}0\}$ induces order $\bar{x}y \succ_3 \bar{x}\bar{y} \sim_3 xy \succ_3 x\bar{y}$

As the induced orders need not be strict linear orders, we use a *generalisation of the Borda rule:* an alternative gets as many points as she dominates other alternatives. So we get these Borda scores:

$$\begin{array}{ll} xy : 3+1+1 = 5 & \bar{x}y : 1+0+3 = 4 \\ x\bar{y} : 2+1+0 = 3 & \bar{x}\bar{y} : 0+1+1 = 2 \end{array}$$

So combinatorial alternative xy wins.

Combinatorial vote *proper* would be to compute the winner *directly* from the goalbases, without the detour via the induced orders.

Single Goals and Generalised Plurality

Next a complexity result exemplifying the limitations of the approach.

We use the following language and voting rule:

- Using the *language of single goals*, each voter specifies just one goal (an arbitrary propositional formula) with priority 1.
- Under the *generalised plurality rule*, a voter gives 1 point to each undominated alternative.

Here are two examples, for the set of variables $\{X, Y\}$:

- The goal $\neg X \wedge Y$ induces the order $\bar{x}y \succ xy \sim x\bar{y} \sim \bar{x}\bar{y}$, so only combination $\bar{x}y$ receives 1 point.
- The goal $X \vee Y$ induces the order $xy \sim \bar{x}y \sim x\bar{y} \succ \bar{x}\bar{y}$, so combinations xy, $\bar{x}y$, $x\bar{y}$ receive 1 point each.

Winner Determination under Plurality

Define the following decision problem, for a preference representation language $\mathcal{L}$ and a voting rule F:

> $\textsc{WinDet}(\mathcal{L}, F)$
> **Input:** Profile $\boldsymbol{R}$ expressed in $\mathcal{L}$; combination $x^\star$.
> **Question:** Is $x^\star \in F(\boldsymbol{R})$?

Bad news:

Proposition 1 (Lang, 2004) $\textsc{WinDet}$ *is coNP-complete for the language of single goals and the generalised plurality rule.*

Recall that coNP is the complement of the complexity class NP (i.e., it is the complexity class of checking validity in propositional logic).

J. Lang. Logical Preference Representation and Combinatorial Vote. *Annals of Mathematics and Artificial Intelligence*, 2004.

Proof

We show, equivalently to the claim, that deciding whether $x^\star$ is *not* a winner under plurality is NP-complete:

- NP-membership: Let $\varphi_1, \ldots, \varphi_n$ be the goals of the voters. The plurality score of any alternative x can be computed by adding one point for each voter i with $x \models \varphi_i$ or φ_i being inconsistent (the two cases in which x is undominated). But to *compare* two alternatives x and y, we only need to check $x \models \varphi_i$ and $y \models \varphi_i$, which we can do in polynomial time. Hence, if someone claims that $x^\star$ is *not* a winner and names a stronger alternative x, then this can be verified in polynomial time. ✓

- NP-hardness: By reduction from SAT. Let φ be a formula for which we want to check satisfiability. Let p be a new propositional symbol; then φ is satisfiable iff $\varphi \wedge p$ is. Create a single voter with goal $\varphi \wedge p$. Consider candidate combination $x^\star$ with $x^\star \not\models p$ (must exist), i.e., also $x^\star \not\models \varphi \wedge p$. Then $x^\star$ is *not* a plurality winner iff there exists another alternative x with $x \models \varphi \wedge p$, i.e., iff $\varphi \wedge p$ is satisfiable. ✓

Voting with Incomplete Preferences

Let's go over some further research topics related to voting . . .

In the classical (Arrovian) model of preference aggregation in SCT all agents are assumed to have (and report) *complete preferences*.

But in many scenarios preferences actually are *incomplete:*

- Bounded rationality: agents cannot reason about all alternatives
- Bounded attention: agents do not care about all alternatives
- Bounded scope: agents are not being asked about all alternatives

Exercise *Any initial ideas for designing voting rules for this setting?*

Z. Terzopoulou and U. Endriss. Aggregating Incomplete Pairwise Preferences by Weight. IJCAI-2019.

Possible Election Winners

Suppose we currently only have partial information about the voter preferences (but they themselves do have complete preferences).

We may ask: what are the *possible winners* under voting rule F?

Why is this interesting?

- Elicitation: can stop once (possible winner = necessary winner)
- Postal ballots may arrive late
- Some alternatives available only after voting has started
- Relationship to coalition manipulation (Exercise: *What is it?*)

Lots of complexity-theoretic results in the literature.

Example: General case in P for Plurality but NP-complete for Borda.

K. Konczak and J. Lang. Voting Procedures with Incomplete Preferences. Proc. Multidisciplinary Workshop on Advances in Preference Handling 2005.

Compilation Complexity

For voting rule F, given a partial profile, how many bits do we need to store (in the worst case) so we can compute the outcome later on, once the profile is complete? This is the *compilation complexity* of F.

Let n be the number of (old) voters and m the number of alternatives. We need $\lceil \log m \rceil$ bits to represent the name of one alternative. So the CC of any rule F is at most $n\lceil \log(m!) \rceil$ (just store all ballots!).

Exercise: *Prove these following observations to be correct!*

- for *anonymous* rule is at most $\min\{n\lceil \log(m!) \rceil, m!\lceil \log(n+1) \rceil\}$
- for *dictatorial* rules it is $\lceil \log m \rceil$
- for *constant* rules (always electing the same winner) it is 0

Chevaleyre et al. (2009) establish further such results, e.g., the fact that the CC of *Borda* is $\Theta(m \log(nm))$. Exercise: *Upper bound clear?*

Y. Chevaleyre, J. Lang, N. Maudet, and G. Ravailly-Abadie. Compiling the Votes of a Subelectorate. IJCAI-2009.

Communication Complexity

How much information do voters need to transmit?

Let n be the number of voters and m the number of alternatives. We need $\lceil \log m \rceil$ bits to communicate the identity of an alternative.

Upper bounds are easy to derive:

- Plurality: $O(n \log m)$ — each voter sends one alternative name
- Approval: $O(nm)$ — let each voter communicate a bit-string of length m to encode their approved subset of alternatives
- Borda (any rule with linear orders as ballots): $O(n\, m \log m)$ — each voter sends m alternative names in turn

Conitzer and Sandholm (2005) show that many of these bounds are tight, using tools from the theory of *communication complexity*.

V. Conitzer and T. Sandholm. Communication Complexity of Common Voting Rules. EC-2005.

Logic for Social Choice Theory

It can be insightful to model SCT problems in logic (Pauly, 2008):

- One research direction is to explore how far we can get using a *standard logic*, such as classical FOL. Do we need second-order constructs to capture IIA? (Grandi and Endriss, 2013)
- Another direction is to design *tailor-made logics* specifically for SCT (for instance, a modal logic). Can we cast the proof of Arrow's Theorem in natural deduction? (Ciná and Endriss, 2016)

M. Pauly. On the Role of Language in Social Choice Theory. *Synthese*, 2008.

U. Grandi and U. Endriss. First-Order Logic Formalisation of Impossibility Theorems in Preference Aggregation. *Journal of Philosophical Logic*, 2013.

G. Ciná and U. Endriss. Proving Classical Theorems of Social Choice Theory in Modal Logic. *Journal of Autonomous Agents and Multiagent Systems*, 2016.

Automated Reasoning for Social Choice Theory

Past work on *model checking* concrete implementations of voting rules and verifying proofs of classical results via *interactive proof assistants*.

Maybe most exiting is the use of *SAT solvers* to prove theorems in SCT. To prove an *impossibility theorem*, proceed as follows:

- model your social choice scenario using (some kind of) logic
- for fixed small values of n and m, rewrite in propositional logic
- use a SAT solver to establish unsatisfiability for these small values
- prove that impossibilities remain when we increase n or m

Results include: new proofs of known results, sharpening of known results (relaxation of conditions on n or m), and entirely new results. For simple domains, even automated *discovery* of results is feasible.

<u>Limitation:</u> So far mostly restricted to impossibility theorems.

C. Geist and D. Peters. Computer-Aided Methods for Social Choice Theory. In U. Endriss (ed.), *Trends in Computational Social Choice*. AI Access, 2017.

Axiomatic Justification of Election Outcomes

Can we also use axioms to directly justify a given election outcome (rather than to justify the rule used to compute the outcome)?

Example:

$a \succ b \succ c \quad \mapsto \quad \{a\}$ *Clear winner!* (FAITHFULNESS)

$\left.\begin{array}{l} c \succ b \succ a \\ a \succ b \succ c \end{array}\right] \mapsto \quad \{a, b, c\}$ *Note the symmetry!* (CANCELLATION)

$\longrightarrow \quad \{a\}$ *First voter breaks tie!* (REINFORCEMENT)

For a given corpus of axioms, can we compute such justifications automatically for any given profile?

A. Boixel and U. Endriss. Automated Justification of Collective Decisions via Constraint Solving. AAMAS-2020.

Summary

We have briefly touched upon several further topics related to voting:

combinatorial domains, incomplete ballots, possible winners, compilation and communication complexity, logical modelling, automated reasoning, and outcome justification

What next? Brief glance at a different research area: *fair allocation*.