Computational Social Choice 2024

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

[`http://www.illc.uva.nl/~ulle/teaching/comsoc/2024/`]

Plan for Today

Today we review several issues of a computational nature arising in social choice. The main one is voting in *combinatorial domains:*

- Voting on ℓ propositions (yes/no) in a referendum.
- Choosing k out of m alternatives, possibly under constraints

Clearly, the number of meta-alternatives can quickly get *very large*. So we face both a *choice-theoretic* and a *computational challenge*.

Further topics (one slide each):

- Compilation complexity of voting rules
- Communication complexity of voting rules
- Iterative voting

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*, 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
- Approach 4: vote sequentially on issues (to avoid the paradox)
- Approach 5: vote on compactly represented preferences

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

Approach 1: Vote on Combinations

Idea: Vote for combinations directly. Could ask each voter for her most preferred combination and apply the *plurality rule*.

This avoids the paradox and is computationally light.

Problem: This may lead to almost "random" decisions, unless domains are fairly small and there are *lots of* voters.

Example: Suppose there are 10 binary issues and 20 voters. Then there are $2^{10} = 1024$ combinations to vote for. Under the plurality rule, chances are very high ($\sim 83\%$) that no combination receives more than one vote (so the tie-breaking rule decides everything).

Remark: Similar comments apply for other voting rules that only elicit a small part of the voter preferences (e.g., k-approval with small k).

Using More Expressive Voting Rules

Idea: Vote for combinations directly, using your favourite voting rule with the full set of combinations as the set of alternatives.

If we use a voting rule that elicits *more information* than the plurality rule, then we can avoid the arbitrariness problem noted before.

Problem: Possible only for very small domains, certainly when the voting rule requires complete rankings (such as Borda).

Example: Suppose there are six binary issues. This makes $2^6 = 64$ possible combinations. Hence, under the Borda rule, each voter has to choose from amongst $64! \approx 1.27 \cdot 10^{89}$ possible ballots.

Approach 2: Preselect Admissible Combinations

Idea: Select a small number of combinations and then use your favourite voting rule to elect a winner from amongst those.

Problem: Who selects the admissible combinations? Not at all clear what criteria to use. This gives the chooser undue powers and opens up new opportunities for controlling elections.

Approach 3: Distance-based Aggregation

Idea: Elicit most preferred choices issue-by-issue (as in the paradox), but find a better way to aggregate this information.

Distance-based approaches are promising candidates:

- Define a *distance* metric on ballots (0-1 vectors).
- Extend it to measure distance of a ballot/outcome to a profile.
- Choose the outcome that *minimises* the distance to the profile.
- Possibly restrict attention to outcomes from some *admissible set* (e.g., the set of ballots submitted by at least one voter).

Example: The Minimax Rule

Brams et al. (2007) propose to elect the combination that minimises the *maximal Hamming distance* to any of the voter ballots:

- Distance between two vectors = # issues on which they differ
- Distance between vector and profile = maximum of distances
- Admissible set of outcomes = all outcomes

So if unhappiness = number of issues on which you don't get your way, this *minimax rule* maximises the happiness of the unhappiest voter.

Exercise: *What about the corresponding minisum rule?*

S.J. Brams, D.M. Kilgour, and M.R. Sanver. A Minimax Procedure for Electing Committees. *Public Choice*, 2007.

Consistent Distance-based Aggregation

We may also decide to choose from some *admissible set* of possible outcomes the combination that minimises the distance to the profile:

- The admissible set could be all outcomes that meet a certain *integrity constraint* (e.g., "say YES at least once").
- If we don't know the constraint, we could equate the admissible set with the set of ballots received: *representative-voter rules*

U. Endriss and U. Grandi. Binary Aggregation by Selection of the Most Representative Voter. AAAI-2014.

Approach 4: Sequential Voting

Idea: Vote separately on each issue, but do so sequentially to give voters the opportunity to make their vote for one issue dependent on other issues already decided upon.

Sequential Voting and Condorcet Losers

One interpretation of our paradox is that the winning combination might have been the *Condorcet loser*. Good news:

Proposition (Lacy and Niou, 2000): *Sequential plurality on binary issues never results in a winning combination that is a Condorcet loser.*

Proof: Just think what happens during the election for the final issue. The winning combination cannot be a Condorcet loser, because it wins against some other combination in the final round. ✓

A stronger requirement would be elect the *Condorcet winner* whenever it exists. Sequential voting does *not* guarantee this.

D. Lacy and E.M.S. Niou. A Problem with Referendums. *Journal of Theoretical Politics*, 2000.

Approach 5: 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:

$\mathrm{WDP}(\mathcal{L}, F)$
Input: Profile $\boldsymbol{R}$ expressed in $\mathcal{L}$ and combination $x^\star$.
Question: Is $x^\star \in F(\boldsymbol{R})$?

Bad news:

Proposition (Lang, 2004): WDP *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 $\textsc{Sat}$. 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. ✓

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.

Iterative Voting

Suppose voters are shown the profile and may, one by one, decide to update their ballot in response. *Will this converge?*

Convergence is hard to get, but there are positive results (w/o proof):

Theorem (Meir et al., 2010): *Under mild assumptions, if voters play best responses, iterative voting will converge under plurality.*

Proposition (Reijngoud and Endriss, 2012): *If voters promote their favourite of k frontrunners, iterative voting will converge for any PSR.*

Remark: $k = m$ means being *truthful*. $k = 1$ is the *bandwagon effect*.

R. Meir, M. Polukarov, J.S. Rosenschein, and N.R. Jennings. Convergence to Equilibria in Plurality Voting. AAAI-2010.

A. Reijngoud and U. Endriss. Voter Response to Iterated Poll Information. AAMAS-2012.

Summary

We have seen several approaches for tackling the problem of voting in *combinatorial domains* (i.e., voting in multi-issue elections).

To date, no clear solution has emerged. Good candidates:

- Distance-based approaches
- Sequential voting
- Voting with compactly expressed preferences

Any approach has to balance a *choice-theoretic challenge* (eliciting too little information from voters leads to paradoxical outcomes) and a *computational challenge* (eliciting too much information is intractable).

We also briefly discussed three other topics of a computational nature: *compilation complexity*, *communication complexity*, *iterative voting*.

What next? Beyond preference aggregation: judgment aggregation.