Aggregating Dependency Graphs into Voting Agendas in Multi-Issue Elections

Stéphane Airiau, Ulle Endriss, Umberto Grandi, Daniele Porello and Joel Uckelman

Institute for Logic, Language and Computation, University of Amsterdam

Many collective decision problems have a combinatorial structure: the agents involved must decide on multiple issues and their preferences over one issue may depend on the choices for some of the others. Voting is an attractive method for making such decisions, but multi-issue elections are challenging. On the one hand, requiring agents to consider all combinations of issues is computationally infeasible; On the other, decomposing the problem into several elections on smaller sets of issues can lead to paradoxical outcomes. Any pragmatic method for running a multi-issue election will have to balance these two concerns. We identify and analyse the problem of generating an agenda for a given election, specifying which issues to vote on together in local elections and in which order to schedule them.

An example

Three voters need to elect an element of a combinatorial domain:

$\{\texttt{salad}, \texttt{oysters}\} \times \{\texttt{trout}, \texttt{veal}\} \times \{\texttt{red}, \texttt{white}\}$

Wine advice: choose `white` with `oysters` or `trout`; `red` otherwise.

- Voter 1 wants as many of her favourite dishes (`salad` and `veal`) as she can; (getting a matching wine has secondary importance):
  Voter 1: $\texttt{svr} \succ \texttt{svw} \succ \texttt{ovw} \sim \texttt{stw} \succ \texttt{str} \sim \texttt{ovr} \succ \texttt{otw} \succ \texttt{otr}$
- Voter 2 loves `oysters` and `veal` and detests getting the wrong wine:
  Voter 2: $\texttt{ovw} \succ \texttt{svr} \sim \texttt{otw} \succ \texttt{stw} \succ \texttt{otr} \sim \texttt{ovr} \sim \texttt{str} \sim \texttt{svw}$
- Voter 3 loves `salad` and `trout` and prefers matching wine:
  Voter 3: $\texttt{stw} \succ \texttt{svr} \sim \texttt{otw} \succ \texttt{ovw} \succ \texttt{otr} \sim \texttt{ovr} \sim \texttt{str} \sim \texttt{svw}$

Three solutions

Voting Issue-by-Issue

- On the ballot sheet, ask each voter to indicate her preferred choice for each of the three issues.
- Not clear how the voters would react, as their issue-preferences will be *uncertain*. Suppose the are *optimists*: when voting on X, assume all other issues have been settled in the best possible way.

Then we get one of the worst possible outcomes: `salad-veal-white`.

This is the "paradox of multiple elections".

Voting on Combinations

We could vote on *combinations*, using our favourite voting rule:

- *Plurality:* tie between `svr`, `ovw` and `stw` $\rightsquigarrow$ tie-breaking rule
- *Borda* etc.: We'd have to adapt the rule to weak orders (ok). For larger examples we'd run into computational difficulties.
- *Approval voting:* Might be ok here, depending on how many combinations everyone approves. For larger examples, also heavy reliance on tie-breaking rule.

Sequential Voting

Each of the following agendas will result in the best possible outcome:

- Majority(starter) → Majority(main) → Majority(wine)
- Majority(main) → Majority(starter) → Majority(wine)
- Issue-by-issue(starter & main) → Majority(wine) [complex agenda]
- Borda on (starter & main)→ Majority(wine) [complex agenda]

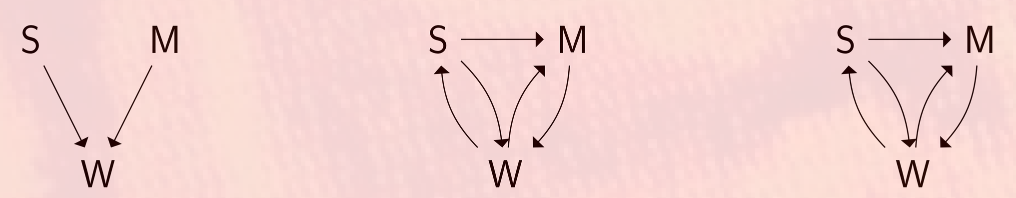
In all three cases, we'll elect `salad-veal-red` (which is the Condorcet winner).

How do we determine a suitable agenda?

Dependency Graph

We say that issue X *depends* on issue Y if there exists a situation where you need to know the value of Y to tell which of two possible values for X should be weakly preferred.

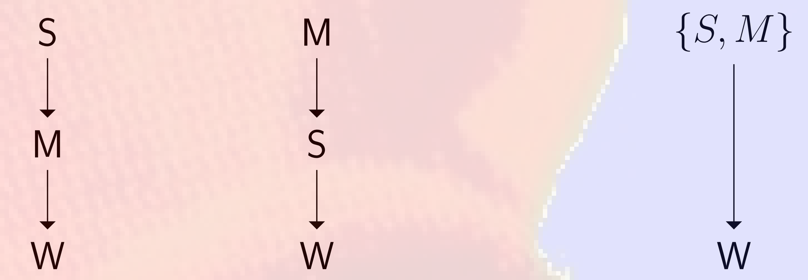
The Dependency Graphs of our Voters:



voter 1 voter 2 voter 3

What Agenda Should we Choose?

- Respect all dependencies: need to vote on all issues together, e.g., using Borda* (possible but computationally demanding).
- Respect dependencies shared by at least two voters:



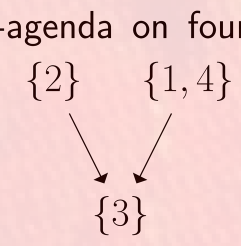
unnecessarily complex first election.

Agendas and Meta-Agendas

An *agenda* for a set of issues $\mathcal{I}$ is a linear order on a partition of $\mathcal{I}$.

A *meta-agenda* for $\mathcal{I}$ is an acyclic graph on a partition of $\mathcal{I}$.

An example for a meta-agenda on four issues $\mathcal{I} = \{1, 2, 3, 4\}$:



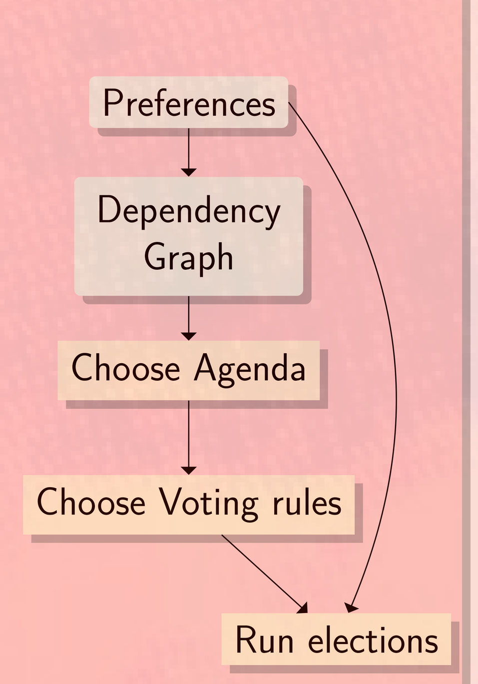
This meta-agenda represents two agendas:

- first run an election on issue 2, then an election on combinations of issues 1 and 4, and finally an election on issue 3
- first vote on 1 and 4, then on 2, and finally on 3

Approach: Sequential Voting with Complex Agendas

An approach to designing voting procedures for multi-issue elections:

1. Elicit some basic information from the voters (here: everyone's *dependency graph* over the issues at stake).
2. Choose an *agenda* (which issues to vote on together in local elections + order of local elections), based on dependencies.
3. Choose a *local voting procedure* for each local election.



The Agenda Choice Problem

Given a profile of preferential dependencies between issues, as reported by the voters, choose an agenda.

Our formal object of study are *meta-agenda choice functions:*

$$F : \mathrm{DG}(\mathcal{I})^{\mathcal{N}} \to 2^{\mathrm{MAG}(\mathcal{I})} \setminus \{\emptyset\}$$

Our proposal:

Study "*agenda choice problem*" in its own right.

Basic Meta-Agenda Choice Functions

All procedures given below map a profile of dependency graphs into a single collective dependency graph: $F : \mathrm{DG}(\mathcal{I})^{\mathcal{N}} \to \mathrm{DG}(\mathcal{I})$. We can then *condense* the collective graph to get a meta-agenda.

- *Majority aggregation:* include edge if a majority of voters do
- *Quota-based aggregation:* include edge if $\geqslant q\%$ of voters do
- *Canonical aggregation:* take the union of the input graphs
- *Distance-based aggregation:* choose a graph that is closest to the input profile, for a given metric (e.g., sum of Hamming distances)
- *Constraint-based aggregation:* choose a graph with clusters $\leqslant \ell$ that generates $\leqslant k$ dependency violations (there a several ways of counting violations: sum of all violations; no. of voter/election pairs where the voter experiences at least one uncertainty; ...)

Axiomatic Analysis

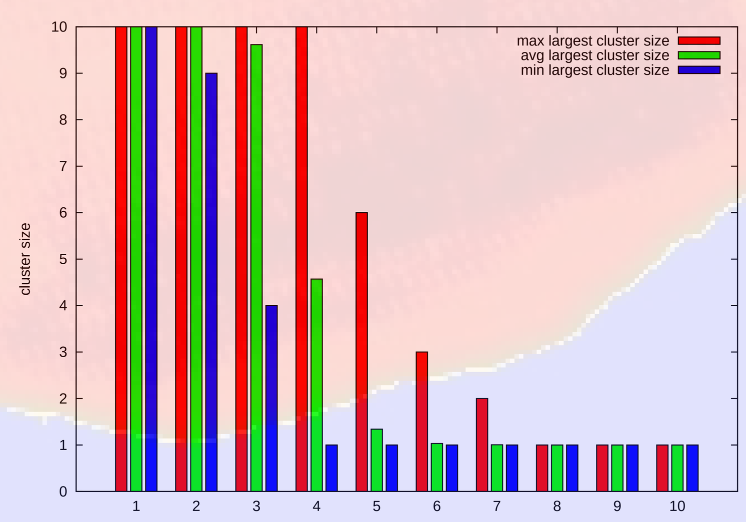
We can apply the axiomatic method to the study of MACFs. For example, *quota-based procedures* satisfy all of these axioms:

- *Anonymity:* symmetry wrt. input graphs
- *Dependency-neutrality:* for dependencies (a, b) and (a', b'), if each voter accepts both or neither, then so does the meta-agenda
- *Reinforcement:* if the intersection S of sets of meta-agendas for two subelectorates is $\neq \emptyset$, then S is the outcome for their union

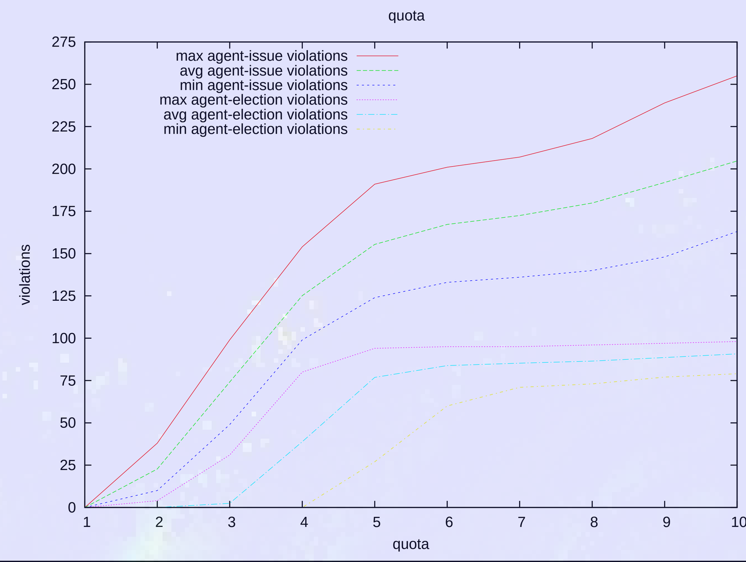
For *distance-based procedures*, some axiomatic properties are inherited from properties of the distances chosen:

- Any MACF defined in terms of a *neutral* distance (= invariant under renaming of vertices) on graphs is *dependency-neutral.*
- Any MACF defined in terms of a *symmetric* operator for extending distances between pairs of graphs to a distance between a graph and a set of graphs is *anonymous.*

Experiments



Largest cluster size for edgewise voting with 10 agents, 10 issues, and a full range of quotas



Number of violations for edgewise voting with 10 agents, 10 issues, and a full range of quotas

s.airiau@uva.nl, ulle.endriss@uva.nl, u.grandi@uva.nl, d.porello@uva.nl, j.d.uckelman@uva.nl