Arguing about Voting Rules

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

[joint work with Olivier Cailloux]

Talk Outline

Paper and talk focus on the problem of *justifying an election outcome* by means of a sequence of simple arguments:

- example of what a future system might be able to do
- logic for expressing arbitrary arguments about voting rules
- algorithm for justifying Borda outcomes

Example

Not always obvious who should win. For example, for the profile below the *Veto* rule recommends b, while the *Borda* rule recommends a:

Voter 1:	$a \succ b \succ c$
Voter 2:	$a \succ b \succ c$
Voter 3:	$c \succ b \succ a$

Suppose you want to convince a user that a should win ...

Voter 1:	$a \succ b \succ c$
Voter 2:	$a \succ b \succ c$
Voter 3:	$c \succ b \succ a$

System: Take the *red subprofile*. Here, a *should win*, right? [unanimity]

User: Obviously!

System: Now consider the *green subprofile*. For symmetry reasons, there should be a *three-way tie*, right? [cancellation]

User: Sounds reasonable.

System: So, as there was a three-way tie for the green part, the red part should decide the overall winner, right? [reinforcement]

User: Yes.

System: To summarise, you agree that a should win.

Voting Theory for Variable Electorates

Basic ingredients:

- $\mathcal{A}$: finite set of *alternatives*
- $\mathcal{L}(\mathcal{A})$: linear orders (*preferences*) on $\mathcal{A}$
- $\mathcal{N}$: infinite set of potential *voters*

A *profile* is a partial function $\boldsymbol{R} : \mathcal{N} \to \mathcal{L}(\mathcal{A})$ (pref's of some voters).

A *voting rule* f maps any given profile $\boldsymbol{R}$ to a nonempty set $A \subseteq \mathcal{A}$.

The Logic

Propositional language over atoms $[\boldsymbol{R} \mapsto A]$, one for each profile $\boldsymbol{R}$ and each nonempty set A of alternatives, interpreted on voting rules f:

$$f \models [\boldsymbol{R} \mapsto A] \;\; \underline{\text{iff}} \;\; f(\boldsymbol{R}) = A$$

Can express anything about voting rules, albeit in a brute force fashion.

For example, the *reinforcement* axiom can be written as the set of all the following formulas with $\text{dom}(\boldsymbol{R}) \cap \text{dom}(\boldsymbol{R'}) = \emptyset$ and $A \cap A' \neq \emptyset$:

$$[\boldsymbol{R} \mapsto A] \wedge [\boldsymbol{R'} \mapsto A'] \rightarrow [\boldsymbol{R} \oplus \boldsymbol{R'} \mapsto A \cap A']$$

Justifying Election Outcomes

Write $\Delta \models \varphi$ to say that every voting rule f that satisfies all the formulas in Δ also satisfies φ. For example:

- Δ might be a set of intuitively appealing properties (axioms)
- φ might be a claim about a specific outcome, such as $[\boldsymbol{R} \mapsto f(\boldsymbol{R})]$

Theorem 1 (Completeness) *$\Delta \models \varphi$ in our logic iff $\Delta \cup \textsc{Func} \vdash \varphi$ in classical propositional logic, where:*

$$\textsc{Func} \; = \; \bigcup_{\boldsymbol{R}} \left\{ \bigvee_{A} [\boldsymbol{R} \mapsto A] \right\} \cup \bigcup_{\boldsymbol{R}} \bigcup_{A \neq A'} \left\{ [\boldsymbol{R} \mapsto A] \wedge [\boldsymbol{R} \mapsto A'] \rightarrow \bot \right\}$$

Thus, we can prove claims φ about voting rules given assumptions Δ using, say, natural deduction. At least in theory.

In practice, Δ will usually be huge and deciding $\vdash$ is coNP-complete.

Justifying Borda Outcomes in Practice

Main technical contribution of the paper is an algorithm to compute, for any profile $\boldsymbol{R}$, a proof for $[\boldsymbol{R} \mapsto \text{Borda}(\boldsymbol{R})]$ from some axioms.

Main axioms used are:

- Reinforcement: $[\boldsymbol{R} \mapsto A] \wedge [\boldsymbol{R'} \mapsto A'] \to [\boldsymbol{R} \oplus \boldsymbol{R'} \mapsto A \cap A']$
- Cancellation: if all majority contests are tied, everyone wins

Main trick is to build a profile $\boldsymbol{R'}$ with (i) "obvious" winners $f(\boldsymbol{R})$ and (ii) same weighted majority graph as $k\boldsymbol{R}$. Claim then follows:

$$k\boldsymbol{R} \;\oplus\; \overline{k\boldsymbol{R}} \;\oplus\; \boldsymbol{R'}$$

Profile $\boldsymbol{R'}$ is built using Reinforcement on *basic profiles* such as:

$$\begin{bmatrix} a \succ b \succ c \succ d \\ b \succ a \succ d \succ c \end{bmatrix} \mapsto \{a, b\} \qquad \begin{bmatrix} a \succ b \succ c \succ d \\ d \succ a \succ b \succ c \\ c \succ d \succ a \succ b \\ b \succ c \succ d \succ a \end{bmatrix} \mapsto \{a, b, c, d\}$$

Last Slide

We have seen:

- *logic* for describing *example-based* properties of voting rules
- can be used to *justify outcomes* (in theory very general)
- concrete *algorithm* to compute short justifications for *Borda*

Long-term agenda: *arguing* about voting rules, beyond justification