# Advanced Topics in Computational Social Choice

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

## Plan for Today

So far our focus has been on the task of using SAT solvers to prove (base cases of) impossibility theorems.

In this final lecture, we are going to go beyond this specific task:

- Broader perspective: logic and automated reasoning for SCT
- Beyond SAT: what other tools can we use?
- Beyond impossibilities: what other types of results can we aim for?

# Logic for Social Choice Theory

*Why model social choice problems in logic?* Besides offering a deeper understanding of SCT and besides sometimes being of direct practical use, there also are philosophical arguments for doing so.

Pauly (2008) argues for *formal minimalism:*

- When considering an axiom in SCT, besides its *normative appeal* and its *logical strength*, we should also take into account the *expressivity of the language* used to define it. *Less is better.*

This perspective allows us, for instance, to investigate whether a given rule *can be axiomatised* at all, using a given logical language.

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

# Examples

Approaches to modelling social choice problems in logic:

- 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)

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.

# Formal Verfication

Logic has long been used to help verify the correctness of hardware and software. Can we use this methodology also here? *Yes!*

- Automated *verification* of a (known) proof of Arrow's Theorem in the HOL proof assistant Isabelle (Nipkow, 2009).
- Use of *model checking* to verify correctness of *implementations* (e.g., in Java) of voting rules (Beckert et al., 2017).

T. Nipkow. Social Choice Theory in HOL. *J. Automated Reasoning*, 2009.

B. Beckert, T. Bormer, R. Goré, M. Kirsten, and C. Schürmann. An Introduction to Voting Rule Verification. In *Trends in COMSOC*. AI Access, 2017.

## Beyond SAT

Beyond SAT, there are also a number of other (often logic-based) tools we might use in similar ways as we used SAT solvers:

- (Mixed) Integer Programming
- Constraint Programming
- SAT Modulo Theories (SMT)
- Answer Set Programming

# Beyond Impossibilities

Our focus has been on using SAT solvers to (search for and) prove (the base cases of) impossibility theorems. But we can do more:

- Prove set of axioms involved in an impossibility to be independent, by showing that every proper subset is satisfiable.
- Find an aggregation rule that satisfies a given set of axioms.
- Prove that axioms in $\Phi$ imply another axiom $\varphi$: $\text{UNSAT}(\Phi \cup \{\neg\varphi\})$.
- Justify an outcome $x$ for a given profile $p$ by appealing to axioms in $\Phi$ directly: $\text{UNSAT}(\Phi \cup \{\neg\text{`}x \text{ wins in } p\text{'}\})$ ... but still $\text{SAT}(\Phi)$.

However, there are some challenges associated with some of the above:

- Some findings only apply to fixed $n/m$ (no "inductive lemma").
- Interpreting findings saying that some set is satisfiable can be hard.

Btw: A very different application of SAT solvers would be to use them to implement computationally intractable aggregation rules.

## Summary

In this final lecture we briefly talked about:

- benefits of modelling social choice problems in logic
- application of formal verification tools to SCT
- alternatives to SAT solvers
- obtaining results other than impossibility theorems