Computational Social Choice 2023

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

[http://www.illc.uva.nl/~ulle/teaching/comsoc/2023/]

Plan for Today

Today's lecture will conclude our introduction to the SAT technique for proving axiomatic impossibility theorems in voting theory:

- MUS extraction to construct human-readable proofs
- literature review: other domains + refinements of technique
- first glance beyond impossibility theorems

Reminder

We saw how to formalise axioms as formulas in propositional logic with *variable* $p_{r,x}$ encoding that x is elected in profile r.

We saw that, for small n and m, such formulas are of manageable size and that we can use SAT solvers to determine *satisfiability*.

We saw that we can extend the (automatically derived) base case of an impossibility to the general case using an *inductive argument*.

We argued that we can *trust* the results obtained in this manner. But also noted that this is not the same as *understanding* those results.

Minimally Unsatisfiable Subsets

Think of a formula in CNF as a (possibly very large!) set of clauses.

To understand *why* a given set Φ of clauses is unsatisfiable, it can be helpful to inspect a *minimally unsatisfiable subset* (MUS) of Φ.

For a given unsatisfiable set Φ, any set $\Phi^\star$ is called an MUS of Φ if:

- $\Phi^\star \subseteq \Phi$,
- $\Phi^\star$ is unsatisfiable, but
- every proper subset of $\Phi^\star$ is satisfiable.

MUS Extraction

An MUS can serve as an *explanation* for the unsatisfiablity observed.
Let's try this for the Gibbard-Satterthwaite Theorem ...

Use `getMUS()` to compute an MUS of our 1,445-clause CNF:

```
>>> mus = getMUS(cnf)
>>> len(mus)
199
```

That's much better ... but not good enough. :(

Remark: This MUS includes all (three) surjectivity-clauses, all (two) nondictatorship-clauses, and 158 (out of 1,296) SP-clauses.

Weakening the Theorem

Geist and Peters suggest a weakening of G-S that works better:

Theorem (Geist and Peters, 2017) *For $n \geqslant 3$ and $m \geqslant 3$, no resolute voting rule is both strategyproof and majoritarian.*

Here being *majoritarian* means that x is elected whenever a strict majority ranks it at the top. Check Jupyter Notebook for the code.

Exercise: *Explain why this is weaker than G-S (at least for $n \geqslant 3$).*

Exercise: *Explain why the theorem is false for the case of $n = 2$.*

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

Proving the Theorem

Let's try it (with `n = 3`):

```
>>> cnf = ( cnfAtLeastOne() + cnfResolute()
...        + cnfStrategyProof() + cnfMajoritarian() )

>>> len(cnf)
12696

>>> solve(cnf)
'UNSATISFIABLE'

>>> mus = getMUS(cnf)
>>> len(mus)
21
```

So need to inspect just 21 of the 12,696 clauses to obtain a proof.

Seems feasible. *But can we do even better?*

Shuffle!

Note that an MUS need not be minimal in terms of its *cardinality*.

Trying a *different solver*—or just *shuffling* the CNF can help!

Re-run this code a few times and you should get an MUS of size 7:

```
>>> shuffle(cnf)
>>> mus = getMUS(cnf)
>>> len(mus)
7
```

Interpreting the MUS

Here's the small MUS we just found:

```
 >>> print(mus)
[[205,206,207],[-205,-152],[-206,-639],[-207,-199],[199],[152],[639]]
```

Easy to write code (*look it up!*) to help us interpret this:

```
>>> explainCNF(mus)
AtLeastOne: (102,210,021)->0 or (102,210,021)->1 or (102,210,021)->2
StrategyProof: not (102,210,021)->0 or not (102,102,021)->1
StrategyProof: not (102,210,021)->1 or not (102,210,210)->2
StrategyProof: not (102,210,021)->2 or not (012,210,021)->0
Majoritarian: (012,210,021)->0
Majoritarian: (102,102,021)->1
Majoritarian: (102,210,210)->2
```

The impossibility now is obvious! Done. ✓

Exercise: *The MUS does not include any resoluteness-clauses. Why?*

SAT Solving in the Research Process

What if the MUS is too big? What if the inductive proof doesn't work?

Fear not! In research, knowing what to prove before proving it is rare. Rather: the main challenge is often to identify good hypotheses.

SAT solving can be a great tool for this:

- use the SAT oracle to quickly check base cases for dozens (or hundreds) of axiom variations and combinations
- investigate further only the most interesting of those for which you get an UNSAT result (possibly using entirely traditional methods)

Literature Review

Geist and Peters (2017) also review parts of the literature up to 2017.

Chatterjee and Sen (2014) comment on early contributions regarding the SAT approach to SCT from the perspective of Economics.

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

S. Chatterjee and A. Sen. Automated Reasoning in Social Choice Theory: Some Remarks. *Mathematics in Computer Science*, 2014.

The Original Paper

Tang and Lin (2009) were the first to use the approach and applied it to construct a new proof of Arrow's Impossibility Theorem.

Their focus was on the inductive proof, while their main impact later turned out to be the idea of automating the proof of the base case.

P. Tang and F. Lin. Computer-Aided Proofs of Arrow's and other Impossibility Theorems. *Artificial Intelligence*, 2009.

Ranking Sets of Objects

In the field of 'ranking sets of objects' people formulate axioms for how to extend an agent's preferences from objects to sets of objects.

In my paper with Christian Geist (2011), we applied the approach to proving impossibility theorems in this new domain. Novel ideas:

- *general reduction lemma:* base case implies full theorem for any combination of axioms meeting certain syntactic constraints
- *automatic theorem discovery:* systematic search over 20 axioms

Found 84 impossibility theorems (some classical; some new—ranging from trivial to interesting; one contradicting wrong result in literature).

C. Geist and U. Endriss. Automated Search for Impossibility Theorems in Social Choice Theory: Ranking Sets of Objects. *Journal of AI Research*, 2011.

Tournament Solutions

Brandt and Geist (2016) provide in-depth analysis of strategyproofness for irresolute voting rules (and specifically tournament solutions).

Important paper pioneering *advanced encoding techniques* you might use when a naïve encoding (which was sufficient for G-S) won't work.

F. Brandt and C. Geist. Finding Strategyproof Social Choice Functions via SAT Solving. *Journal of AI Research*, 2016.

The No-Show Paradox

Moulin (1988) showed that every *Condorcet-consistent* voting rule suffers from the *no-show paradox:* sometimes it's best to abstain!

Brandt et al. (2017) used the SAT approach to find a *minimal profile* exhibiting the no-show paradox. So here the theorem was known, but SAT helped find a simpler and more interesting base case. Novel ideas:

- *incremental proof discovery:* start with weaker claims and use results to guide proof search over restricted range of profiles
- proof by *graphical representation* extracted from MUS

H. Moulin. Condorcet's Principle Implies the No Show Paradox. *Journal of Economic Theory*, 1988.

F. Brandt, C, Geist, and D. Peters. Optimal Bounds for the No-Show Paradox via SAT Solving. *Mathematical Social Sciences*, 2017.

Multiwinner Voting

Peters (2018) proved an important result for multiwinner voting regarding the incompatibility of *proportionality* and *strategyproofness* (at least for resolute rules and under mild efficiency requirements).

Particularly worth reading for these reasons:

- insightful discussion of how to look for an impossibility result in a new domain, with *axioms of varying strengths* being considered
- complex *inductive proof* over three variables (n, m, k)

Kluiving et al. (2020) discuss the generalisation to irresolute rules.

D. Peters. Proportionality and Strategyproofness in Multiwinner Elections. AAMAS-2018. Important erratum at `bit.ly/prop-sp-18`.

B. Kluiving, A. de Vries, P. Vrijbergen, A. Boixel, and U. Endriss. Analysing Irresolute Multiwinner Voting Rules with Approval Ballots via SAT Solving. ECAI-2020.

Probabilistic Social Choice

Brandl et al. (2018) used SMT solving (*Satisfiability Modulo Theories*) to obtain results in the domain of probabilistic social choice.

Shows that the approach can work also in domains that might at first seem ill-suited to a logic-based approach (due to involving numbers).

F. Brandl, F. Brandt, M. Eberl, and C. Geist. Proving the Incompatibility of Efficiency and Strategyproofness via SMT Solving. *Journal of the ACM*, 2018.

Matching Markets

My 2020 paper applies the SAT approach to matching mechanisms.

Includes a particularly simple case of a *general reduction lemma*.

U. Endriss. Analysis of One-to-One Matching Mechanisms via SAT Solving: Impossibilities for Universal Axioms. AAAI-2020.

Fair Division

Brandl et al. (2021) used the SAT approach to settle a 15-year-old conjecture on the impossibility of designing efficient and stratgeyproof rules for distributing money between projects approved by voters.

Aside: Shows that sometimes proofs of claims about models involving numbers might not actually need to refer to those numbers.

F. Brandl, F. Brandt, D. Peters, and C. Stricker. Distribution Rules Under Dichotomous Preferences: Two Out of Three Ain't Bad. EC-2021.

Beyond Impossibilities

What are opportunities for SAT in SCT beyond proving impossibilities?

- proving axioms involved in an impossibility to be *independent*, by showing that every proper subset is satisfiable
- finding an aggregation rule that *satisfies* a given set of axioms
- proving that axioms in Φ *imply* axiom φ: $\text{UNSAT}(\Phi \cup \{\neg\varphi\})$
- *justifying* and *explaining* collective decisions ($\hookrightarrow$ more soon)

However, there are certain challenges associated with some of this:

- might require new ideas to generalise beyond fixed n/m
- interpreting findings saying that CNF is *satisfiable* might be hard

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

Beyond SAT

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

- (Mixed) Integer Programming
- Constraint Programming
- SAT Modulo Theories (SMT)
- Answer Set Programming
- First-Order Theorem Proving

Summary

The past three lectures have been an introduction to automated reasoning, and specifically SAT solving, in social choice theory.

- Impossibilities: base case via SAT, human-readable proof via MUS, full proof via induction (*or just use the approach as a heuristic!*)
- Beyond impossibilities: opportunities mostly still unexplored

What next? Broadening the scope: beyond the classical voting model.