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 will complete our computer-aided proof of the G-S Theorem:

- *encoding* relevant axioms in propositional logic
- proving the *base case* by calling a SAT solver
- extending the result to the *general case*

Reminder

Aiming for a proof of the *G-S Thm*, we want to encode the special case of $n = 2$ voters and $m = 3$ alternatives as a *SAT instance*.

We want to use *variables* $p_{r,x}$ to say that when a voting rule is applied to profile r the outcome should include alternative x.

We decided to encode all components of the model as *integers:*

- *voters* from 0 to `n-1` (0 to 1 for now)
- *alternatives* from 0 to `m-1` (0 to 2 for now)
- *preferences* from 0 to `m!-1` (0 to 5 for now)
- *profiles* from 0 to $\mathtt{m!}^{\mathtt{n}}\mathtt{-1}$ (0 to 35 for now)

We implemented some basic methods to retrieve objects of interest:

- `allVoters()`, `allAlternatives()`, `allProfiles()`
- `voters(c)`, `alternatives(c)`, `profiles(c)`

And some further methods to answer yes-no questions:

- `prefers(i,x,y,r)`, `top(i,x,r)`, `iVariants(i,r1,r2)`

Literals

Want propositional *variable* $p_{r,x}$ to say that in *profile* r the outcome should include *alternative* x. Enumerate them from 1 to `m!`$^{\mathtt{n}}$ `* m`:

```
def posLiteral(r, x):
    return r * m + x + 1

def negLiteral(r, x):
    return (-1) * posLiteral(r, x)
```

Given a literal $p_{r,x}$ (as a number), we need to be able to determine which profile r and which alternative x it is talking about.

Exercise: *How can we compute* `r` *and* `x` *when given* `r * m + x + 1`*?*

We can use this insight to implement a method to pretty-print literals:

```
>>> strLiteral(1)
'(012,012)->0'

>>> strLiteral(-108)
'not (210,210)->2'
```

Encoding the Requirements on Voting Rules

Now we can encode our requirements. Recall our basic formula saying that for every profile at least one alternative must be elected:

$$\varphi_{\text{at-least-one}} \quad = \quad \bigwedge_r \left(\bigvee_x \, p_{r,x} \right)$$

Translating this into code is immediate:

```
def cnfAtLeastOne():
    cnf = []
    for r in allProfiles():
        cnf.append([posLiteral(r,x) for x in allAlternatives()])
    return cnf
```

Try it on the Jupyter Notebook:

```
>>> cnfAtLeastOne()
[[1,2,3], [4,5,6], [7,8,9], [10,11,12], ..., [106,107,108]]
```

Resoluteness

Resoluteness says that for any profile r and any distinct alternatives x and y, not both alternatives are in the outcome for that profile.

Note: Can restrict last quantification to $x < y$ (taken as numbers).

$$\varphi_{\text{res}} \quad = \quad \bigwedge_{r} \left(\bigwedge_{x} \left(\bigwedge_{y \,|\, x<y} \neg p_{r,x} \vee \neg p_{r,y} \right) \right)$$

Again, coding this is immediate:

```
def cnfResolute():
    cnf = []
    for r in allProfiles():
        for x in allAlternatives():
            for y in alternatives(lambda y : x < y):
                cnf.append([negLiteral(r,x), negLiteral(r,y)])
    return cnf
```

Remark: For the following axioms, we now can *presuppose resoluteness*.

Strategyproofness

SP says: for any voter i, any (truthful) profile r, any of its i-variants r', any alternative x, any alternative y dispreferred to x by i in r, either y (bad) loses in r (truthful) or x (good) loses in r' (manipulated).

$$\varphi_{\mathsf{sp}} = \bigwedge_{i} \left(\bigwedge_{r} \left(\bigwedge_{r' \in i\text{-var}(r)} \left(\bigwedge_{x} \left(\bigwedge_{y \mid x \succ_i^r y} \neg p_{r,y} \vee \neg p_{r',x} \right)\right)\right)\right)$$

```
def cnfStrategyProof():
    cnf = []
    for i in allVoters():
        for r1 in allProfiles():
            for r2 in profiles(lambda r2 : iVariants(i,r1,r2)):
                for x in allAlternatives():
                    for y in alternatives(lambda y : prefers(i,x,y,r1)):
                        cnf.append([negLiteral(r1,y), negLiteral(r2,x)])
    return cnf
```

Surjectivity

Surjectivity really is a conjunction of disjunctions of conjunctions: for all alternatives x, there is a profile r where x wins *and all others lose*. Could translate to CNF. But given resoluteness, this is easier:

$$\varphi_{\text{sur}} \quad = \quad \bigwedge_{x} \left(\bigvee_{r} p_{r,x} \right)$$

```
def cnfSurjective():
    cnf = []
    for x in allAlternatives():
        cnf.append([posLiteral(r,x) for r in allProfiles()])
    return cnf
```

Nondictatorship

A resolute rule is nondictatorial if for every voter i there is a profile r where $\text{top}(i)$ loses (so: some alternative x equal to $\text{top}(i)$ loses).

$$\varphi_{\text{nd}} = \bigwedge_{i} \left(\bigvee_{r} \left(\bigvee_{x \mid x = \text{top}_r(i)} \neg p_{r,x} \right) \right)$$

Remark: Instead of the last disjunction, we could just write $\neg p_{r,\text{top}_r(i)}$. The chosen encoding (no function in subscript) arguably is cleaner.

```
def cnfNonDictatorial():
    cnf = []
    for i in allVoters():
        clause = []
        for r in allProfiles():
            for x in alternatives(lambda x : top(i,x,r)):
                clause.append(negLiteral(r,x))
        cnf.append(clause)
    return cnf
```

Running the SAT Solver

We need to determine whether the *master formula* is satisfiable:

$$\varphi_{\text{gs}} \quad = \quad \varphi_{\text{at-least-one}} \wedge \varphi_{\text{res}} \wedge \varphi_{\text{sp}} \wedge \varphi_{\text{sur}} \wedge \varphi_{\text{nd}}$$

Btw: this is a conjunction of 1,445 clauses (using 108 variables).

The method `solve()` provides access to a SAT solver.

Let's see what happens:

```
>>> cnf = ( cnfAtLeastOne() + cnfResolute() + cnfStrategyProof()
...       + cnfSurjective() + cnfNonDictatorial() )

>>>> len(cnf)
1445

>>> solve(cnf)
'UNSATISFIABLE'
```

So φ_{gs} really is unsatisfiable! Thus: G-S for $n=2$ and $m=3$ is true! ✓

Discussion: *Does this count? Do we believe in computer proofs?*

Computer Proofs

We can proof-read our *Python script* just like we would proof-read a mathematical proof. And we can use multiple *SAT solvers* and check they agree. So we can have some confidence in the result.

Missing Pieces

But some pieces are still missing:

- *Does the theorem generalise to arbitrary $n \geqslant 2$ and $m \geqslant 3$?*

  Intuitively almost obvious, though technically not that easy.
  Basic idea: *induction* over both n and m

- *Why does the theorem hold?* This proof does not tell us.

  But SAT technology can help here as well: *MUS extraction*

Completing the Proof of the G-S Theorem

Recall the theorem we want to prove:

Gibbard-Satterthwaite Theorem: *For $m \geqslant 3$ alternatives, no resolute voting rule is strategyproof, surjective, and nondictatorial.*

Instead we proved:

Base Case Lemma: *For $n = 2$ voters and $m = 3$ alternatives, no resolute voting rule is strategyproof, surjective, and nondictatorial.*

To complete the proof of G-S we require two further lemmas:

- impossible for $n \geqslant 2$ and $m = 3$ $\Rightarrow$ impossible for $n+1$ and $m = 3$
- impossible for $n \geqslant 2$ and $m = 3$ $\Rightarrow$ impossible for n and any $m > 3$

Proving these lemmas is tricky but possible ($\hookrightarrow$ next). A write-up can be found in the PhD thesis of Pingzhong Tang (2010).

P. Tang. Computer-aided Theorem Discovery: A New Adventure and its Application to Economic Theory. PhD thesis. HKUST, 2010.

Preparation

Recall that we have already seen a (fairly simple) proof of the fact that any resolute voting rule that is *surjective* and *strategyproof* must also be *Paretian*. We will use this fact for the two proofs that follow.

For the second proof, we also will make use of the fact that the G-S axioms entail *independence*, for which we saw another simple proof.

For each lemma, we prove the contrapositive of our first statement . . .

First Lemma: Induction on Voters

Lemma 1 *If there exists a resolute voting rule for $n + 1 > 2$ voters and three alternatives that is surjective, strategyproof, and nondictatorial, then there also exists such a rule for n voters and three alternatives.*

Proof sketch: Let $A = \{a, b, c\}$ and $N = \{1, \ldots, n\}$. Now take any resolute rule $F : \mathcal{L}(A)^{n+1} \to A$ that is surjective, SP, and nondictatorial.

For every $i \in N$, define $F_i : \mathcal{L}(A)^n \to A$ via $F_i(\boldsymbol{R}) = F(\boldsymbol{R}, R_i)$. And check:

- All F_i are *surjective:* Immediate from F being Paretian. ✓
- All F_i are *SP:* First, no $j \neq i$ can manipulate, given that F is SP.

  Now suppose voter i can manipulate in $\boldsymbol{R}$ using R'_i. Thus, i prefers $F(\boldsymbol{R}_{-i}, R'_i, R'_i)$ to $F(\boldsymbol{R}_{-i}, R_i, R_i)$. But then i also must prefer $F(\boldsymbol{R}_{-i}, R'_i, R'_i)$ to $F(\boldsymbol{R}_{-i}, R'_i, R_i)$ or $F(\boldsymbol{R}_{-i}, R'_i, R_i)$ to $F(\boldsymbol{R}_{-i}, R_i, R_i)$. But F would be manipulable in both cases (contradiction!) ✓
- At least one F_i is *nondictatorial* (proof on next slide). ✓

Proof Detail

Recall: $F : \mathcal{L}(A)^{n+1} \to A$. Fix $F_i : \mathcal{L}(A)^n \to A$ via $F_i(\boldsymbol{R}) = F(\boldsymbol{R}, R_i)$.

Claim: At least one F_i (for some voter $i \in N$) is *nondictatorial*.

Proof sketch: Towards a contradiction, suppose all F_i are dictatorial.

Then for at least one F_i, we must have $\mathrm{dictator}(F_i) = i$:

- In case all F_i have the same dictator, true by assumption. ✓
- Otherwise, focus on i, j with $i \neq \mathrm{dictator}(F_i) \neq \mathrm{dictator}(F_j) \neq j$. Now consider $(n{+}1)$-profile in which i and j vote as voter $n{+}1$, but $\mathrm{dictator}(F_i)$ and $\mathrm{dictator}(F_j)$ do not. *Contradiction!* ✓

But $\mathrm{dictator}(F_i) = i$ entails that *voter $n{+}1$ can manipulate rule F_i* by copying i's ballot when i has $n{+}1$'s second best alternative on top. ✓

Second Lemma: Reduction of Alternatives

Lemma 2 *If there exists a resolute voting rule for n voters and $m > 3$ alternatives that is surjective, strategyproof, and nondictatorial, then there also exists such a rule for n voters and three alternatives.*

Proof sketch: Let $m > 3$ and let $A = \{a_1, a_2, a_3, \ldots, a_m\}$. Take any resolute rule $F : \mathcal{L}(A)^n \to A$ that is surjective, SP, and nondictatorial.

For any $\{a,b,c\} \subseteq A$ and $R \in \mathcal{L}(\{a,b,c\})$, let $R^+ = R(1) \succ R(2) \succ R(3) \succ \cdots$

Now define a rule $F^{a,b,c} : \mathcal{L}(\{a,b,c\})^n \to \{a,b,c\}$ for three alternatives:

$$F^{a,b,c}(R_1, \ldots, R_n) \quad = \quad F(R_1^+, \ldots, R_n^+)$$

$F^{a,b,c}$ is well-defined (really maps to $\{a,b,c\}$) and surjective, because F is Paretian. $F^{a,b,c}$ also is immediately seen to be SP (given that F is).

Now show that $\{a,b,c\} \subseteq A$ can be selected so that $F^{a,b,c}$ is nondictatorial. If *all* subsets $\{x,y,z\}$ yield dictatorial rules, we obtain a contradiction: By *independence*, if $F^{x,y,z}$ has dictator i, that i is a "local dictator" for $\{x,y,z\}$ under F. So F has some local dictator for every triple. But these local dictators cannot be distinct voters, so F in fact must be dictatorial. ✓

Critique of the Approach

Proving such lemmas can be quite difficult, almost as difficult as proving the theorem itself. This is a valid concern. But:

- A successful proof for a special case with small n and m provides *strong evidence* for (though no formal proof of) a general result.

  Indeed: The G-S Theorem is surprising. Our lemmas are not at all!

  Can use this as a *heuristic* to decide what to investigate further.

- Sometimes you can prove a *general reduction lemma:* if the axioms meet certain conditions, every impossibility generalises from small to large scenarios (see examples cited below).

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

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

Summary

We completed the proof of the Gibbard-Satterthwaite Theorem:

- base case corresponds to an unsatisfiable formula
- general case can be settled using an inductive argument

In methodological terms, we understood that, at least in principle, any axiom can be expressed in propositional logic; and we saw that, at the very least, some common axioms can be expressed rather easily.

What next? Understanding the impossibility through MUS extraction.