Initial Plan for Generating Human-Readable Logical Proofs via the Use of SAT Solvers

CM3203 Individual Project - 40 credits

By Henrijs Princis (c1800857)

Supervised by: Richard Booth

Moderated by: Hiroyuki Kido

Project Description

The Problem

It is difficult and time-consuming to prove theorems in computational social choice by hand due to their complexity. Often, these theorems may be proven with the aid of a computer. A relatively recent technique for finding such proofs was described by Geist and Peters [1]. There are possible improvements that could be made to the technique, and it could be applied to a different field - argumentation theory.

Aims

The aim of this project is to implement the workflow described by Geist and Peters [1] for generating concise human-readable mathematical proofs for theorems via the use of SAT solvers. The approach will then be extended to find new theorems in computational social choice or argumentation theory, for example, multi-winner elections. Another possibility is to apply this technique on existing proofs in computational social choice to gain new insights.

Background and Importance

Computational Social choice is a field of study concerned with methods for collective decision making. It seeks to answer how best to aggregate individual preferences of several agents to arrive at a decision for a group. Computational social choice overlaps with economic theory known as *ranking sets of objects*.

In 1951, Kenneth J. Arrow constructed a proof (Arrow's Impossibility Theorem) which states that when voters have three or more candidates, no ranked voting electoral system can convert the ranked preferences of individuals into a community-wide ranking while also satisfying some *properties* (like transitivity and non-dictatorship).

A better understanding of social choice could lead to better designed and more informed methods for running elections which would more accurately capture the views of the voters.

In 2009 Tang and Lin used a new method of proving Arrow's Impossibility Theorem [4]. They encoded the base case as a *satisfiability problem* (SAT) and then used an SAT solver to prove the base case. They completed the proof by using induction.

The benefits of this approach are that SAT problems are *decidable* and much work has gone into finding fast algorithms to solve them.

This downside is that the proofs (or contradictions) generated are usually too long for humans to read, understand and trust. Therefore, the use Minimal Unsatisfiable Set (MUS) is used to reduce the base-case solution into the smallest possible form. From the MUS it is possible to create a graph which helps visualise the proof for the base-case and may offer a starting point for the inductive part of the argument.

Project Aims and Objectives

The project consists of five goals but its main end-goal is to prove a conjecture in argumentation theory proposed in [3].

Preliminary Work

- Generate human readable proofs for social choice theory using SAT solvers.
    - Generate a SAT encoding for the simplified Gibbard-Satterthwaite theorem.
    - Use a SAT solver to prove the base case of the theorem.
    - Use a MUS solver on the simplified Gibbard–Satterthwaite Theorem.
    - Generate a human readable tree diagram from the output given by the MUS solver.
    - Encode additional properties required for Gibbard–Satterthwaite Theorem. Verify that the SAT solver is able to prove the full theorem's base case.
    - Encode the properties required for the Arrow's impossibility theorem and prove the base case using the SAT solver.

SAT Solver Performance

- Performance comparison of different SAT solvers.
    - Performance impact using more efficient encoding for the properties.

Human Readable Proofs

- Use a MUS solver to generate minimum number of clauses which prove the base case.
- Generate or make a human readable tree diagram which displays the proof's base case.

New Insight and Results Replication

- Result replication of previously published results.
    - Compare the proof given by proposed approach to previously published proofs (The Campbell-Kelley theorem and May's theorem).

Proving a Conjecture

- Proving a new result in social choice theory or argumentation theory
    - Argumentation theory: satisfaction of in/out *Monotonicty* for the DAUC versions of interval methods.
    - Multi-winner elections: impossibility theorems for multi-winner elections.

Work Plan

Week 1 Initial Report

Deliverables

Install Linux (Ubuntu) to be able to install open-source SAT solvers.

Install a SAT solver and verify it's expected format (typically CNF).

Write the initial report.

Reading

Argumentation Theory and Computational Social choice background reading.

Milestone

Set up a workspace for the project.

Week 2 Encoding into SAT

Deliverables

By end of Wednesday

Write a Python script which encodes the simplified Gibbard-Satterhwaite Theorem's (GST) base case into CNF. Prove that the base case is unsatisfiable by using the SAT solver on the encoded GST.

By end of week

Follow the description outlined in [1] to encode additional properties required for the complete Gibbard–Satterthwaite Theorem using Python. Verify that the SAT solver can prove the base case.

Encode additional properties required for Arrow's Impossibility Theorem.

Foreseen difficulties

This step might take longer if axioms need to be adjusted or finetuned to be solved quickly by the SAT solver.

It may be difficult to verify that my encoding of the properties is correct.

Mitigation and Prevention

I will look at whether others who have implemented a similar workflow have a similar encoding. Perhaps their code is open source. I will then compare the way encodings are generated.

If the above fails and properties need additional steps to encode, I will ask my AI lecturer for help. He is an expert in the field and teaches how to encode properties into SAT as a part of his module.

To verify my encoding is correct, I will check whether the output of my solver matches the one reported in [1].

Milestone

Replicate the proof's base case outlined in [1].

Week 3 Minimal Unsatisfiable Set (Human Readable)

Deliverables

By end of Wednesday

Implement the MUS solver to find the minimal unsatisfiable set for the simplified Gibbard-Satterhwaite Theorem's base case.

By end of week

Generate or make a proof diagram from the MUS.

Reading

What are other ways to display the satisfiability result in human readable form? Perhaps natural language? Look into Proof theory and how to simplify proofs.

Optional

If time allows, look at generating proof diagram from the output of MUS automatically? Has this been done? If not, it could help a lot of people working on social choice.

Foreseen difficulties

It is unclear how difficult it is to convert the output of MUS into a proof diagram.

Setting up the MUS solver may be harder than it seems and I am unsure how long it will it take to run (since MUS is harder than SAT).

Mitigation and Prevention

The book [1] offers a brief explanation about how to encode the output of MUS into a diagram. However, additional reading up on the topic may be required. If there isn't any useful information on the topic, then finding a method to automate this procedure could be an interesting topic to investigate further.

Milestone

Human readable proofs can be generated using the method outlined in [1].

Week 4 SAT Solver Performance

Deliverables

By end of Wednesday

Choose 3 different SAT solvers from [2] to compare their performance on the simplified and full Gibbard Theorem.

Record the runtime and memory usage. Check whether this aligns with the expected results.

By end of week

Good understanding of Argumentation Theory and a plan how current approach could be applied to the new axiom that's described by Booth. et. al.

Optional

Try to improve performance by reducing the number of formulas generated by a naïve encoding of the axioms. Has this already been done? What's the performance difference?

Try to encode the axioms in first order logic for a more compact notation. Has this been done? What's the effect on performance?

Foreseen difficulties

Some SAT solvers may be difficult to install & run. They may accept different formats.

Mitigation and Prevention

Choose SAT solvers which are well documented and easy to install. Do not spend a long time on cutting edge SAT solvers if they require in-depth tinkering. The main aim of the project is to prove a conjecture in argumentation theory, not to examine the performance of SAT solvers.

Milestone

Choose the best SAT solver to use for impossibility theorem. This is the final peace of the puzzle that before being able to tackle the conjecture in [3].

Week 5-7 Argumentation Theory (1st Progress Review)

Progress Review

The first progress review meeting is scheduled for the 6th week beginning (**7th of March**). By this point I should have completed all of the tasks leading up to week 5 and should have started on trying to prove the property in argumentation theory.

Deliverables

Apply the SAT solver approach to the conjecture proposed by Booth et. al. in [3] to produce a proof of the minimum base case of the theorem.

Check more complicated base cases of the conjecture.

(Dis)prove by induction the general statement if a contradiction for the base cases does not exist.

Foreseen Difficulties

The conjecture cannot easily be encoded into SAT or the encoding will generate impossibly large SAT problem.

Mitigation and Prevention

If applying the method does not work, writeup why and discuss with supervisor. Attempt to prove a weaker version of the conjecture or focus on special cases of the conjecture.

Explore other conjectures which may be more suited for the current approach.

Week 8 Multi Winner Elections

Optional (It is unlikely I will (dis)prove an impossibility theorem without a lot more time; however, it is still worth allocating time to and looking at if only to outline the limitations of the proposed method. The alternative is to spend this week looking further into argumentation theory or replicating other famous result's in the hopes of new insight. See week 9 for more details.)

Deliverables

Generate a proof for the base case of the impossibility theorem. Apply the SAT solver approach to multi-winner elections to attempt to prove.

Mitigation and Prevention

Prove simpler cases. Read up on what has been done already. Discuss with supervisor. Writeup why the method fails or succeeds.

Week 9 Results Replication

Deliverables

Replicate previously published proofs in social theory (such as The Campbell-Kelley theorem and May's theorem) and in argumentation theory using human readable SAT solver approach.

Note the differences and similarities between proofs generated by the SAT solver and the original.

NB: This is different to preliminary work because [1] offers little information about how to apply this method to other proofs.

Milestone

Finish experimenting with the proposed technique and move on to

Week 10 Writeup Initial (2nd Progress Review)

Progress Review

The second progress review meeting is scheduled for the 10th weekend (**29th of April**). By this point I should have completed all and started writing up the report. I think I will have questions regarding the final report and how best to present the data I will have gathered.

Deliverables

Very basic first draft of the report containing "approach", "implementation" and "Results and Evaluation" sections.

Week 11 Writeup Continued

Deliverables

Almost complete draft of the report with the entire "Main body" section filled in.

Week 12 Writeup Final

Deliverables

Fully written report including the two support sections.

Gnatt Chart

Gnatt Chart SAT Solvers

ACTIVITY	PLAN START	PLAN DURATION	ACTUAL START	ACTUAL DURATION	PERCENT COMPLETE
Initial Report	1	1	1	0	80%
Initial Reading	1	1	1	0	50%
Workspace Setup	1	1	0	0	35%
Simplified Gibbard Proof	2	0.5	0	0	0%
Full Gibbard Proof	2.5	0.5	0	0	0%
Arrow's impossibility Proof	2.5	0.5	0	0	0%
MUS solver on Gibbard's theorem	3	0.5	0	0	0%
Tree diagram from MUS	3.5	0.5	0	0	0%
SAT solver performance analysis	4	1	0	0	0%
Preparation for Argumentation theory	4	0.5	0	0	0%
Argumentation theory	5	3	0	0	0%
Multi-winner elections	8	1	0	0	0%
Results Replication	9	1	0	0	0%
Writeup Initial	10	1	0	0	0%
Writeup Continued	11	1	0	0	0%
Writeup Final	12	1	0	0	0%

References

[1] - Trends in Computational Social Choice. University of Amsterdam. 2017. Edited by Ulle Endriss, Amsterdam. AI Access.

[2] - The International SAT Competition Web Page. available at: http://www.satcompetition.org/. Date accessed: 04/02/2022.

[3] - Booth et. al. 2014. Interval Methods for Judgment Aggregation in Argumentation. Fourteenth International Conference on the Principles of Knowledge Representation and Reasoning. Available at: https://www.aaai.org/ocs/index.php/KR/KR14/paper/viewPaper/7980. Date accessed: 04/02/2022

[4] - P. Tang and F. Lin. 2009. Computer-aided proofs of Arrow's and other impossibility theorems. Artificial Intelligence, 173(11):1041–1053. Available at: https://doi.org/10.1016/j.artint.2009.02.005 Date accessed: 04/02/2022