University of Amsterdam

MSc Artificial Intelligence
Master's Thesis

A Graph-Based Algorithm for the Automated Justification of Collective Decisions

by

Oliviero Nardi
12680001

July 2021

48 ECTS
26 October 2020 – 5 July 2021

Supervisors:
Arthur Boixel, MSc
Prof. Dr. Ulle Endriss

Assessor:
Dr. Ronald de Haan

Institute for Logic, Language and Computation

ABSTRACT

Social Choice Theory is the study of collective decision making. A well-known lesson from this field is that there is no perfect way to aggregate the individual views of a group into a single choice. To mitigate this, and support group deliberation, scholars recently proposed a formal model for justifying collective decisions. In this framework, a justification consists of a concrete explanation for why said decision is the "best" one, according to some normative conditions for collective decision procedures. An algorithm to automatically compute such justifications was also proposed, grounded in constraint programming. Unfortunately, this is a computationally hard task, and the proposed approach only works for decision problems of a very limited size. In this thesis, we build on this idea and propose an alternative algorithm for the automated justification of collective decisions, which takes inspiration from popular graph search algorithms. We prove the correctness of our approach and propose two general families of heuristics that can be used in combination with our algorithm. Then, we empirically assess the performance of our method, finding that it is capable of justifying moderately-sized input problems within minutes. These tests are conducted both on randomly generated inputs and on inputs extracted from real-world data. Finally, we analyse the justifications retrieved by our algorithm, and use the insights gathered to propose new directions of research to improve the approach further.

Usefulness depends on tractability.

— Kenneth J. Arrow (1995)

ACKNOWLEDGEMENTS

First and foremost, I would like to give my biggest thanks to my supervisors, Arthur Boixel and Ulle Endriss. Your insightful comments and meticulous feedback helped me immensely during this work, and I couldn’t have asked for better support. Besides this, thank you for helping me develop my mathematical writing skills, for introducing me to the beautiful field of Computational Social Choice, and for all your advice regarding my future.

Secondly, I would like to thank Ronald de Haan, for your suggestions about Answer Set Programming at the beginning of this work, and more broadly, for sparking my interest in automated reasoning through your course.

Furthermore, I want to thank all friends I made during my stay in Amsterdam. David and Tom, working with you on so many courses was a wonderful experience, and I learned so much from you. All my friends from the Carolina MacGillavrylaan street, thank you for all the beautiful moments and fun we had, and for making me feel at home abroad. In particular, thank you Paolo and Patrik for being such amazing flatmates, and for bearing with my fixation with Italian music.

To the people back home, in Vicenza and around Italy, thank you for your support in this. I cannot name you all, but I will make some honourable mentions. Alessandro, Edoardo, Giacomo, Giovanni, Jacopo, Leonardo, thank you for always rooting for me, and for listening to my ramblings about elections and theorems. Max, thank you for your almost daily support, and for reminding me so often to take it easy: you played a bigger part in this than you might think. Marco, thank you for convincing me to start this adventure abroad, and for inspiring me to keep on studying with passion and dedication.

Finally, I want to thank my family, the most important thing I have. To my nieces and nephews, aunts and uncles, in-laws, thank you for making my hometown always a lovely place to come home to. To my siblings, thank you for being such awesome role models, for teaching me so much about everything, and for always having my back. Last, but not least, I want to thank my parents. In such a tiny space I could never explain how thankful I am for everything you did for me, but nonetheless, thank you for always supporting me through my studies (materially and morally), for believing in me, and for always pushing me to follow my dreams.

CONTENTS

1 INTRODUCTION

This thesis is concerned with the automated justification of collective decisions. Precisely, our aim is to introduce an algorithm for this task that can be used in practice. In this chapter, we motivate our goal and discuss some related work. Furthermore, we define our objectives and the specific research question we address. Lastly, we give an overview of the structure of the thesis.

1.1 MOTIVATION AND RELATED WORK

When faced with the problem of collective decision making, how should we aggregate the individual preferences of agents into a single group choice? In Social Choice Theory, the study of collective decision processes (Arrow, Sen, and Suzumura, 2002), it is a well-established fact that there is no perfect way to do so.

One possible solution is to apply the axiomatic method. Pioneered by Arrow (1951), it is one of the main tools employed by Social Choice theorists. In this approach, researchers formalise a set of intuitively appealing normative principles (*axioms*) and reason about which kind of collective decision procedures (*voting rules*) satisfy them. Broadly, we can distinguish between two kinds of axiomatic results: *impossibilities* and *characterisations*. Impossibility results are theorems stating that no voting rule can satisfy a certain set of axioms. Classic examples of such theorems can be found in the works of Arrow (1951), Sen (1970), Moulin (1988), and Gibbard (1973) (also independently proved by Satterthwaite (1975)). Characterisation results, on the other hand, define a voting rule (or a class of voting rules) by showing it to be the only one satisfying a certain set of axioms (see the works of May (1952), Young (1974), Henriet (1985), Ching (1996), and Sekiguchi (2012)).

Thus, to return to our original question, we could use such characterisations to support collective decisions. Suppose the voters agree upon a set of axioms. If we can show that all voting processes that satisfy these axioms elect the same outcome, then we have constructed a *proof* in support of this outcome. In other words, if one accepts the axioms, then one must also accept the outcome.

Albeit interesting, this approach has some drawbacks regarding real-world applicability. Firstly, it is geared towards a limited public, as it requires users to be able to manipulate formal axioms and construct (or, at least, understand) mathematical proofs. Secondly, a proof of this kind could be of limited explanatory power: an abstract proof that speaks about all voting rules, without explaining *how* the outcome is related to the normative principles (or referring to the concrete situation the agents are in), might be unsatisfactory for a non-expert audience. Indeed, there is a difference between a *proof* (which merely shows that a fact holds) and a *proof that explains* (which explains why a fact holds) (Hanna, 2000). This seems particularly important in the context of group decisions: if we aim at satisfying all voters, we want them to understand *why* the elected outcome is appealing. Thus, it might be better to provide users with a set of concrete arguments (grounded on the agreed-upon axioms) in favour of the target outcome. By concrete arguments, we mean arguments that directly refer to the specific entities involved in the decision problem at hand.

Procaccia (2019) recently expressed similar views. He states that, in real-world applications, axioms should be used to explain outcomes to voters, rather than assisting designers in choosing a suitable rule. In the related field of fair division (Thomson, 2016), this view has been successfully applied. `Spliddit`[1] (Goldman and Procaccia, 2015), a website implementing algorithms for various division tasks (e.g., rent division), provides users with a detailed explanation, automatically computed, for why the returned outcome is the only one satisfying certain fairness requisites. In contrast, Procaccia argues, a similar tool does not exist in the context of voting. The nearest example is `RoboVote`,[2] but the explanations provided by this system (as Procaccia mentions) are much more obscure, and multiple users reached out to the `RoboVote` team asking for alternative explanations.

Nevertheless, Social Choice theorists have recently started to address the issue of automatic explanation of voting outcomes. Cailloux and Endriss (2016) propose a formal language to construct such concrete arguments (based on a given set of axioms) used to justify the outcomes of collective decisions. Furthermore, they define an algorithm to automatically construct these justifications when the outcome is the one selected by the Borda rule, a well-known voting rule (Zwicker, 2016). In particular, their justifications consist in step-by-step explanations for why a given outcome should be picked in a decision problem. Peters et al. (2020) showed that, for the same rule, it is possible to construct an explanation consisting of $\mathcal{O}(m^2)$ steps (where m is the number of the alternatives the agents can choose from). Moreover, they give a lower bound (for the length of such explanations) which applies to a broad class of voting rules. Kirsten and Cailloux (2018) present an algorithm, grounded in bounded model checking (Biere, 2009), to automatically explain why a voting rule violates a given axiom. This can be seen as a form of automatic argumentation *against* a voting rule. Finally, Boixel and Endriss (2020) extended the approach of Cailloux and Endriss by introducing a formal notion of *justification* for a voting outcome, along with an algorithm to compute them, which is based on constraint programming (Rossi, Van Beek, and Walsh, 2006). Interestingly, their solution works (in principle) for any set of axioms. Their approach addresses the two drawbacks raised earlier: firstly, the justifications are automatically derived. This

1 `http://www.spliddit.org/`

2 At the time when this chapter was originally written, `RoboVote` was accessible at `https://www.robovote.org/` (January 2021). However, by July 2021, it seems like the site is not longer online.

lifts the burden of constructing a proof from the user, who now only has to read and accept it. Secondly, they consist of concrete arguments (grounded in some normative principles) which refer to the specific situation the agents find themselves in.

Their work fits into the recent trend of applying tools from Automated Reasoning to the domain of *Computational Social Choice* (Brandt et al., 2016), a field that lies at the intersection of Computer Science and Social Choice Theory. A very prolific strain of work in this trend concerns automatic theorem proving via SAT-solving, a methodology initially pioneered by Tang and Lin (2009). More examples of this approach can be found in the works of Geist and Endriss (2011), Brandt and Geist (2016), Brandt, Geist, and Peters (2017), Endriss (2020), and Kluiving et al. (2020). For a broader introduction, refer to the review by Geist and Peters (2017). For different applications of Automated Reasoning to Social Choice, see the work of Beckert et al. (2017) (for the automatic verification of voting rules implementations via bounded model checking) or the works of Nipkow (2009) and Wiedijk (2009) (for an automatic verification of proofs of Arrow's Theorem in higher-order logic). More generally, for other logical formalisations of classical choice-theoretic results, see the works of Grandi and Endriss (2013) (first-order logic), Ågotnes, van der Hoek, and Wooldridge (2011), Troquard, van der Hoek, and Wooldridge (2011), and Ciná and Endriss (2016) (modal logics).

On the other hand, the work of Boixel and Endriss also aligns with a renewed surge of interest into "Explainable AI" (Guidotti et al., 2018; Arrieta et al., 2020). In particular, as Boixel and Endriss note, their concept of justification is consistent with the view on Explainable AI expressed by Langley (2019), where an intelligent system is said to exhibit *justified agency* "if it follows society's norms and explains its activities in those terms". Indeed, the algorithm proposed by Boixel and Endriss can *explain* why, according to some *normative principles*, a certain voting outcome is "the right one". Finally, let us note that other authors agree on the potential of Automated Reasoning as a tool to achieve explainability in AI (see, for example, the works of Bonacina (2017) or Benzmüller and Lomfeld (2020)).

One might wonder whether the approach of Boixel and Endriss is computationally feasible. Since we have adopted the point of view of real-world applicability so far, this seems like a legitimate concern. As it turns out, justifying an outcome is a hard task: the algorithm proposed by Boixel and Endriss can only handle decision problems of a limited size. More fundamentally, Boixel and de Haan (2021) show that the problem of automatic justification is computationally intractable in general. For some version of the problem, checking whether a justification is correct is DP-complete (with $\mathrm{DP} = \mathrm{NP} \wedge \mathrm{coNP}$,[3] as defined by Papadimitriou and Yannakakis (1982)) while checking whether a justification exists is Σ_2^P-complete. For another version of the problem, checking correctness is in $\mathrm{NEXP} \wedge \mathrm{coNEXP}$ (an exponential-time analogue of DP) and

3 Given two complexity classes A and B, the complexity class $A \wedge B$ corresponds to $\{ L_1 \cap L_2 \mid L_1 \in A, L_2 \in B \}$. Notice that this is not the same as $A \cap B$.

NEXP-hard, while deciding existence is in EXP^{NP} (an exponential-time analogue of the class $\text{P}^{\text{NP}} = \Delta_2^{\text{P}}$) and NEXP-hard.[4] Thus, in both cases, the problem is riddled with negative results.

However, although discouraging, these results do not necessarily mean that this enterprise is without hope. Indeed, there are many examples of provably hard problems that have algorithms that perform well in practice. For instance, deciding whether a propositional formula is satisfiable is known to be an NP-complete problem. Despite this, modern SAT solvers routinely handle problem instances involving millions of clauses (Gomes et al., 2008). This thesis will address this computational issue: despite the negative complexity-theoretic results, is it possible to build a justification algorithm that performs well in practice?

1.2 RESEARCH QUESTION

In this thesis, we aim at improving the algorithm proposed by Boixel and Endriss (2020). The research question can be summarised as:

> *"Is it possible to design an algorithm for the problem of computing justifications for collective decisions that achieves good performance in practice?"*

Of course, it is hard to define what "good performance in practice" is. In this setting, we mean that the resulting algorithm should be able to meet the practical needs of potential real-world users. However, since this is a computationally demanding task, we will restrict our focus to moderately-sized inputs. That is, the algorithm should be able to find a justification for voting problems with around ten voters and three to four candidates within a reasonable time (i.e., minutes). We argue that this is already enough to cover some interesting and high-stakes use cases. For example, a research group looking for a PhD candidate to hire, a small committee voting to enact a new policy, or the strategic unit of a company choosing where to open the next branch. Furthermore, in such small-scale scenarios, the stakeholders can concretely meet and discuss the justifications we provide. Thus, the automated justification of collective decisions seems particularly well-suited for these cases, as voters can integrate this procedure directly into their deliberation process. Moreover, since we are interested in explanations that directly refer to the concrete situations the voters are in, one would expect that, for large-scale scenarios, the computed justifications would be large as well. Clearly, an overly long explanation might be hard to read and understand. Thus, even if it could be computed, a justification for a situation involving many voters and candidates might not be that useful in practice.

4 For more details on Complexity Theory, refer to the book by Arora and Barak (2009).

1.3 THESIS OVERVIEW

This thesis is structured as follows:

- In Chapter 2, we present the formal model we use to talk precisely about electoral problems. In contrast to most of the work in Social Choice Theory (and in particular, to what Boixel and Endriss (2020) did), we employ an anonymous model. That is, we focus on elections where the names of the voters do not matter. Furthermore, we introduce the notion of axiom in Social Choice Theory, along with some useful additional notions. Finally, we present a small selection of axioms that we will use in our experiments.

- In Chapter 3, we present our main contribution, that is, the algorithm. We start by recalling the framework introduced by Boixel and Endriss (2020), presenting the definitions of *justification* and *justification problem*. Next, we define the notion of *justification algorithm* and specify what it means for such an algorithm to be *sound* and *complete*. Roughly, a sound justification algorithm can only return justifications for the input problem, whereas a complete algorithm finds all the justifications of the input problem that we consider *interesting*. If an algorithm is sound and complete, then it is *correct*. Then, we define a formal language (based on propositional logic) in which we can encode all axioms presented in Chapter 2. This is useful, as it enables us to delegate part of the computational task to state-of-the-art SAT solvers. Then, after briefly recalling the main algorithm proposed by Boixel and Endriss (2020), we present our approach, inspired by the Breadth-First Search and Iterative Deepening Search algorithms for graphs (Russell and Norvig, 2015). Our algorithm is grounded in the notion of *instance graph*: intuitively, an instance graph is a structured, graph-theoretic view of the kind of arguments that our justifications can express. After proving the correctness of our method, we introduce two (correct) families of heuristics. In this context, a heuristic is an axiom-specific variation of our algorithm that can be used to improve its performance.

- In Chapter 4, we directly address the research question. First, we present the implementation of the algorithm we used in our experiments. To do so, we specify some concrete heuristics, based on the two general families introduced in Chapter 3. These heuristics are tailored for the axioms presented in Chapter 2. That is, they aim at improving the performance for these particular axioms. However, we stress that adding new axioms does *not* require us to retract such heuristics. Next, we present our experimental setup, where we used the aforementioned implementation to tackle the research question. To do so, we ran our algorithm on several voting problems to assess its performance. In these tests, we employed randomly generated inputs, as well as inputs extracted from real-world data. We obtained this data from `Preflib`,[5] an online library of preference datasets (Mattei and Walsh, 2013). We then analyse our findings, noting that most inputs can be justified within a reasonable time (especially in the case of real-world data), and briefly relate them with the results of Boixel and Endriss (2020). All in all, we consider our results a positive answer to the research question.

- In Chapter 5, we take a closer look at the justifications that we can compute with our method (in combination with the axioms presented in Chapter 2). More precisely, we present the most common *patterns of explanation* or *argumentative structures* that we obtained in the justifications we have seen during our experiments. We then present a partial categorisation for these justifications. Finally, we reflect on the notions of explanation and explanatoriness and, based on these reflections, present some ideas on how to improve the performance of our approach further.

- In Chapter 6, we summarise our findings and propose some directions for future work.

5 https://www.preflib.org/

2 BACKGROUND

In this chapter, we introduce the model used throughout the rest of this work to speak precisely about collective decisions. Furthermore, we give a short introduction to the axiomatic approach in Social Choice Theory, along with a selection of some well-known axioms. This is relevant to the rest of the work, as the notion of justification we use is based on the notion of axiom.

2.1 MODEL

Let us start with an informal overview of the problem. In a group decision, a group of agents (or *voters*) need to collectively choose between a set of *alternatives*. Generally, each voter might have its own specific *preference* about the alternatives, and, to make the group choice, we ought to *aggregate* the preferences into one single choice. We call the mechanisms to aggregate the preferences *voting rules*.

We will model *anonymous voting rules*, that is, rules that only consider the number of voters who express a certain preference. Thus, in contrast to the standard (and more general) model employed in Social Choice Theory (see, for example, the introduction by Zwicker (2016)), our model will not capture the identities of the individual voters. Although less general, our model allows for a much more compact representation of the problem, and we consider this to be an acceptable restriction. Indeed, it seems reasonable to assume that in most real-world group decisions the identities of the voters are not taken into account. We will briefly compare our approach with the non-anonymous model at the end of this section.

Let us now introduce the model.

2.1.1 *Voters, Alternatives and Preferences*

A *voting scenario* is a pair $\langle n^\star, X \rangle$ where $n^\star$ is a positive natural number denoting the number of voters and X is a finite (and non-empty) set of alternatives. We represent a voter's preference as a strict linear order over the alternatives, that is, a complete, transitive and irreflexive binary relation. We denote the set of all such orders over X as $\mathcal{L}(X)$. For a given preference $\succ \in \mathcal{L}(X)$, we write $a \succ b$ if $(a, b) \in \succ$, meaning that a is (strictly) preferred to b.

2.1.2 *Anonymous Profiles*

We can think of an election as a collection of preferences. We call such a collection a *profile* and, in line with our anonymous approach, we represent profiles as *multisets*, which is a kind of set that allows for repetitions. Similar modellings of anonymous profiles can be found, for example, in the works of Gehrlein and Fishburn (1976), Merlin (2003) and Peters et al. (2020).

Formally, we can represent a multiset as a function $\mathcal{M} : M \to \mathbb{N}$ where M is called the *underlying set* of $\mathcal{M}$. Given an $x \in M$, we call $\mathcal{M}(x)$ the *multiplicity* of x. Further, we call $|\mathcal{M}|$ the cardinality of $\mathcal{M}$, defined as $|\mathcal{M}| = \sum_{x \in M} \mathcal{M}(x)$ (i.e., the sum of the multiplicities of all its objects). With a slight abuse of notation, we write $x \in \mathcal{M}$ if $x \in M$ and $\mathcal{M}(x) > 0$. Intuitively, this means that x occurs at least once in $\mathcal{M}$. Finally, given two multisets $\mathcal{M}$ and $\mathcal{N}$ with the same underlying set M, we define the *sum* of $\mathcal{M}$ and $\mathcal{N}$ as the multiset $\mathcal{M} + \mathcal{N} : M \to \mathbb{N}$ such that $(\mathcal{M} + \mathcal{N})(x) = \mathcal{M}(x) + \mathcal{N}(x)$.

We can then think of a profile $\mathbf{R}_n$ for n voters as a multiset with underlying set $\mathcal{L}(X)$ such that $|\mathbf{R}_n| = n$. In other words, a profile counts, for every possible preference $\succ \in \mathcal{L}(X)$, the number of voters with preference $\succ$. Furthermore, a profile for n voters must count n preferences in total. Following our multiset notation, given a preference $\succ$, we write $\mathbf{R}(\succ)$ for the number of voters who express the preference $\succ$.

Example 2.1. Consider the set of alternatives $X = \{a, b, c\}$. A profile with 3 voters could be:

$$\#1 : a \succ b \succ c$$
$$\#2 : c \succ a \succ b$$

This expresses the fact that one voter has preference $a \succ b \succ c$ and two voters each have preference $c \succ a \succ b$. A further example (this time with 2 voters) is profile $\{a \succ c \succ b, b \succ c \succ a\}$. This expresses the fact that one voter has preference $a \succ c \succ b$ and one voter has preference $b \succ c \succ a$. △

2.1.3 *Voting Rules*

Let $\mathcal{R}(X)^n$ be the set of all profiles over X for n voters. For a given voting scenario $\langle n^\star, X \rangle$, the set of all profiles up to $n^\star$ voters is $\mathcal{R}(X)^+ = \bigcup_{k=1}^{n^\star} \mathcal{R}(X)^k$. We can model a voting rule as a *social choice function* (SCF). An SCF for a scenario $\langle n^\star, X \rangle$ is a function of the form:

$$F : \mathcal{R}(X)^+ \to 2^X \setminus \{\varnothing\}$$

In other words, an SCF assigns a set of winning alternatives to each profile (of up to $n^\star$ voters). Notice that we are here focusing on the case of single-winner voting: an SCF should ideally return a unique choice, and multiple winners should be interpreted as ties. The problem of multiwinner voting (for example, when voting for a committee) is studied as a separate challenge in Social Choice Theory (Faliszewski et al., 2017).

Some well-known voting rules include:

- The **Plurality rule**, where the alternative that is ranked first most frequently is elected. More precisely, Plurality is defined as:

  $$\mathtt{Plurality}(\mathbf{R}) = \arg\max_{x \in X} \left\{ \sum_{\succ \in \mathbf{R}} \mathbf{R}(\succ) \cdot \mathbb{1}[\mathrm{top}(\succ) = x] \right\}$$

  Here, $\mathbb{1}$ is the indicator function[1] and $\mathtt{top}(\succ)$ returns the alternative that is ranked at the top in $\succ$.

- The **Borda rule**, where each alternative x is assigned a score from every voter, corresponding to the number of alternatives that the voter ranks below x. The alternative(s) with the highest total score win. More precisely, it is defined as:

  $$\mathtt{Borda}(\mathbf{R}) = \arg\max_{x \in X} \left\{ \sum_{\succ \in \mathbf{R}} \mathbf{R}(\succ) \cdot b(\succ, x) \right\}$$

  Here, $b(\succ, x)$ is the Borda score of x in $\succ$:

  $$b(\succ, x) = |\{ y \in X \setminus \{x\} \mid x \succ y \}|$$

- A prominent family of voting rules is the class of **Condorcet extensions**, the set of rules satisfying the *Condorcet Principle*. An SCF satisfies this principle if, whenever a *Condorcet winner* exists, it is elected. A Condorcet winner is an alternative $x^\star \in X$ such that, for all alternatives $y \in X \setminus \{x^\star\}$, a strict majority of voters prefer $x^\star$ to y. **Young's rule**, for example, is a Condorcet extension: this rule elects in $\mathbf{R}$ the alternative $x^\star$ which minimises the number of preferences you have to remove from $\mathbf{R}$ for $x^\star$ to become a Condorcet winner in $\mathbf{R}$.

  Notice that there are profiles with no such winners. For instance, in the following profile, no alternative is preferred by a strict majority over the other two:

  $$\#1 : a \succ b \succ c$$
  $$\#1 : b \succ c \succ a$$
  $$\#1 : c \succ a \succ b$$

1 The indicator function is defined as $\mathbb{1}[\varphi] = 1$ if φ is true, otherwise $\mathbb{1}[\varphi] = 0$.

For example, here, in a majority contest a wins against b but loses against c. Furthermore, interestingly, in voting scenarios with at least three alternatives and three voters, neither Plurality nor the Borda rule are Condorcet extensions. For example, in the following profile c is the Condorcet winner, but Borda elects $\{a, c\}$:

$$\begin{aligned}\#1 &: a \succ b \succ c\\ \#2 &: c \succ a \succ b\end{aligned}$$

Similarly, in the next profile b is the Condorcet winner, but Plurality elects $\{a, b, c\}$:

$$\begin{aligned}\#1 &: a \succ b \succ c\\ \#1 &: b \succ a \succ c\\ \#1 &: c \succ b \succ a\end{aligned}$$

As an aside, notice that in this profile Borda elects $\{b\}$ as well.

Coincidentally, from the above discussion, we can notice that on the same profile different voting rules might elect different outcomes. We will discuss the significance of this in Section 2.2.

2.1.4 *Comparison with a Non-Anonymous Model*

In their work, Boixel and Endriss (2020) consider the more general model of non-anonymous SCFs. Let us briefly sketch this model. In this framework, we have a set of *named* voters, $N^\star$, and a set of alternatives, X. A profile for a set $N \subseteq N^\star$ of voters is an element of $\mathcal{L}(X)^{|N|}$, that is, a vector of preferences (one for each specific voter).

This approach is more general, but this comes at the cost of a much larger space of possible profiles. Given that our goal is to implement an efficient algorithm, a more compact representation might be preferable in this setting. To make things more concrete, let us show some actual numbers. Given a voting scenario $\langle n^\star, X \rangle$ with $n^\star$ voters and alternatives in X (where $|X| = m$), how many possible profiles are there in the anonymous model? To count the total number of profiles, recall that a profile for n voters is a multiset of cardinality n and underlying set $\mathcal{L}(X)$. The number of multisets of cardinality n with an underlying set of cardinality k is $\binom{n+k-1}{n}$ (Stanley, 1997). Thus, given that $|\mathcal{L}(X)| = m!$, we can count the number of profiles for n voters as:

$$|\mathcal{R}(X)^n| = \binom{n + m! - 1}{n}$$

With this, we can compute the total number of profiles for our voting scenario:

$$|\mathcal{R}(X)^+| = \sum_{k=1}^{n^\star} \binom{k + m! - 1}{k}$$

Table 2.1 compares, for some voting scenarios, the size of this set to the number of profiles under the non-anonymous model employed by Boixel and Endriss (2020). Under their framework, for a voting scenario of $n^\star$ voters and m alternatives, we have $\sum_{k=1}^{n^\star} \binom{n^\star}{k} m!^k$ profiles.

		2 alternatives	3 alternatives	4 alternatives	5 alternatives
2 voters	*anon*	5	27	324	7380
	non-anon	8	48	624	$1.46 \cdot 10^4$
3 voters	*anon*	9	83	2924	$3.03 \cdot 10^5$
	non-anon	26	342	$1.56 \cdot 10^4$	$1.77 \cdot 10^6$
4 voters	*anon*	14	209	$2.05 \cdot 10^4$	$9.38 \cdot 10^6$
	non-anon	80	2400	$3.91 \cdot 10^5$	$2.14 \cdot 10^8$
5 voters	*anon*	20	461	$1.19 \cdot 10^5$	$2.35 \cdot 10^8$
	non-anon	242	$1.68 \cdot 10^4$	$9.77 \cdot 10^6$	$2.59 \cdot 10^{10}$

Table 2.1: A comparison, for some small voting scenarios, of the number of possible profiles under our anonymous model (*anon*) and the non-anonymous model employed by Boixel and Endriss (2020) (*non-anon*).

2.2 AXIOMS AND AXIOM INSTANCES

In this section, we introduce the notion of *axiom* in Social Choice Theory, along with some additional notions and terminology we will use throughout the rest of this work. We will also present a selection of some well-known axioms.

2.2.1 *The Axiomatic Approach in Social Choice Theory*

In the previous section, we have seen some examples of voting rules which, on a given profile, might elect different outcomes. This, however, poses a problem: how should we choose the *right* voting rule? What does "right" even mean in this context?

Some SCFs seem "obviously" unappealing, such as the function that always elects all alternatives. However, when comparing some intuitively reasonable rules, such as Plurality and Borda, how do we decide which one to use in our elections? One approach could be the following. We could decide on some properties that we regard as essential for a voting rule, and try to pick a mechanism that satisfies these properties.

This idea is at the core of the *axiomatic approach* (Endriss, 2011; Zwicker, 2016). This approach, pioneered by Arrow (1951), is one of the fundamental techniques within the field of Social Choice Theory.

In this context, an axiom is, generally speaking, an intuitively appealing normative requirement for a voting rule, specified in a precise mathematical way. An SCF may or may not satisfy an axiom, and axioms can be used to argue for or against a particular rule. Examples of such axioms are *Neutrality* (stating that all candidates must be treated equally), *Monotonicity* (stating that if a winning candidate receives increased support, then it must still be a winner) and the aforementioned *Condorcet Principle*. We have already seen how an SCF might violate an axiom. For instance, Plurality and the Borda rule both violate the Condorcet Principle. One could interpret this fact as an argument against these two rules.

More radically, an important fact to notice is that not all axioms are mutually compatible. Indeed, some combinations of normative principles are impossible to satisfy. We call a result stating that such a combination is impossible an *impossibility theorem*. There are plenty of impossibilities in the literature. Most notably, Arrow's Theorem (Arrow, 1951), one of the most important results (if not *the* most important result) of Social Choice Theory. In his work, Arrow proposed five reasonable and seemingly innocuous axioms and then proved that no voting rule can possibly satisfy all of these conditions at the same time.

However, not all axiomatic results are negative results. A more encouraging kind of result is a *characterisation theorem*, which defines a particular SCF (or a class of SCFs) by showing it to be the only one satisfying a certain set of axioms. For example, the Borda rule has been characterised in a seminal paper by Young (1974). This can be interpreted as an argument in favour of this rule: if all voters accept the axioms that characterise this SCF, then they must also accept to use it in their group decision.

So far, we discussed informally what it means for an SCF F to satisfy an axiom. To clarify this notion, one can define a formal language, with precise syntax and semantics. We will introduce a concrete language based on basic propositional logic in the next chapter. For now, irrespectively of the language chosen, we can define the *interpretation* of an axiom A as the set of SCFs that satisfy A. Similarly to Boixel and Endriss (2020), we denote this set as $\mathbb{I}(A)$, and the following holds:

$$\mathbb{I}(A) \subseteq \mathcal{R}(X)^{+} \to 2^{X} \setminus \{\varnothing\}$$

Here, $F \in \mathbb{I}(A)$ if and only if F satisfies A. Further, for a given set of axioms $\mathcal{A}$, we define the interpretation of $\mathcal{A}$ as:

$$\mathbb{I}(\mathcal{A}) = \bigcap_{A \in \mathcal{A}} \mathbb{I}(A)$$

In other words, an SCF F is in $\mathbb{I}(\mathcal{A})$ if and only if F satisfies all axioms in $\mathcal{A}$. Note that, given a set of axioms $\mathcal{A}$, an impossibility result would state that $\mathbb{I}(\mathcal{A}) = \emptyset$ (i.e., no SCF satisfies $\mathcal{A}$). Conversely, a characterisation result regarding an SCF F and a set of axioms $\mathcal{A}$ would state that $\mathbb{I}(\mathcal{A}) = \{F\}$ (i.e., F is the only rule that satisfies $\mathcal{A}$).

2.2.2 *Existential and Universal Axioms*

A large number of axioms have been proposed in the literature. One informal categorisation, slightly adapted from the work of Fishburn (1973), is the distinction between *existential* and *universal* axioms. This is only a partial categorisation; nonetheless, it will be useful for our presentation. Hence, let us now briefly review it.

Both existential and universal axioms restrict the behaviour of an SCF F, albeit differently. An existential axiom is an axiom that existentially quantifies over the profiles. For example, *Non-Imposition* is satisfied by F if, for every $x \in X$, there *exists* a profile $\mathbf{R}$ such that $F(\mathbf{R}) = \{x\}$. Notice that we used also a universal quantifier ("for every x"), but still, what matters is that we quantified existentially over the profiles. Furthermore, note that we did not impose for existential axioms to *only* quantify existentially over the profiles. That is, an existential axiom might quantify profiles both universally and existentially.[2]

On the other hand, a universal axiom quantifies *only* universally over the profiles. They usually have the following form: *For all profiles* $\mathbf{R}_1 \dots \mathbf{R}_k$*, if* $\mathbf{R}_1 \dots \mathbf{R}_k$ *satisfy some* condition*, then some* constraint *should hold over* $F(\mathbf{R}_1) \dots F(\mathbf{R}_k)$. Moreover, Fishburn calls the universal axioms that restrict one profile at a time (e.g., $k = 1$ in our example) *intraprofile* axioms. Further, he calls universal axioms that restrict multiple profiles *interprofile* axioms. For example, the Condorcet Principle is an intraprofile axiom, as it prescribes the following: "for every profile $\mathbf{R}$ with a Condorcet winner (*condition*), $\mathbf{R}$ must have the Condorcet winner as the sole winner (*constraint*)". Conversely, as we will see in more detail later, Neutrality is an interprofile axiom. This condition states the following: "for every pair of profiles $\mathbf{R}_1$ and $\mathbf{R}_2$, if the two profiles are equal up to a renaming of the alternatives (*condition*), then the outcomes of these two profiles should be equal up to the same renaming (*constraint*)". Intuitively, this expresses that the names of the alternatives should not matter.[3]

2 Perhaps, then, a better name for existential axioms might have been *non-universal* axioms. However, we stick to this terminology to highlight the fact that existential axioms *do* contain an existential quantifier over the profiles. As we will see, that is what matters here.

3 Note that this categorisation does not cover axioms that do not quantify over profiles. Although technically possible, these axioms do not seem to be particularly interesting. Therefore, we leave them out of our discussion.

2.2.3 *A Selection of Axioms*

In the following, we recall some well-known axioms of Social Choice Theory, adapted to our model of anonymous SCFs. Later, these axioms will be used in our experiments.

First, let us introduce some additional notation. Given a profile $\mathbf{R}$, recall that $\mathbf{R}(\succ)$ is the number of voters that express the preference $\succ$. With a slight abuse of notation, given a profile $\mathbf{R}$ and two alternatives x and y, we define as $\mathbf{R}(x \succ y)$ the number of voters who prefer x over y in $\mathbf{R}$:

$$\mathbf{R}(x \succ y) = \sum_{\substack{\succ \in \mathcal{L}(X): \\ x \succ y}} \mathbf{R}(\succ)$$

We will now present a collection of axioms for an SCF F defined over a scenario $\langle n^\star, X \rangle$. Note that all of these axioms, except for POSITIVE RESPONSIVENESS, were already employed in the experiments of Boixel and Endriss (2020).

- NEUTRALITY. This axiom requires that all alternatives should be treated equally, that is, the names of the alternatives should not matter. Given a pair of profiles $\mathbf{R}$ and $\mathbf{R}'$, if there is a permutation $\pi : X \to X$ such that $\mathbf{R} = \pi(\mathbf{R}')$, then $F(\mathbf{R}) = \pi(F(\mathbf{R}'))$ must hold as well.[4]

- PARETO PRINCIPLE. For two alternatives $x, y \in X$, if all voters prefer x to y, then y cannot win: if, for some $x, y \in X$, it holds that $|\mathbf{R}| = \mathbf{R}(x \succ y)$, then $y \notin F(\mathbf{R})$.

- FAITHFULNESS. If there is only one voter, then their top-ranked choice should win: if $|\mathbf{R}| = 1$ and $\succ \in \mathbf{R}$, then $F(\mathbf{R}) = \{\mathrm{top}(\succ)\}$.

- CANCELLATION. If all alternatives tie in a pairwise majority contest, then all alternatives should win: if, for all alternatives x and y, it holds that $\mathbf{R}(x \succ y) = \mathbf{R}(y \succ x)$, then $F(\mathbf{R}) = X$.

- REINFORCEMENT. If two profiles $\mathbf{R}_1$ and $\mathbf{R}_2$ have some winners in common, then if we combine the two profiles in a single profile $\mathbf{R}$, these alternatives should win in $\mathbf{R}$ (also known as "Consistency"): if $F(\mathbf{R}_1) \cap F(\mathbf{R}_2) \neq \varnothing$ and $|\mathbf{R}_1 + \mathbf{R}_2| \leqslant n^\star$, then $F(\mathbf{R}_1 + \mathbf{R}_2) = F(\mathbf{R}_1) \cap F(\mathbf{R}_2)$.[5]

- CONDORCET PRINCIPLE. If there is an alternative $x^\star$ such that, for all alternatives $y \in X \setminus \{x^\star\}$, a majority of voters prefer $x^\star$ to y, then $x^\star$ should be the sole winner: if for all $y \in X \setminus \{x^\star\}$ it holds that $\mathbf{R}(x^\star \succ y) > |\mathbf{R}|/2$, then $F(\mathbf{R}) = \{x^\star\}$.

4 Given a permutation $\pi : X \to X$ and a profile $\mathbf{R}$, we denote as $\pi(\mathbf{R})$ the profile obtained by substituting every alternative x mentioned in $\mathbf{R}$ with $\pi(x)$. Similarly, given a set of alternatives $X' \subseteq X$, we indicate by $\pi(X')$ the set $\{\pi(x) \mid x \in X'\}$.

5 The restriction $|\mathbf{R}_1 + \mathbf{R}_2| \leqslant n^\star$ ensures that we only speak about profiles relevant to the current scenario, $\langle n^\star, X \rangle$. This will be discussed later on.

- Positive Responsiveness. This axiom requires that if we increase the support of a winner, then this alternative should become the sole winner. If a profile $\mathbf{R}^+$ can be obtained from a profile $\mathbf{R}$ by raising the support of an $\mathbf{R}$-winner $x^\star$, then $x^\star$ must be the sole winner in $\mathbf{R}^+$: if, for two *distinct* profiles $\mathbf{R}$ and $\mathbf{R}^+$ it holds that $\mathbf{R}(y \succ z) = \mathbf{R}^+(y \succ z)$ for all $y, z \in X \setminus \{x^\star\}$, and that $\mathbf{R}(x^\star \succ y) \leqslant \mathbf{R}^+(x^\star \succ y)$ for all $y \in X \setminus \{x^\star\}$, then $x^\star \in F(\mathbf{R})$ implies $F(\mathbf{R}^+) = \{x^\star\}$.

Notice that all of these axioms fit the informal category of universal axioms. That is, they all have the following form: "for all profiles that satisfy a certain *condition*, a certain *constraint* must hold over their outcomes". The choice of focusing on this kind of axioms for our experiments will be discussed later (Section 3.4.5). For now, we stress that, despite this choice, the approach that we are going to propose works for *any* kind of axiom.

Moreover, notice that, as already discussed by Boixel and Endriss (2020), our version of Reinforcement is weaker than the one commonly adopted in the Social Choice literature, as introduced by Young (1974). Indeed, the classical formulation of Reinforcement refers to *any* possible electorate of *any* size, whereas we restrict attention to profiles in the voting scenario (that is, with at most $n^\star$ voters). The rationale behind this choice is two-fold. First, recall that we aim at proposing an algorithm for the automatic justification of voting outcomes. In particular, we want to find a justification for why a certain outcome should win in a given profile (the "goal profile"). As we will see, these justifications are composed of arguments based on a certain set of axioms. Thus, a justification based on the classical formulation of Reinforcement might deal with profiles that are bigger than the goal profile. Such a justification, as argued by Boixel and Endriss, might be of limited explanatory power. Indeed, it would require agents to reason about what would happen if more voters would join, and this might be puzzling to some: after all, why should *they* care about what would happen in that case? Secondly, having an axiom that involves an unbounded number of profiles would be a challenge from the computational point of view. This will be made clear later, when the details of the algorithm will be described; for now, consider that (in our approach) to search for a justification one has to explore the space of all profiles. If this space is infinite, then, when no justification exists, the search could (in principle) go on forever.

2.2.4 *Axiom Instances*

Above, we have introduced axioms as conditions that an SCF might or might not satisfy, and thus as something we can use to argue about voting rules. More concretely, axioms can also be used to argue for or against a particular *outcome*, which is the goal of this work. In principle, an argument in favour of a certain outcome $X^\star \subseteq X$ in a profile $\mathbf{R}^\star$, based on some axioms $\mathcal{A}$, could be the following: "every SCF F that satisfies axioms $\mathcal{A}$ elects $X^\star$ in $\mathbf{R}^\star$; therefore, if you accept $\mathcal{A}$, then you must accept that $X^\star$ wins in $\mathbf{R}^\star$".

Although valid, this argument is of limited explanatory power, as it does not explain "why", according to the principles expressed by $\mathcal{A}$, the set $X^\star$ is the "right" outcome. Rather, it is an abstract argument about SCFs in general. To understand what a more concrete argument could look like, consider the following example, based on NEUTRALITY.

Example 2.2. Let $\mathbf{R}$ be the following profile:

$$\begin{aligned} \#1 &: a \succ b \succ c \\ \#1 &: b \succ c \succ a \\ \#1 &: c \succ a \succ b \end{aligned}$$

Notice that this profile is, in some sense, symmetric: each alternative is ranked first, second and third exactly once. Thus, the three alternatives differ only in name, since they cannot be distinguished from the structure of the profile alone.[6] By NEUTRALITY, then, we must have that the three alternatives tie, since we cannot make any distinction based on their names.

More formally, consider the permutation $\pi : X \to X$ such that $\pi(a) = b$, $\pi(b) = c$ and $\pi(c) = a$. Notice that $\mathbf{R} = \pi(\mathbf{R})$. Therefore, by NEUTRALITY, it must hold that $F(\mathbf{R}) = \pi(F(\mathbf{R}))$. But the only non-empty outcome $X^\star \subseteq X$ where $X^\star = \pi(X^\star)$ holds is X itself. Thus, we must have that $F(\mathbf{R}) = \{ a, b, c \}$.[7] △

Notably, instead of referring to the full axiom of NEUTRALITY, we only referred to one particular *instance* of it. That is, we instantiated the quantified variables to some specific objects (relevant to the current problem).

The justifications we will automatically construct will consist of concrete and specific arguments, similar to the one exemplified above. Thus, the notion of axiom instance is at the core of our approach. For now, we gave an intuitive example of what an instance is; later on, as we will introduce a language to speak about axioms, we will make this notion precise. Regardless, following the notation employed by Boixel and Endriss (2020), given an axiom A, we write $A' \lhd A$ if A' is an instance of A. Similarly, for a set $\mathcal{A}'$, we write $\mathcal{A}' \lhd A$ if every $A' \in \mathcal{A}'$ is an instance of A. Furthermore, given a set of axioms $\mathcal{A}$, we write $A' \lhd \mathcal{A}$ if A' is an instance of some $A \in \mathcal{A}$. Finally, we write $\mathcal{A}' \lhd \mathcal{A}$ if every $A' \in \mathcal{A}'$ is an instance of some $A \in \mathcal{A}$.

6 To see why, imagine that you have to refer to one of these alternatives without mentioning its name. How would you distinguish it from the other two? Notice that, for example, you cannot refer to a as "the alternative ranked first in the first ballot". This is because we are dealing with anonymous profiles: since multisets have no order, there is no "first ballot".

7 Notice that, if we were to work with non-anonymous profiles, NEUTRALITY would not be sufficient for this. Without going into the details, in that case, we could use the names of the voters to distinguish the alternatives (e.g., "the alternative ranked first in the ballot of *that* voter"). To argue for this outcome, we would also need the axiom of ANONYMITY, which states that the names of voters should not matter.

3 A GRAPH-BASED JUSTIFICATION ALGORITHM

In this chapter, we are going to present the main contribution of this work, namely, a graph-based algorithm for the automatic justification of group decisions. Firstly, we are going to recall the framework introduced by Boixel and Endriss (2020) to talk about justifications in Social Choice problems. Secondly, we are going to present a way to encode axioms, and then present the base algorithm introduced by the aforementioned authors. Thirdly, we are going to introduce our algorithm and prove its correctness. Lastly, we discuss two general families of heuristics for our approach.

3.1 AUTOMATIC JUSTIFICATION OF COLLECTIVE DECISIONS

In this section, we recall the background notions concerning justifications and justification problems. We also introduce *justification algorithms*.

3.1.1 *Justifications and Justification Problems*

In the following, we use the notion of *justification* and *justification problem* as introduced by Boixel and Endriss (2020). We will first give an intuitive account of what we mean by justification, and then introduce the formal definitions.

Intuitively, when we talk about a "justification" for some decision, we mean some kind of argument that explains why such decision is, as the name goes, *just* (that is, "good"). Generally speaking, such explanations may rest upon some normative principles that express in some way what a "good decision" should look like. Let us give a concrete example. Suppose Edward is driving a car and, suddenly, a dog jumps out of a fence, running in the middle of the road. Of course, Edward loves animals, and thus at the last moment steers violently, hitting and destroying an innocent shop window. A justification for this decision could be: "I had to crash into the window in order to save the dog". This justification relies upon the implicit assumption (a normative principle) that it is better to destroy a window rather than hurting an animal. Furthermore, it explains *why* Edward's decision was a consequence of this principle (assuming, for simplicity, that hurting the dog or crashing the window were the only two options). If the shop owner wanted to give a counterargument, then she would need to attack the norm Edward's decision rested upon.

Boixel and Endriss (2020) applied this idea of a justification (an explanation based on a set of normative principles) to the problem of collective decision making. More precisely, they formalise what constitutes a justification, grounded in a certain set of axioms, for why a particular outcome should win an election. Let us review their definition.

Definition 3.1 (Justification Problems and Justifications; Boixel and Endriss, 2020)**.** *For a given voting scenario $\langle n^\star, X \rangle$, let $\mathbf{R}^\star$ be a profile, $X^\star$ be a non-empty subset of X and $\mathbb{A}$ a set of axioms for SCFs of this scenario. We call such a triple $\langle \mathbf{R}^\star, X^\star, \mathbb{A} \rangle$ a* justification problem. *Further, a* justification *for $\langle \mathbf{R}^\star, X^\star, \mathbb{A} \rangle$ is a pair of sets of axioms $\langle \mathcal{A}^N, \mathcal{A}^E \rangle$ (with* normative basis *$\mathcal{A}^N$ and* explanation *$\mathcal{A}^E$) such that:*

(i) ***Explanatoriness****. $\mathcal{A}^E$ satisfies this if it* minimally explains *the outcome $X^\star$ in $\mathbf{R}^\star$, that is, (1) for any $F \in \mathbb{I}(\mathcal{A}^E)$ it holds that $F(\mathbf{R}^\star) = X^\star$ and (2) for any $\mathcal{A} \subset \mathcal{A}^E$ there is an $F \in \mathbb{I}(\mathcal{A})$ such that $F(\mathbf{R}^\star) \neq X^\star$. In other words, all SCFs that satisfy $\mathcal{A}^E$ elect $X^\star$ in $\mathbf{R}^\star$, and for no subset of $\mathcal{A}^E$ this holds.*

(ii) ***Relevance****. The explanation is an instance of the normative basis: $\mathcal{A}^E \lhd \mathcal{A}^N$.*

(iii) ***Adequacy****. The normative basis should be composed of axioms in the corpus: $\mathcal{A}^N \subseteq \mathbb{A}$.*

(iv) ***Non-triviality****. There should be at least one SCF that satisfies the normative basis: $\mathbb{I}(\mathcal{A}^N) \neq \varnothing$. If this holds, we say that $\mathcal{A}^N$ is* non-trivial.

Thus, intuitively, a justification for $\langle \mathbf{R}^\star, X^\star, \mathbb{A} \rangle$ is a justification (grounded in $\mathbb{A}$) in favour of outcome $X^\star$ (the *target outcome*) for profile $\mathbf{R}^\star$ (the *goal profile*). Furthermore, we call a justification $\langle \mathcal{A}^N, \mathcal{A}^E \rangle$ *minimal* if for all proper subsets $\mathcal{A} \subset \mathcal{A}^N$ of the normative basis it does not hold that $\mathcal{A}^E \lhd \mathcal{A}$. Intuitively, this means that all axioms in $\mathcal{A}^N$ are "used" in $\mathcal{A}^E$.

To return to our previous discussion of justifications, $\mathcal{A}^N$ contains the normative principles (axioms) which our justification appeals to, whereas $\mathcal{A}^E$ constitutes a concrete argument[1] showing that a certain outcome is the only one that satisfies $\mathcal{A}^N$ (that is, if you accept $\mathcal{A}^N$, then you must accept that $X^\star$ wins in $\mathbf{R}^\star$). Although we have yet to introduce the formal definitions of axioms and instances, let us give an intuitive example.

Example 3.1. Consider the voting scenario $\langle n^\star, X \rangle$, where $n^\star = 3$ and $X = \{ a, b, c \}$. Let the goal profile $\mathbf{R}^\star$ be:

$$\#2 : a \succ b \succ c$$
$$\#1 : c \succ b \succ a$$

Furthermore, let the target outcome $X^\star$ be $\{ a \}$. We can construct a justification $\langle \mathcal{A}^N, \mathcal{A}^E \rangle$ for $X^\star$ in $\mathbf{R}^\star$ as follows:

1 Technically, $\mathcal{A}^E$ is an (unstructured) set of smaller arguments that collectively explain $F(\mathbf{R}^\star) = X^\star$. However, to give a readable explanation, these need to be put together in some structured way. In this work, we focus on the problem of finding the explanations, and we leave the separate issue of presenting them to future work.

- $\mathcal{A}^N$ contains three axioms: Reinforcement, Cancellation and Faithfulness.
- $\mathcal{A}^E$ is consists of the following instances:
  - (Cancellation) In profile $\mathbf{R}_1 = \{ a \succ b \succ c, c \succ b \succ a \}$ (a subprofile of $\mathbf{R}^\star$) all alternatives tie in a pairwise comparison, thus $F(\mathbf{R}_1) = \{ a, b, c \}$ (we call such a profile a *Cancellation profile*).
  - (Faithfulness) Profile $\mathbf{R}_2 = \{ a \succ b \succ c \}$ (another subprofile of $\mathbf{R}^\star$) has a single voter, thus $F(\mathbf{R}_2) = \{ a \}$.
  - (Reinforcement) Profile $\mathbf{R}^\star$ is equal to $\mathbf{R}_1 + \mathbf{R}_2$, thus if $F(\mathbf{R}_1) \cap F(\mathbf{R}_2) \neq \varnothing$, then $F(\mathbf{R}^\star) = F(\mathbf{R}_1) \cap F(\mathbf{R}_2)$.

Clearly, $\mathcal{A}^E \lhd \mathcal{A}^N$, that is, every element of $\mathcal{A}^E$ is an instance of some axiom in $\mathcal{A}^N$. Moreover, notice that the three instances together imply that $F(\mathbf{R}^\star) = \{ a \}$, and that if we were to remove any one of the three, this would no longer hold. Thus, $\langle \mathcal{A}^N, \mathcal{A}^E \rangle$ is a justification for $X^\star$ in $\mathbf{R}^\star$. $\triangle$

Another thing to consider is the following. As Boixel and Endriss (2020) observe (Theorem 1 of their paper), we cannot justify two different outcomes for a given profile using the same normative basis. An immediate consequence of this (that we will use later on in our experiments) is the following:

Corollary 3.1. *Let $\mathbb{A}$ be a set of axioms and $\mathbf{R}^\star$ a profile defined over a set of alternatives X. If $\mathbb{A}$ is non-trivial, then we cannot justify two different outcomes $X_1 \subseteq X$ and $X_2 \subseteq X$ for $\mathbf{R}^\star$ using axioms from $\mathbb{A}$.*

Proof. Suppose there are two (distinct) sets of alternatives X_1 and X_2, subsets of X, such that $\langle \mathbf{R}^\star, X_1, \mathbb{A} \rangle$ and $\langle \mathbf{R}^\star, X_2, \mathbb{A} \rangle$ have justifications $\langle \mathcal{A}^N_1, \mathcal{A}^E_1 \rangle$ and $\langle \mathcal{A}^N_2, \mathcal{A}^E_2 \rangle$, respectively. Moreover, suppose that $\mathbb{A}$ is a non-trivial set of axioms. Then, clearly $\langle \mathbb{A}, \mathcal{A}^E_1 \rangle$ and $\langle \mathbb{A}, \mathcal{A}^E_2 \rangle$ are also justifications for the same justification problems, as justifications need not be minimal. But this contradicts Theorem 1 by Boixel and Endriss (2020), as we justified two different outcomes for $\mathbf{R}^\star$ using the same normative basis. Thus, we cannot justify more than one outcome using a non-trivial corpus of axioms. $\square$

3.1.2 *Justification Algorithms*

We now introduce *justification algorithms*, along with their core properties. We assume that the general notion of "algorithm" is known.

Definition 3.2 (Justification Algorithms)**.** *A* justification algorithm *is an algorithm that takes a justification problem as input and returns a (possibly empty) collection of pairs of sets of axioms as output. Furthermore, the following properties are properties that a justification algorithm might satisfy:*

- ***Soundness**: For any input $\mathcal{J}$, if the algorithm outputs the pair $\langle \mathcal{A}^N, \mathcal{A}^E \rangle$, then it is a justification for $\mathcal{J}$.*
- ***Completeness**: For any input problem $\mathcal{J}$ and for all sets of axioms $\mathcal{A}^N$, if a justification for $\mathcal{J}$ that uses $\mathcal{A}^N$ as a normative basis exists, then on input $\mathcal{J}$ the algorithm returns at least one justification with normative basis $\mathcal{A}$ (for some $\mathcal{A} \subseteq \mathcal{A}^N$).*
- ***Correctness**: A justification algorithm is correct if it is both sound and complete.*

Let us comment on our notion of completeness. Intuitively, this condition states that, for every possible normative basis, we need to return at least one justification that only uses axioms from this basis. An alternative (and stronger) definition of completeness could be the following: an algorithm is complete if it returns *every* possible justification for the input $\mathcal{J}$. We argue that this definition is not required for practical purposes, as our definition is already strong enough to satisfy the users. In fact, the normative basis is where two electorates could disagree on, based on which axioms they decide to accept. If we guarantee that we can find a justification for every (subset of a) normative basis that admits one, then we know that we can provide every electorate that could be satisfied by a certain outcome and a certain normative basis with at least a justification that they like. The fact that we find only one explanation per normative basis (or rather, we are guaranteed to find at least one) is non-critical in this regard. Indeed, the role of an explanation is only to explain *why* the outcome is the only one satisfying the normative basis. Moreover, two explanations grounded in the same normative basis will generally be quite similar, as they appeal to the same normative principles.[2]

The fact that we only require to find a subset of any possible normative basis is not problematic either. Indeed, if one would accept the normative basis $\mathcal{A}^N$, then one would accept every normative basis $\mathcal{A} \subseteq \mathcal{A}^N$, as it requires subscribing to the same or fewer axioms as $\mathcal{A}^N$. Notice that, by allowing this, we can design an algorithm that returns only *minimal* justifications. For if $\langle \mathcal{A}^N, \mathcal{A}^E \rangle$ is a justification for some justification problem $\mathcal{J}$, then there is a minimal justification $\langle \mathcal{A}, \mathcal{A}^E \rangle$ for $\mathcal{J}$ with $\mathcal{A} \subseteq \mathcal{A}^N$. For instance, we can remove axioms from $\mathcal{A}^N$ one-by-one, and stop as soon as the resulting normative basis $\mathcal{A}$ leads to a minimal justification. Thus, returning $\langle \mathcal{A}, \mathcal{A}^E \rangle$ would be enough to satisfy our definition of completeness (with respect to $\mathcal{A}^N$).

3.2 A LANGUAGE FOR AXIOMS

As it will be made clear later, in both our approach and the approach proposed by Boixel and Endriss (2020), part of the computational task is delegated to some external state-of-the-art reasoning tools. Thus, we need to encode axioms and axiom instances

2 Although this needs not to be true in general, we found this to be true for almost all of the experiments we ran.

in such a way that the reasoner can understand.[3] Consequently, the choice of the encoding language has a direct impact on the performance of the algorithm, as it constrains the choice of which tools we can use. Since we are interested in building a practical algorithm, this is crucial.

In their experiments, Boixel and Endriss (2020) employ Constraint Satisfaction Problems (CSPs) (Rossi, Van Beek, and Walsh, 2006) as a formalism. Instead, we will use propositional logic. Propositional logic has been used with success in various applications of Automated Reasoning to Computational Social Choice (see the review by Geist and Peters (2017)). Moreover, there is an active and lively research community devoted to the development and improvement of high-performing propositional reasoning tools (see, for example, the "SAT Competition").[4] Encoding axioms in this language allows us to employ these SAT solvers out of the box. Finally, encoding choice-theoretic axioms in propositional logic is relatively straightforward, especially given the various examples of such encodings to be found in the literature (as mentioned above). Thus, we choose this formalism for our experiments.

3.2.1 *The Propositional Language*

In this section, we present the background notions used to define our language. Fix any indexing of the profiles $\mathcal{R}(X)^+ = \{\mathbf{R}_1, \ldots, \mathbf{R}_\ell\}$ and of the alternatives $X = \{x_1, \ldots, x_m\}$. Our language is made of propositional variables p_{ij}, where i ranges over all profiles and j over all alternatives. Intuitively, p_{ij} indicates that in $\mathbf{R}_i$ alternative x_j wins.

We adopt the standard machinery of propositional logic: literals, connectives, assignments, conjunctive normal form (CNF), and so on (see the chapter by Lifschitz, Morgenstern, and Plaisted (2008) for an overview of these concepts). We mostly consider formulae in CNF, usually referred to with the variable φ. With a slight abuse of notation, we occasionally treat CNF formulae like sets: a CNF formula is a set of clauses, and a clause is a set of literals.

Fix a voting scenario $\langle n^\star, X\rangle$. Let $\mathcal{V}$ be the set of all relevant propositional variables (that is, all p_{ij} such that $\mathbf{R}_i \in \mathcal{R}(X)^+$ and $x_j \in X$). For every SCF F over $\langle n^\star, X\rangle$, there is a corresponding assignment α_F over $\mathcal{V}$ such that:

For every $p_{ij} \in \mathcal{V}$, $\alpha_F(p_{ij}) = 1$ if and only if $x_j \in F(\mathbf{R}_i)$.

Through this notion, we can define precisely the *interpretation* of a formula φ as a set $\mathbb{I}(\varphi)$ of SCFs such that $F \in \mathbb{I}(\varphi)$ if and only if $\alpha_F \models \varphi$. That is, for a given formula φ,

3 Recall that (the explanation of) a justification is composed of axiom instances. In brief, to search for a justification for $\langle \mathbf{R}^\star, X^\star, \mathbb{A}\rangle$, we need first to generate the instances of the axioms in $\mathbb{A}$, and then use said reasoning tools to find the explanations among the generated instances.

4 http://www.satcompetition.org/

its interpretation is the set of SCFs whose corresponding assignments make φ true.[5] This definition is useful, as we will encode axioms through propositional formulae; thus, it translates directly to a precise notion of axiom interpretation. In other words, if we encode an axiom A through a CNF formula φ, we will use the interpretation of φ to define the interpretation of A.

Usually, one real-world constraint can only be represented as a complex (that is, multi-clause) CNF formula, rather than with one single clause. This is true for the instances of most axioms we will use in our work. Recall that the notion of instance is at the core of the definition of justification we employ. Consequently, it seems reasonable to treat the clauses that arise from the same instance as a single object that can be manipulated. To this end, we introduce the notion of a group CNF (gCNF). A formula in gCNF is a *finite* set of CNF formulae.[6] Intuitively, this object corresponds to a CNF formula where the clauses that arise from the same constraint (in our case, the same instance) are grouped together. We will see in the next section how this can be used to encode axioms; now, we discuss some more background notions regarding gCNF formulae.

We will usually refer to a gCNF formula with the variable Φ. Given a gCNF formula Φ, we define the CNF formula φ_Φ given by the concatenation of all formulae in Φ as:[7]

$$\varphi_\Phi = \bigcup_{\varphi \in \Phi} \varphi$$

A gCNF formula Φ is satisfiable (respectively, unsatisfiable) if φ_Φ is (respectively, is not) satisfiable. Likewise, the interpretation of a gCNF formula Φ is defined as $\mathbb{I}(\Phi) = \mathbb{I}(\varphi_\Phi)$.

The last notion that we need is the notion of *group minimal unsatisfiable set*. Let us first recall the standard notion of *minimal unsatisfiable set* (MUS) in propositional logic. Intuitively, we can think of a MUS as an "unsatisfiable nucleus" of φ. Formally, a MUS of a CNF formula φ is a formula $\varphi' \subseteq \varphi$ such that φ' is unsatisfiable and none of its proper subsets are.

Likewise, a group minimal unsatisfiable set (gMUS) of a formula in gCNF Φ is a gCNF formula $\Phi' \subseteq \Phi$ such that Φ' is unsatisfiable and none of its proper subsets are. So, instead of removing clauses one-by-one, we remove all clauses of a given formula at the same time. This notion is useful, as it has a direct connection to the notion of

5 Note that not all assignments have a corresponding SCF. For instance, the assignment α_0 such that for every p it holds that $\alpha_0(p) = 0$ corresponds to no SCF. This is because such an SCF would violate the requirement that at least an alternative wins in all profiles. We will show how to deal with this problem later on.

6 The requirement of this to be finite will be discussed later.

7 Notice that we use the set-theoretic view of CNF formulae for this definition. This is not problematic, as this concatenation is used only to define the semantics (that it, satisfiability and interpretation) of a gCNF formula. Having some redundant clauses has no impact on this.

explanation we use. This will be made clear later (Theorem 3.2), but first, we will show how to encode axioms in our language.

3.2.2 *Encoding Axioms and Instances*

We will encode axioms as formulae in gCNF. Further, we will encode axiom instances as *singleton* gCNF formulae, that is, gCNF formulae containing a single CNF formula.[8] Given an axiom (gCNF formula) A, we say that A' is an instance of A (written $A' \lhd A$) if $A' \subseteq A$ and $|A'| = 1$. Thus, every CNF formula $\varphi \in A$ corresponds to some instance A' (namely, $A' = \{\varphi\}$), and an axiom can be seen as a "collection" of its instances.

Notice that this definition complies with the three requirements laid down by Boixel and Endriss (2020):

(i) *Every instance A' of an axiom A must itself be an axiom.* Axioms are formulae in gCNF, and instances are (singleton) gCNF formulae; thus, every instance is an axiom.

(ii) *The interpretation of an axiom is always equal to the intersection of the interpretations of all its instances.* This requirement is also met; recall that an axiom A is encoded as a gCNF. Given our definition of the interpretation of a gCNF, an SCF F satisfies A if and only if it satisfies all the CNF formulae $\varphi \in A$. But since there is an instance $A' = \{\varphi\}$ for every such φ, we have that F satisfies A if and only if F satisfies all instances $A' \lhd A$.

(iii) *The number of instances of any given axiom A is finite.* This follows naturally from the fact that a formula in gCNF is finite.[9]

Furthermore, given a set of axioms $\mathcal{A}$, we will indicate by $\Phi_{\mathcal{A}}$ the gCNF formula obtained by "unrolling" all axioms into a single set (this notation will be used when presenting the algorithm):

$$\Phi_{\mathcal{A}} = \{\varphi \mid \varphi \in A, \text{ for some } A \in \mathcal{A}\}$$

Let us show how to encode axioms within this language. As a starting example, consider the following. In our approach, we must explicitly encode the requirement that at least one winner must be elected for every profile (that is, for every $\mathbf{R}$ we must

8 A more natural view could be to define axioms as formulae in gCNF and instances as formulae in CNF. However, we stick to the current definition to comply with the requirements for axiom instances stipulated by Boixel and Endriss (2020), as discussed later.

9 Notice that the restriction on the finiteness of gCNF formulae is not problematic in our setting, as we are interested in axioms that restrict only the profiles of some specific voting scenario $\langle n^*, X \rangle$ (which are, indeed, finite). Our language is powerful enough to encode all axioms in Section 2.2.3, that are of this kind; still, one could imagine an axiom that restricts an infinite number of profiles (such as the classical formulation of REINFORCEMENT by Young (1974)).

have that $F(\mathbf{R}) \neq \varnothing$). This is to make sure that we only consider assignments that have a corresponding SCF when testing for the satisfiability of a gCNF. We can indeed represent this as a separate axiom, that we call AtLeastOne, and encode it as a gCNF. An instance of it, regarding a profile $\mathbf{R}_i$, can be encoded as the singleton gCNF formula LEAST($\mathbf{R}_i$):

$$\text{LEAST}(\mathbf{R}_i) = \left\{ \bigvee_{x_j \in X} p_{ij} \right\}$$

This states that at least one alternative must win in $\mathbf{R}_i$. The full axiom can then be encoded as the gCNF formula LEAST, composed of one instance per profile in the scenario:

$$\text{LEAST} \quad = \bigcup_{\mathbf{R} \in \mathcal{R}(X)^+} \text{LEAST}(\mathbf{R})$$

This axiom is always encoded in our corpus and is always assumed to be part of the corpora we use for our experiments.

More generally, our language is expressive enough to encode all the axioms shown in Section 2.2.3. We will now present two examples.

Example 3.2. Recall the axiom of Faithfulness: if a profile contains only one vote, then the top-preference of that vote should be the unique winner. Let $\mathbf{R}_i$ be a profile with $|\mathbf{R}_i| = 1$. Further, with a slight abuse of notation, let $\text{top}(\mathbf{R}_i) = x_j$ be the top alternative of the unique preference order contained in $\mathbf{R}_i$. We can encode the corresponding instance of Faithfulness as the singleton gCNF formula FAI($\mathbf{R}_i$, x_j):

$$\text{FAI}(\mathbf{R}_i, x_j) = \left\{ \left(\bigwedge_{x_k \in X \setminus \{x_j\}} \neg p_{ik} \right) \wedge p_{ij} \right\}$$

This states that no alternative should win in $\mathbf{R}_i$ save for x_j. We can encode the full axiom as the gCNF formula FAI combining the above formula for each single-vote profile:

$$\text{FAI} \quad = \bigcup_{\mathbf{R} \in \mathcal{R}(X)^1} \text{FAI}(\mathbf{R}, \text{top}(\mathbf{R})) \qquad \triangle$$

Example 3.3. Consider the axiom of Positive Responsiveness: if we increase the support of a (possibly tied) winner $x^\star$, then $x^\star$ must become the sole winner. Let $\mathbf{R}_i$ and $\mathbf{R}_j$ be two profiles, where $\mathbf{R}_j$ has been obtained from $\mathbf{R}_i$ by raising alternative x_k in at least one voter's ranking. We can encode the corresponding instance with the singleton gCNF formula $\text{POS}(\mathbf{R}_i, \mathbf{R}_j, x_k)$:

$$\text{POS}(\mathbf{R}_i, \mathbf{R}_j, x_k) = \left\{ \left(\bigwedge_{x_\ell \in X \setminus \{x_k\}} (\neg p_{ik} \vee \neg p_{j\ell}) \right) \wedge (\neg p_{ik} \vee p_{jk}) \right\}$$

This states that, for any alternative $x_\ell \neq x_k$, if x_k wins in $\mathbf{R}_i$ then x_ℓ cannot win in $\mathbf{R}_j$ (recall that $\neg p \vee q$ is equivalent to $p \implies q$). Further, if x_k wins in $\mathbf{R}_i$, then it must also win in $\mathbf{R}_j$. Again, we can encode the full axiom as the gCNF formula POS:

$$\text{POS} \quad = \bigcup_{(\mathbf{R}, \mathbf{R}', x^\star) \in P} \text{POS}(\mathbf{R}, \mathbf{R}', x^\star)$$

Here, $P \subseteq \mathcal{R}(X)^{n^\star} \times \mathcal{R}(X)^{n^\star} \times X$ is a set such that $(\mathbf{R}, \mathbf{R}', x^\star) \in P$ if and only if $\mathbf{R}'$ can be obtained from $\mathbf{R}$ by raising the support of $x^\star$ in at least one preference order. $\triangle$

3.3 FULL GMUS-BASED ALGORITHM

In this section, we recall the base algorithm given by Boixel and Endriss (2020) to automatically compute justifications, expressed with the machinery of our propositional language. We will build our algorithm upon this approach.

Intuitively, the algorithm works as follows. Let $\mathcal{J} = \langle \mathbf{R}^\star, X^\star, \mathbb{A} \rangle$ be the input problem. First, we encode all axioms in $\mathbb{A}$ as a single gCNF formula (**generation step**), plus a formula that encodes the fact that $X^\star$ *must not be* the outcome for $\mathbf{R}^\star$ (called the *goal formula*). We call this whole gCNF formula $\Phi_{\mathcal{J}}$. Then, we use an external reasoning tool to extract the gMUSes of $\Phi_{\mathcal{J}}$ (**solving step**). As we will show, the gMUSes containing the goal formula correspond to the explanations of $\mathcal{J}$. Intuitively, this holds because such a gMUS is a minimal set of axiom instances that enforces $F(\mathbf{R}^\star) = X^\star$. Indeed, if we were to remove the goal formula, such a set of instances would become satisfiable (by virtue of it being a gMUS): this means that the target outcome is the only possible outcome in $\mathbf{R}^\star$. Similarly, if we were to remove any other axiom instance, we would again obtain a satisfiable gCNF, and we could find an SCF F which satisfies this smaller formula that has $F(\mathbf{R}^\star) \neq X^\star$ (as the goal formula requires this).[10]

10 For simplicity, we omitted the problem of non-triviality from this presentation. It will be addressed in the sequel.

3.3.1 *Encoding Justification Problems*

We can encode a justification problem $\mathcal{J} = \langle \mathbf{R}^\star, X^\star, \mathbb{A} \rangle$ in a gCNF formula $\Phi_{\mathcal{J}}$ as follows. Let the goal formula be:

$$\varphi_{\mathrm{Goal}} = \left(\bigvee_{x_j \in X^\star} \neg p_{gj} \right) \vee \left(\bigvee_{x_j \in X \setminus X^\star} p_{gj} \right)$$

Here, g indicates the index of the goal profile, $\mathbf{R}^\star$. This formula expresses the fact that $F(\mathbf{R}^\star) \neq X^\star$ must hold. Then:

$$\Phi_{\mathcal{J}} = \Phi_{\mathbb{A}} \cup \{ \varphi_{\mathrm{Goal}} \}$$

This gCNF formula corresponds to all instances of all axioms in the corpus $\mathbb{A}$, plus the goal formula. Notice that an SCF F is in $\mathbb{I}(\Phi_{\mathcal{J}})$ if and only if it satisfies all axioms in $\mathbb{A}$, while at the same time electing an outcome different from $X^\star$ in $\mathbf{R}^\star$.

3.3.2 *gMUS Algorithm*

We will now make the connection between a gMUS of $\Phi_{\mathcal{J}}$ and a justification for $\mathcal{J}$ precise. Intuitively, we will show that a gMUS of $\Phi_{\mathcal{J}}$ corresponds to a justification for a justification problem $\mathcal{J}$. As such, this theorem guarantees the **correctness** of the algorithm introduced later in this section. It is a variant of Theorem 2 as proved by Boixel and Endriss (2020), but expressed with the machinery of gCNFs and gMUSes.

Theorem 3.2 (Correctness of the gMUS Algorithm; Boixel and Endriss, 2020)**.** *Let* $\mathcal{J} = \langle \mathbf{R}^\star, X^\star, \mathbb{A} \rangle$ *be a justification problem. Then a pair of sets of axioms* $\langle \mathcal{A}^N, \mathcal{A}^E \rangle$ *is a justification for* $\mathcal{J}$ *if and only if the following conditions are satisfied:*

(i) $\Phi_{\mathcal{A}^E} \cup \{ \varphi_{\mathrm{Goal}} \}$ *is a gMUS of* $\Phi_{\mathcal{J}}$*;*

(ii) $\mathcal{A}^N \subseteq \mathbb{A}$ *and* $\mathcal{A}^E \triangleleft \mathcal{A}^N$*;*

(iii) $\Phi_{\mathcal{A}^N}$ *is satisfiable.*

Proof. The proof of this theorem mirrors the proof of Theorem 2 by Boixel and Endriss (2020). Indeed, the notions of gMUS for a gCNF formula and MUS for a CSP as used by the authors are essentially the same. Furthermore, the encoding we used in this setting is virtually equivalent to the encoding used in the original paper (a CNF formula in our encoding corresponds to a constraint in the CSP network). $\square$

The practical importance of this theorem is that it guarantees **correctness** of the following justification algorithm, that we call Justify. Given a problem $\mathcal{J} = \langle \mathbf{R}^\star, X^\star, \mathbb{A} \rangle$, the execution of Justify($\mathcal{J}$) evolves as follows:

1. Generate $\Phi_{\mathcal{J}}$. While doing so, keep track of which axiom $A \in \mathbb{A}$ and which instance A' gave rise to each formula $\varphi_{A'} \in \Phi_{\mathcal{J}}$;
2. Check whether $\Phi_{\mathcal{J}}$ is satisfiable. If it is, stop and announce that justifying $X^\star$ is impossible.[11] If it is not,[12] repeat the steps below until no new gMUS can be computed:
   (2a) Compute a new gMUS $\Phi^\star$ of $\Phi_{\mathcal{J}}$ such that $\varphi_{\mathsf{Goal}} \in \Phi^\star$. Define $\mathcal{A}^E$ as the set of instances corresponding to $\Phi^\star \setminus \{\varphi_{\mathsf{Goal}}\}$;
   (2b) For every $\mathcal{A}^N$ such that $\mathcal{A}^N \triangleleft \mathcal{A}^E$ and for no $\mathcal{A} \subset \mathcal{A}^N$ the same holds,[13] if $\Phi_{\mathcal{A}^N}$ is satisfiable,[14] return the justification $\langle \mathcal{A}^N, \mathcal{A}^E \rangle$ (and continue the execution).

This is essentially the algorithm proposed by Boixel and Endriss (2020). This algorithm is **sound**, as we only return a pair of sets of axioms that satisfy the three conditions of Theorem 3.2. Further, provided that our reasoning tool enumerates all the gMUSes, it is **complete**. Indeed, for any justification $\langle \mathcal{A}^N, \mathcal{A}^E \rangle$, we will surely return a justification with explanation $\mathcal{A}^E$ (as by looking at the gMUSes of $\Phi_{\mathcal{J}}$ we find all explanations) and normative basis $\mathcal{A} \subseteq \mathcal{A}^N$ (as we return all minimal justifications that have $\mathcal{A}^E$ as the explanation). Therefore, this algorithm is **correct**.

Notice that, at Step 1, we refer to the generation of $\Phi_{\mathcal{J}}$. This step (**generation step**) is the true bottleneck of the algorithm, computationally speaking. Indeed, since we delegate the reasoning tasks (that is, the checks for satisfiability and the gMUSes extraction) to state-of-the-art SAT solvers, Step 2 (**solving step**) can usually be handled relatively quickly (compared to the first step).

To address this, we are now going to propose our main contribution: an algorithm based on a different generative approach, able to interleave the generation and solving phases in a principled way. In general, this algorithm can find a justification by looking only at a portion of the full gCNF formula $\Phi_{\mathcal{J}}$, while still being guaranteed to find all possible justifications.

11 As this means that there is an SCF satisfying all axioms that elects some outcome different than $X^\star$ in $\mathbf{R}^\star$.

12 Conversely, this means that either $\mathbb{A}$ is trivial (no SCF satisfies $\mathbb{A}$) or that some non-trivial subset of axioms $\mathcal{A} \subseteq \mathbb{A}$ enforces that all $F \in \mathbb{I}(\mathcal{A})$ must have $F(\mathbf{R}^\star) = X^\star$. Notice that both might be true.

13 With the second condition, we guarantee that we only return minimal justifications. Further, note that, as already noticed by Boixel and Endriss (2020), if for every instance $A' \triangleleft \mathbb{A}$ there is only one axiom $A \in \mathbb{A}$ such that $A' \triangleleft A$, then for every explanation $\mathcal{A}^E$ we can only have one possible (minimal) normative basis $\mathcal{A}^N$ such that $\mathcal{A}^E \triangleleft \mathcal{A}^N$. Finally, note that since there are only $2^{|\mathbb{A}|} - 1$ total normative bases (the number of non-empty subsets of $\mathbb{A}$), with small corpora this search does not significantly impact the performance of this algorithm.

14 This check is necessary to guarantee **non-triviality** for $\mathcal{A}^N$.

3.4 GRAPH-BASED APPROACH

In this section, we introduce our main contribution, a graph-based algorithm for the automatic justification of collective decisions. As said above, we mainly address the **generation step** of the algorithm. The main idea behind our algorithm is the following. Instead of generating the whole set of instances directly, we procedurally generate them, starting from the instances that mention profiles that are in some sense the "closest" to the goal profile. This notion of closeness will be made precise as we introduce *instance graphs*, but intuitively a profile is close to the goal profile if it is mentioned in an instance that also mentions the goal profile or some profile close to it.[15] The main advantage is that, if a justification is found only by using instances that mention the profiles close to the goal profile, then we can avoid generating the whole set of instances.

3.4.1 *Instance Graphs*

An instance graph is an undirected multi-hypergraph of profiles and instances. The *multi* part means that there can be multiple edges between nodes (as there could be multiple instances connecting the same profiles), whereas the *hyper* part means that an edge is not restricted to connect only two nodes (as some axioms mention more than two profiles). This sounds rather clumsy, but it is just a fancy way to describe a set of overlapping sets and inherit the notions of path, connectedness and such.

Before going further, let us define some additional notation. Recall that an axiom instance is a singleton gCNF formula (a set containing a single CNF formula). We say that $\mathbb{P}(A')$ is the set of profiles mentioned in an instance A', i.e., $\mathbf{R} \in \mathbb{P}(A')$ if and only if there is some clause in φ (with $A' = \{\varphi\}$) that contains a literal referring to $\mathbf{R}$.

We will now introduce the notion of *instance graphs*. Intuitively, this object corresponds to a structured view on a set of instances $\mathcal{A}'$. Indeed, note that, in the following definition, an instance graph is fully determined by the instances it is composed of.

Definition 3.3 (Instance Graphs)**.** *An* instance graph *is a pair $\mathcal{G} = \langle \mathcal{P}, \mathcal{A}' \rangle$, where:*

- *$\mathcal{A}'$ is a set of axiom instances (or* edges*). We say that an instance $A' \in \mathcal{A}'$ connects any two profiles in $\mathbb{P}(A')$.*
- *$\mathcal{P}$ is a finite set of profiles (or* nodes*). We require that $\mathcal{P} = \bigcup_{A' \in \mathcal{A}'} \mathbb{P}(A')$. That is, the set of nodes is the set of all profiles that are mentioned in at least one instance.*

Note that we can see an instance as an edge between the profiles it mentions. Further, we refer to elements of $\mathcal{A}'$ both as instances and as edges, depending on the context. To clarify this definition, consider the following example.

15 Formally, this notion of closeness is not well-founded, but it is intended here as a mere intuition.

Example 3.4. Let $\mathbf{R}_1$ be the following profile:

$\#2 : a \succ b \succ c$

Moreover, let $\mathbf{R}_2$ be:

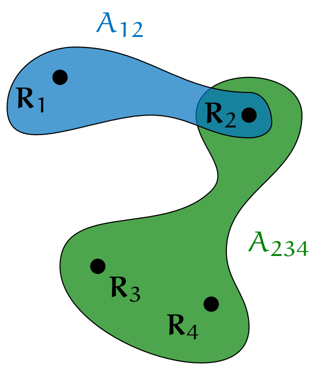
$\#1 : a \succ b \succ c$

$\#1 : b \succ a \succ c$

Notice that we can obtain $\mathbf{R}_1$ by raising the support of a in $\mathbf{R}_2$. Therefore, there is an instance of Positive Responsiveness (let it be A_{12}) connecting $\mathbf{R}_1$ and $\mathbf{R}_2$.

Now, consider the two profiles $\mathbf{R}_3 = \{ a \succ b \succ c \}$ and $\mathbf{R}_4 = \{ b \succ a \succ c \}$. Note that $\mathbf{R}_2 = \mathbf{R}_3 + \mathbf{R}_4$. Hence, there is an instance of Reinforcement that mentions $\mathbf{R}_2$, $\mathbf{R}_3$ and $\mathbf{R}_4$. Let this instance be A_{234}.

Finally, consider the instance graph $\mathcal{G} = \langle \mathcal{P}, \mathcal{A}' \rangle$, where $\mathcal{P} = \{ \mathbf{R}_1, \mathbf{R}_2, \mathbf{R}_3, \mathbf{R}_4 \}$ and $\mathcal{A}' = \{ A_{12}, A_{234} \}$. This instance graph can be represented as follows:



Here, note that the instance of Reinforcement connects three different profiles. △

We say that $\mathcal{G}_{\mathbb{A}} = \langle \mathcal{P}, \mathcal{A}' \rangle$ is the *corresponding instance graph* of $\mathbb{A}$ if $\mathcal{A}'$ is the set of all instances of $\mathbb{A}$ (that is, $\mathcal{A}' = \{ A' \mid A' \lhd \mathbb{A} \}$). Again, notice that all the information is already contained within $\mathbb{A}$; indeed, the instance-graph $\mathcal{G}_{\mathbb{A}}$ is just a structured view on the instances, to simplify the presentation of the upcoming generation algorithm. Let us now review some basic graph-theoretic notions.

A *path* is a sequence of edges $(A_1 \dots A_k)$ such that for every $i \in \{1, \dots, k-1\}$, it holds that $\mathbb{P}(A_i) \cap \mathbb{P}(A_{i+1}) \neq \varnothing$. We say that such a path is of length k.

We say that two nodes $\mathbf{R}$ and $\mathbf{R}'$ are connected if there is path between them, that is, if there is a path $(A_1 \dots A_k)$ such that $\mathbf{R} \in \mathbb{P}(A_1)$ and $\mathbf{R}' \in \mathbb{P}(A_k)$. We call an instance graph connected if all of its nodes are. Finally, the distance between two connected (distinct) nodes $\mathbf{R}$ and $\mathbf{R}'$ is defined as the length of the shortest path connecting the two. The distance between a node and itself is 0, whereas the distance between two unconnected nodes is undefined.

Given two instance graphs $\mathcal{G}_1 = \langle \mathcal{P}_1, \mathcal{A}_1 \rangle$ and $\mathcal{G}_2 = \langle \mathcal{P}_2, \mathcal{A}_2 \rangle$, we say that $\mathcal{G}_1$ is a subgraph of $\mathcal{G}_2$ (written $\mathcal{G}_1 \sqsubseteq \mathcal{G}_2$) if and only if:

- The nodes of $\mathcal{G}_1$ are a subset of the nodes of $\mathcal{G}_2$: $\mathcal{P}_1 \subseteq \mathcal{P}_2$.
- The edges of $\mathcal{G}_1$ are a subset of the edges of $\mathcal{G}_2$: $\mathcal{A}_1 \subseteq \mathcal{A}_2$.

Note that the constraint for $\mathcal{P}_1$ to be composed of the profiles mentioned in $\mathcal{A}_1$ must still hold. Thus, we cannot remove an edge A' without removing the profiles it mentions (if these profiles are mentioned only by A'), or have other similarly problematic situations. Further, notice that, for a set of axioms $\mathcal{A}$ and a set of instances $\mathcal{A}'$ such that $\mathcal{A}' \lhd \mathcal{A}$, it trivially holds that $\mathcal{G}_{\mathcal{A}'} \sqsubseteq \mathcal{G}_{\mathcal{A}}$. This is because $\mathcal{G}_{\mathcal{A}}$ is defined by the instances of $\mathcal{A}$, and the set of instances of $\mathcal{A}'$ (which is $\mathcal{A}'$ itself) is a subset of those instances. This fact will be used in the proof of the upcoming lemma.

3.4.2 *Instance Graphs and Justifications*

We are now going to show that, for all justifications $\langle \mathcal{A}^N, \mathcal{A}^E \rangle$ of a given justification problem $\langle \mathbf{R}^\star, X^\star, \mathbb{A} \rangle$, the explanation $\mathcal{A}^E$ corresponds to a connected instance graph that contains the goal profile. This means that, during the **generation step**, we only have to generate the instances mentioning the profiles connected to the goal profile. This has several computational advantages:

- If $\mathbb{A}$ is such that in the graph $\mathcal{G}_{\mathbb{A}}$ the profiles connected to $\mathbf{R}^\star$ are only a portion of the full set of profiles $\mathcal{R}(X)^+$, this generation method (compared to the "full approach") will always generate a portion of the full set of instances.
- In an intuitive sense, the *depth* of an explanation (defined as the maximum distance from the goal profile to some profile employed in the explanation) is related to how complex the resulting explanation is to read. That is, a very deep explanation may require reasoning about a very "distant" profile, and perform multiple reasoning steps far away from the goal profile. Therefore, in real-world applications, where the goal is to *explain* the decision to a non-expert public, we might want to limit the depth of the explanations. This view of explanations as instance graphs gives us a concrete way to perform this restriction.
- Finally, this notion of depth gives us a principled way to interleave generation and solving. For example, we could first search for justifications with a maximum depth of 0; if we do not find anything, we increase the maximum depth to 1, and so on. We can stop this process when we exhaust the computing resources, or when we generate the full graph. Alternatively, if our users are happy with just

one justification, we can stop as soon as we find one. An advantage of this is that, if a justification can be found within depth d, then we do not need to generate anything from depth $d+1$ on.

So then, let us prove that all explanations correspond to an instance graph that is connected and contains the goal profile.

Notice that the following lemma *does not* say that all connected graphs containing the goal profile are explanations. In other words, we are not reducing the **explanatoriness** condition of a set of instances to a set of graph-theoretic structural properties. We are just showing the opposite direction: we are showing that, if the **explanatoriness** condition is met for a set of instances $\mathcal{A}^E$, then some structural conditions hold for the corresponding graph. This is useful as, in some sense, it tells us where to look for an explanation, and will be used to prove the **correctness** of our main algorithm.

Lemma 3.3. *Let* $\langle \mathcal{A}^N, \mathcal{A}^E \rangle$ *be a justification for some justification problem* $\langle \mathbf{R}^\star, X^\star, \mathbb{A} \rangle$. *Further, let* $\mathcal{G}_{\mathcal{A}^E} = \langle \mathcal{P}^E, \mathcal{A}^E \rangle$ *be the instance graph corresponding to* $\mathcal{A}^E$. *Then, the following conditions must hold:*

(i) The graph $\mathcal{G}_{\mathcal{A}^E}$ *is a subgraph of the full graph corresponding to the corpus:* $\mathcal{G}_{\mathcal{A}^E} \sqsubseteq \mathcal{G}_{\mathbb{A}}$.

(ii) The graph $\mathcal{G}_{\mathcal{A}^E}$ *contains the goal profile:* $\mathbf{R}^\star \in \mathcal{P}^E$.

(iii) $\mathcal{G}_{\mathcal{A}^E}$ *is connected.*

Proof. We will show that, given a justification $\langle \mathcal{A}^N, \mathcal{A}^E \rangle$ of a justification problem $\langle \mathbf{R}^\star, X^\star, \mathbb{A} \rangle$, the graph $\mathcal{G}_{\mathcal{A}^E} = \langle \mathcal{P}^E, \mathcal{A}^E \rangle$ satisfies Conditions *(i)*, *(ii)* and *(iii)*.

(i) By **relevance** and **adequacy** of $\langle \mathcal{A}^N, \mathcal{A}^E \rangle$, we have that $\mathcal{A}^E \triangleleft \mathbb{A}$. Thus, as discussed above, $\mathcal{G}_{\mathcal{A}^E} \sqsubseteq \mathcal{G}_{\mathbb{A}}$.

(ii) Suppose that $\mathbf{R}^\star \notin \mathcal{P}^E$. Then, the goal profile is never mentioned in $\mathcal{A}^E$, which means that no constraint is placed over it. Consequently, adding the constraint that any F must have $F(\mathbf{R}^\star) \neq X^\star$ cannot make $\mathcal{A}^E$ unsatisfiable. But this means that there is a $F \in \mathbb{I}(\mathcal{A}^E)$ such that $F(\mathbf{R}^\star) \neq X^\star$. This contradicts the definition of justification; therefore, $\mathbf{R}^\star \in \mathcal{P}^E$.

(iii) Suppose that $\mathcal{G}_{\mathcal{A}^E}$ is not connected. Then, it is possible to partition $\mathcal{G}_{\mathcal{A}^E}$ into a set of subgraphs $\{\mathcal{G}_1, \dots, \mathcal{G}_k\}$ such that each of these subgraphs is connected. Let $\mathcal{G}^\star$ be the subgraph containing $\mathbf{R}^\star$ (which must surely exist, by virtue of the previous condition). Let $\mathcal{A}^\star \subset \mathcal{A}^E$ be the set of axiom instances corresponding to $\mathcal{G}^\star$. Since $\mathcal{A}^\star$ is a proper subset of $\mathcal{A}^E$, by **explanatoriness**, $\mathcal{A}^\star$ is not strong enough to enforce that $X^\star$ wins in $\mathbf{R}^\star$. Furthermore, since none of the other connected subgraphs of $\mathcal{G}_{\mathcal{A}^E}$ mention $\mathbf{R}^\star$, adding their corresponding instances to $\mathcal{A}^\star$ cannot possibly change this. But this implies that $\mathcal{A}^E$ does not enforce that $X^\star$ wins in $\mathbf{R}^\star$, which contradicts **explanatoriness**. Thus, $\mathcal{G}_{\mathcal{A}^E}$ must be connected.

This concludes the proof. $\square$

Let $\mathcal{G} = \langle \mathcal{P}, \mathcal{A}' \rangle$ be an instance graph. For any justification $\langle \mathcal{A}^{N}, \mathcal{A}^{E} \rangle$, we write $\langle \mathcal{A}^{N}, \mathcal{A}^{E} \rangle \sim \mathcal{G}$ (read as "$\langle \mathcal{A}^{N}, \mathcal{A}^{E} \rangle$ is taken from $\mathcal{G}$") if $\mathcal{A}^{E} \subseteq \mathcal{A}'$. In other words, this means that $\mathcal{A}^{E}$ is, in some sense, "contained" in $\mathcal{G}$.

For a given justification problem $\mathcal{J} = \langle \mathbf{R}^{\star}, X^{\star}, \mathbb{A} \rangle$, let $\mathcal{G}_{\mathcal{J}}$ be the maximal connected subgraph of $\mathcal{G}_{\mathbb{A}}$ containing $\mathbf{R}^{\star}$. Maximal, here, means that no connected subgraph of $\mathcal{G}_{\mathbb{A}}$ containing $\mathbf{R}^{\star}$ is a supergraph of $\mathcal{G}_{\mathcal{J}}$. More precisely, $\mathcal{G}_{\mathcal{J}} = \langle \mathcal{P}_{\mathcal{J}}, \mathcal{A}_{\mathcal{J}} \rangle$, where:

- $\mathcal{P}_{\mathcal{J}}$ is the set of all profiles that are connected to $\mathbf{R}^{\star}$ in $\mathcal{G}_{\mathbb{A}}$.
- $\mathcal{A}_{\mathcal{J}}$ is the set of all instances of $\mathbb{A}$ that mention some profile in $\mathcal{P}_{\mathcal{J}}$, that is, $\mathcal{A}_{\mathcal{J}} = \{ A' \lhd \mathbb{A} \mid \mathbb{P}(A') \subseteq \mathcal{P}_{\mathcal{J}} \}$.

Thus, an immediate consequence of the above lemma is the following.

Corollary 3.4. *Let $\mathcal{J} = \langle \mathbf{R}^{\star}, X^{\star}, \mathbb{A} \rangle$ be a justification problem. Then, every $\langle \mathcal{A}^{N}, \mathcal{A}^{E} \rangle$ for $\mathcal{J}$ is taken from $\mathcal{G}_{\mathcal{J}}$.*

Proof. By Lemma 3.3, we have that, for any justification $\langle \mathcal{A}^{N}, \mathcal{A}^{E} \rangle$, the graph $\mathcal{G}_{\mathcal{A}^{E}}$ is a connected subgraph of $\mathcal{G}_{\mathbb{A}}$ that contains $\mathbf{R}^{\star}$. Therefore, by definition of $\mathcal{G}_{\mathcal{J}}$, we have that $\mathcal{G}_{\mathcal{A}^{E}} \sqsubseteq \mathcal{G}_{\mathcal{J}}$. Hence, $\mathcal{A}^{E}$ is a subset of the instances of $\mathcal{G}_{\mathcal{J}}$ (by definition of subgraph). Thus, $\langle \mathcal{A}^{N}, \mathcal{A}^{E} \rangle$ is taken from $\mathcal{G}_{\mathcal{J}}$. $\square$

Consequently, we can limit ourselves to generate only the instances of $\mathcal{G}_{\mathcal{J}}$ (encoded as a gCNF). We are going to show how to do so in the following.

3.4.3 *Graph-Based Generation Algorithm*

Let $\mathcal{J} = \langle \mathbf{R}^{\star}, X^{\star}, \mathbb{A} \rangle$ be a justification problem. We will introduce an algorithm that, given $\mathcal{J}$, returns $\mathcal{G}_{\mathcal{J}}$. This approach is based on the Breadth-First Search graph algorithm (Russell and Norvig, 2015). Intuitively, we start by *expanding* the goal profile, that is, we generate all the instances that mention $\mathbf{R}^{\star}$. By doing so, we reach all profiles that are mentioned in these instances, that is, all profiles with depth 1. Once this is done, we expand all profiles reached in the previous step (thus reaching all profiles at a distance of 2 from the goal profile). This goes on until we explore the whole graph.

As discussed in Section 3.4.2, in practice, it might be useful to impose a maximum depth to the explanations. We can do so by limiting the maximum distance from the goal profile of the profiles we want to expand. Indeed, our algorithm will accept a parameter d controlling this distance. In particular, if we set d to 0, we will only expand the goal profile, and thus generate only the instances mentioning this profile. Conversely, if we do not place any restriction (i.e., we set d to ∞), the whole graph $\mathcal{G}_{\mathcal{J}}$ will be generated.

Before presenting the algorithm, called GraphGen, let us introduce some data structures which will be used in its presentation. Q is a FIFO queue that contains the profiles that have been reached and need to be expanded. Moreover, the sets $\mathcal{P}$ and $\mathcal{A}'$ contain the profiles and the instances of the graph generated so far, respectively. So then, let us now present the algorithm. For every justification problem $\mathcal{J}$ and $d \in \mathbb{N} \cup \{\infty\}$, the execution of GraphGen$(\mathcal{J}, d)$ goes as follows:[16]

1. Initialise $\mathcal{P}$ and $\mathcal{A}'$ as empty sets;
2. Initialise Q as a FIFO queue containing only the tuple $(\mathbf{R}^\star, 0)$ (the second element indicates the depth, i.e., the distance from $\mathbf{R}^\star$);
3. If Q is not empty, pop a tuple $(\mathbf{R}, d')$ from Q. Otherwise, stop and return $\langle \mathcal{P}, \mathcal{A}' \rangle$;
4. If $\mathbf{R}$ is not in $\mathcal{P}$:[17]
   (4a) Add $\mathbf{R}$ to $\mathcal{P}$;
   (4b) If $d' = d$, compute the set $\mathcal{A}_\mathbf{R} = \{ A' \mid A' \triangleleft \mathbb{A}, \mathbb{P}(A') = \{\mathbf{R}\}\}$ of *intraprofile* instances mentioning $\mathbf{R}$;[18]
   (4c) Otherwise, compute the set of $\mathcal{A}_\mathbf{R} = \{ A' \mid A' \triangleleft \mathbb{A}, \mathbf{R} \in \mathbb{P}(A')\}$ of *all* instances mentioning $\mathbf{R}$. Furthermore, for all profiles $\mathbf{R}'$ (distinct from $\mathbf{R}$) that are mentioned by some instance in $\mathcal{A}_\mathbf{R}$, add $(\mathbf{R}', d'+1)$ to Q;
   (4d) Add all instances in $\mathcal{A}_\mathbf{R}$ to $\mathcal{A}'$;
   (4e) Return to Step 3.

Note that, if a profile is at depth d, we only generate the instances of the intraprofile axioms which mention it. This is because, if we expand the graph up to depth d, we only want to consider the profiles up to that depth. Furthermore, since we only consider voting scenarios with a finite number of profiles, this algorithm always terminates.

We will now show that, if no bound is placed on the maximum depth, then this algorithm returns $\mathcal{G}_\mathcal{J}$. In particular, we will show that, for a large enough d, this graph is always fully explored. This will be useful later to prove the **correctness** of our main justification algorithm. To this end, let $\mathtt{dist}_\mathcal{J}(\mathbf{R}, \mathbf{R}')$ be the distance between $\mathbf{R}$ and $\mathbf{R}'$ in $\mathcal{G}_\mathcal{J}$.

Lemma 3.5. *For every justification problem $\mathcal{J} = \langle \mathbf{R}^\star, X^\star, \mathbb{A} \rangle$, there is a $d^\star \in \mathbb{N}$ such that, for every $d \geqslant d^\star$, the algorithm* GraphGen$(\mathcal{J}, d)$ *returns $\mathcal{G}_\mathcal{J}$.*

16 Throughout the rest of this work, we will sometimes set the value of d to ∞. Intuitively, this means that we place no bound on the maximum depth. Formally, the domain of this parameter is $\mathbb{N} \cup \{\infty\}$, where for every $d \in \mathbb{N}$ it holds that $d < \infty$. How this is dealt with in terms of implementation is not important. For example, in `Python`, one could use the value `float("inf")` to handle this case.
17 This means that $\mathbf{R}$ has not been expanded yet.
18 We will see in the next chapter how to construct all instances of an axiom that mention a given profile.

Proof. Without loss of generality, fix a justification problem $\mathcal{J}$. Let $\mathcal{C}_{\mathcal{J}}(\mathbf{R}^\star)$ be the set of profiles connected to $\mathbf{R}^\star$ in $\mathcal{G}_{\mathcal{J}}$. Then, let $d^\star$ be:

$$d^\star = \max_{\mathbf{R} \in \mathcal{C}_{\mathcal{J}}(\mathbf{R}^\star)} \{\mathtt{dist}_{\mathcal{J}}(\mathbf{R}, \mathbf{R}^\star)\} + 1$$

That is, the maximum depth of any profile connected to $\mathbf{R}^\star$ plus one. Note that $d^\star \in \mathbb{N}$, as we only consider a finite number of profiles.

Let $\langle \mathcal{P}, \mathcal{A}' \rangle$ be the output of GRAPHGEN$(\mathcal{J}, d^\star)$ and let $\mathcal{G}_{\mathcal{J}} = \langle \mathcal{P}_{\mathcal{J}}, \mathcal{A}_{\mathcal{J}} \rangle$. We will show that these two graphs are the same. First, we will show by induction that $\mathcal{P}_{\mathcal{J}} \subseteq \mathcal{P}$. Then, we will argue that $\mathcal{P}_{\mathcal{J}} \supseteq \mathcal{P}$ is also true. Finally, we will show that $\mathcal{A}_{\mathcal{J}} = \mathcal{A}'$. We will finally argue that the same holds for every depth greater than $d^\star$. This is enough to prove our claim.

Thus, let us show that $\mathcal{P}$ contains all $\mathbf{R} \in \mathcal{P}_{\mathcal{J}}$ by induction over $\mathtt{dist}_{\mathcal{J}}(\mathbf{R}, \mathbf{R}^\star)$. In the following, recall that every $\mathbf{R} \in \mathcal{P}_{\mathcal{J}}$ has $\mathtt{dist}_{\mathcal{J}}(\mathbf{R}, \mathbf{R}^\star) < d^\star$.

- Clearly, $\mathbf{R}^\star$ is the only profile with a distance of 0 from itself. This profile is trivially always added to $\mathcal{P}$.

- Suppose that all profiles that lie at a distance d from $\mathbf{R}^\star$ are added to $\mathcal{P}$. Then, given a profile $\mathbf{R}$ at depth $d+1$, there is an edge A' connecting $\mathbf{R}$ and a profile $\mathbf{R}'$ at depth d (by the definitions of distance and path). Since we generate all the edges connected to $\mathbf{R}'$ (as $d < d^\star$), $\mathbf{R}$ will be added to the queue (Step (4c)). Thus, $\mathbf{R}$ will be eventually added to $\mathcal{P}$ as well.

Hence, by induction, $\mathcal{P} \subseteq \mathcal{P}_{\mathcal{J}}$. Furthermore, no profile that is not connected to $\mathbf{R}^\star$ can be reached, as profiles are only reached through their connections to $\mathbf{R}^\star$. Consequently, $\mathcal{P} = \mathcal{P}_{\mathcal{J}}$.

Finally, let us look at $\mathcal{A}'$ and $\mathcal{A}_{\mathcal{J}}$. Consider any instance $A' \in \mathcal{A}_{\mathcal{J}}$. Surely, by definition of $\mathcal{A}_{\mathcal{J}}$, we must have that $\mathbb{P}(A') \subseteq \mathcal{P}_{\mathcal{J}}$. Since we reach and expand all profiles in $\mathcal{P}_{\mathcal{J}}$, surely A' will be generated.[19] Hence, $\mathcal{A}' \subseteq \mathcal{A}_{\mathcal{J}}$. Since no other kind of instances can be generated, we have that $\mathcal{A}' = \mathcal{A}_{\mathcal{J}}$.

Finally, note that, since we only made use of the fact that $\mathtt{dist}_{\mathcal{J}}(\mathbf{R}, \mathbf{R}^\star) < d^\star$ for every $\mathbf{R} \in \mathcal{P}_{\mathcal{J}}$, the above arguments hold for every $d > d^\star$. This concludes the proof. □

Notice that an immediate consequence of the above lemma is that GRAPHGEN$(\mathcal{J}, \infty)$ returns $\mathcal{G}_{\mathcal{J}}$.

Corollary 3.6. *For every justification problem* $\mathcal{J}$, GRAPHGEN$(\mathcal{J}, \infty)$ *returns* $\mathcal{G}_{\mathcal{J}}$.

Proof. This is an immediate consequence of Lemma 3.5. □

Next, we are going to show how this generation algorithm can be used to interleave generation and solving in a principled way.

19 Note that the condition in Step (4b) is never verified in this case, as $d^\star$ is greater than the distance of any profile from $\mathbf{R}^\star$.

3.4.4 *Iterative Deepening Algorithm*

We now propose our main justification algorithm. In this variant, conceptually inspired by the Iterative Deepening Search graph algorithm (Russell and Norvig, 2015), we interleave the **generation step** with the **solving step**. That is, every time we explore all profiles of depth d, we try to search for a justification, and then continue the generation (with depth $d+1$). This goes on until either the full graph is explored or when we reach some maximum depth $d_{\max}$. The main advantage of this is that, if an explanation exists "close" (in the sense discussed earlier) to the goal profile, we do not need to generate the whole graph. We now present the algorithm, which is called ITERJUSTIFY.

Notice that we will define it in terms of a *generic* graph generation algorithm GEN.[20] After, we will we prove that, if we use GRAPHGEN (as defined above) in combination with ITERJUSTIFY, we obtain a **correct** justification algorithm. Furthermore, given a graph $\mathcal{G} = \langle \mathcal{P}, \mathcal{A}' \rangle$, we define $\Phi(\mathcal{G}) = \Phi_{\mathcal{A}'}$, that is, the gCNF encoding of the instances of $\mathcal{G}$. Then, given a justification problem $\mathcal{J} = \langle \mathbf{R}^\star, X^\star, \mathbb{A} \rangle$ and a maximum depth $d_{\max} \in \mathbb{N} \cup \{\infty\}$, the execution of ITERJUSTIFY($\mathcal{J}$, $d_{\max}$, GEN) unfolds as follows:

1. Initialise d as 0 and $\mathcal{G}^{-1}$ as the empty graph $\langle \varnothing, \varnothing \rangle$;
2. If $d > d_{\max}$, then stop;
3. Otherwise, generate $\mathcal{G}^d$ by calling GEN($\mathcal{J}$, d);
4. If $\mathcal{G}^d = \mathcal{G}^{d-1}$, then stop;[21]
5. Otherwise, let Φ_d be $\Phi(\mathcal{G}^d) \cup \{\varphi_{Goal}\}$. While computing this, keep track of which axiom $A \in \mathbb{A}$ and which instance A' gave rise to each formula $\varphi_{A'} \in \Phi_d$;
6. Check whether Φ_d is satisfiable. If it is, jump to Step 7. If it is not, repeat the steps below until no new gMUS can be computed:
   (6a) Compute a new gMUS $\Phi^\star$ of Φ_d such that $\varphi_{Goal} \in \Phi^\star$. Define $\mathcal{A}^E$ as the set of instances corresponding to $\Phi^\star \setminus \{\varphi_{Goal}\}$;
   (6b) For every $\mathcal{A}^N$ such that $\mathcal{A}^E \triangleleft \mathcal{A}^N$ and for no $\mathcal{A} \subset \mathcal{A}^N$ the same holds, if $\Phi_{\mathcal{A}^N}$ is satisfiable, return the justification $\langle \mathcal{A}^N, \mathcal{A}^E \rangle$ (and continue the execution);
7. Set d to $d+1$ and then return to Step 2.

Before discussing the algorithm, notice that, strictly speaking, ITERJUSTIFY defines a *family* of justification algorithms (one for every $d_{\max}$ and GEN). Thus, we will denote the algorithm obtained by fixing these two parameters as ITERJUSTIFY($\cdot$, $d_{\max}$, GEN).

20 We do so because, later on in this work, we will introduce an alternative graph generation algorithm. Thus, having a modular notation will allow to discuss more easily about the different generation algorithms.

21 This means that we reached a fixed point, that is, the whole graph has been explored (and we won't find any more justifications).

Furthermore, every time d is increased, we regenerate the whole graph up to depth d. An alternative approach would be to *resume* the generation. That is, we expand the previously generated graph of one "step" (i.e., increase the depth of one). In terms of complexity, this makes no difference, as the complexity of the search is dominated by the generation of the deepest graph. We found that, in our experiments, this made no substantial difference as well. The only difference is that some justifications might be returned twice, but this is an implementation detail that is not a problem in practice. In light of this, we choose to present this version, as we find it leads to a much cleaner presentation and is more in line with the Iterative Deepening Search algorithm which we have taken inspiration from (Russell and Norvig, 2015).

As a final comment, notice that if GEN always terminates and eventually reaches a fixed point in the generation (i.e., for every justification problem $\mathcal{J}$ there is a d such that $\text{GEN}(\mathcal{J}, d) = \text{GEN}(\mathcal{J}, d+1)$), then $\text{ITERJUSTIFY}(\mathcal{J}, d_{max}, \text{GEN})$ terminates for every $\mathcal{J}$ and d_{max}. As we discussed, GRAPHGEN satisfies both conditions.

We will now prove the **soundess** (for every d_{max}) and **completeness** (for $d_{max} = \infty$) of ITERJUSTIFY when using the graph generation algorithm GRAPHGEN introduced above. To do so, we will first prove two intermediate results where we give some conditions on the graph generation algorithm that guarantee **soundness** and **completeness**.[22] Then, we will show that GRAPHGEN satisfies both conditions. In the following, a *graph generation algorithm* is an algorithm that, given a justification problem and an element of $\mathbb{N} \cup \{\infty\}$, returns an instance graph.

The first condition regards **soundness**. Essentially, it states that, if a generation algorithm GEN only returns subgraphs of $\mathcal{G}_{\mathcal{J}}$, then, if we use it in combination with ITERJUSTIFY, we obtain a **sound** algorithm.

Lemma 3.7. *Let* GEN *be a graph generation algorithm. If, for every justification problem* $\mathcal{J}$ *and* $d \in \mathbb{N} \cup \{\infty\}$, *the algorithm* $\text{GEN}(\mathcal{J}, d)$ *returns a subgraph of* $\mathcal{G}_{\mathcal{J}}$, *then for every* $d_{max} \in \mathbb{N} \cup \{\infty\}$ *the algorithm* $\text{ITERJUSTIFY}(\cdot, d_{max}, \text{GEN})$ *is* ***sound****.*

Proof. Suppose that, for every justification problem $\mathcal{J}$ and $d \in \mathbb{N} \cup \{\infty\}$, the algorithm $\text{GEN}(\mathcal{J}, d)$ returns a subgraph of $\mathcal{G}_{\mathcal{J}}$. We will show that then, for every $d_{max} \in \mathbb{N} \cup \{\infty\}$, $\text{ITERJUSTIFY}(\cdot, d_{max}, \text{GEN})$ is **sound**.

Without loss of generality, fix a justification problem $\mathcal{J}$. Let $\mathcal{G}^d$ be the graph returned by $\text{GEN}(\mathcal{J}, d)$. Further, let $\Phi_d = \Phi(\mathcal{G}^d) \cup \{\varphi_{\mathsf{Goal}}\}$ (as defined in Step 5 of ITERJUSTIFY). Since, by hypothesis, for every d we have that $\mathcal{G}^d$ is a subgraph of $\mathcal{G}_{\mathcal{J}}$, we must have that, for every d, $\Phi_d \subseteq \Phi_{\mathcal{J}}$ (the full gCNF formula). This holds because any such Φ_d can only contain the encodings of some instances of $\mathbb{A}$, and $\Phi_{\mathcal{J}}$ is composed of the encoding of *all* the instances of $\mathbb{A}$ (and both formulae contain φ_{Goal}). This implies that, for any d, any gMUS of Φ_d is a gMUS of $\Phi_{\mathcal{J}}$, as any subset of Φ_d is a subset of $\Phi_{\mathcal{J}}$.

22 We take this approach because, later in this chapter, we will need these conditions for another result.

Therefore, for every $d \in \mathbb{N} \cup \{\infty\}$, the following holds. If in Step (6b) we return a pair of axiom sets $\langle \mathcal{A}^N, \mathcal{A}^E \rangle$, then it means that $\langle \mathcal{A}^N, \mathcal{A}^E \rangle$ is a justification for $\mathcal{J}$. This holds because such a pair must surely satisfy all conditions of Theorem 3.2. Indeed, by the above argument, if we return $\langle \mathcal{A}^N, \mathcal{A}^E \rangle$, then it means that $\Phi_{\mathcal{A}^E} \cup \{\varphi_{goal}\}$ is a gMUS of $\Phi_{\mathcal{J}}$. This holds because $\Phi_{\mathcal{A}^E} \cup \{\varphi_{goal}\}$ is equal to some gMUS $\Phi^\star$ of Φ_d (as seen in Step (6a)), and thus of $\Phi_{\mathcal{J}}$. This satisfies Condition *(i)* of Theorem 3.2. Furthermore, all the other conditions of the theorem are checked in Step (6b).

Since the above holds for every $d \in \mathbb{N} \cup \{\infty\}$, surely for any $d_{max} \mathbb{N} \cup \{\infty\}$ we must have that the algorithm IterJustify$(\cdot, d_{max}, \text{Gen})$ can only return justifications for $\mathcal{J}$. Therefore, this algorithm is **sound** for every d_{max}. □

The second condition regards **completeness**. Essentially, it states the following. Suppose that the graph generation algorithm Gen, for some d, returns a graph that contains at least one justification for every (subset of a) normative basis that admits one. Then, if we use it in combination with IterJustify, we obtain a **complete** algorithm for $d_{max} = \infty$.

Lemma 3.8. *Let* Gen *be a graph generation algorithm. Suppose that for every justification problem $\mathcal{J}$, there is a $d^\star \in \mathbb{N}$ such that the following conditions are satisfied:*

(i) For every ***distinct*** *$d, d' \in \{0, \dots, d^\star\}$, the graph returned by* Gen$(\mathcal{J}, d)$ *is not equal to the graph returned by* Gen$(\mathcal{J}, d')$.[23]

(ii) Gen$(\mathcal{J}, d^\star)$ *returns a graph $\mathcal{G}^\star$ such that, for every set of axioms $\mathcal{A}^N$ such that a justification for $\mathcal{J}$ with normative basis $\mathcal{A}^N$ exists, there is a justification with normative basis $\mathcal{A}^N$ for $\mathcal{J}$ taken from $\mathcal{G}^\star$.*

Then, IterJustify$(\cdot, \infty, \text{Gen})$ *is* ***complete****.*

Proof. Without loss of generality, fix any justification problem $\mathcal{J}$. Suppose that, for some $d^\star \in \mathbb{N}$, the two conditions stated in the lemma hold. We will refer to these conditions as Condition *(i)* and *(ii)*, respectively. Then, we will show that IterJustify$(\cdot, \infty, \text{Gen})$ is **complete**.

Before going on, recall that a justification algorithm is **complete** if the following holds. For any input problem $\mathcal{J}$ and for all sets of axioms $\mathcal{A}^N$, if a justification for $\mathcal{J}$ that uses $\mathcal{A}^N$ as a normative basis exists, then on input $\mathcal{J}$ the algorithm returns at least one justification with normative basis $\mathcal{A}$ (for some $\mathcal{A} \subseteq \mathcal{A}^N$).

So then, let $\langle \mathcal{A}^N, \mathcal{A}^E \rangle$ be a justification for $\mathcal{J}$. If we show that IterJustify$(\mathcal{J}, \infty, \text{Gen})$ returns some justification with normative basis $\mathcal{A}$ (where $\mathcal{A} \subseteq \mathcal{A}^N$) we are done. Indeed, since no restriction is placed over $\langle \mathcal{A}^N, \mathcal{A}^E \rangle$, the same must hold for any justification. Note that this satisfies our notion of **completeness**.

23 Intuitively, this means that, if $d_{max} = \infty$, we will reach depth $d^\star$ before termination. Indeed, recall that IterJustify halts when a fixed point in the generation is encountered.

Since $d_{\max} = \infty$, the only way for ITERJUSTIFY to stop is by the check in Step 4. That is, we stop when we reach a fixed point in the graph generation. However, by Condition *(i)*, if the check in Step 4 is verified, we must have generated some graph $\mathcal{G}^\star$ with the property stated in Condition *(ii)*. By this condition, there must exist a justification $\langle \mathcal{A}^N, \mathcal{A}^{E\star} \rangle$ for $\mathcal{J}$ taken from $\mathcal{G}^\star$ (for some $\mathcal{A}^{E\star}$). Therefore, while searching for the gMUSes of $\Phi(\mathcal{G}^\star) \cup \{ \varphi_{Goal} \}$ (Step 6 of ITERJUSTIFY), we will find $\Phi_{\mathcal{A}^{E\star}} \cup \{ \varphi_{Goal} \}$.

Hence, we will surely return a justification $\langle \mathcal{A}, \mathcal{A}^{E\star} \rangle$ such that $\mathcal{A} \subseteq \mathcal{A}^N$. This is because we return all minimal justifications with explanation $\mathcal{A}^{E\star}$ (Step (6b) of ITERJUSTIFY), and a minimal justification $\langle \mathcal{A}, \mathcal{A}^{E\star} \rangle$ for some $\mathcal{A} \subseteq \mathcal{A}^N$ must exist.[24]

Thus, ITERJUSTIFY$(\cdot, \infty,$ GEN$)$ is **complete**. □

We are now ready to state our main result.

Theorem 3.9 (Correctness of the Graph-Based Algorithm)**.** *For every* $d_{\max} \in \mathbb{N} \cup \{ \infty \}$, *the justification algorithm* ITERJUSTIFY$(\cdot, d_{\max},$ GRAPHGEN$)$ *is* ***sound****. Furthermore, the justification algorithm* ITERJUSTIFY$(\cdot, \infty,$ GRAPHGEN$)$ *is* ***correct****.*

Proof. First, we will prove that, regardless of the maximum depth $d_{\max}$, the algorithm is **sound** (1). Then, we will show that for $d_{\max} = \infty$ the algorithm is also **complete** (2). The claim that for $d_{\max} = \infty$ the algorithm is **correct** follows from these two conditions.

(1) By construction, the graphs returned by GRAPHGEN are always composed of the profiles connected to $\mathbf{R}^\star$ and the relative instances. Hence, any such graph is a subgraph of $\mathcal{G}_{\mathcal{J}}$. Thus, by Lemma 3.7, ITERJUSTIFY$(\cdot, d_{\max},$ GRAPHGEN$)$ is **sound** for every $d_{\max} \in \mathbb{N} \cup \{ \infty \}$.

(2) Fix, without any loss of generality, a justification problem $\mathcal{J}$. Suppose that $d^\star$ is the smallest $d \in \mathbb{N}$ such that GRAPHGEN$(\mathcal{J}, d)$ returns $\mathcal{G}_{\mathcal{J}}$. Such a $d^\star$ surely exists (Lemma 3.5). We will prove that $d^\star$ that satisfies the Conditions *(i)* and *(ii)* of Lemma 3.8.

(i) Clearly, for every d, if GRAPHGEN$(\mathcal{J}, d)$ = GRAPHGEN$(\mathcal{J}, d+1)$, then it means that GRAPHGEN$(\mathcal{J}, d) = \mathcal{G}_{\mathcal{J}}$. This holds because, if by augmenting the maximum distance we do not add any new profiles or instances, it means that the graph returned by GRAPHGEN$(\mathcal{J}, d)$ already contains all the profiles connected to the goal profile and the instances that mention them. But $d^\star$ is, by hypothesis, the smallest d such that GRAPHGEN$(\mathcal{J}, d) = \mathcal{G}_{\mathcal{J}}$. Therefore, all graphs generated for depths smaller than $d^\star$ are distinct from one another. This satisfies the first condition of Lemma 3.8.

(ii) By Corollary 3.4, all justifications of $\mathcal{J}$ are taken from $\mathcal{G}_{\mathcal{J}}$, which is the graph returned by GRAPHGEN$(\mathcal{J}, d^\star)$. This is enough to satisfy the second condition of Lemma 3.8.

24 This holds because, given $\langle \mathcal{A}^N, \mathcal{A}^{E\star} \rangle$, we can remove all the unused axioms in $\mathcal{A}^N$ one-by-one until we are left with a minimal justification.

Therefore, by Lemma 3.8, ITERJUSTIFY($\cdot$, ∞, GRAPHGEN) is **complete**.

This concludes the proof. □

We argue that this algorithm has some interesting practical advantages. First, abstract **completeness** might not be that important in real-world applications, as users might be interested only in *one* justification. Thus, it might not be worth it to spend an unreasonable amount of computing time to search for more once one is found. This algorithm enables us to stop the search as soon as a justification is returned, without generating the full graph. Secondly, by generating the instances closer to the goal profile first, we will first return the most shallow explanations (in the sense we used so far). As we argued earlier, such explanations might be shorter, and thus easier to understand. This aligns well with our goal: providing explanatory justifications to the users. Thirdly, $d_{\max}$ acts as a parameter tuning the trade-off between **completeness** and performance. Indeed, we can choose how much of the graph to explore, from the goal profile only ($d_{\max} = 0$) to the whole graph ($d_{\max} = \infty$). In other words, we have an algorithm that allows for theoretical **completeness** and leaves us to decide whether we are interested in finding all justifications or not. This allows for greater control over the amount of computational resources we want to use.[25]

3.4.5 *Existential and Universal Axioms (Reprise)*

To conclude, let us briefly comment on some limitations of our algorithm. In particular, we will discuss the performance of ITERJUSTIFY when *existential axioms* (as defined in Section 2.2.2) are included in the corpus.

First, recall that the algorithm we introduced works (in principle) for any corpus of axioms (provided that the axioms can be encoded as gCNF formulae). However, as noted in Section 2.2.3, all the axioms that we will use in our experiments are universal. To see why we decided to focus on this case, recall that the main advantage of our algorithm is that it can be used to interleave generation and solving. When we include existential axioms in our corpus, this is no longer possible.

To understand this, consider the following. In our approach, with the notion of instance graph, we impose a *structure* to the space of all profiles. Intuitively, two profiles are near each other if there is some axiom instance that mentions them together. This structure is then used to guide the generation phase. This, in turn, enables us to generate the instances procedurally and to interleave the **generation** and **solving steps**.

For this very reason, our algorithm might be less suitable to work with axioms that existentially quantify over profiles. Indeed, consider the axiom of NON-IMPOSITION.

25 As we said, real-world users might be satisfied with just one explanation. Thus, in our experiments, we will focus on this case. However, scholars might be interested in finding *all* justifications (e.g., for gaining deeper insights into the nature of this problem). That is why we still insisted on the importance of (and hence proved) **completeness**.

This axiom states the following: "for any $x \in X$, there exists one profile $\mathbf{R}$ such that $F(\mathbf{R}) = \{x\}$".[26] Then, for every $x \in X$, we have an instance stating the following:

"There exists a profile $\mathbf{R} \in \mathcal{R}(X)^+$ such that $F(\mathbf{R}) = \{x\}$ holds."

Equivalently, for a voting scenario with ℓ profiles, we could express this as a disjunction over all profiles:

"The disjunction $\Big[F(\mathbf{R}_1) = \{x\}\Big] \vee \cdots \vee \Big[F(\mathbf{R}_\ell) = \{x\}\Big]$ must be true."

Therefore, every instance $A' \lhd$ NON-IMPOSITION has that $\mathbb{P}(A') = \mathcal{R}(X)^+$.[27] This would mean that, for any corpus $\mathbb{A}$ containing NON-IMPOSITION, in the corresponding instance graph all profiles would be connected with a distance of 1. Thus, we would not be able to properly interleave the **generation** and **solving steps**, as we would need to generate all the instances in one iteration. A similar argument can be made for all axioms that existentially quantify over $\mathcal{R}(X)^+$.

However, considering the goal of this work, we claim that this is not problematic. Indeed, justifications grounded in existential axioms might not be that interesting. Although such a justification is *conceivable*, we argue that it would be contrived and, in a sense, unappealing. This is because an explanation involving an existential axiom would somehow need to refer to (almost) the whole set of profiles. To see why, consider the following example. A justification for $\{x\}$ in $\mathbf{R}^\star$ that uses NON-IMPOSITION could have the following form. Suppose that we could justify an outcome different from $\{x\}$ for every profile in $\mathcal{R}(X)^+ \setminus \{\mathbf{R}^\star\}$. Then, if we want to satisfy NON-IMPOSITION, we must have that $F(\mathbf{R}^\star) = \{x\}$, since one profile with outcome $\{x\}$ must exist. Thus, $F(\mathbf{R}^\star) = \{x\}$. Notice that such an explanation would be prohibitively large, since it would need to concretely refer to every profile in the scenario.[28]

Of course, we do not claim that this is the only conceivable kind of explanation involving NON-IMPOSITION. More generally, we do not claim that no interesting justifications that use existential axioms exist. For instance, with very small scenarios, where the number of profiles is limited, one could imagine an appealing explanation that mentions all profiles. However, we argue that having an algorithm that is tailored (in terms of performance) towards universal axioms might still be of interest. Indeed, as argued above, from the point of view of explanatoriness,[29] universal axioms seem to be more useful in general. Hence, we deem this focus to be reasonable.

26 Perhaps, a more reasonable (albeit more restrictive) condition would be the following: "for every n with $1 \leqslant n \leqslant n^\star$ and every $x \in X$, there exists a profile $\mathbf{R} \in \mathcal{R}(X)^n$ such that $F(\mathbf{R}) = \{x\}$". However, for clarity, we stick to the less restrictive definition. The substance of the argument still stands.

27 We omit the gCNF encoding for simplicity. What is important here is that the resulting formula would necessarily need to refer to all profiles in the scenario.

28 Indeed, note that, in our language, there are no quantifiers. Thus, every object that the instances in an explanation constrain must be mentioned concretely.

29 Besides the formal notion of **explanatoriness** of a justification $\langle \mathcal{A}^N, \mathcal{A}^E \rangle$, we also use the word "explanatoriness" to refer to "the quality of being explanatory". The context should always be enough to determine which notion we are referring to. The meaning of "being explanatory" will be discussed in Chapter 5.

At this point, one final remark is in order. In sum, the difficulty regarding existential axioms stems from the fact that the instances of such axioms constrain together a large number of profiles. One could, in contrast, (potentially) conceive an axiom that existentially quantifies over a small set of profiles. Such an axiom seems not to be as problematic for our approach, as its instances would mention a small number of profiles. However, albeit logically possible, we are not aware of an appealing example from the literature of an axiom in this form.

3.5 HEURISTICS

In this section, we introduce two classes of heuristics that preserve **correctness** when combined with the justification algorithm introduced above.

Up to this point, we described two approaches that work (in principle) for any corpus. However, although one could define as many axioms as one likes, it seems reasonable to assume that the interesting, appealing and useful axioms are a subset of the logically possible ones. Therefore, it might be fruitful to focus on a particular set of appealing axioms and design a set of algorithmic improvements specific to these. We call, perhaps with an abuse of terminology, these algorithmic improvements *heuristics*. Intuitively, these are ways to alter the generation phase of the algorithm in order to improve the performance of the general algorithm. We define two general ways to do so, that is, two families of heuristics.[30]

We stress that, while the heuristics of these families are *specific to some (sets of) axioms* (that is, they exploit the concrete form of some specific axioms), they are not constrained to a particular corpus. That is, if we design a heuristic for a set of axioms $\mathcal{A}$, we want this to work for all corpora $\mathbb{A}$ such that $\mathcal{A} \subseteq \mathbb{A}$.

3.5.1 *Avoiding Redundancy: Implied Instances*

An instance graph is, as the name goes, a structure of axiom instances. Each of these instances *constrains* in some way the possible outcomes for the profiles it mentions. Let $A' \triangleleft A$ be one such instance for some axiom A. Suppose that, in the same instance graph, we have another set of axiom instances $\mathcal{A}^\star \triangleleft A$ that enforces the same constraint that A' enforces.[31] Then, the constraint enforced by A' is, in some sense, redundant. We call such an instance an *implied instance*. Before giving the formal definition, let us look at an example.

Example 3.5. Consider the three profiles $\mathbf{R}$, $\mathbf{R}'$ and $\mathbf{R}''$. Suppose that profile $\mathbf{R}'$ can be obtained from $\mathbf{R}$ by raising the support of alternative $x^\star$. Thus, we have an instance

30 The specific heuristics that we use will be discussed in the next chapter.

31 Note that, in general, $\mathcal{A}^\star$ might be stronger than A'. That is, $\mathcal{A}^\star$ implies A', but the opposite needs not to hold.

A' of POSITIVE RESPONSIVENESS which constrains $\mathbf{R}$ and $\mathbf{R}'$, stating that if $x^\star$ wins in $\mathbf{R}$ then it must be the sole winner in $\mathbf{R}'$.

Further, similarly as before, suppose that $\mathbf{R}''$ can be obtained from $\mathbf{R}'$ by raising the support of $x^\star$. Again, we have an instance A'' of the same axiom connecting $\mathbf{R}'$ and $\mathbf{R}''$. This instance states that if $x^\star$ wins in $\mathbf{R}'$ then it must be the sole winner in $\mathbf{R}''$.

However, if we can obtain $\mathbf{R}'$ from $\mathbf{R}$ and $\mathbf{R}''$ from $\mathbf{R}'$ by raising the support of $x^\star$, then we can obtain $\mathbf{R}''$ from $\mathbf{R}$ in the same way. Thus, we also have an instance $A^\star$ connecting $\mathbf{R}$ and $\mathbf{R}''$, stating that if $x^\star$ wins in $\mathbf{R}$ then it must be the sole winner in $\mathbf{R}''$

In this case, $A^\star$ is an implied instance of $\{A', A''\}$. Indeed, suppose that A' and A'' hold. Then, if $x^\star$ wins in $\mathbf{R}$, by A', we have $F(\mathbf{R}') = \{x^\star\}$. But by A'', then $F(\mathbf{R}'') = \{x^\star\}$ must hold as well. Hence, if F satisfies A' and A'', it must also satisfy $A^\star$. Thus, $\{A', A''\}$ implies $A^\star$. $\triangle$

We will prove that we can avoid generating such instances and still have a **correct** algorithm (provided that we generate the instances that imply them). In other words, we can add this heuristic to our graph algorithm without losing **correctness** (for $d_{\max} = \infty$). The advantage here is in terms of computational resources. In general, generating an instance A' has a certain computational cost, both in terms of computing time and in terms of the memory needed to store it. If we know that this constraint is already enforced by some other instances that have already been generated (or surely will be), we can avoid generating it and save the associated computational costs. Let us now look at this more formally.

Definition 3.4 (Implicant Graphs). *Let $\mathcal{G} = \langle \mathcal{P}, \mathcal{A}' \rangle$ be an instance graph and $\mathbb{A}$ a set of axioms such that $\mathcal{A}' \triangleleft \mathbb{A}$. An* implicant graph *of $\mathcal{G}$ with respect to $\mathbb{A}$ is an instance graph $\mathcal{G}^- = \langle \mathcal{P}, \mathcal{A}^- \rangle$ where $\mathcal{G}^- \sqsubseteq \mathcal{G}$ and, for any instance $A' \in \mathcal{A}'$ and axiom $A \in \mathbb{A}$ such that $A' \triangleleft A$, there is a set of instances $\mathcal{A}^\star \subseteq \mathcal{A}^-$ with $\mathcal{A}^\star \triangleleft A$ and $\mathbb{I}(\mathcal{A}^\star) \subseteq \mathbb{I}(A')$.*

In other words, for a given graph $\mathcal{G}$, an implicant graph is a subgraph of $\mathcal{G}$ whose instances logically entail all the instances of $\mathcal{G}$. Given an instance $A' \triangleleft A$, we call a set of instances $\mathcal{A}^\star \triangleleft A$ such that $\mathbb{I}(\mathcal{A}^\star) \subseteq \mathbb{I}(A')$ an *implicant set* of A' for A. The requirement of an implicant set to be composed of instances of the same axiom of the implied instance serves to guarantee **completeness**, and will be made clear during the proof of the next lemma (Lemma 3.10).

For a given justification problem $\mathcal{J} = \langle \mathbf{R}^\star, X^\star, \mathbb{A} \rangle$, recall that $\mathcal{G}_{\mathcal{J}}$ is the maximum connected subgraph of $\mathcal{G}_{\mathbb{A}}$ containing $\mathbf{R}^\star$. By Corollary 3.4, all justifications of $\mathcal{J}$ can be found within this graph. In the following, we are going to show we can restrict our generation to any implicant graph $\mathcal{G}^-_{\mathcal{J}}$ of $\mathcal{G}_{\mathcal{J}}$ with respect to $\mathbb{A}$ and still have a **correct** algorithm. To do so, we will first show that, if we restrict ourselves to an implicant graph of $\mathcal{G}_{\mathcal{J}}$ (with respect to $\mathbb{A}$), we can find at least one justification for all normative bases that admit one.

Lemma 3.10. *Let $\langle \mathcal{A}^N, \mathcal{A}^E \rangle$ be a justification for some justification problem $\mathcal{J} = \langle \mathbf{R}^\star, X^\star, \mathbb{A} \rangle$. If $\mathcal{G}^-_{\mathcal{J}}$ is an implicant graph of $\mathcal{G}_{\mathcal{J}}$ with respect to $\mathbb{A}$, then, for every set of axioms $\mathcal{A}^N$ such* *that a justification for $\mathcal{J}$ with normative basis $\mathcal{A}^N$ exists, there is a justification with normative basis $\mathcal{A}^N$ taken from $\mathcal{G}^-_{\mathcal{J}}$.*

Proof. Let $\mathcal{J} = \langle \mathbf{R}^\star, X^\star, \mathbb{A}\rangle$ be a justification problem and $\mathcal{G}^-_{\mathcal{J}}$ an implicant graph of $\mathcal{G}_{\mathcal{J}}$ with respect to $\mathbb{A}$. Moreover, let $\mathcal{A}^N$ be a set of axioms such that a justification for $\mathcal{J}$ exists. Let $\langle \mathcal{A}^N, \mathcal{A}^E\rangle$ be such a justification. We will proceed constructively to find a justification $\langle \mathcal{A}^N, \mathcal{A}^{E\star}\rangle \sim \mathcal{G}^-_{\mathcal{J}}$ (for some $\mathcal{A}^{E\star}$). This is enough to show our goal.

Recall that $\mathcal{A}^E$ is composed of instances of $\mathcal{G}_{\mathcal{J}}$ (Corollary 3.4). Thus, since $\mathcal{G}^-_{\mathcal{J}}$ is an implicant graph of $\mathcal{G}_{\mathcal{J}}$, every instance in $\mathcal{A}^E$ is either in $\mathcal{G}^-_{\mathcal{J}}$ or implied by some set of instances in $\mathcal{G}^-_{\mathcal{J}}$. So then, let us construct the set $\mathcal{A}$ from $\mathcal{A}^E$ as follows. Consider a $A' \in \mathcal{A}^E$ and some $A \in \mathcal{A}^N$ such that $A' \lhd A$ (such an A surely exists, by the **relevance** of $\langle \mathcal{A}^N, \mathcal{A}^E\rangle$). If A' is not within the instances of $\mathcal{G}^-_{\mathcal{J}}$, then substitute A' with one of its implicant sets for A (such an implicant set exists by definition of implicant graph, as $A \in \mathbb{A}$). Otherwise, keep A' in $\mathcal{A}$.

Clearly, $\mathcal{A}$ entails $\mathcal{A}^E$, because every instance in $\mathcal{A}^E$ is either in $\mathcal{A}$ or implied by some instances in it. Thus, any $F \in \mathbb{I}(\mathcal{A})$ is also in $\mathbb{I}(\mathcal{A}^E)$. By **explanatoriness** of $\mathcal{A}^E$, this means that any such F elects the target outcome in the goal profile. Now, take $\mathcal{A}^{E\star}$ to be any subset of $\mathcal{A}$ that minimally explains $\mathcal{J}$. Such a set exists, because $\mathcal{A}$ already enforces $F(\mathbf{R}^\star) = X^\star$: thus, we can remove instances from $\mathcal{A}$ one-by-one until we are left with a set that minimally explains the goal. Hence, **explanatoriness** holds for $\langle \mathcal{A}^N, \mathcal{A}^{E\star}\rangle$.[32]

Further, by construction, $\mathcal{A}^{E\star}$ is composed only of instances of $\mathcal{A}^N$.[33] Thus, **relevance** also holds. Moreover, **adequacy** and **non-triviality** hold by virtue of the fact that $\langle \mathcal{A}^N, \mathcal{A}^E\rangle$ is a justification for $\mathcal{J}$ (and hence we have that $\mathcal{A}^N$ is a non-trivial subset of $\mathbb{A}$).

Finally, $\langle \mathcal{A}^N, \mathcal{A}^{E\star}\rangle$ is a justification for $\mathcal{J}$, and by construction all instances of $\mathcal{A}^{E\star}$ are in $\mathcal{G}^-_{\mathcal{J}}$. Therefore, $\langle \mathcal{A}^N, \mathcal{A}^{E\star}\rangle \sim \mathcal{G}^-_{\mathcal{J}}$. □

In the next chapter, we are going to show how to modify the GraphGen algorithm to generate an implied graph of $\mathcal{G}_{\mathcal{J}}$ instead of the full graph. Now, we will lay some basic conditions for a heuristic based on these notions. Further, we will prove that any such heuristic, when used in combination with IterJustify, leads to a **sound** (for every $d_{\max}$) and **complete** (for $d_{\max} = \infty$). Let us first define precisely this notion.

Definition 3.5 (Implied Instance Heuristics)**.** *A graph generation algorithm* Heur$(\cdot, \cdot)$ *is an* implied instance heuristic *if, for every justification problem $\mathcal{J} = \langle \mathbf{R}^\star, X^\star, \mathbb{A}\rangle$, the following conditions hold:*

(i) For every $d \in \mathbb{N} \cup \{\infty\}$, Heur$(\mathcal{J}, d)$ *returns a subgraph of $\mathcal{G}_{\mathcal{J}}$.*

32 Here, we do not take $\mathcal{A}$ directly as $\mathcal{A}$ might be stronger than $\mathcal{A}^E$. That is, in general, we are not guaranteed that $\mathcal{A}$ *minimally* explains $\mathcal{J}$.

33 That is why we want any set of implicants of some instance $A' \lhd A$ to be composed of instances of A; if this were not true, we could not guarantee **completeness** in the general case.

(ii) *There is a* $d^\star \in \mathbb{N}$ *such that, for every* distinct $d, d' \in \{0, \ldots, d^\star\}$, *the graph returned by* HEUR$(\mathcal{J}, d)$ *is not equal to the graph returned by* HEUR$(\mathcal{J}, d')$. *Further,* HEUR$(\mathcal{J}, d^\star)$ *returns an implicant graph of* $\mathcal{G}_\mathcal{J}$ *with respect to* $\mathbb{A}$.

Intuitively, this definition prescribes that any such heuristic must only return graphs that are composed of the instances of $\mathcal{G}_\mathcal{J}$, and that, eventually, it must generate an implicant graph of $\mathcal{G}_\mathcal{J}$. We require all graphs that we generate before the implicant graph to be distinct from each other because ITERJUSTIFY halts once a fixed point in the graph generation is reached. Note that, in particular, GRAPHGEN is an implied instance heuristic.

We will now show that we can use any such heuristic in combination with ITERJUSTIFY and obtain a **correct** algorithm. More precisely, we will show that, for every maximum depth, we will obtain a **sound** algorithm. Further, when we generate the whole graph, we have a **complete** algorithm.

Theorem 3.11 (Correctness of Implied Instance Heuristics)**.** *Let* HEUR *be an implied instance heuristic. Then, for every* $d_{\max} \in \mathbb{N} \cup \{\infty\}$, *algorithm* ITERJUSTIFY$(\cdot, d_{\max}, \text{HEUR})$ *is* ***sound****. Further,* ITERJUSTIFY$(\cdot, \infty, \text{HEUR})$ *is* ***correct****.*

Proof. Clearly, Condition *(i)* of the definition of implied instance heuristics satisfies the condition of **soudness** that we described in Lemma 3.7. Thus, for every $d_{\max} \in \mathbb{N} \cup \{\infty\}$, the algorithm ITERJUSTIFY$(\cdot, d_{\max}, \text{HEUR})$ is **sound**.

Furthermore, fix a justification problem $\mathcal{J} = \langle \mathbf{R}^\star, X^\star, \mathbb{A}\rangle$. By the Condition *(ii)* of the definition of implied instance heuristics, for some $d^\star \in \mathbb{N}$, HEUR$(\mathcal{J}, d^\star)$ returns an implicant graph of $\mathcal{G}_\mathcal{J}$ with respect to $\mathbb{A}$. Let this graph be $\mathcal{G}^-_\mathcal{J}$. Further, by the same condition, all the graphs generated up to that point are distinct from each other. Finally, recall that, by Lemma 3.10, for every justification $\langle \mathcal{A}^N, \mathcal{A}^E\rangle$ of $\mathcal{J}$, there is a justification $\langle \mathcal{A}^N, \mathcal{A}^{E\star}\rangle$ for $\mathcal{J}$ taken from $\mathcal{G}^-_\mathcal{J}$. But this means that HEUR satisfies the condition of **completeness** we described in Lemma 3.8. Therefore, ITERJUSTIFY$(\cdot, \infty, \text{HEUR})$ is **complete**, and hence **correct**.

This concludes the proof. □

3.5.2 *Reducing Depth: Derived Axioms*

We present another way to speed up the search for justifications. In this approach, common patterns of interaction between axioms (that involve multiple profiles) that are often found in explanations are "aggregated" into a single intraprofile axiom. In other words, a common explanation pattern that might involve multiple profiles is reduced to a single instance which mentions a single profile.

To see the advantage of this, suppose we are using the algorithm ITERJUSTIFY, and are interested in finding only *one* justification. That is, we stop the execution as soon as one is found. Intuitively, this algorithm tries to find one explanation that is as close to the goal profile as possible (in the sense we used so far). It does so by generating all profiles at depth d, searching for a justification within this depth, and then, in case of failure, moving on to depth $d+1$. This heuristic's objective is to "reduce" an explanation that would overall require a depth of d into a single instance that only mentions a profile at depth d' (with $d' < d$). Clearly, doing this allows stopping the generation at depth d' instead of generating the whole graph up to depth d. Let us make this notion more precise.

Definition 3.6 (Derived Axioms)**.** *Let $\mathcal{A}$ be a set of axioms with $\mathbb{I}(\mathcal{A}) \neq \varnothing$. A* derived axiom *of $\mathcal{A}$ is an axiom $A_d \notin \mathcal{A}$ such that $\mathbb{I}(\mathcal{A}) \subseteq \mathbb{I}(A_d)$ and, for all $A' \lhd A_d$, it holds that* $|\mathbb{P}(A')| = 1$.

Example 3.6. Recall that the axiom of POSITIVE RESPONSIVENESS states that, if profile $\mathbf{R}'$ can be obtained from $\mathbf{R}$ by raising the support of an alternative $x^\star$ in some preference order, then $x^\star \in F(\mathbf{R})$ implies $F(\mathbf{R}') = \{x^\star\}$. Furthermore, the axiom of CANCELLATION states that for all profiles $\mathbf{R}$, if all alternatives tie in a majority contest, then $F(\mathbf{R}) = X$ must hold.

Given the set $\{$POSITIVE RESPONSIVENESS, CANCELLATION$\}$, we can define the following derived axiom $A^\star$:

> "For all profiles $\mathbf{R}$, if $\mathbf{R}$ can be obtained from some Cancellation profile by raising the support an alternative $x^\star$, then $F(\mathbf{R}) = \{x^\star\}$."

It is trivial to see that any F that satisfies POSITIVE RESPONSIVENESS and CANCELLATION must satisfy this axiom as well. △

Let A_d be a derived axiom of some (non-trivial) set $\mathcal{A}$. We will show that, for every instance $A'_d \lhd A_d$, we can find a set of instances of $\mathcal{A}$ that imply A'_d. Intuitively, this allows us to do the following. Given an explanation containing some instances of A_d, we can substitute these instances by the instances of $\mathcal{A}$ that imply them. This fact will be used in the presentation of the upcoming algorithm. Thus, let us show this.

Lemma 3.12. *Let A_d be a derived axiom of a non-trivial set of axioms $\mathcal{A}$. For every instance $A'_d \lhd A_d$, there is a set of instances $\mathcal{A}' \lhd \mathcal{A}$ such that $\mathbb{I}(\mathcal{A}') \subseteq \mathbb{I}(A'_d)$.*

Proof. Let A_d be a derived axiom of $\mathcal{A}$. Without loss of generality, fix a specific instance $A^\star_d \lhd A_d$. We will construct a set $\mathcal{A}^\star \lhd \mathcal{A}$ such that $\mathbb{I}(\mathcal{A}^\star) \subseteq \mathbb{I}(A^\star_d)$.

Recall that the interpretation of an axiom is equal to the intersection of the interpretations of all its instances (see Section 3.2.2). Consequently, given any instance $A'_d \lhd A_d$, the interpretation of A_d must be a subset of the interpretation of A'_d. Hence, $\mathbb{I}(A_d) \subseteq \mathbb{I}(A^\star_d)$. However, by the definition of derived axioms, we have that $\mathbb{I}(\mathcal{A}) \subseteq \mathbb{I}(A_d)$; thus, $\mathbb{I}(\mathcal{A}) \subseteq \mathbb{I}(A^\star_d)$.

Now, let $\mathcal{A}^\star = \{ A' \mid A' \lhd \mathcal{A} \}$ (i.e., the set of all instances of $\mathcal{A}$). Clearly, $\mathbb{I}(\mathcal{A}^\star) = \mathbb{I}(\mathcal{A})$, as the interpretation of both sets is equal to $\bigcap_{A' \lhd \mathcal{A}} \mathbb{I}(A')$. Therefore, by the above arguments, we have that $\mathbb{I}(\mathcal{A}^\star) \subseteq \mathbb{I}(A^\star_d)$. This concludes the proof.[34] $\square$

We will now introduce a variant of our justification algorithm, ITERJUSTIFY, based on derived axioms. This algorithm is called DERIVEDITERJUSTIFY, and will accept a set of axioms $\mathcal{A}_d$ as a parameter. Essentially, for an input problem $\mathcal{J} = \langle \mathbf{R}^\star, X^\star, \mathbb{A} \rangle$, this algorithm will select the axioms of $\mathcal{A}_d$ that are derived axioms of $\mathbb{A}$. During the generation phase, it will generate the instances of these axioms as well. The core mechanism of the algorithm remains the same. However, note that a justification retrieved in this way might not be, technically speaking, a justification for $\mathcal{J}$. This is because it could be grounded in some axioms in $\mathcal{A}_d$ that are not in $\mathbb{A}$. To solve this, the algorithm will also convert back these justifications into justifications for $\mathcal{J}$. So then, let us present the algorithm. The execution of DERIVEDITERJUSTIFY($\mathcal{J}$, d_{max}, GEN, $\mathcal{A}_d$) (with $\mathcal{J} = \langle \mathbf{R}^\star, X^\star, \mathbb{A} \rangle$) goes as follows:

1. Let $\mathcal{A}^\star_d = \{ A_d \in \mathcal{A}_d \mid A_d$ is a derived axiom of some non-trivial $\mathcal{A} \subseteq \mathbb{A} \}$;[35]
2. Let $\mathcal{J}' = \langle \mathbf{R}^\star, X^\star, \mathbb{A} \cup \mathcal{A}^\star_d \rangle$;
3. Execute ITERJUSTIFY($\mathcal{J}'$, d_{max}, GEN). When a justification $\langle \mathcal{A}^N, \mathcal{A}^E \rangle$ is returned, freeze the execution and perform the following steps:
   (3a) If $\mathcal{A}^N \subseteq \mathbb{A}$, return $\langle \mathcal{A}^N, \mathcal{A}^E \rangle$ and jump to Step (3g);
   (3b) Compute $\mathcal{A}$ by replacing in $\mathcal{A}^E$ every instance A' that is not an instance of $\mathbb{A}$ with the instances of $\mathbb{A}$ that imply it (which exist, as per Lemma 3.12);[36]
   (3c) Let $\Phi^\star$ be a gMUS of $\Phi_{\mathcal{A}} \cup \{ \varphi_{Goal} \}$ such that $\varphi_{Goal} \in \Phi^\star$;[37]
   (3d) Let $\mathcal{A}^{E\star}$ be the set of instances corresponding to $\Phi^\star \setminus \{ \varphi_{Goal} \}$;
   (3e) Let $\mathcal{A}^{N\star}$ be a set such that $\mathcal{A}^{N\star} \subseteq \mathbb{A}$ and $\mathcal{A}^{E\star} \lhd \mathcal{A}^{N\star}$;
   (3f) Check whether $\Phi_{\mathcal{A}^{N\star}}$ is satisfiable.[38] If it is, return $\langle \mathcal{A}^{N\star}, \mathcal{A}^{E\star} \rangle$;
   (3g) Resume the execution of ITERJUSTIFY.

34 Note that, although we have shown that the lemma holds for the set of all instances of $\mathcal{A}$, in practice we will be able to find much smaller sets. In particular, for the derived axioms we will present in the next chapter, this set will be composed of only a handful of instances.

35 In practice, this can be checked by querying a (manually crafted) database that stores, for every derived axiom A_d, the non-trivial sets $\mathcal{A} \subseteq \mathbb{A}$ that imply A_d.

36 As we will see in the next chapter, derived axioms will be defined in such a way that finding these instances will be straightforward.

37 This step is necessary because we are substituting some instances of $\mathcal{A}^E$ with some other (possibly) *stronger* instances. Thus, we need to make sure that the resulting set is still minimally unsatisfiable.

38 As $\mathcal{A}^{N\star}$ might be, in general, stronger than $\mathcal{A}^N$, even if $\mathcal{A}^N$ is non-trivial, we need to check for the non-triviality of $\mathcal{A}^{N\star}$ as well.

We will now prove that, for every graph generation algorithm, this approach preserves **soundness** and **completeness**.

Theorem 3.13 (Correctness of Derived Axiom Heuristics). *Let $\mathcal{A}_d$ be a set of axioms, $d_{max} \in \mathbb{N} \cup \{\infty\}$ and* Gen *a graph generation algorithm. If* IterJustify$(\cdot, d_{max}, \text{Gen})$ *is* ***sound****, then* DerivedIterJustify$(\cdot, d_{max}, \text{Gen}, \mathcal{A}_d)$ *is also* ***sound****. If it is* ***complete****, then* DerivedIterJustify$(\cdot, d_{max}, \text{Gen}, \mathcal{A}_d)$ *is also* ***complete****.*

Proof. Let us prove the two statements separately.

- Suppose that IterJustify$(\cdot, d_{max}, \text{Gen})$ is **sound**. Therefore, every pair of sets of axioms returned by this algorithm is a justification for the input justification problem. Thus, every $\langle \mathcal{A}^N, \mathcal{A}^E \rangle$ found in Step 3 is also a justification for $\mathcal{J}'$ (as defined in Step 2 of the algorithm). Firstly, if $\langle \mathcal{A}^N, \mathcal{A}^E \rangle$ is a justification for $\mathcal{J}$, then we are done, as it is returned as it is in Step (3a). Thus, let us suppose that $\langle \mathcal{A}^N, \mathcal{A}^E \rangle$ is not a justification for $\mathcal{J}$. In this case, the algorithm continues the execution, until it reaches Step (3f) and returns a pair of axioms $\langle \mathcal{A}^{N\star}, \mathcal{A}^{E\star} \rangle$. We will show that this is a justification for $\mathcal{J}$. This is enough to show our claim. To do so, we will show that this pair satisfies the three conditions of Theorem 3.2.

  (i) This condition requires $\Phi_{\mathcal{A}^{E\star}} \cup \{\varphi_{\text{Goal}}\}$ to be a gMUS of $\Phi_{\mathcal{J}}$. Consider the set $\mathcal{A}$ we compute in Step (3b). Surely, $\Phi_{\mathcal{A}} \cup \{\varphi_{\text{Goal}}\}$ is a subset of $\Phi_{\mathcal{J}}$, the full gCNF encoding of $\mathcal{J}$ (as $\mathcal{A}$ only contains instances of $\mathbb{A}$). Thus, a gMUS of the former is a gMUS of the latter. Consequently, the formula $\Phi^\star$ (as computed in Step (3c)) is a gMUS of $\Phi_{\mathcal{J}}$. Note that $\Phi^\star = \Phi_{\mathcal{A}^{E\star}} \cup \{\varphi_{\text{Goal}}\}$; therefore, $\mathcal{A}^{E\star}$ satisfies this condition.

  (ii) This condition requires that $\mathcal{A}^{N\star} \subseteq \mathbb{A}$ and $\mathcal{A}^{E\star} \lhd \mathcal{A}^{N\star}$. This is checked at Step (3e).

  (iii) This condition requires that $\Phi_{\mathcal{A}^{N\star}}$ is satisfiable. This is checked in Step (3f).

  Therefore, $\langle \mathcal{A}^{N\star}, \mathcal{A}^{E\star} \rangle$ is a justification for $\mathcal{J}$.

- Suppose that IterJustify$(\cdot, d_{max}, \text{Gen})$ is **complete**. We will first show that any justification for $\mathcal{J}$ is also a justification for $\mathcal{J}'$. To do so, notice that both justification problems aim at justifying the outcome $X^\star$ for profile $\mathbf{R}^\star$. Then, let $\langle \mathcal{A}^N, \mathcal{A}^E \rangle$ be a justification for $\mathcal{J}$. Clearly, $\langle \mathcal{A}^N, \mathcal{A}^E \rangle$ is a justification for $\mathcal{J}'$: the only condition that might be violated is **adequacy**, as the conditions of **explanatoriness**, **relevance** and **non-triviality** do not depend on $\mathbb{A}$ (which is the only thing that differentiates $\mathcal{J}$ and $\mathcal{J}'$). However, if **adequacy** holds for $\mathcal{J}$, this means that $\mathcal{A}^N \subseteq \mathbb{A}$; but this implies that $\mathcal{A}^N \subseteq (\mathbb{A} \cup \mathcal{A}_d^\star)$, which satisfies adequacy for $\mathcal{J}'$. Thus, $\langle \mathcal{A}^N, \mathcal{A}^E \rangle$ is a justification for $\mathcal{J}'$.

  Recall that, by hypothesis, IterJustify$(\cdot, d_{max}, \text{Gen})$ is **complete**. Thus, for every justification $\langle \mathcal{A}^N, \mathcal{A}^E \rangle$ of $\mathcal{J}'$, we find a justification with normative basis $\mathcal{A}$ where

$\mathcal{A} \subseteq \mathcal{A}^N$. So then, let $\langle \mathcal{A}^N, \mathcal{A}^E \rangle$ be a justification for $\mathcal{J}$. As shown above, it is also a justification for $\mathcal{J}'$, and thus we will find some justification with normative basis $\mathcal{A}$ such that $\mathcal{A} \subseteq \mathcal{A}^N$. Let this justification be $\langle \mathcal{A}, \mathcal{A}^{E\star} \rangle$. Since $\mathcal{A} \subseteq \mathbb{A}$ holds, this justification will be returned as it is (Step (3a)). Therefore, we return $\langle \mathcal{A}, \mathcal{A}^{E\star} \rangle$. This satisfies our definition of **completeness**.

This conclude the proof. □

Finally, note that to be helpful, a derived axiom needs to be quickly computable. Intuitively, computing the instances of this axiom A_d should not slow down the generation so much that it would be preferable to generate a deeper graph rather than incorporating this derived axiom. Therefore, not all "common explanation patterns" will be turned into derived axioms, but only those for which we were able to find a fast implementation.

3.5.3 *Redundancy versus Depth*

At this point, a final discussion is in order. The two heuristics that we defined seem to go in two opposite directions. On the one hand, implied instance heuristics aim to remove redundant constraints to speed up the computation. On the other hand, derived axioms aim at adding more (possibly redundant) constraints to facilitate the reasoner to extract a solution more easily (up to the same depth). The two seem to contradict each other, as adding more constraints requires more computing power, and removing constraints makes it harder for the solver to find a gMUS.[39] However, we stress that, regarding algorithmic optimisation, there is not always a clear "best" heuristic. That is, many opposite optimisation directions might exist, and these generally need to be carefully balanced. Indeed, usually, we cannot merely choose the best optimisation strategy and "maximise" that one.

We believe that this idea applies to this problem as well. Indeed, as stated in the introduction, the automated justification of collective decisions is a hard problem in the complexity-theoretic sense (Boixel and de Haan, 2021). Therefore, no "best" (in general) optimisation approach likely exists. Consequently, we formalised these two design principles (reducing computation time and adding more constraints) as the two above heuristics. It is then a matter of experimentation to find the right balance, but we found that both can be useful together. We will discuss the concrete heuristics used in the next chapter.

39 Indeed, recall that gMUSes are unsatisfiable sets. Thus, the more constraints we have, the easier it is to find such a "contradicting" collection of constraints.

3.6 SUMMARY OF THE RESULTS

In this section, we will give a summary of the formal results presented in this chapter. An overview is given in Table 3.1.

Algorithm	**Sound?**	**Complete?**
JUSTIFY	Yes (Theorem 3.2)	Yes (Theorem 3.2)
ITERJUSTIFY($\cdot$, d_{max}, GEN)	Yes, if GEN is GRAPHGEN (Theorem 3.9) or an implied instance heuristic (Theorem 3.11)	Yes for $d_{max} = \infty$, if GEN is GRAPHGEN (Theorem 3.9) or an implied instance heuristic (Theorem 3.11)
DERIVEDITERJUSTIFY($\cdot$, d_{max}, GEN)	Yes, if ITERJUSTIFY($\cdot$, d_{max}, GEN) is **sound** (Theorem 3.13)	Yes, if ITERJUSTIFY($\cdot$, d_{max}, GEN) is **complete** (Theorem 3.13)

Table 3.1: A summary of the formal results presented in this chapter.

First, we recalled the algorithm proposed by Boixel and Endriss (2020), which we named JUSTIFY. As an immediate consequence of Theorem 3.2 (already proved by the aforementioned authors), this algorithm is **sound** and **complete** (and hence **correct**).

Then, we have introduced ITERJUSTIFY, a family of justification algorithms that are parametrised by a generation algorithm GEN and a maximum depth d_{max}. We have shown that, if GEN is equal to GRAPHGEN, then for every $d_{max} \in \mathbb{N} \cup \{\infty\}$ the algorithm ITERJUSTIFY($\cdot$, d_{max}, GEN) is **sound**. In particular, if $d_{max} = \infty$, then it is **complete** (and hence **correct**). Both facts are stated in our main result, Theorem 3.9.

Next, we have shown that we can substitute GRAPHGEN with any *implied instance heuristic* and still have a **sound** (for any $\mathbf{d_{max}}$) and **complete** (for $\mathbf{d_{max}} = \infty$) algorithm. This is stated in Theorem 3.11, which can be seen as a generalisation of Theorem 3.9 (since GRAPHGEN is technically an implied instance heuristic).

Finally, we have shown that adding the derived axioms during the generation preserves **soundness** and **completeness** (Theorem 3.13).

4 IMPLEMENTATION AND EXPERIMENTS

In this chapter, we propose a concrete implementation of the graph-based algorithm, used to empirically address the research question. In particular, we will define a variant of our main justification algorithm, IterJustify, based on the heuristics introduced in Chapter 3. Then, we present our experimental setup; in brief, we ran our algorithm on a set of test justification problems to assess its performance. Per our research goal, we focused on moderately-sized scenarios: 2 to 12 voters and 3 to 4 alternatives. Finally, we report and discuss our results. We found that, for most of the input problems, we were able to find at least one justification within minutes. We consider this as a positive answer to our research question.

4.1 IMPLEMENTATION

In this section, we present our implementation of the graph-based algorithm. First, we are going to recall the high-level structure of our approach. Then, we will describe and discuss how the instances are generated and detail the specific heuristics used in our experiments.

4.1.1 *Overview*

Before diving into the specific details, let us review the high-level structure of our algorithm. We can identify three main algorithmic ideas behind our approach.

- *Graph-based Generation.* The main idea behind our algorithm, IterJustify, is to employ a graph-like view of the axiom instances to generate them procedurally. We can use this to interleave generation and solving and to generate only a portion of the whole set of instances.
- *Heuristics.* The above idea works, in principle, for any corpus of axioms. Indeed, it only needs a way to generate, given an axiom A and a profile $\mathbf{R}$, the set of instances $A' \lhd A$ that mention $\mathbf{R}$. Since we will work with a specific corpus, we can use this knowledge to our advantage. We are going to propose some axiom-specific variations to the algorithm IterJustify and the generation algorithm GraphGen with the aim of improving their performance. In particular, we will propose some heuristics to reduce the number of generated instances (thus making the algorithm faster) and some heuristics to decrease the minimum depth needed to find some justifications (thus enabling us to stop the execution sooner).

- *SAT solver.* Once the instances have been generated and encoded as a group CNF formula, we delegate the task of finding a group MUS to an external, state-of-the-art SAT solver. We can thus exploit the performance of such tools without worrying about the implementation.[1]

For our experiments, we implemented the algorithm in `Python 3`.[2] To check whether a formula is satisfiable we use the SAT solver `lingeling`,[3] accessed from Python through the `pylgl` library.[4] Further, to enumerate the gMUSes of a gCNF formula, we use `MARCO`,[5] a MUS-enumeration tool capable of handling group CNF formulae (Zhao and Liffiton, 2016). Our code is publicly available on `Github`.[6]

We now present the details of the heuristic algorithm we implemented and tested. Specifically, we will present multiple variations to be applied to our generation and justification algorithms, that is, GraphGen and IterJustify. All of these changes will be combined in our final justification algorithm, HeurJustify. This algorithm is tailored for the specific corpus of axioms that we used in our experiments.

More precisely, the remainder of this section is divided into four parts. First, we exemplify how we generate the instances mentioning a given profile during the execution of GraphGen. Then, we show how we modify this generation algorithm to reduce the number of generated instances of some axioms. Next, we show the derived axioms that we use in our experiments. Finally, we conclude by defining HeurJustify, a variant of IterJustify based on all the heuristics and variations introduced above. We discuss briefly its **correctness**.

Note that the heuristics we will present are specific to the axioms listed in Section 2.2; however, we stress that new axioms can be added to the corpus without needing to remove or modify these heuristics.

4.1.2 *Generating the Instances*

We will now show how to generate, given an axiom A and a profile $\mathbf{R}$, the instances $A' \lhd A$ that mention $\mathbf{R}$. Recall that this is needed in the generation phase of our algorithm (see Section 3.4.3 for the details). More specifically, given an axiom A, we will show how to write a function that, given a profile $\mathbf{R}$, returns a gCNF encoding the instances of A mentioning $\mathbf{R}$. If there is no instance $A' \lhd A$ such that $\mathbf{R} \in \mathbb{P}(A')$, then we will return the empty set.

1 Indeed, recall that the goal of this thesis was not to improve the **solving step**, as there are already large communities of researchers working on the performance of SAT solvers. Our goal was to improve the generation algorithm, while still relying on these powerful reasoning tools.
2 `https://www.python.org/`
3 `http://fmv.jku.at/lingeling/`
4 `https://pypi.org/project/pylgl/`
5 `https://sun.iwu.edu/~mliffito/marco/`
6 `https://github.com/olinarr/AutomatedJustification`

We will first describe the general idea and then give some concrete examples. We adopt two different approaches for intraprofile axioms and interprofile axioms. For intraprofile axioms, we essentially check whether some property holds over a given profile. For example, we can check whether, in a given profile $\mathbf{R}$, all alternatives tie in a majority contest. If this holds, then we return a single instance of CANCELLATION stating that all alternatives must win in $\mathbf{R}$. We could also have multiple checks on the same profile. For example, for the PARETO PRINCIPLE, we will have one check for every alternative. That is, for every alternative $y \in X$ that is Pareto dominated by some alternative $x \in X \setminus \{ y \}$ in $\mathbf{R}$, we return an instance stating that y must not win in $\mathbf{R}$.

Conversely, for an interprofile axiom A, to generate the instances mentioning a given profile $\mathbf{R}$, we exhaustively construct all tuples of profiles $(\mathbf{R}_1, \dots, \mathbf{R}_k)$ such that there is an instance $A' \lhd A$ with $\mathbb{P}(A') = \{ \mathbf{R}, \mathbf{R}_1, \dots, \mathbf{R}_k \}$. How to make this construction depends on how A "links" the different profiles. Once these profiles are constructed, for each tuple $(\mathbf{R}_1, \dots, \mathbf{R}_k)$, we generate the corresponding instance(s). Let us now look at two examples.

Let us start from a simple case: FAITHFULNESS. Recall that this intraprofile axiom states that, for every $\mathbf{R}$, if $\mathbf{R}$ has a single voter, then its top preference should be the sole winner. Therefore, there is an instance of FAITHFULNESS for every singleton profile $\mathbf{R} \in \mathcal{R}(X)^1$, stating that $F(\mathbf{R}) = \{ \mathrm{top}(\mathbf{R}) \}$ must hold. Given any profile $\mathbf{R}$, we can generate the corresponding instances (encoded as gCNF formulae) by following these steps:

1. If $|\mathbf{R}| = 1$, return FAI($\mathbf{R}$, $\mathrm{top}(\mathbf{R})$);[7]
2. Otherwise, return $\varnothing$.

Here, FAI($\mathbf{R}$, $\mathrm{top}(\mathbf{R})$) is the gCNF encoding of the instance of FAITHFULNESS mentioning profile $\mathbf{R}$ (as defined in Section 3.2.2).

Let us now consider a slightly more complex case, NEUTRALITY. Recall that this axiom states that, given a pair of profiles $\mathbf{R}$ and $\mathbf{R}'$, if there is a permutation $\pi : X \to X$ such that $\mathbf{R} = \pi(\mathbf{R}')$, then $F(\mathbf{R}) = \pi(F(\mathbf{R}'))$ must hold as well. So then, let us consider any profile $\mathbf{R} \in \mathcal{R}(X)^+$. Which instances of NEUTRALITY mention $\mathbf{R}$? Clearly, any such instance constrains $\mathbf{R}$ with some profile $\mathbf{R}'$ such that, for some $\pi : X \to X$, we have $\mathbf{R} = \pi(\mathbf{R}')$. We can thus generate all these instances by considering *all* permutations $\pi : X \to X$, and for each of these, generate the instance that mentions $\mathbf{R}$ and $\pi(\mathbf{R})$. More specifically, given $\mathbf{R}$:

1. Initialise Φ as the empty set;

7 Note that, for simplicity, we are generating directly the gCNF encoding of the instance. Technically, while generating the graph, one could represent the instances as an abstract object, and *then*, during the solving phase, encode them. However, this is an implementation detail and makes no important difference.

2. For every non-trivial permutation $\pi : X \to X$, repeat the following steps:[8]
   (2a) Compute $\mathbf{R}'$ as $\pi(\mathbf{R})$;
   (2b) Add NEU($\mathbf{R}$, $\mathbf{R}'$, π) to Φ;
3. Return Φ.

Where NEU($\mathbf{R}$, $\mathbf{R}'$, π) is the gCNF encoding of the instance of NEUTRALITY that mentions $\mathbf{R}$ and $\mathbf{R}'$.

Similar generation algorithms can be written for all axioms in Section 2.2.3. Next, we will briefly focus on three specific axioms. We will show how we restrict the instances generated by each of these axioms and discuss how this impacts the **correctness** of our algorithm.

4.1.3 *Reducing the Number of Generated Instances*

In the following, for REINFORCEMENT, NEUTRALITY and POSITIVE RESPONSIVENESS, we will present a way to reduce the number of instances generated for a given profile. For the latter two axioms, we will implement the idea of *implied instance heuristics*. We will show that we can use these alternative approaches in place of the "full" generation methods and still have a **correct** algorithm.

Let us introduce some notation first. Recall that GRAPHGEN (as defined in Section 3.4.3) is the algorithm we use during the **generation step**. Let HEUR_A be an algorithm that, given a profile $\mathbf{R}$, returns (some of) the instances of some axiom A that mention $\mathbf{R}$. We then define $\text{GRAPHGEN} + \text{HEUR}_A$ as the algorithm obtained by using HEUR_A to generate the instances of A (instead of the "full" method) when running the algorithm GRAPHGEN. Let us now look at the three axioms separately.

4.1.3.1 *Reinforcement*

Let $\mathbf{R}$, $\mathbf{R}_1$ and $\mathbf{R}_2$ be three profiles such that $\mathbf{R} = \mathbf{R}_1 + \mathbf{R}_2$ holds. That is, $\mathbf{R}$ can be partitioned into two subprofiles $\mathbf{R}_1$ and $\mathbf{R}_2$. In this scenario, we have an instance of REINFORCEMENT mentioning $\mathbf{R}$, $\mathbf{R}_1$ and $\mathbf{R}_2$ stating the following:

"If $F(\mathbf{R}_1) \cap F(\mathbf{R}_2) \neq \varnothing$, then $F(\mathbf{R}) = F(\mathbf{R}_1) \cap F(\mathbf{R}_2)$ must hold."

In this case, we say that $\mathbf{R}$ is the superprofile of the above instance, and $\mathbf{R}_1$ and $\mathbf{R}_2$ are the subprofiles. More generally, if for some $\mathbf{R}$ and $\mathbf{R}'$ there is a profile $\mathbf{R}''$ such that $\mathbf{R}' + \mathbf{R}'' = \mathbf{R}$ holds, we say that $\mathbf{R}$ is a superprofile of $\mathbf{R}'$.

Recall that, in our generation algorithm (see Section 3.4.3), each time we *expand* a profile $\mathbf{R}$ we need to generate the instances of A mentioning $\mathbf{R}$ (for all axioms $A \in \mathbb{A}$).

8 By trivial permutation we mean the permutation $\pi_{id} : X \to X$ such that, for every $x \in X$, we have that $\pi_{id}(x) = x$.

Looking at the case of REINFORCEMENT, some special care is needed. Indeed, a profile **R** could be mentioned in an instance of REINFORCEMENT either as a superprofile or as a subprofile. In other words, we need to generate (1) all instances where **R** is mentioned as the superprofile (let us call them *super-instances*) and (2) all the instances where **R** is mentioned as one of the two subprofiles (let us call them *sub-instances*). Let us look at the two cases separately.

The first is, at least conceptually, straightforward. Indeed, to generate all super-instances, it is enough to enumerate all possible partitions of **R** into two subprofiles $\mathbf{R}_1$ and $\mathbf{R}_2$ (where both subprofiles have at least one voter). For each partition, we generate the corresponding super-instance.

Let us focus on the second case, that is, sub-instances. Let $\mathbf{R} \in \mathcal{R}(X)^+$ be a profile and let $n = |\mathbf{R}|$. What are the instances that mention **R** as a subprofile? Clearly, we have an instance for every $\mathbf{R}' \in \mathcal{R}(X)^+$ such that $|\mathbf{R} + \mathbf{R}'| \leqslant n^\star$ holds. Indeed, recall that our scenario contains all (and only) the profiles up to $n^\star$ voters. Therefore, we can add up to $n - n^\star$ voters to **R** and obtain a valid superprofile. Thus, for every profile $\mathbf{R}'$ with up to $n - n^\star$ voters, we have a sub-instance mentioning **R** and $\mathbf{R}'$ as subprofiles of $\mathbf{R} + \mathbf{R}'$. More precisely, we have a sub-instance for every $\mathbf{R}'$ in the set:

$$\bigcup_{i=1}^{n^\star - n} \mathcal{R}(X)^i$$

Recall that, as seen in Section 2.1.4, the cardinality of this set is given by the formula:

$$\left| \bigcup_{i=1}^{n^\star - n} \mathcal{R}(X)^i \right| = \sum_{i=1}^{n^\star - n} \binom{i + |X|! - 1}{i}$$

Clearly, by the above arguments, this formula counts the number of sub-instances that mention **R**. Thus, if we actually wanted to generate all sub-instances of a given profile, we would need to perform a prohibitive number of computations just for *one* profile. In the following, we propose a way to compute only a part of the sub-instances that still guarantees **completeness**.

Given a profile **R** with $n < n^\star$ voters,[9] we only generate the following subset of sub-instances. For any singleton profile $\mathbf{R}' \in \mathcal{R}(X)^1$, we generate the instance that constrains **R** and $\mathbf{R}'$ as subprofiles and $\mathbf{R} + \mathbf{R}'$ as the superprofile. In other words, while generating the sub-instances of **R**, we consider only the instances obtained by adding one single ballot to **R**. Notice that there are exactly $|X|!$ of these instances, as there are $|X|!$ possible preference orders.

9 This restriction is in place because we only speak about profiles in our voting scenario. Indeed, if **R** were to have $n^\star$ voters, it cannot be mentioned in any sub-instance, as any superprofile of **R** would necessarily have more than $n^\star$ voters. Notice that this holds because we are using a weaker version of REINFORCEMENT.

To make things more precise, to generate the instances of REINFORCEMENT of a profile $\mathbf{R}$, we will not use the standard approach we described for FAITHFULNESS and NEUTRALITY. Instead, we will use the following algorithm, called REINFHEUR:

1. Initialise Φ as the empty set;
2. For every unordered pair of profiles $\{\mathbf{R}_1, \mathbf{R}_2\}$ such that $\mathbf{R} = \mathbf{R}_1 + \mathbf{R}_2$, add REI$(\mathbf{R}, \mathbf{R}_1, \mathbf{R}_2)$ to Φ;[10]
3. If $|\mathbf{R}| < n^\star$, then, for every singleton profile $\mathbf{R}' \in \mathcal{R}(X)^1$, add REI$(\mathbf{R} + \mathbf{R}', \mathbf{R}, \mathbf{R}')$ to Φ;
4. Return Φ.

Here, REI$(\mathbf{R}, \mathbf{R}_1, \mathbf{R}_2)$ is the gCNF encoding of the instance of REINFORCEMENT with superprofile $\mathbf{R}$ and subprofiles $\mathbf{R}_1$ and $\mathbf{R}_2$.

We claim that, by using this approach, we still generate all the instances and profiles that we would generate with the standard approach (if no bound is placed on the maximum depth). To see why, consider the following. Suppose that we expand some profile $\mathbf{R}$ (with $|\mathbf{R}| < n^\star$). Then, we will reach (and expand) all superprofiles of $\mathbf{R}$ that are equal to $\mathbf{R}$ plus one single ballot (Step 3). When expanding these profiles, we will reach all profiles that are equal to $\mathbf{R}$ plus two ballots, and so on, by induction, until *every* superprofile $\mathbf{R}$ is reached. Let us make this claim more precise.

Lemma 4.1. *For every justification problem $\mathcal{J}$, the algorithm* GRAPHGEN + REINFHEUR$(\mathcal{J}, \infty)$ *returns $\mathcal{G}_\mathcal{J}$.*

Proof. Fix a justification problem $\mathcal{J} = \langle \mathbf{R}^\star, X^\star, \mathbb{A}\rangle$. Recall that GRAPHGEN$(\mathcal{J}, \infty)$ returns $\mathcal{G}_\mathcal{J}$ (by Corollary 3.6), the graph containing all the profiles connected to $\mathbf{R}^\star$ and the instances that mention these profiles. Further, fix a profile $\mathbf{R}$. Let us assume that REINFORCEMENT $\in \mathbb{A}$ (as otherwise nothing changes). We will show that, if $\mathbf{R}$ gets selected for expansion during the execution of GRAPHGEN + REINFHEUR$(\mathcal{J}, \infty)$, then all the instances of REINFORCEMENT that mention $\mathbf{R}$ as a subprofile will be generated. This is enough to show our goal, as the generation of sub-instances is the only difference between GRAPHGEN and GRAPHGEN + REINFHEUR.

Before doing so, let us introduce some additional notation. For an instance A' of REINFORCEMENT, let $\mathbf{R}_{\text{sup}}(A')$ denote the superprofile of A'. Further, let $C_\mathbf{R}(A') = |\mathbf{R}_{\text{sup}}(A')| - |\mathbf{R}|$. Intuitively, $C_\mathbf{R}(A')$ counts how bigger $\mathbf{R}_{\text{sup}}(A')$ is with respect to $\mathbf{R}$. Note that for a sub-instance A' of $\mathbf{R}$ it holds that $C_\mathbf{R}(A') \in \{1, \ldots, n^\star - |\mathbf{R}|\}$. With this, we will show that every sub-instance $A^\star$ of $\mathbf{R}$ is generated. We will do so by induction over $C_\mathbf{R}(A^\star)$. As argued above, this is enough to prove our claim.

10 Notice that, by enumerating all *unordered* pair of subprofiles, we are implicitly stating the following. For any pair of profiles $\mathbf{R}_1$ and $\mathbf{R}_2$, we consider the two instances REI$(\mathbf{R}_1 + \mathbf{R}_2, \mathbf{R}_1, \mathbf{R}_2)$ and REI$(\mathbf{R}_1 + \mathbf{R}_2, \mathbf{R}_2, \mathbf{R}_1)$ to be equivalent. Indeed, it is trivial to see that the two instances enforce the same constraint and mention exactly the same profiles. Therefore, we restrict ourselves to generate only one of the two.

- Consider any sub-instance $A^\star$ of $\mathbf{R}$ with $C_{\mathbf{R}}(A^\star) = 1$. Note that we can reach $\mathbf{R}_{\text{sup}}(A^\star)$ by adding one ballot to $\mathbf{R}$. Thus, when expanding $\mathbf{R}$, this instance is generated in Step 3 of ReinfHeur.

- Suppose that every sub-instance A' with $C_{\mathbf{R}}(A') = n$ is generated (with $n \in \{1, \ldots, n^\star - |\mathbf{R}| - 1\}$). Then, consider any sub-instance $A^\star$ of $\mathbf{R}$ with $C_{\mathbf{R}}(A^\star) = n+1$. Furthermore, consider any superprofile $\mathbf{R}'$ of $\mathbf{R}$ such that $|\mathbf{R}'| = |\mathbf{R}_{\text{sup}}(A^\star)| - 1$. Such a profile exists, as for a (large enough) superprofile $\mathbf{R}_{\text{sup}}(A^\star)$ of $\mathbf{R}$ we can remove any of the extra ballots from $\mathbf{R}_{\text{sup}}(A^\star)$ and still have a superprofile of $\mathbf{R}$. Notice that there must be a sub-instance A' of $\mathbf{R}$ with $\mathbf{R}_{\text{sup}}(A') = \mathbf{R}'$ (since $\mathbf{R}'$ is a superprofile of $\mathbf{R}$) and $C_{\mathbf{R}}(A') = n$ (since $\mathbf{R}'$ has one fewer ballot than $\mathbf{R}_{sup}(A^\star)$). Such instances are generated by hypothesis, and thus $\mathbf{R}'$ is added to the expansion queue and eventually expanded (since there is no restriction on the maximum depth). Since $|\mathbf{R}'| = |\mathbf{R}_{\text{sup}}(A^\star)| - 1$, we can reach $\mathbf{R}_{\text{sup}}(A^\star)$ by adding one ballot to $\mathbf{R}'$. Thus, during the expansion of $\mathbf{R}'$ we will generate an instance mentioning $\mathbf{R}_{\text{sup}}(A^\star)$ (as per Step 3 of ReinfHeur). Consequently, $\mathbf{R}_{\text{sup}}(A^\star)$ gets expanded, and thus $A^\star$ generated (as per Step 2 of ReinfHeur).

Finally, by induction, every sub-instance $A^\star$ of $\mathbf{R}$ is generated during the execution of GraphGen + ReinfHeur($\mathcal{J}, \infty$). This concludes the proof. □

At this point, a clarification is in order. This approach clearly changes the "structure" of the instance graph, that is, it might make some profiles appear later and further away from the goal profile than they would be in the normal graph. More specifically, we reduce the *breadth* of the graph (that is, we reduce the number of edges connected to a given profile $\mathbf{R}$) but might increase the distance between two profiles (and thus, the *depth*). As a result, a profile that could be reached within a certain depth d with the "full" approach might not be reached at the same depth with this approach, but only at some depth d' (where $d' > d$).

As such, this design choice clearly has an impact on our results, and on whether a certain profile (possibly needed for a justification) is generated within a given maximum depth. However, there are some advantages to this. First, the computational improvement enables us to deal with much harder justification problems. Indeed, for large voting scenarios, generating all the Reinforcement instances is unfeasible. Further, we claim that this heuristic is acceptable also in terms of *explanatoriness*. Let $\mathbf{R}$ be some profile with $|\mathbf{R}| < n^\star$ and $\mathbf{R}'$ be one of its superprofiles. For the sake of the argument, let us assume that $\mathbf{R}'$ has yet to be generated when $\mathbf{R}$ is selected for expansion. Intuitively, the larger $\mathbf{R}'$ is compared to $\mathbf{R}$, the more distant $\mathbf{R}'$ will be from $\mathbf{R}$ (with respect to ReinfHeur). Since we impose a maximum depth when generating the graph, this means that we will generate first the explanations that regard smaller superprofiles. Thus, in a sense, this heuristic prioritises the explanations involving smaller superprofiles (or, more precisely, it generates this kind of explanations first).

As we argued when introducing our restricted version of REINFORCEMENT, appealing to a superprofile might be of limited explanatory power; indeed, this prioritisation is in line with this argument.

All in all, we consider this trade-off between the depth and the breadth to be justified.

4.1.3.2 *Neutrality*

We will now present a heuristic based on the idea of *implied instance heuristics* to reduce the number of instances generated for the axiom of NEUTRALITY. Intuitively, this heuristic works as follows. Suppose that a given profile $\mathbf{R}$ is selected for expansion. Further, suppose that, somehow, we know that at this point of the execution we have already generated some instances of NEUTRALITY that mention $\mathbf{R}$. Then, we claim that we can avoid generating more instances of NEUTRALITY during the expansion of $\mathbf{R}$ and still obtain an implicant graph of the full graph. Let us present the algorithm, and then discuss our claim. To generate the instances of NEUTRALITY mentioning a profile $\mathbf{R}$, we will use the following algorithm, called NEUHEUR:

1. If an instance A' of NEUTRALITY such that some $\mathbf{R} \in \mathbb{P}(A')$ has been already generated,[11] return $\varnothing$;

2. Otherwise, return all instances of NEUTRALITY mentioning $\mathbf{R}$ (see Section 4.1.2).

Recall that, by Corollary 3.6, for every justification problem $\mathcal{J} = \langle \mathbf{R}^\star, X^\star, \mathbb{A} \rangle$, the algorithm GRAPHGEN$(\mathcal{J}, \infty)$ returns the full graph $\mathcal{G}_\mathcal{J}$ (the graph containing the goal profile, all of its connected profiles and the instances which mention them). We will now show that, if we use this alternative generation approach, we generate an implicant graph of $\mathcal{G}_\mathcal{J}$ with respect to $\mathbb{A}$ (see Definition 3.4). Intuitively, this holds because the NEUTRALITY connections (in the instance graph sense) are *transitive*. Suppose that there are two instances $A_{1\to2}$ and $A_{2\to3}$ of NEUTRALITY connecting $\mathbf{R}_1$ to $\mathbf{R}_2$ and $\mathbf{R}_2$ to $\mathbf{R}_3$, respectively. Then, there is an instance $A_{1\to3}$ of NEUTRALITY connecting $\mathbf{R}_1$ to $\mathbf{R}_3$. This holds because if we can rename $\mathbf{R}_1$ to $\mathbf{R}_2$ and $\mathbf{R}_2$ to $\mathbf{R}_3$, then we can rename $\mathbf{R}_1$ to $\mathbf{R}_3$ (where, by "renaming", we mean a renaming of the alternatives). Furthermore, as we will show, $A_{1\to2}$ and $A_{2\to3}$ imply $A_{1\to3}$. Indeed, let us make this statement more precise.

Lemma 4.2. *Let $\mathcal{J} = \langle \mathbf{R}^\star, X^\star, \mathbb{A} \rangle$ be a justification problem. Then, the algorithm* GRAPHGEN + NEUHEUR$(\mathcal{J}, \infty)$ *returns an implicant graph of $\mathcal{G}_\mathcal{J}$ with respect to $\mathbb{A}$.*

Proof. Recall that, given any graph $\mathcal{G}$, an implicant graph $\mathcal{G}^-$ of $\mathcal{G}$ with respect to $\mathbb{A}$ must have the same profiles as $\mathcal{G}$ and, for every axiom $A \in \mathbb{A}$ and instance $A' \lhd A$ in

11 Technically, the implementation of GRAPHGEN we presented does not allow to make this check. We leave out the details of how to achieve this as a matter of implementation.

$\mathcal{G}$, there must be a set of instances $\mathcal{A}' \lhd A$ that together imply A' (called an implicant set of A' for A). We need to show that, for every justification problem $\mathcal{J} = \langle \mathbf{R}^\star, X^\star, \mathbb{A}\rangle$, $\textsc{GraphGen} + \textsc{NeuHeur}(\mathcal{J}, \infty)$ returns an implicant graph of $\mathcal{G}_\mathcal{J}$ with respect to $\mathbb{A}$.

To this end, fix a justification problem $\mathcal{J} = \langle \mathbf{R}^\star, X^\star, \mathbb{A}\rangle$. Recall that $\textsc{GraphGen}(\mathcal{J}, \infty)$ returns $\mathcal{G}_\mathcal{J}$. Let us assume that $\textsc{Neutrality} \in \mathbb{A}$ (as otherwise nothing changes). We will show that, if a profile $\mathbf{R}$ gets selected for expansion during the execution of $\textsc{GraphGen} + \textsc{NeuHeur}(\mathcal{J}, \infty)$, then for every instance A' of $\textsc{Neutrality}$ mentioning $\mathbf{R}$ we generate an implicant set of A' and generate the profile connected to $\mathbf{R}$ through A'. This is enough to show our goal, as the generation of $\textsc{Neutrality}$ is the only difference between $\textsc{GraphGen}$ and $\textsc{GraphGen} + \textsc{NeuHeur}$. In other words, for every other axiom in $\mathbb{A}$, all instances are generated.

To do so, fix a profile $\mathbf{R}_1$ and suppose it is selected for expansion. Then, if no instance of $\textsc{Neutrality}$ mentioning $\mathbf{R}_1$ has been generated yet, surely all of the instances will be generated (and thus all of its connected profiles reached). Therefore, in this case, we are done. So, let us suppose that, when expanding a profile $\mathbf{R}_2$, an instance of $\textsc{Neutrality}$ that mentions $\mathbf{R}_1$ has been generated. Let this instance be $A_{2\to1}$, and suppose it refers to a permutation $\pi_{2\to1}$ such that $\pi_{2\to1}(\mathbf{R}_2) = \mathbf{R}_1$. Notice that, in this case, all the instances mentioning $\mathbf{R}_2$ must have been generated, as $\textsc{NeuHeur}$ returns either all or none of the instances mentioning a profile.

Let $A_{1\to3}$ be some instance (distinct from $A_{2\to1}$) mentioning $\mathbf{R}_1$ and another profile $\mathbf{R}_3$ such that, for a permutation $\pi_{1\to3} : X \to X$, $\pi_{1\to3}(\mathbf{R}_1) = \mathbf{R}_3$ holds. We will show that an implicant set of $A_{1\to3}$ has been generated and $\mathbf{R}_3$ has been reached. Again, this is enough to prove our claim, as no restriction has been placed over $A_{1\to3}$. Thus the same must hold in general.

First of all, notice that $\mathbf{R}_2$ (the profile that we expanded) and $\mathbf{R}_3$ are connected through $\textsc{Neutrality}$. Indeed, consider the permutation $\pi_{2\to3} = \pi_{1\to3} \circ \pi_{2\to1}$. Trivially, by construction, $\pi_{2\to3}(\mathbf{R}_2) = \mathbf{R}_3$. Thus, there is an instance $A_{2\to3}$ mentioning $\mathbf{R}_2$ and $\mathbf{R}_3$. This instance has been generated (as we generated all instances mentioning $\mathbf{R}_2$). Therefore, $\mathbf{R}_3$ has been reached. (Figure 4.1 gives a visual representation of this situation.)

$$\begin{tikzcd}
\mathbf{R}_1 \arrow[r, dash, dashed, "A_{1\to3}"] \arrow[d, dash, "A_{2\to1}"'] & \mathbf{R}_3 \\
\mathbf{R}_2 \arrow[ur, dash, "A_{2\to3}"'] &
\end{tikzcd}$$

Figure 4.1: If we already generated the $\textsc{Neutrality}$-connections between $\mathbf{R}_1$ and $\mathbf{R}_2$ and between $\mathbf{R}_2$ and $\mathbf{R}_3$, we can avoid generating the instance between $\mathbf{R}_1$ and $\mathbf{R}_3$, as it is implied by the two aforementioned instances.

Furthermore, we can show that $A_{2\to1}$ and $A_{2\to3}$ (the instances that we generated) imply $A_{1\to3}$ (our target instance). Suppose F satisfies $A_{2\to1}$ and $A_{2\to3}$. By $A_{2\to1}$, we have the following:

$$\pi_{2\to1}(F(\mathbf{R}_2)) = F(\mathbf{R}_1) \tag{4.1}$$

Moreover, by $A_{2\to3}$, we have that:

$$\pi_{2\to3}(F(\mathbf{R}_2)) = F(\mathbf{R}_3) \tag{4.2}$$

But since $\pi_{2\to3} = \pi_{1\to3} \circ \pi_{2\to1}$, we also have that:

$$\pi_{2\to3}(F(\mathbf{R}_2)) = \pi_{1\to3}(\pi_{2\to1}(F(\mathbf{R}_2))) \tag{4.3}$$

Notice that, in Equation 4.3, we can substitute both $\pi_{2\to1}(F(\mathbf{R}_2))$ with $F(\mathbf{R}_1)$ and $\pi_{2\to3}(F(\mathbf{R}_2))$ with $F(\mathbf{R}_3)$. We can do so by virtue of the identities expressed in Equations 4.1 and 4.2, respectively. Thus, we finally have that:

$$F(\mathbf{R}_3) = \pi_{1\to3}(F(\mathbf{R}_1))$$

Recall that $A_{1\to3}$ connects $\mathbf{R}_1$ to $\mathbf{R}_3$, where $\pi_{1\to3}(\mathbf{R}_1) = \mathbf{R}_3$. Thus, the above means that F satisfies $A_{1\to3}$. Therefore, set $\{A_{2\to1}, A_{2\to3}\}$ implies $A_{1\to3}$, our target instance. This concludes the proof. □

The advantage of this is purely in terms of computational resources. Indeed, in general, for a profile $\mathbf{R}$, there are $|X|!$ many instances of NEUTRALITY mentioning $\mathbf{R}$. This is because there are $|X|!$ permutations of the alternatives, and thus, $|X|!$ possible profiles that are equal to $\mathbf{R}$ up to a renaming of the alternatives. By using NEUHEUR in place of the full implementation, we avoid generating all of these instances for some profiles. To give a rough estimation, if we generate the instances for $\mathbf{R}$, then we can avoid doing it for its connected profiles, which are, again $|X|!$.

4.1.3.3 *Positive Responsiveness*

We apply a similar approach on the POSITIVE RESPONSIVENESS axiom, again based on the notion of implied instances. Recall that this axiom states that, given two profiles $\mathbf{R}$ and $\mathbf{R}^+$, if $\mathbf{R}^+$ can be obtained from $\mathbf{R}$ by raising the support of an alternative $x^\star$ in some preference order, then $x^\star \in F(\mathbf{R})$ implies $F(\mathbf{R}) = \{x^\star\}$. More intuitively, this axiom prescribes the following. If x is a (possibly tied) winner in $\mathbf{R}$ and we increase its support (that is, some voter raises x in its preference order), then x must become the sole winner. In our implementation, we will generate only the instances that mention a profile $\mathbf{R}^+$ where one single voter has raised the support of a single alternative by one single "step" (i.e., increased its rank by one) from $\mathbf{R}$. We will call such instances *single-step* instances. To clarify, we will give an example.

Example 4.1. Consider the profile $\mathbf{R}$:

$$\begin{array}{l}
\#1 \; : \; b \succ_1 a \succ_1 c \\
\#1 \; : \; c \succ_2 b \succ_2 a
\end{array}$$

Here, we named the two preference orders to clarify the upcoming presentation. By raising the support of a in both preference orders, we can obtain the profile $\mathbf{R}^+$:

$$\begin{array}{l}
\#1 \; : \; a \succ_3 b \succ_3 c \\
\#1 \; : \; a \succ_4 c \succ_4 b
\end{array}$$

Clearly, there is an instance of Positive Responsiveness connecting $\mathbf{R}$ and $\mathbf{R}^+$ (let this instance be called A'). However, consider the profiles $\mathbf{R}_1$ and $\mathbf{R}_2$, defined, respectively, as:

$$\begin{array}{l}
\#1 \; : \; a \succ_3 b \succ_3 c \\
\#1 \; : \; c \succ_2 b \succ_2 a \\
\hline
\#1 \; : \; a \succ_3 b \succ_3 c \\
\#1 \; : \; c \succ_5 a \succ_5 b
\end{array}$$

Note that we can reach $\mathbf{R}_1$ from $\mathbf{R}$ by raising the support of a in preference order $\succ_1$ (obtaining $\succ_3$). Similarly, we can reach $\mathbf{R}_2$ from $\mathbf{R}_1$ (by raising a in $\succ_2$, obtaining $\succ_5$), and $\mathbf{R}^+$ from $\mathbf{R}_2$ (raising a in $\succ_5$, obtaining $\succ_4$). Moreover, notice that for each of these pairs of profiles, there is a single-step instance of Positive Responsiveness. Finally, note that the conjunction of these single-step instances implies A'. A visual representation of this scenario is given in Figure 4.2. △

$$\begin{tikzcd}
\mathbf{R} \arrow[r] \arrow[rrr, dashed, bend right] & \mathbf{R}_1 \arrow[r] & \mathbf{R}_2 \arrow[r] & \mathbf{R}^+
\end{tikzcd}$$

Figure 4.2: If two profiles are linked by Positive Responsiveness, then we can connect them with a sequence of single-step instances. This figure represents the situation described in Example 4.1, and each arrow is an instance of Positive Responsiveness. Here, the direction of the arrow represents the direction of the increase in support. The solid lines represent the single-step instances, whereas the dashed line represents an instance that is not a single-step instance.

Therefore, to generate the instances mentioning a given profile $\mathbf{R}$, we will use the following algorithm, called PosHeur:

1. Initialise Φ as the empty set;

2. For every alternative $x \in X$ and for every preference order $\succ \in \mathbf{R}$, repeat the following steps:
   (2a) Compute $\mathbf{R}^+$ from $\mathbf{R}$ by increasing the rank of x in $\succ$ of one position. If this is not possible, jump to Step (2c);
   (2b) Add $\text{POS}(\mathbf{R}, \mathbf{R}^+, x)$ to Φ;[12]
   (2c) Compute $\mathbf{R}^-$ from $\mathbf{R}$ by decreasing the rank of x in $\succ$ of one position. If this is not possible, skip (do not execute) Step (2d);
   (2d) Add $\text{POS}(\mathbf{R}^-, \mathbf{R}, x)$ to Φ;
3. Return Φ.

As we did for the previous heuristic, we will now show that, if we use PosHeur, we generate an implicant graph of the full graph $\mathcal{G}_\mathcal{J}$. Let us first give an intuitive explanation. Suppose that a profile $\mathbf{R}$ is selected for expansion. With this approach, all the profiles that can be reached from $\mathbf{R}$ with a single-step increase in the support of some alternative will be generated (Step (2a)), and thus selected for expansion. However, during the expansion of these profiles, we will reach all the profiles that can be reached with two single-step increases from $\mathbf{R}$. This goes on, until we reach, by induction, *every* profile $\mathbf{R}^+$ that can be reached from $\mathbf{R}$ in this way. Moreover, the instances resulting from this chain imply the instance between $\mathbf{R}$ and $\mathbf{R}^+$ (this holds because, in a sense, the constraints prescribed by Positive Responsiveness are *transitive*). Finally, notice that this holds for the other direction as well. If $\mathbf{R}$ can be reached by raising the support of some alternative from $\mathbf{R}^-$ (by any number of ranks), we will eventually generate the instance mentioning $\mathbf{R}$ and $\mathbf{R}^-$ (per Step (2c)). Let us now prove our claim.

Lemma 4.3. *Let $\mathcal{J} = \langle \mathbf{R}^\star, X^\star, \mathbb{A} \rangle$ be a justification problem. Then, the algorithm* GraphGen + PosHeur$(\mathcal{J}, \infty)$ *returns an implicant graph of $\mathcal{G}_\mathcal{J}$ with respect to $\mathbb{A}$.*

Proof. Suppose $\mathbf{R}$ is selected for expansion. Let $A^\star$ be an instance of Positive Responsiveness that mentions $\mathbf{R}$ and some profile $\mathbf{R}^+$. We will show that by using PosHeur we will generate an implicant set of $A^\star$ and reach $\mathbf{R}^+$. As argued in the proof of the previous lemma, this is enough to show our goal. We will assume that $\mathbf{R}^+$ can be obtained from $\mathbf{R}$ by raising the support of some $x^\star$ (by any number of steps); the case where the opposite holds is completely symmetrical.

12 For more details on the definition of this formula, see Section 3.2.2. Note that, in this axiom, the order of the profiles matter. Indeed, what we are stating here is that $\mathbf{R}^+$ can be obtained from $\mathbf{R}$ by raising the support of x, not the opposite.

Before proving our claim, let us define some additional notation. Let $I_{\mathbf{R}}(\mathbf{R}^+)$ (as in "increase") be the number of single-step increases in the support of $x^\star$ needed to obtain $\mathbf{R}^+$ from $\mathbf{R}$. This can be defined as:

$$I_{\mathbf{R}}(\mathbf{R}^+) = \left[\sum_{\succ \in \mathbf{R}^+} b(\succ, x^\star)\right] - \left[\sum_{\succ \in \mathbf{R}} b(\succ, x^\star)\right]$$

Here, $b(\succ, x^\star)$ is the Borda score of $x^\star$ in $\succ$. Recall that the Borda score of $x^\star$ in $\succ$ counts the number of alternatives $y \in X \setminus \{x^\star\}$ such that $x^\star \succ y$. Thus, if $b(\succ, x^\star)$ increases of 1, we have increased the support of $x^\star$ of one step. Notice that $I_{\mathbf{R}}(\mathbf{R}^+) \geqslant 1$, since we assume that $\mathbf{R}^+$ can be reached from $\mathbf{R}$ by raising the support of $x^\star$.

With this, let us prove that by using PosHeur we will generate an implicant set of $A^\star$ and reach $\mathbf{R}^+$. We will do so by induction over $I_{\mathbf{R}}(\mathbf{R}^+)$.

- Suppose $I_{\mathbf{R}}(\mathbf{R}^+) = 1$. Then, there is a single-step increase connecting $\mathbf{R}$ and $\mathbf{R}^+$. Our claim trivially holds in this case.

- Let A' be an instance of Positive Responsiveness mentioning $\mathbf{R}$ and some $\mathbf{R}'$. Further, let $I_{\mathbf{R}}(\mathbf{R}') = n$ (where $n \geqslant 1$). Suppose that for any such instance A', the claim holds. That is, $\mathbf{R}'$ is reached and an implicant set $\mathcal{A}'$ of A' is generated. Consider an instance $A^\star$ of Positive Responsiveness mentioning $\mathbf{R}$ and some $\mathbf{R}^\star$ with $I_{\mathbf{R}}(\mathbf{R}^\star) = n + 1$. Since $I_{\mathbf{R}}(\mathbf{R}^\star) = I_{\mathbf{R}}(\mathbf{R}') + 1$, $\mathbf{R}'$ and $\mathbf{R}^\star$ are connected by a single-step instance. Let this instance be A_s (as in "single"). During the expansion of $\mathbf{R}'$ (which will happen, by hypothesis), we will generate A_s. Thus, we reach $\mathbf{R}^\star$.

  Further, $\mathcal{A}' \cup \{A_s\}$ implies $A^\star$. Suppose $F \in \mathbb{I}(\mathcal{A}' \cup \{A_s\})$. Then, if $x^\star \in F(\mathbf{R})$, $F(\mathbf{R}') = \{x^\star\}$. This holds because $\mathcal{A}'$ entails A' by hypothesis, and A' prescribes that if $x^\star$ wins in $\mathbf{R}$ then $x^\star$ is the sole winner in $\mathbf{R}'$. Further, $F(\mathbf{R}^\star) = \{x^\star\}$, because A_s prescribes that if $x^\star$ wins in $\mathbf{R}'$ then $x^\star$ is the sole winner in $\mathbf{R}^\star$. Thus, we have shown that $x^\star \in F(\mathbf{R})$ implies $F(\mathbf{R}) = \{x^\star\}$; but this means that F satisfies $A^\star$ as well. This concludes the inductive step.

Thus, by induction, $\mathbf{R}^+$ is reached and an implicant set of $A^\star$ is generated. This concludes the proof. □

Once again, let us comment on this heuristic. Similarly to ReinfHeur, PosHeur changes the structure of the generated graph; some profiles that would normally be generated at a given depth d might only be reached at some depth d' (with $d' > d$). As we discussed in the previous chapter, most of these heuristics imply some sort of compromise. In this case, we claim that for the scenarios considered, this heuristic is generally beneficial. Indeed, generating all the instances of Positive Responsiveness would make this problem much harder computationally. By using this approach, we can deal with larger scenarios.

Still, by doing so, we might lose some interesting profiles (within a certain maximum depth). Again, this is a matter of compromise. Still, we stress that some of the heuristics presented in the rest of this chapter, based on the idea of derived axioms, will partly address this. Indeed, these heuristics capture some common patterns of explanation involving POSITIVE RESPONSIVENESS. In other words, although we might lose some interesting profiles, we are still able to find some of the explanations that use these ungenerated profiles.

4.1.3.4 *Heuristic Generation Algorithm*

In this section, we will show that we can use the three heuristics introduced above, and still retain a **correct** justification algorithm. So then, let $\textsc{GraphGen}_H$ (as in "heuristic") be the graph generation algorithm $\textsc{GraphGen}$ that uses the above heuristics to generate the instances of REINFORCEMENT, NEUTRALITY and POSITIVE RESPONSIVENESS. Or, with a slight abuse of notation:

$$\textsc{GraphGen}_H = \textsc{GraphGen} + \textsc{ReinfHeur} + \textsc{NeuHeur} + \textsc{PosHeur}$$

We will show that we can use this algorithm in place of $\textsc{GraphGen}$ during the generation phase of $\textsc{IterJustify}$ and have a **correct** algorithm. To do so, we will show that $\textsc{GraphGen}_H$ is an implied instance heuristic (Definition 3.5).

Proposition 4.4. *For every* $d_{max} \in \mathbb{N} \cup \{\infty\}$, $\textsc{IterJustify}(\cdot, d_{max}, \textsc{GraphGen}_H)$ *is* ***sound****. Furthermore,* $\textsc{IterJustify}(\cdot, \infty, \textsc{GraphGen}_H)$ *is* ***correct****.*

Proof. We will show that $\textsc{GraphGen}_H$ is an implied instance heuristic (Definition 3.5). The claim will then follow directly from Theorem 3.11. To do so, we need to show that the two conditions of Definition 3.5 hold. We will prove that these hold separately in the following.

(i) Let $\mathcal{J}$ be an arbitrary justification problem. This condition states that, for every $d \in \mathbb{N} \cup \{\infty\}$, $\textsc{GraphGen}_H(\mathcal{J}, d)$ returns a subgraph of $\mathcal{G}_{\mathcal{J}}$. Clearly, all of the three heuristics only restrict the instances you can generate. Thus, since $\textsc{GraphGen}$ generates only subgraphs of $\mathcal{G}_{\mathcal{J}}$, the three heuristics cannot possibly add instances and profiles that are not in this graph. Therefore, for every $d \in \mathbb{N} \cup \{\infty\}$, $\textsc{GraphGen}_H(\mathcal{J}, d)$ returns a subgraph of $\mathcal{G}_{\mathcal{J}}$. Thus, the first condition holds.

(ii) Again, fix a justification problem $\mathcal{J} = \langle \mathbf{R}^\star, X^\star, \mathbb{A} \rangle$. This condition states that there must be a $d^\star \in \mathbb{N}$ such that $\textsc{GraphGen}_H(\mathcal{J}, d^\star)$ returns an implicant graph of $\mathcal{G}_{\mathcal{J}}$ for $\mathbb{A}$. Moreover, for every *distinct* $d, d' \in \{0, \ldots, d^\star\}$, the graph returned by $\textsc{GraphGen}_H(\mathcal{J}, d)$ is not equal to the graph returned by $\textsc{GraphGen}_H(\mathcal{J}, d')$.

Recall that, in $\textsc{GraphGen}$ (and thus in $\textsc{GraphGen}_H$), parameter d_{max} controls the maximum depth of the graph exploration. Starting from $d_{max} = 0$, we can keep increasing d_{max} until we reach a value where we have expanded all the profiles that are connected to $\mathbf{R}$. Let this value be $d^\star$. Clearly, all graphs we generate before reaching $d^\star$ are distinct from each other, as every time we increase d_{max} we expand more profiles. Thus, if we prove that $\textsc{GraphGen}_H(\mathcal{J}, d^\star)$ returns an implicant graph of $\mathcal{G}_\mathcal{J}$ with respect to $\mathbb{A}$, we are done.

Trivially, the graph returned by $\textsc{GraphGen}_H(\mathcal{J}, d^\star)$ is equal to the graph returned by $\textsc{GraphGen}_H(\mathcal{J}, \infty)$, as in both cases all the profiles connected to $\mathbf{R}^\star$ have been expanded. But we know that, for each of the three heuristics independently, when we add them to $\textsc{GraphGen}(\mathcal{J}, \infty)$ we return an implicant graph of $\mathcal{G}_\mathcal{J}$ with respect to $\mathbb{A}$. This was proved in Lemmas 4.1, 4.2 and 4.3.[13] Clearly, this must still hold if we use the three heuristics together, as each heuristic only changes the way a single axiom is implemented, and the arguments we used to prove these lemmas do not depend on the other axioms in the corpus. Therefore, $\textsc{GraphGen}_H(\mathcal{J}, \infty)$ returns an implicant graph of $\mathcal{G}_\mathcal{J}$ with respect to $\mathbb{A}$.

This concludes the proof. □

4.1.4 *Derived Axioms*

Now, we describe some derived axioms, as introduced in Section 3.5.2. Intuitively, these are intraprofile axioms that encapsulate common patterns of explanation that have been observed across different justification problems. The idea is that, by generating these common patterns early on during the search, one can find a justification that would normally require a certain depth without expanding the graph completely up to that depth.

4.1.4.1 *Equivalence of Symmetric Alternatives*

Consider the profile $\mathbf{R}$:

$$\#1 : a \succ b \succ c$$
$$\#1 : b \succ c \succ a$$
$$\#1 : c \succ a \succ b$$

As we already show in Example 2.2, any F that satisfies NEUTRALITY must have that $F(\mathbf{R}) = \{a, b, c\}$.[14] Intuitively, this holds because the three alternatives are, in some sense, *symmetric* with respect to this profile. In other words, they all appear at any given rank an equal amount of times.

13 In the case of REINFHEUR, we actually proved a stronger claim, but this is not a problem here: $\mathcal{G}_\mathcal{J}$ is trivially an implicant graph of itself.

14 Note that, were we to consider non-anonymous profiles (as Boixel and Endriss (2020) did), then to justify the same outcome we would also need the axiom of ANONYMITY.

In general, given a profile $\mathbf{R}$, if there is a permutation π of the alternatives such that $\mathbf{R} = \pi(\mathbf{R})$, NEUTRALITY prescribes that an alternative $x \in X$ wins in $\mathbf{R}$ if and only if $\pi(x)$ wins as well: $x \in F(\mathbf{R}) \iff \pi(x) \in F(\mathbf{R})$ (as $\mathbf{R} = \pi(\mathbf{R})$). Clearly, this argument can be repeated further: $x \in F(\mathbf{R}) \iff \pi(x) \in F(\mathbf{R}) \iff \pi(\pi(x)) \in F(\mathbf{R})$ (as $\mathbf{R} = \pi(\mathbf{R}) = \pi(\pi(\mathbf{R}))$). We can generalise this idea even more. Let π^k denote k applications of π:

$$\pi^k(x) = \underbrace{\pi \circ \cdots \circ \pi}_{k \text{ times}}(x)$$

Then, for every $k \in \mathbb{N}$, $x \in F(\mathbf{R}) \iff \pi^k(x) \in F(\mathbf{R})$. Moreover, let π^+ denote the transitive closure of π:

$$\pi^+ = \{ (x, y) \in X^2 \mid y = \pi^k(x), \text{ for some } k \in \mathbb{N} \}$$

Note that this is an equivalence relation over X.[15] Let $[x]_\pi$ be the equivalence class of x in the partitioning induced by π^+. Intuitively, this is the set of alternatives such that any $y \in [x]_\pi$ can be "reached" from x by repeatedly applying π. Thus, for every $\mathbf{R}$, if a permutation of the alternative π such that $\mathbf{R} = \pi(\mathbf{R})$ exists, then for every alternative $x \in X$ either $[x]_\pi \subseteq F(\mathbf{R})$ or $F(\mathbf{R}) \cap [x]_\pi = \varnothing$ must hold. Indeed, suppose $x \in F(\mathbf{R})$. Then, by NEUTRALITY, every $y \in [x]_\pi$ must win as well. This is because any such y can be obtained by repeatedly applying π over x. As argued above, this implies that $y \in F(\mathbf{R})$.

Later, we will formalise this statement into a derived axiom of NEUTRALITY. For now, to clarify the above discussion, we will present another example.

Example 4.2. Consider the profile $\mathbf{R}$:

$$\#1 : a \succ b \succ c$$
$$\#1 : b \succ a \succ c$$

Here, we have two equivalence classes: $\{ a, b \}$ and $\{ c \}$. For the latter, we have that c either wins or does not. Clearly, that is trivially true for every F, regardless of this axiom. However, for the first class, we have that either both a and b should win, or both should lose. Thus, either $\{ a, b \} \subseteq F(\mathbf{R})$ or $\{ a, b \} \cap F(\mathbf{R}) = \varnothing$ must hold. △

Let us now introduce the derived axiom, called SYMMETRY:

> "For every profile $\mathbf{R}$, if there is a permutation $\pi : X \to X$ such that $\mathbf{R} = \pi(\mathbf{R})$, then for all $x \in X$ either $[x]_\pi \subseteq F(\mathbf{R})$ or $F(\mathbf{R}) \cap [x]_\pi = \varnothing$ hold."

As argued above, NEUTRALITY implies this axiom. In particular, given an instance A' of SYMMETRY, it is easy to exhibit the instance of NEUTRALITY that implies it. For

15 To see why, consider the directed graph $\langle X, \pi \rangle$. Notice that this graph is composed of a set of disjoint cycles. This holds because every node must have exactly one in-going edge and one out-going edge (by virtue of π being a permutation). Trivially, each cycle corresponds to an equivalence class induced by π^+.

example, if A' regards a profile $\mathbf{R}$ and a permutation $\pi : X \to X$, then the corresponding instance is $\text{NEU}(\mathbf{R}, \mathbf{R}, \pi)$.

We will now give an algorithm to compute the set of instances of SYMMETRY mentioning a profile $\mathbf{R}$. Intuitively, it works as follows. Let $\succ^\star$ be any preference order in $\mathbf{R}$. For every $\succ \in \mathbf{R}$ such that $\succ \neq \succ^\star$, we will construct the permutation $\pi : X \to X$ such that $\pi(\succ^\star) = \succ$. If $\pi(\mathbf{R}) = \mathbf{R}$, then we will construct an instance of SYMMETRY regarding $\mathbf{R}$ and π.[16] With this approach we find all interesting permutations, as any permutation π such that $\pi(\mathbf{R}) = \mathbf{R}$ must map $\succ^\star$ into some other preference order of $\mathbf{R}$. More precisely, we run the following algorithm:

1. Initialise Φ as $\varnothing$;
2. Choose a preference order $\succ^\star \in \mathbf{R}$;
3. For each preference order $\succ \in \mathbf{R}$ such that $\succ \neq \succ^\star$ and $\mathbf{R}(\succ) = \mathbf{R}(\succ^\star)$:[17]
   - (3a) Let π be a permutation such that $\succ = \pi(\succ')$;
   - (3b) If $\mathbf{R} \neq \pi(\mathbf{R})$, continue the execution of Step 3;
   - (3c) Else, compute the equivalence classes by finding the connected components on the graph $\langle X, \pi \rangle$;[18]
   - (3d) Add $\text{SYM}(\mathbf{R}, \pi)$ to Φ;
4. Return Φ.

Here, $\text{SYM}(\mathbf{R}, \pi)$ is the gCNF encoding of an instance of SYMMETRY regarding profile $\mathbf{R}$ and permutation π.

4.1.4.2 *Quasi-Tied Winners and Quasi-Tied Losers*

In this section, we will present two derived axioms of POSITIVE RESPONSIVENESS and CANCELLATION.

To introduce the first, let us start with an example. Consider profile $\mathbf{R}$:

$\#1 : a \succ b \succ c$

$\#1 : c \succ a \succ b$

Which alternative should win? To answer this, consider the profile $\mathbf{R}' = \{ a \succ b \succ c, c \succ b \succ a \}$. By CANCELLATION, we must have that $F(\mathbf{R}') = \{ a, b, c \}$. Further, notice

16 Note that, technically, we should also generate the permutation π' such that $\succ^\star = \pi'(\succ)$. However, an instance regarding this permutation would place the same constraint as the instance regarding the permutation π. Thus, we do not generate such instances.

17 The equality of multiplicities is checked to avoid useless comparisons: $\succ^\star$ must be mapped to a preference order of $\mathbf{R}$ which has been expressed by the same number of voters.

18 Recall that, as we briefly mentioned in a note above, each equivalence class corresponds to the node of some disjoint cycle in $\langle X, \pi \rangle$.

that we can obtain $\mathbf{R}$ from $\mathbf{R}'$ by raising the support of a. Therefore, by POSITIVE RESPONSIVENESS, we have that $F(\mathbf{R}) = \{a\}$. In this case, we say that $\mathbf{R}$ has a *quasi-tied winner*, namely, a. More generally, we can define the following derived axiom, called QUASI-TIED WINNERS:

> "For every profile $\mathbf{R}$, if $\mathbf{R}$ can be obtained from some Cancellation profile by raising the support an alternative $x^\star$, then $F(\mathbf{R}) = \{x^\star\}$."

It is trivial to see that POSITIVE RESPONSIVENESS and CANCELLATION imply this axiom. In particular, given an instance A' of QUASI-TIED WINNERS, we can construct the instances of POSITIVE RESPONSIVENESS and CANCELLATION that imply it. For example, if A' regards a profile $\mathbf{R}$ which can be obtained from a Cancellation profile $\mathbf{R}'$ by raising the support of $x^\star$, then these instances are CAN($\mathbf{R}'$) and POS($\mathbf{R}'$, $\mathbf{R}$, $x^\star$) (where CAN($\mathbf{R}'$) is the instance of CANCELLATION regarding $\mathbf{R}'$).[19]

We can get a derived axiom in the other direction as well. Indeed, consider the profile $\mathbf{R}$:

$$\#1 : a \succ b \succ c$$
$$\#1 : b \succ a \succ c$$

We can establish that c cannot win in $\mathbf{R}$ as follows. Consider profile $\mathbf{R}' = \{a \succ b \succ c, c \succ b \succ a\}$, which can be obtained by raising the support of c from $\mathbf{R}$. By CANCELLATION, $F(\mathbf{R}') = \{a, b, c\}$. So then, suppose $c \in F(\mathbf{R})$. This would mean that $F(\mathbf{R}') = \{c\}$ (by POSITIVE RESPONSIVENESS), which contradicts CANCELLATION. Thus, $c \notin F(\mathbf{R})$. Similarly as before, we say that c is a *quasi-tied loser* of $\mathbf{R}$. Again, we define the axiom QUASI-TIED LOSERS, derived from POSITIVE RESPONSIVENESS and CANCELLATION:

> "For every profile $\mathbf{R}$, if by raising the support of $x^\star$ in $\mathbf{R}$ we obtain a Cancellation profile, then $x^\star \notin F(\mathbf{R})$."

We stress that these two derived axioms can be checked on a profile $\mathbf{R}^\star$ quickly, that is, without exhaustively lowering (or raising) all alternatives until a Cancellation profile is found. We exemplify how to do so in the case of the first one, as the second one is completely symmetric in the other direction.

To generate the instances of QUASI-TIED WINNERS that mention a profile $\mathbf{R}$, we can use the following algorithm:

1. Check whether the two following conditions hold for some $x^\star$:
   - For any $y, z \in X \setminus \{x^\star\}$, we must have that $\mathbf{R}(y \succ z) = \mathbf{R}(z \succ y)$.
   - For any $y \in X \setminus \{x^\star\}$, we must have that $\mathbf{R}(x^\star \succ y) \geqslant \mathbf{R}(y \succ x^\star)$, and this inequality is strict for at least one y.

19 From $\mathbf{R}$, we can obtain profile the Cancellation $\mathbf{R}'$ by exhaustively lowering the support of $x^\star$ in all possible ways. Note that these instances can be constructed *after* the search for the justifications is finished. Hence, this does not slow down the algorithm significantly.

2. If the two above conditions hold for $x^\star$, exhaustively lower the support of $x^\star$ in $\mathbf{R}$ in all possible ways;
3. If during this process you find a profile $\mathbf{R}'$ that is a Cancellation profile, then return $\{\,\text{WIN}(\mathbf{R}, x^\star)\,\}$. Otherwise, return $\varnothing$.

The first two conditions can be checked quickly to find whether a candidate $x^\star$ exists for $\mathbf{R}$. Indeed, they are *necessary* but not *sufficient* conditions. To see why they are necessary, suppose that $\mathbf{R}'$ is a Cancellation profile and that $\mathbf{R}$ can be obtained by raising the support of $x^\star$ in $\mathbf{R}'$. The first condition states that all alternatives (except for $x^\star$) must tie in a majority comparison. This must hold, as in $\mathbf{R}'$ every alternative must tie in such a comparison. Since we can only raise the support of $x^\star$, this must still hold for every other alternative in $\mathbf{R}$. Furthermore, the second condition states that $x^\star$ must win in at least one majority contest, and never lose any. Indeed, in $\mathbf{R}'$, $x^\star$ ties with every other alternative: if we raise its support, then surely in at least one comparison $x^\star$ must become a winner. Further, we do not lower its support, thus it cannot lose in any majority contest.

However, the two conditions are not sufficient. For example, consider the following profile:

$$\begin{aligned}
\#1 &: a \succ c \succ d \succ b \\
\#1 &: a \succ d \succ c \succ b \\
\#1 &: b \succ c \succ a \succ d \\
\#1 &: b \succ d \succ a \succ c \\
\#1 &: c \succ b \succ d \succ a \\
\#1 &: d \succ b \succ c \succ a
\end{aligned}$$

In this profile, the two conditions hold for b. Indeed, b wins a majority contest against a, and ties against c and d. Furthermore, every other pair of alternatives ties in a majority contest. However, if we exhaustively try to find a Cancellation profile by lowering the support of b, we find that no such profile exists. This can be checked computationally.

Therefore, the last step confirms whether $\mathbf{R}$ can be obtained by raising $x^\star$ from some Cancellation profile. It does so by exhaustively searching for such a profile. This is expensive computationally, but it will only be executed on the small portion of the profiles that satisfy the two conditions. In other words, the conditions act as a "filter". On average, we found that this does not slow down the algorithm significantly.

Finally, note that the algorithm either returns one instance or none. This is not a problem, as there cannot be two distinct instances of this axiom that mention the same profile. If that were the case, with this axiom, we could justify two different outcomes for some $\mathbf{R}$. That, in turn, would mean that we could justify two different outcomes with corpus $\{$ Positive Responsiveness, Cancellation $\}$. But this is impossible: by Corollary 3.1, we cannot justify two different outcomes with a non-trivial corpus. Clearly, this is the case for this corpus: at least the Borda rule satisfies both axioms. Thus, we can only have one instance per profile.

4.1.5 *Putting It All Together: The Heuristic Algorithm*

We will now summarise all heuristics presented above into one single algorithm.

By Proposition 4.4, we know that the graph generation algorithm GraphGen$_H$ can be used to perform the **generation step** of our main justification algorithm, IterJustify, and that this guarantees **correctness**. Further, recall that DerivedIterJustify is an extension of IterJustify designed to incorporate derived axioms. More specifically, this variant accepts a set of axioms $\mathcal{A}_d$ as a parameter. If these are implied by the axioms in the corpus of the input justification problem, then DerivedIterJustify uses the axioms in this set to aid the search for justifications. By Theorem 3.13, this approach preserves **correctness**. Then, let $\mathcal{A}_d^\star$ be:

$$\mathcal{A}_d^\star = \{\textsc{Symmetry}, \textsc{Quasi-Tied Winners}, \textsc{Quasi-Tied Losers}\}$$

Putting together all the heuristics we introduced so far, we define the following algorithm:

$$\textsc{HeurJustify}(\mathcal{J}, d_{\max}) = \textsc{DerivedIterJustify}(\mathcal{J}, d_{\max}, \textsc{GraphGen}_H, \mathcal{A}_d^\star)$$

In our experiments, we will use HeurJustify. As a trivial consequence of the above discussion, we can state the following.

Proposition 4.5. *For every* $d_{\max} \in \mathbb{N} \cup \{\infty\}$*, the algorithm* HeurJustify$(\cdot, d_{\max})$ *is* ***sound****. Moreover,* HeurJustify$(\cdot, \infty)$ *is* ***correct****.*

Proof. This is an immediate consequence of Theorem 3.13 and Proposition 4.4. □

4.2 EXPERIMENTS

The purpose of this section is to directly address the research question, that is: *is it possible to design an algorithm for the problem of computing justifications for collective decisions that achieves good performance in practice?* To achieve this, we run our algorithm on several moderately-sized test cases, and check whether at least one justification can be found in a reasonable amount of time. To achieve this, we stop the execution of HeurJustify as soon as one justification is found.

We will first present the test data, then outline the experimental setup; finally, we will present and discuss the results.

4.2.1 *Test Data*

We test our algorithm on election profiles of different sizes. Due to limitations of a computational nature, we focus on the case of three and four alternatives. Further, we test our algorithm on profiles ranging from two to twelve voters. We argue that such moderately-sized scenarios are still of interest for this approach, as the notion of justification we use is suitable for small-scale collective decisions. Intuitively, with this approach, we will obtain "small" justifications (in terms of the size of the explanation) that can be discussed within a group of agents. Given that this notion is grounded in the concept of *axiom instances*, the explanations retrieved with our approach will consist of concrete arguments that mention the specific entities involved in the election: alternatives, profiles, and preference orders. For large-scale elections, with a vast amount of concrete objects involved, the resulting explanations might be too complex to be human-readable.

We will now present the test data. For the smaller scenarios we consider, it is possible to test the algorithm exhaustively on all profiles. For larger scenarios, this is no longer feasible, and thus a sampling of the profiles is necessary. The approaches are detailed in the following.

EXHAUSTIVE GENERATION. In this approach, we exhaustively test our algorithm on all profiles with n voters and m alternatives. Similarly to Boixel and Endriss (2020), we can restrict our search to a subset of the whole space, due to the symmetries within the set of all profiles. Indeed, once we run a test on a profile $\mathbf{R}$, we can avoid testing all profiles $\mathbf{R}'$ that are equal to $\mathbf{R}$ up to some permutation of the alternatives. For example, consider profile $\mathbf{R}$:

$$\#1 : a \succ b \succ c$$
$$\#1 : b \succ c \succ a$$

Further, consider profile $\mathbf{R}'$:

$$\#1 : b \succ c \succ a$$
$$\#1 : c \succ a \succ b$$

Let π be a permutation of the alternatives such that $\pi(a) = b$, $\pi(b) = c$ and $\pi(c) = a$. Notice that $\pi(\mathbf{R}) = \mathbf{R}'$. If we obtain a justification $\langle \mathcal{A}^N, \mathcal{A}^E \rangle$ for some outcome $X^\star$ for $\mathbf{R}$, we can "translate" such a justification into a justification for $\pi(X^\star)$ in $\mathbf{R}'$ by applying the same permutation to $\mathcal{A}^E$. To do so, we can apply π to every object mentioned in every instance $A' \in \mathcal{A}^E$.[20] Notice that we can do this regardless of whether we include NEUTRALITY in our corpus. We can thus partition the space of profiles into a

[20] For example, an instance of CANCELLATION that mentions a profile $\mathbf{R}$ can be "translated" into an instance that mentions a profile $\pi(\mathbf{R})$ (for a permutation of the alternatives π).

set of equivalence classes induced by these symmetries, and limit our tests to only one profile per class.

Clearly, this approach gives a full account of the performance of the algorithm in the given scenario. However, as stated earlier, when the number of voters and alternatives grows, the number of profiles (even up to our symmetry breaking approach) drastically increases (see Section 2.1.4 for some concrete numbers). Therefore, for larger voting scenarios, we generate only a portion of the full set of profiles, using the following approaches.

RANDOM GENERATION. In this approach, profiles are randomly generated. To this end, one has to choose a *statistical culture* (a term originally introduced by Garman and Kamien (1968)). Essentially, a statistical culture is a probability distribution over the elections (that is, profiles). Perhaps the simplest and most commonly used one is the *Impartial Culture* (IC), which corresponds to a uniform distribution over all profiles (originally named like this by Garman and Kamien (1968)). Although natural, this approach might be too simplistic to capture real-world elections. Intuitively, IC fails to capture that the preferences of real-world agents might be correlated: for example, there might exist similarities between the preferences of voters that belong to the same party. For more rigorous accounts on the limitations of IC, see the book by Regenwetter et al. (2006) or the paper by Szufa et al. (2020). Still, due to its simple nature and its widespread use, we employ IC as a baseline in our experiments.

An alternative statistical culture, which tries to model explicitly these similarities, is the Polya-Eggenberger (PE) urn model (Berg, 1985; Szufa et al., 2020). This model is parametrised by the *contagion rate* α (which controls how much the preferences are correlated) and can be sampled as follows. Start with an urn containing one copy of each of the $|X|!$ possible preferences in $\mathcal{L}(X)$. To draw a vote, we sample a preference from this urn and then return it to the urn together with $\alpha|X|!$ additional copies. We keep sampling until we reach the desired amount of votes. Notice that, in particular, for $\alpha = 0$, PE is equal to IC, whereas for $\alpha = \infty$ PE only generates unanimous profiles (that is, profiles where all voters share the same preference order).

Szufa et al. (2020) propose a framework to study the properties of the elections generated by different statistical cultures. They propose a distance measure between elections, which is used to understand how diverse the profiles generated by a certain culture are. Within this framework, PE seems to give good results in terms of the diversity of generated elections, and thus we employ it to generate our data. Furthermore, we adopt the values of α as used by the aforementioned authors, and generate the same number of profiles. More specifically, we adopt the following values of α: $\{0.01, 0.02, 0.05, 0.1, 0.2, 0.5\}$. For each α, we generate 30 elections, for a total of 180. For consistency with the PE model, we generate 180 profiles using IC as well.

REAL-WORLD DATA. In this approach, profiles are generated using real-world election data from `Preflib`.[21] `Preflib` (Mattei and Walsh, 2013) is a "reference library of preference data", which contains over 3000 datasets regarding elections and preferences. Since this work is focused on practical usefulness, this test data seems particularly interesting. Indeed, if the algorithm turns out to be useful on `Preflib` data, this would be strong evidence for its real-world applicability.

We consider the dataset of linear orders from `Preflib`.[22] A profile is encoded as a file consisting of a sequence of complete preference orders, where each preference order is paired with the number of voters who expressed it. This is, indeed, very close to our representation of profiles as multisets.

Still, most of the data from `Preflib` regards elections with large electorates, which are currently out of the scope of our algorithm. Thus, we consider the following approach to generate small profiles from `Preflib` data. Given a real-world profile $\mathbf{R}^\star$ with n voters, we can generate a profile $\mathbf{R}$ with $|\mathbf{R}| < n$ by sampling preferences orders from $\mathbf{R}^\star$. To do so, we assign to each preference order $\succ \in \mathbf{R}^\star$ a probability of $\frac{\mathbf{R}^\star(\succ)}{n}$ (recall that $\mathbf{R}^\star(\succ)$ is the number of voters in $\mathbf{R}^\star$ who express preference $\succ$). That is, we sample preference orders in a way that is proportionate to the fraction of voters who expressed these orders. We claim that this is a reasonably faithful way to restrict the size of `Preflib`-generated profiles, as, on expectation, it preserves the distribution of the preferences. Note that other authors have employed similar approaches to generate small profiles from `Preflib` data (for example, see the work of Hashemi and Endriss (2014)).

To generate our data, we selected 30 random files from those available on `Preflib` with the correct number of alternatives, depending on the experiment. From each of these, we sampled 6 profiles, for a total of 180.

SINGLE-PEAKED GENERATION. In this approach, we randomly generate profiles with a certain structure. A well-known idea in Social Choice Theory is the idea of *domain restrictions*. A domain restriction is, as the name goes, a restriction on the set of allowed profiles (that is, the domain over which SCFs are defined). This approach is used to study whether, by focusing on a certain subset of the profiles, some negative results can be mitigated (such as impossibility theorems). In other words, researchers investigate whether these negative results still hold in the restricted setting. The oldest (and perhaps most famous) domain restriction is the case of *single-peaked* profiles, introduced by Black (1948). In this case, we restrict our attention to profiles in which the alternatives can be ordered in a way such that the liking of voters first increases and then decreases. In other words, the alternatives can be positioned along a dimension such that the preferences have just one peak. More formally, a profile $\mathbf{R}$ is single-peaked if there is a dimension $\gg$ such that, for every $\succ \in \mathbf{R}$, we have that $x \succ y$

21 https://www.preflib.org/

22 https://www.preflib.org/data/format.php#soc

holds whenever x is between y and $\mathrm{top}(\succ)$ (according to $\gg$). For example, consider the following profile:

$$\begin{aligned}
\#1 &: d \succ c \succ b \succ a \\
\#1 &: c \succ b \succ a \succ d \\
\#1 &: c \succ d \succ b \succ a \\
\#1 &: b \succ a \succ c \succ d
\end{aligned}$$

This profile is single-peaked with respect to dimension $a \gg b \gg c \gg d$ (see Figure 4.3 for a visual representation).

This restriction is a natural assumption for some electoral problems: for instance, when voting for the legal drinking age, or when the candidates are distributed along the left–right political spectrum. For example, it has been successfully applied as a barrier against strategic manipulation in voting (Moulin, 1980).[23] Intuitively, this means that by focusing on this class of profiles, some negative results concerning strategic voting can be mitigated.

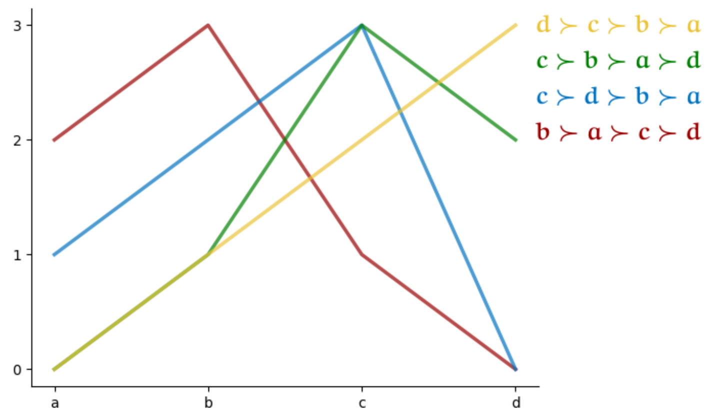


Figure 4.3: A visual representation of an example single-peaked profile, shown in the upper right corner. Each line (corresponding to the preference order of the same colour) represents the liking of a given voter for the alternatives. The horizontal axis represents the ordered dimension of the alternatives, whereas the vertical axis represents the appreciation (plotted as the Borda score). Notice the unique peak for each line.

We test our algorithm with this setting to see whether it leads to better results than the general case. To generate the data, we adopt the algorithm proposed by Walsh (2015) to sample 180 single-peaked profiles uniformly at random. We can see this

23 Briefly, strategic manipulation refers to a situation where voters misrepresent their true preferences in order to obtain better results in the election. Refer to the chapter by Conitzer and Walsh (2016) for an introduction on this problem and how it can be addressed.

approach as a restriction of IC over single-peaked profiles. Other statistical cultures for single-peaked profiles have been suggested; for instance, by Conitzer (2009). However, as Walsh notes in the aforementioned work, this model is much more likely to return profiles with peaks at the two ends of the spectrum than the uniform model. Since our baseline is IC, we stick to Walsh's approach.

4.2.2 *Corpus of Axioms*

We focus on a corpus composed of the following axioms: NEUTRALITY, the PARETO PRINCIPLE, FAITHFULNESS, CANCELLATION, REINFORCEMENT and the POSITIVE RESPONSIVENESS. This is very similar to the corpus used by Boixel and Endriss (2020), save for the inclusion of POSITIVE RESPONSIVENESS and the exclusion of the CONDORCET PRINCIPLE. Notice that the authors also included the axiom of ANONYMITY, which in our case is already encoded in the modelling of anonymous SCFs. Further, as explained in the previous chapter (Section 3.2.2), our corpus will also include the axiom ATLEASTONE.

We decided to focus on a single corpus because, as stated earlier, the scope of this experimental work is to test whether it is feasible to implement a justification algorithm, rather than studying the behaviour and explanatory power of different corpora. Moreover, by focusing on a corpus similar to the one used by Boixel and Endriss (2020), we can directly compare our approach to their method. Since the performance of the algorithm greatly depends on the corpus, the only way to understand the performance gain is to focus on a comparable set of axioms.

Furthermore, this corpus has some desirable properties. First, it is composed of well-known axioms within the Social Choice Theory literature. This means that we are not working with a toy example, but with a corpus that is widely accepted and could be used in real-world scenarios. Moreover, all of these axioms are *universal axioms*. As discussed in Section 3.4.5, our algorithm is tailored (in terms of performance) towards this category. Further, the axioms cover a small but diverse set of "connections" (in the graph-theoretic sense) between profiles. Indeed, REINFORCEMENT connects profiles across different sizes of electorates, NEUTRALITY connects profiles that are equivalent up to a renaming of the alternatives, whereas POSITIVE RESPONSIVENESS connects profiles where one single voter has changed their preference order. This leads to a reasonably broad exploration of the space of profiles, as the three axioms connect the profiles differently. This is important, as we want to check whether our algorithm can handle such complexity in the instance graph. Finally, notice that the corpus is non-trivial, as at least the Borda rule satisfies all axioms. Therefore, we can avoid checking whether the obtained justifications are non-trivial, and we know that one cannot justify multiple outcomes (Corollary 3.1).

POSITIVE RESPONSIVENESS has been added as a representative of a very common class of axioms within Social Choice Theory: monotonicity requirements. These axioms generally constrain the response of an SCF to the increase in the support of some alternative (the work of Zwicker (2016) contains a brief introduction to this kind of axioms). In particular, we found this axiom to be strong enough to have good explanatory power, while on the other being weak enough to have some interesting SCFs that satisfy it. Furthermore, this axiom is not too complex computationally to implement (at least in our graph-like framework), as only one voter at a time may change his preference order in a very restricted way.

The Condorcet Principle has been excluded from the experiments for reasons already outlined by Boixel and Endriss (2020). Including the Condorcet Principle would mean that all profiles with a Condorcet winner would be trivially justified with a depth of 0 (that is, just by looking at the goal profile). Although this could still be a satisfying explanation for some potential users, we are interested here in the performance of the algorithm, and thus we deem it reasonable to avoid this simple scenario. Furthermore, adding the Condorcet Principle would make the corpus non-trivial (for example, all SCFs that satisfy Reinforcement violate the Condorcet Principle (Zwicker, 2016)), thus making these experiments more challenging (as one would need to test for non-triviality).[24]

Finally, notice that our corpus is very closely related to the axiomatization of the Borda rule given by Young (1974): Neutrality, Anonymity (which in our case is built-in the definition of SCFs), Reinforcement and Cancellation. The only difference lies in our weaker formulation of Reinforcement, as previously discussed.

4.2.3 *Experimental Setup*

Recall that the purpose of the experiment is to show that, for most profiles tested, the implemented algorithm is capable of finding at least one justification in a reasonable amount of time. The rationale behind this is that, if we assume that the developed technology is to be used in a real-world tool, we would like users to be presented with at least one justification for their query within minutes. Thus, the details of the found justifications and a more specific account of the behaviour of the algorithm will be left for a qualitative study in the next chapter.

For the case of three alternatives, we exhaustively search for justifications for all profiles from 2 to 8 voters. For the case of 10 and 12 voters, we sample the space of profiles as described earlier.

For the case of four alternatives, we exhaustively search for justifications for all profiles of 2, 3 and 4 voters. For the case of 5, 6, 7 and 8 voters, we sample the space of profiles as described earlier. We also apply the sampling approaches to the case of

24 This does not hold for all the scenarios we consider in our experiments: for example, for 3 voters and 3 alternatives, the full corpus (i.e., including the Condorcet Principle) would be non-trivial, as observed by Boixel and Endriss (2020). Despite this, we decide to leave out this axiom from all experiments for consistency and simplicity, along with the aforementioned reasons.

4 voters, in order to compare how each of the proposed methods fares if compared to the exhaustive generation.

A further caveat is in order in this case. When running the experiments for four alternatives, the maximum depth for the graph generation has been fixed to 3. This is in line with the scope of the experiments, which is to test whether a justification can be found within a *reasonable* amount of time, rather than whether a justification can be found at all. Indeed, it has been observed (by previous experiments) that, for the case of four alternatives, beyond the depth of 3 the time and memory requirements of the algorithm drastically increase. Moreover, it seems that for the greater part of profiles for which a justification can be found, a justification can be found within this depth. Finally, explanations of depth greater than 3 are usually of considerable size and complexity, thus defeating the very purpose of producing explanations for a non-expert public. Consequently, we consider this restriction reasonable.

All experiments have been run on the LISA supercomputer, provided by SurfSARA. The details of the machine can be found online.[25]

4.2.4 *Results*

In the following, we analyze the results. We first consider the case of three alternatives, and then the case of four alternatives. Finally, we relate our findings to the results obtained by Boixel and Endriss (2020). We stress that all of our results are relative to the corpus we used and the scenarios we considered.

Before moving into the discussion, a general statement is in order. Notice that, for all of the justified profiles, the justified outcome coincides with the Borda winner. This does not come as a surprise: as we stated earlier, our corpus closely resembles a characterisation of the Borda rule.

4.2.4.1 *Three Alternatives*

Table 4.1 presents the results of the experiments for the case of three alternatives.

As can be seen from the table, in this case, for most profiles one justification can be found within a reasonable amount of time: the absolute maximum time encountered is barely more than half a minute. The average is consistently within seconds, even when accounting for the standard deviation. This is true for both the exhaustive generation and the sampling approaches. Note that, as the size of the scenario increases, the standard deviation rises quite sharply, becoming greater than the average for every scenario with 5 or more voters. This means that there are some outliers in the data: indeed, consider the difference in average and maximum time. Although most of the time we can explain the profiles quite quickly (low average), in most cases the maxi-

25 `https://userinfo.surfsara.nl/systems/lisa`

Method (Voters)	Explained profiles	Average time (s)	Maximum time (s)
Exhaustive			
2 voters	5/5 (100%)	0.11 ($\pm$0.01)	0.13
3 voters	9/10 (88.89%)	0.12 ($\pm$0.01)	0.14
4 voters	22/23 (95.65%)	0.14 ($\pm$0.13)	0.77
5 voters	42/42 (100%)	0.23 ($\pm$0.31)	1.50
6 voters	83/83 (100%)	0.23 ($\pm$0.37)	2.33
7 voters	132/132 (100%)	0.41 ($\pm$0.69)	4.02
8 voters	222/222 (100%)	0.45 ($\pm$1.06)	9.99
IC			
10 voters	180/180 (100%)	0.40 ($\pm$1.12)	10.74
12 voters	180/180 (100%)	0.93 ($\pm$1.99)	9.26
PE			
10 voters	180/180 (100%)	0.82 ($\pm$2.61)	30.16
12 voters	180/180 (100%)	1.42 ($\pm$4.05)	33.25
SP			
10 voters	180/180 (100%)	0.47 ($\pm$1.48)	9.34
12 voters	180/180 (100%)	0.66 ($\pm$3.13)	25.44
Pf			
10 voters	180/180 (100%)	0.48 ($\pm$1.28)	9.36
12 voters	180/180 (100%)	1.19 ($\pm$3.95)	25.69

Table 4.1: Results of the experiments for the case of three alternatives. *IC* is the impartial culture model and *PE* is the Polya-Eggenberger urn model. Moreover, *SP* and *Pf* are the single-peaked and `Preflib`-based generation methods, respectively. The average time is presented with standard deviation in brackets. For the exhaustive generation, one profile per equivalence class is tested.

mum time is much greater (e.g., half a minute). The presence of these outliers explains the high value of the standard deviation.

Notice that the only unjustifiable (equivalence classes of) profiles are found within the cases of 3 and 4 voters. For 3 voters, a representative of the only unjustifiable class is the following profile:

$$\#1 : a \succ b \succ c$$
$$\#2 : c \succ a \succ b$$

Let this profile be called $\mathbf{R}_3$. Notice that the Borda winners, a and c, could be explained with the classical (stronger) formulation of REINFORCEMENT (together with NEUTRALITY, the PARETO PRINCIPLE and CANCELLATION) if we consider a profile of 5 voters. We will now sketch how to do so.

Example 4.3. Let $\mathbf{R}_5$ be:

$$\begin{aligned}\#1 &: a \succ b \succ c\\ \#2 &: c \succ a \succ b\\ \#1 &: a \succ c \succ b\\ \#1 &: b \succ c \succ a\end{aligned}$$

This profile can be partitioned into the two following subprofiles:

$$\begin{array}{l}\#1 : a \succ b \succ c\\ \#1 : c \succ a \succ b\\ \#1 : b \succ c \succ a\\ \hline \#1 : c \succ a \succ b\\ \#1 : a \succ c \succ b\end{array}$$

For the first subprofile, by NEUTRALITY, the outcome must be $\{a, b, c\}$. This was shown multiple times in this work (for instance, see Example 2.2). The second subprofile must have outcome $\{a, c\}$, as either both or none of these alternatives must win (again, by NEUTRALITY) and b is Pareto dominated. Thus, by REINFORCEMENT, $F(\mathbf{R}_5) = \{a, c\}$.

However, $\mathbf{R}_5$ can also be partitioned as:

$$\begin{array}{l}\#1 : a \succ b \succ c\\ \#2 : c \succ a \succ b\\ \hline \#1 : a \succ c \succ b\\ \#1 : b \succ c \succ a\end{array}$$

Note that, here, the first subprofile is $\mathbf{R}_3$, the profile for which we are trying to justify an outcome. Since, by CANCELLATION, in the second subprofile all alternatives win, whatever wins in $\mathbf{R}_3$ must also win in $\mathbf{R}_5$ (by REINFORCEMENT). Therefore, $F(\mathbf{R}_3) = \{a, c\}$, as every other outcome would contradict the fact that $F(\mathbf{R}_5) = \{a, c\}$, derived above. △

Let us now consider the case of 4 voters. In this case, a representative of the unjustifiable class of profiles is (let it be $\mathbf{R}_4$):

$$\#1 : a \succ b \succ c$$
$$\#3 : c \succ a \succ b$$

This profile is similar to $\mathbf{R}_3$, but with an additional preference order $c \succ a \succ b$. Indeed, one could use the previous explanation for $F(\mathbf{R}_3) = \{a, c\}$ (that required a profile with 5 voters) and the fact that $F(c \succ a \succ b) = \{c\}$ (by Faithfulness or the Pareto Principle) to show that, by Reinforcement, $F(\mathbf{R}_4) = \{c\}$ holds.

We can now explain why all scenarios from 5 voters are completely explainable. Recall that all profiles up to 4 voters are explainable when dealing with a voting scenario of 5 voters and 3 alternatives (as the only unexplainable profiles, $\mathbf{R}_3$ and $\mathbf{R}_4$, can be explained using 5 voters). However, one can also verify that, for all profiles $\mathbf{R}$ of 5 voters, there exists a pair of subprofiles $\mathbf{R}'$ and $\mathbf{R}''$ such that $\mathbf{R}' + \mathbf{R}'' = \mathbf{R}$ and $\mathit{Borda}(\mathbf{R}') \cap \mathit{Borda}(\mathbf{R}'') = \mathit{Borda}(\mathbf{R})$. To verify this, one can exhaustively enumerate, for every profile with 5 voters, all its partitions into two subprofiles, and check whether at least one good partition exists. Thus, one can use the explanations for the Borda outcome for $\mathbf{R}'$ and $\mathbf{R}''$ (which as discussed above, surely exist) to explain, by Reinforcement, the Borda outcome for $\mathbf{R}$.

One can show that a similar argument holds for at least all scenarios of three alternatives with 6, 7, 8 and 10 voters. That is, for any profile $\mathbf{R}$ in these scenarios, we can find a partitioning into two subprofiles $\mathbf{R}'$ and $\mathbf{R}''$ such that $\mathtt{Borda}(\mathbf{R}) = \mathtt{Borda}(\mathbf{R}') \cap \mathtt{Borda}(\mathbf{R}'')$ and where the Borda outcome for both $\mathbf{R}'$ and $\mathbf{R}''$ can be automatically justified. This was computationally checked (again, by exhaustively enumerating all possible partitions). For 12 voters, our computational resources did not allow for a throughout check; however, for all profiles that we generated for our experiments, this property holds. Note that, for example, this does not hold for all profiles of three voters and three alternatives (not surprisingly, since we were not able to justify all of these profiles). Indeed, consider the aforementioned unjustifiable profile $\mathbf{R}_3$:

$$\#1 : a \succ b \succ c$$
$$\#2 : c \succ a \succ b$$

For this profile, the above property does not hold: no pair of subprofiles of this profile can be used to justify $\{a, c\}$ through Reinforcement.

Before moving to the case of four alternatives, let us briefly comment on the different generation approaches. Since we can justify all the profiles that we generated, we can only comment about the different time performances of the four methods. Looking at Table 4.1, we can notice that the IC, single-peaked, and `Preflib` generation approaches all obtain comparable average and maximum times. In contrast, PE seems to have slightly worse performance, both in terms of average and maximum time. This might mean that, on average, profiles generated through this culture are harder to explain. That is, we might have to generate larger portions of the instance graph. Indeed, we have found that, in these experiments, PE was the only culture requiring explanations of depth 4. Specifically, among the PE-generated profiles, there was one profile requiring an explanation of depth 4 for both the case of 10 and 12 voters. Every other profile (also regarding the other generation methods) was explained within a depth of 3. Furthermore, let us look at Table 4.2. This table shows the average depth needed to explain profiles for each generation method (aggregated over the 10 and 12 voters cases). From this data, we can notice that, as we said, PE seems to require on average deeper explanations. Moreover, PE has the highest standard deviation, which is consistent with the fact that this culture was the only one requiring a depth of 4 (that is, PE has more outliers). As we will see, the experiments for four alternatives will be consistent with this, as PE will be the culture with the lowest performance in terms of explainability (that is, in terms of the percentage of justified profiles). Moreover, it seems that the single-peaked and IC generation methods require the shallowest explanations, on average. As we shall see, this seems not to match our results for the case of four alternatives, where `Preflib`-generated profiles were the easier to explain.

Gen. Method	**Avg. Depth**
IC	0.83 ($\pm$ 0.63)
PE	1.15 ($\pm$ 0.82)
SP	0.79 ($\pm$ 0.60)
Pf	0.98 ($\pm$ 0.64)

Table 4.2: Average depth needed to explain the profiles in each generation method (three alternatives case, 10 and 12 voters cases aggregated). *IC* is the impartial culture model and *PE* is the Polya-Eggenberger urn model. Moreover, *SP* and *Pf* are the single-peaked and `Preflib`-based generation methods, respectively. Standard deviation is shown in brackets.

4.2.4.2 *Four Alternatives*

Table 4.3 presents the results of the experiments for the case of four alternatives.

As can be seen, compared to the case of three alternatives, we have a much harder computational problem. In this case, the increase in the number of voters results in a stark rise in computing time. Indeed, recall that the number of possible profiles increases drastically as the number of alternatives rises (see Table 2.1). A detailed analysis of the results follows.

Method (Voters)	**Explained profiles**	**Average time (s)**	**Maximum time (s)**
Exhaustive			
2 voters	15/17 (88.24%)	0.21 (± 0.09)	0.35
3 voters	68/111 (61.26%)	1.11 (± 1.14)	4.95
4 voters	509/762 (66.80%)	3.15 (± 3.82)	16.76
IC			
4 voters	124/180 (68.89%)	3.05 (± 3.82)	15.67
5 voters	113/180 (62.78%)	12.40 (± 12.67)	42.12
6 voters	132/180 (73.33%)	16.58 (± 21.92)	77.32
7 voters	140/180 (77.78%)	63.23 (± 65.19)	183.15
8 voters	167/180 (92.78%)	155.85 (± 189.46)	545.78 [9′05″]
PE			
4 voters	112/180 (62.22%)	3.18 (± 3.57)	14.95
5 voters	115/180 (63.89%)	10.74 (± 10.74)	40.13
6 voters	122/180 (67.78%)	23.36 (± 27.59)	99.40
7 voters	110/180 (61.11%)	42.12 (± 51.36)	182.15
8 voters	136/180 (75.56%)	109.83 (± 150.79)	539.60 [9′00″]
SP			
4 voters	143/180 (79.44%)	1.34 (± 2.11)	9.14
5 voters	115/180 (63.89%)	7.52 (± 8.55)	38.29
6 voters	132/180 (73.33%)	10.16 (± 14.71)	70.56
7 voters	134/180 (74.44%)	21.22 (± 28.69)	124.95
8 voters	140/180 (77.78%)	40.57 (± 64.23)	277.81 [4′38″]
Pf			
4 voters	149/180 (82.78%)	1.93 (± 3.35)	16.18
5 voters	133/180 (73.89%)	7.18 (± 10.09)	40.82
6 voters	145/180 (80.56%)	14.40 (± 22.84)	91.57
7 voters	149/180 (82.78%)	25.83 (± 43.37)	176.67
8 voters	161/180 (89.44%)	57.49 (± 103.45)	493.46 [8′13″]
Total			
–	3264/4490 (72.69%)	25.69 (± 69.03)	545.78 [9′05″]

Table 4.3: Results of the experiments for the case of four alternatives. *IC* is the impartial culture model and *PE* is the Polya-Eggenberger urn model. Moreover, *SP* and *Pf* are the single-peaked and `Preflib`-based generation methods, respectively. The average time is presented with standard deviation in brackets. For the exhaustive generation, one profile per equivalence class is tested. For the slowest test cases, the approximated time in minutes is reported in square brackets. We also aggregate the results across all cultures and scenarios (last row).

First of all, we can notice that, for all the experiments considered, the majority of profiles (73%) can be explained within a reasonable amount of time. This can be seen from the final row, which aggregates all results. Furthermore, all queries are answered within minutes: the absolute worst time is slightly more than 9 minutes. Although this is too much for a real-time application, we can envision a tool that works in an asynchronous fashion. For example, we could think of a web application that returns the result via e-mail. A further general tendency to notice regards the number of voters. As the number of voters increases, the percentage of justified profiles (although with exceptions) generally rises as well. This mirrors the pattern observed in the case of three alternatives, where the only unjustifiable profiles were to be found in the smaller scenarios.

One intuitive explanation for this trend could be the following. As we will see in the next chapter, the Reinforcement axiom plays a crucial role in the explanatory power of our corpus. Intuitively, most justifications are obtained by "breaking down" the goal profile into two subprofiles that have a computable justification (that is, for which we can find a justification with our approach). In this case, if the two subprofiles have some winners in common, we can use Reinforcement to justify an outcome for the goal profile.

As the size of the voting scenario increases (that is, the size of the goal profile, $\mathbf{R}^\star$, increases), the likelihood of finding a justification with the above structure increases as well. One reason for this is simply that, the bigger a profile is, the more way there are to partition it into two subprofiles. Hence, the more likely it is that at least one partition leads to a justification. Furthermore, with a larger voting scenario, some subprofiles have a computable justification that could not be retrieved in smaller scenarios. This is because some profiles need to refer to a bigger profile through Reinforcement (see, for example, the explanation involving the unjustifiable profile $\mathbf{R}_3$ in Example 4.3). Thus, the larger the scenario is, the more likely is it that these explanations will be computable for these subprofiles.

Let us now have a look at the different generation approaches. Contrary to the case of three alternatives, here not all profiles have been explained. Thus, we can comment on the differences in performance (in terms of explainability) between the four methods.[26]

First of all, we notice that the four methods exhibit different results. Although the number of profiles analysed is limited (due to the amount of computational resources needed), we can already glimpse some interesting patterns. For example, we notice that `Preflib`-generated profiles are generally easier to explain. This result, albeit limited to our specific corpus, is encouraging, as our goal is to explain real-world data. Indeed, they give the best results (in terms of percentage of justified profiles) in all sce-

26 For this case, we do not report the average depth of the explained profiles. Indeed, since we imposed a maximum depth and the four methods have different percentages of explained profiles, this data would not be a good indicator of the explainability of the four methods.

narios, save for the case of 8 voters (where IC obtains the best results). Furthermore, it is surprising to see that PE seems not to lead to better results compared to IC. In fact, it seems to be the worst culture in terms of explainability. This result correlates with the results obtained for the case of three alternatives, where PE profiles seemed to require on average more time to be explained. Thus, our intuition that it better captures the structure of real-world data (if compared to IC) turns out to be wrong, at least with respect to explainability. A possible explanation for this is how PE crudely replicates the preferences of voters, without actually inducing any further structure: that is, when generating with PE, two preference orders are either identical or absolutely independent from one another. This seems not to capture real-world preferences, as one would assume that some voters might share extremely similar views, while only disagreeing about the relative ordering of *some* alternatives.

We also notice that single-peaked profiles seem to be slightly easier to justify than PE, and overall comparable to IC. Thus, we find that constraining the structure of profiles might lead to better results than just reducing the number of unique ballots (which is what PE does). More radically, it seems that greater values of α (that is, a greater correlation between voters) does not *always* lead to an increased percentage of explained profiles (Table 4.4). Although a bigger number of profiles should be considered to make a conclusive remark, these early results seem to suggest that it is not the number of unique ballots that matters, but rather the structure of profiles.

Contagion Rate	**Profiles**
0.01	109/150 (73%)
0.02	111/150 (74%)
0.05	104/150 (69%)
0.1	95/150 (63%)
0.2	86/150 (57%)
0.5	90/150 (60%)

Table 4.4: Percentage of profiles explained per α value in our experiments. We have aggregated all results (from 4 to 8 voters) in one number to show the overarching trend.

This raises the question: what structural quality makes `Preflib`-generated profiles easier to explain? First of all, one needs to remember that the answer to this question has to be related to the corpus used for these experiments. Indeed, what is easy to explain for this selection of axioms might be unjustifiable for a different one. Intuitively, whether a profile has a computable justification is merely a question of whether this profile exhibits a certain *structure* that one or more axioms can predicate over. In other words, each of the studied axioms can "match" a certain pattern of profiles. For instance, Cancellation can only constrain profiles where all alternatives tie in a majority contest.

Consequently, it seems reasonable to investigate what makes `Preflib`-generated profiles easier to explain *in relation to the Borda winner*. This is because, as we have said before, all justified profiles have the Borda winner as the outcome, due to our choice of axioms. In the upcoming section, we propose two ways to investigate this.

4.2.4.3 *Investigating Preflib Profiles*

In the following, we try to investigate the explainability of `Preflib`-generated profiles in relation to our corpus (i.e., to the Borda rule). As all of the profiles with three alternatives were explained, we focus here on the case of four alternatives. We propose two ways to do so.

CONDORCET AND BORDA. One first natural explanation for why it might be easier to justify a `Preflib`-generated profile is that real-world profiles might be, in some sense more "well-behaved". That is, for real elections, it might be much more likely for an "uncontroversial winner" to exist. By uncontroversial winner, intuitively, we mean an outcome it is extremely easy to argue in favour of. The most well-known notion of such a winner in the literature is probably the notion of Condorcet winner. Indeed, as it is generally understood, a Condorcet winner (when it exists) is considered an extremely reasonable outcome.[27] This is especially interesting in relation to our study if we note that pathological cases, such as the *Condorcet paradox* (where there is no Condorcet winner), might be much more common in theoretical considerations rather than real-world elections (Gehrlein, 2006; Regenwetter et al., 2006; Van Deemen, 2014). In light of this, we investigate whether the existence of a Condorcet winner is related to the increased explainability of `Preflib`-generated profiles.

We could, then, test if it is easier to justify profiles with a Condorcet winner. However, since our algorithm (due to the choice of axioms) seems to always justify the Borda winner, we focus on the following property. We check the relative frequency of finding a justification when the Borda rule elects the Condorcet winner as the sole winner. That is, a Condorcet winner exists *and* Borda chooses it as the sole winner. We call this frequency $p(J \mid B = C)$. Conversely, $p(J \mid B \neq C)$ is the relative frequency of finding a justification when the Borda rule and the Condorcet winner disagree (or when the Condorcet winner does not exist). Both these frequencies were computed from the data resulting from the above experiments. We then relate this to the relative frequency, for each statistical culture, of the Condorcet winner being the sole Borda outcome (called $p(B = C)$). We do so to understand whether this could explain the observed differences in explainability. This last frequency, for every culture we consider, was computed on a set of 100 000 randomly generated profiles, to give a more precise account.

27 Notice that some have criticised the appeal of Condorcet winners (and thus, of the CONDORCET PRINCIPLE): for example, Fishburn (1974) constructs a large profile with a Condorcet winner, and argues that it might not be the most reasonable winner for that particular profile.

Scenario	**Probabilities**	**Gen. Method**			
		IC	*PE*	*SP*	*Pf*
4 voters	$p(J \mid B = C)$	0.91	0.85	0.90	0.96
	$p(J \mid B \neq C)$	0.56	0.44	0.63	0.59
	$p(B = C)$	0.34	0.43	0.53	**0.62**
5 voters	$p(J \mid B = C)$	0.79	0.77	0.80	0.83
	$p(J \mid B \neq C)$	0.14	0.17	0.00	0.09
	$p(B = C)$	0.73	0.75	0.85	**0.86**
6 voters	$p(J \mid B = C)$	0.95	0.85	0.88	0.86
	$p(J \mid B \neq C)$	0.58	0.54	0.48	0.67
	$p(B = C)$	0.39	0.48	0.63	**0.71**
7 voters	$p(J \mid B = C)$	0.89	0.75	0.81	0.93
	$p(J \mid B \neq C)$	0.52	0.21	0.19	0.34
	$p(B = C)$	0.71	0.74	0.85	**0.87**
8 voters	$p(J \mid B = C)$	0.96	0.82	0.88	0.91
	$p(J \mid B \neq C)$	0.90	0.68	0.54	0.83
	$p(B = C)$	0.43	0.52	0.68	**0.77**
total	$p(J \mid B = C)$	0.88	0.75	0.85	0.90
	$p(J \mid B \neq C)$	0.58	0.36	0.45	0.56
	$p(B = C)$	0.52	0.58	0.71	**0.77**

Table 4.5: Results of our experiment about Condorcet winners and Borda winners (four alternatives). *IC* is the impartial culture model and *PE* is the Polya-Eggenberger urn model. Moreover, *SP* and *Pf* are the single-peaked and `Preflib`-based generation methods, respectively. We include an extra row showing the total results (i.e., aggregated across all the scenarios). The bold case for $p(B = C)$ shows, across the four generation approaches, which has the highest relative frequency (per row).

Table 4.5 shows the results of this experiment. We can notice two main aspects. Firstly, this early data suggests that it is easier to find a justification when Borda agrees with the Condorcet winner. Indeed, for all scenarios considered, the frequency of finding a justification when this happens is relatively high (the lowest being 0.75, generally above). Furthermore, this is usually comparatively higher than the opposite case. Notice that this holds even when $p(B = C)$ is low: for example, in the case of 4 voters, for IC, this frequency is only 0.34. However, in the same case, $p(J \mid B = C)$ is 0.91.

The second thing to notice is that this phenomenon happens consistently more often in `Preflib`-generated profiles. However, from this data alone we cannot conclude that this is the reason for the better performance obtained in this case. Indeed, notice how $p(B = C)$ is consistently higher for PE generation than for IC. This is not consistent with our previous findings, where IC seemed to perform better than PE. Further, notice how this frequency is, for all the cultures, generally lower for scenarios with an even number of voters. This might be related to the fact that in these scenarios it is much easier to obtain a tie (for instance, Cancellation profiles can only happen with an even number of voters). However, we do not register in our experimental results (Table 4.3) this difference in explainability between profiles with an odd and even number of voters.

All in all, we register a correlation between explainability with this corpus and the presence of a Condorcet winner (that the Borda rule elects). However, more investigation is warranted to understand the correlation between this and the explainability of the different statistical cultures.

SUBPROFILE CONSISTENCY. In the above, we argued that, for well-behaved profiles, it might be easier to come up with a justification for at least one outcome. We then proposed a notion of well-behavedness based on the Condorcet winner. We now propose an alternative notion, based on the ideas of stability and consistency. We propose a measure that, for a profile $\mathbf{R}$, counts the number of subprofiles $\mathbf{R}'$ of $\mathbf{R}$ such that the Borda outcome for $\mathbf{R}'$ is a superset of that for $\mathbf{R}$. We focus on the Borda outcome, as again, in our experiments we were only able to justify this outcome.

The rationale behind this measure is the following. Intuitively, this notion tries to capture some notion of *stability* (with respect to voter withdrawal) or *consistency* (between a profile and its parts).[28] Since the notion of subprofile is central to our justification algorithm (given the choice of axioms), we hypothesise that this measure of "subprofile consistency" might correlate positively with the explainability of a profile.

Let us introduce our measure formally. First, let $\mathcal{S}(\mathbf{R})$ be the set of all possible subprofiles of $\mathbf{R}$:

$$\mathcal{S}(\mathbf{R}) = \{\, \mathbf{R}_1 \in \mathcal{R}(X)^+ \mid \mathbf{R}_1 + \mathbf{R}_2 = \mathbf{R},\ \text{for some } \mathbf{R}_2 \in \mathcal{R}(X)^+ \,\}$$

Then, we define our measure $\mathcal{M}(\mathbf{R})$ as:

$$\mathcal{M}(\mathbf{R}) = \frac{|\{\, \mathbf{R}' \in \mathcal{S}(\mathbf{R}) \mid \mathtt{Borda}(\mathbf{R}) \subseteq \mathtt{Borda}(\mathbf{R}') \,\}|}{|\mathcal{S}(\mathbf{R})|}$$

To give a further intuition about this measure, suppose that we were to stop a group of voters outside of an election booth and ask about their casted ballots. Knowing that we will use the Borda rule to select the winner, we compute the Borda outcome for the (incomplete) preferences we polled. Let this outcome be the *projected outcome*. This measure gives the likelihood that the real outcome of the elections will be a subset of

28 Indeed, the REINFORCEMENT axiom was originally called CONSISTENCY by Young (1974), and is based on an intuition related to ours, capturing the relationship between a profile and its subprofiles.

the projected outcome. In other words, this is the probability that the real winners will be all among the projected winners.

Similar to the previous paragraph, we will now study the correlation between this measure and the explainability of a profile. To do so, we will present the average value of this measure for profiles that we were able to justify (*justified profiles*) and for profiles we were unable to (*unjustified profiles*). Further, we will show the average measure for the different statistical cultures (*culture*), in order to understand whether this could be related to the observed differences in explainability. This last value was computed on a set of 100 000 randomly generated profiles.

Scenario	**Average $\mathcal{M}(\mathbf{R})$ for...**	**Gen. Method**			
		IC	*PE*	*SP*	*Pf*
4 voters	justified profiles	0.72	0.77	0.75	0.81
	unjustified profiles	0.42	0.40	0.44	0.39
	culture	0.62	0.65	0.70	**0.75**
5 voters	justified profiles	0.74	0.77	0.82	0.84
	unjustified profiles	0.43	0.45	0.47	0.54
	culture	0.61	0.66	0.72	**0.77**
6 voters	justified profiles	0.69	0.73	0.79	0.83
	unjustified profiles	0.42	0.46	0.48	0.53
	culture	0.61	0.65	0.71	**0.77**
7 voters	justified profiles	0.64	0.74	0.80	0.82
	unjustified profiles	0.34	0.48	0.56	0.45
	culture	0.60	0.65	0.72	**0.78**
8 voters	justified profiles	0.59	0.68	0.79	0.83
	unjustified profiles	0.48	0.52	0.50	0.60
	culture	0.60	0.65	0.72	**0.79**
total	justified profiles	0.67	0.80	0.79	0.83
	unjustified profiles	0.40	0.49	0.49	0.50
	culture	0.61	0.65	0.71	**0.77**

Table 4.6: Results of the experiments about our measure $\mathcal{M}(\cdot)$ (four alternatives). *IC* is the impartial culture model and *PE* is the Polya-Eggenberger urn model. Moreover, *SP* and *Pf* are the single-peaked and `Preflib`-based generation methods, respectively. We include an extra row showing the total results (i.e., aggregated across all the scenarios). The bold case for the *culture* rows shows, across the four generation approaches, which has the highest value (per row).

Table 4.6 shows the results of this experiment. Overall, we find results similar to the previous experiment. First, we can notice how, across all scenarios, the average of this measure is consistently higher for the justified profiles than it is for the unjustified profiles. This suggests, indeed, that this measure positively correlates with the explainability of a profile. Furthermore, we can see that, for all scenarios considered, `Preflib`-generated profiles have the greatest average measure. Hence, it seems that this measure might in part explain the better performance obtained with this statistical culture. However, we observe the same issues as the previous experiment. Indeed, IC has generally the lowest scores, which contradicts the results of Table 4.3. Therefore, more research is warranted to understand the link between this measure and explainability.

4.2.4.4 *Comparison with the Base Justification Algorithm*

Finally, let us now briefly compare our results with the results of Boixel and Endriss (2020). In their experimental study, they focused on the scenario with 3 voters and 3 alternatives, and exhaustively generated all profiles of that scenario. More precisely, similar to what we did, they generated one profile per equivalence class (due to the symmetries in the space of all profiles). Furthermore, they worked with a corpus similar to our own. The only differences, as already mentioned, were our exclusion of the Condorcet Principle and the inclusion of Positive Responsiveness. Their corpus also included Anonymity, which in our case was built-in in our definition of SCFs. Notice that our exclusion of the Condorcet Principle does not significantly impact the performance, as it is mostly interprofile axioms that determine the complexity of the graph generation phase.

We find that our approach leads to a dramatic improvement in performance. According to the authors, their algorithm can return at least one justification for all profiles of that scenario in 5 to 30 minutes. With our approach, we can do the same in less than 0.14 seconds (with an extra interprofile axiom, Positive Responsiveness).[29] Further, their approach does not scale beyond such restricted scenarios. Indeed, although we did not run an extensive study of the performance of their algorithm, handling scenarios with 5 or more voters seems unfeasible in a reasonable amount of time, even in the three alternatives case. More radically, justifying profiles with 10 to 12 voters seems inconceivable for the original approach. All in all, we find our approach to be an improvement in computational terms.

Note that the investigation conducted by the aforementioned authors focused more on a qualitative study of the retrieved justifications rather than a throughout study of the performance of their algorithm. We will focus on similar aspects in the following chapter.

29 To be precise, the two experimental studies were executed on two different machines. However, our results, especially for that scenario, do not change noticeably if executed on less powerful machines. We do not consider the difference in computational power as significant in determining the gain in performance.

4.2.5 *Addressing the Research Question*

The purpose of this experimental work was to directly address the research question. We consider our results a positive answer to it. Indeed, we have shown how it is possible to implement a justification algorithm capable of handling moderately-sized instances. For the case of three alternatives, the overwhelming majority of the problem instances are handled within seconds. For the case of four alternatives, it seems that the algorithm is capable of quickly justifying real-world data in most cases. This is especially true for the larger scenarios considered. Furthermore, we highlighted some interesting trends in our data, noticing that `Preflib`-generated profiles seem to be, in general, easier to explain. However, more experiments are warranted to confirm or reject our observations on the different performances registered across different statistical cultures.

As a final comment, we once again stress that our results are highly dependant on the corpus we used. Indeed, at least in principle, one could design a set of much stronger axioms that could be used to justify at least one outcome for every profile. However, all of our axioms are well-known axioms taken from the literature.[30] Therefore, although not universal, our findings still show that for at least *some* reasonable and widely accepted axioms the automatic justification of collective decisions is viable.

30 Save for REINFORCEMENT, which we slightly adapted for our purposes.

5 A CLOSER LOOK AT JUSTIFICATIONS

In this chapter, we are going to have a better look at some justifications. More precisely, we are going to review some common patterns of explanation that occur when using the corpus we employed so far. Furthermore, we will briefly discuss the notion of explanation, and what makes something *explanatory*. This, we argue, might help us to develop better justification algorithms.

The explicit objective of this chapter is to facilitate further research in this area, by gaining insight into how each axiom can be used in a justification, and how different axioms interact. This, in turn, can help in the discovery of new approaches or more axiom-specific heuristics. Notice that this chapter does not aim to be (and perhaps *cannot* be) an exhaustive list of all possible "explanation structures" that we can find with this corpus (whatever that precisely means). Rather, it is an enumeration of the most common kinds of justifications that we found, to exemplify what sort of explanations our method can find with these axioms.

5.1 PATTERNS OF EXPLANATION

In this section, we will show some common patterns of explanation. Again, the purpose of this chapter is to give an insight into what kind of arguments and proof structures our algorithm can produce. To do so, we will give multiple examples of possible explanations involving these patterns. Consequently, we believe that it would be detrimental to the presentation to show the explanations in formal terms, that is, as sets of axiom instances. Instead, we will write the explanations in natural language, in the form of high-level mathematical proofs. The underlying arguments will be, of course, the same. We believe that this gives a better understanding of the explanations we show.

Note that we do not aim at defining formally what a pattern is here. Intuitively, by *pattern of explanation* we mean some *argumentative structure* or *scheme of interaction* between the instances of some axioms that have been observed in the explanations of multiple justifications. Note that an explanation can be composed of multiple patterns. To see why, consider that to justify the target outcome for the goal profile, we might have to determine the outcomes of some other profiles needed in the main explanation. Intuitively, for all of these other profiles, we can have a "subexplanation", composed of one or more patterns. For this reason, when giving the examples later on, we will not refer to the goal profile, but rather, to an arbitrary profile (as these patterns need not refer to the goal profile).

However, such patterns can also be used to *eliminate* a given outcome. In other words, a (sub)explanation might also be used to rule out a certain outcome for a given profile. Note that this can lead to a justification of a particular outcome $X^\star$ for some profile $\mathbf{R}$. Indeed, suppose that we eliminate all non-empty outcomes except for $X^\star$. Then, we can appeal to the fact that at least one alternative should win in $\mathbf{R}$ to show that $X^\star$ must be the outcome.

Thus, in the following, we will divide our presentation into two parts. First, we will show some patterns that can be used to directly determine the outcome. Then, we will show some patterns that can be used to eliminate outcomes. Finally, notice that all patterns that we will show only involve the axioms we employed in our experiments, namely, NEUTRALITY, the PARETO PRINCIPLE, FAITHFULNESS, CANCELLATION, REINFORCEMENT and POSITIVE RESPONSIVENESS.[1]

5.1.1 *Direct Winner Determination*

In the following, we show some common patterns that can be used to directly determine the outcome.

5.1.1.1 *Immediate Application of Intraprofile Instances*

Arguably, the simplest kind of explanation pattern is to be found in those cases where an outcome for a given profile $\mathbf{R}$ can be justified with one single intraprofile axiom instance. In our corpus, this happens when $\mathbf{R}$ has a single voter (explained through FAITHFULNESS), or when all alternatives tie in a majority contest (explained with CANCELLATION).

Example 5.1. Consider the profile $\mathbf{R} = \{ a \succ b \succ c \}$. Here, there is only one voter. By FAITHFULNESS, we have that a (their top choice) must be the sole winner. △

Another example could be a profile with a Condorcet winner (if we were to include the CONDORCET PRINCIPLE in our corpus).

5.1.1.2 *Raising a Winner*

Suppose we can justify that $x^\star \in F(\mathbf{R})$. If a profile $\mathbf{R}^+$ can be obtained by raising the support of $x^\star$, then by POSITIVE RESPONSIVENESS, we can justify that $F(\mathbf{R}^+) = \{ x^\star \}$.

In our experiments, this pattern emerged most commonly when a non-singleton outcome (containing $x^\star$) has been determined for $\mathbf{R}$. For instance, if $\mathbf{R}$ is a Cancellation profile, or when multiple alternatives can be justified through SYMMETRY (the axiom derived from NEUTRALITY).

1 Also, recall that our definition of SCF already includes a notion of *anonymity*, which is usually encoded as a separate axiom (called, indeed, ANONYMITY).

Example 5.2. Consider the profile $\mathbf{R}$:

$$\#1 : a \succ c \succ b \succ d$$
$$\#1 : b \succ c \succ a \succ d$$
$$\#1 : c \succ a \succ b \succ d$$

Notice that $\mathbf{R}$ can be obtained by raising the support of c from profile $\mathbf{R}'$:

$$\#1 : a \succ b \succ c \succ d$$
$$\#1 : b \succ c \succ a \succ d$$
$$\#1 : c \succ a \succ b \succ d$$

In $\mathbf{R}'$, by the PARETO PRINCIPLE, d cannot win. Further, by SYMMETRY, a, b and c must either all win or all lose. Thus, since at least one alternative must win, we have that $F(\mathbf{R}') = \{a, b, c\}$. Hence, by POSITIVE RESPONSIVENESS, we have that $F(\mathbf{R}) = \{c\}$. △

5.1.1.3 *Composition by Reinforcement*

Suppose that $\mathbf{R}$ can be partitioned into two profiles, $\mathbf{R}_1$ and $\mathbf{R}_2$, such that in both subprofiles an outcome can be justified. Let these outcomes be $X_1 \subseteq X$ and $X_2 \subseteq X$, respectively. If $X_1 \cap X_2 \neq \emptyset$, then we can justify outcome $X_1 \cap X_2$ for $\mathbf{R}$ via REINFORCEMENT.

This pattern is very generic, and in a sense, conceptually straightforward. It can be seen as a composition of other patterns: that is, we justify the two subprofiles through some pattern, and then compose this justification through REINFORCEMENT. In our opinion, this conceptual clarity makes this pattern also quite interesting from an explanatory point of view. In a sense, we can see an explanation of this form as a "divide-and-conquer" explanation: as long as the explanations for the subprofiles are clear, the composed explanation should be clear as well.

For example, one case that frequently occurs is the case where, say, for $\mathbf{R}_1$, we can justify outcome X (that is, all alternatives win), and for $\mathbf{R}_2$ we can justify $\{x^\star\}$ (for some $x^\star \in X$). Then, for $\mathbf{R}$, we can justify outcome $\{x^\star\}$ as well.

Example 5.3. Consider profile $\mathbf{R}$:

$$\#1 : a \succ b \succ c$$
$$\#1 : c \succ b \succ a$$
$$\#1 : a \succ c \succ b$$

Notice that this profile can be partitioned in profile $\{a \succ c \succ b\}$, where a wins by FAITHFULNESS, and profile $\{a \succ b \succ c, c \succ b \succ a\}$, where all alternatives win by CANCELLATION. Thus, by REINFORCEMENT, $\{a\}$ wins in $\mathbf{R}$. △

5.1.1.4 *Explanations By Exhaustion*

Suppose that we want to justify outcome $X^\star \subseteq X$ for profile $\mathbf{R}$ through some axioms $\mathcal{A}$ (with $\mathbb{I}(\mathcal{A}) \neq \varnothing$). To do so, this pattern employs the following argumentative structure:

(1) Consider another profile $\mathbf{R}'$.

(2) Consider all possible (non-empty) outcomes $X' \subseteq X$ for $\mathbf{R}'$.

(3) For every such outcome X', by appealing to $\mathcal{A}$, show that if X' wins in $\mathbf{R}'$ then $X^\star$ must win in $\mathbf{R}$.

(4) Since at least one non-empty outcome must win in $\mathbf{R}'$, conclude that $X^\star$ must win in $\mathbf{R}$.

In other words, we show that, regardless of what wins in $\mathbf{R}'$, $X^\star$ must win in $\mathbf{R}$. This is akin to a proof by exhaustion, where we enumerate all possible cases (i.e., all possible outcomes for $\mathbf{R}'$), and show that the goal statement holds for each one of them. To clarify this, we will now show an example. Since, in our experiments, we observed this pattern emerging almost exclusively with the axioms of Neutrality and Positive Responsiveness, we will focus on these axioms.

Example 5.4. Consider profile $\mathbf{R}$:

$$\#1 : a \succ b \succ c \succ d$$
$$\#1 : d \succ c \succ a \succ b$$

We will justify, for this profile, outcome $\{a\}$. To this end, consider the following profile, called $\mathbf{R}_1$:

$$\#1 : a \succ b \succ c \succ d$$
$$\#1 : d \succ c \succ b \succ a$$

Now, we will consider two cases: (1) $a \in F(\mathbf{R}_1)$ and (2) $a \notin F(\mathbf{R}_1)$. Trivially, there are no other cases. We will show that, in both cases, $F(\mathbf{R}) = \{a\}$ must hold.

(1) Assume that $a \in F(\mathbf{R}_1)$. Notice that $\mathbf{R}$ can be reached from $\mathbf{R}_1$ by raising the support of a. Thus, by Positive Responsiveness, $F(\mathbf{R}) = \{a\}$ must hold.

(2) Assume that $a \notin F(\mathbf{R}_1)$. Note that, by Symmetry (and thus by Neutrality), the only possible outcomes of $\mathbf{R}_1$ are $\{a, d\}$, $\{b, c\}$ and $\{a, b, c, d\}$ (to see why, notice that a and d are symmetric, and the same holds for b and c). Thus, if a cannot win, the only possible outcome for $\mathbf{R}_1$ is $\{b, c\}$. Now, consider profile $\mathbf{R}_2$:

$$\#1 : b \succ a \succ c \succ d$$
$$\#1 : d \succ c \succ b \succ a$$

Notice that $\mathbf{R}_2$ can be reached from $\mathbf{R}_1$ by raising the support of b. Therefore, by POSITIVE RESPONSIVENESS, $F(\mathbf{R}_2) = \{b\}$. Finally, consider the permutation $\pi : X \to X$ with $\pi(a) = b$, $\pi(b) = a$, $\pi(c) = c$ and $\pi(d) = d$ (that is, we swap a and b). Notice that $\pi(\mathbf{R}_2) = \mathbf{R}$. Therefore, by NEUTRALITY, $F(\mathbf{R}) = \{a\}$ must hold.

Thus, $F(\mathbf{R}) = \{a\}$ must hold. △

5.1.2 *Outcome Elimination*

In this section, we show some common outcome elimination patterns. These structures determine that a certain non-empty set of alternatives cannot be the outcome (of some profile). Note that, as said earlier, this can be used in a justification as follows. Consider profile $\mathbf{R}$ and an outcome $X^\star \subseteq X$. If we eliminate for $\mathbf{R}$ all non-empty $X' \subseteq X$ distinct from $X^\star$, then we have justified that $F(\mathbf{R}) = X^\star$. Note that for this to work we need to use the fact that $F(\mathbf{R}) = \varnothing$ cannot be the case.

5.1.2.1 *Pareto Principle*

The simplest pattern occurs when an alternative y is Pareto-dominated (that is, for some other alternative x, every voter agrees that $x \succ y$). In this case, we can use an instance of the PARETO PRINCIPLE to eliminate every outcome containing y.

Example 5.5. Consider profile $\mathbf{R}$:

$$\#1 : a \succ b \succ c$$
$$\#1 : a \succ c \succ b$$

Here, all voters agree that $a \succ b$. Further, they all agree that $a \succ c$. Therefore, by the PARETO PRINCIPLE, neither b nor c can be in $F(\mathbf{R})$, so the outcome must be $\{a\}$. △

5.1.2.2 *Raising a Loser*

Given a profile $\mathbf{R}$, suppose that, by raising some alternative $x^\star$, we obtain a profile $\mathbf{R}^+$ such that, for $\mathbf{R}^+$, we can justify an outcome $X' \subseteq X$ such that $\{x^\star\} \subset X'$ (i.e., $x^\star$ is *not* the only winner, as X' contains other alternatives). Then, surely $x^\star$ cannot win in $\mathbf{R}$: for if this were true, by POSITIVE RESPONSIVENESS $x^\star$ would be the sole winner in $\mathbf{R}^+$. However, this would contradict the fact that $F(\mathbf{R}^+) = X'$. Thus, we eliminate all outcomes containing $x^\star$ from the possible winners for $\mathbf{R}$.

Example 5.6. Consider profile $\mathbf{R}$:

$$\#1 : a \succ b \succ c$$
$$\#1 : c \succ a \succ b$$

We will eliminate every outcome containing b for $\mathbf{R}$. Now, consider profile $\mathbf{R}'$:

$$\#1 : a \succ b \succ c$$
$$\#1 : c \succ b \succ a$$

Note that $\mathbf{R}'$ is a Cancellation profile. Thus, $F(\mathbf{R}') = \{a, b, c\}$. Then, suppose $b \in F(\mathbf{R})$. Notice that we can obtain $\mathbf{R}'$ from $\mathbf{R}$ by raising the support of b. By POSITIVE RESPONSIVENESS, we have that $F(\mathbf{R}') = \{b\}$, which contradicts CANCELLATION. Thus, $b \notin F(\mathbf{R})$. △

5.1.2.3 *Symmetric Alternatives*

For a given profile $\mathbf{R}$, suppose that there is a permutation $\pi : X \to X$ such that $\pi(\mathbf{R}) = \mathbf{R}$. Then, as discussed when introducing the derived axiom of SYMMETRY (Section 4.1.4.1), it is possible to partition X into a set of equivalence classes based on π. For all classes $[x]_\pi$, by NEUTRALITY, either $[x]_\pi \cap F(\mathbf{R}) = \varnothing$ or $[x]_\pi \subseteq F(\mathbf{R})$ must hold. In other words, the alternatives in an equivalence class must either all win or all lose. Thus, for every non-empty set of alternatives $X' \subseteq X$, we eliminate X' if X' is neither an equivalence class nor the union of several classes.

Example 5.7. Consider profile $\mathbf{R}$:

$$\#1 : a \succ b \succ c \succ d$$
$$\#1 : b \succ a \succ d \succ c$$

Moreover, consider the mapping π such that $\pi(a) = b$, $\pi(b) = a$, $\pi(c) = d$ and $\pi(d) = c$ (that is, we swap a with b and c with d). Notice that $\pi(\mathbf{R}) = \mathbf{R}$. Further, π induces two equivalence classes: $\{a, b\}$ and $\{c, d\}$. By NEUTRALITY, we can eliminate the outcomes $X' \subseteq X$ that are not a union of equivalence classes. For example, $\{a, c\}$ cannot be the outcome, as it is not an equivalence class, and it cannot be obtained by combining several classes. Therefore, we are left with the following possible outcomes: $\{a, b\}$, $\{c, d\}$ and $\{a, b, c, d\}$. △

Notice that this pattern rests upon the assumption that voting rules are *anonymous*. Indeed, NEUTRALITY would not be enough for this pattern to function if we were to work with non-anonymous profiles. To explain this, we will show an example involving a non-anonymous profile.

Example 5.8. Let $N = \{1, \ldots, n\}$ be a set of n voters. Recall that $\mathcal{L}(X)$ is the set of all linear orders over X. A *non-anonymous* profile $\mathbf{P}$ for voters N and alternatives X is a vector of preference orders $(\succ_1, \ldots, \succ_n) \in \mathcal{L}(X)^n$.[2] Here, $\succ_i$ corresponds to

[2] We use the variable $\mathbf{P}$ instead of $\mathbf{R}$ to avoid confusion between non-anonymous profiles and anonymous profiles.

the preference order expressed by voter i. A *non-anonymous* SCF for voters N and alternatives X is a function $F : \mathcal{L}(X)^n \to 2^X \setminus \{\varnothing\}$. Finally, given a permutation of the alternatives $\pi : X \to X$ and a profile $\mathbf{P}$, we define $\pi(\mathbf{P})$ as the profile we obtain by replacing every occurence of x by $\pi(x)$ in $\mathbf{P}$. The axiom of NEUTRALITY (for non-anonymous SCFs) prescribes that, for every non-anonymous profiles $\mathbf{P}$ and $\mathbf{P}'$, if for some permutation $\pi : X \to X$ it holds that $\mathbf{P} = \pi(\mathbf{P}')$, then $F(\mathbf{P}) = \pi(F(\mathbf{P}'))$.

Then, consider the following non-anonymous profile $\mathbf{P}^\star = (\, a \succ_1 b, b \succ_2 a\,)$. For the anonymous variant of this profile, we argued multiple times in this work that, by NEUTRALITY, the outcome must be $\{\, a, b \,\}$. Notice that the same does not hold for the non-anonymous $\mathbf{P}^\star$. Indeed, consider the SCF F_1 that always elect the top-ranked alternative of voter 1 as the sole winner. Rules of this kind are known as *dictatorships* (famously characterised by Arrow (1951)). Notice that F_1 trivially does not satisfy ANONYMITY. Further, this rule satisfies NEUTRALITY: regardless of the names of the alternatives, it is always the top choice of voter 1 that wins. Importantly, note that $F_1(\mathbf{P}^\star) = \{\, a \,\}$.

Indeed, to show that $\{\, a, b \,\}$ must win in $\mathbf{P}^\star$, we need an extra axiom. Given a permutation of the voters $\tau : N \to N$, we define:

$$\tau((\succ_1, ..., \succ_n)) = (\succ_{\tau(1)}, ..., \succ_{\tau(n)})$$

That is, we swap the names of the voters according to τ. The axiom of ANONYMITY prescribes that, for every non-anonymous profiles $\mathbf{P}$ and $\mathbf{P}'$, if for some permutation $\tau : N \to N$ it holds that $\mathbf{P} = \tau(\mathbf{P}')$, then $F(\mathbf{P}) = F(\mathbf{P}')$. Intuitively, this means that the names of the voters should not matter: if we swap the names of voters, the outcome should not change.

With this, consider again profile $\mathbf{P}^\star = (\, a \succ_1 b, b \succ_2 a\,)$. Assume that the non-anonymous SCF F satisfies ANONYMITY and NEUTRALITY. For the sake of the contradiction, assume that $F(\mathbf{P}^\star) = \{\, a \,\}$. Now, consider profile $\mathbf{P} = (\, b \succ_1 a, a \succ_2 b\,)$, obtained by swapping the names of a and b. Clearly, by NEUTRALITY, $F(\mathbf{P}) = \{\, b \,\}$ must hold. But notice that, if we swap the names of voters 1 and 2 in $\mathbf{P}$, we obtain again $\mathbf{P}^\star$. Thus, by ANONYMITY, $F(\mathbf{P}^\star) = \{\, b \,\}$ must hold. Since we assumed that $F(\mathbf{P}^\star) = \{\, a \,\}$, we derived a contradiction. Therefore, $F(\mathbf{P}^\star) \neq \{\, a \,\}$. Note that a similar argument can be made to show that $F(\mathbf{P}^\star) \neq \{\, b \,\}$. Then, by ANONYMITY and NEUTRALITY, we have that $F(\mathbf{P}^\star) = \{\, a, b \,\}$ (as at least one alternative must win). $\triangle$

5.1.2.4 *Elimination by Contradiction via Reinforcement*

Consider a profile $\mathbf{R}_1$ and an outcome $X_1 \subseteq X$. We will eliminate X_1 for $\mathbf{R}_1$.

To this end, consider another profile $\mathbf{R}_2$ and the superprofile $\mathbf{R}_1 + \mathbf{R}_2$. Intuitively, this pattern works as follows. If we already know what wins in $\mathbf{R}_2$ and $\mathbf{R}_1 + \mathbf{R}_2$, we can eliminate every outcome for $\mathbf{R}_1$ that would contradict REINFORCEMENT. More formally, suppose that we justified some outcomes $X_2 \subseteq X$ and $X_{1+2} \subseteq X$ for $\mathbf{R}_2$ and $\mathbf{R}_1 + \mathbf{R}_2$, respectively. Then, if $X_1 \cap X_2 \neq \varnothing$ and $X_1 \cap X_2 \neq X_{1+2}$ hold, we can eliminate X_1 for $\mathbf{R}_1$, as this would contradict Reinforcement.

This is akin to a proof by contradiction, as we show that, were some "bad" outcome to win in $\mathbf{R}_1$, we would have a contradiction. This might sound rather abstract: let us see an example.

Example 5.9. Let $\mathbf{R}$ be the following:

$$\#1 : a \succ b \succ c$$
$$\#2 : c \succ a \succ b$$

We will use the above pattern to eliminate $\{b\}$ for $\mathbf{R}$. To do so, consider profile $\mathbf{R}'$:

$$\#2 : b \succ a \succ c$$
$$\#1 : c \succ b \succ a$$

Without going into the details, notice that we can justify $F(\mathbf{R}') = \{b\}$ through Reinforcement, Positive Responsiveness, Cancellation and Faithfulness.[3] Further, notice that the profile $\mathbf{R} + \mathbf{R}'$ is a Cancellation profile:

$$\#1 : a \succ b \succ c$$
$$\#2 : c \succ a \succ b$$
$$\#2 : b \succ a \succ c$$
$$\#1 : c \succ b \succ a$$

This holds because every pair of alternatives ties in a majority contest. Thus, by Cancellation, $F(\mathbf{R} + \mathbf{R}') = \{a, b, c\}$. Therefore, $F(\mathbf{R}) \neq \{b\}$. For if this was the case, then we would have, by Reinforcement, that $F(\mathbf{R} + \mathbf{R}') = \{b\}$. But $\{b\}$ cannot win in the superprofile (by Cancellation). Hence, $F(\mathbf{R}) \neq \{b\}$.[4] △

For another example, see the justification shown in Example 4.3. Specifically, in that case, we showed that every outcome, except for the Borda outcome, leads to a contradiction.

5.1.2.5 *Elimination by Contradiction via Neutrality and Positive Responsiveness*

Consider a profile $\mathbf{R}$ and an $x^\star \in X$. We will show a pattern involving Neutrality and Positive Responsiveness that eliminates all outcomes containing $x^\star$ for profile $\mathbf{R}$.

3 Briefly, to justify $\{b\}$, we can partition $\mathbf{R}'$ into the profile $\{b \succ a \succ c\}$ (where only b wins by Faithfulness) and a profile where b is a *quasi-tied winner*. Such winners can be justified through Cancellation and Positive Responsiveness, as explained in Section 4.1.4.2.

4 Coincidentally, notice that the above argument actually eliminates for $\mathbf{R}$ every outcome containing b. However, we talked about a specific outcome for simplicity.

Intuitively, this pattern works as follows. Consider a profile $\mathbf{R}'$ (not necessarily distinct from $\mathbf{R}$). We show that, if $x^\star$ wins in $\mathbf{R}$, then by POSITIVE RESPONSIVENESS and NEUTRALITY there is no possible outcome for $\mathbf{R}'$. More specifically, we will show that, if $x^\star \in F(\mathbf{R})$, then we can justify two different outcomes for $\mathbf{R}'$ (by POSITIVE RESPONSIVENESS and NEUTRALITY). Clearly, this cannot be the case, as any SCF can only elect *one* outcome. Thus, $x^\star \notin F(\mathbf{R})$.

There are many and diverse ways in which this pattern can be realised, and it is not feasible to show them all here. Instead, consider this example.

Example 5.10. Consider the following profile, called $\mathbf{R}$:

$$\#1 : a \succ b \succ c$$
$$\#1 : c \succ a \succ b$$

Suppose that $b \in F(\mathbf{R})$. Then, let $\pi : X \to X$ be a permutation that swaps a and b. Consider profile $\pi(\mathbf{R})$:

$$\#1 : b \succ a \succ c$$
$$\#1 : c \succ b \succ a$$

By NEUTRALITY, we derive that $a \in F(\pi(\mathbf{R}))$, because $\pi(b) = a$. Now, consider profile $\mathbf{R}'$, obtained by raising the support of b in $\mathbf{R}$:

$$\#1 : a \succ b \succ c$$
$$\#1 : c \succ b \succ a$$

By POSITIVE RESPONSIVENESS, $F(\mathbf{R}') = \{ b \}$. But notice that we can also obtain $\mathbf{R}'$ from $\pi(\mathbf{R})$ by raising the support of a. Since we proved that a wins in $\pi(\mathbf{R})$, by POSITIVE RESPONSIVENESS, $F(\mathbf{R}') = \{ a \}$. We derived a contradiction: therefore, $b \notin F(\mathbf{R})$. △

5.1.3 *Summary*

Overall, we can observe three main macro-patterns of explanation.

- **Immediate explanations.** These are the explanations where the goal profile can be immediately justified, without looking at any other profile. Let us call such profiles *immediately justifiable*. These are the explanations grounded in one of the following axioms: CANCELLATION, FAITHFULNESS, NEUTRALITY (through SYMMETRY), and PARETO PRINCIPLE.
- **Direct explanations**. These are the explanations where the goal profile is not immediately justifiable, but can be connected to one (or more) immediately justi-

fiable profiles through a chain of instances of either REINFORCEMENT or POSITIVE RESPONSIVENESS.

- **Indirect explanations**. These are the explanations that derive a contradiction for all but one of the outcomes for the given profile **R**, or that justify an outcome for **R** by exhaustively considering some cases. For example, this happens when we refer to a superprofile of **R**, and by REINFORCEMENT, show that if we do not rule out some outcomes we get a contradiction for this bigger profile.[5] Alternatively, indirect explanations can be grounded in POSITIVE RESPONSIVENESS and NEUTRALITY, where these two axioms can be used to derive a contradiction for every outcome (except the target outcome).[6]

We found that the first two categories are of particular interest. Indeed, they are much less varied than the third case, and much more predictable, two qualities that can help when designing axiom-specific heuristics. Furthermore, in terms of explanatoriness, they seem much easier to understand. On the one hand, immediate explanations are extremely straightforward. On the other hand, direct explanations have some sort of "constructive flavour". In a sense, the target outcome is constructed (through the interprofile axioms) from the outcomes of some immediately justifiable profiles. In other words, we start from some very easy-to-explain profiles, and then build our goal profile through some interprofile axioms instances, showing that, under this construction, the target outcome must win.

5.2 EXPLANATIONS AND EXPLANATORINESS

To conclude, we will give some remarks on the notions of *explanation* and *explanatoriness*.

First, let us recall the definition of justification that we adopted. Given a profile **R** and an outcome $X^\star$, a justification $\langle \mathcal{A}^N, \mathcal{A}^E \rangle$ for $X^\star$ in **R** is composed of a *normative basis* $\mathcal{A}^N$ and an *explanation* $\mathcal{A}^E$. The normative basis contains some intuitively appealing conditions that a voting rule might satisfy. In particular, the axioms in $\mathcal{A}^N$ entail that $X^\star$ must be the outcome for **R**. Therefore, the normative basis already prescribes, by itself, that $X^\star$ must win in **R**. The role of an explanation is to show *why* this is the case.

5 Notice that this construction, that refers to arbitrarily large superprofiles, is a very common and powerful technique used in proofs concerning (the classical formulation of) REINFORCEMENT. See, for example, the original characterisation of the Borda rule by Young (1974) (in which Young introduces this axiom), or the alternative proof of the same result given by Hansson and Sahlquist (1976). Furthermore, a similar construction is used in the algorithm proposed by Cailloux and Endriss (2016) to automatically justify the Borda outcomes. However, as already argued multiple times, this structure might not be entirely satisfactory in terms of explanatoriness.

6 Note that some of these indirect explanations involving POSITIVE RESPONSIVENESS and NEUTRALITY would require ANONYMITY when working with the non-anonymous model.

Historically, the notion of *explanation* has been linked with the notion of *causation* by philosophers (Mayes, 2001). This captures well the intuitive meaning of explanation: to explain a natural phenomenon means, in some sense, to identify its cause. The notion of *explanatory proof* in the philosophy of mathematics is perhaps more relevant to our discussion, but is grounded in similar intuitions. In her work, Hanna (2000) lists several epistemic aims of mathematical proofs, and identifies the fundamental functions as *verification* (which is concerned with the truth of a statement) and *explanation* (providing insight into why something is true). In particular, she insists that, in the educational domain, proofs should be first and foremost explanations. Given the scope of our work, we claim that this holds for our case as well.

Thus, both in natural sciences and in mathematics, we can make the distinction between *verification* and *understanding*. Indeed, in science, that a certain phenomenon is "true" might be already established without an explanation. For instance, we know that if we drop an apple the apple lands on the ground, and we can *verify* that this is true by observation. However, to understand *why* this holds, we need to identify the cause of this event (e.g., gravity). Similarly, in mathematics, we might know that some statement is *true* without knowing *why*. For example, this happens with some computer-generated proofs, whose size might be in the hundreds of terabytes (Benzmüller, 2019). These proofs can be used to *verify* that something holds, but it is hard to argue that they give any understanding for *why* it does hold.

How does this relate to the notion of justification proposed by Boixel and Endriss (2020)? Consider a justification grounded in normative basis $\mathcal{A}^N$ in favour of outcome $X^\star$ for $\mathbf{R}^\star$. Intuitively, what this justification states is that "if you accept $\mathcal{A}^N$, then you must accept that $X^\star$ wins in $\mathbf{R}^\star$". More formally, this can be expressed as a statement $\mathcal{S}$:

$$\left[\mathbb{I}(\mathcal{A}^N) \neq \varnothing\right] \;\wedge\; \left[F \in \mathbb{I}(\mathcal{A}^N) \implies F(\mathbf{R}^\star) = X^\star\right]$$

Now, consider a program P that *verifies* $\mathcal{S}$ as follows. Given $\mathcal{A}^N$, $\mathbf{R}^\star$ and $X^\star$, P checks whether $\mathcal{A}^N$ is non-trivial and whether every F in $\mathbb{I}(\mathcal{A}^N)$ satisfies $F(\mathbf{R}^\star) = X^\star$. How this program does so is not important (e.g., it might enumerate all $F \in \mathbb{I}(\mathcal{A}^N)$). If the statement $\mathcal{S}$ holds, then P prints the following sentence:

> "It has been computationally verified that every rule that satisfies $\mathcal{A}^N$ elects $X^\star$ in $\mathbf{R}^\star$."

This program, by construction, verifies that our goal statement $\mathcal{S}$ holds (modulo some proof of correctness for P itself). This program might even be *convincing* (which is not the same as *explanatory*) in the sense that the users might *believe* that $\mathcal{S}$ is true without understanding why this is so. For example, a non-expert user might be convinced by the authority of P ("some experts designed this program, and proved its correctness; since the program claims that $\mathcal{S}$ holds, $\mathcal{S}$ *has* to be true"). However, it does not explain *why* $X^\star$ is the right outcome for $\mathbf{R}^\star$ in terms of the normative principles encoded in $\mathcal{A}^N$. With an abuse of language, P gives no understanding of what causes every rule in $\mathbb{I}(\mathcal{A}^N)$ to elect $X^\star$ in $\mathbf{R}^\star$.

All this discussion serves to state the following, perhaps trivial, fact. The explicit goal of the algorithm we proposed, and of the framework introduced by Boixel and Endriss (2020), is to produce *understanding* in the users. Indeed, if we want to have a truly open, transparent and fair democratic process, the users must understand *why* a given outcome fully satisfies their notions of fairness. Were this not to be the case, we would have designed a simpler algorithm, such as P, that verifies statements such as $\mathcal{S}$ without explaining them. Therefore, it is perfectly reasonable to use the notion of *explanatoriness* (the quality of being explanatory) to guide the search for justifications. That is, as our explicit goal is to generate explanatory justifications, a viable line of research would be to focus on an algorithm that only considers a subset of all possible justifications (i.e., those that we deem as *explanatory*) in the hope of achieving a better performance (and retrieve better justifications). For example, as already noted by Boixel and Endriss (2020), a concrete way to do so could be to employ the notion *preferred explanation* proposed in the context of constraint programming (Junker, 2004). Alternatively, we could focus on what we called immediate and direct explanations above. As we said, such explanations are more limited and quite well-behaved. Therefore, developing a notion of "direct justification" could lead to a simplification of the problem, and thus to a faster justification algorithm.

So far, we discussed the function and role of an explanation. However, this raises a question: what are the characteristics of an explanatory proof? We have already introduced a possible distinction between direct and indirect justifications. However, this distinction was induced from our observations, and thus might be biased. Therefore, let us consider this problem from a broader angle.

One way to determine what makes a proof explanatory (again, already proposed by Boixel and Endriss (2020)) would be to conduct an empirical study to find out what users consider explanatory. Given that the algorithm we propose can handle much bigger (and thus more interesting) problems, we hope that our contributions can be used to conduct such a study. However, this was out of the scope of our work. For now, let us consider what philosophers proposed in the context of mathematical proofs. In her work, Hanna (2018) reviews some philosophical models of explanation (in the context of mathematics). A model that is particularly relevant for our discussion is the one that Steiner (1978) put forward. According to his view, a proof of a theorem is explanatory when it makes evident that some "characterising property" is responsible for making the theorem true. In his words, "an explanatory proof makes reference to a characterizing property of an entity or structure mentioned in the theorem, such that from the proof it is evident that the result depends on the property" (Steiner, 1978, p. 143). What this property depends on the context and changes from theorem to theorem. It is loosely defined as an essential property of a given mathematical object which distinguishes it from other objects. This sounds rather abstract, so let us give an example.

Example 5.11. Let $S_n = \sum_{i=1}^{n} i$. Consider the formula characterising the sum of the first n natural numbers S_n:

$$S_n = \frac{n(n+1)}{2} \tag{5.1}$$

This equivalence can be proved, for instance, by induction. This is a well-known proof, and it is often used to exemplify the principle of induction. Thus, let us only briefly recall it. First, we show that Equation 5.1 holds for S_1. Then, we show that, if we assume that the equation holds for S_n, then it must hold for S_{n+1}. The claim that Equation 5.1 holds then follows from the principle of induction.

According to Steiner (1978), a more explanatory proof is the following, which is based on the characterising property of *symmetry*.[7] Consider the sum of the first n natural numbers:

$$S_n = 1 + 2 + \cdots + (n-1) + n$$

We can add to this quantity the same sequence reversed:

$$\begin{aligned} 2S_n = & 1 + 2 \quad + \cdots + (n-1) + n + \\ & n + (n-1) + \cdots + 2 \quad + 1 \end{aligned}$$

Trivially, this can be rewritten as:

$$2S_n = \underbrace{(n+1) + \cdots + (n+1)}_{n \text{ times}} = n(n+1)$$

Thus, by dividing both sides by 2, we obtain Equation 5.1. Steiner argues that the explanatory power of this proof lies in the fact that it highlights a characterising property of S_n. That is, it makes it evident that the result depends on the *symmetry* between $1 + 2 + \cdots + (n-1) + n$ and $n + (n-1) + \cdots + 2 + 1$.

In sum, the first proof can be used to verify that Equation 5.1 holds, but it does not provide an intuitive understanding of why it holds. The second proof is more illuminating, as it explains that $2S_n = n(n+1)$ holds *because of the symmetry of* $2S_n$. The claim that Equation 5.1 holds follows trivially from this. △

7 This proof is commonly known as the *Gaussian proof*.

We claim that the notion of characterising property can be applied to voting problems as well. For example, consider profile **R**:

$$\begin{aligned} \#2 &: a \succ b \succ c \\ \#1 &: a \succ c \succ b \end{aligned}$$

Outcome $\{ a \}$ can be easily justified by appealing to the Pareto Principle. Indeed, it is easy to explain that $\{ a \}$ should win because **R** is *unanimously* ranked as the top choice. Thus a dominates every alternative. Similarly, consider **R**′:

$$\begin{aligned} \#1 &: a \succ b \succ c \\ \#1 &: b \succ c \succ a \\ \#1 &: c \succ a \succ b \end{aligned}$$

As we argued multiple times, by Neutrality, it must hold that $F(\mathbf{R}') = \{ a, b, c \}$. Again, it is easy to see that this profile is *symmetric*, in the sense that the alternatives differ only in name. Thus, if we want to be neutral, we cannot favour any alternative.

Finally, consider the explanation shown in Example 5.3, where we justify outcome $\{ a \}$ for **R** through Reinforcement. This explanation justifies the outcome by referring to the fact that **R** is *composed* by two subprofiles for which we can determine the outcome. Clearly, the explanatoriness of this depends on the two "subexplanations". However, the composition step itself can be thought of as a characterising property: "a wins in **R** because **R** is composed of two profiles where a wins".

In contrast, consider the explanation shown in Example 5.4. This explanation does not seem to make use of such an intuitive property. It is rather obscure, and it is hard to express concisely *why* the outcome must be $\{ a \}$. Indeed, this feels more like a *verification* of why $\{ a \}$ should win, rather than an explanation.

Note that we are not claiming that the explanations shown before "have" a characterising property, and the latter does not. Perhaps, we were just unable to see it and state it explicitly. In fact, this aims not to be a formal definition of explanatoriness. Thus, the importance of conducting an empirical assessment of what the users consider as explanatory still stands. What we are highlighting here is merely a conceptual framework, or model, to understand the notion of explanatoriness, which could be useful to guide the experimental work (either in combination, or as an alternative to, the notion of direct justifications discussed earlier). Possibly, this concept could then be used to achieve better performance in the search for justifications. For instance, we could build a set of profile properties (akin to the ones outlined above, such as symmetry, unanimity and composition) that we consider as "easy to explain". These could be extracted, for example, as a by-product of the aforementioned experimental study, in combination with pattern matching tools (e.g., unsupervised learning or clustering algorithms). Then, we could use these structures to guide the search. For example, we could use them to define more *derived axioms* (by formalising them in terms of the axioms we use). But we might even focus completely on such easy-to-explain structures. In other words, we could "force" the system to use only the explanation schemes we derived. Indeed, as argued above, focusing on explanatory justifications is perfectly reasonable, and might help in terms of both performance and user satisfaction.

6 CONCLUSION

In this chapter, we summarise our findings. Moreover, we propose some directions for future research.

6.1 THESIS SUMMARY

The automated justification of collective decisions, as developed by Boixel and Endriss (2020), is a framework that can be used to support groups in their decision making, designed to mitigate the choice-theoretic hardships of collective deliberation. In this context, a justification consists of a set of arguments in favour of a particular choice, and it is grounded in some normative conditions for collective decision processes (so-called *axioms*). Specifically, a justification explains why, if you accept these conditions, then you ought to accept said choice.

Despite its theoretical usefulness, this model poses some serious algorithmic challenges. For example, the base algorithm proposed by the above authors can only provide justifications for problem instances of a very limited size. More radically, as shown by Boixel and de Haan (2021), the automated justification of collective decisions is riddled with discouraging complexity-theoretic results.

In this thesis, we addressed this computational issue. Our research question can be summarised as: "is it possible to design an algorithm for the problem of computing justifications for collective decisions that achieves good performance in practice?". Here, by "good performance in practice", we mean an algorithm that performs well enough to satisfy some potential real-world users. In particular, we focused on the case of scenarios with around ten voters and three to four candidates. Note that our work, as well as the work of all the aforementioned authors, fits into the domain of *Computational Social Choice* (Brandt et al., 2016), a recent and thriving field that lies at the intersection of Computer Science and Social Choice Theory. We will now give a summary of our main results.

To address our research question, we proposed a justification algorithm that can be used in practice, which we called IterJustify. This can be regarded as our main contribution. The key idea behind our approach is the notion of *instance graphs*, which can be seen as a structured (graph-theoretic) view of the kind of arguments that a justification might express. This view allowed us to naturally define a notion of "justification depth" which, in turn, enabled us to place a bound on the maximum depth during the search. This resulted in a significant restriction of the search space. We proved that

ITERJUSTIFY is *sound* (that is, it can only return justifications for the input problem) and, when no bound is placed on the maximum depth, *complete* (that is, it returns all justifications of the input problem that we consider interesting). Furthermore, we showed two general families of heuristics for our algorithm, which we also proved to be sound and complete.

The significance of this contribution is two-fold. First, it allowed us to show that the automated justification of collective decisions is, in practice, viable. To establish this, we defined a concrete implementation of ITERJUSTIFY (with the addition of some heuristics) and tested it on several input problems. Such problems were both randomly generated and extracted from real-world data.[1] In these experiments, we used a set of well-known axioms from the Social Choice Theory literature. When analysing our findings, we found that most of the input problems could be explained in a reasonable amount of time, especially in the case of real-world data. Although our results are specific to the axioms we employed during the experiments, they show that the automated justification of collective decisions is feasible. Thus, we consider our results a positive answer to the research question, and our approach an improvement in terms of performance over the algorithm proposed by Boixel and Endriss (2020).

Secondly, our algorithm can be used as a tool for researchers interested in this problem. Indeed, since it can handle problems large enough to be interesting, it can be used to analyse the explanations retrieved with a particular corpus, to understand how the different axioms interact. For example, we performed such an analysis for the justifications retrieved with the axioms we employed in our experiments. Specifically, we gave a rundown of the most common *patterns of explanation* (or argumentative structures) that we observed. We summarised these observations and presented an informal and partial categorisation of the justifications we have seen. To conclude, we reflected on the notions of explanation and explanatoriness, and discussed how we could leverage on these notions to obtain better performing algorithms.

6.2 FUTURE WORK

The main goal of this work was to assert the viability of the automated justification of collective decisions in practice. We consider this goal achieved. However, there are still many exciting lines of research in this direction.

Firstly, one could study the performance of our approach with different axioms. For instance, one prominent class of axioms that were left out of our study is the class of *strategy-proofness* conditions. Intuitively, these axioms prescribe that voters should not be able to improve the outcome of the elections (from their point of view) by misrepresenting their true preferences. The problem of strategic voting is one of the most studied problems in Social Choice Theory (see the chapters by Barberà (2011) and Conitzer and Walsh (2016) for an overview), and arguably these axioms are among the

1 Real-world electoral data was taken from `Preflib`: https://www.preflib.org/

most appealing. Indeed, it seems reasonable that the voters should be incentivised to vote truthfully. Therefore, it might be interesting to include them in our experiments. Another class that was left out, as already discussed in Section 3.4.5, is the class of existential axioms (such as NON-IMPOSITION). Perhaps, if a different encoding language is used (e.g., where the explanations have access to quantifiers), some interesting justifications involving these axioms might exist. Since the approach we introduced seems not to work that well for these axioms, one could design some heuristics specifically for this case.

Secondly, as already extensively discussed in Chapter 5, it would be interesting to define a formal model of explanatoriness (in the context of group decisions). Such a model should be at least strong enough to separate the explanatory justifications from the non-explanatory ones. As a starting point for this, one could consider the notions described in the previous chapter (direct justifications and characterising properties), or conduct an empirical study of what users deem explanatory. We believe that our algorithm can play a role in this, as it is capable of justifying voting problems that are large enough to be interesting for users. This model could then be used to frame the problem from a different angle (by focusing solely on explanatory justifications). This, in turn, could lead to a radically different (and perhaps, faster) algorithm.

Thirdly, one could consider to further restrict the space of the possible profiles. Recall that the performance of our algorithm greatly depends on the size of this space, since, in a sense, during the execution we need to walk through this set. Consider how, by focusing on an anonymous model, we restricted this space (when compared to the model employed by Boixel and Endriss (2020)). This idea can be taken further. For example, one could focus on a *symmetric* model, that is, a model that is anonymous and neutral (see the work of Merlin (2003) for an example of such a model). Moreover, one could apply the idea of domain restrictions here as well. Indeed, recall the idea of single-peaked profiles, as used in our experiments. One could focus on justifications that only mention such profiles. This could be helpful, for example, when the alternatives are arranged on the left–right spectrum. In such a scenario, single-peaked profiles seem natural; thus, it sounds reasonable to provide justifications that only mention such natural profiles.

Lastly, one could add some rule-specific heuristics. Consider, for example, the algorithm proposed by Cailloux and Endriss (2016). This algorithm, given a profile, returns a justification in favour of the Borda outcome. The justifications computed by this method are similar to the ones we considered: they are composed of axiom instances, and consist of a step-by-step explanation for why the target outcome should win. Here, the user is not free to choose the normative basis: all justifications are grounded on (a variant of) the axioms used by Young (1974) to characterise the Borda rule. Furthermore, Peters et al. (2020) showed that, in this framework, the Borda winner can always be explained in $\mathcal{O}(m^2)$ steps (where m is the number of alternatives). More generally, they derived a lower bound for the length of the explanations of a broad number of voting rules (and the corresponding axiomatisations). Based on this framework, one could define such an algorithm for a selection of some widely adopted voting rules. Then, to justify *any* outcome (grounded on *any* set of axioms), the system could check the following: "is there a voting rule (among those we have an algorithm for) that is characterised by some axioms in the input corpus?". If this holds, instead of the general graph-based algorithm, the system could run the SCF-specific method.

BIBLIOGRAPHY

Ågotnes, Thomas, Wiebe van der Hoek, and Michael Wooldridge (2011). "On the Logic of Preference and Judgment Aggregation." In: *Autonomous Agents and MultiAgent Systems* 22.1, pp. 4–30.

Arora, Sanjeev and Boaz Barak (2009). *Computational Complexity: A Modern Approach*. Cambridge University Press.

Arrieta, Alejandro Barredo et al. (2020). "Explainable Artificial Intelligence (XAI): Concepts, Taxonomies, Opportunities and Challenges Toward Responsible AI." In: *Information Fusion* 58, pp. 82–115.

Arrow, Kenneth J. (1951). *Social Choice and Individual Values*. Wiley.

Arrow, Kenneth J. (1995). "Interview with Kenneth Arrow". `https://www.minneapolisfed.org/article/1995/interview-with-kenneth-arrow`. Accessed: 2021-06-10.

Arrow, Kenneth J., Amartya Sen, and Kotaro Suzumura, eds. (2002). *Handbook of Social Choice and Welfare*. Elsevier.

Barberà, Salvador (2011). "Strategyproof Social Choice." In: *Handbook of Social Choice and Welfare*. Ed. by Kenneth J. Arrow, Amartya Sen, and Kotaro Suzumura. Vol. 2. Elsevier.

Beckert, Bernhard, Thorsten Bormer, Rajeev Goré, Michael Kirsten, and Carsten Schürmann (2017). "An Introduction to Voting Rule Verification." In: *Trends in Computational Social Choice*. Ed. by Ulle Endriss. AI Access.

Benzmüller, Christoph (2019). *What is a Proof? What should it be?* `arXiv:1904.06332 [math.HO]`.

Benzmüller, Christoph and Bertram Lomfeld (2020). "Reasonable Machines: A Research Manifesto." In: *Proceedings of the 43rd German Conference on Artificial Intelligence (KI-2020)*.

Berg, Sven (1985). "Paradox of Voting under an Urn Model: The Effect of Homogeneity." In: *Public Choice* 47.2, pp. 377–387.

Biere, Armin (2009). "Bounded Model Checking." In: *Handbook of Satisfiability*. Ed. by Armin Biere, Marijn Heule, Hans van Maaren, and Toby Walsh. IOS Press.

Black, Duncan (1948). "On the Rationale of Group Decision-Making." In: *Journal of Political Economy* 56.1, pp. 23–34.

Boixel, Arthur and Ulle Endriss (2020). "Automated Justification of Collective Decisions via Constraint Solving." In: *Proceedings of the 19th International Conference on Autonomous Agents and MultiAgent Systems (AAMAS-2020)*.

Boixel, Arthur and Ronald de Haan (2021). "On the Complexity of Finding Justifications for Collective Decisions." In: *Proceedings of the 35th AAAI Conference on Artificial Intelligence (AAAI-2021)*.

Bonacina, Maria Paola (2017). "Automated Reasoning for Explainable Artificial Intelligence." In: *Proceedings of the First Workshop on Automated Reasoning: Challenges, Applications, Directions, Exemplary Achievements (ARCADE)*.

Brandt, Felix, Vincent Conitzer, Ulle Endriss, Jérôme Lang, and Ariel D. Procaccia, eds. (2016). *Handbook of Computational Social Choice*. Cambridge University Press.

Brandt, Felix and Christian Geist (2016). "Finding Strategyproof Social Choice Functions via SAT Solving." In: *Proceedings of the 13th International Conference on Autonomous Agents and MultiAgent Systems (AAMAS-2014)*.

Brandt, Felix, Christian Geist, and Dominik Peters (2017). "Optimal Bounds for the No-Show Paradox via SAT Solving." In: *Mathematical Social Sciences* 90, pp. 18–27.

Cailloux, Olivier and Ulle Endriss (2016). "Arguing about Voting Rules." In: *Proceedings of the 15th International Conference on Autonomous Agents and MultiAgent Systems (AAMAS-2016)*.

Ching, Stephen (1996). "A Simple Characterization of Plurality Rule." In: *Journal of Economic Theory* 71.1, pp. 298–302.

Cinà, Giovanni and Ulle Endriss (2016). "Proving Classical Theorems of Social Choice Theory in Modal Logic." In: *Journal of Autonomous Agents and MultiAgent Systems* 30.5, pp. 963–989.

Conitzer, Vincent (2009). "Eliciting Single-peaked Preferences using Comparison Queries." In: *Journal of Artificial Intelligence Research* 35, pp. 161–191.

Conitzer, Vincent and Toby Walsh (2016). "Barriers to Manipulation in Voting." In: *Handbook of Computational Social Choice*. Ed. by Felix Brandt, Vincent Conitzer, Ulle Endriss, Jérôme Lang, and Ariel D. Procaccia. Cambridge University Press.

Endriss, Ulle (2011). "Logic and Social Choice Theory." In: *Logic and Philosophy Today*. Ed. by Amitabha Gupta and Johan van Benthemvan. Vol. 2. College Publications.

Endriss, Ulle (2020). "Analysis of One-to-One Matching Mechanisms via SAT Solving: Impossibilities for Universal Axioms." In: *Proceedings of the 34th AAAI Conference on Artificial Intelligence (AAAI-2020)*.

Faliszewski, Piotr, Piotr Skowron, Arkadii Slinko, and Nimrod Talmon (2017). "Multiwinner Voting: A New Challenge for Social Choice Theory." In: *Trends in Computational Social Choice*. Ed. by Ulle Endriss. AI Access.

Fishburn, Peter C. (1973). *The Theory of Social Choice*. Princeton University Press.

Fishburn, Peter C. (1974). "Paradoxes of Voting." In: *The American Political Science Review* 68.2, pp. 537–546.

Garman, Mark B. and Morton I. Kamien (1968). "The Paradox of Voting: Probability Calculations." In: *Behavioral Science* 13.4, pp. 306–316.

Gehrlein, William V. (2006). *Condorcet's Paradox*. Springer.

Gehrlein, William V. and Peter C. Fishburn (1976). "Condorcet's Paradox and Anonymous Preference Profiles." In: *Public Choice* 26.1, pp. 1–18.

Geist, Christian and Ulle Endriss (2011). "Automated Search for Impossibility Theorems in Social Choice Theory: Ranking Sets of Objects." In: *Journal of Artificial Intelligence Research* 40, pp. 143–174.

Geist, Christian and Dominik Peters (2017). "Computer-Aided Methods for Social Choice Theory." In: *Trends in Computational Social Choice*. Ed. by Ulle Endriss. AI Access.

Gibbard, Allan (1973). "Manipulation of Voting Schemes: A General Result." In: *Econometrica: Journal of the Econometric Society* 41.4, pp. 587–601.

Goldman, Jonathan and Ariel D. Procaccia (2015). "Spliddit: Unleashing Fair Division Algorithms." In: *ACM SIGecom Exchanges* 13.2, pp. 41–46.

Gomes, Carla P., Henry Kautz, Ashish Sabharwal, and Bart Selman (2008). "Satisfiability Solvers." In: *Handbook of Knowledge Representation*. Ed. by Frank van Harmelen, Vladimir Lifschitz, and Bruce Porter. Elsevier.

Grandi, Umberto and Ulle Endriss (2013). "First-Order Logic Formalisation of Impossibility Theorems in Preference Aggregation." In: *Journal of Philosophical Logic* 42.4, pp. 595–618.

Guidotti, Riccardo, Anna Monreale, Salvatore Ruggieri, Franco Turini, Fosca Giannotti, and Dino Pedreschi (2018). "A Survey of Methods for Explaining Black Box Models." In: *ACM Computing Surveys* 51.5, pp. 1–42.

Hanna, Gila (2000). "Proof, Explanation and Exploration: An Overview." In: *Educational Studies in Mathematics* 44.1, pp. 5–23.

Hanna, Gila (2018). "Reflections on Proof as Explanation." In: *Advances in Mathematics Education Research on Proof and Proving*. Ed. by Andreas J. Stylianides and Harel van Guershon. Springer.

Hansson, Bengt and Henrik Sahlquist (1976). "A Proof Technique for Social Choice with Variable Electorate." In: *Journal of Economic Theory* 13.2, pp. 193–200.

Hashemi, Vahid and Ulle Endriss (2014). "Measuring Diversity of Preferences in a Group." In: *Proceedings of the 21st European Conference on Artificial Intelligence (ECAI 2014)*.

Henriet, Dominique (1985). "The Copeland Choice Function: An Axiomatic Characterization." In: *Social Choice and Welfare* 2.1, pp. 49–63.

Junker, Ulrich (2004). "QuickXplain: Preferred Explanations and Relaxations for Over-Constrained Problems." In: *Proceedings of the 19th AAAI Conference on Artificial Intelligence (AAAI-2004)*.

Kirsten, Michael and Olivier Cailloux (2018). "Towards Automatic Argumentation about Voting Rules." In: *Actes de la 4ème conférence sur les Applications Pratiques de l'Intelligence Artificielle (APIA-2018)*.

Kluiving, Boas, Adriaan de Vries, Pepijn Vrijbergen, Arthur Boixel, and Ulle Endriss (2020). "Analysing Irresolute Multiwinner Voting Rules with Approval Ballots via

SAT Solving." In: *Proceedings of the 24th European Conference on Artificial Intelligence (ECAI-2020)*.

Langley, Pat (2019). "Explainable, Normative, and Justified Agency." In: *Proceedings of the 33rd AAAI Conference on Artificial Intelligence (AAAI-2019)*.

Lifschitz, Vladimir, Leora Morgenstern, and David Plaisted (2008). "Knowledge Representation and Classical Logic." In: *Handbook of Knowledge Representation*. Ed. by Frank van Harmelen, Vladimir Lifschitz, and Bruce Porter. Elsevier.

Mattei, Nicholas and Toby Walsh (2013). "PrefLib: A Library of Preference Data." In: *Proceedings of the 3rd International Conference on Algorithmic Decision Theory (ADT-2013)*. URL: http://preflib.org.

May, Kenneth O. (1952). "A Set of Independent Necessary and Sufficient Conditions for Simple Majority Decision." In: *Econometrica: Journal of the Econometric Society* 20.4, pp. 680–684.

Mayes, G. Randolph (2001). "Theories of Explanation". In: *The Internet Encyclopedia of Philosophy*, https://iep.utm.edu/explanat/. Accessed: 2021-06-10.

Merlin, Vincent (2003). "The Axiomatic Characterizations of Majority Voting and Scoring Rules." In: *Mathématiques et Sciences Humaines* 41.161, pp. 87–109.

Moulin, Hervé (1980). "On Strategy-Proofness and Single Peakedness." In: *Public Choice* 35.4, pp. 437–455.

Moulin, Hervé (1988). "Condorcet's Principle implies the No Show Paradox." In: *Journal of Economic Theory* 45.1, pp. 53–64.

Nipkow, Tobias (2009). "Social Choice Theory in HOL." In: *Journal of Automated Reasoning* 43.3, pp. 289–304.

Papadimitriou, Christos H. and Mihalis Yannakakis (1982). "The Complexity of Facets (and Some Facets of Complexity)." In: *Proceedings of the Fourteenth Annual ACM Symposium on Theory of Computing (ACM-1982)*.

Peters, Dominik, Ariel D. Procaccia, Alexandros Psomas, and Zixin Zhou (2020). "Explainable Voting." In: *Advances in Neural Information Processing Systems 33 (NeurIPS 2020)*.

Procaccia, Ariel D. (2019). "Axioms Should Explain Solutions." In: *The Future of Economic Design*. Ed. by Jean-François Laslier, Hervé Moulin, M. Remzi Sanver, and William S. Zwicker. Springer.

Regenwetter, Michel, Bernard Grofman, Ilia Tsetlin, and Anthony A.J. Marley (2006). *Behavioral Social Choice: Probabilistic Models, Statistical Inference, and Applications*. Cambridge University Press.

Rossi, Francesca, Peter Van Beek, and Toby Walsh, eds. (2006). *Handbook of Constraint Programming*. Elsevier.

Russell, Stuart and Peter Norvig (2015). *Artificial Intelligence: A Modern Approach*. Pearson.

Satterthwaite, Mark Allen (1975). "Strategy-Proofness and Arrow's Conditions: Existence and Correspondence Theorems for Voting Procedures and Social Welfare Functions." In: *Journal of Economic Theory* 10.2, pp. 187–217.

Sekiguchi, Yohei (2012). "A Characterization of the Plurality Rule." In: *Economics Letters* 116.3, pp. 330–332.

Sen, Amartya (1970). "The Impossibility of a Paretian Liberal." In: *Journal of Political Economy* 78.1, pp. 152–157.

Stanley, Richard P. (1997). *Enumerative Combinatorics*. Vol. 1. Cambridge University Press.

Steiner, Mark (1978). "Mathematical Explanation." In: *Philosophical Studies: An International Journal for Philosophy in the Analytic Tradition* 34.2, pp. 135–151.

Szufa, Stanisław, Piotr Faliszewski, Piotr Skowron, Arkadii Slinko, and Nimrod Talmon (2020). "Drawing a Map of Elections in the Space of Statistical Cultures." In: *Proceedings of the 19th International Conference on Autonomous Agents and MultiAgent Systems (AAMAS-2020)*.

Tang, Pingzhong and Fangzhen Lin (2009). "Computer-Aided Proofs of Arrow's and Other Impossibility Theorems." In: *Artificial Intelligence* 173.11, pp. 1041–1053.

Thomson, William (2016). *Introduction to the Theory of Fair Allocation*. Ed. by Felix Brandt, Vincent Conitzer, Ulle Endriss, Jérôme Lang, and Ariel D. Procaccia.

Troquard, Nicolas, Wiebe van der Hoek, and Michael Wooldridge (2011). "Reasoning about Social Choice Functions." In: *Journal of Philosophical Logic* 40.4, pp. 473–498.

Van Deemen, Adrian (2014). "On the Empirical Relevance of Condorcet's Paradox." In: *Public Choice* 158.3, pp. 311–330.

Walsh, Toby (2015). *Generating Single Peaked Votes*. `arXiv:1503.02766 [cs.GT]`.

Wiedijk, Freek (2009). "Formalizing Arrow's Theorem." In: *Sadhana* 34.1, pp. 193–220.

Young, H. Peyton (1974). "An Axiomatization of Borda's Rule." In: *Journal of Economic Theory* 9.1, pp. 43–52.

Zhao, Wenting and Mark H. Liffiton (2016). "Parallelizing Partial MUS Enumeration." In: *28th International Conference on Tools with Artificial Intelligence (ICTAI-2016)*.

Zwicker, William S. (2016). "Introduction to Voting Theory." In: *Handbook of Computational Social Choice*. Ed. by Felix Brandt, Vincent Conitzer, Ulle Endriss, Jérôme Lang, and Ariel D. Procaccia. Cambridge University Press.