Judgment Aggregation in Dynamic Logic of Propositional Assignments

MSc Thesis *(Afstudeerscriptie)*

written by

Arianna Novaro
(born March 09, 1992 in Genova, Italy)

under the supervision of **Umberto Grandi** and **Ulle Endriss**, and submitted to the Board of Examiners in partial fulfillment of the requirements for the degree of

MSc in Logic

at the *Universiteit van Amsterdam.*

Date of the public defense:	**Members of the Thesis Committee:**
September 26, 2016	Prof. Dr. Jan van Eijck
	Dr. Ulle Endriss
	Dr. Umberto Grandi
	Prof. Dr. Sonja Smets
	Dr. Jakub Szymanik (chair)

Institute for Logic, Language and Computation

Abstract

Judgment Aggregation studies how agents take a collective decision on a certain number of issues based on their individual opinions. In recent years, a line of research in Judgment Aggregation investigates how to model this framework within a logical calculus — usually designed *ad hoc* for this purpose. By contrast, in this thesis we show how it is possible to translate any aggregation problem formulated in Binary Aggregation with Integrity Constraints (a model for Judgment Aggregation) into Dynamic Logic of Propositional Assignments (an instance of Propositional Dynamic Logic). In the first part of our work, we show how to appropriately express many well-known aggregation procedures as programs. Then, we translate some desirable properties of aggregation rules, i.e. axioms, as formulas of the logic. Finally, we connect the work carried out in the previous parts to model results about a problem known as the safety of the agenda. The positive outcome of our approach thus suggests this to be a promising path for future investigation, both as a means to formally express other areas of Judgment Aggregation, and as opening the way towards the use of automated reasoning techniques in this field.

Acknowledgements

First and foremost, I want to thank my supervisor Umberto Grandi for his guidance and support during the writing of this thesis. His patience in (re)reading and correcting updated versions of this work at any moment, as well as his endless enthusiasm and motivation to make further steps onwards, made the whole experience enjoyable and enriching for me.

I am also grateful to my other thesis supervisor, and academic mentor, Ulle Endriss. Even from abroad he managed to provide prompt and meticulous feedback, which has been of great value to enhance the pages that follow. I also owe him a sincere thanks for guiding me into the world of scientific research, for his words of encouragement and wisdom throughout these years, and for introducing me to the fine art of slides-making.

I would like to thank Jan van Eijck, Sonja Smets and Jakub Szymanik for having been part of my thesis committee and for the interesting questions and remarks they raised during the defence. Moreover, thanks to all the members of the audience for coming and for listening to my presentation.

The work presented here is deeply indebted to the stimulating discussions with Andreas Herzig that took place during our frequent meetings in Toulouse. I want to thank him for carefully reading ongoing drafts of this thesis and for pointing out to me ways to improve it. Regarding my stay in Toulouse in Spring 2016, I want to thank all the members of the local research community for their hospitality, as well as the Universities of Amsterdam and Toulouse which made the visit possible in the first place.

Finally, I thank all the great people I met at the ILLC, both the international fellows and the members of Little Italy. Among them, a special thank goes to Almudena for her uncountable help and her inspirational mindset, and to Anna and Guillaume for turning a house into a home. Last but not least, I am grateful to my friends of many years and to my family for their ever-present love and support.

Table of contents

Introduction

Opening and Related Work

Groups of agents often need to take a collective decision over some issues, based on their individual opinions. This phenomenon can be encountered in real life at many levels: from a group of friends deciding on how to spend a weekend together, to the citizens of a country asked to vote in a referendum.

There are multiple ways according to which agents might aggregate their opinions. For instance, they may want to collectively accept an issue if and only if a majority of agents wants it. They might as well choose a leader among them and then take the collective decision on the issues according to what she wants. As can be easily seen, different procedures possess different properties. Some of them give to each issue the same importance, others treat the opinion of each agent equally, and so on.

Social Choice Theory studies both aggregation procedures and their properties, using different frameworks depending on how the agents express their opinions [Brandt et al., 2013; Endriss, 2011]. In case the agents are asked to provide a preference order over some alternatives, the usual model is the one of Preference Aggregation (PA). On the other hand, when agents merely have to either accept or reject issues, the formal model is the one of Judgment Aggregation (JA). In the rest of this work we will focus our attention on the latter scenario.

More precisely, we will work in the formulation of JA known in the literature as Binary Aggregation with Integrity Constraints (BA) [Grandi and Endriss, 2011; Grandi, 2012]. Here, the opinions of the agents are modeled as vectors, with a zero for the issues that an agent rejects and a one for the issues that the agent accepts. The integrity constraint is a propositional formula modelling the relationship existing among the issues, or the absence thereof. An alternative formulation, known as formula-based JA [List and Pettit, 2002], makes use of individual sets of propositional formulas instead of vectors. Agents then have to consistently include in their sets either a formula or its negation,

or the non-negated subformula if the main connective was a negation (for surveys see [Endriss, 2016; List, 2012]).

Despite these differences, it is possible to move between the two formulations via a suitable translation [Grandi and Endriss, 2011]. Depending on how aggregation problems are modeled (or translated), choosing BA over formula-based JA might bring some advantages. For instance, if complementary formulas are modeled as a single issue the aggregation procedure never leaves an issue undecided.

An important observation is that both BA and formula-based JA use logic just as a tool to model logical connections among issues. That is, while they both borrow some concepts and techniques from logic, neither of them expresses JA within a proper logical calculus. In recent years, some studies have been carried out to fill this gap — which can be done by following two approaches. The first is by designing a new logic tailored to JA, while the second is by translating the framework inside an already existing logic.

The main representative of the first approach is the modal logic Judgment Aggregation Logic (JAL), whose language is parametrised by a set of agents and a set of issues [Ågotnes et al., 2011]. In JAL there are three types of atomic propositions: one referring to the agents, one referring to the formulas on which the decision is taken and the last referring to the outcome of an aggregation rule. In addition to conjunction $\wedge$ and negation $\neg$, there are two modal operators $\Box$ and $\blacksquare$ to build complex formulas. In particular, $\Box\varphi$ is true whenever φ holds in every profile (i.e., the collection of the individual decisions of all the agents on the issues) and $\blacksquare\varphi$ is true whenever φ holds for every issue. Axiomatisations in both Hilbert-style [Ågotnes et al., 2011] and Natural Deduction [Perkov, 2016] have been provided.

Similarly, the logic for Social Choice Functions has been devised for dealing with the PA framework [Troquard et al., 2011]. The modal language here is parametrised by a set of agents and possible outcomes, and by considering both the agent's truthful and reported preferences it is possible to model strategic reasoning. A fragment of this logic has been used to derive a syntactic proof of Arrow's Theorem [Ciná and Endriss, 2015], a central result in Social Choice Theory.

In the present thesis, we will follow the second approach: namely, we will translate JA procedures and properties into Dynamic Logic of Propositional Assignments (DL-PA) [van Eijck, 2000; Balbiani et al., 2013], an instance of Propositional Dynamic Logic (PDL) [Pratt, 1976; Fischer and Ladner, 1979] where atomic programs consist of assignments of truth values to propositional variables. We list below some of the advantages of choosing this approach.

In the first place, DL-PA has been proven to be a fruiful framework to express both belief change operations [Herzig, 2014] and abstract argumentation problems [Doutre et al., 2014]. Being able to express a framework in an already established logic means that general results about that logic are already in place, and when many framworks can be expressed in the same logic it is possible to compare them and notice some similarities.

Moreover, it is quite natural to see vectors of zeroes and ones as assignments of truth values to propositional variables, and aggregation procedures taking the individual opinions of the agents to return a collective decision as the inputs and outputs of programs. This makes the choice of DL-PA a promising path to pursue for our investigation.

Finally, and perhaps more interestingly, formulas of DL-PA can be reduced to propositional logic. This means that there is a procedure to translate every modal expression of the logic as a propositional formula [van Eijck, 2000; Balbiani et al., 2013]. Given this translation, showing that the JA framework is expressible in DL-PA implies the possibility in the future of using SAT-solvers to guide research in JA.

Thesis Overview

The following is an overview of the structure of the thesis with a brief synopsis of the content of its chapters.

Chapter 1: Preliminaries. We introduce the basic tools and definitions for our study. We begin by presenting the formal framework of Binary Aggregation with Integrity Constraints as well as the syntax and semantics of Dynamic Logic of Propositional Assignments. At the end of the chapter we show how to appropriately translate any instance of a BA problem into DL-PA.

Chapter 2: Aggregation Rules in DL-PA. We cover here the topic of aggregation procedures. We analyse many different types of rules that have been studied in the JA literature, and we show how to translate them as DL-PA programs. We give a proof of the correctness of our translation for a relevant selection of these rules and we comment on the properties of our translations.

Chapter 3: Axioms in DL-PA. We focus on axioms of JA — i.e., desirable properties of aggregation rules. We distinguish between two types of axioms: the ones satisfied whenever a certain relationship exists between some given input and the outcome of a rule for that input, and the ones satisfied whenever two outcomes of a rule for different inputs relate in a certain way. We translate these axioms as DL-PA formulas and we again provide a proof of the correctness of our translation.

Chapter 4: Agenda Safety in DL-PA. We deal with the problem of the safety of the agenda: namely, we investigate how an integrity constraint relates to the axioms. More precisely, we check in which cases the outcome of a rule is guaranteed to be consistent with a given constraint depending on the structure of the constraint itself.

Conclusion. We summarise and review the results of our work. Moreover, we provide some interesting directions for future research.

Chapter 1

Preliminaries

In this chapter we formally introduce the definitions and concepts of both Binary Aggregation with Integrity Constraints and Dynamic Logic of Propositional Assignments. Notice, however, that Binary Aggregation is not the only framework in which it is possible to model Judgment Aggregation. For instance, aggregation problems can also be expressed in the so-called *formula-based* JA [List, 2012] and in various other frameworks combining the presence of propositional formulas and of constraints [Lang and Slavkovik, 2014; Endriss et al., 2016].

Though the same problems can be expressed in one framework or another, Binary Aggregation with Integrity Constraints seems to be the most suitable for our study. In fact, we can model the issues as propositional variables and the logical structure among them can be all placed on the integrity constraint, which we express as a propositional formula. This in turn makes it easier to check whether the outcome of a rule is rational or not (i.e., whether the constraint is satisfied).

Therefore, problems in BA will be translated into Dynamic Logic of Propositional Assignments [van Eijck, 2000; Balbiani et al., 2013]. This logic has already been used to model many multi-agent scenarios. In fact, in addition to embedding belief change operations [Herzig, 2014] and abstract argumentation frameworks [Doutre et al., 2014], DL-PA has been used to model interactions of agents in normative systems [Herzig et al., 2011] as well as social simulations [Gaudou et al., 2011].

The remainder of this chapter is organised as follows. We explain the formal framework of Binary Aggregation with Integrity Constraints in Section 1.1. Then, in Section 1.2 we present the logic of propositional assignments DL-PA. Section 1.3 bridges the first two parts by providing a way to translate BA problems into DL-PA. We conclude in Section 1.4 with a summary of the Chapter.

1.1 Binary Aggregation with Integrity Constraints

Binary Aggregation is a general model for groups of agents wanting to reach a collective decision based on their individual opinions [Grandi and Endriss, 2011]. The agents usually are not allowed to abstain, and they have to either accept or reject the issues as they are. We first introduce the standard notation of BA, defining the concepts involved in the analysis of our problem: from agents and issues, to the idea of aggregating the views of the first on the second. Then, in order to measure the agreement (or disagreement) of the agents on the issues at different levels, we adapt to the setting of BA the well-known Hamming distance metric.

Basic Notation

Let $\mathcal{I} = \{1, \dots, m\}$ be a finite non-empty set of *issues*, on which the *agents* in the finite non-empty set $\mathcal{N} = \{1, \dots, n\}$ express an opinion. We assume the number of agents $|\mathcal{N}| = n$ to be always odd. Though this might appear as a strong restriction, we will see that it is just a technical assumption to simplify the discussion on some rules.

For each issue $j \in \mathcal{I}$ we define $D_j = \{0, 1\}$ as the set of possible opinions an agent can have on j, where "1" denotes acceptance and "0" denotes rejection. Hence, we have that $\mathcal{D} = D_1 \times \cdots \times D_m = \{0, 1\}^m$ is a boolean *domain* for the issues. Given a set of propositional symbols $PS = \{p_1, \dots, p_m\}$, one for each issue, we let $\mathcal{L}_{PS}$ be the corresponding propositional language.

The *integrity constraint* is a formula $\mathrm{IC} \in \mathcal{L}_{PS}$ expressing the existence of a certain logical relation among the issues. In the special case where the issues are all independent from one another, we let $\mathrm{IC} = \top$. Given an integrity constraint IC, only some choices of zeroes and ones for the propositional variables in PS make the constraint true. We refer to the set of all such assignments as the set Mod(IC) of *models* satisfying IC.

We call *any* assignment $B = (b_1, \dots, b_m) \in \mathcal{D}$, regardless of whether it satisfies the integrity constraint or not, a *ballot*. In particular, for $i \in \mathcal{N}$ we denote by B_i the *individual ballot* of agent i. We assume our agents to be *rational*: this means that, given an IC, we require that $B_i \in \mathrm{Mod(IC)}$ for all $i \in \mathcal{N}$. We then collect all the individual ballots of the agents in the *profile* $\boldsymbol{B} = (B_1, \dots, B_n)$, where b_{ij} indicates the j-th element of ballot B_i in $\boldsymbol{B}$.

For any profile and any issue j there are thus two disjoint subsets of $\mathcal{N}$: one for the agents who accept j and one for the agents who reject j. More formally, we have $N^{\boldsymbol{B}}_{j:v} = \{i \in \mathcal{N} \mid b_{ij} = v\}$ for $v \in \{0, 1\}$, where if $v = 1$ we speak of the *coalition of supporters* of issue j in $\boldsymbol{B}$.

Example 1. *Consider the set of issues* $\mathcal{I} = \{1, 2\}$ *for agents* $\mathcal{N} = \{1, 2, 3\}$. *Suppose that issue 1 is "Going to the cinema at 8 p.m." and issue 2 is "Spend the evening at home". The domain of aggregation is* $\mathcal{D} = \{0, 1\}^2$ *and the propositional symbols we can use are* $PS = \{p_1, p_2\}$. *Given our issues, a suitable constraint could be* $\mathrm{IC} = \neg(p_1 \wedge p_2)$. *Suppose now that the first agent wants to go to the cinema, the second agent wants to spend the evening outside but not at the cinema, and the third agent prefers to stay at home. The following is a profile* $\boldsymbol{B}$ *representing the described situation:*

	p_1	p_2
Agent 1	1	0
Agent 2	0	0
Agent 3	0	1

The set of models of the constraint is $\mathrm{Mod}(\mathrm{IC}) = \{(1, 0), (0, 1), (0, 0)\}$ *and the individual ballots of the agents are* $B_1 = (1, 0)$, $B_2 = (0, 0)$ *and* $B_3 = (0, 1)$. *Therefore, all the agents are rational because their individual ballots are models of the constraint. The ballot* $B = (1, 1)$ *is not a model of* IC *and thus no agent can have it as her individual ballot. The coalition of supporters of issue 1 and issue 2 are* $N^{\boldsymbol{B}}_{1:1} = \{1\}$ *and* $N^{\boldsymbol{B}}_{2:1} = \{3\}$ *respectively.*

Now we shall turn to the problem of aggregating the opinions of the individuals into a collective decision. We define an *aggregation procedure*, or *aggregation rule*, or *aggregator*, as a function F mapping a rational profile to a non-empty set of ballots. Namely, we have that $F : \mathrm{Mod}(\mathrm{IC})^n \to \mathcal{P}(\mathcal{D}) \setminus \{\emptyset\}$, where $\mathcal{P}(\mathcal{D})$ is the *powerset* of $\mathcal{D}$ (i.e., the set of all subsets of $\mathcal{D}$). An aggregation rule is called *resolute* if for every profile $\boldsymbol{B}$ the outcome is a singleton, while we call it *irresolute* if for some profiles it returns tied outcomes. The outcome of a resolute aggregation rule for issue $j \in \mathcal{I}$ is denoted by $F(\mathbf{B})_j$.

Example 1 (Continued)**.** *Let the profile* $\boldsymbol{B}$ *be as the one explained before, and consider the case where the agents collectively accept an issue if and only if a majority of them wants it. By denoting this majoritarian rule as* $F = \mathrm{Maj}$ *we get the following result:*

	p_1	p_2
Agent 1	1	0
Agent 2	0	0
Agent 3	0	1
$\mathrm{Maj}(\boldsymbol{B})$	0	0

The (unique) outcome of the rule on this profile is thus $\mathrm{Maj}(\boldsymbol{B}) = \{(0,0)\}$. *In particular, the outcome for the two issues is* $\mathrm{Maj}(\boldsymbol{B})_1 = 0$ *and* $\mathrm{Maj}(\boldsymbol{B})_2 = 0$ *respectively. In this case the outcome is also rational because it satisfies the integrity constraint.*

Notice that in our definition of F we require the input to be rational, while the output can be any set of ballots B. This is, of course, just one possible choice of definition: we might want as well to define an aggregation procedure by imposing rationality conditions on the input, on the output, or on neither.

Measuring (Dis)Agreement

We now introduce a useful tool to measure the agreement (or disagreement) on the issues among the agents, or among the agents and a possible outcome, or among two profiles. The *Hamming distance* is a metric that counts on how many positions two (binary) strings differ. For instance, the Hamming distance between $s_1 = 10101$ and $s_2 = 00110$ is 3, since s_1 and s_2 differ on the first, the fourth and the fifth position.

First of all, we define the Hamming distance between two ballots B and B^* as the number of issues on which they report different opinions. More precisely, we have that:

$$H(B, B^*) \;:=\; |\{j \in \mathcal{I} \mid b_j \neq b_j^*\}|.$$

In the same way, we measure how distant a single ballot B is from a profile $\boldsymbol{B}$ by adding up the Hamming distances between B and all the ballots in $\boldsymbol{B}$. Namely, we write:

$$\mathcal{H}(B, \boldsymbol{B}) \;:=\; \sum_{i \in \mathcal{N}} H(B, B_i).$$

Lastly, to inspect the distance between two profiles $\boldsymbol{B}$ and $\boldsymbol{B}^*$ we count the sum of the Hamming distances between all the individual ballots in $\boldsymbol{B}$ and the corresponding individual ballots in $\boldsymbol{B}^*$. Therefore, we have:

$$\boldsymbol{\mathcal{H}}(\boldsymbol{B}, \boldsymbol{B}^*) \;:=\; \sum_{i \in \mathcal{N}} H(B_i, B_i^*)$$

Example 1 (Continued)**.** *Consider the individual ballots of agent 1 and agent 2 of our example. We can see that* $H(B_1, B_2) = 1$ *because the two ballots differ only on issue 1. Moreover, we have that* $\mathcal{H}(B_1, \boldsymbol{B}) = 0 + 1 + 2 = 3$ *because the Hamming distance between any ballot and itself is always 0, and* B_1 *differs from* B_2 *and* B_3 *on issue 1 and both issues, respectively. Lastly, compare* $\boldsymbol{B}$ *with the following profile* $\boldsymbol{B}^*$*:*

$\boldsymbol{B}$	p_1	p_2
Agent 1	1	0
Agent 2	0	0
Agent 3	0	1

$\boldsymbol{B}^*$	p_1	p_2
Agent 1	0	0
Agent 2	1	0
Agent 3	0	0

The Hamming distance between $\boldsymbol{B}$ *and* $\boldsymbol{B}^*$ *is* $\mathcal{H}(\boldsymbol{B}, \boldsymbol{B}^*) = 1 + 1 + 1 = 3$ *because the individual ballot of each agent in* $\boldsymbol{B}$ *differs from that of the corresponding agent in* $\boldsymbol{B}^*$ *on exactly one issue.*

1.2 Dynamic Logic of Propositional Assignments

Propositional Dynamic Logic, abbreviated as PDL, is a modal logic designed to reason about the behaviour of programs in computer science [Pratt, 1976; Fischer and Ladner, 1979; Troquard and Balbiani, 2015]. These programs can be atomic or complex, and each one of them has its own modal operator (diamond or box). Thanks to these modalities, concepts such as "after *some* execution of program π, a state where formula φ holds is reached" or "after *every* execution of program π', a state where formula ψ holds is reached" can be expressed in the logic.

Since in PDL the atomic programs are left abstract, by specifying their actual behaviour we can obtain different logics. In particular, in Dynamic Logic of Propositional Assignments (DL-PA) atomic programs assign truth value true or false to propositional variables. We now introduce the syntax and semantics of the logic, as well as some useful abbreviations.

Syntax and Semantics

Given a countable set of propositional variables $\mathbb{P} = \{p, q, \dots\}$, the language of DL-PA is defined by the following grammar in Backus–Naur form [Balbiani et al., 2013]:

$$\begin{aligned} \varphi &::= p \mid \top \mid \bot \mid \neg\varphi \mid \varphi \vee \varphi \mid \langle\pi\rangle\varphi \\ \pi &::= +p \mid -p \mid \pi\,;\pi \mid \pi \cup \pi \mid \pi^* \mid \varphi? \end{aligned}$$

where p ranges over $\mathbb{P}$.

In DL-PA we have thus two types of expressions: *formulas* and *programs*. Atomic formulas consist of propositional variables, while complex formulas are built from atomic ones using negation $\neg$, disjunction $\vee$, and a diamond modality for each program $\langle\pi\rangle$. The other Boolean connectives, such as conjunction $\wedge$, implication $\rightarrow$, biconditional $\leftrightarrow$,

and exclusive disjunction $\oplus$ are defined in the usual way. Additionally, the dual operator $[\pi]\varphi$ abbreviates $\neg\langle\pi\rangle\neg\varphi$.

The two atomic programs, $+p$ and $-p$, assign truth value true or false to the propositional variable p respectively. For the complex programs, sequential composition $\pi\,;\pi'$ executes first π and then π'. Nondeterministic union of two programs $\pi \cup \pi'$ nondeterministically chooses to execute either π or π'. Unbounded iteration π^*, also called the Kleene star, executes the program π some nondeterministically chosen $k \in \mathbb{N}_0$ number of times. Lastly, the test $\varphi?$ checks whether formula φ holds and it fails if this is not the case.

A *valuation* v is a subset of $\mathbb{P}$ that specifies the truth value of every propositional variable in $\mathbb{P}$. Therefore the set of all valuations is $\mathbb{V} = \mathcal{P}(\mathbb{P}) = \{v_1, v_2, \dots\}$. When some propositional variable p belongs to valuation v, we say that p is true in v (and we say that p is false in v when $p \notin v$). DL-PA programs are interpreted through a unique relation between valuations, as illustrated in Table 1.1.

Notice that it is possible to construct a Kripke model $M^{\text{DL-PA}} = (W, R, V)$ for PDL [Balbiani et al., 2013]. We let the nonempty set of states W to be equal to $\mathcal{P}(\mathbb{P})$, the set of atomic programs Π_0 contains $\{+p \mid p \in \mathbb{P}\} \cup \{-p \mid p \in \mathbb{P}\}$, the relation $R : \Pi_0 \to \mathcal{P}(W \times W)$ associates every atomic program π_0 of the form $+p$ or $-p$ with (v_1, v_2) if and only if $(v_1, v_2) \in \|\,\pi_0\,\|$, and the valuation $V : W \to \mathcal{P}(\mathbb{P})$ is the identity function.

$$
\begin{aligned}
\|\,p\,\| &= \{v \in \mathbb{V} \mid p \in v\} \\
\|\,\top\,\| &= \mathcal{P}(\mathbb{P}) \\
\|\,\bot\,\| &= \emptyset \\
\|\,\neg\varphi\,\| &= \mathcal{P}(\mathbb{P}) \setminus \|\,\varphi\,\| \\
\|\,\varphi \vee \psi\,\| &= \|\,\varphi\,\| \cup \|\,\psi\,\| \\
\|\,\langle\pi\rangle\varphi\,\| &= \{v \in \mathbb{V} \mid \text{there is } v_1 \text{ s.t. } (v, v_1) \in \|\,\pi\,\| \text{ and } v_1 \in \|\,\varphi\,\|\} \\
\|\,+p\,\| &= \{(v_1, v_2) \mid v_2 = v_1 \cup \{p\}\} \\
\|\,-p\,\| &= \{(v_1, v_2) \mid v_2 = v_1 \setminus \{p\}\} \\
\|\,\pi\,;\pi'\,\| &= \|\,\pi'\,\| \circ \|\,\pi\,\| \\
\|\,\pi \cup \pi'\,\| &= \|\,\pi\,\| \cup \|\,\pi'\,\| \\
\|\,\pi^*\,\| &= \textstyle\bigcup_{k \in \mathbb{N}_0} (\|\,\pi\,\|)^k \\
\|\,\varphi?\,\| &= \{(v, v) \mid v \in \|\,\varphi\,\|\}
\end{aligned}
$$

Table 1.1 Interpretation of DL-PA connectives and programs

Abbreviations

Some abbreviations and expressions have been introduced in the literature [Balbiani et al., 2013; Blackburn et al., 2001; Herzig, 2014] to refer to complex programs which are often used as "building blocks" for other programs. Their use makes programs in general easier to understand to the reader familiar with programming languages. As a notational convention, we will use (initial) capital letters for formulas, and lowercase letters for programs and counters: namely, we will have Formulas, programs and counters (introduced below).

The program skip expresses $\top?$, with the meaning that nothing happens (since $\top$ is always true). The expression if φ then π_1 else π_2 abbreviates the program $(\varphi?\,;\pi_1) \cup (\neg\varphi?\,;\pi_2)$. Namely, if the formula φ holds, we execute the program π_1, and if it does not hold we execute the program π_2. The loop while φ do π stands for $(\varphi?\,;\pi)^*\,;\neg\varphi?$, where the last test for $\neg\varphi$ exits the loop, in the sense that π has been exectued exactly up until φ was true.

We can recursively define two complex programs, to express that π is exectued exactly n times (n-th iteration of π) and that π is executed at most n times (iteration up to n of π), in the following way:

$$\pi^n := \begin{cases} \mathsf{skip} & \text{if } n = 0 \\ \pi\,;\pi^{n-1} & \text{if } n > 0 \end{cases}$$

$$\pi^{\leq n} := \begin{cases} \mathsf{skip} & \text{if } n = 0 \\ (\mathsf{skip} \cup \pi)\,;\pi^{\leq n-1} & \text{if } n > 0 \end{cases}$$

Whenever we want to assign to some propositional variable p the truth value of some (other) propositional variable q, we can use the abbreviation $p \leftarrow q$ for if q then $+p$ else $-p$. Notice that this program can be used also to flip the truth value of some propositional variable p, by writing $p \leftarrow \neg p$.

We now introduce a program that executes π in case φ holds, but it does nothing in case φ does not hold. This is a slight modification of the test program, to avoid the failure of a complex program which makes use of a test in case the φ tested does not hold:

$$\text{if } \varphi \text{ do } \pi \;:=\; \text{if } \varphi \text{ then } \pi \text{ else skip}.$$

We can write any number $s \in \mathbb{N}_0$ in DL-PA via its binary expression thanks to a conjunction of $t = \lfloor \log s \rfloor + 1$ propositional variables [Balbiani et al., 2013]. If x is the binary expression of some number s, we thus use a conjunction of q_i and $\neg q_i$ propositional

variables, with $i \in \{0, \ldots, \lfloor \log s \rfloor\}$, such that a non-negated variable means that the corresponding binary digit in x is a 1, while a negated variable indicates a 0.

Example 2. *Suppose we want to express the number $s = 11$ in* DL-PA*. We have that $\lfloor \log 11 \rfloor + 1 = 3 + 1 = 4$, which means that we need 4 bits to express the number 11 in binary form. This is indeed the case, as the binary expression of 11 is $x = 1011$. Therefore, we need exactly 4 propositional variables to express this number in* DL-PA *and the corresponding formula is thus* $\mathsf{11} := q_3 \wedge \neg q_2 \wedge q_1 \wedge q_0$.

It is possible to define a DL-PA program that increments the value of some given counter each time it is executed [Balbiani et al., 2013]. Here, we make it explicit in the abbreviation which one is the counter whose value has to be incremented, since we may want to handle (finitely) many counters at the same time. Let $\mathsf{x}^t := \{q_i^x \mid 0 \leq i < t\}$ be a set of n propositional variables on which the program $\mathsf{incr}(\mathsf{x}^t)$ operates as follows:

$$\mathsf{incr}(\mathsf{x}^t) := \neg\Big(\bigwedge_{0\leq i\leq t-1} q_i^x\Big)?\,;\ \bigcup_{0\leq k\leq t-1}\Big((\neg q_k^x \wedge \bigwedge_{0\leq i\leq k-1} q_i^x)?\,;+q_k^x\,;\ \bigfatsemi_{0\leq i\leq k-1} -q_i^x\Big).$$

The program first checks whether some propositional variable in x^t is false — which means that the corresponding binary digit is a 0. Then, it finds the rightmost 0 and it turns it into a 1. Afterwards, it turns all the 1s following it into 0s. In case the first test fails, causing the whole program to fail, it means that all the t propositional variables in x^t are true, and therefore we already reached the maximal value $s = 2^t - 1$ of the counter.

Analogously, we can define a DL-PA program which decrements a given counter in the following way:

$$\mathsf{decr}(\mathsf{x}^t) := (\bigvee_{0\leq i\leq t-1} q_i^x)?\,;\ \bigcup_{0\leq k\leq t-1}\Big((q_k^x \wedge \bigwedge_{0\leq i\leq k-1} \neg q_i^x)?\,;-q_k^x\,;\ \bigfatsemi_{0\leq i\leq k-1} +q_i^x\Big).$$

The program works symmetrically to $\mathsf{incr}(\mathsf{x}^t)$. Namely, it first checks whether some digit is a 1; then it finds the rightmost 1, turns that 1 into a 0 and all the following 0s into 1s. In case the first test fails, it means that we have a string of zeroes, and thus it is not possible to decrement the counter anymore.

Finally, the following DL-PA program assigns value 0 to some given counter:

$$\mathsf{zero}(\mathsf{x}^t) := \bigfatsemi_{0\leq i<t} -q_i^x.$$

The program assigns truth value false to all the propositional variables q_i^x in x^t. Hence, all the digits in the corresponding binary number are turned into zeroes.

1.3 Translation from BA to DL-PA

In this section we analyse how to translate aggregation problems into DL-PA. Recall that an aggregation procedure F is a function mapping a rational profile $\boldsymbol{B}$, consisting of the individual ballots of n agents expressing an opinion on m issues, to a set of ballots of the form $B \in \{0,1\}^m$. There might be multiple tied ballots as the outcome of an aggregation rule in case it is irresolute.

As a first step, we introduce the set $\mathbb{B} := \{p_{ij} \mid i,j \in \mathbb{N}\}$ as the subset of $\mathbb{P}$ whose propositional variables encode the opinion of any agent i on any issue j. Analogously, we let the set $\mathbb{O} := \{p_j \mid j \in \mathbb{N}\}$ be the subset of $\mathbb{P}$ whose propositional variables refer to the possible output for any issue j. From these two infinite sets, we can then define two finite subsets for a specific set of n agents $\mathcal{N}$ and a set of m issues $\mathcal{I}$.

Namely, let the finite subset $\mathbb{B}^{n,m} := \{p_{ij} \mid i \in \mathcal{N} \text{ and } j \in \mathcal{I}\}$ of $\mathbb{B}$ be the set of propositional variables referring to the decision of the agents in $\mathcal{N}$ on the issues in $\mathcal{I}$. Likewise, let the finite subset $\mathbb{O}^m := \{p_j \mid j \in \mathcal{I}\}$ of $\mathbb{O}$ be the set of propositional variables referring to the collective decision on the issues in $\mathcal{I}$. Now that we have a way to refer both to the possible input and to the possible output, we can provide a translation for profiles and aggregation rules.

A given profile $\boldsymbol{B} = (B_1, \ldots, B_n)$ on issues $\mathcal{I} = \{1, \ldots, m\}$ corresponds to a (finite) valuation $v_{\boldsymbol{B}}$, such that the following holds:

(i) $v_{\boldsymbol{B}} \subseteq \mathbb{B}^{n,m}$,

(ii) $p_{ij} \in v_{\boldsymbol{B}} \iff b_{ij} = 1$.

The first condition tells us that *only* propositional variables corresponding to the decision of the agents on the issues could possibly be true in $v_{\boldsymbol{B}}$. The truth value of the propositional variables in $\mathbb{P} \setminus \mathbb{B}^{n,m}$ is thus set to false. The second condition tells us that a certain propositional variable in $v_{\boldsymbol{B}}$ is true if and only if the corresponding entry in profile $\boldsymbol{B}$ has value 1.

Example 3. *Let $\mathcal{I} = \{1,2\}$ and $\mathcal{N} = \{1,2,3\}$. The opinions of the agents are expressed in profile $\boldsymbol{B} = ((0,1),(0,0),(1,1))$. The set of propositional variables $\mathbb{B}^{3,2} = \{p_{11}, p_{12}, p_{21}, p_{22}, p_{31}, p_{32}\}$ corresponds to the entries in the profile $\boldsymbol{B}$ and the set of propositional variables $\mathbb{O}^2 = \{p_1, p_2\}$ is used to deal with the outcome of aggregation rules. Valuation $v_{\boldsymbol{B}} = \{p_{12}, p_{31}, p_{32}\} \subseteq \mathbb{B}^{3,2}$ thus encodes profile $\boldsymbol{B}$ in* DL-PA.

We define another subset of $\mathbb{P}$ to handle counters, whose values can be incremented or decremented thanks to incr and decr programs respectively. Let the set $\mathbb{U} := \{q_i \mid i \in \mathbb{N}\}$

be the subset of $\mathbb{P}$ whose propositional variables are used to express (a finite number of) counters for our programs. Notice that for how we defined $v_{\boldsymbol{B}}$, both the propositional variables in $\mathbb{O}^m$ and the propositional variables in $\mathbb{U}$ will be false by default, which means that all counters are set to zero.

An aggregation function F corresponds to a program f in DL-PA. We write $\mathsf{f}(\mathbb{B}^{n,m})$ to indicate that program f inspects the propositional variables of some profile, but it (possibly) modifies the truth value of the propositional variables in $\mathbb{O}^m$ and $\mathbb{U}$. We now distinguish two cases for resolute and irresolute aggregation rules F.

If F is a *resolute* aggregation rule, we let v' be the valuation reached after the execution of program f on a given valuation $v_{\boldsymbol{B}}$. For any profile $\boldsymbol{B}$, in case program f on valuation $v_{\boldsymbol{B}}$ is indeed computing the aggregation rule F on $\boldsymbol{B}$, where $v_{\boldsymbol{B}}$ is the translation of profile $\boldsymbol{B}$ in DL-PA, we will have that:

$$p_j \in v' \iff F(\boldsymbol{B})_j = 1$$

for $p_j \in \mathbb{O}^m$.

If F is an *irresolute* aggregation rule, we define the set of valuations $V^{\mathsf{f}}_{v_{\boldsymbol{B}}} = \{v' \mid (v_{\boldsymbol{B}}, v') \in \| \mathsf{f}(\mathbb{B}^{n,m}) \|\}$. For any profile $\boldsymbol{B}$, in case program f on valuation $v_{\boldsymbol{B}}$ is indeed computing the aggregation rule F on $\boldsymbol{B}$, where $v_{\boldsymbol{B}}$ is the translation of profile $\boldsymbol{B}$ in DL-PA, there must be a bijection $g : F(\boldsymbol{B}) \to V^{\mathsf{f}}_{v_{\boldsymbol{B}}}$ such that if $g(B) = v'$, where $B \in F(\boldsymbol{B})$ and $v' \in V^{\mathsf{f}}_{v_{\boldsymbol{B}}}$, then:

$$p_j \in v' \iff b_j = 1$$

for $p_j \in \mathbb{O}^m$.

Another part to be taken care of is how to express an integrity constraint IC. We let it be a propositional formula IC whose variables are a subset of $\mathbb{O}^m$. The constraint is expressed with the propositional variables for the outcome to make it easier to check whether the execution of a program f yields a result consistent with IC.

In order to inspect whether our profile is rational, we need to check whether the individual ballot of every agent in the profile satisfies IC. Namely, whether by copying into the propositional variables for the outcome the truth values of the propositional variables for the ballot of each agent, IC is true. The following formula is true if and only if the profile is rational: by prefixing its test to any program f corresponding to the translation of some aggregation rule F, we can inspect the rationality of a given profile.

$$\mathsf{Rational}_{\mathrm{IC}}(\mathbb{B}^{n,m}) := \bigwedge_{i \in \mathcal{N}} [\,;_{j \in \mathcal{I}}\, p_j \leftarrow p_{ij}]\mathrm{IC}.$$

Finally, consider the case where we start from some valuation $v_{\boldsymbol{B}}$ obtained as described above and after the execution of program f for aggregation rule F we get to valuation v'. As we will see, we might wonder whether some property holds after executing f again on a different (encoding of a) profile. Thus, we would need to create the "initial conditions" for a valuation that possibly corresponds to some profile $\boldsymbol{B}$ in order to safely execute f again.

The first program below sets to false all the propositional variables for the outcome in $\mathbb{O}^m$. The second program combines a check for rationality with the initialisation of the propositional variables for the outcome at zero.

$$\begin{aligned} \mathsf{init}(\mathbb{O}^m) &:= \mathop{;}_{j\in\mathcal{I}} -p_j. \\ \mathsf{prof}_{\mathrm{IC}}(\mathbb{B}^{n,m},\mathbb{O}^m) &:= \mathsf{init}(\mathbb{O}^m)\,;\mathsf{Rational}_{\mathrm{IC}}(\mathbb{B}^{n,m})?. \end{aligned}$$

When we introduced the conditions for checking whether some valuation $v_{\boldsymbol{B}}$ corresponds to profile $\boldsymbol{B}$, we required $v_{\boldsymbol{B}}$ to be a subset of $\mathbb{B}^{n,m}$. After the execution of program $\mathsf{prof}_{\mathrm{IC}}(\mathbb{B}^{n,m},\mathbb{O}^m)$, on the other hand, we just know that all the propositional variables used for expressing the outcome are set to false. While the two conditions differ, we will see that all programs encoding aggregation rules will inspect the propositional variables in $\mathbb{B}^{n,m}$ and will (possibly) change only the propositonal variables in $\mathbb{O}^m$. The counters used in programs will always be initialised at zero as the first step. Therefore, the valuation v reached after the execution of $\mathsf{prof}_{\mathrm{IC}}(\mathbb{B}^{n,m},\mathbb{O}^m)$ will be considered as encoding a rational profile $\boldsymbol{B}$ for IC as well.

We now stress an important remark to conclude the section. Since aggregation rules in BA are defined on a specific number of issues, a specific number of agents and a certain constraint IC, the programs that we will provide as their DL-PA translation are to be intended as general "schemas" for programs. Namely, in order to spell out completely the programs, a set of issues $\mathcal{I}$, a set of agents $\mathcal{N}$ and a constraint IC need to be provided.

1.4 Summary

In this chapter we introduced the preliminary definitions for both *Binary Aggregation with Integrity Constraints* and *Dynamic Logic of Propositional Assignments*. The chapter consisted of three parts.

In the first place, we laid out the framework of *Binary Aggregation with Integrity Constraints*, which is used to model groups of agents having to take a collective decision over some issues. We explained the basic concepts with the aid of some examples and

we described how to use the Hamming Distance metric to measure disagreement among agents.

In the second part, we presented the syntax and semantics of *Dynamic Logic of Propositional Assignments*. We listed and explained some common abbreviations and useful programs for this modal logic. We will be using them to define more complex expressions in the coming chapters.

Finally, we provided our first contribution by connecting these two frameworks through a translation of all the concepts of Binary Aggregation into DL-PA. In particular, we discussed how aggregation rules can be expressed in DL-PA. We will continue this discussion in the next chapter, by providing actual translations of specific aggregation rules as DL-PA programs.

Chapter 2

Aggregation Rules in DL-PA

We present here some known aggregation rules and we translate them as DL-PA programs. For every aggregation procedure there are multiple ways in which we could express it, and in some cases we will give more than one alternative translation of the same rule. In general, as it is always possible to devise different computer programs performing the same task, it is also possible to express each aggregation rule in more than one way.

After having introduced and translated as a DL-PA program each rule, we will state a proposition ensuring that our translation is correct. We will prove only a selection of these propositions, namely one for a resolute and one for an irresolute aggregator, since their proofs are straightforward once the underlying idea is understood. The interest thus does not lie in each one of these propositions taken individually, but on a more general level in the fact that we show it possible to express JA procedures in DL-PA.

Once a translation is in place, we can also model the *winner determination* problem for aggregation rules [Endriss et al., 2012; Lang and Slavkovik, 2014]: namely, computing the outcome of a rule for a given profile. This problem differs in case we are considering resolute or irresolute aggregators (and for irresolute aggregators there are multiple formulations as well). For a resolute rule F we are usually interested in asking whether a certain issue j is accepted in the outcome for profile $\boldsymbol{B}$: namely, whether $v_{\boldsymbol{B}} \models [\mathsf{f}(\mathbb{B}^{n,m})]p_j$ is the case. For an irresolute rule F, on the other hand, we could ask whether there is a possible outcome where j is accepted for profile $\boldsymbol{B}$: that is, whether $v_{\boldsymbol{B}} \models \langle \mathsf{f}(\mathbb{B}^{n,m})\rangle p_j$.

We introduce in Section 2.1 some basic DL-PA expressions, while the second part of the chapter is devoted to a discussion and translation of different types of aggregators. We start with some simple rules in Section 2.2. Then, we present rules based on maximisation and minimisation in Section 2.3, rules inspired from Preference Aggregation in Section 2.4, and rules that select the most representative voter in a profile in Section 2.5. We conclude with a summary and a discussion in Section 2.6.

2.1 Basic DL-PA Expressions

When introducing the DL-PA framework, we presented some abbreviations for formulas and programs that recall the ones used in programming languages. In this section we add other basic programs and formulas more specific to the task we have to perform. The aim will be also in this case to express complex programs more concisely and with better readability. We first present some formulas about properties and relations of numbers. Then, we discuss how to flip the truth value of some propositional variables in a given set and we end by providing two formulas expressing minimisation properties.

Relations of Numbers

The three DL-PA formulas presented here are true if and only if a certain relation exists between two numbers. The general idea is to compare the digits at the same position in the two binary expressions of the numbers. A potential problem arises when two numbers can be expressed with a different number of binary digits: in fact, bigger numbers usually require more digits than smaller numbers to be expressed in binary form.

Therefore, in all programs we will take the maximal value that a counter could have as the upper bound for *all* the counters in that program. For this reason, we drop the superscript in the notation of our counters and we write x instead of x^t. Hence, if t is the maximal number of propositional variables needed to express the maximal value a counter can take in a program, and some other number is expressible in binary form using only k variables (where $k < t$), it will nonetheless be expressed with t variables by imposing $\neg q_i$ for all $k < i \leq t$.

Example 4. *Suppose that in our program π we need to compare the number of agents accepting some issue j with the number of issues accepted by an even number of agents. Let* accept(j) *be the counter for the first parameter and* support(even) *be the counter for the second. Suppose we have that $|\mathcal{N}| = 9$ and $|\mathcal{I}| = 7$. Then, the maximal value that* accept(j) *can take is 9 (if all agents accept j), while the maximal value that* support(even) *can take is 7 (if every issue is accepted by an even number of agents).*

Notice that while 9 is expressible in binary form with four digits (1001), the number 7 only needs three (111). Suppose that in program π there are no other counters: then, we will take 9 as our upper bound and we will have that $|\mathsf{accept(j)}| = |\mathsf{support(even)}| = 4$. *If we let q'_i for $i \in \{0, 1, 2, 3\}$ be the propositional variables of* support(even)*, we will have that this counter has value 7 whenever $\neg q'_3 \wedge q'_2 \wedge q'_1 \wedge q'_0$ is true.*

The first DL-PA formula defined below is true if an only if the first number is greater than the second. Analogously, the second formula is true if the two numbers are equal, and the third if and only if the first number is greater than or equal to the second.

$$\begin{aligned}
\mathsf{x} > \mathsf{y} &:= \bigvee_{0 \le k < t} \Big(\Big(\bigwedge_{k < i < t} (q_i^x \leftrightarrow q_i^y) \Big) \wedge q_k^x \wedge \neg q_k^y \Big). \\
\mathsf{x} = \mathsf{y} &:= \bigwedge_{0 \le k < t} q_k^x \leftrightarrow q_k^y. \\
\mathsf{x} \ge \mathsf{y} &:= \mathsf{x} > \mathsf{y} \vee \mathsf{x} = \mathsf{y}.
\end{aligned}$$

Notice that in the first formula we use the convention of letting $\bigwedge_{k<i<t}(q_i^x \leftrightarrow q_i^y) = \top$ in case $k = t - 1$.

The next DL-PA formula is true if and only if the number we are considering is even:

$$\mathsf{Even}(\mathsf{x}) := \neg q_0^x.$$

As we can see, we do not care about the value of any digit except the rightmost one: in case the number is even, that digit will be a 0. We could define in the same way an analogous predicate $\mathsf{Odd}(\mathsf{x})$ which would be true if and only if the last digit is a 1. Another option would be to simply define $\mathsf{Odd}(\mathsf{x})$ as $\neg\mathsf{Even}(\mathsf{x})$.

Flipping

We introduce here programs that flip the truth value of some propositional variables in a given set P. As we will see, they will be mostly used in formulas expressing minimisation properties.

The first program flips the truth value of just one of the propositional variables in P, but it is not known which one has been flipped.

$$\mathsf{flip}^1(P) := \begin{cases} \mathsf{skip} & \text{if } P = \emptyset \\ \bigcup_{p \in P} (p \leftarrow \neg p) & \text{otherwise} \end{cases}$$

The second program, on the other hand, resets the truth value of all propositional variables in P to some new value (as a result, either their truth value has been flipped or it has not).

$$\mathsf{flip}^{\ge 0}(P) := \begin{cases} \mathsf{skip} & \text{if } P = \emptyset \\ \mathbin{;}_{p \in P} (+p \cup -p) & \text{otherwise} \end{cases}$$

Since the choice is nondeterministic, after the execution of $\mathsf{flip}^{\geq 0}$ it might happen that the truth value of no, some, or every propositional variable in P has been flipped.

The third program flips the truth value of *all* the propositional variables in P.

$$\mathsf{flip}^{\mathsf{all}}(P) \;:=\; \begin{cases} \mathsf{skip} & \text{if } P = \emptyset \\ ;_{p \in P}(p \leftarrow \neg p) & \text{otherwise} \end{cases}$$

Notice that $\mathsf{flip}^{\mathsf{all}}$ is a deterministic program and it corresponds to the execution of the program $\mathsf{flip}^{\geq 0}$ where each propositional variable in P is assigned the opposite truth value (with respect to its current one).

Minimisation

We present here two DL-PA formulas that are true when different types of minimisations are achieved. The first formula is true if and only if $\neg\varphi$ holds whenever we do not change the truth value of some propositional variable in the non-empty set P. This is to say that P is minimal: by taking any subset of P with *exactly* one element less, and by using the program $\mathsf{flip}^{\geq 0}$ on it (which flips the truth value of none, some or every variable in P), we are considering every possible strict subset of P — and none of them is sufficient to reach a state where φ holds.

$$\mathsf{D}(\varphi, P) \;:=\; \neg\langle \bigcup_{p \in P} \mathsf{flip}^{\geq 0}(P \setminus \{p\})\rangle\varphi.$$

Notice that this definition of the formula does not imply that *it is* possible to reach φ by flipping the truth value of all the propositional variables in P, neither do we require the propositional variables in the set P to have any relationship with the propositional variables of φ (but it is possible to define a predicate with both these conditions [Herzig, 2014]).

The second formula is true if and only if it is impossible to reach a state where φ holds by flipping the truth value of less than s propositional variables in the set P [Doutre et al., 2014]. This corresponds to finding the minimal Hamming distance between the states of before and after flipping the propositional variables in P, so that φ holds afterwards.

$$\mathsf{H}(\varphi, P, \geq s) \;:=\; \begin{cases} \top & \text{if } s = 0 \\ \neg\langle \mathsf{flip}^1(P)^{\leq s-1}\rangle\varphi & \text{if } s > 0 \end{cases}$$

Therefore, it is not possible to execute $s-1$ times the program flip^1, which flips exactly one of the propositional variables in the set P, so that afterwards the formula

φ holds. Here as well we do not require the formula φ to hold in case we flip the truth value of (more than) s propositional variables in P.

2.2 Simple Aggregation Rules

In this section we present some simple aggregation rules and their DL-PA translation. We refer to these rules as *simple* in the sense that they are either relatively easy to explain or that they are used in many real-world scenarios and thus quite intuitive to grasp. Moreover, all the rules presented here are resolute: namely, for all profiles $\boldsymbol{B}$ the outcome $F(\boldsymbol{B}) = \{B\}$ is a singleton.

First, we introduce the dictatorship of agent i [Dietrich, 2006; List, 2012]. Next, we present the family of constant rules [List, 2012] and the parity rule [List and Puppe, 2009]. The nomination rule and the majority rule follow [List, 2012; May, 1952]: these rules are both instances of uniform quota rules. We conclude by introducing the more general class of quota rules [Dietrich and List, 2007; Grandi and Endriss, 2011], of which uniform quota rules are a special case.

Dictatorship of Agent i

In a dictatorship every decision is made by a single individual, disregarding the opinions of the other agents. Framed more positively, a dictatorial rule can be implemented whenever groups of agents decide to follow the guide of a leader or of an expert.

Formally, for all profiles $\boldsymbol{B}$, the outcome of the dictatorship of some fixed agent $i \in \mathcal{N}$ is her individual ballot:

$$\text{Dictatorship}_i(\boldsymbol{B}) = B_i,$$

which means that $\text{Dictatorship}_i(\boldsymbol{B})_j = 1 \iff b_{ij} = 1$, for all $j \in \mathcal{I}$.

This rule can be expressed by a simple DL-PA program that runs issue by issue and that copies into the propositional variables of the outcome the decision of agent i on that issue.

$$\mathsf{dictatorship}_i(\mathbb{B}^{n,m}) \;:=\; \mathop{;}_{j\in\mathcal{I}} (p_j \leftarrow p_{ij}).$$

We now provide a proof that the program $\mathsf{dictatorship}_i$ presented here is indeed a translation of the aggregation rule Dictatorship_i.

Proposition 1. *Let $\boldsymbol{B}$ be a rational profile for $\mathcal{N} = \{1, \dots, n\}$ and $\mathcal{I} = \{1, \dots, m\}$. Let $v_{\boldsymbol{B}}$ be the valuation corresponding to $\boldsymbol{B}$. Let $(v_{\boldsymbol{B}}, v') \in \|\, \mathsf{dictatorship}_i(\mathbb{B}^{n,m}) \,\|$. Then, $\text{Dictatorship}_i(\boldsymbol{B})_j = 1 \iff p_j \in v'$ for all $j \in \mathcal{I}$.*

Proof. Let $F = \text{Dictatorship}_i$ and $\mathsf{f} = \mathsf{dictatorship}_i$, for some $i \in \mathcal{N}$. For the left-to-right direction, consider an arbitrary $j \in \mathcal{I}$ such that $F(\boldsymbol{B})_j = 1$: we have to show that $p_j \in v'$. From the definition of F, we have that $F(\boldsymbol{B})_j = 1 \iff b_{ij} = 1$ for all $j \in \mathcal{I}$, where agent i is the dictator. By assumption $v_{\boldsymbol{B}}$ is the valuation corresponding to $\boldsymbol{B}$, and thus by definition we have that $p_{ij} \in v_{\boldsymbol{B}}$ and that $p_j \notin v_{\boldsymbol{B}}$.

From definitions and program equivalences, we get that the interpretation of $\mathsf{f}(\mathbb{B}^{n,m})$ is the following:

$$\| \mathsf{f}(\mathbb{B}^{n,m}) \| = \| (p_{im}? ; +p_m) \cup (\neg p_{im}? ; -p_m) \| \circ \cdots \circ \| (p_{i1}? ; +p_1) \cup (\neg p_{i1}? ; -p_1) \| .$$

By assumption we have that $(v_{\boldsymbol{B}}, v') \in \| \mathsf{f}(\mathbb{B}^{n,m}) \|$. This means that there are valuations $v_1, \ldots, v_{m+1}$ such that $v_1 = v_{\boldsymbol{B}}$, $v_{m+1} = v'$ and:

$$\begin{aligned} (v_1, v_2) &\in \| (p_{i1}? ; +p_1) \cup (\neg p_{i1}? ; -p_1) \|, \\ (v_2, v_3) &\in \| (p_{i2}? ; +p_2) \cup (\neg p_{i2}? ; -p_2) \|, \\ &\ldots, \\ (v_m, v_{m+1}) &\in \| (p_{im}? ; +p_m) \cup (\neg p_{im}? ; -p_m) \| . \end{aligned}$$

In particular, let $a \in \{1, \ldots, m\}$ and b be two indices such that $b = a + 1$, and such that:

$$(v_a, v_b) \in \| (p_{ij}? ; +p_j) \cup (\neg p_{ij}? ; -p_j) \|$$

for the arbitrary issue $j \in \mathcal{I}$ we are considering. Again, from definitions we get that:

$$(v_a, v_b) \in (\{(v, v^*) \mid p_{ij} \in v \text{ and } v^* = v \cup \{p_j\}\} \cup \{(v, v^*) \mid \neg p_{ij} \in v \text{ and } v^* = v \setminus \{p_j\}\}).$$

Let us call $A = \{(v, v^*) \mid p_{ij} \in v \text{ and } v^* = v \cup \{p_j\}\}$ and $B = \{(v, v^*) \mid \neg p_{ij} \in v \text{ and } v^* = v \setminus \{p_j\}\}$, so that $(v_a, v_b) \in A \cup B$. We now have to check whether $(v_a, v_b) \in A$ or $(v_a, v_b) \in B$, since clearly A and B are disjoint sets. In order to do so, we have to inspect whether $p_{ij} \in v_a$ or $\neg p_{ij} \in v_a$.

We now show that $p_{ij} \in v_a$, which will allow us to conclude what we wanted to show: namely, that $p_j \in v'$. In case $a = 1$, we have that $v_a = v_{\boldsymbol{B}}$ and thus $p_{ij} \in v_{\boldsymbol{B}}$ by assumption. Otherwise, notice that $v_1, \ldots, v_a$ only (possibly) differ in the truth value assigned to $p_1, \ldots, p_{j-1}$, respectively. Therefore, since $p_{ij} \in v_{\boldsymbol{B}} = v_1$ and the truth value of p_{ij} has not been modified in $v_2, \ldots, v_{a-1}$, we have that $p_{ij} \in v_a$. Hence, $(v_a, v_b) \in A$, and $v_b = v_a \cup \{p_j\}$. Notice also that $v_{b+1}, \ldots, v_{m+1}$ only (possibly) differ in the truth value assigned to $p_{j+1}, \ldots, p_m$. Therefore, we can conclude that $p_j \in v_{m+1} = v'$.

For the right-to-left direction, we can equivalently show that if $F(\boldsymbol{B})_j = 0$ then $p_j \notin v'$. The proof is analogous to the one for the other direction, with the sole exception that we have to show that $(v_a, v_b) \in B$ in order to conclude that $p_j \notin v'$. □

Constant Rules

The family of constant rules includes rules always returning some fixed ballot B, regardless of the profile given as input. Hence, every ballot B characterises a different instance of a constant rule. Notice that they are *degenerate* rules, in the sense that while satisfying the formal definition of a *rule*, it is difficult to claim that they are indeed *aggregation* rules.

Formally, for all profiles $\boldsymbol{B}$, we have:

$$\text{Constant}_B(\boldsymbol{B}) = B.$$

Let $\mathbb{V}_B^m := \{p_{bj} \mid j \in \mathcal{I}\}$ be the subset of $\mathbb{P}$ encoding the entries of the constant ballot B on the issues in $\mathcal{I}$. We assume here that $v_{\boldsymbol{B}}$ also includes propositional variables of the form p_{bj}, for $j \in \mathcal{I}$, so that

$$p_{bj} \in v_{\boldsymbol{B}} \iff b_j = 1.$$

The following is the general structure of a DL-PA program to express any instance of a constant rule, depending on the truth values assigned to the variables in $\mathbb{V}_B^m$. The program is analogous to the one provided for the Dictatorship of agent i.

$$\mathsf{constant}_B(\mathbb{B}^{n,m}) := \mathop{;}_{j \in \mathcal{I}} (p_j \leftarrow p_{bj}).$$

The program runs through all the issues and it copies into the propositional variables of the outcome the truth value of the corresponding propositional variables of the constant ballot B.

Proposition 2. *Let $\boldsymbol{B}$ be a rational profile for $\mathcal{N} = \{1, \ldots, n\}$ and $\mathcal{I} = \{1, \ldots, m\}$. Let $v_{\boldsymbol{B}}$ be the valuation corresponding to $\boldsymbol{B}$ and $B \in \{0,1\}^m$ be the constant ballot. Let $(v_{\boldsymbol{B}}, v') \in \|\, \mathsf{constant}_B(\mathbb{B}^{n,m}) \,\|$. Then, $\text{Constant}_B(\boldsymbol{B})_j = 1 \iff p_j \in v'$ for all $j \in \mathcal{I}$.*

Parity Rule

The parity even (odd) rule accepts an issue in the outcome if and only if the number of agents accepting it in the profile is even (odd). This rule can be used whenever, for

instance, the feasibility of a project depends on whether the agents can be organised in pairs or in two equinumerous groups.

Formally, for any issue $j \in \mathcal{I}$, the parity even rule (analogously for odd) is defined as:

$$\text{Parity}_{\text{even}}(\boldsymbol{B})_j = 1 \iff |N^{\boldsymbol{B}}_{j:1}| \mod 2 = 0.$$

We present here two DL-PA programs expressing the parity even rule: the program for the parity odd rule can be defined similarly. The first one makes use of a counter, while the second one only needs a single bit to check parity.

Counter. In this version of the program for the parity rule we use a counter named supp, whose propositional variables are all initialised to false.

$$\mathsf{parity}_{\mathsf{Even}}(\mathbb{B}^{n,m}) := \mathop{;}_{j\in\mathcal{I}} \Big(\mathsf{zero(supp)}\,;\, \mathop{;}_{i\in\mathcal{N}} \Big(\mathsf{if}\ p_{ij}\ \mathsf{do}\ \mathsf{incr(supp)}\Big)\,;\mathsf{if}\ \mathsf{Even(supp)}\ \mathsf{do}\ +p_j\Big).$$

For each issue, the program counts the number of supporters of that issue in the profile by incrementing the supp counter. The program then checks whether the last digit of the counter is a zero or a one, and it accepts the issue in the outcome accordingly. The supp counter is set to zero before repeating the same process for a different issue.

Single bit. This program keeps track step-by-step of the parity of the coalition of supporters for any issue j by using a single propositional variable.

$$\mathsf{parity}^{*}_{\mathsf{Even}}(\mathbb{B}^{n,m}) := \mathop{;}_{j\in\mathcal{I}} \Big(+p_j\,;\, \mathop{;}_{i\in\mathcal{N}} (\mathsf{if}\ p_{ij}\ \mathsf{do}\ p_j \leftarrow \neg p_j)\Big).$$

The program first assigns truth value true to the propositional variable p_j of the outcome for the issue j currently inspected. Then, it proceeds agent by agent and it switches the truth value of p_j whenever an agent accepts it in her individual ballot. In this way, if there is an even number of agents accepting j, the propositional variable p_j stays true.

Proposition 3. *Let $\boldsymbol{B}$ be a rational profile for $\mathcal{N} = \{1,\dots,n\}$ and $\mathcal{I} = \{1,\dots,m\}$. Let $v_{\boldsymbol{B}}$ be the valuation corresponding to $\boldsymbol{B}$ and let $\mathsf{f} \in \{\mathsf{parity}_{\mathsf{Even}}, \mathsf{parity}^{*}_{\mathsf{Even}}\}$. Let $(v_{\boldsymbol{B}}, v') \in \|\,\mathsf{f}(\mathbb{B}^{n,m})\,\|$. Then, $\text{Parity}_{\text{Even}}(\boldsymbol{B})_j = 1 \iff p_j \in v'$ for all $j \in \mathcal{I}$.*

Nomination Rule

The nomination rule accepts an issue in the outcome if and only if there is at least one agent in the profile who accepts that issue. As the name suggests, this rule can be useful at the stage of selecting some alternatives, and in general whenever it is important to give some weight to the opinion of individuals (or minorities).

Formally, for every issue $j \in \mathcal{I}$ we have that:

$$\text{Nomination}(\boldsymbol{B})_j = 1 \iff |N^{\boldsymbol{B}}_{j:1}| \geq 1.$$

The rule can be expressed by the following DL-PA program:

$$\mathsf{nomination}(\mathbb{B}^{n,m}) := \mathop{;}_{j\in\mathcal{I}} (\mathsf{if} \bigvee_{i\in\mathcal{N}} p_{ij} \ \mathsf{do} \ +p_j).$$

For every issue, the program checks whether at least one agent accepts that issue in her individual ballot. If this condition is true, the issue is accepted in the outcome. Notice that we do not have to explicitly state what happens when the condition is false, because we have as an assumption that the propositional variables for the outcome in valuation $v_{\boldsymbol{B}}$ are false.

Proposition 4. *Let $\boldsymbol{B}$ be a rational profile for $\mathcal{N} = \{1, \ldots, n\}$ and $\mathcal{I} = \{1, \ldots, m\}$. Let $v_{\boldsymbol{B}}$ be the valuation corresponding to $\boldsymbol{B}$. Let $(v_{\boldsymbol{B}}, v') \in \| \mathsf{nomination}(\mathbb{B}^{n,m}) \|$. Then,* $\text{Nomination}(\boldsymbol{B})_j = 1 \iff p_j \in v'$ *for all $j \in \mathcal{I}$.*

Majority Rule

The majority rule is perhaps one of the most widespread ways of aggregating judgments in real life. An issue is accepted in the outcome if and only if more than half of the agents accept it in their individual ballots.

Formally, we have that for every issue $j \in \mathcal{I}$:

$$\text{Maj}(\boldsymbol{B})_j = 1 \iff |N^{\boldsymbol{B}}_{j:1}| > \frac{n}{2}.$$

Notice that while for an odd number of agents it is quite clear what a "majority" is, this is not the case for an even number of agents. In fact, we could either decide to accept an issue if and only if more than half of the agents accept it individually (*strict* majority rule) or we could decide to accept an issue if and only if at least half of the agents accept it (*weak* majority rule).

On the other hand, this would imply a biased outcome towards 0 or 1. As an example, suppose we have $n = 8$ agents, so that exactly 4 agents accept issue j and 4 agents reject it. According to the weak majority rule, the outcome for j should be 1; while according to the strict majority rule, the outcome for j should be 0. In both cases, the number of acceptances and rejections for j is exactly the same while the outcome favours one over the other. For this reason it is usual to simply assume an odd number of agents.

We present here two alternative translations of the majority rule in DL-PA. The first program makes use of two counters, while the second program works under the special condition that the set of issues includes both an issue and its complement.

Counters. In this program we have a pro counter, which counts the number of agents accepting some issue, and the con counter, which counts the number of agents rejecting that issue, and both of them are initialised at zero.

$$\mathsf{maj}(\mathbb{B}^{n,m}) \;:=\; \mathop{;}_{j\in\mathcal{I}} \Big(\mathsf{zero(pro)}\,;\mathsf{zero(con)}\,;\; \mathop{;}_{i\in\mathcal{N}} (\mathsf{if}\ p_{ij}\ \mathsf{then\ incr(pro)\ else\ incr(con)})\,;\; \mathsf{if\ pro} > \mathsf{con\ do} + p_j\Big).$$

Proceeding issue by issue and agent by agent, whenever an agent accepts an issue, the pro counter is incremented; otherwise, the con counter is incremented. The program checks whether it is true that the number of agents who accept the issue is higher than the one of agents who reject that issue: the outcome for that issue is decided accordingly. The two counters are set to zero before repeating the process for a different issue.

Complements. To explain this program, we have to make some assumptions about the issues. In particular, here we assume that in addition to the propositional variables of the form p_{ij} and p_j (for referring to the profile and the outcome), we also have for any issue $j \in \mathcal{I}$ propositional variables of the form $\overline{p_{ij}}$ such that for every $i \in \mathcal{N}$ we initially have p_{ij} if and only if $\neg\overline{p_{ij}}$.

$$\mathsf{maj}^*(\mathbb{B}^{n,m}) \;:=\; \mathop{;}_{j\in\mathcal{I}} \Big((\mathsf{while}\ (\bigvee_{i\in\mathcal{N}} p_{ij}) \wedge (\bigvee_{i\in\mathcal{N}} \overline{p_{ij}})\ \mathsf{do}\ \bigcup_{i\in\mathcal{N}} p_{ij}?\,;-p_{ij}\,;\bigcup_{k\in\mathcal{N}} \overline{p_{kj}}?\,;-\overline{p_{kj}})\,;\; (\mathsf{if}\ \neg \bigvee_{i\in\mathcal{N}} \overline{p_{ij}}\ \mathsf{do} + p_j)\Big).$$

Proceeding issue by issue, the program checks whether it is the case that some agent is accepting this issue and that at the same time someone (else) is accepting the complement of that issue. If this is the case, we find two agents who respectively accept the positive and the negative version of this issue and we set to false both their judgments on that issue. We repeat this procedure until the condition does not hold anymore: namely, either all the remaining agents accept the positive version or they all accept the negative version of the issue. Then, we accept an issue in the outcome if there are no more agents accepting the negative version in their ballots.

Proposition 5. *Let $\boldsymbol{B}$ be a rational profile for $\mathcal{N} = \{1,\dots,n\}$ and $\mathcal{I} = \{1,\dots,m\}$. Let $v_{\boldsymbol{B}}$ be the valuation corresponding to $\boldsymbol{B}$ and let $\mathsf{f} \in \{\mathsf{maj}, \mathsf{maj}^*\}$. Let $(v_{\boldsymbol{B}}, v') \in \|\,\mathsf{f}(\mathbb{B}^{n,m})\,\|$. Then, $\mathrm{Maj}(\boldsymbol{B})_j = 1 \iff p_j \in v'$ for all $j \in \mathcal{I}$.*

Quota Rules

A quota rule specifies for each issue a certain threshold of acceptance that has to be met in order for the issue to be accepted in the outcome. Namely, if the quota for issue j is k, there have to be at least k agents who accept j in their individual ballots for j to be accepted collectively. The quota k can be any integer such that $0 \leq k \leq n+1$, where n is the number of agents. Notice that in case $k = 0$ the issue is always accepted, while for $k = n+1$ the issue is always rejected: the outcome for these issues is constant. In case all the issues have the same quota, we speak of *uniform* quota rules: we have seen two such examples with the nomination rule and the majority rule.

Letting q_j denote the quota for issue $j \in \mathcal{I}$, we thus have that:

$$\mathrm{Quota}_{q_1,\dots,q_m}(\boldsymbol{B})_j = 1 \iff |N^{\boldsymbol{B}}_{j:1}| \geq q_j.$$

Any instance of a quota rule can be expressed in DL-PA with a similar program:

$$\mathsf{quota}_{q_1,\dots,q_m}(\mathbb{B}^{n,m}) := \mathop{;}_{1\leq i\leq m} \mathsf{zero}(\mathsf{quota}_i)\,; \mathop{;}_{1\leq i\leq m} \mathsf{incr}(\mathsf{quota}_i)^{q_i}\,; \mathop{;}_{j\in\mathcal{I}} \Big(\mathsf{zero}(\mathsf{supp})\,; \big(\mathop{;}_{i\in\mathcal{N}} \mathsf{if}\ p_{ij}\ \mathsf{do}\ \mathsf{incr}(\mathsf{supp})\big)\,; \mathsf{if}\ \mathsf{supp} \geq \mathsf{quota}_j\ \mathsf{do} + p_j\Big).$$

First, the program sets each counter quota_i to the corresponding quota q_i. Proceeding issue by issue, the program counts the number of supporters of that particular issue in the profile. If the number of supporters for that issue is greater than or equal to the corresponding quota, the issue is accepted in the outcome. The counter supp, which is initialised at zero and which stores the number of supporters for the inspected issue, is set to zero at each iteration.

Proposition 6. *Let $\boldsymbol{B}$ be a rational profile for $\mathcal{N} = \{1,\dots,n\}$ and $\mathcal{I} = \{1,\dots,m\}$. Let $v_{\boldsymbol{B}}$ be the valuation corresponding to $\boldsymbol{B}$. Let $(v_{\boldsymbol{B}}, v') \in \|\,\mathsf{quota}_{q_1,\dots,q_m}(\mathbb{B}^{n,m})\,\|$. Then, $\mathrm{Quota}_{q_1,\dots,q_m}(\boldsymbol{B})_j = 1 \iff p_j \in v'$ for all $j \in \mathcal{I}$.*

2.3 Maximisation and Minimisation Rules

In this section, we present two rules based on maximisation or minimisation. The first one is the maximum subagenda rule [Endriss et al., 2016], which is also known in the literature as the maximal subagenda rule [Lang and Slavkovik, 2014]. The second one is the minimal number of atomic changes rule [Lang and Slavkovik, 2014], also known as Full_d for d being the Hamming distance [Miller and Osherson, 2009]. Both rules aim

at amending the outcome of the majority rule, in case it does not satisfy the integrity constraint IC.

Maximum Subagenda Rule

The maximum subagenda rule (MSA) returns as outcome a ballot which satisfies the integrity constraint and that has maximal agreement with the majority outcome, with respect to set inclusion and not cardinality. Namely, we have that for profile $\boldsymbol{B}$:

$$\mathrm{MSA}_{\mathrm{IC}}(\boldsymbol{B}) = \operatorname*{arg\,\subseteq\text{-}max}_{B \models \mathrm{IC}} \{j \in \mathcal{I} \mid b_j = \mathrm{Maj}(\boldsymbol{B})_j\}.$$

Example 5. *We adapt here an example to explain how the outcome of the MSA is computed* [Lang and Slavkovik, 2014]. *Consider agents* $\mathcal{N} = \{1, 2, 3\}$ *and issues* $\mathcal{I} = \{1, 2, 3, 4\}$ *for constraint* $\mathrm{IC} = (p_4 \to p_3) \wedge ((p_3 \wedge (p_1 \vee p_2)) \to p_4)$.

	p_1	p_2	p_3	p_4
Agent 1	1	1	1	1
Agent 2	1	1	0	0
Agent 3	0	0	1	0
$\mathrm{Maj}(\boldsymbol{B})$	1	1	1	0

The majority outcome on $\boldsymbol{B}$ *does not satisfy the constraint* IC. *Now, we check which maximal subsets of the outcome (or maximal partial assignments to the propositional variables for the issues) are consistent. These partial assignments are* $(1, 1, 1, -)$, $(-, -, 1, 0)$ *and* $(1, 1, -, 0)$. *We now complete them in such a way that the constraint is satisfied, thus obtaining:* $\mathrm{MSA}(\boldsymbol{B}) = \{(1, 1, 1, 1), (0, 0, 1, 0), (1, 1, 0, 0)\}$.

We present two DL-PA programs translating the MSA, which are very similar to the translations provided for the PMA update operator for Belief Change [Herzig, 2014]. The first one is simpler but has length exponential in the number of issues, while the second one does not have this feature.

Exponential. The first program is defined as follows:

$$\mathsf{msa}_{\mathrm{IC}}(\mathbb{B}^{n,m}) \;:=\; \mathsf{maj}(\mathbb{B}^{n,m})\,;\mathsf{if}\ \neg\mathrm{IC}\ \mathsf{do} \bigcup_{P \in (\mathcal{P}(\mathbb{O}^m) \setminus \emptyset)} \Big(\mathsf{D}(\mathrm{IC}, P)?\,;\mathsf{flip}^{\mathsf{all}}(P)\Big)\,;\mathrm{IC}?.$$

First, the program computes the outcome of the majority rule. Then, in case the constraint is not satisfied, the program non-deterministically chooses the minimal subset

P of the outcome of the majority such that the truth value of its propositional variables has to be flipped to obtain a result satisfying IC. Then, it flips the truth value of all the propositional variables in P and finally it checks whether the constraint holds. Notice that the length of this program grows exponentially in the number of issues because we use the non-deterministic choice operator over all the subsets of $\mathbb{O}^m$.

Storing. First of all, we introduce some useful DL-PA programs. The first stores the truth value of the propositional variables in P in some fresh propositional variables p'. The second restores the truth value of just one propositional variable p' in the corresponding propositional variable in P. The last restores the truth value of none, some, or all propositional variables p' in P.

$$\mathsf{store}(P) := \begin{cases} \mathsf{skip} & \text{if } P = \emptyset \\ \mathbin{;}_{p \in P} p' \leftarrow p & \text{otherwise} \end{cases}$$

$$\mathsf{restore}^1(P) := \begin{cases} \mathsf{skip} & \text{if } P = \emptyset \\ \bigcup_{p \in P} (p \oplus p'? \,; p \leftarrow p') & \text{otherwise} \end{cases}$$

$$\mathsf{restore}^{\geq 0}(P) := \begin{cases} \mathsf{skip} & \text{if } P = \emptyset \\ \mathbin{;}_{p \in P} (\mathsf{skip} \cup p \leftarrow p') & \text{otherwise} \end{cases}$$

The second program for expressing the MSA is the following:

$$\begin{aligned} \mathsf{msa}^*_{\mathrm{IC}}(\mathbb{B}^{n,m}) := {} & \mathsf{maj}(\mathbb{B}^{n,m}) \,; \mathsf{store}(\mathbb{O}^m) \,; \mathsf{flip}^{\geq 0}(\mathbb{O}^m) \,; \mathrm{IC}? \,; \\ & [\mathsf{restore}^1(\mathbb{O}^m) \,; \mathsf{restore}^{\geq 0}(\mathbb{O}^m)] \neg \mathrm{IC}?. \end{aligned}$$

The program first computes the outcome of the majority rule. Then, it stores the outcome in some fresh variables and tries to find a valuation such that the constraint holds (which, in case the majority outcome already satisfies IC, will be reached by flipping the truth value of no propositional variable in $\mathbb{O}^m$). At this point, the program checks whether by restoring the truth value of at least one propositional variable in $\mathbb{O}^m$ we get to a valuation that does not satisfy IC: in this way we can be sure that we considered a maximal subset of the majority outcome. In case the majority outcome was already satisfying the contraint, the modal formula will hold vacuously because $\mathsf{restore}^1$ will fail.

Proposition 7. *Let $\boldsymbol{B}$ be a rational profile for $\mathcal{N} = \{1, \ldots, n\}$, $\mathcal{I} = \{1, \ldots, m\}$ and some* IC*. Let $v_{\boldsymbol{B}}$ be the valuation corresponding to $\boldsymbol{B}$ and let $\mathsf{f} \in \{\mathsf{msa}_{\mathrm{IC}}, \mathsf{msa}^*_{\mathrm{IC}}\}$. Let $V^{\mathsf{f}}_{v_{\boldsymbol{B}}} = \{v' \mid (v_{\boldsymbol{B}}, v') \in \| \mathsf{f}(\mathbb{B}^{n,m}) \|\}$. Then, the function $g : \mathrm{MSA}_{\mathrm{IC}}(\boldsymbol{B}) \to V^{\mathsf{f}}_{v_{\boldsymbol{B}}}$ such that if $g(B) = v'$ then $b_j = 1 \iff p_j \in v'$ for all $j \in \mathcal{I}$ is a bijection.*

Proof. We are going to provide a proof just for the case of $\mathsf{f} = \mathsf{msa}_{\mathrm{IC}}$, in order to give the general idea of how such a proof looks like. Consider an arbitrary rational profile $\boldsymbol{B}$ for some set of agents $\mathcal{N}$ and for some set of issues $\mathcal{I}$, and let $v_{\boldsymbol{B}}$ be its corresponding valuation. Given that $F = \mathrm{MSA}_{\mathrm{IC}}$ is an irresolute rule, in some cases it might return an outcome $F(\boldsymbol{B})$ which is not a singleton. Therefore, given any $B \in F(\boldsymbol{B})$ we are going to show that there is a unique valuation $v' \in V^{\mathsf{f}}_{v_{\boldsymbol{B}}}$ such that $p_j \in v' \Leftrightarrow b_j = 1$ for all $j \in \mathcal{I}$. We will then construct a function g such that each $B \in F(\boldsymbol{B})$ is associated with the described valuation $v' \in V^{\mathsf{f}}_{v_{\boldsymbol{B}}}$. Finally, we will show that g is injective and surjective to conclude that g is a bijection.

As a first step, we thus have to show that $v' \in V^{\mathsf{f}}_{v_{\boldsymbol{B}}}$. Namely, that:

$$(v_{\boldsymbol{B}}, v') \in (\|\,\mathrm{IC}?\,\| \circ \|\,\mathsf{if}\ \neg\mathrm{IC}\ \mathsf{do} \bigcup_{P \in (\mathcal{P}(\mathbb{O}^m)\setminus\emptyset)} \Big(\mathsf{D}(\mathrm{IC}, P)?; \mathsf{flip}^{\mathsf{all}}(P)\Big)\,\| \circ \|\,\mathsf{maj}(\mathbb{B}^{n,m})\,\|).$$

Hence, we have to provide valuations v_a, v_b, v_c, v_d such that $v_a = v_{\boldsymbol{B}}$, $v_d = v'$ and

$$\begin{aligned}
&(v_a, v_b) \in \|\,\mathsf{maj}(\mathbb{B}^{n,m})\,\| \,. \\
&(v_b, v_c) \in \|\,\mathsf{if}\ \neg\mathrm{IC}\ \mathsf{do} \bigcup_{P \in (\mathcal{P}(\mathbb{O}^m)\setminus\emptyset)} \Big(\mathsf{D}(\mathrm{IC}, P)?; \mathsf{flip}^{\mathsf{all}}(P)\Big)\,\| \,. \\
&(v_c, v_d) \in \|\,\mathrm{IC}?\,\| \,.
\end{aligned}$$

Recall that the program $\mathsf{maj}(\mathbb{B}^{n,m})$ is deterministic: we thus let v_b be the unique valuation reachable from v_a after the execution of $\mathsf{maj}(\mathbb{B}^{n,m})$. By Proposition 5 we know that $p_j \in v_b \Leftrightarrow \mathrm{Maj}(\boldsymbol{B})_j = 1$ for all $j \in \mathcal{I}$. We now have to consider two cases: either (a) $\mathrm{Maj}(\boldsymbol{B}) \models \mathrm{IC}$, or (b) $\mathrm{Maj}(\boldsymbol{B}) \not\models \mathrm{IC}$.

(a) In this case, we have that $B = \mathrm{Maj}(\boldsymbol{B})$ and $v_b \models \mathrm{IC}$. Therefore, by the definition of the program "$\mathsf{if}\ \varphi\ \mathsf{do}\ \pi$", valuations v_b and v_c have to be linked via the program skip. We thus take $v_b = v_c = v_d$, so that $v_d \models \mathrm{IC}$ and therefore $v_d \in \|\,\mathrm{IC}\,\|$. Since $v' = v_d = v_c = v_b$, and $p_j \in v_b \Longleftrightarrow \mathrm{Maj}(\boldsymbol{B})_j = 1$ for all $j \in \mathcal{I}$, and $B = \mathrm{Maj}(\boldsymbol{B})$, we provided a valuation $v' \in V^{\mathsf{f}}_{v_{\boldsymbol{B}}}$ such that $p_j \in v' \Leftrightarrow b_j = 1$ for all $j \in \mathcal{I}$.

(b) Notice that since $B \in F(\boldsymbol{B})$, by the definition of MSA we know that $B \models \mathrm{IC}$ and that there is a maximal (with respect to set inclusion) subset Q of $\mathrm{Maj}(\boldsymbol{B})$ such that $b_j = \mathrm{Maj}(\boldsymbol{B})_j$. Let now $S = \{p_i \mid i \notin Q\}$: we take v_c to be the valuation that differs from v_b only in that the truth values of all the propositional variables in S have been flipped. We thus want to show that:

$$(v_b, v_c) \in \|\,\mathsf{D}(\mathrm{IC}, S)?; \mathsf{flip}^{\mathsf{all}}(S)\,\| \,.$$

We start by showing that $(v_b, v_b) \in \| \mathsf{D}(\mathrm{IC}, S)? \|$. Suppose for reductio that this is not the case: namely, that $v_b \notin \| \neg\langle \bigcup_{p_i \in S} \mathsf{flip}^{\geq 0}(S \setminus \{p_i\})\rangle \mathrm{IC} \|$. This means that there is some $p_q \in S$ such that $v_b \models \langle \mathsf{flip}^{\geq 0}(S \setminus \{p_q\})\rangle \mathrm{IC}$. Therefore, by fixing the truth value of p_q there is still some way to make IC true.

Consider now the set Q: by definition of S we have that $q \notin Q$. By assumption Q was maximal, and thus for B to model IC we have to have $b_q \neq \mathrm{Maj}(\boldsymbol{B})_q$. This however contradicts the fact that by not changing the truth value of p_q (i.e., leaving p_q agree with the majority result), there is a way to satisfy IC.

In order to show that $(v_b, v_c) \in \| \mathsf{flip}^{\mathsf{all}}(S) \|$, it suffices to notice that this deterministic program by definition leads to a valuation where exactly the propositional variables in S have their truth values flipped and nothing else.

The last thing left to show is that $v' \in \| \mathrm{IC} \|$ and that $p_j \in v' \Leftrightarrow b_j = 1$ for all $j \in \mathcal{I}$. Let $v_c = v_d = v'$ and consider an arbitrary $s \in \mathcal{I}$. Assume without loss of generality that $b_s = 1$, since the following reasoning can be performed in a similar way for $b_s = 0$. We distinguish the case where (i) $b_s = 1 = \mathrm{Maj}(\boldsymbol{B})_s$, and (ii) $b_s = 1 \neq \mathrm{Maj}(\boldsymbol{B})_s$.

(i) In this case, we have that $s \in Q$ and by definition $p_s \notin S$. Therefore, the truth value of p_s has not been changed between v_b and v_c. Since by Proposition 5 we have that $p_j \in v_b \Leftrightarrow \mathrm{Maj}(\boldsymbol{B})_j = 1$ for all $j \in \mathcal{I}$, this means that $p_s \in v_b$ and thus $p_s \in v_c$. Since $v_c = v_d = v'$, we have that $p_s \in v'$.

(ii) Otherwise, we have that $s \notin Q$ and by definition $p_s \in S$. Thus, the truth value of p_s has been flipped between v_b and v_c. Again, since by Proposition 5 we have that $p_j \in v_b \Leftrightarrow \mathrm{Maj}(\boldsymbol{B})_j = 1$ for all $j \in \mathcal{I}$, this means that $p_s \in v_b$ and thus $p_s \notin v_c$. Since $v_c = v_d = v'$, we have that $p_s \notin v'$.

In any case, we have shown that $p_j \in v' \Leftrightarrow b_j = 1$ for all $j \in \mathcal{I}$. Since $B \in \mathrm{Mod}(\mathrm{IC})$, we thus get that $v' \models \mathrm{IC}$.

Take now $g : F(\boldsymbol{B}) \to V^{\mathsf{f}}_{v_{\boldsymbol{B}}}$ to be the function associating to each $B \in F(\boldsymbol{B})$ the valuation $v' \in V^{\mathsf{f}}_{v_{\boldsymbol{B}}}$ as described above. To conclude the proof, we have to show that g is a bijection: namely, that it is both injective and surjective.

Showing that g is injective means that whenever $g(B') = g(B'')$, then $B' = B''$. Consider two valuation v' and v'' in $V^{\mathsf{f}}_{v_{\boldsymbol{B}}}$ such that $v' = g(B')$, $v'' = g(B'')$ and $v' = v''$. Suppose for reductio that $B' \neq B''$ for $B', B'' \in F(\boldsymbol{B})$.

This means that there is some issue $k \in \mathcal{I}$ such that $b'_k \neq b''_k$. Assume without loss of generality that $b'_k = \mathrm{Maj}(\boldsymbol{B})_k$ and that $b''_k \neq \mathrm{Maj}(\boldsymbol{B})_k$, since a very similar reasoning can be carried out in the other case. We thus have that $k \in Q'$ while $k \notin Q''$, which in turn

means that $k \notin S'$ and $k \in S''$. Hence, the execution of program $\mathsf{msa}_{\mathrm{IC}}$ will flip the truth value of proposition p_k in the second case but not in the first. Therefore, we will have that $p_k \in v' \Leftrightarrow p_k \notin v''$. Since this contradicts the fact that $v' = v''$ we have that g is injective.

For surjectivity, we have to show that for all $v' \in V^{\mathsf{f}}_{v_{\boldsymbol{B}}}$ there is some $B \in F(\boldsymbol{B})$ such that $g(B) = v'$. Consider an arbitrary $v' \in V^{\mathsf{f}}_{v_{\boldsymbol{B}}}$, namely a valuation such that $(v_{\boldsymbol{B}}, v') \in \| \mathsf{msa}_{\mathrm{IC}}(\mathbb{B}^{n,m}) \|$. Let B be the ballot such that $b_j = 1 \Leftrightarrow p_j \in v'$, for all $j \in \mathcal{I}$. By definition of g, we have that $g(B) = v'$. Since $(v_{\boldsymbol{B}}, v') \in \| \mathsf{msa}_{\mathrm{IC}}(\mathbb{B}^{n,m}) \|$, this means that there are some valuations v_a, v_b, v_c, v_d such that $v_a = v_{\boldsymbol{B}}$, $v_d = v'$, and

$$
\begin{aligned}
&(v_a, v_b) \in \| \mathsf{maj}(\mathbb{B}^{n,m}) \| \, . \\
&(v_b, v_c) \in \| \mathsf{if}\ \neg \mathrm{IC}\ \mathsf{do} \bigcup_{P \in (\mathcal{P}(\mathbb{O}^m) \setminus \emptyset)} \Big(\mathsf{D}(\mathrm{IC}, P)?; \mathsf{flip}^{\mathsf{all}}(P) \Big) \| \, . \\
&(v_c, v_d) \in \| \, \mathrm{IC}? \, \| \, .
\end{aligned}
$$

We now have to show that $B \in F(\boldsymbol{B})$, i.e. that $B \in \mathrm{argmax}^{\subseteq}_{B \models \mathrm{IC}} \{ j \in \mathcal{I} \mid b_j = \mathrm{Maj}(\boldsymbol{B})_j \}$. First of all, notice that since $v' = v_d \models \mathrm{IC}$, we have that $B \models \mathrm{IC}$ as well. Suppose for reductio that $B \notin F(\boldsymbol{B})$: hence, there is some issue $k \in \mathcal{I}$ such that $b_k \neq \mathrm{Maj}(\boldsymbol{B})_k$, but in case $b_k = \mathrm{Maj}(\boldsymbol{B})_k$ there would be a way to complete it such that $B \models \mathrm{IC}$.

Since $b_k \neq \mathrm{Maj}(\boldsymbol{B})_k$ we have that $p_k \in S$, for S being the set introduced at the beginning of this proof. Notice that since v_b corresponds to the outcome of the majority, and since both $v' = v_d = v_c$ and $p_k \in S$, we have that the truth value of p_k must have changed between v_b and v_c. This is the case if $p_k \in P$ for P being the set of the program $\mathsf{flip}^{\mathsf{all}}(P)$, and since a similar reasoning can be made for all propositional variables for the issues on which B differs from $\mathrm{Maj}(\boldsymbol{B})$, we get that $P = S$.

On the other hand, we have that $v_b \models \neg \langle \bigcup_{p \in S} \mathsf{flip}^{\geq 0}(S \setminus \{p\}) \rangle \mathrm{IC}$. However, we assumed that in case $b_k = \mathrm{Maj}(\boldsymbol{B})_k$ we would still have a way to obtain IC. Therefore, we have some propositional variable $p \in S$ such that $\langle \bigcup_{p \in S} \mathsf{flip}^{\geq 0}(S \setminus \{p\}) \rangle \mathrm{IC}$ and we thus can conclude that $B \in F(\boldsymbol{B})$. □

Minimal Number of Atomic Changes Rule

The minimal number of atomic changes (MNAC) rule returns in the outcome set a ballot with the following characteristics: (i) it satisfies the integrity constraint, (ii) there is a rational profile $\boldsymbol{B}^*$ for which computing majority yields B as a result, (iii) any other rational profile $\boldsymbol{B}'$ is further away (in terms of the Hamming distance) from the initial

profile than $\boldsymbol{B}^*$. This rule can be used whenever it would be convenient to convince some agents to slightly modify their votes for the greater good of a consistent outcome.

Formally, we have that for profile $\boldsymbol{B}$:

$$\mathrm{MNAC}_{\mathrm{IC}}(\boldsymbol{B}) \;=\; \{B \mid \mathrm{Maj}(\boldsymbol{B}^*) = B, B \models \mathrm{IC} \text{ and for all } \boldsymbol{B}' : \boldsymbol{\mathcal{H}}(\boldsymbol{B}, \boldsymbol{B}^*) \leq \boldsymbol{\mathcal{H}}(\boldsymbol{B}, \boldsymbol{B}')\}.$$

The MNAC rule can be expressed in DL-PA with the following program:

$$\begin{aligned}\mathsf{mnac}_{\mathrm{IC}}(\mathbb{B}^{n,m}) \;:=\; & \bigcup_{0\leq d\leq m\cdot n} \Big(\mathsf{H}(\langle \mathsf{prof}_{\mathrm{IC}}(\mathbb{B}^{n,m}, \mathbb{O}^m)\,;\mathsf{maj}(\mathbb{B}^{n,m})\rangle \mathrm{IC}, \mathbb{B}^{n,m}, \geq d)?\,;\mathsf{flip}^1(\mathbb{B}^{n,m})^d\Big)\,; \\ & \mathsf{prof}_{\mathrm{IC}}(\mathbb{B}^{n,m}, \mathbb{O}^m)\,;\mathsf{maj}(\mathbb{B}^{n,m})\,;\mathrm{IC}?.\end{aligned}$$

The program finds the minimal distance d to the profile $\boldsymbol{B}$ such that every agent is still rational (satisfies IC), and such that after the execution of the majority rule, the outcome satisfies the integrity constraint. Then, the truth value of the propositional variables in the profile $\boldsymbol{B}$ is changed, the majority outcome computed, and the program finishes by checking whether the constraint is satisfied.

Proposition 8. *Let $\boldsymbol{B}$ be a rational profile for $\mathcal{N} = \{1, \ldots, n\}$, $\mathcal{I} = \{1, \ldots, m\}$ and some* IC*. Let $v_{\boldsymbol{B}}$ be the valuation corresponding to $\boldsymbol{B}$ and let $\mathsf{f} = \mathsf{mnac}_{\mathrm{IC}}$. Let $V^{\mathsf{f}}_{v_{\boldsymbol{B}}} = \{v' \mid (v_{\boldsymbol{B}}, v') \in \|\mathsf{f}(\mathbb{B}^{n,m})\|\}$. Then, the function $g : \mathrm{MNAC}_{\mathrm{IC}}(\boldsymbol{B}) \to V^{\mathsf{f}}_{v_{\boldsymbol{B}}}$ such that if $g(B) = v'$ then $b_j = 1 \iff p_j \in v'$ for all $j \in \mathcal{I}$ is a bijection.*

2.4 Preference Aggregation Rules

In this section we present other rules trying to make the outcome consistent with a given IC, in case the majority outcome is not. They however differ from the ones seen before in that they all closely recall some rules known in the Preference Aggregation literature [Brams and Fishburn, 2002].

The first rule to be introduced will be the Kemeny rule [Kemeny, 1959], which is also sometimes called *the* Distance Based rule [Endriss and Grandi, 2014] or Prototype$_d$ for the Hamming distance d [Miller and Osherson, 2009]. We then present the Slater rule [Nehring et al., 2011], also known as the Maxcard Subagenda rule [Lang and Slavkovik, 2014] or Endpoint$_d$ for Hamming distance d [Miller and Osherson, 2009]. Finally, we discuss the Ranked Pairs rule, which also comes under the name of the Ranked Agenda rule [Lang and Slavkovik, 2014].

Kemeny Rule

The outcome of the Kemeny rule includes ballots that satisfy the IC and that minimise the sum of the Hamming distance to the individual ballots in the profile. Formally, we have that for profile $\boldsymbol{B}$:

$$\text{Kemeny}_{\text{IC}}(\boldsymbol{B}) = \underset{B \models \text{IC}}{\text{argmin}} \sum_{i \in \mathcal{N}} H(B, B_i).$$

As a first step, we define a program that computes the sum of the Hamming distances between the outcome and the profile. Namely, whenever the value of the propositional variables of the outcome and of the corresponding issues in the profile differ, the counter dis is incremented.

$$\mathsf{sH}(\mathbb{O}^m, \mathbb{B}^{n,m}) \;:=\; \mathsf{zero}(\mathsf{dis})\,;\; \underset{i \in \mathcal{N}}{;} \Big(\underset{j \in \mathcal{I}}{;}\; \mathsf{if}\ p_j \oplus p_{ij}\ \mathsf{do}\ \mathsf{incr}(\mathsf{dis}) \Big).$$

We now define a DL-PA formula which is true if and only if whenever some (other) outcome is closer to the profile than the current one, with respect to the Hamming distance, then IC is not satisfied. The formula makes use of two counters, dis and $\mathsf{dis'}$, the second of which stores the result of initially computing the Hamming distance between the outcome and the rest of the profile.

$$\begin{aligned}\mathsf{MinD}(\mathbb{O}^m, \mathbb{B}^{n,m}, \text{IC}) \;:=\; & [\mathsf{sH}(\mathbb{O}^m, \mathbb{B}^{n,m})\,;\mathsf{store}(\mathsf{dis})\,;\mathsf{flip}^{\geq 0}(\mathbb{O}^m)\,;\mathsf{sH}(\mathbb{O}^m, \mathbb{B}^{n,m})] \\ & (\mathsf{dis'} > \mathsf{dis} \rightarrow \neg\text{IC}).\end{aligned}$$

In case the majority outcome satisfies IC, there will be no other outcome which is strictly closer to the profile, and therefore the conditional will always hold vacuously. Otherwise, suppose we are considering an assignment to the propositional variables of the outcome that corresponds to a Kemeny outcome (different from the majority): the majority outcome will have a smaller value for dis, but IC will not be satisfied and thus the conditional will hold again.

A program for the Kemeny rule is then defined in the following way:

$$\begin{aligned}\mathsf{kemeny}_{\text{IC}}(\mathbb{B}^{n,m}) \;:=\; & \bigcup_{0 \leq d \leq m} \Big(\langle \mathsf{flip}^1(\mathbb{O}^m)^d \rangle (\mathsf{MinD}(\mathbb{O}^m, \mathbb{B}^{n,m}, \text{IC}) \wedge \text{IC})?\,;\mathsf{flip}^1(\mathbb{O}^m)^d \Big)\,; \\ & \mathsf{MinD}(\mathbb{O}^m, \mathbb{B}^{n,m}, \text{IC}) \wedge \text{IC}?.\end{aligned}$$

Recall that the propositional variables in $\mathbb{O}^m$ are always set to false by default. The program first guesses some d such that after flipping the truth value of d propositional

variables in the outcome, this outcome achieves minimal Hamming distance from the rest of the profile and IC is satisfied. Then, the program flips the truth value of d propositional variables in the outcome. The final test ensures that just in case the truth value of the correct d propositional variables has been flipped, such that the outcome satisfies IC and it has minimal distance to the profile, the program does not fail.

Proposition 9. *Let $\boldsymbol{B}$ be a rational profile for $\mathcal{N} = \{1, \ldots, n\}$, $\mathcal{I} = \{1, \ldots, m\}$ and some* IC*. Let $v_{\boldsymbol{B}}$ be the valuation corresponding to $\boldsymbol{B}$ and let $\mathsf{f} = \mathsf{kemeny}_{\mathrm{IC}}$. Let $V^{\mathsf{f}}_{v_{\boldsymbol{B}}} = \{v' \mid (v_{\boldsymbol{B}}, v') \in \| \mathsf{f}(\mathbb{B}^{n,m}) \|\}$. Then, the function $g : \mathrm{Kemeny}_{\mathrm{IC}}(\boldsymbol{B}) \to V^{\mathsf{f}}_{v_{\boldsymbol{B}}}$ such that if $g(B) = v'$ then $b_j = 1 \iff p_j \in v'$ for all $j \in \mathcal{I}$ is a bijection.*

Slater Rule

The outcome of the Slater rule contains those ballots satisfying the given IC and minimising the Hamming distance from the outcome of the majority rule for that profile. That is, given a profile $\boldsymbol{B}$ we have:

$$\mathrm{Slater}_{\mathrm{IC}}(\boldsymbol{B}) = \operatorname*{argmin}_{B \models \mathrm{IC}} H(B, \mathrm{Maj}(\boldsymbol{B})).$$

Example 6. *Though Kemeny and Slater might seem very similar rules, their outcomes need not necessarily coincide. Consider the following example for $\mathcal{N} = \{1, \ldots, 13\}$ and $\mathcal{I} = \{1, 2, 3, 4\}$. Let the integrity constraint be* IC $= (p_3 \leftrightarrow p_1 \downarrow p_2) \wedge (p_4 \leftrightarrow p_1 \oplus p_2)$*, where $\varphi \downarrow \psi \ := \ \neg(\varphi \vee \psi)$.*

	p_1	p_2	p_3	p_4
3 voters	1	1	0	0
3 voters	1	0	0	1
3 voters	0	1	0	1
4 voters	0	0	1	0
Maj($\boldsymbol{B}$)	0	0	0	0

The outcome of the majority does not satisfy the constraint. The outcome of the Slater rule is $(0, 0, 1, 0)$, because the Hamming distance d from the majority outcome is only 1. The outcome of the Kemeny rule, on the other hand, is one of the ballots in the set $\{(1, 1, 0, 0), (0, 1, 0, 1), (1, 0, 0, 1)\}$. In fact, in order to maximise agreement with the profile we have to reject issue p_3, since nine agents rejected it in their individual ballots.

The Slater rule is expressible in DL-PA by the following program:

$$\mathsf{slater}_{\mathrm{IC}}(\mathbb{B}^{n,m}) \;:=\; \mathsf{maj}(\mathbb{B}^{n,m})\,;\; \bigcup_{0\leq d\leq m} \Big(\mathsf{H}(\mathrm{IC},\mathbb{O}^m,{\geq}d)?\,;\mathsf{flip}^1(\mathbb{O}^m)^d\Big)\,;\mathrm{IC}?.$$

The program first computes the majority outcome for that profile; then, it finds the minimal Hamming distance between the current outcome and one where IC is satisfied. In case the majority outcome already satisfies the constraint, we have that $d = 0$.

Proposition 10. *Let $\boldsymbol{B}$ be a rational profile for $\mathcal{N} = \{1,\ldots,n\}$, $\mathcal{I} = \{1,\ldots,m\}$ and some* IC*. Let $v_{\boldsymbol{B}}$ be the valuation corresponding to $\boldsymbol{B}$ and let $\mathsf{f} = \mathsf{slater}_{\mathrm{IC}}$. Let $V^{\mathsf{f}}_{v_{\boldsymbol{B}}} = \{v' \mid (v_{\boldsymbol{B}}, v') \in \|\,\mathsf{f}(\mathbb{B}^{n,m})\,\|\}$. Then, the function $g : \mathrm{Slater}_{\mathrm{IC}}(\boldsymbol{B}) \to V^{\mathsf{f}}_{v_{\boldsymbol{B}}}$ such that if $g(B) = v'$ then $b_j = 1 \iff p_j \in v'$ for all $j \in \mathcal{I}$ is a bijection.*

Ranked Pairs Rule

We need some additional notation to define the ranked pairs rule (RPR) [Endriss and Grandi, 2014]. The *majority strength* of issue j in profile $\boldsymbol{B}$ is defined as $\mathrm{MS}^{\boldsymbol{B}}(j) = \max\{|N^{\boldsymbol{B}}_{j:0}|, |N^{\boldsymbol{B}}_{j:1}|\}$. Hence, if a majority of agents accepts j in the profile the majority strength of j is the total number of these agents; and similarly if a majority of agents rejects j.

Therefore, we can order the issues according to their majority strength with $\succ^{\boldsymbol{B}}_{\tau}$, where $\tau : \mathcal{I} \to \mathcal{I}$ is a permutation that breaks the ties, in case two issues have the same majority strength. We let $\ell \subseteq B$ mean that $\ell(j) = x$ implies $b_j = x$, for partial function $\ell : \mathcal{I} \to \{0,1\}$ and ballot B.

Example 7. *Consider the following profile $\boldsymbol{B}$ for set of issues $\mathcal{I} = \{1,2,3\}$, set of agents $\mathcal{N} = \{1,\ldots,5\}$ and constraint* IC $= p_2 \to (\neg p_1 \vee \neg p_3)$.

	p_1	p_2	p_3
Agent 1	1	1	0
Agent 2	1	1	0
Agent 3	1	1	0
Agent 4	0	1	1
Agent 5	0	1	1

Notice that $\mathrm{MS}^{\boldsymbol{B}}(1) = 3$, $\mathrm{MS}^{\boldsymbol{B}}(2) = 5$ *and* $\mathrm{MS}^{\boldsymbol{B}}(3) = 3$. *Suppose that τ breaks the ties in lexicographic order: we have that $2 \succ^{\boldsymbol{B}}_{\tau} 1 \succ^{\boldsymbol{B}}_{\tau} 3$. Consider now the partial function ℓ defined only for the first two issues, so that $\ell(1) = 0$ and $\ell(2) = 1$. We can see that for ballots $B' = (0,1,0)$ and $B'' = (0,1,1)$ we have $\ell \subseteq B'$ and $\ell \subseteq B''$.*

Given profile $\boldsymbol{B}$, integrity constraint IC and permutation $\tau : \mathcal{I} \to \mathcal{I}$, we define the total function $\ell^{\boldsymbol{B}}_{\tau,\mathrm{IC}}$ as follows:

$$\textbf{for } j \in \mathcal{I}, \text{ following order } \succ^{\boldsymbol{B}}_{\tau} \textbf{ do}$$

$$\ell^{\boldsymbol{B}}_{\tau,\mathrm{IC}}(j) \;:=\; \begin{cases} \mathrm{Maj}(\boldsymbol{B})_j & \text{if } \exists B \models \mathrm{IC} \text{ such that } \ell^{\boldsymbol{B}}_{\tau,\mathrm{IC}} \subseteq B \\ 1 - \mathrm{Maj}(\boldsymbol{B})_j & \text{otherwise} \end{cases}$$

The RPR thus returns a ballot given by the above procedure, when some tie-breaking rule and an integrity constraint are provided. Namely,

$$\mathrm{RP}_{\mathrm{IC}}(\boldsymbol{B}) \;=\; \{\ell^{\boldsymbol{B}}_{\tau,\mathrm{IC}} \mid \tau \text{ is a permutation on } \mathcal{I}\}.$$

We present here a DL-PA program computing the RPR for the special case of no ties in the majority strengths of the issues in the profile. For any tie-breaking rule τ we could devise an additional program whose role would be to modify the count of the majority strengths for the issues in such a way that they are strictly ordered according to τ.

We first define a program computing the majority strength of the issues in profile $\boldsymbol{B}$, assuming the result of the majority rule has already been computed.

$$\begin{aligned} \mathsf{majSt}(\mathbb{B}^{n,m}) \;:=\; \mathop{;}_{j\in\mathcal{I}} \Big(\mathsf{zero}(\mathsf{MS}_j)\,;\mathsf{if}\ p_j\ \mathsf{then}\ \Big(\mathop{;}_{i\in\mathcal{N}} (\mathsf{if}\ p_{ij}\ \mathsf{do}\ \mathsf{incr}(\mathsf{MS}_j))\Big) \\ \mathsf{else}\ \Big(\mathop{;}_{i\in\mathcal{N}} (\mathsf{if}\ \neg p_{ij}\ \mathsf{do}\ \mathsf{incr}(\mathsf{MS}_j))\Big)\Big). \end{aligned}$$

Proceeding issue by issue, the program checks whether the propositional variable in the outcome for that issue j is true or false, and then counts in the profile how many agents agreed with the outcome — storing the result in the MS_j counter.

We now define the sets Q_i, for $i \in \mathcal{I}$, in order to add step-by-step the issues according to their majority strength (starting from the strongest and ending with the weakest). Since we assumed that there are no ties, at each step we are going to add a singleton.

$$\begin{aligned} &Q_1 := \{p_j \mid |\mathrm{MS}^{\boldsymbol{B}}_j| \geq |\mathrm{MS}^{\boldsymbol{B}}_{j'}| \text{ for all } j' \in \mathcal{I}\} \\ &Q_k := Q_{k-1} \cup \{p_j \mid p_j \notin Q_{k-1} \text{ and for all } j' \in \mathcal{I} \text{ such that } p_{j'} \notin Q_{k-1}, |\mathrm{MS}^{\boldsymbol{B}}_j| \geq |\mathrm{MS}^{\boldsymbol{B}}_{j'}|\} \end{aligned}$$

The DL-PA program computing the RPR is thus defined as follows:

$$\begin{aligned} \mathsf{rp}^{\tau}_{\mathrm{IC}}(\mathbb{B}^{n,m}) \;:=\; \mathsf{maj}(\mathbb{B}^{n,m})\,;\mathsf{if}\ \neg\mathrm{IC}\ \mathsf{do}\ \Big(\mathsf{majSt}(\mathbb{B}^{n,m})\,;\mathop{;}_{1\leq i\leq m}\Big(\bigcup_{j\in\mathcal{I}}\big(\bigwedge_{l\in\mathcal{I}} \mathsf{MS}_j \geq \mathsf{MS}_l?; \\ \mathsf{if}\ \neg\langle\mathsf{flip}^{\geq 0}(\mathbb{O}^m \setminus Q_i)\rangle\mathrm{IC}\ \mathsf{do}\ p_j \leftarrow \neg p_j\,;\mathsf{zero}(\mathsf{MS}_j)\big)\Big)\Big). \end{aligned}$$

The program first computes the majority outcome. In case IC is not satisfied yet, the program calculates the majority strenght of all the issues. Then, for m times, the program executes the following procedure.

First, the issue j with the highest majority strength at that stage is selected. The program checks whether it is possible to modify the outcome in some way (without changing the truth value of j *and* of all the issues already inspected at some previous step) to make IC satisfied. In case this is not possible, the truth value of the propositional variable of the outcome for j is flipped — and the value of its majority strenght is set to zero. In the following step, thus, the nondeterministic choice operator will select the next issue in the ranking; this is possible because this issue is now the one with highest majority strength, since the one of j has been set to zero.

Proposition 11. *Let $\boldsymbol{B}$ be a rational profile for $\mathcal{N} = \{1, \dots, n\}$, $\mathcal{I} = \{1, \dots, m\}$ and some* IC*. Let $v_{\boldsymbol{B}}$ be the valuation corresponding to $\boldsymbol{B}$ and let $\mathsf{f} = \mathsf{rp}^{\tau}_{\mathrm{IC}}$. Let $V^{\mathsf{f}}_{v_{\boldsymbol{B}}} = \{v' \mid (v_{\boldsymbol{B}}, v') \in \| \mathsf{f}(\mathbb{B}^{n,m}) \|\}$. Then, the function $g : \mathrm{RP}^{\tau}_{\mathrm{IC}}(\boldsymbol{B}) \to V^{\mathsf{f}}_{v_{\boldsymbol{B}}}$ such that if $g(B) = v'$ then $b_j = 1 \iff p_j \in v'$ for all $j \in \mathcal{I}$ is a bijection.*

2.5 Representative Voter Rules

We present in this section rules that select the most representative voter in a profile, according to different principles. They are all based on the PA rules presented in Section 2.4: this is the reason why here we will often refer to notions and programs introduced before, rather than redefining them. We start by defining the Average-Voter rule [Endriss and Grandi, 2014], also known in the literature as the Distance-Based Generalised Dictatorship [Grandi and Endriss, 2011]. Then, we present the Majority-Voter rule [Endriss and Grandi, 2014], and we conclude with the Ranked-Voter rule. These rules correspond, respectively, to the Kemeny rule, the Slater rule, and the Ranked-Pairs rule.

Average-Voter Rule

The Average-Voter rule (AVR) returns as outcome the individual ballot of the agents in the profile such that the sum of the Hamming distance between their ballots and the ones of the other agents is minimal. Namely, for profile $\boldsymbol{B}$, we have:

$$\mathrm{AVR}(\boldsymbol{B}) = \operatorname*{argmin}_{\{B_i \mid i \in \mathcal{N}\}} \sum_{k \in \mathcal{N}} H(B_i, B_k).$$

We here provide two alternative programs to compute the AVR in DL-PA.

Initial sum. In this version, we define a program that computes the sum of the Hamming distance between a given individual ballot B_i and the rest of the profile $\boldsymbol{B}$. We define $\mathbb{V}_i^m := \{p_{ij} \mid j \in \mathcal{I}\}$ as the set of propositional variables encoding the decision of agent i on the issues. For each agent i, a counter dis_i, which can take at most value $m \cdot n$, is used. Notice that we could as well have included agent i's ballot in the iteration, since her Hamming distance to her own ballot is 0: we leave it out to make the program shorter.

$$\mathsf{sH}^*(\mathbb{V}_i^m, \mathbb{B}^{n,m}) := \mathop{;}_{k \in \mathcal{N} \setminus \{i\}} \Big(\mathsf{zero}(\mathsf{dis}_i) \,;\, \mathop{;}_{j \in \mathcal{I}} \mathsf{if}\ p_{ij} \oplus p_{kj}\ \mathsf{do}\ \mathsf{incr}(\mathsf{dis}_i) \Big).$$

A program expressing AVR in DL-PA is the following:

$$\mathsf{avr}^*(\mathbb{B}^{n,m}) := \mathop{;}_{k \in \mathcal{N}} \mathsf{sH}^*(\mathbb{V}_k^m, \mathbb{B}^{n,m}) \,;\, \bigcup_{i \in \mathcal{N}} \Big((\bigwedge_{k \in \mathcal{N}} \mathsf{dis}_k \geq \mathsf{dis}_i)? \,;\, \mathop{;}_{j \in \mathcal{I}} p_j \leftarrow p_{ij} \Big).$$

First, the program computes the sum of the Hamming distances to the profile for each individual ballot B_k. Then, it nondeterministically chooses the agent i whose Hamming distance to the profile is lower than or equal to the ones of the other agents. Finally, it copies her individual ballot into the outcome.

Kemeny adaptation. As the name suggests, this program to express the AVR in DL-PA is quite similar to the one provided for the Kemeny rule. Since we have as an assumption that the agents satisfy IC in their individual ballots, we will only check that the outcome is equal to the individual ballot of some agent, which implies that it is consistent.

The following formula is true if and only if there is some agent who is a *representative* for the other agents, in the sense that her individual ballot is equal to the outcome.

$$\mathsf{Repr} := \bigvee_{i \in \mathcal{N}} \Big(\bigwedge_{j \in \mathcal{I}} p_{ij} \leftrightarrow p_j \Big).$$

We refer to Section 2.4 for an explanation of how the following program works, based on the program used to express the Kemeny rule. Notice that the only difference between the two programs lies in the use of the Repr formula instead of IC.

$$\mathsf{avr}(\mathbb{B}^{n,m}) := \bigcup_{0 \leq d \leq m} \Big(\langle \mathsf{flip}^1(\mathbb{O}^m)^d \rangle (\mathsf{MinD}(\mathbb{O}^m, \mathbb{B}^{n,m}, \mathsf{Repr}) \wedge \mathsf{Repr})? \,;\, \mathsf{flip}^1(\mathbb{O}^m)^d \Big) \,;$$
$$\mathsf{MinD}(\mathbb{O}^m, \mathbb{B}^{n,m}, \mathsf{Repr}) \wedge \mathsf{Repr}?.$$

Proposition 12. *Let $\boldsymbol{B}$ be a rational profile for $\mathcal{N} = \{1, \dots, n\}$ and $\mathcal{I} = \{1, \dots, m\}$. Let $v_{\boldsymbol{B}}$ be the valuation corresponding to $\boldsymbol{B}$ and let $\mathsf{f} \in \{\mathsf{avr}, \mathsf{avr}^*\}$. Let $V^{\mathsf{f}}_{v_{\boldsymbol{B}}} = \{v' \mid (v_{\boldsymbol{B}}, v') \in \| \mathsf{f}(\mathbb{B}^{n,m}) \|\}$. Then, the function $g : \mathrm{AVR}(\boldsymbol{B}) \to V^{\mathsf{f}}_{v_{\boldsymbol{B}}}$ such that if $g(B) = v'$ then $b_j = 1 \iff p_j \in v'$ for all $j \in \mathcal{I}$ is a bijection.*

Majority-Voter Rule

The Majority-Voter rule (MVR) returns as the outcome the individual ballot of some agent in the profile such that the Hamming distance between her ballot and the outcome of the majority rule for that profile is minimal. Namely, for profile $\boldsymbol{B}$, we have that:

$$\mathrm{MVR}(\boldsymbol{B}) = \operatorname*{argmin}_{\{B_i \mid i \in \mathcal{N}\}} H(B_i, \mathrm{Maj}(\boldsymbol{B})).$$

Here as well two alternative programs for computing the MVR in DL-PA are possible. The first one is an adaptation of the avr^* program: we first compute the outcome of the majority rule and then use the program sH^* to compute the distance between each individual ballot and the outcome of the majority (instead of the rest of the profile).

The second version, on the other hand, is an adaptation of the program provided for the Slater rule in Section 2.4. Namely, we have:

$$\mathsf{mvr}(\mathbb{B}^{n,m}) \;:=\; \mathsf{maj}(\mathbb{B}^{n,m})\,;\; \bigcup_{0 \le d \le m} \Big(\mathsf{H}(\mathsf{Repr}, \mathbb{O}^m, {\geq}d)?\,;\mathsf{flip}^1(\mathbb{O}^m)^d\Big)\,;\mathsf{Repr}?.$$

The program first computes the outcome of the majority rule. Then, it finds the minimal number d of propositional variables of the outcome whose truth value has to be flipped in order to obtain an outcome equal to the individual ballot of some agent in the profile.

Proposition 13. *Let $\boldsymbol{B}$ be a rational profile for $\mathcal{N} = \{1, \dots, n\}$ and $\mathcal{I} = \{1, \dots, m\}$. Let $v_{\boldsymbol{B}}$ be the valuation corresponding to $\boldsymbol{B}$ and let $\mathsf{f} = \mathsf{mvr}$. Let $V^{\mathsf{f}}_{v_{\boldsymbol{B}}} = \{v' \mid (v_{\boldsymbol{B}}, v') \in \| \mathsf{f}(\mathbb{B}^{n,m}) \|\}$. Then, the function $g : \mathrm{MVR}(\boldsymbol{B}) \to V^{\mathsf{f}}_{v_{\boldsymbol{B}}}$ such that if $g(B) = v'$ then $b_j = 1 \iff p_j \in v'$ for all $j \in \mathcal{I}$ is a bijection.*

Ranked-Voter Rule

The outcome of the Ranked-Voter rule (RVR) is obtained with a procedure analogous to the one provided for the RPR in Section 2.4. We proceed in an identical fashion up to the construction of the total function $\ell^{\boldsymbol{B}}_{\tau}$. Here, instead of checking the existence of

a ballot satisfying the given integrity constraint, we check whether there is some agent $i \in \mathcal{N}$ such that $\ell_\tau^{\boldsymbol{B}} \subseteq B_i$.

Hence, the RVR for profile $\boldsymbol{B}$ returns the ballot selected by the described procedure given some tie-breaking rule.

$$\mathrm{RVR}(\boldsymbol{B}) \;=\; \{\ell_\tau^{\boldsymbol{B}} \mid \tau \text{ is a permutation on } \mathcal{I}\}.$$

As for the RPR, here as well we assume that there are no ties in the majority strength of the issues: we refer to Section 2.4 for a discussion of this point.

$$\mathsf{rvr}^\tau(\mathbb{B}^{n,m}) \;:= \mathsf{maj}(\mathbb{B}^{n,m})\,;\mathsf{if}\ \neg\mathsf{Repr}\ \mathsf{do}\ \Big(\mathsf{majSt}(\mathbb{B}^{n,m})\,;\ \mathop{;}_{1\le i\le m}\ \Big(\bigcup_{j\in\mathcal{I}}\big(\bigwedge_{l\in\mathcal{I}}\mathsf{MS}_j \ge \mathsf{MS}_l?\,;$$
$$\mathsf{if}\ \neg\langle\mathsf{flip}^{\ge 0}(\mathbb{O}^m \setminus Q_i)\rangle\mathsf{Repr}\ \mathsf{do}\ p_j \leftarrow \neg p_j\,;\mathsf{zero}(\mathsf{MS}_j)\big)\Big)\Big).$$

The program works in an analogous way to the one provided for the RPR, with the sole exception that instead of the integrity constraint, we check whether the outcome corresponds to the individual ballot of some agent in the profile.

Proposition 14. *Let $\boldsymbol{B}$ be a rational profile for $\mathcal{N} = \{1,\dots,n\}$ and $\mathcal{I} = \{1,\dots,m\}$. Let $v_{\boldsymbol{B}}$ be the valuation corresponding to $\boldsymbol{B}$ and let $\mathsf{f} = \mathsf{rvr}^\tau$. Let $V^{\mathsf{f}}_{v_{\boldsymbol{B}}} = \{v' \mid (v_{\boldsymbol{B}}, v') \in \| \mathsf{f}(\mathbb{B}^{n,m}) \|\}$. Then, the function $g : \mathrm{RVR}^\tau(\boldsymbol{B}) \to V^{\mathsf{f}}_{v_{\boldsymbol{B}}}$ such that if $g(B) = v'$ then $b_j = 1 \iff p_j \in v'$ for all $j \in \mathcal{I}$ is a bijection.*

2.6 Summary

In this chapter we provided several DL-PA programs expressing different aggregation rules. We distinguished between *simple* rules, rules based on *maximisation and minimisation*, rules inspired by *Preference Aggregation* and rules selecting a *representative voter* in the profile. After having introduced each of these rules, their translation into DL-PA has been linked to the BA definition via some propositions ensuring the correctness of our work and we provided a proof of some of them.

When relevant, we presented multiple possible translations for the same rule. Except for the program maj^*, computing the majority rule, all the programs presented here do not modify the truth value of the propositional variables in $\mathbb{B}^{n,m}$ and they never make use of the Kleene star operator.

Moreover, most programs do not need counters. The programs that do make use of them are: $\mathsf{parity}_{\mathsf{Even}}$, maj, $\mathsf{quota}_{q_1,\dots,q_n}$, $\mathsf{kemeny}_{\mathrm{IC}}$, $\mathsf{rp}^\tau_{\mathrm{IC}}$, avr, avr^* and rvr^τ. In fact, the first needs to inspect the parity of the coalition of supporters of the issues, the second and the

third need to check whether the coalition of supporters is above a certain threshold, and in the last cases either the majority strength of the issues or some Hamming distance has to be computed to output the result.

It is also interesting to consider how the length of a program behaves as the number of agents or the number of issues increases. For the programs $\mathsf{constant}_B$ and $\mathsf{dictatorship}_i$, computing the constant rules and the dictatorship of agent i respectively, an increase in the number of agents does not affect their length. This is to be expected because the programs only consider the constant and the dictatorial ballots while ignoring the rest of the profile.

On the other hand, the length of the program $\mathsf{msa}_{\mathrm{IC}}$ computing the maximum subagenda rule grows exponentially as the number of issues increases. In fact, inside the program there is a non-deterministic choice operator over all the possible subsets of the set $\mathbb{O}^m$ for the propositional variables of the outcome. Notice however that we provided another translation for the same rule, and no other program has this problem.

In conclusion, this chapter has focused on specific aggregation rules and how to make sure that the DL-PA programs correctly compute their outcomes. In the next chapter we will take a step back from the details of each program: we will look at more general properties that we would want aggregation rules to satisfy and how to express them in DL-PA.

Chapter 3

Axioms in DL-PA

In Chapter 2 we have seen how to tranlsate aggregation procedures as DL-PA programs. The presented techniques could be applied to any other aggregation rule we may want to encode, and then we could analyse the properties of the program we came up with. Alternatively, we might be interested in studying aggregation procedures at a higher level: namely, according to some general properties they satisfy. A research direction in JA tries to find a characterisation of aggregation rules based on these properties, in the sense that a rule belongs to a certain family if and only if it satisfies a specific set of properties [Dietrich and List, 2007; May, 1952].

In the literature, these properties are usually referred to as *axioms* [Endriss et al., 2012; Grandi and Endriss, 2011; List, 2012]. We here distinguish between *single-profile* and *multi-profile* axioms. The former type expresses a relationship existing between the structure of a profile and the outcome of a rule on that profile. The latter, on the other hand, relates the structure of two profiles to the outcomes of some rule applied on them. A similar distinction is found in Preference Aggregation as well, sometimes under the names of *intra-profile* and *inter-profiles* conditions [Rubinstein, 1984]. As we will see, in our case this is not only a useful conceptual separation, but the two types of axioms will require a different formal treatment.

The Chapter thus focuses on expressing JA axioms in a formal language, a task that has been also tackled in Judgment Aggregation Logic for the axioms of (positive) Unanimity, Independence and Non-Dictatorship [Ågotnes et al., 2011]. We start in Section 3.1 by discussing in a general way how to inspect whether a given aggregation rule satisfies an axiom or not. Then, in Section 3.2 we introduce the single-profile axioms of Unanimity, Issue-Neutrality, Domain-Neutrality and N-Monotonicity. In Section 3.3 we present the multi-profile axioms of Independence, I-Monotonicity and Anonimity. For each axiom we state a proposition ensuring the correctness of our translation, and we

provide a proof for one single-profile and one multi-profile axiom, since they all share a similar structure. We conclude in Section 3.4 with a summary of the Chapter.

3.1 Axioms and Rules

We defined aggregation procedures with respect to a set of agents $\mathcal{N} = \{1, \ldots, n\}$ and a set of issues $\mathcal{I} = \{1, \ldots, m\}$. This implies that when we say that a certain rule satisfies an axiom, we mean that for every possible profile $\boldsymbol{B}$ on m issues, that property must hold. Therefore, all the axioms we are going to study begin with the expression "for any profile $\boldsymbol{B}$". We discuss here how this can be formulated in DL-PA.

Given a set of agents $\mathcal{N}$, a set of issues $\mathcal{I}$, and an integrity constraint IC, we let $\mathsf{f}(\mathbb{B}^{n,m})$ be the program translating aggregation rule F in DL-PA. In case rule F satisfies axioms $A_1, \ldots, A_k$ we will have that the following holds:

$$\models [\mathsf{prof}_{\mathrm{IC}}(\mathbb{B}^{n,m}, \mathbb{O}^m)\,;\mathsf{f}(\mathbb{B}^{n,m})] \bigwedge_{1 \leq i \leq k} \mathsf{A}_i$$

where $\mathsf{A}_1, \ldots, \mathsf{A}_k$ are the translations of the axioms in DL-PA. Namely, the problem of checking whether a rule satisfies some axioms boils down to inspecting whether the above formula is valid, i.e. true for every valuation v.

In fact, either the arbitrary valuation v under consideration corresponds to a rational profile or it does not. In the first case, every individual ballot satisfies IC and the propositional variables for the outcome are set to false with the execution of program $\mathsf{prof}_{\mathrm{IC}}(\mathbb{B}^{n,m}, \mathbb{O}^m)$. Then, the program corresponding to rule F is run and we get to a valuation v' which has to make true all the formulas translating the axioms satisfied by F. In the second case, the program $\mathsf{prof}_{\mathrm{IC}}(\mathbb{B}^{n,m}, \mathbb{O}^m)$ inside the modality fails and the formula thus holds vacuously.

At a first glance, it might seem odd that even though single-profile and multi-profile axioms have an inherently different nature we use the same general formula to check whether a rule satisfies them. More precisely, we might be wondering how can we be referring to two different outcomes if the program translating a rule appears only once inside the modality of the formula presented above. The answer, as we will see, is that the program executing the rule is called again *inside* the formulas expressing multi-profile axioms — a simple device allowed by the language of DL-PA that lets us compare two different outcomes at the same time. Notice that analogous approaches have been used for expressing inter-profile conditions in PA [Ciná and Endriss, 2015].

A couple of brief remarks before proceeding. First, in what follows we assume that the programs translating aggregation rules do not modify the truth value of the propositional variables in $\mathbb{B}^{n,m}$. Hence, we exclude all programs like maj^*, expressing the majority rule, presented in Chapter 2. However, we could recover them by using some additional propositonal variables to store the initial profile and then execute the program for rule F.

Secondly, notice that we could have defined an aggregation rule F as a function taking *any* profile, and not just the rational ones, as input. In this case, we would not have to check that the agents satisfy IC in their individual ballots and we could replace program $\mathsf{prof}_{\mathrm{IC}}(\mathbb{B}^{n,m}, \mathbb{O}^m)$ with $\mathsf{init}(\mathbb{O}^m)$ from the translation of all the axioms and from the formula introduced above, thus obtaining $\models [\mathsf{init}(\mathbb{O}^m)\,;\mathsf{f}(\mathbb{B}^{n,m})] \bigwedge_{1\leq i\leq k} \mathsf{A}_i$.

3.2 Single-profile Axioms

In this section we present axioms that make reference to just one profile at a time. This means that to check whether an aggregation rule F satisfies a single-profile axiom it is sufficient to inspect the outcome of the rule on some profile and the profile itself.

Since in our framework profiles correspond to valuations, and since for single-profile axioms we do not have to make reference to different valuations (corresponding to different profiles), it is possible to express them as simple propositional formulas. Therefore, in order to express Unanimity, Issue-Neutrality, Domain-Neutrality and N-Monotonicity we will not need to use the modal part of our language.

Unanimity

A rule F is *unanimous* if in case all agents agree on some issue j, the outcome of F for issue j agrees with them. Namely, issue j should be collectively accepted if all the agents accept it, and it should be rejected if they all reject it in their individual ballots. This axiom can also be separated into two different versions, depending on whether all agents accept or reject j: we speak of *positive* or *negative* unanimity, respectively.

Formally, the Unanimity axiom states that for rule F:

$$\begin{aligned} \mathrm{U} \;:\; &\text{For any } \boldsymbol{B}, \text{ for all } j \in \mathcal{I} \text{ and for } x \in \{0,1\}, \\ &\text{if } b_{ij} = x \text{ for all } i \in \mathcal{N} \text{ then } F(\boldsymbol{B})_j = x. \end{aligned}$$

We can express Unanimity as a propositional formula in the following way:

$$\mathsf{U} := \bigwedge_{j\in\mathcal{I}} \Big(\big((\bigwedge_{i\in\mathcal{N}} p_{ij}) \to p_j \big) \wedge \big((\bigwedge_{i\in\mathcal{N}} \neg p_{ij}) \to \neg p_j \big) \Big).$$

The first big conjunction ensures that what follows is going to hold for any issue. Inside the parentheses, the two conjuncts deal with the positive and the negative version of unanimity. Now we can show that our translation of the axiom is correct with the proof of the proposition below.

Proposition 15. *Let $\mathcal{N} = \{1, \dots, n\}$ and $\mathcal{I} = \{1, \dots, m\}$ be the sets of agents and issues, respectively. Let $\mathbb{B}^{n,m}$ be the set of propositional variables encoding the opinions of agents in $\mathcal{N}$ on issues in $\mathcal{I}$. Let f be a* DL-PA *program translating some rule F. Then,*

$$\mathrm{U} \textit{ holds} \iff \models [\mathsf{prof}_{\mathrm{IC}}(\mathbb{B}^{n,m}, \mathbb{O}^m)\,;\mathsf{f}(\mathbb{B}^{n,m})]\mathsf{U}.$$

Proof. For the left-to-right direction, consider an arbitrary valuation v and an arbitrary valuation v' such that $(v, v') \in \| \mathsf{prof}_{\mathrm{IC}}(\mathbb{B}^{n,m}, \mathbb{O}^m)\,;\mathsf{f}(\mathbb{B}^{n,m}) \|$. We want to show that $v' \in \| \mathsf{U} \|$, or more precisely that:

$$v' \in \| \bigwedge_{j \in \mathcal{I}} \Big(((\bigwedge_{i \in \mathcal{N}} p_{ij}) \to p_j) \wedge ((\bigwedge_{i \in \mathcal{N}} \neg p_{ij}) \to \neg p_j) \Big) \| \,.$$

Suppose, for reductio, that $v' \notin \| \mathsf{U} \|$. Given the definition of the interpretation, this means that there is some $j \in \mathcal{I}$ such that $v' \notin \| ((\bigwedge_{i \in \mathcal{N}} p_{ij}) \to p_j) \wedge ((\bigwedge_{i \in \mathcal{N}} \neg p_{ij}) \to \neg p_j) \|$. Assume, without loss of generality, that $v' \notin \| (\bigwedge_{i \in \mathcal{N}} p_{ij}) \to p_j) \|$ is the case (a similar reasoning can be done in the other case). Hence, we have that $v' \in \| \bigwedge_{i \in \mathcal{N}} p_{ij} \|$ and $v' \notin \| p_j \|$.

Since $(v, v') \in \| \mathsf{prof}_{\mathrm{IC}}(\mathbb{B}^{n,m}, \mathbb{O}^m)\,;\mathsf{f}(\mathbb{B}^{n,m}) \|$, we have that v corresponds to some rational profile $\boldsymbol{B}$ and v' corresponds to $F(\boldsymbol{B})$. By assumption, we have that F satisfies (U): in particular, this means that for all $j \in \mathcal{I}$, if for all $i \in \mathcal{N}$ we have $b_{ij} = 1$, then $F(\boldsymbol{B})_j = 1$.

Recall that we assumed that v and v' do not differ on the propositional variables in $\mathbb{B}^{n,m}$. Hence, the fact that $v' \in \| \bigwedge_{i \in \mathcal{N}} p_{ij} \|$ implies that $v \in \| \bigwedge_{i \in \mathcal{N}} p_{ij} \|$. Therefore in profile $\boldsymbol{B}$ we have that $b_{ij} = 1$ for all $i \in \mathcal{N}$, which implies that $F(\boldsymbol{B})_j = 1$. On the other hand, this contradicts the fact that v' corresponds to $F(\boldsymbol{B})$ and that $v' \notin \| p_j \|$. Hence, $v' \in \| \mathsf{U} \|$.

For the right-to-left direction, take an arbitrary rational profile $\boldsymbol{B}$. We want to show that for any $j \in \mathcal{I}$, if $b_{ij} = 1$ for all $i \in \mathcal{N}$, then $F(\boldsymbol{B})_j = 1$ (and analogously for $b_{ij} = 0$, so we can focus on this case without loss of generality). Suppose, for reductio, that this is not the case. Hence, for some $j \in \mathcal{I}$ in profile $\boldsymbol{B}$, we have $b_{ij} = 1$ for all $i \in \mathcal{N}$ and $F(\boldsymbol{B})_j = 0$.

Consider now valuations v and v' corresponding to $\boldsymbol{B}$ and $F(\boldsymbol{B})$ respectively. This means that $(v, v') \in \| \mathsf{prof}_{\mathrm{IC}}(\mathbb{B}^{n,m}, \mathbb{O}^m) ; \mathsf{f}(\mathbb{B}^{n,m}) \|$ and that $v' \in \| \mathsf{U} \|$. Since v and v' do not differ on the propositional variables in $\mathbb{B}^{n,m}$, by spelling out the definition of the interpretation we get that $v' \in \| p_j \|$. This contradicts the fact that v' corresponds to $F(\boldsymbol{B})$ and that $F(\mathbf{B})_j = 0$. Therefore, we have that $F(\boldsymbol{B}) = 1$. □

Issue-Neutrality

A rule F is *neutral with respect to the issues* if, whenever two issues are treated in the same way in the input, they are treated in the same way in the output. That is, if the agents express the very same pattern of acceptance or rejection for the issues, the issues should either be both collectively accepted or rejected.

Formally, we have that for rule F:

$$\begin{aligned} \mathrm{N}^{\mathcal{I}} \ : \ & \text{For any two issues } j, k \in \mathcal{I} \text{ and any profile } \boldsymbol{B}, \\ & \text{if for all } i \in \mathcal{N} \ b_{ij} = b_{ik} \text{ then } F(\boldsymbol{B})_j = F(\boldsymbol{B})_k. \end{aligned}$$

A translation of Issue-Neutrality as a propositional formula is then the following:

$$\mathsf{N}^{\mathcal{I}} := \bigwedge_{j \in \mathcal{I}} \bigwedge_{k \in \mathcal{I}} \Big(\big(\bigwedge_{i \in \mathcal{N}} (p_{ij} \leftrightarrow p_{ik}) \big) \rightarrow (p_j \leftrightarrow p_k) \Big).$$

With the first two big conjunctions we can quantify over all possible choices of two issues. The conditional inside the parentheses makes sure that if the agents make the same decisions on the two issues, the output for them will be the same.

Proposition 16. *Let $\mathcal{N} = \{1, \ldots, n\}$ and $\mathcal{I} = \{1, \ldots, m\}$ be the sets of agents and issues, respectively. Let $\mathbb{B}^{n,m}$ be the set of propositional variables encoding the opinions of agents in $\mathcal{N}$ on issues in $\mathcal{I}$. Let f be a* DL-PA *program translating some rule F. Then,*

$$\mathrm{N}^{\mathcal{I}} \textit{ holds} \iff \models [\mathsf{prof}_{\mathrm{IC}}(\mathbb{B}^{n,m}, \mathbb{O}^m) ; \mathsf{f}(\mathbb{B}^{n,m})] \mathsf{N}^{\mathcal{I}}.$$

Domain-Neutrality

An aggregation rule is *neutral with respect to the domain* if, whenever two issues are treated in an opposite way in the input, their output should be opposite. Namely, in case there is an opposite pattern of acceptance or rejection from the agents, one of the issues should be accepted in the outcome and the other rejected.

More formally, we have that for rule F the following holds:

$$\mathrm{N}^{\mathcal{D}} \ : \ \text{For any two issues } j,k \in \mathcal{I} \text{ and any profile } \boldsymbol{B},$$
$$\text{if for all } i \in \mathcal{N} \ b_{ij} = 1 - b_{ik} \text{ then } F(\boldsymbol{B})_j = 1 - F(\boldsymbol{B})_k.$$

A propositional formula expressing Domain-Neutrality is the following:

$$\mathsf{N}^{\mathcal{D}} := \bigwedge_{j\in\mathcal{I}} \bigwedge_{k\in\mathcal{I}} \Big(\big(\bigwedge_{i\in\mathcal{N}} (p_{ij} \leftrightarrow \neg p_{ik}) \big) \rightarrow (p_j \leftrightarrow \neg p_k) \Big).$$

In an analogous way to how we expressed Issue-Neutrality, we first quantify over all the possible choices of two issues. Then, the implication inside the parentheses makes sure that if the issues are treated in an opposite way in the profile, the outcome for them should be opposite.

Proposition 17. *Let $\mathcal{N} = \{1,\dots,n\}$ and $\mathcal{I} = \{1,\dots,m\}$ be the sets of agents and issues, respectively. Let $\mathbb{B}^{n,m}$ be the set of propositional variables encoding the opinions of agents in $\mathcal{N}$ on issues in $\mathcal{I}$. Let f be a* DL-PA *program translating some rule F. Then,*

$$\mathrm{N}^{\mathcal{D}} \textit{ holds} \iff \models [\mathsf{prof}_{\mathrm{IC}}(\mathbb{B}^{n,m}, \mathbb{O}^m)\,;\mathsf{f}(\mathbb{B}^{n,m})]\mathsf{N}^{\mathcal{D}}.$$

N-Monotonicity

An aggregation rule is *neutral-monotonic* if, whenever two issues j and k are such that if an agent in the profile accepts j then she also accepts k, and if there is some agent s that rejects j and accepts k, if j is accepted in the outcome then also k must be accepted in the outcome. The intuitive idea of monotonicity captured here is thus one which compares the pattern of acceptance between two issues *within* the same profile: we will see later a different axiom of monotonicity that instead compares the pattern of acceptance of the same issue *between* two profiles. More formally, we have that the following has to hold for rule F:

$$\mathrm{M}^{\mathrm{N}} := \text{For any two } j,k \in \mathcal{I} \text{ and any } \boldsymbol{B}, \text{ if } b_{ij} = 1 \text{ implies } b_{ik} = 1 \text{ for all } i \in \mathcal{N}$$
$$\text{and there is } s \in \mathcal{N} \text{ such that } b_{sj} = 0 \text{ and } b_{sk} = 1, \text{ then } F(\boldsymbol{B})_j = 1 \text{ implies } F(\boldsymbol{B})_k = 1.$$

A propositional formula expressing N-Monotonicity is as follows:

$$\mathsf{M}^{\mathsf{N}} := \bigwedge_{j\in\mathcal{I}} \bigwedge_{k\in\mathcal{I}} \Big(\big(\bigwedge_{i\in\mathcal{N}} (p_{ij} \rightarrow p_{ik}) \wedge \bigvee_{s\in\mathcal{N}} (\neg p_{sj} \wedge p_{sk}) \big) \rightarrow (p_j \rightarrow p_k) \Big).$$

First, we quantify over all the possible choices of two issues j and k. The antecedent inside the parentheses is going to be true if and only if all the agents who accept j also

accept k and if there is some agent who rejects j but accepts k. Then, the consequent is true if in case j is accepted in the outcome, also k is accepted in the outcome.

Proposition 18. *Let $\mathcal{N} = \{1, \dots, n\}$ and $\mathcal{I} = \{1, \dots, m\}$ be the sets of agents and issues, respectively. Let $\mathbb{B}^{n,m}$ be the set of propositional variables encoding the opinions of agents in $\mathcal{N}$ on issues in $\mathcal{I}$. Let f be a* DL-PA *program translating some rule F. Then,*

$$\mathrm{M}^{\mathrm{N}} \textit{ holds} \iff \models [\mathsf{prof}_{\mathrm{IC}}(\mathbb{B}^{n,m}, \mathbb{O}^m)\,;\mathsf{f}(\mathbb{B}^{n,m})]\mathsf{M}^{\mathsf{N}}.$$

3.3 Multi-profile Axioms

In this section we present axioms that refer to more than one profile at the same time. Namely, to check whether an aggregation rule F satisfies one of them, we have to inspect and compare the outcomes of the rule on different profiles. Dealing with more than one profile means referring to more than one valuation, and in particular applying the program expressing rule F multiple times to compare (possibly) different results. For this reason, propositional logic alone is not sufficient anymore to express multi-profile axioms and we will make use of DL-PA formulas for translating Independence, I-Monotonicity and Anonimity.

Independence

A rule F is *independent* if, whenever an issue j is treated in the same way in two profiles, the outcome of the rule for j should be identical in the two profiles. That is, we look at the pattern of acceptance or rejection for just that issue in the two profiles and the two outcomes should correspond.

More formally, we have that an independent rule F is such that:

$$\begin{aligned}&\mathrm{I}\ :\ \text{For any issue } j \in \mathcal{I} \text{ and profiles } \boldsymbol{B} \text{ and } \boldsymbol{B}'\\ &\text{if } b_{ij} = b'_{ij} \text{ for all } i \in \mathcal{N}, \text{ then } F(\boldsymbol{B})_j = F(\boldsymbol{B}')_j.\end{aligned}$$

By letting $\mathbb{B}^n_j := \{p_{ij} \mid i \in \mathcal{N}\}$ be the set of propositional variables encoding the opinion of the agents on j and by letting f be a program expressing the voting rule F in DL-PA, we have that Independence can be expressed in the following way:

$$\begin{aligned}\mathsf{I} := \bigwedge_{j \in \mathcal{I}} \Big(&(p_j \to [\mathsf{flip}^{\geq 0}(\mathbb{B}^{n,m} \setminus \mathbb{B}^n_j)\,;\mathsf{prof}_{\mathrm{IC}}(\mathbb{B}^{n,m}, \mathbb{O}^m)\,;\mathsf{f}(\mathbb{B}^{n,m})]p_j) \wedge\\ &(\neg p_j \to [\mathsf{flip}^{\geq 0}(\mathbb{B}^{n,m} \setminus \mathbb{B}^n_j)\,;\mathsf{prof}_{\mathrm{IC}}(\mathbb{B}^{n,m}, \mathbb{O}^m)\,;\mathsf{f}(\mathbb{B}^{n,m})]\neg p_j)\Big).\end{aligned}$$

We quantify over all possible issues j with the first big conjunction. The two conjuncts inside the parentheses take into account the case where j is currently accepted or currently rejected in the outcome. Then, we have that in any way we modify the acceptances and rejections of the agents in the profile (without changing those for j) such that we still have a rational profile, the new outcome for j is the same as the one for the original profile.

Proposition 19. *Let $\mathcal{N} = \{1, \ldots, n\}$ and $\mathcal{I} = \{1, \ldots, m\}$ be the sets of agents and issues, respectively. Let $\mathbb{B}^{n,m}$ be the set of propositional variables encoding the opinions of agents in $\mathcal{N}$ on issues in $\mathcal{I}$. Let f be a* DL-PA *program translating some rule F. Then,*

$$\mathrm{I} \textit{ holds} \iff \models [\mathsf{profile}_{\mathrm{IC}}(\mathbb{B}^{n,m}, \mathbb{O}^m)\,;\mathsf{f}(\mathbb{B}^{n,m})]\mathsf{I}.$$

I-Monotonicity

A rule F is *independent-monotonic* if, whenever we consider two profiles such that the second one differs from the first in that some agent i first rejected some issue j and then she accepts it, if j was accepted in the first outcome then it should still be accepted in the second. Namely, additional support for an issue should not make it first accepted and then rejected.

More formally, if we let $(\boldsymbol{B}_{-i}, B'_i) = (B_1, \ldots, B'_i, \ldots, B_n)$ for some profile $\boldsymbol{B}$, we have that for rule F the following holds:

M^{I} : For any issue $j \in \mathcal{I}$, agent $i \in \mathcal{N}$, profiles $\boldsymbol{B} = (B_1, \ldots, B_n)$ and $\boldsymbol{B}' = (\boldsymbol{B}_{-i}, B_i)$, if $b_{ij} = 0$ and $b'_{ij} = 1$ then $F(\boldsymbol{B})_j = 1$ implies $F(\boldsymbol{B}')_j = 1$.

We can now express I-Monotonicity in DL-PA as follows:

$$\mathsf{M}^{\mathsf{I}} := \bigwedge_{j\in\mathcal{I}} \Big(p_j \to \bigwedge_{i\in\mathcal{N}} [+p_{ij}\,;\mathsf{prof}_{\mathrm{IC}}(\mathbb{B}^{n,m}, \mathbb{O}^m)\,;\mathsf{f}(\mathbb{B}^{n,m})]p_j\Big).$$

The first conjunction quantifies over all possible choices of issues. If it is the case that issue j is currently accepted in the outcome, then if any agent now accepts j in her individual ballot (so that the profile is still rational) the outcome of the program encoding F should still accept j.

Proposition 20. *Let $\mathcal{N} = \{1, \ldots, n\}$ and $\mathcal{I} = \{1, \ldots, m\}$ be the sets of agents and issues, respectively. Let $\mathbb{B}^{n,m}$ be the set of propositional variables encoding the opinions of agents in $\mathcal{N}$ on issues in $\mathcal{I}$. Let f be a* DL-PA *program translating some rule F. Then,*

$$\mathrm{M}^{\mathrm{I}} \textit{ holds} \iff \models [\mathsf{prof}_{\mathrm{IC}}(\mathbb{B}^{n,m}, \mathbb{O}^m)\,;\mathsf{f}(\mathbb{B}^{n,m})]\mathsf{M}^{\mathsf{I}}.$$

Proof. For the left-to-right direction, consider an arbitrary v_1 and v_2 such that $(v_1, v_2) \in \| \mathsf{prof}_{\mathrm{IC}}(\mathbb{B}^{n,m}, \mathbb{O}^m) \,;\, \mathsf{f}(\mathbb{B}^{n,m}) \|$. We want to show that $v_2 \in \| \mathsf{M}^{\mathsf{I}} \|$, that is:

$$v_2 \in \| \bigwedge_{j \in \mathcal{I}} \Big(p_j \to \bigwedge_{i \in \mathcal{N}} [+p_{ij} \,;\, \mathsf{prof}_{\mathrm{IC}}(\mathbb{B}^{n,m}, \mathbb{O}^m) \,;\, \mathsf{f}(\mathbb{B}^{n,m})] p_j \Big) \| \,.$$

Suppose, for reductio, that this is not the case. Then, there is some issue $k \in \mathcal{I}$ for which (a) $v_2 \in \| \; p_k \; \|$, and (b) $v_2 \notin \| \; \bigwedge_{i \in \mathcal{N}} [+p_{ik} \,;\, \mathsf{prof}_{\mathrm{IC}}(\mathbb{B}^{n,m}, \mathbb{O}^m) \,;\, \mathsf{f}(\mathbb{B}^{n,m})] p_k \; \|$. The latter implies that there is some agent $s \in \mathcal{N}$ such that (c) there is v_3 where $(v_2, v_3) \in \| \; +p_{sk} \,;\, \mathsf{prof}_{\mathrm{IC}}(\mathbb{B}^{n,m}, \mathbb{O}^m) \,;\, \mathsf{f}(\mathbb{B}^{n,m}) \; \|$, and (d) $v_3 \notin \| \; p_k \; \|$. Since $(v_2, v_3) \in \| \; +p_{sk} \,;\, \mathsf{prof}_{\mathrm{IC}}(\mathbb{B}^{n,m}, \mathbb{O}^m); \mathsf{f}(\mathbb{B}^{n,m}) \|$, it means that there is valuation v_a such that $(v_2, v_a) \in \| +p_{sk} \|$ and $(v_a, v_3) \in \| \, \mathsf{prof}_{\mathrm{IC}}(\mathbb{B}^{n,m}, \mathbb{O}^m) \,;\, \mathsf{f}(\mathbb{B}^{n,m}) \|$. Hence, v_a corresponds to a rational profile $\boldsymbol{B}^a$ and v_3 corresponds to $F(\boldsymbol{B}^a)$.

Consider now valuations v_1, v_2 and v_a. While v_1 and v_2 by assumption do not differ on the propositional variables in $\mathbb{B}^{n,m}$, v_2 and v_a possibly differ on p_{sk}. We now focus on the interesting case: namely, the one where they do differ.

If they differ, we have that $p_{sk} \notin v_1$ and $p_{sk} \notin v_2$ but $p_{sk} \in v_a$. Notice that since $(v_1, v_2) \in \| \, \mathsf{prof}_{\mathrm{IC}}(\mathbb{B}^{n,m}, \mathbb{O}^m) \,;\, \mathsf{f}(\mathbb{B}^{n,m}) \|$, this means that v_1 corresponds to a rational profile $\boldsymbol{B}^1$ and v_2 corresponds to $F(\boldsymbol{B}^1)$. Since $b^1_{sk} = 0$ and by (b) we have $F(\boldsymbol{B}^1)_j = 1$, and since I-Monotonicity holds by assumption, and since $\boldsymbol{B}^a$ differs from $\boldsymbol{B}^1$ only on b_{sk}, because $b^a_{sk} = 1$, we must have that $F(\boldsymbol{B}^a)_k = 1$. But this contradicts the fact that by (d) we have $p_k \notin v_3$.

For the right-to-left direction, consider an arbitrary rational profile $\boldsymbol{B}$. We want to show that for any issue $j \in \mathcal{I}$, agent $i \in \mathcal{N}$ and profile $\boldsymbol{B}' = (\boldsymbol{B}_{-i}, B_i)$, if $b_{ij} = 0$ and $b'_{ij} = 1$ then $F(\boldsymbol{B})_j = 1$ implies $F(\boldsymbol{B}')_j = 1$. Suppose, for reductio, that this is not the case: hence, there is some issue $j \in \mathcal{I}$, some agent $i \in \mathcal{N}$ and profile $\boldsymbol{B}'$, such that we have $b_{ij} = 0$, $b'_{ij} = 1$, $F(\boldsymbol{B})_j = 0$ and $F(\boldsymbol{B}')_j = 0$.

Consider now valuation v_1 corresponding to profile $\boldsymbol{B}$ and valuation v_2 corresponding to $F(\boldsymbol{B})$. Moreover, consider valuation v_a corresponding to profile $\boldsymbol{B}'$ and valuation v_b corresponding to $F(\boldsymbol{B}')$. Since by assumption M^{I} holds, and $p_j \in v_2$, and v_a only differs from v_1 in that $p_{ij} \in v_a$, we see that from the axiom we should get $p_j \in v_b$. This contradicts the fact that v_b is a translation of $F(\boldsymbol{B}')$ and that $F(\boldsymbol{B}')_j = 0$. □

Anonymity

An *anonymous* rule F treats each agent in the same way. That is, when we permute the order of the individual ballots in the input, the output for all the issues does not change.

For rule F this formally translates as follows:

A : For any profile $\mathbf{B}$ and any permutation $\sigma : \mathcal{N} \to \mathcal{N}$, we have that
$$F(B_1, \ldots, B_n) = F(B_{\sigma(1)}, \ldots, B_{\sigma(n)}).$$

A first attempt to translate Anonymity is the following DL-PA formula:

$$\mathsf{A} := \bigwedge_{\substack{(k_1,\ldots,k_n)\\ k_i \in \mathcal{N}}} [\mathsf{store}(\mathbb{O}^m) \,;\, \mathop{;}_{i\in\mathcal{N}} (\mathop{;}_{j\in\mathcal{I}} (p'_{ij} \leftarrow p_{k_i j})) \,;\, \mathsf{init}(\mathbb{O}^m) \,;\, \mathsf{f}(\mathbb{B}^{n,m})] \bigwedge_{j\in\mathcal{I}} (p_j \leftrightarrow p'_j).$$

We start by taking into account any possible permutation of the agents. Then, whenever we store the result of the rule, shuffle the acceptances and rejections of the agents on the issues according to the permutation we are considering, and compare the new outcome of the rule with the stored one, the two should coincide. Notice that here we do not check whether the new profile is rational because we are not modifying the individual ballots but just their order.

Notice, however, that it is possible to generate all the $n!$ permutations of n agents by repeatedly swapping the positions of just two agents [Wells, 1961] with at most $n-1$ swaps. Therefore, we can reformulate the previous axiom in the following way:

$$\begin{aligned}\mathsf{A}^* := [\mathsf{store}(\mathbb{O}^m) \,;\, \Big(\bigcup_{i,k\in\mathcal{N}} \mathop{;}_{j\in\mathcal{I}} (\mathsf{if}\ p_{ij} \oplus p_{kj}\ \mathsf{do}\ \mathsf{flip}^{\mathsf{all}}(\{p_{ij}, p_{kj}\}))\Big)^{n-1} \,;\\ \mathsf{init}(\mathbb{O}^m) \,;\, \mathsf{f}(\mathbb{B}^{n,m})] \bigwedge_{j\in\mathcal{I}} (p_j \leftrightarrow p'_j).\end{aligned}$$

We first store the result in some fresh variables. Then, we consider for $n-1$ times some (possibly different) pair of agents and we swap their individual ballots issue by issue. Afterwards, we apply the program for the rule on the new input and the new result should be equal to the previous one on every issue.

Proposition 21. *Let $\mathcal{N} = \{1, \ldots, n\}$ and $\mathcal{I} = \{1, \ldots, m\}$ be the sets of agents and issues, respectively. Let $\mathbb{B}^{n,m}$ be the set of propositional variables encoding the opinions of agents in $\mathcal{N}$ on issues in $\mathcal{I}$. Let f be a* DL-PA *program translating some rule F. Then,*

$$\text{A } \textit{holds} \iff \models [\mathsf{profile}_{\mathrm{IC}}(\mathbb{B}^{n,m}, \mathbb{O}^m) \,;\, \mathsf{f}(\mathbb{B}^{n,m})]\mathsf{A}^{(*)}.$$

3.4 Summary

In this chapter we delved into the study of *axioms*, i.e. arguably desirable properties that we would want aggregation rules to satisfy. Checking whether a rule satisfies an axiom corresponds to checking the validity of a general DL-PA formula provided at the beginning

of the Chapter. We distinguished between *single-profile* and *multi-profile* axioms and we translated them as DL-PA formulas.

For single-profile axioms, we observed that a translation in propositional logic was sufficient. In fact, since we have propositional variables to encode a profile and propositional variables to encode the outcome of a rule on a profile, it is possible to compare the status of the propositional variables of the profile and that of the propositional variables of the outcome within a unique valuation.

On the other hand, the multi-profile axioms have a more complex nature that it is not possible to easily capture within propositional logic. In fact, since they refer to any choice of two profiles we have to find a way to consider two valuations at the same time. The language of DL-PA made this possible, and the axioms were thus translated as modal formulas.

For both types of axioms we provided proofs of correctness of our translation. Each section thus ended with a proposition linking the fact that aggregation rules satisfy an axiom in BA with the fact that a certain formula is valid in DL-PA. By checking that these formulas are valid for specific instances of aggregation rules (i.e., when the number of agents and issues are given) and certain axioms, we could gain some insight towards characterisation results such as May's Theorem [May, 1952].

The theorem, reformulated in BA, states that for an odd number of agents and one issue, an aggregation rule satisfies $\mathrm{N}^{\mathcal{D}}$, M^{I} and A if and only if it is the majority rule. The right-to-left direction of the theorem amounts to saying that for any odd n the formulas $[\mathsf{profile}_{\mathrm{IC}}(\mathbb{B}^{n,1}, \mathbb{O}^{1})\,;\mathsf{maj}(\mathbb{B}^{n,1})]\mathsf{A}^{(*)} \wedge \mathsf{M}^{\mathsf{I}} \wedge \mathsf{N}^{\mathcal{D}}$ are valid. On the other hand, the left-to-right direction cannot be expressed in the logic, because we would have to consider a generic program for an aggregation rule to write inside the axioms, which would then turn to be equivalent to the one for majority.

With our encoding we can verify whether many aggregators introduced in Chapter 2 satsify the axioms presented here or not. For instance, we have that while $[\mathsf{profile}_{\mathrm{IC}}(\mathbb{B}^{n,m}, \mathbb{O}^{m})\,;\mathsf{dictatorship}_{i}(\mathbb{B}^{n,m})]\mathsf{U} \wedge \mathsf{N}^{\mathcal{I}} \wedge \mathsf{N}^{\mathcal{D}}$ is valid for any number of agents and issues, formulas such as $[\mathsf{profile}_{\mathrm{IC}}(\mathbb{B}^{n,m}, \mathbb{O}^{m})\,;\mathsf{parity}_{\mathsf{even}}(\mathbb{B}^{n,m})]\mathsf{U} \wedge \mathsf{M}^{\mathsf{I}}$ are not. Notice also that while the properties we explained are clearly understood for resolute aggregation rules, there are different ways to capture them for irresolute aggregators [Lang et al., 2011] — which then require an easy DL-PA adaptation.

After having inspected properties of aggregation rules both closely and from afar, we will now focus on integrity constraints. In the next chapter we will study how properties of integrity constraints are linked to classes of rules satisfying certain axioms in such a way that a consistent outcome is always guaranteed.

Chapter 4

Agenda Safety in DL-PA

Integrity constraints make explicit the relationship existing among the issues, if any. A research direction in JA investigates whether depending on how the issues relate to one another, we can ensure that the outcome of certain classes of aggregation rules always satisfies a given IC. This problem is known in formula-based JA as the *safety of the agenda* problem [Endriss et al., 2012; Grandi, 2012], and in order to adapt it to our discussion we first have to explain the problem in its original framework.

The Chapter is thus organised as follows. In Section 4.1 we introduce the problem of the safety of the agenda in formula-based JA. In Section 4.2 we adapt some known agenda properties of formula-based JA to DL-PA. Then, Section 4.3 shows how to model in DL-PA theorems linking the aforementioned properties to classes of aggregation rules. We conclude in Section 4.4 with a summary of the Chapter.

4.1 Agenda Safety in JA

In formula-based JA we have a set $\Phi = \{\varphi_1, \neg\varphi_1, \ldots, \varphi_m, \neg\varphi_m\}$ named *agenda*, which consists of m non-negated formulas plus their negations, and we also have a set $\mathcal{N} = \{1, \ldots, n\}$ of *agents*. Any agent $i \in \mathcal{N}$ expresses her opinion on the agenda by including in her *individual judgment set* J_i either φ_j or $\neg\varphi_j$ for all $j \in \{1, \ldots, m\}$ so that J_i is consistent. Thus, individual judgment sets have to be both *complete*, i.e. no formula is left undecided, and *complement-free*, i.e. they cannot include both a formula and its negation. Analogously to BA, where the input for an aggregation procedure F is defined over a specific integrity constraint, in formula-based JA the input for an aggregation rule F is defined over an agenda Φ.

Example 8. *Let the agents be* $\mathcal{N} = \{1, 2, 3\}$ *and consider the agenda* $\Phi = \{a, b, a \to b, \neg a, \neg b, \neg(a \to b)\}$. *Now, look at the following three individual judgment sets:*

$$\begin{aligned} J_1 &= \{a, \neg b, \neg(a \to b)\} \\ J_2 &= \{a, \neg a, b, a \to b\} \\ J_3 &= \{b, a \to b\} \end{aligned}$$

The first set J_1 *is the only admissible one. In fact, notice that* J_2 *fails complement-freeness by including both* a *and* $\neg a$*, while* J_3 *fails completeness by not including either* a *or* $\neg a$.

We can now state the problem of the safety of the agenda. Let a *class* of aggregation procedures $\mathcal{F}_\Phi[\mathrm{AX}]$ be the set of rules defined over the agenda Φ satisfying all the axioms listed in the set AX. Given some agenda Φ, we want to know for which class of JA procedures it is the case that every rule in the class returns a consistent outcome when applied to judgment sets over Φ. If this is the case, we say that agenda Φ is *safe* for that class of rules.

We anticipated before that it is possible to translate any instance of a problem formulated in formula-based JA into an instance of a BA problem [Grandi and Endriss, 2011]. This is usually done by modelling each formula in the agenda Φ as a different issue, so that $|\Phi| = |\mathcal{I}|$. The integrity constraint consists then of two parts: the first part is a conjunction of $p_\alpha \vee p_{\neg\alpha}$ for every α in Φ (the *completeness* requirement). For the second part, we need the concept of *minimally inconsistent set* (mi-set): an inconsistent set whose every strict subset is consistent. Thus, we have a conjunction of $\neg(\bigwedge_{\alpha\in S} p_\alpha)$ for every mi-set $S \subseteq \Phi$ (the *consistency* requirement).

Example 9. *For agenda* $\Phi = \{a, b, a \wedge b, \neg a, \neg b, \neg(a \wedge b)\}$*, the corresponding set of issues* $\mathcal{I} = \{a, b, a \wedge b, \neg a, \neg b, \neg(a \wedge b)\}$ *generates the following set of propositional variables* $PS = \{p_a, p_b, p_{a\wedge b}, p_{\neg a}, p_{\neg b}, p_{\neg(a\wedge b)}\}$. *The integrity constraint is the following:*

$$\begin{aligned} \mathrm{IC} = {} & (p_a \vee p_{\neg a}) \wedge (p_b \vee p_{\neg b}) \wedge (p_{a\wedge b} \vee p_{\neg(a\wedge b)}) \wedge \\ & \neg(p_a \wedge p_{\neg a}) \wedge \neg(p_b \wedge p_{\neg b}) \wedge \neg(p_{a\wedge b} \wedge p_{\neg(a\wedge b)}) \wedge \\ & \neg(p_a \wedge p_b \wedge p_{\neg(a\wedge b)}) \wedge \neg(p_{\neg a} \wedge p_{a\wedge b}) \wedge \neg(p_{\neg b} \wedge p_{a\wedge b}). \end{aligned}$$

Notice that the first line encodes completeness, the second encodes complement-freeness and the third encodes all the other consistency requirements given by the mi-sets.

The safety of the agenda problem has been studied in BA when the translation is carried out as explained above, so that the integrity constraint ends up having this form.

We can thus have a syntactic characterisation of constraints with respect to agenda properties such that the outcomes of certain classes of aggregation rules are consistent with IC [Grandi, 2012].

Alternatively, we could consider aggregation problems already expressed in the BA framework, or translate a formula-based JA problem so that each pair of complementary formulas corresponds to a single issue (and if a mi-set contains a negated formula $\neg\alpha$, the corresponding literal in IC will be $\neg p_\alpha$). The completeness requirement would then be given by default, since agents express either a 0 or a 1 for each issue in their individual ballots. Moreover, the inconsistency generated by agents accepting both the positive and the negative version of some formula would be ruled out, since they cannot express both a 0 and a 1 for the same issue in their ballots.

Example 9 (Continued)**.** *Consider again agenda* $\Phi = \{a, b, a \wedge b, \neg a, \neg b, \neg(a \wedge b)\}$*, but now take as a set of corresponding issues* $\mathcal{I} = \{a, b, a \wedge b\}$*, with* $PS = \{p_a, p_b, p_{a\wedge b}\}$*. The integrity constraint is now the following:*

$$\begin{aligned} \text{IC} &= \neg(p_a \wedge p_b \wedge \neg p_{(a\wedge b)}) \wedge \neg(\neg p_a \wedge p_{a\wedge b}) \wedge \neg(\neg p_b \wedge p_{a\wedge b}) \\ &= p_{a\wedge b} \leftrightarrow p_a \wedge p_b. \end{aligned}$$

Notice that we called the issues "a", "b" and "a $\wedge$ *b" just to make clear the correspondence with agenda* Φ*: it makes no difference to consider instead a set of issues* $\{1, 2, 3\}$ *and an* $\text{IC} = p_3 \leftrightarrow p_1 \wedge p_2$*.*

In the remainder of the Chapter, we will always consider this second way of translating (or expressing) integrity constraints.

4.2 Properties of IC

In this section we provide some definitions [Marchi et al., 2010] that will be used to express constraints with a different perspective. Let a *literal* be either a variable p or its negation $\neg p$. A *term* D is a conjunction of distinct literals. Let $D - D'$ be the *subtraction operation* over terms, resulting in all the literals of D that are not present in D'. A term D is an *implicant* of formula φ if and only if $D \models \varphi$. In particular, D is a *prime implicant* of φ if and only if

(i) D is an implicant of φ;

(ii) for all literals L in D, $(D - \{L\}) \not\models \varphi$.

The concept of minimally inconsistent set of an agenda Φ directly relates to that of a prime implicant of $\neg$IC, where IC is the constraint obtained from Φ via the second method of translation provided. Then, IC can equivalently be seen as a conjunction of negations of prime implicants of $\neg$IC [Marquis, 2000].

Now we can introduce a DL-PA formula which is true if and only if the valuation we are considering makes *exactly one* of the prime implicants of $\neg$IC true. Given an integrity constraint IC, we let $\mathbb{P}_{\text{IC}}$ to be the set of propositional variables used in IC:

$$\mathsf{MI}_{\text{IC}} := \bigvee_{D \in \mathcal{P}(\mathbb{P}_{\text{IC}})} \Big([\mathsf{flip}^1(D)]\langle \mathsf{flip}^{\geq 0}(\mathbb{P}_{\text{IC}} \setminus D)\rangle \text{IC} \wedge [\mathsf{flip}^{\geq 0}(\mathbb{P}_{\text{IC}} \setminus D)]\neg \text{IC} \wedge \bigwedge_{D' \in (\mathcal{P}(\mathbb{P}_{\text{IC}}) \setminus D)} (\langle \mathsf{flip}^1(D')\rangle [\mathsf{flip}^{\geq 0}(\mathbb{P}_{\text{IC}} \setminus D')]\neg \text{IC} \vee \langle \mathsf{flip}^{\geq 0}(\mathbb{P}_{\text{IC}} \setminus D')\rangle \text{IC}) \Big).$$

The formula states that there is some subset D of the propositional variables in IC such that: by flipping the truth value of just one of them we can then possibly reach a state where IC holds, there is no way of flipping the truth value of the propositional variables not in D such that IC is true and no other subset has these characteristics. Notice that if the formula holds, there is some D such that $[\mathsf{flip}^{\geq 0}(\mathbb{P}_{\text{IC}} \setminus D)]\neg$IC, which includes the case where the truth value of no propositional variable gets flipped: hence, $\neg$IC has to hold as well.

Proposition 22. *Let* IC *be an integrity constraint. Then,* MI_{IC} *is true in a valuation* v *if and only if there is a unique prime implicant* D *of* $\neg$IC *true in* v*.*

Proof. We prove the right-to-left direction directly. Assume that v is valuation such that exactly one prime implicant D of $\neg$IC is true. In particular, this means that all the literals L in D are true in v. If L is of the form p_i we have that $p_i \in v$, while if L is of the form $\neg p_k$ we have that $p_k \notin v$.

Consider now the formula MI_{IC} and let D_1 be the set of propositional variables in the literals of the prime implicant D. First, we show that $[\mathsf{flip}^1(D_1)]\langle \mathsf{flip}^{\geq 0}(\mathbb{P}_{\text{IC}} \setminus D_1)\rangle$IC is true in v: namely, that for all valuations v' such that $(v, v') \in \| \mathsf{flip}^1(D_1) \|$ there is some valuation v'' such that $(v', v'') \in \| \mathsf{flip}^{\geq 0}(\mathbb{P}_{\text{IC}} \setminus D_1) \|$ where $v'' \models$ IC.

Recall that program $\mathsf{flip}^1(D_1)$ flips the truth value of one propositional variable in the set D_1. Thus, in any valuation v' exactly one propositional variable $p_i \in D_1$ has opposite truth value with respect to v. Hence, the corresponding literal L_i in D does not hold anymore. Therefore, by condition (ii) of D being a prime implicant of $\neg$IC, we have that it is possible to get to a valuation where IC holds: which corresponds to the fact that $\langle \mathsf{flip}^{\geq 0}(\mathbb{P}_{\text{IC}} \setminus D_1)\rangle$IC is the case.

We also have to show that $[\mathsf{flip}^{\geq 0}(\mathbb{P}_{\mathrm{IC}} \setminus D_1)]\neg\mathrm{IC}$ holds: namely, that for all v' such that $(v, v') \in \| \mathsf{flip}^{\geq 0}(\mathbb{P}_{\mathrm{IC}} \setminus D_1) \|$ we have that $\neg\mathrm{IC}$ is true. Notice that this follows immediately from the condition (i) of D being a prime implicant of $\neg\mathrm{IC}$. As far as $\bigwedge_{D' \in (\mathcal{P}(\mathbb{P}_{\mathrm{IC}}) \setminus D_1)}(\langle \mathsf{flip}^1(D') \rangle [\mathsf{flip}^{\geq 0}(\mathbb{P}_{\mathrm{IC}} \setminus D')]\neg\mathrm{IC} \vee \langle \mathsf{flip}^{\geq 0}(\mathbb{P}_{\mathrm{IC}} \setminus D') \rangle \mathrm{IC}$ is concerned, it is easily seen that its truth follows from the uniqueness condition of the prime implicant D. Therefore, we can conclude that $\mathsf{MI}_{\mathrm{IC}}$ is true in v.

We prove the left-to-right direction by contraposition. First we discuss the case for a valuation v which makes true more than one prime implicant of $\neg\mathrm{IC}$, and then the case where no prime implicant of $\neg\mathrm{IC}$ is true in v. Assume that there is some other prime implicant D' of $\neg\mathrm{IC}$ true in v: we want to show that $\mathsf{MI}_{\mathrm{IC}}$ is false in v. Since by assumption v makes true both D and D' and they are prime implicants of $\neg\mathrm{IC}$, this means that $\neg\mathrm{IC}$ is true in v. In order to show that $\mathsf{MI}_{\mathrm{IC}}$ is false in v, we have to show that for all $S \in \mathcal{P}(\mathbb{P}_{\mathrm{IC}})$, either

(a) $\neg[\mathsf{flip}^1(S)]\langle \mathsf{flip}^{\geq 0}(\mathbb{P}_{\mathrm{IC}} \setminus S) \rangle \mathrm{IC}$, or

(b) $\langle \mathsf{flip}^{\geq 0}(\mathbb{P}_{\mathrm{IC}} \setminus S) \rangle \mathrm{IC}$, or

(c) $\bigvee_{Q \in (\mathcal{P}(\mathbb{P}_{\mathrm{IC}}) \setminus S)}([\mathsf{flip}^1(Q)]\langle \mathsf{flip}^{\geq 0}(\mathbb{P}_{\mathrm{IC}} \setminus D) \rangle \mathrm{IC} \wedge [\mathsf{flip}^{\geq 0}(\mathbb{P}_{\mathrm{IC}} \setminus Q)]\neg\mathrm{IC})$.

Consider an arbitrary $S \in \mathcal{P}(\mathbb{P}_{\mathrm{IC}})$, and let D_1 and D_2 be the sets containing the propositional variables of the literals used in D and D' respectively. Notice that by condition (ii) of the definition of prime implicant, we get that neither $D_1 \subseteq D_2$ nor $D_2 \subseteq D_1$ can be the case. In fact, otherwise in both cases the superset among the two would fail condition (ii) and it would not be a prime implicant of $\neg\mathrm{IC}$. Therefore, they must each have some propositional variable which is not included also in the other set.

The arbitrary set S could be equal to D_1, to D_2 or to neither of them. In all cases, we are going to show that (c) holds: in the first case we show that $Q = D_2$, in the second case that $Q = D_1$ and in the last case that either $Q = D_1$ or $Q = D_2$. Since the three cases are similar, we are going to discuss without loss of generality just if $S = D_1$.

Consider $Q = D_2$, which by assumption is the set of propositional variables of the literals used in D'. From the fact that D' is a prime implicant of $\neg\mathrm{IC}$, by condition (i) we have that if the current assignment makes the literals in D' true, $\neg\mathrm{IC}$ holds regardless of the truth values assigned to the other propositional variables. Hence, we have that $[\mathsf{flip}^{\geq 0}(\mathbb{P}_{\mathrm{IC}} \setminus D_2)]\neg\mathrm{IC}$ is the case.

Moreover, by condition (ii) of being a prime implicant of $\neg\mathrm{IC}$ we know that for any literal L' of D' we have $(D' - \{L'\}) \not\models \neg\mathrm{IC}$. This means that in case one of the literals of D' turns false, there is some assignment to the other propositional variables that makes

IC true. Notice that since the propositional variables in D_2 correspond to the literals in D', this is just as saying that $[\mathsf{flip}^1(D_2)]\langle\mathsf{flip}^{\geq 0}(\mathbb{P}_{\mathrm{IC}} \setminus D_2)\rangle\mathrm{IC}$ is the case. Therefore, we can conclude that $\mathsf{MI}_{\mathrm{IC}}$ is false in v since we showed that (c) holds.

On the other hand, assume that v is a valuation such that no prime implicant of $\neg\mathrm{IC}$ is true in v. Since IC is equivalent to a conjunction of negations of prime implicants of $\neg\mathrm{IC}$ [Marquis, 2000], we have that IC is true. In order to falsify $\mathsf{MI}_{\mathrm{IC}}$ we show that for all $S \in \mathcal{P}(\mathbb{P}_{\mathrm{IC}})$ point (b) holds. In fact, since IC holds in v, it sufficies to consider the execution of program $\mathsf{flip}^{\geq 0}(\mathbb{P}_{\mathrm{IC}} \setminus S)$ where no propositional variable gets its truth value flipped. Hence, $\langle\mathsf{flip}^{\geq 0}(\mathbb{P}_{\mathrm{IC}} \setminus S)\rangle\mathrm{IC}$ holds and we can conclude that $\mathsf{MI}_{\mathrm{IC}}$ is false in v. □

Example 10. *Consider the following two assignments v and v' for issues $\mathcal{I} = \{1, 2, 3, 4\}$ and constraint* $\mathrm{IC} = p_2 \to p_1 \wedge p_4 \to p_3$*:*

	p_1	p_2	p_3	p_4
v	0	1	0	1
v'	1	0	0	1

In both cases $\neg\mathrm{IC}$ holds and both $D = \neg p_3 \wedge p_4$ and $D' = \neg p_1 \wedge p_2$ are prime implicants of $\neg\mathrm{IC}$. However, while v makes both D and D' true, in valuation v' only the prime implicant D of $\neg\mathrm{IC}$ is true. Therefore, we have that $\mathsf{MI}_{\mathrm{IC}}$ is true in v' and is false in v.

Median Property

The Median Property (MP) in formula-based JA is stated as follows: All minimally inconsistent subsets of the agenda have size 2 [Endriss et al., 2012; Grandi, 2012]. In our framework, this means that in all valuations falsifying IC the unique true prime implicant of $\neg\mathrm{IC}$ has to have just two literals. We can thus express the MP in DL-PA in the following way:

$$\mathsf{MP}_{\mathrm{IC}} := \mathsf{MI}_{\mathrm{IC}} \to \bigvee_{p_i,p_k \in \mathbb{P}_{\mathrm{IC}}} \Big([\mathsf{flip}^1(\{p_i,p_k\})]\langle\mathsf{flip}^{\geq 0}(\mathbb{P}_{\mathrm{IC}} \setminus \{p_i,p_k\})\rangle\mathrm{IC} \wedge [\mathsf{flip}^{\geq 0}(\mathbb{P}_{\mathrm{IC}} \setminus \{p_i,p_k\})]\neg\mathrm{IC}\Big).$$

Suppose the current valuation falsifies the integrity constraint and only one prime implicant of $\neg\mathrm{IC}$ is true. Then, it must be possible to find two propositional variables among the ones in the constraint such that: if we flip the truth value of either of the two the constraint might hold, and if we flip the truth value of any *other* set of propositional variables, the constraint does not hold.

Proposition 23. *Let* Φ *be an agenda in formula-based* JA *and let* IC *be its translation in* BA*. Then,* Φ *satisfies the* MP *if and only if* $\models \mathsf{MP}_{\mathrm{IC}}$*.*

Proof. We start from the left-to-right direction. Consider an agenda Φ that satisfies MP. The definition of the median property states that every mi-set of Φ has size two. Recall that the encoding of Φ in a constraint IC results in a big conjunction where each conjunct is a negation of a term D, whose literals corresponds to an element of some mi-set. Hence, each conjunct of IC is of the form $\neg(L \wedge L')$, for some literals L and L'.

Consider now an arbitrary valuation v and assume that $\mathsf{MI}_{\mathrm{IC}}$ holds. We want to show that the consequent of $\mathsf{MP}_{\mathrm{IC}}$ holds as well. By Proposition 22 we know that there is a unique term D which is true in v, and from what we explained above we know that $D = L_i \wedge L_k$. Hence, consider the propositional variables p_i and p_k corresponding to the literals L_i and L_k.

By condition (ii) of being a prime implicant of $\neg$IC, we have that if we flip the truth value of either one among $\{p_i, p_k\}$ there is a way to make true the constraint IC. Moreover, since D was an implicant of $\neg$IC, it cannot happen that by leaving the truth values of p_i and p_k unchanged the constraint IC turns out true. Therefore, the conclusion of $\mathsf{MP}_{\mathrm{IC}}$ is true, which means that $v \models \mathsf{MP}_{\mathrm{IC}}$, and since v was arbitrary we can conclude that $\models \mathsf{MP}_{\mathrm{IC}}$.

We prove the right-to-left direction by contraposition. Assume $\not\models \mathsf{MP}_{\mathrm{IC}}$: thus, there is some valuation v such that $v \models \mathsf{MI}_{\mathrm{IC}}$ and the consequent of $\mathsf{MP}_{\mathrm{IC}}$ is false in v. Since $\mathsf{MI}_{\mathrm{IC}}$ is true in v, by Proposition 22 we know that there is a unique prime implicant D of $\neg$IC true in v.

Suppose that $D = L_i \wedge L_k$ where L_i and L_k are two literals: namely, suppose that D has size two. Notice that in this case, we could choose propositional variables p_i and p_k in $\mathbb{P}_{\mathrm{IC}}$ corresponding to L_i and L_k. For a similar argument to the one stated above, the consequent of $\mathsf{MP}_{\mathrm{IC}}$ would turn out to be true contradicting our assumption. Therefore, $D \neq L_i \wedge L_k$.

Since D is by assumption a prime implicant of $\neg$IC, and since IC encodes agenda Φ, we have that the literals $L_1, \ldots, L_p$ in D correspond to the elements of some mi-set $S \subseteq \Phi$. Moreover, since $D \neq L_i \wedge L_k$ we have that $|S| \neq 2$: hence, we found a mi-set of Φ whose size is not two. Therefore, Φ does not have the MP. □

k-Median Property

The k-Median Property (kMP) is usually expressed in formula-based JA in the following way: All minimally inconsistent subsets of the agenda have size at most k. Notice, in

particular, that we get precisely the median property for $k = 2$. In our framework, then, in all the valuations falsifying IC the unique prime implicant of ¬IC which is true has to be of size at most k. We can express the kMP in DL-PA as follows:

$$k\mathsf{MP}_{\mathrm{IC}} := \mathsf{MI}_{\mathrm{IC}} \to \bigvee_{\substack{p_1,\ldots,p_j \in \mathbb{P}_{\mathrm{IC}} \\ 2 \leq j \leq k}} \Big([\mathsf{flip}^1(\{p_1,\ldots,p_j\})] \langle \mathsf{flip}^{\geq 0}(\mathbb{P}_{\mathrm{IC}} \setminus \{p_1,\ldots,p_j\}) \rangle \mathrm{IC} \wedge [\mathsf{flip}^{\geq 0}(\mathbb{P}_{\mathrm{IC}} \setminus \{p_1,\ldots,p_j\})] \neg \mathrm{IC} \Big).$$

This formula has the same structure as $\mathsf{MP}_{\mathrm{IC}}$, used to describe MP. The main difference is that here we are considering any set of at most k issues and not just sets of size 2.

Proposition 24. *Let* Φ *be an agenda in formula-based* JA *and let* IC *be its translation in* BA*. Then,* Φ *satisfies the* kMP *if and only if* $\models k\mathsf{MP}_{\mathrm{IC}}$.

Simplified Median Property

We can express the Simplified Median Property (SMP) in formula-based JA as follows: All minimally inconsistent subsets of the agenda are of the form $\{\varphi, \psi\}$ with $\models \varphi \leftrightarrow \neg\psi$. This means, in particular, that every minimally inconsistent set has size 2. Therefore, if an agenda has the SMP it also has the MP. In our framework, if a valuation falsifies IC, then not only the unique true prime implicant of ¬IC has to have size two, but the two issues must behave as if they were each other's negation. The SMP can be expressed in DL-PA in the following way:

$$\mathsf{SMP}_{\mathrm{IC}} := \mathsf{MI}_{\mathrm{IC}} \to \bigvee_{p_i,p_k \in \mathbb{P}_{\mathrm{IC}}} \Big([\mathsf{flip}^1(\{p_i,p_k\})] \langle \mathsf{flip}^{\geq 0}(\mathbb{P}_{\mathrm{IC}} \setminus \{p_i,p_k\}) \rangle \mathrm{IC} \wedge [\mathsf{flip}^{\geq 0}(\mathbb{P}_{\mathrm{IC}} \setminus \{p_i,p_k\})] \neg \mathrm{IC} \wedge p_i \leftrightarrow \neg p_k \Big).$$

Again, the structure of this formula is similar to the one used to describe the MP. The only difference is that we added a third conjunct to impose the constraint on the issues p_i and p_k that have to be mutually exclusive.

Proposition 25. *Let* Φ *be an agenda in formula-based* JA *and let* IC *be its translation in* BA*. Then,* Φ *satisfies the* SMP *if and only if* $\models \mathsf{SMP}_{\mathrm{IC}}$.

Syntactic Simplified Median Property

The Syntactic Simplified Median Property (SSMP) in formula-based JA is stated as follows: All minimally inconsistent subsets of the agenda are of the form $\{\varphi, \neg\varphi\}$.

Therefore, the only possible inconsistency is the one resulting from accepting at the same time the positive and the negative version of some formula. Hence, all the issues are independent from each other. In DL-PA, thus, we will translate this property of the constraint as follows:

$$\mathsf{SSMP}_{\mathrm{IC}} \;:=\; \mathrm{IC} \leftrightarrow \top$$

Namely, the constraint is equivalent to $\top$ because there is no relationship whatsoever among the issues. The agents can then submit any individual ballot without violating the constraint.

Proposition 26. *Let* Φ *be an agenda in formula-based* JA *and let* IC *be its translation in* BA*. Then,* Φ *satisfies the* SSMP *if and only if* $\models \mathsf{SSMP}_{\mathrm{IC}}$*.*

4.3 Safety Results in DL-PA

We are finally ready to model in DL-PA the problem of relating properties of the agenda Φ with classes of rules $\mathcal{F}_\Phi[\mathrm{AX}]$ satisfying certain axioms. This result makes it possible to know whether every rule in this set always returns an outcome which satisfies IC.

The following results have been proven for formula-based JA and in order to understand their statements, we need to introduce two new axioms. The *Weak Rationality* axiom (WR) is satisfied whenever a rule F is always complete and complement-free: namely, the judgment set for the outcome contains either the non-negated or the negated version of each formula in the agenda. The *Systematicity* axiom (S) holds whenever a rule F satisfies both issue-neutrality and independence.

Proposition 27 ([Dietrich and List, 2007; Grandi, 2012])**.**

(i) *If the number of individuals is odd, then agenda* Φ *is safe for the majority rule if and only if* Φ *has the* MP*.*

(ii) *Agenda* Φ *is safe for* $\mathcal{F}_\Phi[\mathrm{WR}, \mathrm{A}, \mathrm{S}, \mathrm{U}]$ *if and only if* Φ *has the* SMP*.*

(iii) *Agenda* Φ *is safe for* $\mathcal{F}_\Phi[\mathrm{WR}, \mathrm{A}, \mathrm{N}^{\mathcal{I}}, \mathrm{U}]$ *if and only if* Φ *has the* SMP*.*

(iv) *Agenda* Φ *is safe for* $\mathcal{F}_\Phi[\mathrm{WR}, \mathrm{A}, \mathrm{I}, \mathrm{U}]$ *if and only if* Φ *has the* SSMP*.*

(v) *Agenda* Φ *is safe for uniform quota rules such that* $q > n - \frac{n}{k}$ *if and only if* Φ *has the* kMP*.*

Notice that in case we consider problems formulated in BA, or translations from formula-based JA where each pair of complementary formulas corresponds to a single issue, the axiom of Weak Rationality is satisfied by design. Moreover, recall that one of our initial assumptions was that the number of agents $n = |\mathcal{N}|$ is always odd.

Therefore, we can reformulate the five results above in DL-PA as valid formulas that will hold for any n, m, k and for some program f encoding an aggregation rule F.

1. $\models [\mathsf{prof}_{\mathrm{IC}}(\mathbb{B}^{n,m}, \mathbb{O}^m)\,;\mathsf{maj}(\mathbb{B}^{n,m})]\mathrm{IC} \leftrightarrow \mathsf{MP}_{\mathrm{IC}}$.
2. $\models \Big(([\mathsf{prof}_{\mathrm{IC}}(\mathbb{B}^{n,m}, \mathbb{O}^m)\,;\mathsf{f}(\mathbb{B}^{n,m})](\mathsf{A} \wedge \mathsf{N}^{\mathcal{I}} \wedge \mathsf{I} \wedge \mathsf{U})) \rightarrow \mathrm{IC}\Big) \leftrightarrow \mathsf{SMP}_{\mathrm{IC}}$.
3. $\models \Big(([\mathsf{prof}_{\mathrm{IC}}(\mathbb{B}^{n,m}, \mathbb{O}^m)\,;\mathsf{f}(\mathbb{B}^{n,m})](\mathsf{A} \wedge \mathsf{N}^{\mathcal{I}} \wedge \mathsf{U})) \rightarrow \mathrm{IC}\Big) \leftrightarrow \mathsf{SMP}_{\mathrm{IC}}$.
4. $\models \Big(([\mathsf{prof}_{\mathrm{IC}}(\mathbb{B}^{n,m}, \mathbb{O}^m)\,;\mathsf{f}(\mathbb{B}^{n,m})](\mathsf{A} \wedge \mathsf{I} \wedge \mathsf{U})) \rightarrow \mathrm{IC}\Big) \leftrightarrow \mathsf{SSMP}_{\mathrm{IC}}$.
5. $\models \Big([\mathsf{prof}_{\mathrm{IC}}(\mathbb{B}^{n,m}, \mathbb{O}^m)\,;\mathsf{quota}_{q_1,\ldots,q_m}(\mathbb{B}^{n,m})\,;\mathsf{zero}(\mathsf{x})\,;\mathsf{incr}(\mathsf{x})^{n-\frac{n}{k}}\,;$
   $\bigwedge_{1\leq i<m}\mathsf{quota}_i{=}\mathsf{quota}_{i+1}?\,;\bigwedge_{1\leq i\leq m}\mathsf{quota}_i > \mathsf{x}?]\mathrm{IC}\Big) \leftrightarrow k\mathsf{MP}_{\mathrm{IC}}$.

We now explain the structure of these formulas, since it is the same for all of them. On the right handside of the biconditional, we have the property that the constraint must satisfy. In case the constraint does satisfy that property, and the rule we are considering satisfies the axioms listed on the left handside of the biconditional, then IC will hold after the execution of this rule. In fact, recall that IC is formulated using the propositional variables for the outcome.

The interest for these kind of results comes from the observation that often the outcome of a rule is not consistent with the constraint IC. As we have seen in Chapter 2, many aggregation procedures try to circumvent this shortcoming by performing alternative computations in case the majority outcome does not satisfy IC. Here, on the other hand, a different approach is in place: by checking whether IC satisfies certain properties and the aggregation rule satisfies certain axioms, we can be sure that the outcome will always be consistent with IC.

4.4 Summary

In this chapter we studied the problem of the *safety of the agenda*. This line of research in Judgment Aggregation studies which classes of aggregation rules (characterized by the axioms they satisfy) are ensured to always return a consistent outcome depending on the syntactic properties of the integrity constraint.

In particular, we have first seen a brief recap of the relevant concepts of formula-based JA for the safety of the agenda. We have seen the notion of *minimally inconsistent subset*, essential to translate a problem in formula-based JA into a problem of Binary Aggregation with IC. We then approached the structure of an integrity constraint from the point of view of its *prime implicants*. Then, we provided a DL-PA formula which is true if and only if the current valuation is essentially making true all the elements of a single mi-set of the agenda Φ (or of a single prime implicant of $\neg$IC).

Lastly, we showed how to express various agenda properties as DL-PA formulas. In particular, we discussed the *median*, *k-median*, *simplified median* and *syntactic simplified median* properties. Then, we linked the work of the previous Chapters (where we learned how to express aggregation rules and axioms) with the current one (on properties of IC) by writing known results about the safety of the agenda in DL-PA.

Conclusion

Thesis Review

The starting point of our investigation has been the idea of expressing the framework of Judgment Aggregation in an existing logical calculus, instead of coming up with a new logic. More precisely, we wondered to what extent it would have been possible to translate the model of Binary Aggregation with Integrity Constraints for JA into Dynamic Logic of Propositional Assignments.

In the first place, we introduced both Binary Aggregation with Integrity Constraints and Dynamic Logic of Propositional Assignments. We then figured out how to encode *any* instance of a BA problem into DL-PA: positions in a profile were turned into propositional variables, and their values into an assignment of true or false. Aggregation rules were then naturally encoded as DL-PA programs taking these variables as input.

This led to a further exploration of aggregation rules. We showed how to concretely express many aggregators as DL-PA programs, and we linked the outcome of the former with the output of the latter through some propositions ensuring the correctness of the proposed translation. The propositions were different in case we were considering a resolute or an irresolute aggregation rule, and we showed the structure of a proof for both of them.

After having carried out these translations, we wondered whether it was possible to also express axioms, i.e. properties of aggregation rules, in DL-PA. We showed that the answer was positive both for axioms referring to a single profile and for axioms referring to multiple profiles. In particular, the second type required us to really make use of the language of DL-PA to consider two profiles at the same time. Again, we showed how to link statements in JA about a rule satisfying an axiom with the formulas we built.

In the final part, we turned to the formula-based JA problem of the safety of the agenda. This was motivated from the fact that we managed in the previous parts to translate both aggregation rules and axioms, and from the fact that this area studies which classes of aggregators (defined according to the axioms they satisfy) are ensured

to always return an outcome consistent with IC. We adapted to our setting the concept of minimally inconsistent subset of the agenda, and we proposed a new characterisation of properties of agendas by making use of the language of DL-PA. We concluded by expressing known results of this field in DL-PA.

Closing and Future Work

There are many interesting directions for future research to continue the work presented here. We provide a brief overview of some of them, to conclude the thesis.

First of all, though quite extensive, our work of translating aggregation procedures as DL-PA programs is not complete. This line of research could be continued, for instance, by working on the recently studied family of *binomial rules* [Costantini et al., 2016]. These rules are particularly interesting because they allow to take into account some hidden dependencies among the issues, so that even though two outcomes may both satisfy a given IC, one of them could be intuitively a better choice than the other (e.g. when choosing combinations of foods and drinks).

Secondly, it could be interesting to translate as DL-PA formulas more axioms known in Binary Aggregation. One of them could be the axiom of *reinforcement* [Costantini et al., 2016; Endriss and Grandi, 2014]. Roughly speaking, this axiom is satisfied in case if the intersection between the outcome(s) of a rule on two different profiles on the same issues is non-empty, then it should be equal to the outcome of the rule on the profile resulting from concatenating the two profiles.

Along the same lines, we could consider extending the study of agenda properties. For instance, this could mean trying to give a characterisation and a DL-PA translation of agenda properties such as *even-number negatable*: namely, an agenda containing a minimally inconsistent subset Y such that $(Y \setminus Z) \cup \{\neg p \mid p \in Z\}$ is consistent, for $Z \subseteq Y$ and Z having even cardinality [Dietrich and List, 2009].

The last suggested direction for further investigation is of a different nature. A characteristics of DL-PA is that this modal logic is grounded on propositional logic. Hence, there is a procedure to translate DL-PA formulas as formulas of propositional logic [van Eijck, 2000; Balbiani et al., 2013]. We can thus conceive the possibility of using SAT-solvers to enhance research in Judgment Aggregation via a preliminary translation into DL-PA as the one we provided here, and then into propositional logic.

The approach of using computers to guide research has been proven successful in Computational Social Choice and PA [Tang and Lin, 2009; Geist and Endriss, 2011]. Moreover, there has also been some study for a translation of DL-PA into the Quantified

Boolean Formulas language and its subsequent implementation [Cultien, 2011]. Therefore, it is quite natural to consider the exploration of this path for our work as well, and we leave it for future analysis.

References

T. Ågotnes, W. van der Hoek, and M. Wooldridge. On the logic of preference and judgment aggregation. *Autonomous Agents and Multi-Agent Systems*, 22(1):4–30, 2011.

P. Balbiani, A. Herzig, and N. Troquard. Dynamic logic of propositional assignments: a well-behaved variant of PDL. In *Proceedings of the 28th Annual ACM/IEEE Symposium on Logic in Computer Science, (LICS 2013)*, 2013.

P. Blackburn, M. de Rijke, and Y. Venema. *Modal Logic*, volume 53. Cambridge University Press, 2001.

S. J. Brams and P. C. Fishburn. Voting procedures. In *Handbook of social choice and welfare*, volume 1, pages 173–236. Elsevier, 2002.

F. Brandt, V. Conitzer, and U. Endriss. Computational Social Choice. In G. Weiss, editor, *Multiagent Systems*, pages 213–283. MIT Press, 2013.

G. Ciná and U. Endriss. A Syntactic Proof of Arrow's Theorem in a Modal Logic of Social Choice Functions. In *Proceedings of the 14th International Conference on Autonomous Agents and Multiagent Systems, (AAMAS 2015)*, 2015.

M. Costantini, C. Groenland, and U. Endriss. Judgment Aggregation under Issue Dependencies. In *Proceedings of the 30th AAAI Conference on Artificial Intelligence (AAAI 2016)*, 2016.

C. Cultien. Création d'un solveur pour une logique dynamique avec affectation propositionnelle à l'aide d'une traduction en QBF. Master's thesis, Université Paul Sabatier, Toulouse, 2011.

F. Dietrich. Judgment aggregation: (im)possibility theorems. *Journal of Economic Theory*, 126(1):286–298, 2006.

F. Dietrich and C. List. Judgment aggregation by quota rules: Majority voting generalized. *Journal of Theoretical Politics*, 19(4):391–424, 2007.

F. Dietrich and C. List. Propositionwise judgment aggregation. 2009.

S. Doutre, A. Herzig, and L. Perrussel. A Dynamic Logic Framework for Abstract Argumentation. In *Proceedings of the 14th International Conference on Principles of Knowledge Representation and Reasoning, (KR 2014)*, 2014.

J. van Eijck. Making Things Happen. *Studia Logica*, 66(1):41–58, 2000.

U. Endriss. Logic and Social Choice Theory. In A. Gupta and J. van Benthem, editors, *Logic and Philosophy Today*, volume 2, pages 333–377. College Publications, 2011.

U. Endriss. Judgment Aggregation. In F. Brandt, V. Conitzer, U. Endriss, J. Lang, and A. D. Procaccia, editors, *Handbook of Computational Social Choice*. Cambridge University Press, 2016.

U. Endriss and U. Grandi. Binary Aggregation by Selection of the Most Representative Voter. In *Proceedings of the 28th AAAI Conference on Artificial Intelligence, (AAAI 2014)*, 2014.

U. Endriss, U. Grandi, and D. Porello. Complexity of Judgment Aggregation. *Journal of Artificial Intelligence Research*, 45:481–514, 2012.

U. Endriss, U. Grandi, R. de Haan, and J. Lang. Succinctness of Languages for Judgment Aggregation. In *Proceedings of the 15th International Conference on Principles of Knowledge Representation and Reasoning, (KR 2016)*, 2016.

M. J. Fischer and R. E. Ladner. Propositional Dynamic Logic of Regular Programs. *Journal of computer and system sciences*, 18(2):194–211, 1979.

B. Gaudou, A. Herzig, E. Lorini, and C. Sibertin-Blanc. How to do social simulation in logic: modelling the segregation game in a dynamic logic of assignments. In *International Workshop on Multi-Agent Systems and Agent-Based Simulation*, pages 59–73. Springer, 2011.

C. Geist and U. Endriss. Automated search for impossibility theorems in social choice theory: Ranking sets of objects. *Journal of Artificial Intelligence Research*, 40:143–174, 2011. doi: 10.1613/jair.3126.

U. Grandi. *Binary Aggregation with Integrity Constraints*. PhD thesis, ILLC, University of Amsterdam, 2012.

U. Grandi and U. Endriss. Binary Aggregation with Integrity Constraints. In *Proceedings of the 22nd International Joint Conference on Artificial Intelligence, (IJCAI 2011)*, 2011.

A. Herzig. Belief Change Operations: A Short History of Nearly Everything, Told in Dynamic Logic of Propositional Assignments. In *Proceedings of the 14th International Conference on Principles of Knowledge Representation and Reasoning, (KR 2014)*, 2014.

A. Herzig, E. Lorini, F. Moisan, and N. Troquard. A dynamic logic of normative systems. In *IJCAI*, volume 2011, pages 228–233, 2011.

J. G. Kemeny. Mathematics without Numbers. *Daedalus*, 88(4):577–591, 1959.

J. Lang and M. Slavkovik. How Hard is it to Compute Majority-Preserving Judgment Aggregation Rules? In *Proceedings of the 21st European Conference on Artificial Intelligence, (ECAI 2014)*, 2014.

J. Lang, G. Pigozzi, M. Slavkovik, and L. van der Torre. Judgment Aggregation Rules Based on Minimization. In *Proceedings of the 13th Conference on Theoretical Aspects of Rationality and Knowledge*, pages 238–246. ACM, 2011.

C. List. The theory of judgment aggregation: An introductory review. *Synthese*, 187(1): 179–207, 2012.

C. List and P. Pettit. Aggregating sets of judgments: An impossibility result. *Economics and Philosophy*, 18(01):89–110, 2002.

C. List and C. Puppe. Judgment aggregation: A survey. In P. Anand, P. Pattanaik, and C. Puppe, editors, *Handbook of Rational and Social Choice*. Oxford University Press, 2009.

J. Marchi, G. Bittencourt, and L. Perrussel. Prime Forms and Minimal Change in Propositional Belief Bases. *Annals of Mathematics and Artificial Intelligence*, 59(1): 1–45, 2010.

P. Marquis. Consequence Finding Algorithms. In *Handbook of Defeasible Reasoning and Uncertainty Management Systems*, pages 41–145. Springer, 2000.

K. O. May. A Set of Independent Necessary and Sufficient Conditions for Simple Majority Decision. *Econometrica: Journal of the Econometric Society*, pages 680–684, 1952.

M. K. Miller and D. Osherson. Methods for distance-based judgment aggregation. *Social Choice and Welfare*, 32(4):575–601, 2009.

K. Nehring, M. Pivato, and C. Puppe. Condorcet admissibility: Indeterminacy and path-dependence under majority voting on interconnected decisions. 2011.

T. Perkov. Natural Deduction for Modal Logic of Judgment Aggregation. *Journal of Logic Language and Information*, pages 1–20, 2016. http://dx.doi.org/10.1007/s10849-016-9235-x.

V. R. Pratt. Semantical considerations on Floyd-Hoare logic. In *Foundations of Computer Science, 1976., 17th Annual Symposium on*, pages 109–121. IEEE, 1976.

A. Rubinstein. The single profile analogues to multi profile theorems: Mathematical logic's approach. *International Economic Review*, pages 719–730, 1984.

P. Tang and F. Lin. Computer-aided proofs of Arrow's and other impossibility theorems. *Artificial Intelligence*, 173(11):1041–1053, 2009.

N. Troquard and P. Balbiani. Propositional Dynamic Logic. In E. N. Zalta, editor, *The Stanford Encyclopedia of Philosophy*. Spring 2015 edition, 2015. http://plato.stanford.edu/archives/spr2015/entries/logic-dynamic/.

N. Troquard, W. van der Hoek, and M. Wooldridge. Reasoning About Social Choice Functions. *Journal of Philosophical Logic*, 40(4):473–498, 2011.

M. B. Wells. Generation of Permutations by Transposition. *Mathematics of Computation*, pages 192–195, 1961.