Judgement Aggregation in Dynamic Logic of Propositional Assignments

Arianna Novaro
IRIT, University of Toulouse
arianna.novaro@irit.fr

Umberto Grandi
IRIT, University of Toulouse
umberto.grandi@irit.fr

Andreas Herzig
CNRS-IRIT, University of Toulouse
andreas.herzig@irit.fr

ABSTRACT

Judgment aggregation studies situations where groups of agents take a collective decision over a number of logically interconnected issues. A recent stream of papers is dedicated to modelling frameworks of social choice theory, including judgment aggregation, within logical calculi usually designed *ad hoc* for this purpose. In contrast, we propose the use of dynamic logic of propositional assignments (DL-PA), an instance of propositional dynamic logic based on atomic programs modifying propositional evaluations. We provide logical equivalents in DL-PA for the most known aggregation procedures from the literature, for axiomatic properties, and for properties of the constraints, thus showing the versatility of this language for dealing with judgment aggregation.

CCS Concepts

•**Computing methodologies** → **Multi-agent systems;**

Keywords

Social Choice Theory; Dynamic logic; Modal logic; Computational Social Choice; Automated Reasoning

1. INTRODUCTION

Social choice theory gathers a number of mathematical models for the study of collective decisions, such as voting and elections, or the allocation of resources among a group of agents. Judgment aggregation is one such model, in which individuals express binary judgments over a set of interconnected issues, which are then aggregated into a collective choice by means of an aggregation rule. This model can be traced back to work by legal scholars [22] and it is now an established framework in artificial intelligence to study complex collective decisions [12, 18].

In judgment aggregation, the correlation among the issues is typically modelled by making use of simple propositional languages. This explicit link with logic inspired researchers to look for a full logical formalisation of the setting, developing logical formalisms that are able to express and reason about aggregation rules and their properties. These efforts are part of a fertile research agenda connecting logic with social choice theory (see, e.g., Endriss, [11]). To cite some examples, Arrow's Theorem [2], one of the cornerstones of social choice theory, has been formalised into higher-order logics [33, 27], first-order logic [17] and modal logic [7]. The ultimate goal of this program is to use automated reasoning techniques to discover new results, an objective that has been partially reached by combining the use of SAT solvers with mathematical lemmas, in preference aggregation [31], ranking sets of objects [15], and in classical social choice theory [5, 6].

Two full-fledged formalisation of judgment aggregation and preference aggregation made use of modal logic: namely, Judgment Aggregation Logic — of which both Hilbert-style [1] and natural deduction [29] axiomatisations have been provided — and the Logic for Social Choice Functions proposed by Troquard et al. [32]. In both cases, the authors develop their own modal languages to formalise judgment aggregation, making the application of automated reasoning techniques less immediate. In this paper, instead, we propose to use the existing language of Dynamic Logic of Propositional Assignments DL-PA [10, 3]. This logic is an instance of Propositional Dynamic Logic PDL (see, e.g., [4]), where atomic programs consist of assignments of truth values to propositional variables. DL-PA is also grounded on propositional logic: in other words, there exists a procedure to translate every modal formula in DL-PA as a propositional formula [10, 3], showing a direct connection with automated reasoning via the use of SAT solvers. Moreover, numerous knowledge representation problems have been expressed in DL-PA, such as belief change operations [19] and abstract argumentation problems [9], and it is arguably a natural choice for the setting of judgment aggregation, where individual opinions are represented as binary evaluations.

We translate most aggregation rules proposed in the literature on judgment aggregation as DL-PA programs, ensuring that the size of each program remains polynomial in the number of agents and issues. Consider for instance the classical majority rule, which collectively accepts a given issue if the number of agents accepting it is greater than the number of agents rejecting it. A straightforward translation of this rule would make use of the explicit description of all possible majorities (i.e., coalitions of more than half of the agents), which would take exponential space. The formalisation in DL-PA we propose solves this problem by a clever use of counters.

Aggregation rules are typically classified and justified by means of axiomatic properties, which are then used in the literature to obtain limitative results on the boundaries of

Appears at: 4th Workshop on Exploring Beyond the Worst Case in Computational Social Choice (EXPLORE 2017). Held as part of the Workshops at the 16th International Conference on Autonomous Agents and Multiagent Systems. May 8th-9th, 2017. Sao Paulo, Brazil.

aggregation — the notorious impossibility theorems. We provide DL-PA formulas for the most used aggregation axioms, which can then be interpreted on the translation of a rule. As an aside, we obtain an interesting distinction between axioms that bound the result of the aggregation on one profile, for which we find a translation into propositional logic, and those that require reasoning about multiple profiles, for which DL-PA needs to be used to obtain a compact representation. The final part of the paper focuses on the problem of guaranteeing a safe aggregation, i.e., identifying those types of logical dependencies among the issues such that aggregating individual judgments yields a result consistent with them. In our framework, this problem boils down to checking the validity of a corresponding DL-PA formula.

The paper is organized as follows. In Section 2 we provide the basic definitions of judgment aggregation and of the DL-PA language, as well as setting the stage for a translation of the former into the latter. In Section 3 we propose DL-PA programs to compute the most known judgment aggregation procedures. Section 4 provides translations for the axiomatic properties of aggregation functions, and Section 5 focuses on characterising formulas for safe aggregation. Section 6 concludes the paper and points at a number of directions for future work. We omit most of the proofs in the interest of space: full proofs of the main results can be found in Novaro's Master Thesis [28].

2. PRELIMINARIES

In this section we introduce the formal framework of both binary aggregation with integrity constraints and *star-free* Dynamic Logic of Propositional Assignments. Moreover, we provide our first contribution by showing how to translate aggregation problems into the logic of our choice.

2.1 Binary Aggregation with Integrity Constraints

Two main frameworks can be considered for judgment aggregation: the classic *formula-based* model [24], in which individuals vote directly on complex logical formulas, and *binary aggregation with integrity constraints* [16] where agents have binary opinions on atomic issues linked by an integrity constraint. In this paper we choose the latter setting, and we present it briefly below.

Let $\mathcal{I} = \{1, \dots, m\}$ be a finite non-empty set of *issues*, on which the *agents* in the finite non-empty set $\mathcal{N} = \{1, \dots, n\}$, for odd n (as we shall see, this is just a technical assumption), express a binary opinion. Individual opinions form a *boolean combinatorial domain* $\mathcal{D} = \{0,1\}^m$, where "1" denotes acceptance and "0" rejection. A simple propositional language $\mathcal{L}_{PS}$ can be defined from the set of propositional symbols $PS = \{p_1, \dots, p_m\}$, with one atom per issue in $\mathcal{I}$. Then, *integrity constraints* can be defined as formulas IC $\in \mathcal{L}_{PS}$, to express the existence of logical interdependencies among the issues. If there is none, we let IC $= \top$. Consider the following classical example of aggregation, known in the literature as the *discursive dilemma* [22]:

EXAMPLE 1. *Three judges have to decide whether (1) a defendant is liable for breaching a contract, depending on whether (2) the contract forbade a particular action and (3) the defendant did it anyway. Let thus* IC $= p_1 \leftrightarrow p_2 \wedge p_3$, *and consider the profile below:*

	p_1	p_2	p_3
Judge 1	*1*	*1*	*1*
Judge 2	*0*	*0*	*1*
Judge 3	*0*	*1*	*0*
Majority	*0*	*1*	*1*

As we can see, while the three judges all respect the integrity constraint, the majority outcome does not. Hence, it is not clear whether the judges should give their sentence based on the collective judgment on the conclusion (the defendant is not liable) or the premises (the defendant did an action that was forbidden by the contract).

A *ballot* $B = (b_1, \dots, b_m) \in \mathcal{D}$ is a particular choice of zeroes and ones for the issues. The set of all ballots satisfying IC, written $\text{Mod}(\text{IC}) = \{B \mid B \models \text{IC}\}$, is called the *models* of IC. We denote by B_i the *individual ballot* of agent i, and we assume $B_i \in \text{Mod}(\text{IC})$ for all $i \in \mathcal{N}$: the agents are *rational*. A *profile* $\boldsymbol{B} = (B_1, \dots, B_n)$ collects all the individual ballots of the agents, such that b_{ij} indicates the j-th element of ballot B_i in $\boldsymbol{B}$. The set $N^{\boldsymbol{B}}_{j:1} = \{i \in \mathcal{N} \mid b_{ij} = 1\}$ is the *coalition of supporters* of issue j in $\boldsymbol{B}$.

An *aggregation procedure* (*aggregation rule*, *aggregator*) is a function F mapping a rational profile to a (possibly irrational) non-empty set of ballots.

DEFINITION 1. *Given a set of agents $\mathcal{N}$, a set of issues $\mathcal{I}$ and an integrity constraint* IC, *an* aggregation procedure *is a function* $F : \text{Mod}(\text{IC})^{\mathcal{N}} \to 2^{\mathcal{D}} \setminus \emptyset$, *for* $2^{\mathcal{D}}$ *the powerset of* $\mathcal{D}$. *A rule is called* resolute *if its outcome is a singleton for every profile, and* irresolute *otherwise. We denote by* $F(\boldsymbol{B})_j$ *the outcome of a resolute aggregator on issue j.*

The *Hamming distance* measures how much two ballots disagree on the issues, and is defined as $H(B, B^*) = |\{j \in \mathcal{I} \mid b_j \neq b^*_j\}|$. For example, if $B_1 = (1,0,0)$ and $B_2 = (1,1,1)$, we have $H(B_1, B_2) = 2$, since they only differ on the last two issues.

2.2 Dynamic Logic of Propositional Assignments

To describe problems in judgment aggregation we choose the language of Dynamic Logic of Propositional Assignments DL-PA [10, 3], an instance of Propositional Dynamic Logic PDL, where atomic programs assign truth value true or false to propositional variables. This logic has already been used to model multi-agent scenarios, such as interactions of agents in normative systems [20] and social simulations [14]. More precisely, we focus on the *star-free* version of DL-PA, without unbounded iteration — which can be obtained from DL-PA via the elimination of the Kleene star [3].

The language of star-free DL-PA is given by the following Backus-Naur grammar:

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

where p ranges over $\mathbb{P} = \{p, q, \dots\}$, a countable set of propositional variables.

Atomic *formulas* consist of variables and constants $\top$ and $\bot$. Complex formulas are built via negation $\neg$, disjunction $\vee$, and a diamond modality for each program $\langle\pi\rangle$. Other Boolean connectives (e.g., conjunction $\wedge$, implication $\to$, biconditional $\leftrightarrow$, exclusive disjunction $\oplus$) and the dual operator $[\pi]\varphi$ are defined in the usual way. Atomic *programs*

$+p$ and $-p$ assign truth value true or false to variable p, respectively. Sequential composition $\pi ; \pi'$ executes first π and then π', nondeterministic union $\pi \cup \pi'$ nondeterministically chooses to execute either π or π', and test $\varphi?$ checks that φ holds.

A *valuation* v is a subset of $\mathbb{P}$ that specifies the truth value of every propositional variable, so that $\mathbb{V} = 2^{\mathbb{P}} = \{v_1, v_2, \dots\}$ is the set of all valuations. When $p \in v$, we say that p is *true* in v (and we say that p is *false* in v otherwise). As illustrated in Table 1, DL-PA programs are interpreted through a unique relation between valuations

$$\begin{aligned}
\|p\| &= \{v \in \mathbb{V} \mid p \in v\} \\
\|\top\| &= 2^{\mathbb{P}} \\
\|\bot\| &= \emptyset \\
\|\neg\varphi\| &= 2^{\mathbb{P}} \setminus \|\varphi\| \\
\|\varphi \vee \psi\| &= \|\varphi\| \cup \|\psi\| \\
\|\langle\pi\rangle\varphi\| &= \{v \in \mathbb{V} \mid \exists v_1 \text{ s.t. } (v, v_1) \in \|\pi\| \text{ and } v_1 \in \|\varphi\|\} \\
\|+p\| &= \{(v_1, v_2) \mid v_2 = v_1 \cup \{p\}\} \\
\|-p\| &= \{(v_1, v_2) \mid v_2 = v_1 \setminus \{p\}\} \\
\|\pi ; \pi'\| &= \|\pi\| \circ \|\pi'\| \\
\|\pi \cup \pi'\| &= \|\pi\| \cup \|\pi'\| \\
\|\varphi?\| &= \{(v, v) \mid v \in \|\varphi\|\}
\end{aligned}$$

Table 1: Interpretation of DL-PA expressions

Abbreviations have been introduced in the literature to make programs more readable [3, 4, 19]. As a convention, abbreviations for formulas will start with an uppercase letter, while those for programs and counters will start with a lowercase letter. We thus have $\mathsf{skip} := \top?$, $\mathsf{if}\ \varphi\ \mathsf{then}\ \pi_1\ \mathsf{else}\ \pi_2 := (\varphi? ; \pi_1) \cup (\neg\varphi? ; \pi_2)$, $p \leftarrow q := \mathsf{if}\ q\ \mathsf{then} + p\ \mathsf{else} - p$ and $\mathsf{if}\ \varphi\ \mathsf{do}\ \pi := \mathsf{if}\ \varphi\ \mathsf{then}\ \pi\ \mathsf{else\ skip}$, as well as repeated execution of program π for n times, or up to n times (where both programs execute flip for $n = 0$):

$$\begin{aligned}
\pi^n &:= \pi ; \pi^{n-1} \\
\pi^{\leq n} &:= (\mathsf{skip} \cup \pi) ; \pi^{\leq n-1}
\end{aligned}$$

We can write any number $s \in \mathbb{N}_0$ in DL-PA via its binary expression, thanks to a conjunction of $t = \lfloor \log s \rfloor + 1$ variables [3]. If x is the binary expression of s, we use a conjunction of q_i and $\neg q_i$ propositional variables, with $i \in \{0, \dots, \lfloor \log s \rfloor\}$, such that a non-negated variable means that the corresponding binary digit in x is a 1, while a negated variable indicates a 0. For instance, if $s = 11$, we have that $x = 1011$ and the corresponding formula in DL-PA is $\mathsf{11} := q_3 \wedge \neg q_2 \wedge q_1 \wedge q_0$.

The following two programs increment or set to zero (i.e., assign truth value false to all the variables in P) a given counter [3]. Let $\mathsf{x}^t := \{q_i^x \mid 0 \leq i < t\}$ be a set of variables:

$$\begin{aligned}
\mathsf{incr}(\mathsf{x}^t) &:= \neg\Big(\bigwedge_{0 \leq i \leq t-1} q_i^x\Big)? ; \bigcup_{0 \leq k \leq t-1} \Big((\neg q_k^x \wedge \bigwedge_{0 \leq i \leq k-1} q_i^x)? ; \\
&\qquad + q_k^x ; \mathop{;}_{0 \leq i \leq k-1} -q_i^x\Big) \\
\mathsf{zero}(P) &:= \mathop{;}_{p \in P} -p
\end{aligned}$$

We can compare two numbers and check whether one of them is greater than the other, they are equal, or one of them is greater or equal to the other, via the following DL-PA formulas. The general idea is to compare the digits at the same position in the binary expressions of the two numbers.[1]

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

As a convention, we let $\bigwedge_{k<i<t}(q_i^x \leftrightarrow q_i^y) = \top$ for $k = t - 1$.

Additionally, we may want to flip the truth value of some variables in a set P. The first program below flips the truth value of a single, nondeterministically chosen, variable in P. The second resets the truth value of all variables in P to some new value: as a result, either their truth value has been flipped or not. Both programs execute skip for $P = \emptyset$.

$$\begin{aligned}
\mathsf{flip}^1(P) &:= \bigcup_{p \in P} (p \leftarrow \neg p) \\
\mathsf{flip}^{\geq 0}(P) &:= \mathop{;}_{p \in P} (+p \cup -p)
\end{aligned}$$

The next two formulas hold when different types of minimisation are achieved. The first is true if and only if $\neg\varphi$ holds whenever we *do not* change the truth value of some variable in the non-empty set P. The second holds if and only if we found the minimal Hamming distance s between the states of before and after flipping the variables in P, such that φ holds afterwards:

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

Observe that $\mathsf{D}(\varphi, P)$ does not imply that φ *will* hold if we flip the truth value of all the variables in P. In our setting this definition suffices, but such alternative formulation has been given as well [19].

2.3 Translating Aggregation Problems into DL-PA

We here show how to translate profiles and aggregation rules into DL-PA. The former is turned into a specific type of valuation, while the latter become programs. We also show how to check rationality in DL-PA and how to turn an arbitrary valuation into one corresponding to a profile.

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

[1] Suppose two numbers can be expressed with a different amount of binary digits. In this case, if in some program we need to use many counters, we take the maximal value a counter could take as the upper bound for *all* counters in that program. Hence, if t is the maximal number of variables needed to express the maximal value a counter can take, and some other number is expressible by using only k variables (where $k < t$), it will nonetheless be expressed with t variables by imposing $\neg q_i$ for all $k < i \leq t$. We thus write x instead of x^t.

the variables in $\mathbb{O}^m := \{p_j \mid j \in \mathcal{I}\}$ refer to the collective decision on the issues in $\mathcal{I}$. Finally, $\mathbb{U} := \{q_i \mid i \in \mathbb{N}\}$ is the subset of $\mathbb{P}$ whose variables are used for finitely many counters in our programs.

The following definition carves out the valuations that correspond to a profile in judgment aggregation.

Definition 2. *We say that valuation $v_{\boldsymbol{B}}$* translates *profile $\boldsymbol{B} = (B_1, \ldots, B_n)$ on m issues, in case:*

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

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

The first condition ensures that only variables corresponding to the decision of the agents on the issues could possibly be true in $v_{\boldsymbol{B}}$. This means, in particular, that counters are initially set to zero. According to the second condition, a variable in $v_{\boldsymbol{B}}$ is true if and only if the corresponding entry in profile $\boldsymbol{B}$ has value 1. For example, if we have profile $\boldsymbol{B} = ((0,1),(0,0),(1,0))$ for 3 agents and 2 issues, the set $\mathbb{B}^{3,2} = \{p_{11}, p_{12}, p_{21}, p_{22}, p_{31}, p_{32}\}$ corresponds to the entries in the profile, the set $\mathbb{O}^2 = \{p_1, p_2\}$ handles the outcome of aggregation rules and valuation $v_{\boldsymbol{B}} = \{p_{12}, p_{31}\} \subseteq \mathbb{B}^{3,2}$ encodes the values of the profile.

We now introduce the definition for translating aggregation rules as DL-PA programs.

Definition 3. *A program $\mathsf{f}(\mathbb{B}^{n,m})$* translates *aggregation rule F, if for all profiles $\boldsymbol{B}$ and valuations $v_{\boldsymbol{B}}$ translating $\boldsymbol{B}$ according to Definition 2, it is the case that:*

- *F is resolute and $(v_{\boldsymbol{B}}, v') \in \| \mathsf{f}(\mathbb{B}^{n,m}) \|$, implies that for all $j \in \mathcal{I}$ and $p_j \in \mathbb{O}^m$:*
$$p_j \in v' \iff F(\boldsymbol{B})_j = 1.$$
- *F is irresolute and $V^{\mathsf{f}}_{v_{\boldsymbol{B}}} = \{v' \mid (v_{\boldsymbol{B}}, v') \in \| \mathsf{f}(\mathbb{B}^{n,m}) \|\}$, implies that there is a bijection $g : F(\boldsymbol{B}) \to V^{\mathsf{f}}_{v_{\boldsymbol{B}}}$ such that if $g(B) = v'$ then for all $j \in \mathcal{I}$ and $p_j \in \mathbb{O}^m$:*
$$p_j \in v' \iff b_j = 1.$$

We write the integrity constraint as a formula IC over variables in $\mathbb{O}^m$. In order to check whether a particular choice of truth values over $\mathbb{B}^{n,m}$ corresponds to a profile, i.e., all the individual ballots satisfy the constraint, we check if the following formula holds.

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

Namely, we check whether by copying into the outcome variables the truth values of the variables for each individual ballot, the constraint IC holds.

The following program leads from an arbitrary valuation to one that possibly corresponds to the encoding of a profile, by creating the "right" initial conditions:

$$\mathsf{prof}_{\mathrm{IC}}(\mathbb{B}^{n,m}, \mathbb{O}^m) := \mathsf{zero}(\mathbb{O}^m) ; \mathsf{Rational}_{\mathrm{IC}}(\mathbb{B}^{n,m})?$$

Observe that after its execution all the outcome variables are false, but it is not enough to conclude that condition (i) of Definition 2 holds. Nonetheless, all programs encoding aggregation rules will just need to inspect variables in $\mathbb{B}^{n,m}$ and (possibly) change the truth values of variables in $\mathbb{O}^m$, and they will initialise at zero all counters as the first step. Therefore, we consider the valuation reached after the execution of $\mathsf{prof}_{\mathrm{IC}}(\mathbb{B}^{n,m}, \mathbb{O}^m)$ as encoding a profile as well.

To conclude this section, we highlight an important remark. Since aggregation rules are defined over a specific number of issues, number of agents and integrity constraint, the programs we provide as their DL-PA translation are to be intended as general *program schemas*: a set of issues $\mathcal{I}$, set of agents $\mathcal{N}$ and constraint IC need to be given to completely spell them out.

3. AGGREGATION RULES

Aggregation rules are the basic bricks of judgment aggregation, allowing to reach a group decision from individual choices. In this section we translate known aggregation rules as DL-PA programs, omitting the proof of correctness of our translations for space constraints.

3.1 Expressibility of Aggregation Rules

We begin by proving a general result that shows how any judgment aggregation rule, as introduced in Definition 1, can be expressed as a DL-PA program.

Theorem 1. *All aggregation rules $F : \mathrm{Mod}(\mathrm{IC})^{\mathcal{N}} \to 2^{\mathcal{D}} \setminus \emptyset$ for some $\mathcal{N}$, $\mathcal{I}$ and* IC *are expressible as* DL-PA *programs.*

Proof. We first deal with the case of a resolute aggregation rule F. Consider the DL-PA program consisting of a sequential composition of sub-programs of the form $\mathsf{if}\ \varphi_{\boldsymbol{B}}\ \mathsf{do}\ \pi_{F(\boldsymbol{B})}$ for each profile $\boldsymbol{B}$, where $\varphi_{\boldsymbol{B}} = (\bigwedge_{j \in \mathcal{I}} \bigwedge_{i \in N^{\boldsymbol{B}}_{j:1}} p_{ij}) \wedge (\bigwedge_{j \in \mathcal{I}} \bigwedge_{i \in (\mathcal{N} \setminus N^{\boldsymbol{B}}_{j:1})} \neg p_{ij})$, i.e., $\varphi_{\boldsymbol{B}}$ completely identifies profile $\boldsymbol{B}$, and $\pi_{F(\boldsymbol{B})} = \mathop{;}_{\{j \in \mathcal{I} | F(\boldsymbol{B})_j = 1\}} +p_j ; \mathop{;}_{\{j \in \mathcal{I} | F(\boldsymbol{B})_j = 0\}} -p_j$, i.e., $\pi_{F(\boldsymbol{B})}$ modifies the outcome variables according to the result of F on profile $\boldsymbol{B}$.

For irresolute F it sufficies to consider a sequential composition of sub-programs of the form $\mathsf{if}\ \varphi_{\boldsymbol{B}}\ \mathsf{do}\ \bigcup_{B \in F(\boldsymbol{B})} \pi_B$, where π_B is defined as $\pi_B = \mathop{;}_{\{j \in \mathcal{I} | b_j = 1\}} +p_j ; \mathop{;}_{\{j \in \mathcal{I} | b_j = 0\}} -p_j$, generating a non-deterministic program whose output consists of all outcomes of F. These two types of programs clearly translate resolute and irresolute aggregation rules. □

While on the one hand the result above shows that DL-PA is fully expressive when it comes to translating judgment aggregation rules, on the other hand the formulas used in the proof are all of size exponential in the number of individuals and issues. More precisely, since all profiles are explicitly given in the specification of the programs, the size is in the order of $2^{|\mathcal{I}|} \times |\mathcal{N}|$. In the remainder of this section we thus present compact programs for a selection of well-known judgment aggregation rules.

3.2 Simple Aggregation Rules

We call the following rules *simple* because they are all resolute, they are easy to explain and understand, and they can also be found in real-world examples.

3.2.1 Dictatorship of Agent i

The dictatorial rule is perhaps the simplest and at the same time less attractive aggregation rule. For all profiles $\boldsymbol{B}$, the outcome of the dictatorship of some fixed agent $i \in \mathcal{N}$ is her individual ballot. Namely, $\mathrm{Dictatorship}_i(\boldsymbol{B})_j = 1 \iff b_{ij} = 1$ for all $j \in \mathcal{I}$. Its translation in DL-PA can easily be obtained as the following program:

PROPOSITION 1. *Let $\mathcal{I}$ and $\mathcal{N}$ be given. Then, program* $\mathsf{dict}_i(\mathbb{B}^{n,m}) := ;_{j\in\mathcal{I}}(p_j \leftarrow p_{ij})$ *translates rule* Dictatorship_i.

3.2.2 Quota Rules

The majority rule is an instance of the more general class of *quota rules* [8]. A quota rule specifies for each issue a certain threshold of support that has to be reached in order for the issue to be accepted in the outcome. The quota q can be any integer such that $0 \leq q \leq n+1$, where n is the number of agents. In case all the issues have the same quota, we speak of *uniform* quota rules. If q_j is the quota for issue $j \in \mathcal{I}$ and $\vec{q} = (q_1, \ldots, q_m)$, we have:

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

We now state a result that provides, for every choice of quotas q_j, a DL-PA program translating the corresponding quota rule (by using a counter quota_j for each issue j).

PROPOSITION 2. *For $\mathcal{I}$ a set of issues, $\mathcal{N}$ a set of agents, and $0 \leq q_1, \ldots, q_m \leq |\mathcal{N}|+1$, the* $\text{Quota}_{\vec{q}}$ *rule is translated in the following* DL-PA *program:*

$$\begin{aligned}\mathsf{quota}_{\vec{q}}(\mathbb{B}^{n,m}) \;:=\; & \mathop{;}_{j\in\mathcal{I}} \mathsf{zero}(\mathsf{quota}_j)\,;\, \mathop{;}_{j\in\mathcal{I}} \mathsf{incr}(\mathsf{quota}_j)^{q_j}\,;\\ & \mathop{;}_{j\in\mathcal{I}} \big(\mathsf{zero}(\mathsf{supp})\,;(\mathop{;}_{i\in\mathcal{N}} \mathsf{if}\ p_{ij}\ \mathsf{do}\ \mathsf{incr}(\mathsf{supp}))\,;\\ & \mathsf{if}\ \mathsf{supp} \geq \mathsf{quota}_j\ \mathsf{do}\ +p_j\big).\end{aligned}$$

We refer to the specific program for the majority rule as maj. Moreover, for the uniform quota rule with $q = 1$, called the *nomination rule*, an even more compact program is $\mathsf{nom}(\mathbb{B}^{n,m}) \;:=\; ;_{j\in\mathcal{I}}(\mathsf{if}\ \bigvee_{i\in\mathcal{N}} p_{ij}\ \mathsf{do}\ +p_j)$.

3.3 Maximisation and Minimisation Rules

In this section we focus on two aggregation rules that are based on maximisation or minimisation processes and aim at amending the outcome of the majority rule, in case it does not satisfy the integrity constraint. The first one is the maximal subagenda rule, while the second one is the minimal number of atomic changes rule [23].

3.3.1 Maximal Subagenda Rule

The maximal subagenda rule returns ballots satisfying the integrity constraint and having maximal agreement (with respect to set inclusion) with the majority outcome:

$$\text{MSA}_{\text{IC}}(\boldsymbol{B}) = \underset{B\models\text{IC}}{\overset{\subseteq}{\text{argmax}}}\{j \in \mathcal{I} \mid b_j = \text{Maj}(\boldsymbol{B})_j\}.$$

Before presenting a DL-PA program translating this rule, we need some further notation. Consider the following programs, which all execute skip if $P = \emptyset$:

$$\begin{aligned}\mathsf{store}(P) \;&:=\; \mathop{;}_{p\in P} p' \leftarrow p\\ \mathsf{restore}^1(P) \;&:=\; \bigcup_{p\in P}(p \oplus p'?\,; p \leftarrow p')\\ \mathsf{restore}^{\geq 0}(P) \;&:=\; \mathop{;}_{p\in P}(\mathsf{skip} \cup p \leftarrow p')\end{aligned}$$

Program store stores the truth value of the variables in P in some fresh variables p', program $\mathsf{restore}^1(P)$ restores the truth value of just one variable p' in the corresponding variable in P, and program $\mathsf{restore}^{\geq 0}(P)$ restores the truth value of none, some, or all variables p' in the corresponding variables in P.

We can now present the following program, inspired by analogous work in the literature on belief change [19]. Given that the MSA_{IC} is an irresolute rule we might need to handle multiple outcomes for the same profile: whence its (omitted) proof differs from that of Proposition 1.

PROPOSITION 3. *Let $\mathcal{I}$ be a set of issues, $\mathcal{N}$ a set of agents and IC a propositional formula. The* MSA_{IC} *rule is translated in the following* DL-PA *program:*

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

3.3.2 Minimal Number of Atomic Changes Rule

The minimal number of atomic changes rule returns the following outcome set:

$$\begin{aligned}\text{MNAC}_{\text{IC}}(\boldsymbol{B}) \;=\; \{B \mid \text{Maj}(\boldsymbol{B}^*) = B, B \models \text{IC and for all } \boldsymbol{B}'\\ \sum_{i\in\mathcal{N}} H(B_i, B_i^*) \leq \sum_{i\in\mathcal{N}} H(B_i, B_i')\}.\end{aligned}$$

Recall that the Hamming distance $H(B, B')$ between two ballots is the number of issues on which they differ (cf. Section 2.1). This rule thus looks for profiles which are minimally different from the current one, such that the majority rule applied to them would return an outcome consistent with the constraint.

PROPOSITION 4. *Let $\mathcal{I}$ be a set of issues, $\mathcal{N}$ a set of agents and IC a propositional formula. The* MNAC_{IC} *rule is translated in the following* DL-PA *program:*

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

The program $\mathsf{mnac}_{\text{IC}}$ finds the minimal number d of variables in the set $\mathbb{B}^{n,m}$ whose truth values can be modified such that applying program maj to this new profile leads to a valuation where the outcome satisfies the constraint.

3.4 Preference Aggregation Rules

This section presents rules inspired by the literature on preference aggregation, and that have been formalised in judgment aggregation in a number of papers. The first is the Kemeny rule [21]. Then, we present the Slater rule, also known as the maxcard subagenda rule [23]. A program similar to the ones presented below can be designed to formalise the ranked pairs rule as well (see [28]).

3.4.1 Kemeny Rule

The outcome of the Kemeny rule consists of those ballots that satisfy the constraint and that minimise the sum of the Hamming distance to the individual ballots in the profile.

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

Let us first introduce the following program and formula:

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

$$\mathsf{MD}(\mathbb{O}^m, \mathbb{B}^{n,m}, \mathrm{IC}) := [\mathsf{sH}(\mathbb{O}^m, \mathbb{B}^{n,m})\,;\mathsf{store}(\mathsf{dis})\,;\mathsf{flip}^{\geq 0}(\mathbb{O}^m)\,; \mathsf{sH}(\mathbb{O}^m, \mathbb{B}^{n,m})](\mathsf{dis'} > \mathsf{dis} \to \neg\mathrm{IC}).$$

Program sH computes the sum of the Hamming distances between the outcome and the profile. Formula MD is true if and only if whenever some outcome is closer to the profile than the current one, with respect to the Hamming distance, then IC is not satisfied.

Proposition 5. *Let $\mathcal{I}$ be a set of issues, $\mathcal{N}$ a set of agents and* IC *a propositional formula. The* $\mathrm{Kemeny}_{\mathrm{IC}}$ *rule is translated in the following* DL-PA *program:*

$$\mathsf{kem}_{\mathrm{IC}}(\mathbb{B}^{n,m}) := \bigcup_{0\leq d\leq m} \big(\langle \mathsf{flip}^1(\mathbb{O}^m)^d\rangle(\mathsf{MD}(\mathbb{O}^m, \mathbb{B}^{n,m}, \mathrm{IC}) \wedge \mathrm{IC})?\,;\mathsf{flip}^1(\mathbb{O}^m)^d\big)\,;\mathsf{MD}(\mathbb{O}^m, \mathbb{B}^{n,m}, \mathrm{IC}) \wedge \mathrm{IC}?.$$

The program $\mathsf{kem}_{\mathrm{IC}}$ finds the right d such that by flipping the truth value of d outcome variables we get to a valuation that satisfies the constraint, and such that d is the minimal Hamming distance to the rest of the profile.

3.4.2 Slater Rule

The outcome of the Slater rule consists of those ballots satisfying the constraint and minimising the Hamming distance from the outcome of the majority rule for that profile.

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

Proposition 6. *Let $\mathcal{I}$ be a set of issues, $\mathcal{N}$ a set of agents and* IC *a propositional formula. The* $\mathrm{Slater}_{\mathrm{IC}}$ *rule is translated in the following* DL-PA *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 $\mathsf{slater}_{\mathrm{IC}}$ first computes the majority rule, and then it finds the minimal distance d such that by flipping the truth value of d variables in the outcome we reach a valuation where the constraint is satisfied. In case the majority outcome already satisfies IC, we have that $d = 0$.

4. AXIOMS

Aggregation rules can be characterised according to which general properties they satisfy. These properties are called *axioms* in the literature [8]. In line with similar work in preference aggregation, where properties are sometimes distinguished into *intra-profile* and *inter-profile* conditions [30], we here make a distinction between *single-profile* and *multi-profile* axioms. The former type relates the structure of a profile with the outcome of an aggregation rule applied on that profile. The latter type links the structure of two profiles with the outcomes of the same aggregation rule applied on them.

4.1 Single-profile Axioms

We present four classical single-profile axioms, for which we provide a translation in propositional logic. The full DL-PA machinery is thus not necessary in this case.

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.

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

A rule is *neutral with respect to the issues* if, when two issues are treated in the same way in the input, they are treated in the same way in the output.

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

A 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.

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

A rule is *neutral-monotonic* if the acceptance of an issue j in a given profile implies the acceptance of any other issue k which is accepted by a strict superset of individuals:

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

We are now ready to present the following result:

Theorem 2. *Let $\mathbb{B}^{n,m}$ be the set of variables for agents in $\mathcal{N}$ and issues in $\mathcal{I}$, let F be an aggregation rule for n and m, and let* f *be its* DL-PA *translation. Moreover, let:*

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

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

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

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

Then, the following equivalences hold:

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

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

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

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

4.2 Multi-profile Axioms

We now present three multi-profile axioms, which we translate as DL-PA formulas. In fact, to check whether an aggregation rules satisfies them, we need to compare the outcomes of the rule on different profiles. Dealing with multiple profiles means referring to more than one valuation, and applying the program expressing rule F more than once.

A rule is *independent* if, whenever an issue j is treated in the same way in two profiles, the outcome of the rule for j is identical in both of them. Formally:

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

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 issue j and then she accepts it, if j was accepted in the first outcome then it should still be accepted in the second. Let $(\boldsymbol{B}_{-i}, B'_i) = (B_1, \ldots, B'_i, \ldots, B_n)$ for some profile $\boldsymbol{B}$:

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 $F(\boldsymbol{B}')_j = 1$.

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

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

We can now state the following result:

THEOREM 3. *Let $\mathbb{B}^{n,m}$ be the set of variables for agents in $\mathcal{N}$ and issues in $\mathcal{I}$, let F be an aggregation rule for n and m, and let f be its* DL-PA *translation. Moreover, for $\mathbb{B}^n_j := \{p_{ij} \mid i \in \mathcal{N}\}$ let:*

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

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

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

Then, the following is the case:

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

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

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

5. AGENDA SAFETY

A recurring problem in judgment aggregation is that the outcome of a rule might not respect the given logical dependencies among the issues, even though each agent satisfies the integrity constraint in her individual ballot. As a way out, it can be investigated whether we can ensure that the outcome of certain groups of aggregation rules will always satisfy a given constraint, provided that the constraint relates the issues to one another in a specific way. This approach was first studied in formula-based judgment aggregation under the name of the *safety of the agenda* [13].

5.1 Prime Implicants and Safety

In this preliminary section we make use of the area of logic studying prime implicants to redefine known concepts of the agenda safety problem. Given a set of axioms AX, we call the set $\mathcal{F}_{\mathrm{IC}}[\mathrm{AX}] := \{F \mid F$ satisfies all axioms in AX and the domain of F is $\mathrm{Mod}(\mathrm{IC})^{\mathcal{N}}$ for some $\mathcal{N}\}$ a *class of aggregation procedures*. The idea of safety for constraints is then defined as follows.

DEFINITION 4. *An integrity constraint* IC *is* safe for the class $\mathcal{F}_{\mathrm{IC}}[\mathrm{AX}]$ *if and only if for all $F \in \mathcal{F}_{\mathrm{IC}}[\mathrm{AX}]$, we have $F(\boldsymbol{B}) \models \mathrm{IC}$ for all inputs $\boldsymbol{B} \in \mathrm{Mod}(\mathrm{IC})^{\mathcal{N}}$ for some $\mathcal{N}$.*

Let a *literal* be either a variable p or its negation $\neg p$. A *term* D is a conjunction of distinct literals and $D - D'$ is the *subtraction operation* over terms, resulting in all the literals of D that are not in D'. A term D is an *implicant* of φ if and only if $D \models \varphi$. We follow the presentation of Marchi et al. [25], and give the following definition:

DEFINITION 5. *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, $(D - \{L\}) \not\models \varphi$.

Observe that any constraint IC can be rewritten as a conjunction of negations of prime implicants of $\neg$IC [26]: in the following we assume that constraints have this syntactical form. The following definitions reinterpret for integrity constraints some known *agenda properties* of formula based judgment aggregation, by making use of the concept of prime implicants. Let $\mathbb{P}_\varphi$ be the set of variables used in φ.

DEFINITION 6. *A constraint* IC *has the k*-median property (*k*MP) *if and only if any prime implicant D of $\neg$IC is such that $|\mathbb{P}_D| \leq k$.*

A constraint IC *has the* simplified median property (SMP) *if and only if any prime implicant D of $\neg$IC is such that $|\mathbb{P}_D| = 2$ and for $p, q \in \mathbb{P}_D$ we have that $\neg L_p \wedge \neg L_q$ is also a prime implicant of* $\neg$IC.

For $k = 2$ we speak of the *median-property* (MP). Observe that if IC $= \top$ we do not have any prime implicant of $\neg$IC, which means that the issues are all independent from one another — a condition known as *syntactic simplified median property* (SSMP) in the literature.

5.2 Safety in DL-PA

We start by proving a lemma which characterises by a DL-PA formula the valuations where some prime implicant of formula φ is true. Let thus φ be a formula and let $P \subseteq \mathbb{P}_\varphi$ be a subset of the variables of φ. Given a valuation v, let $P_v := \bigwedge_{1\leq k\leq |P|} L_k$ be the term such that for all $p_k \in P$:

$$L_k := \begin{cases} p_k & \text{if } v \models p_k \\ \neg p_k & \text{otherwise} \end{cases}$$

LEMMA 1. *Let v be a valuation, φ a formula and $P \subseteq \mathbb{P}_\varphi$ a subset of the variables in φ. Term P_v is a prime implicant of φ if and only if $v \models \mathsf{PI}(P, \varphi)$, where*

$$\mathsf{PI}(P, \varphi) := [\mathsf{flip}^1(P)]\langle \mathsf{flip}^{\geq 0}(\mathbb{P}_\varphi \setminus P)\rangle \neg\varphi \wedge [\mathsf{flip}^{\geq 0}(\mathbb{P}_\varphi \setminus P)]\varphi.$$

PROOF. For the left-to-right direction, let P_v be a prime implicant of φ and suppose, for reductio, that $v \models \neg\mathsf{PI}(P, \varphi)$. Observe that, if $\langle \mathsf{flip}^1(P)\rangle[\mathsf{flip}^{\geq 0}(\mathbb{P}_\varphi \setminus P)]\varphi$ is the case, we would have a contradiction with condition (ii) of Definition 5 (P_v is not prime). In fact, we would have that some variable $p_k \in P_v$ corresponding to a literal L_k in P_v would make $(D - \{L_k\}) \models \varphi$ hold. On the other hand, if $\langle \mathsf{flip}^{\geq 0}(\mathbb{P}_\varphi \setminus P)\rangle \neg\varphi$ is the case, we would have a contradiction with condition (i) of Definition 5 (P_v is not an implicant of φ). In fact, there would be some valuation v' where the literals in P_v are true and yet $\neg\varphi$ holds. Therefore, we have $v \models \mathsf{PI}(P, \varphi)$.

We prove the right-to-left direction by contraposition. Suppose P_v is not a prime implicant of φ. By Definition 5 this means that either P_v is not an implicant of φ, which would imply that $v \not\models [\mathsf{flip}^{\geq 0}(\mathbb{P}_\varphi \setminus P)]\varphi$, or that P_v is not prime, which would imply that $v \not\models [\mathsf{flip}^1(P)]\langle\mathsf{flip}^{\geq 0}(\mathbb{P}_\varphi \setminus P)\rangle\neg\varphi$. Thus, in both cases we can conclude that $v \not\models \mathsf{PI}(P, \varphi)$. □

PROPOSITION 7. *Constraint* IC *has the* kMP *if and only if* $\models \neg\mathrm{IC} \to \bigvee_{\substack{P\subseteq\mathbb{P}_{\mathrm{IC}}\\|P|\leq k}} \mathsf{PI}(P, \neg\mathrm{IC})$.

PROOF. For the left-to-right direction, assume that IC has the kMP and suppose, for reductio, that there is some v such that $v \models \neg\mathrm{IC}$ and $v \models \bigwedge_{\substack{P\subseteq\mathbb{P}_{\mathrm{IC}}\\|P|\leq k}} \neg\mathsf{PI}(P, \neg\mathrm{IC})$. Since $v \not\models \mathrm{IC}$ and IC can be written as a conjunction of negations of prime implicants of $\neg$IC, we know that there must be some prime implicant D of $\neg$IC such that $v \models D$ and that $|\mathbb{P}_D| \leq k$. By Lemma 1 we thus get that $v \models \mathsf{PI}(\mathbb{P}_D, \neg\mathrm{IC})$, which contradicts $v \models \bigwedge_{\substack{P\subseteq\mathbb{P}_{\mathrm{IC}}\\|P|\leq k}} \neg\mathsf{PI}(P, \neg\mathrm{IC})$.

We prove the right-to-left direction by contraposition. Suppose IC does not have the kMP: hence, there is some prime implicant D of $\neg$IC such that $|\mathbb{P}_D| \geq k+1$. We now provide a valuation v such that $v \models \neg\mathrm{IC}$ and $v \not\models \bigvee_{\substack{P\subseteq\mathbb{P}_{\mathrm{IC}}\\|P|\leq k}} \mathsf{PI}(P, \neg\mathrm{IC})$. Consider valuation v such that $v \models D$ and for all other prime implicants D' of $\neg$IC, we have $v \not\models D'$ (such a valuation always exists). Since $v \models D$, we get by Definition 5 that $v \models \neg\mathrm{IC}$. Suppose there was some other term D' such that $v \models \mathsf{PI}(\mathbb{P}_{D'}, \neg\mathrm{IC})$, $|D'| \leq k$ and $v \models D'$: by Lemma 1 this would imply that D' is a prime implicant of $\neg$IC, contradicting our choice of valuation. □

PROPOSITION 8. *Constraint* IC *has the* SMP *if and only if* $\models \neg\mathrm{IC} \to \bigvee_{p_i,p_k\in\mathbb{P}_{\mathrm{IC}}} \big(\mathsf{PI}(\{p_i, p_k\}, \neg\mathrm{IC}) \land [\mathsf{flip}(p_i); \mathsf{flip}(p_k)]\ \mathsf{PI}(\{p_i, p_k\}, \neg\mathrm{IC})\big)$.

PROOF. For the left-to-right direction, assume that IC has the SMP and consider an arbitrary valuation v such that $v \models \neg\mathrm{IC}$. Suppose, for reductio, that the consequent does not hold. Hence, either there is no prime implicant of $\neg$IC of size 2 or there is one, but the negation of its literals is not a prime implicant of $\neg$IC. In both cases, this would contradict our assumption that IC has the SMP.

For the right-to-left direction, assume that IC has not the SMP. This means that either it has not the MP, or it has the MP but there is a prime implicant of $\neg$IC such that its negated literals are not also a prime implicant of $\neg$IC. In the first case, we would get by Proposition 7 that there is a valuation v such that $v \not\models \mathsf{PI}(\{p_i, p_k\}, \neg\mathrm{IC})$ thus making the consequent false. In the second case, we would have that $v \not\models [\mathsf{flip}(p_i); \mathsf{flip}(p_k)]\mathsf{PI}(\{p_i, p_k\}, \neg\mathrm{IC})$, thus falsifying the consequent again. Therefore, $\not\models \bigvee_{p_i,p_k\in\mathbb{P}_{\mathrm{IC}}} \big(\mathsf{PI}(\{p_i, p_k\}, \neg\mathrm{IC}) \land [\mathsf{flip}(p_i); \mathsf{flip}(p_k)]\mathsf{PI}(\{p_i, p_k\}, \neg\mathrm{IC})\big)$. □

6. CONCLUSIONS AND FUTURE WORK

In this paper we showed how to translate the framework of judgment aggregation, in its model of binary aggregation with integrity constraints, into the propositional dynamic logic DL-PA. The key ideas of our translation consisted in turning profiles of individual ballots into a specific type of valuation, and aggregation rules into DL-PA programs modifying the truth value of a set of variables for the outcome. We then provided compact representations for a number of aggregation rules from the literature. Next, we focused on the axiomatic characterisation of aggregation rules as well as the safety of the agenda problem in DL-PA.

Our work paves the way to further investigations from both a computational and an agent-based perspective. First of all, a significant characteristics of DL-PA is that this modal logic is grounded on propositional logic. In other words, this means that there exists a procedure to translate any DL-PA formula as a formula of propositional logic [10, 3]. Therefore, thanks to the work presented here we now have a chain of translations from aggregation problems to DL-PA, and from DL-PA to propositional logic — which yields us the tool of SAT solvers to enhance research in judgment aggregation. As we anticipated in the introduction, this computer-based approach has already been proven successful in Computational Social Choice [31, 15, 5].

In the second place, our translation allows us to also model the *winner determination* problem for aggregation rules [13, 23]: i.e., computing the outcome of a rule on a given profile. The formulation of this problem differs between resolute and irresolute aggregators. As an example, for a resolute aggregation rule F the problem is usually formulated as checking for each issue j whether $F(\boldsymbol{B})_j = 1$ for profile $\boldsymbol{B}$. This would hence translate into DL-PA as checking whether $v_{\boldsymbol{B}} \models [\mathsf{f}(\mathbb{B}^{n,m})]p_j$ is the case. Following our previous consideration, it would then be possible to translate instances of such formula into propositional logic.

As far as the questions related to agent-based reasoning are concerned, we propose a possible generalisation and a direction for future research. It is easily seen that our framework could be generalised to deal with a setting where agents are allowed to abstain on the issues. Specifically, it would be sufficient to consider an additional set of propositional variables for the profile, to keep track of the issues on which the agents abstain. This would hence result in two copies of the profile to fully cover the information about abstentions and individual opinions.

Finally, it would be interesting to provide a DL-PA treatment of strategy-proofness for aggregation rules. Given that we have a way to store the values of propositional variables, to compute the Hamming distance and to use counters, incorporating this kind of study in our setting would be fairly straightforward — of course, in case we assume Hamming distance type of preferences over possible outcomes for the agents. We thus leave these questions for future investigation.

REFERENCES

[1] 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.

[2] K. J. Arrow. *Social Choice and Individual Values*. John Wiley & Sons, 2nd edition, 1963.

[3] 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.

[4] P. Blackburn, M. de Rijke, and Y. Venema. *Modal Logic*. Cambridge University Press, 2001.

[5] F. Brandt and C. Geist. Finding strategyproof social choice functions via SAT solving. *Journal of Artificial Intelligence Research (JAIR)*, 55:565–602, 2016.

[6] F. Brandt, C. Geist, and D. Peters. Optimal bounds for the no-show paradox via SAT solving. In *Proceedings of the 2016 International Conference on Autonomous Agents & Multiagent Systems (AAMAS)*, 2016.

[7] 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.

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

[9] 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.

[10] J. van Eijck. Making things happen. *Studia Logica*, 66(1):41–58, 2000.

[11] 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.

[12] 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.

[13] U. Endriss, U. Grandi, and D. Porello. Complexity of judgment aggregation. *Journal of Artificial Intelligence Research*, 45:481–514, 2012.

[14] 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 *Proceedings of the 12th International Workshop on Multi-Agent Systems and Agent-Based Simulation*, 2011.

[15] 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.

[16] U. Grandi and U. Endriss. Binary aggregation with integrity constraints. In *Proceedings of the 22nd International Joint Conference on Artificial Intelligence, (IJCAI)*, 2011.

[17] U. Grandi and U. Endriss. First-order logic formalisation of impossibility theorems in preference aggregation. *Journal of Philosophical Logic*, 42(4):595–618, 2013.

[18] D. Grossi and G. Pigozzi. Judgment aggregation: a primer. *Synthesis Lectures on Artificial Intelligence and Machine Learning*, 8(2):1–151, 2014.

[19] 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.

[20] A. Herzig, E. Lorini, F. Moisan, and N. Troquard. A dynamic logic of normative systems. In *Proceedings of the 22nd International Joint Conference on Artificial Intelligence (IJCAI)*, 2011.

[21] J. G. Kemeny. Mathematics without numbers. *Daedalus*, 88(4):577–591, 1959.

[22] L. A. Kornhauser and L. G. Sager. The one and the many: adjudication in collegial courts. *California Law Review*, 81(1):1–59, 1993.

[23] 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.

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

[25] 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.

[26] P. Marquis. Consequence finding algorithms. In *Handbook of Defeasible Reasoning and Uncertainty Management Systems*, pages 41–145. Springer, 2000.

[27] T. Nipkow. Social choice theory in HOL: Arrow and Gibbard-Satterthwaite. *Journal of Automated Reasoning*, 43(3):289–304, 2009.

[28] A. Novaro. Judgment aggregation in dynamic logic of propositional assignments. Master's thesis, University of Amsterdam, 2016.

[29] T. Perkov. Natural deduction for modal logic of judgment aggregation. *Journal of Logic Language and Information*, pages 1–20, 2016.

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

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

[32] N. Troquard, W. van der Hoek, and M. Wooldridge. Reasoning about social choice functions. *Journal of Philosophical Logic*, 40(4):473–498, 2011.

[33] F. Wiedijk. Arrow's impossibility theorem. *Formalized Mathematics*, 15(4):171–174, 2007.