BULLETIN OF THE
HELLENIC MATHEMATICAL SOCIETY
Volume 63, 2019 (97–135)

ALGORITHMICALLY EFFICIENT SYNTACTIC CHARACTERIZATION OF POSSIBILITY DOMAINS

JOSEP DÍAZ[1], LEFTERIS KIROUSIS[1,2], SOFIA KOKONEZI[2], AND JOHN LIVIERATOS[2]

Abstract. In the field of Judgment Aggregation, a *domain*, that is a subset of a Cartesian power of $\{0,1\}$, is considered to reflect abstract rationality restrictions on vectors of two-valued judgments on a number of issues. We are interested in the ways we can aggregate the positions of a set of individuals, whose positions over each issue form vectors of the domain, by means of unanimous (idempotent) functions, whose output is again an element of the domain. Such functions are called *non-dictatorial*, when their output is not simply the positions of a single individual. Here, we consider domains admitting various kinds of non-dictatorial aggregators, which reflect various properties of majority aggregation: (locally) non-dictatorial, generalized dictatorships, anonymous, monotone, StrongDem and systematic. We show that interesting and, in some sense, democratic voting schemes are always provided by domains that can be described by propositional formulas of specific syntactic types we define. Furthermore, we show that we can efficiently recognize such formulas and that, given a domain, we can both efficiently check if it is described by such a formula and, in case it is, construct it. Our results fall in the realm of classical results concerning the syntactic characterization of domains with specific closure properties, like domains closed under logical AND which are the models of Horn formulas. The techniques we use to obtain our results draw from judgment aggregation as well as propositional logic and universal algebra.

1. Introduction

We call *domain* any arbitrary subset of a Cartesian power $\{0,1\}^n$ ($n \geq 1$) when we think of it as the set of yes/no ballots, or accept/reject judgment vectors on n issues that are "rational", in the sense manifested by being a member of the subset. A domain D has a non-dictatorial aggregator if for some $k \geq 1$ there is a unanimous (idempotent) function $F : D^k \to D$ that is not a projection function. Such domains are called *possibility domains*. The theory of judgment aggregation was put in this abstract framework by Wilson [36], and then elaborated by several others (see e.g. the work by Dietrich [9] and Dokow and Holzman [11, 12]). It can be trivially shown that non-dictatorial aggregators always exist unless we demand that F is defined on an issue by issue fashion (see next section for formal definitions). Such aggregators are called Independent of Irrelevant Alternatives (IIA). In this work aggregators are assumed to be IIA.

It is a well known fact from elementary Propositional Logic that for every subset D of $\{0,1\}^n$, $n \geq 1$, i.e. for every domain, there is a Boolean formula in Conjunctive Normal Form (CNF) whose set of satisfying truth assignments, or models, denoted by $\mathrm{Mod}(\phi)$, is equal to D (see e.g. Enderton [14, Theorem 15B]). Zanuttini and Hébrard [38] give an algorithm that finds such a

Date: Submitted Sep. 3, 2019; accepted Dec. 12, 2019.

2010 *Mathematics Subject Classification.* 91B14, 68Q25.

Key words and phrases. Collective Decision Making, Computational Social Choice, Judgment Aggregation, Logical Relations, Algorithm Complexity.

The first two authors' research was partially supported by TIN2017-86727-C2-1-R, GRAMM.

The research of the second author was carried out while visiting the Computer Science Department of the Universitat Politècnica de Catalunya.

formula and runs in polynomial-time with respect to the size of the representation of D as input. Following Grandi and Endriss [18], we call such a ϕ an *integrity constraint* and think of it as expressing the "rationality" of D (the term comes from databases, see e.g. [13]).

We prove that a domain is a possibility domain, if and only if it admits an integrity constraint of a certain syntactic form to be precisely defined, which we call a *possibility* integrity constraint. Very roughly, possibility integrity constraints are formulas that belong to one of three types, the first two of which correspond to "easy" cases of possibility domains: (i) formulas whose variables can be partitioned into two non-empty subsets so that no clause contains variables from both sets that we call *separable* and (ii) formulas whose clauses are exclusive OR's of their literals (*affine* formulas). The most interesting third type is comprised of formulas such that if we change the logical sign of some of their variables, we get formulas that have a Horn part and whose remaining clauses contain only negative occurrences of the variables in the Horn part. We call such formulas *renamable partially Horn*, whereas we call *partially Horn*[1] the formulas that belong to the third type without having to rename any variables. Furthermore, we show that the unified framework of Zanuttini and Hébrard [38] for producing formulas of a specific type that describe a given domain, and which entails the notion of prime formulas (i.e. formulas that we cannot further simplify its clauses; see Definition 7) works also in the case of possibility integrity constraints. Actually, in addition to the syntactical characterization of possibility domains, we give two algorithms: the first on input a formula decides whether it is a possibility integrity constraint in time linear in the length of the formula (notice that the definition of possibility integrity constraint entails searching over all subsets of variables of the formula); the second on input a domain D halts in time polynomial in the size of D and either decides that D is not a possibility domain or otherwise returns a possibility integrity constraint that describes D. It should be noted that the satisfiability problem remains NP-complete even when restricted to formulas that are partially Horn. On the other hand, in Computational Social Choice, domains are considered to be non-empty (see paragraph preceding Example 3).

We then consider *local possibility domains*, that is, domains admitting IIA aggregators whose components are all different than any projection function. Such aggregators are called *locally non-dictatorial* (see [27]). Local non-dictatorial domains were introduced in [22] as *uniform possibility domains* (the definition entails also non-Boolean domains). We show that local possibility domains are described by formulas we call *local possibility integrity constraints* and again, we provide a linear algorithm that checks if a formula is a local possibility integrity constraint and a polynomial algorithm that checks if a domain is a local possibility one and, in case it is, constructs a local possibility integrity constraint that describes it. As a corollary we also obtain a simpler characterization of local possibility domains in the Boolean framework (see Corollary 2).

Our approach is purely syntactic, in the sense that we construct and identify formulas of a specific syntactic type, that describe (local) possibility domains. Of course, it can well be the case that a formula is not a (local) possibility integrity constraint, but nevertheless describes a (local) possibility domain. This "semantic" approach is not in the scope of this paper. However, we discuss some complexity bounds for these problems, proven by Kirousis et al. [21].

There are various notions of non-dictatorial aggregation, apart from the above, that have been introduced in the field of Aggregation Theory. First, we consider domains that admit aggregators which are not *generalized dictatorships*. A k-ary aggregator is a generalized dictatorship that, on

[1]A weaker notion of Horn formulas has appeared before in the work of Yamasaki and Doshita [37]; however our notion is incomparable with theirs, in the sense that the class of partially Horn formulas in neither a subset nor a superset (nor equal) to the class $\mathbb{S}_0$ they define.

input any k vectors from a domain D, always returns one of those vectors as its output. These aggregators are a natural generalization of the notion of dictatorial aggregators, in the sense that they select a possibly different "dictator" for each set of k feasible voting patters, instead of a single global one. They where introduced by Cariani et al. [4] as *rolling dictatorships*, under the stronger requirement that the above property holds for any k vectors of $\{0,1\}^n$. In that framework, Grandi and Endriss [18] showed that generalized dictatorships are exactly those functions that are aggregators for every Boolean domain. In this work, we show that domains admitting aggregators which are not generalized dictatorships are exactly the possibility domains (apart from some trivial cases), and are thus described by possibility integrity constraints.

Then, we consider *anonymous* aggregators, which are aggregators that are not affected by permutations of their input and *monotone* aggregators, which are aggregators that do not change their output if a voter changes his choice in order to agree with it. Both of these types of aggregators have been extensively studied in the bibliography (see e.g. [11, 12, 17, 18, 22, 26, 27]), as they have properties that are considered important, if not necessary, for democratic voting schemes. Here, we show that domains admitting anonymous aggregators are described by local possibility integrity constraints, while domains admitting non-dictatorial monotone aggregators by separable or renamable partially Horn formulas.

We also consider another kind of non-dictatorial aggregator that shares an important property of majority voting. *StrongDem* aggregators are k-ary aggregators that, on every issue, we can fix the votes of any $k-1$ voters in such a way that the k-th voter cannot change the outcome of the aggregation procedure. These aggregators were introduced by Szegedy and Xu [33]. Here, we show that domains admitting StrongDem aggregators are described by a subclass of local possibility integrity constraints.

Finally, we consider aggregators satisfying *systematicity* (see List [26]). Aggregators are called systematic when they aggregate every issue with a common rule. This property has appeared also as *(issue-)neutrality* in the bibliography (see e.g. Grandi and Endriss [18] and Nehring and Puppe [27]). By viewing a domain D as a Boolean relation, systematic aggregators are in fact *polymorphisms* of D (see Section 5.4). Polymorphisms are a very important and well studied tool of Universal Algebra. Apart from showing, using known results, that domains admitting systematic aggregators are described by specific types of local possibility integrity constraints, we also examine how our previous results concerning the various kinds of non-dictatorial voting schemes are affected by requiring that the aggregators also satisfy systematicity.

It should be mentioned that, to prove our results, we use either implicitly or explicitly what is known as *Post's lattice*. Post [29] completely classified *clones* of Boolean functions, that is sets of Boolean functions containing all the projections, that are closed under *superposition* (see Section 5 for formal definitions). Here, we take advantage of the fact that, when aggregators of a domain are defined in an issue-by-issue fashion, the set of their components on any given issue forms a clone.

As examples of similar classical results in the theory of Boolean relations, we mention that domains component-wise closed under $\wedge$ or $\vee$ have been identified with the class of domains that are models of Horn or dual-Horn formulas respectively (see Dechter and Pearl [6]). Also it is known that a domain is component-wise closed under the ternary sum mod 2 if and only if it is the set of models of a formula that is a conjunction of subformulas each of which is an exclusive OR (the term "ternary" refers to the number of bits to be summed). Finally, a domain is closed under the ternary majority operator if and only it is the set of models of a CNF formula where each clause has at most two literals. The latter two results are due to Schaefer [31]. The ternary majority operator is the ternary Boolean function that returns 1 on input three bits if and only

if at least two of them are 1. It is also known that the respective formulas for each case can be found in polynomial time with respect to the size of D (see Zanuttini and Hébrard [38]).

Our results can be interpreted as verifying that various kinds of non-dictatorial voting schemes can always be generated by integrity constraints that have a specific, easily recognizable syntactic form. This can prove valuable for applications in the field of judgment aggregation, where relations are frequently encountered in compact form, as the sets of models of integrity constraints. As examples of such applications, we mention the work of Pigozzi [28] in avoiding the *discursive dilemma*, the characterization of *safe agendas* by Grandi and Endriss [17], that of Endriss and de Haan [15] concerning the *winner determination problem* and the work of Endriss et al. [16] in succinct representations of domains. Our proofs draw from results in judgment aggregation theory as well as from results about propositional formulas and logical relations. Specifically, as stepping stones for our algorithmic syntactic characterization we use three results. First, a theorem implicit in Dokow and Holzman [11] stating that a domain is a possibility domain if and only if it either admits a binary (of arity 2) non-dictatorial aggregator or it is component-wise closed under the ternary direct sum. This result was generalized by Kirousis et al. [22] for domains in the non-Boolean framework. Second, a characterization of local possibility domains proven by Kirousis et al. in [22]. Lastly, the "unified framework for structure identification" by Zanuttini and Hébrard [38] (see next section for definitions).

Relation to the conference version: A preliminary version of this paper appeared in the Proceedings of the 46th International Colloquium on Automata, Languages and Programming (ICALP 2019) [8]. The present full version, in addition to detailed proofs and several improvements in the presentation, contains the results about generalized dictatorships, anonymous, monotone and StrongDem aggregators, and the discussion about systematicity, which were not included in the conference version.

2. Preliminaries

We first give the notation and basic definitions from Propositional Logic and judgment aggregation theory that we will use.

Let $V = \{x_1, \ldots, x_n\}$ be a set of Boolean variables. A literal is either a variable $x \in V$ (positive literal) or a negation $\neg x$ of it (negative literal). A clause is a disjunction $(l_{i_1} \vee \cdots \vee l_{i_k})$ of literals from different variables. A propositional formula ϕ (or just a "formula", without the specification "propositional", if clear from the context) in Conjunctive Normal Form (CNF) is a conjunction of clauses. A formula is called k-CNF if every clause of it contains exactly k literals. A (truth) assignment to the variables is an assignment of either 0 or 1 to each of the variables. We denote by $a(x)$ the value of x under the assignment a. Truth assignments will be identified with elements of $\{0,1\}^n$, or n-sequences of bits. The truth value of a formula for an assignment is computed by the usual rules that apply to logical connectives. The set of satisfying (returning the value 1) truth assignments, or models, of a formula, is denoted by $\mathrm{Mod}(\phi)$. In what follows, we will assume, except if specifically noted, that n denotes the number of variables of a formula ϕ and m the number of its clauses.

We say that a variable x appears *positively* (resp. *negatively*) in a clause C, if x (resp. $\neg x$) is a literal of C. A variable $x \in V$ is positively (resp. negatively) *pure* if it has only positive (resp. negative) appearances in ϕ.

A Horn clause is a clause with at most one positive literal. A dual Horn is a clause with at most one negative literal. A formula that contains only Horn (dual Horn) clauses is called Horn (dual Horn, respectively). Generalizing the notion of a clause, we will also call clauses sets of

literals connected with exclusive OR (or direct sum), the logical connective that corresponds to summation in $\{0,1\}$ mod 2. Formulas obtained by considering a conjunction of such clauses are called affine. Finally, bijunctive are called the formulas whose clauses, in inclusive disjunctive form, i.e. whose literals are connected with the logical OR, have at most two literals. A domain $D \subseteq \{0,1\}^n$ is called Horn, dual Horn, affine or bijunctive respectively, if there is a Horn, dual Horn, affine or bijunctive formula ϕ of n variables such that $\mathrm{Mod}(\phi) = D$. In the previous section, we mentioned efficient solutions to classical syntactic characterization problems for classes of relations with given closure properties on one hand, and formulas of the syntactic forms mentioned above on the other.

We have presented the above notions and results without many details, as they are all classical results. For the notions that follow we give more detailed definitions and examples. The first one, as far as we can tell, dates back to 1978 (see Lewis [25]).

Definition 1. A formula ϕ whose variables are among the elements of the set $V = \{x_1, \ldots, x_n\}$ is called *renamable Horn*, if there is a subset $V_0 \subseteq V$ so that if we replace every appearance of every negated literal l from V_0 with the corresponding positive one and vice versa, ϕ is transformed to a Horn formula.

The process of replacing the literals of some variables with their logical opposite ones, is called a *renaming* of the variables of ϕ. It is straightforward to see that any dual-Horn formula is renamable Horn (just rename all its variables).

Example 1. Consider the formulas $\phi_1 = (x_1 \vee x_2 \vee \neg x_3) \wedge (\neg x_1 \vee x_3 \vee x_4) \wedge (\neg x_2 \vee x_3 \vee \neg x_5)$ and $\phi_2 = (\neg x_1 \vee x_2 \vee x_3 \vee x_4) \wedge (x_1 \vee \neg x_2 \vee \neg x_3) \wedge (x_4 \vee x_5)$, defined over $V = \{x_1, x_2, x_3, x_4, x_5\}$.

The formula ϕ_1 is renamable Horn. To see this, let $V_0 = \{x_1, x_2, x_3, x_4\}$. By renaming these variables, we get the Horn formula $\phi_1^* = (\neg x_1 \vee \neg x_2 \vee x_3) \wedge (x_1 \vee \neg x_3 \vee \neg x_4) \wedge (x_2 \vee \neg x_3 \vee \neg x_5)$. On the other hand, it is easy to check that ϕ_2 cannot be transformed into a Horn formula for any subset of V, since for the first clause to become Horn, at least two variables from $\{x_2, x_3, x_4\}$ have to be renamed, making the second clause not Horn. ◇

It turns out that whether a formula is renamable Horn can be checked in linear time. There are several algorithms that do that in the literature, with the one of del Val [7] being a relatively recent such example. The original non-linear one was given by Lewis [25]. By the construction presented there, it is easy to observe that a bijunctive formula is renamable Horn if and only if it is satisfiable. Indeed, let α be an assignment satisfying ϕ and rename all the variables $x \in V$ such that $\alpha(x) = 1$. Then, every clause of ϕ either has a positive literal that is renamed, or a negative one that is not renamed.

We now proceed with introducing several syntactic types of formulas:

Definition 2. A formula is called *separable* if its variables can be partitioned into two non-empty disjoint subsets so that no clause of it contains literals from both subsets.

Example 2. The formula $\phi_3 = (\neg x_1 \vee x_2 \vee x_3) \wedge (x_1 \vee \neg x_2 \vee \neg x_3) \wedge (x_4 \vee x_5)$ is separable. Indeed, for the partition $V_1 = \{x_1, x_2, x_3\}$, $V_2 = \{x_4, x_5\}$ of V, we have that no clause of ϕ_3 contains variables from both subsets of the partition. On the other hand, there is no such partition of V for neither ϕ_1 nor ϕ_2 of the previous example. ◇

The fact that separable formulas can be recognized in linear time is relatively straightforward (see Proposition 1 in Section 3).

Remark 1. *The notion of separability in Definition 2 is a purely syntactic property. Lang et al. [24] define a semantic version of separability, which, in our framework, reads as follows.*

Let $A \subseteq V$ and ϕ be a formula on V. Define ϕ_A to be the projection *of ϕ to the variables of A, i.e. ϕ with all appearances of variables not in A deleted. ϕ is called* semantically *separable if there is a partition (A, B) of V such that $Mod(\phi) = Mod(\phi_A) \times Mod(\phi_B)$. Such a partition (A, B) is called an* independent partition *of V.*

Obviously, a separable formula satisfies semantic separability. On the other hand, consider that formula:

$$\psi = (x \vee \neg y \vee z) \wedge (\neg x \vee y \vee z) \wedge (x \vee \neg y \vee \neg z) \wedge (\neg x \vee y \vee \neg z).$$

Clearly, this is not a separable formula in the sense of Definition 2. On the other hand, for the partition $(\{x, y\}, \{z\})$ of its set of variables, it holds that:

$$Mod(\psi) = \{(0, 0), (1, 1)\} \times \{0, 1\} = Mod(\psi_{\{x,y\}}) \times Mod(\psi_{\{z\}}),$$

and thus ψ is semantically separable. ◇

We now introduce the following notions:

Definition 3. A formula ϕ is called *partially Horn* if there is a nonempty subset $V_0 \subseteq V$ such that (i) the clauses containing only variables from V_0 are Horn and (ii) the variables of V_0 appear only negatively (if at all) in a clause containing also variables not in V_0.

If a formula ϕ is partially Horn, then any non-empty subset $V_0 \subseteq V$ that satisfies the requirements of Definition 3 will be called an *admissible set of variables*. Easily, if both V_0, $V_1 \subseteq V$ are admissible sets of variables, then so is $V_0 \cup V_1$. Also the Horn clauses that contain variables only from V_0 will be called *admissible clauses* (the set of admissible clauses might be empty). A Horn clause with a variable in $V \setminus V_0$ will be called *inadmissible* (the reason for the possible existence of such clauses will be made clear in the following example).

Notice that a Horn formula is, trivially, partially Horn too, as is a formula that contains at least one negative pure literal. It immediately follows that the satisfiability problem remains NP-complete even when restricted to partially Horn formulas (just add a dummy negative pure literal). In Computational Social Choice though, domains are considered to be non-empty as a non-degeneracy condition. Actually, it is usually assumed that the projection of a domain to any one of the n issues is the set $\{0, 1\}$.

Example 3. We first examine the formulas of the previous examples. ϕ_1 is partially Horn, since it contains the negative pure literal $\neg x_5$. The Horn formula ϕ_1^* is also trivially partially Horn. On the other hand, ϕ_2 and ϕ_3 are not, since for every possible $V_0 \subseteq \{x_1, x_2, x_3, x_4, x_5\}$, we either get non-Horn clauses containing variables only from V_0, or variables of V_0 that appear positively in inadmissible clauses.

The formula $\phi_4 = (x_1 \vee \neg x_2) \wedge (\neg x_1 \vee x_2) \wedge (\neg x_2 \vee \neg x_3) \wedge (\neg x_1 \vee x_3 \vee x_4)$ is partially Horn. Its first three clauses are Horn, though the third has to be put in every inadmissible set, since x_3 appears positively in the fourth clause which is not Horn. The first two clauses though constitute an admissible set of Horn clauses. Finally, $\phi_5 = (x_1 \vee \neg x_2) \wedge (x_2 \vee \neg x_3) \wedge (\neg x_1 \vee x_3 \vee x_4)$ is not partially Horn. Indeed, since all its variables appear positively in some clause, we need at least one clause to be admissible. The first two clauses of ϕ_5 are Horn, but we will show that they both have to be included in an inadmissible set. Indeed, the second has to belong to every inadmissible set since x_3 appears positively in the third, not Horn, clause. Furthermore, x_2 appears positively in the second clause, which we just showed to belong to every inadmissible set. Thus, the first clause also has to be included in every inadmissible set, and therefore ϕ_5 is not partially Horn. ◇

Accordingly to the case of renamable Horn formulas, we define:

Definition 4. A formula is called *renamable partially Horn* if some of its variables can be renamed (in the sense of Definition 1) so that it becomes partially Horn.

Observe that any Horn, renamable horn or partially Horn formula is trivially renamable partially Horn. Also, a formula with at least one pure positive literal is renamable partially Horn, since by renaming the corresponding variable, we get a formula with a pure negative literal.

Example 4. All formulas of the previous examples are renamable partially Horn: ϕ_1^*, ϕ_1 and ϕ_4 correspond to the trivial cases we discussed above, whereas ϕ_2, ϕ_3 and ϕ_5 all contain the pure positive literal x_4.

Lastly, we examine two more formulas: $\phi_6 = (\neg x_1 \vee x_2 \vee x_3 \vee x_4) \wedge (x_1 \vee \neg x_2 \vee \neg x_3) \wedge (\neg x_4 \vee x_5)$ is easily not partially Horn, but by renaming x_4 and x_5, we obtain the partially Horn formula $\phi_6^* = (\neg x_1 \vee x_2 \vee x_3 \vee \neg x_4) \wedge (x_1 \vee \neg x_2 \vee \neg x_3) \wedge (x_4 \vee \neg x_5)$, where $V_0 = \{x_4, x_5\}$ is the set of admissible variables. On the other hand, the formula $\phi_7 = (\neg x_1 \vee x_2 \vee x_3) \wedge (x_1 \vee \neg x_2 \vee \neg x_3)$ is not renamable partially Horn. Indeed, whichever variables we rename, we end up with one Horn and one non-Horn clause, with at least one variable of the Horn clause appearing positively in the non-Horn clause. ◇

We prove, by Theorem 3 in Section 3 that checking whether a formula is renamable partially Horn can be done in linear time in the length of the formula.

Remark 2. *Let ϕ be a renamable partially Horn formula, and let ϕ^* be a partially Horn formula obtained by renaming some of the variables of ϕ, with V_0 being the admissible set of variables. Let also $\mathcal{C}_0$ be an admissible set of Horn clauses in ϕ^*. We can assume that only variables of V_0 have been renamed, since the other variables are not involved in the definition of being partially Horn. Also, we can assume that a Horn clause of ϕ^* whose variables appear only in clauses in $\mathcal{C}_0$ belongs to $\mathcal{C}_0$. Indeed, if not, we can add it to $\mathcal{C}_0$.*

Another technical point is that, contrary to the case of partially Horn formulas, it can be the case that V_0 and V_1 are distinct subsets of variables of a formula which, when renamed, make the formula partially Horn, but $V_0 \cup V_1$ is not. For example, let

$$\psi = (\neg x \vee \neg y) \wedge (x \vee z) \wedge (y \vee \neg z),$$

which is easily not partially Horn. Let $V_0 = \{x\}$ and $V_1 = \{y, z\}$. By renaming these subsets of variables, we obtain the two Horn formulas:

$$\psi^*_{\{x\}} = (x \vee \neg y) \wedge (\neg x \vee z) \wedge (y \vee \neg z)$$
$$\psi^*_{\{y,z\}} = (\neg x \vee y) \wedge (x \vee \neg z) \wedge (\neg y \vee z),$$

whereas, by renaming $V_0 \cup V_1 = \{x, y, z\}$, we obtain the formula:

$$\psi^* = (x \vee y) \wedge (\neg x \vee \neg z) \wedge (\neg y \vee z),$$

which is not partially Horn. ◇

Definition 5. A formula is called a *possibility integrity constraint* if it is either separable, or renamable partially Horn or affine.

From the above and the fact that checking whether a formula is affine is easy we get Theorem 4 in Section 3, which states that checking whether a formula is a possibility integrity constraint can be done in polynomial time in the size of the formula.

Now, let V, V' be two disjoint sets of variables. By further generalizing the notion of a clause of a CNF formula, we say that a (V, V')-*generalized clause* is a clause of the form:

$$(l_1 \vee \cdots \vee l_s \vee (l_{s+1} \oplus \cdots \oplus l_t)),$$

where the literal l_j corresponds to variable v_j, $j = 1, \ldots, t$, $s < t$, $v_1, \ldots, v_s \in V$ and $v_{s+1}, \ldots, v_t \in V'$. Such a clause is falsified by exactly those assignments that falsify every literal l_i, $i = 1, \ldots, s$ and satisfy an even number of literals l_j, $j = s+1, \ldots, t$. An affine clause is trivially a (V, V')-generalized clause, where all its literals correspond to variables from V'. Consider now the following syntactic type of formulas.

Definition 6. A formula ϕ is a *local possibility integrity constraint (lpic)* if there are two *disjoint* subsets $V_0, V_1 \subseteq V$, with $V_0 \cup V_1 = V$, such that:

(1) by renaming some variables of V_0, we obtain a partially Horn formula ϕ^*, whose set of admissible variables is V_0 and
(2) the clauses containing variables from V_1 are (V_0, V_1)-generalized clauses.

Example 5. Easily, every (renamable) Horn or affine formula is an lpic. On the other hand, consider the following formula:

$$\phi_8 = (\neg x_1 \vee \neg x_2) \wedge (x_1 \vee x_2 \vee (x_3 \oplus x_4 \oplus x_5)).$$

ϕ_8 is not an lpic, since V_0 must contain both x_1 and x_2, and under any renaming. either at least one of them will appear positively in the second, non-admissible clause, or the first will seize being a Horn clause. ◇

By Definition 6, an lpic ϕ over V, where $V_0 \neq \emptyset$, is a renamable partially Horn formula. Otherwise, if $V_0 = \emptyset$, ϕ is affine.

To end this preliminary discussion about propositional formulas, we consider *prime* formulas. Given a clause C of a formula ϕ, we say that a *sub-clause* of C is any non-empty clause created by deleting at least one literal of C. In Quine [30] and Zanuttini and Hébrard [38], we find the following definitions:

Definition 7. A clause C of a formula ϕ is a *prime implicate* of ϕ if no sub-clause of C is logically implied by ϕ. Furthermore, ϕ is prime if all its clauses are prime implicates of it.

In Section 4, we use this notion in order to efficiently construct formulas whose sets of models is a (local) possibility domain.

We now come to some notions from Social Choice Theory (for an introduction, see e.g. List [26]). In the sequel, we will deal with k sequences of n-bit-vectors, each of which belongs to a fixed domain $D \subseteq \{0,1\}^n$. It is convenient to present such sequences with an $k \times n$ matrix $x^i_j, i = 1, \ldots, k, j = 1, \ldots, n$ with bits as entries. The rows of this matrix are denoted by $x^i, i = 1, \ldots, k$ and the columns by $x_j, j = 1, \ldots, n$. Each row represents a row-vector of 0/1 decisions on n issues by one of k individuals. Each column represents the column-vector of the positions of all k individuals on a particular issue.

In the sequel, we will assume that all domains $D \subseteq \{0,1\}^n$ are *non-degenerate*, i.e. for any $j \in \{1, \ldots, n\}$, it holds that $D_j = \{0,1\}$, where D_j denotes the projection of D to the j-th coordinate. This is a common assumption in Social Choice Theory, which reflects the idea that voting is nonsensical when there is only one option. Consequently, we will also assume that the formulas we consider have non-degenerate domains too.

A domain $D \subseteq \{0,1\}^n$ is said to have a k-ary (of arity k) unanimous aggregator if there exists a sequence of n k-ary Boolean functions $(f_1, \ldots, f_n)$, $f_j : \{0,1\}^k \to \{0,1\}, j = 1, \ldots, n$ such that

- all f_j are unanimous, i.e if $b_1 = \cdots = b_k$ are equal bits, then
$$f_j(b_1, \ldots, b_k) = b_1 = \cdots = b_k, \text{ and}$$
- if for a matrix $(x^i_j)_{i,j}$ that represents the opinions of k individuals on n issues we have that the row-vectors $x^i \in D$ for all $i = 1, \ldots, k$, then
$$(f_1(x_1), \ldots, f_n(x_n)) \in D.$$

Notice that in the second bullet above, the f_j's are applied to column-vectors, which have dimension k. The f_j's are called the *components* of the aggregator $(f_1, \ldots, f_n)$. Intuitively, an aggregator is a sequence of functions that when applied onto some rational opinion vectors of k individuals on n issues, in a issue-by-issue fashion, they return a row-vector that is still rational. From now on, we will refer to unanimous aggregators, simply as aggregators. We will also sometimes say that F is an aggregator, meaning that F is a sequence of n functions $(f_1, \ldots, f_n)$ as above.

The fact that we defined aggregators as n-tuples of functions, means that we require that they satisfy a property called *Independence of Irrelevant Alternatives*, in the sense that the way we aggregate an issue, is independent of the way we aggregate the rest. An aggregator $F = (f_1, \ldots, f_n)$ for a domain D is called *dictatorial* if there is a $d = 1. \ldots, k$ such that $f_1 = \cdots = f_n = \mathrm{pr}^k_d$, where $\mathrm{pr}^k_d : (b_1, \ldots, b_k) \mapsto b_d$ is the k-ary projection function on the d'th coordinate.

A k-ary aggregator is called a *projection* aggregator if each of its components is a projection function pr^k_d, for some $d = 1, \ldots, m$. Notice that it is conceivable to have non-dictatorial aggregators that are projection aggregators. Also, we call an aggregator $F = (f_1, \ldots, f_n)$, where $f_1 = f_2 = \ldots = f_n := f$ *systematic*. Notationally, we write $\bar{f}$ to denote the n-tuple $(f, \ldots, f)$, where the number of components of $\bar{f}$ always corresponds to the arity of the given domain.

The only unary (of arity 1) unanimous function is the *identity* function $\mathrm{id} : \{0,1\} \mapsto \{0,1\}$, where $\mathrm{id}(x) = x$, $x \in \{0,1\}$. It follows that there is only one unary aggregator, which is trivially dictatorial, as all of its components equal pr^1_1. Thus, from now on, we will always assume that all the functions we consider have arity at least 2. A binary (of arity 2) Boolean function $f : \{0,1\}^2 \to \{0,1\}$ is called *symmetric* if for all pairs of bits b_1, b_2, we have that $f(b_1, b_2) = f(b_2, b_1)$. A binary aggregator is called symmetric if all its components are symmetric. Let us mention here the easily to check fact that the only unanimous binary functions are the $\wedge$, $\vee$ and the two projection functions $\mathrm{pr}^2_1, \mathrm{pr}^2_2$. Of those four, only the first two are symmetric.

Definition 8. A domain D is called a possibility domain if it has a (unanimous) non-dictatorial aggregator of some arity.

Notice that the search space for such an aggregator is large, as the arity is not restricted. However, from [22, Theorem 3.7] (a result that follows from Dokow and Holzman [11], but without being explicitly mentioned there), we can easily get that:

Theorem 1 (Dokow and Holzman [11])**.** *A domain D is a possibility domain if and only if it admits either: (i) a non-dictatorial binary projection aggregator or (ii) a non-projection binary aggregator (i.e. at least one symmetric component) or (iii) a ternary aggregator all components of which are the binary addition* mod 2*.*

Example 6. Theorem 1 directly implies that the truth set of any affine formula is a possibility domain. Consider now the formula $\phi_7 = (\neg x_1 \vee x_2 \vee x_3) \wedge (x_1 \vee \neg x_2 \vee \neg x_3)$ of Example 4. It holds that:
$$\mathrm{Mod}(\phi_7) = \{0,1\}^3 \setminus \{(1,0,0), (0,1,1)\}.$$

By checking all 4^3 different triples of binary unanimous operators and since $\mathrm{Mod}(\phi_7)$ is not affine, one can see that $\mathrm{Mod}(\phi_7)$ is an impossibility domain. On the other hand, let

$$\phi_9 := (\neg x_1 \vee x_2 \vee x_3) \wedge (x_1 \vee \neg x_2 \vee \neg x_3) \wedge (\neg x_4 \vee x_5 \vee x_6) \wedge (x_4 \vee \neg x_5 \vee \neg x_6).$$

Then, we have that:

$$\mathrm{Mod}(\phi_9) = \mathrm{Mod}(\phi_7) \times \mathrm{Mod}(\phi_7),$$

which is a possibility domain, since every Cartesian product is (see Kirousis et al. [22, Example 2.1]). Finally, for:

$$\phi_6 = (\neg x_1 \vee x_2 \vee x_3 \vee x_4) \wedge (x_1 \vee \neg x_2 \vee \neg x_3) \wedge (\neg x_4 \vee x_5),$$

of Example 4 we have that:

$$\mathrm{Mod}(\phi_6) = (\mathrm{Mod}(\phi_7) \times \{(0,0),(0,1)\}) \cup \Big((\{0,1\}^3 \setminus \{(1,0,0)\}) \times \{(1,1)\}\Big)$$

is a possibility domain, as it admits the binary aggregator $(\mathrm{pr}_1^2, \mathrm{pr}_1^2, \mathrm{pr}_1^2, \vee, \vee)$. ◇

Nehring and Puppe [27] defined a type of non-dictatorial aggregators they called *locally non-dictatorial*. A k-ary aggregator $(f_1, \ldots, f_n)$ is locally non-dictatorial if $f_j \neq \mathrm{pr}_d^k$, for all $d \in \{1, \ldots, k\}$ and $j = 1, \ldots, n$.

Definition 9. D is a *local possibility domain (lpd)* if it admits a locally non-dictatorial aggregator.

Kirousis et al. [22] introduced these domains as *uniform* non-dictatorial domains, both in the Boolean and non-Boolean framework and provided a characterization for them. Consider the following ternary operators on $\{0,1\}$: (i) $\wedge^{(3)}(x,y,z) := \wedge(\wedge(x,y),z))$ (resp. for $\vee^{(3)}$), (ii) maj, where $\mathrm{maj}(x,y,z) = 1$ if and only if *at least* two elements of its input are 1 and (iii) $\oplus$, where $\oplus(x,y,z) = 1$ if an only if *exactly one or all* of the elements of its input are equal to 1.

Theorem 2 (Kirousis et al. [22], Theorem 5.5)**.** *$D \subseteq \{0,1\}^n$ is a local possibility domain if and only if it admits a ternary aggregator $(f_1, \ldots, f_n)$ such that $f_j \in \{\wedge^{(3)}, \vee^{(3)}, \mathrm{maj}, \oplus\}$, for $j = 1, \ldots, n$.*

Example 7. Neither $\mathrm{Mod}(\phi_6)$ nor $\mathrm{Mod}(\phi_7)$ of Example 4, nor $\mathrm{Mod}(\phi_9)$ of Example 6 are local possibility domains, since every aggregator they admit has components that are projection functions. On the other hand, for:

$$\phi_{10} = (\neg x_1 \vee x_2 \vee x_3) \wedge (x_1 \vee x_2 \vee \neg x_3),$$

we have that:

$$\mathrm{Mod}(\phi_{10}) = \{0,1\}^3 \setminus \{(0,0,1),(1,0,0)\},$$

that is a possibility domain, since it admits $(\wedge, \vee, \wedge)$. ◇

3. Identifying (local) possibility integrity constraints

In this section, we show that identifying (local) possibility integrity constraints can be done in time linear in the length of the input formula. By Definitions 5 and 6, it suffices to show that for separable formulas, renamable partially Horn formulas and lpic's, since the corresponding problem for affine formulas is trivial.

In all that follows, we assume that we have a set of variables $V := \{x_1, \ldots, x_n\}$ and a formula ϕ defined on V that is a conjunction of m clauses $C_1, \ldots, C_m$, where $C_j = (l_{j_1}, \ldots, l_{j_{k_j}})$, $j = 1, \ldots, n$, and l_{j_s} is a positive or negative literal of x_{j_s}, $s = 1, \ldots, k_j$. We denote the set of variables corresponding to the literals of a clause C_j by $\mathrm{vbl}(C_j)$.

We begin with the result for separable formulas:

Proposition 1. *There is an algorithm that, on input a formula ϕ, halts in time linear in the length of ϕ and either returns that the formula is not separable, or alternatively produces a* partition *of V in two non-empty and disjoint subsets $V_1, V_2 \subseteq V$, such that no clause of ϕ contains variables from both V_1 and V_2.*

Proof. We construct a graph on the variables of ϕ, where two such vertices are connected if they appear consecutively in a common clause of ϕ. The result is then obtained by showing that ϕ is separable if and only if G is not connected.

Suppose the variables of each clause are ordered by the indices of their corresponding literals in the clause. Thus, we say that x_{j_s}, x_{j_t} are consecutive in C_j, if $t = s + 1$, $s = 1, \ldots, k_j - 1$.

Given a formula ϕ, construct an undirected graph $G = (V, E)$, where :

- V is the set of variables of ϕ, and
- two vertices are connected if they appear consecutively in a common clause of ϕ.

It is easy to see that each clause C_j, where $\text{vbl}(C_j) = \{x_{j_1}, \ldots, x_{j_{k_j}}\}$ induces the path $\{x_{j_1}, \ldots, x_{j_{k_j}}\}$ in G.

For the proof of linearity, notice that the set of edges can be constructed in linear time with respect to the length of ϕ, since we simply need to read once each clause of ϕ and connect its consecutive vertices. Also, there are standard techniques to check connectivity in linear time in the number of edges (e.g. by a *depth-first search* algorithm).

The correctness of the algorithm is derived by noticing that two connected vertices of G cannot be separated in ϕ. Indeed, consider a path $P := \{x_r, \ldots, x_s\}$ in G (this need not be a path induced by a clause). Then, each couple x_t, x_{t+1} of vertices in P belongs in a common clause of ϕ, $t = r, \ldots, s - 1$. Thus, ϕ is separable if and only if G is not connected. □

To deal with renamable partially Horn formulas, we will start with Lewis' idea [25] of creating, for a formula ϕ, a 2SAT formula ϕ' whose satisfiability is equivalent to ϕ being renamable Horn. However, here we need to (i) look for a renaming that might transform only some clauses into Horn and (ii) deal with inadmissible Horn clauses, since such clauses can cause other Horn clauses to become inadmissible too.

Proposition 2. *For every formula ϕ, there is a formula ϕ' such that ϕ is renamable partially Horn if and only if ϕ' is satisfiable.*

Before delving into the proof, we introduce some notation. Assume that after a renaming of some of the variables in V, we get the partially Horn formula ϕ^*, with V_0 being the admissible set of variables. Let $\mathcal{C}_0$ be an admissible set of clauses for ϕ^*. We assume below that only a subset $V^* \subseteq V_0$ has been renamed and that all Horn clauses of ϕ^* with variables exclusively from V_0 belong to $\mathcal{C}_0$ (see Remark 2). Also, let $V_1 := V \setminus V_0$. The clauses of ϕ^*, which are in a one to one correspondence with those of ϕ, are denoted by $C_1^*, \ldots, C_m^*$, where C_j^* corresponds to C_j, $j = 1, \ldots, m$.

Proof. For each variable $x \in V$, we introduce a new variable x'. Intuitively, setting $x = 1$ means that x is renamed (and therefore $x \in V^*$), whereas setting $x' = 1$ means that x is in V_0, but is not renamed. Finally we set both x and x' equal to 0 in case x is not in V_0. Obviously, we should not not allow the assignment $x = x' = 1$ (a variable in V_0 cannot be renamed and not renamed). Let $\overline{V} = V \cup \{x' \mid x \in V\}$.

Consider the formula ϕ' below, with variable set $\overline{V}$. For each clause C of ϕ and for each $x \in \text{vbl}(C)$: if x appears positively in C, introduce the literals x and $\neg x'$ and if it appears negatively, the literals $\neg x$ and x'. ϕ' is the conjunction of the following clauses: for each clause

C of ϕ and for each two variables $x, y \in \mathrm{vbl}(C)$, ϕ' contains the disjunctions of the positive with the negative literals introduced above. Thus:

(i) if C contains the literals x, y, then ϕ' contains the clauses $(x \vee \neg y')$ and $(\neg x' \vee y)$,
(ii) if C contains the literals $x, \neg y$, then ϕ' contains the clauses $(x \vee \neg y)$ and $(\neg x' \vee y')$ (accordingly if C contains $\neg x, y$) and
(iii) if C contains the literals $\neg x, \neg y$, then ϕ' contains the clauses $(\neg x \vee y')$ and $(x' \vee \neg y)$.

Finally, we add the following clauses to ϕ':

(iv) $(\neg x_i \vee \neg x_i')$, $i = 1, \ldots, n$ and
(v) $\bigvee_{x \in \overline{V}} x$.

The clauses of items (i)–(iv) correspond to the intuition we explained in the beginning. For example, consider the case where a clause C_j of ϕ has the literals $x, \neg y$. If we add x to V_0 without renaming it, we should not rename y, since we would have two positive literals in a clause of $\mathcal{C}_0$. Also, we should not add the latter to V_1, since we would have a variable of V_0 appearing positively in a clause containing a variable of V_1. Thus, we have that $x' \to y'$, which is expressed by the equivalent clause $(\neg x' \vee y')$ of item (ii). The clauses of item (iv) exclude the assignment $x = x' = 1$ for any $x \in V$. Finally, since we want V_0 to be non-empty, we need at least one variable of $\overline{V}$ to be set to 1.

To complete the proof of Proposition 2, we now proceed as follows.

($\Rightarrow$) First, suppose ϕ is renamable partially Horn. Let V_0, V_1, V^* and $\overline{V}$ as above. Suppose also that $V_0 \neq \emptyset$.

Set $a = (a_1, \ldots, a_{2n})$ to be the following assignment of values to the variables of $\overline{V}$:

$$a(x) = \begin{cases} 1, & \text{if } x \in V^*, \\ 0, & \text{else,} \end{cases} \text{ and } a(x') = \begin{cases} 0, & \text{if } x \in V^* \cup V_1, \\ 1, & \text{else,} \end{cases}$$

for all $x \in V$. To obtain a contradiction, suppose a does not satisfy ϕ'.

Obviously, the clauses of items (iv) and (v) above are satisfied, by the definition of a and the fact that V_0 is not empty.

Now, consider the remaining clauses of items (i)–(iii) above and suppose for example that some $(\neg x \vee y')$ is not satisfied. By the definition of ϕ', there exists a clause C which, before the renaming takes place, contains the literals $\neg x, \neg y$ (see item (iii)). Since the clause is not satisfied, $a(x) = 1$ and $a(y') = 0$, which in turn means that $x \in V^*$ and $y \in V^* \cup V_1$. If $y \in V_1$, C^* contains, after the renaming, a variable in V_1 and a positive appearance of a variable in V_0. If $y \in V^*$, C^* contains two positive literals of variables in V_0. Contradiction. The remaining cases can be proven analogously and are left to the reader.

($\Leftarrow$) Suppose now that $a = (a_1, \ldots, a_{2n})$ is an assignment of values to the variables of $\overline{V}$ that satisfies ϕ'. We define the following subsets of $\overline{V}$:

- $V^* = \{x \mid a(x) = 1\}$,
- $V_0 = \{x \mid a(x) = 1 \text{ or } a(x') = 1\}$ and
- $V_1 = \{x \mid a(x) = a(x') = 0\}$.

Let ϕ^* be the formula obtained by ϕ, after renaming the variables of V^*.

Obviously, V_0 is not empty, since a satisfies the clause of item (v).

Suppose that a clause C^*, containing only variables from V_0, is not Horn. Then, C^* contains two positive literals x, y. If $x, y \in V_0 \setminus V^*$, then neither variable was renamed and thus C also contains the literals x, y. This means that, by item (i) above, ϕ' contains the clauses $(x \vee \neg y')$ and $(\neg x' \vee y)$. Now, since $x, y \in V_0 \setminus V^*$, it holds that $a(x) = a_(y) = 0$ and $a(x') = a(y') = 1$. Then,

a does not satisfy these two clauses. Contradiction. In the same way, we obtain contradictions in cases that at least one of x and y is in V^*.

Finally, suppose that there is a variable $x \in V_0$ that appears positively in a clause $C^* \notin \mathcal{C}_0$. Let $y \in V_1$ be a variable in C^* (there is at least one such variable, lest $C^* \in \mathcal{C}_0$). Suppose also that y appears positively in C^*.

Assume $x \in V^*$. Then, C contains the literals $\neg x, y$. Thus, by item (ii), ϕ' contains the clause $(\neg x \vee y)$. Furthermore, since $x \in V^*$, $a(x) = 1$ and since $y \in V_1$, $a(y) = 0$. Thus the above clause is not satisfied. Contradiction. In the same way, we obtain contradictions in all the remaining cases. □

To compute ϕ' from ϕ, one would need quadratic time in the length of ϕ. Thus, we introduce the following linear algorithm that decides if a formula ϕ is renamable partially Horn, by tying a property of a graph constructed based on ϕ, with the satisfiability of ϕ'.

Theorem 3. *There is an algorithm that, on input a formula ϕ, halts in time linear in the length of ϕ and either returns that ϕ is not renamable partially Horn or alternatively produces a subset $V^* \subseteq V$ such that the formula ϕ^* obtained from ϕ by renaming the literals of variables in V^* is partially Horn.*

To prove Theorem 3, we define a *directed bipartite* graph G, i.e. a directed graph whose set of vertices is partitioned in two sets such that no vertices belonging in the same part are adjacent. Then, by computing its *strongly connected components (scc)*, i.e. its maximal sets of vertices such that every two of them are connected by a directed path, we show that at least one of them is not *bad* (does not contain a pair of vertices we will specify below) *if and only if* ϕ is renamable partially Horn.

For a directed graph G, we will denote a directed edge from a vertex u to a vertex v by (u, v). A (directed) path from u to v, containing the vertices $u = u_0, \ldots, u_s = v$, will be denoted by $(u, u_1, \ldots, u_{s-1}, v)$ and its existence by $u \to v$. If both $u \to v$ and $v \to u$ exist, we will sometimes write $u \leftrightarrow v$.

Recall that given a directed graph $G = (V, E)$, there are known algorithms that can compute the scc of G in time $O(|V| + |E|)$, where $|V|$ denotes the number of vertices of G and $|E|$ that of its edges (see e.g. Tarjan [35]). By identifying the vertices of each scc, we obtain a *directed acyclic graph (DAG)*. An ordering $(u_1, \ldots, u_n)$ of the vertices of a graph is called *topological* if there are no edges (u_i, u_j) such that $i \geq j$, for all $i, j \in \{1, \ldots, n\}$.

Proof. Given ϕ defined on V, whose set of clauses is $\mathcal{C}$ and let again $\overline{V} = V \cup \{x' \mid x \in V\}$. We define the graph G, with vertex set $\overline{V} \cup \mathcal{C}$ and edge set E such that, if $C \in \mathcal{C}$ and $x \in \mathrm{vbl}(C)$, then:

- if x appears *negatively* in C, E contains (x, C) and (C, x'),
- if x appears *positively* in C, E contains (x', C) and (C, x) and
- E contains no other edges.

Intuitively, if $x, y \in \overline{V}$, then a path (x, C, y) corresponds to the clause $x \to y$ which is logically equivalent to $(\neg x \vee y)$. The intuition behind x and x' is exactly the same as in Proposition 2. We will thus show that the bipartite graph G defined above, contains all the necessary information to decide if ϕ' is satisfiable, with the difference that G can be constructed in time linear in the length of the input formula. To that end, observe that to construct G, we need constant-time access to the formula ϕ. Indeed, one only needs to read ϕ once, from left to right, constructing one vertex for each clause and connecting the vertices of the clause with it in the way described above.

There is a slight technicality arising here since, by the construction above, G always contains either the path (x, C, x') or (x', C, x), for any clause C and $x \in \text{vbl}(C)$, whereas neither $(\neg x \vee x')$ nor $(x \vee \neg x')$ are ever clauses of ϕ'. Thus, from now on, we will assume that no path can contain the vertices x, C and x' or x', C and x *consecutively*, for any clause C and $x \in \text{vbl}(C)$.

Observe that by construction, (i) (x, C) or (C, x) is an edge of G if and only if $x \in \text{vbl}(C)$, $x \in \overline{V}$ and (ii) (x, C) (resp. (x', C)) is an edge of G if and only if (C, x') (resp.(C, x)) is one too.

We now prove several claims concerning the structure of G. To make notation less cumbersome, assume that for an $x \in V$, $x'' = x$. Consider the formula ϕ' of Proposition 2.

Claim 3.1. *Let $x, y \in \overline{V}$. For $z_1, \ldots, z_k \in \overline{V}$ and $C_1, \ldots, C_{k+1} \in \mathcal{C}$, it holds that*

$$(x, C_1, z_1, C_2, \ldots, z_k, C_{k+1}, y)$$

is a path of G if and only if $(\neg x \vee z_1)$, $(\neg z_i \vee z_{i+1})$, $i = 1, \ldots, k-1$ and $(\neg z_k \vee y)$ are all clauses of ϕ'.

Proof of Claim. Can be easily proved inductively to the length of the path, by recalling that a path (u, C, v) corresponds to the clause $(\neg u \vee v)$, for all $u, v \in \overline{V}$ and $C \in \mathcal{C}$. □

Claim 3.2. *Let $x, y \in \overline{V}$. If $x \to y$, then $y' \to x'$.*

Proof of Claim. Since $x \to y$, there exist $z_1, \ldots, z_k \in \overline{V}$ and $C_1, \ldots, C_{k+1} \in \mathcal{C}$, such that $(x, C_1, z_1, C_2, \ldots, z_k, C_{k+1}, y)$ is a path of G. By Claim 3.1, $(\neg x \vee z_1)$, $(\neg z_i \vee z_{i+1})$, $i = 1, \ldots, k-1$ and $(\neg z_k \vee y)$ are all clauses of ϕ'. By Proposition 2, so do $(\neg y' \vee z'_k)$, $(\neg z'_{i+1} \vee z'_i)$, $i = 1, \ldots, k-1$ and $(\neg z'_1 \vee x')$ and the result is obtained by using Claim 3.1 again. □

We can obtain the scc's of G using a variation of a *depth-first search (DFS)* algorithm, that, whenever it goes from a vertex x (resp. x') to a vertex C, it cannot then go to x' (resp. x) at the next step. Since the algorithm runs in time linear in the number of the vertices and the edges of G, it is also linear in the length of the input formula ϕ.

Let S be a scc of G. We say that S is *bad*, if, for some $x \in V$, S contains both x and x'. We can decide if each of the scc's is bad or not again in time linear in the length of the input formula.

Claim 3.3. *Let S be a bad scc of G and $y \in \overline{V}$ be a vertex of S. Then, y' is in S.*

Proof of Claim. Since S is bad, there exist two vertices x, x' of $\overline{V}$ in S. If $x = y$ we have nothing to prove, so we assume that $x \neq y$. Then, we have that $y \to x$, which, by Claim 3.2 implies that $x' \to y'$. Since $x \to x'$, we get that $y \to y'$. That $y' \to y$ can be proven analogously. □.

Let the scc's of G, in *reverse topological order*, be $S_1, \ldots . S_t$. We describe a process of assigning values to the variables of $\overline{V}$:

(1) Set every variable that appears in a bad scc of G to 0.
(2) For each $j = 1, \ldots, t$ assign value 1 to every variable of S_j that has not already received one (if S_j is bad no such variable exists). If some $x \in \overline{V}$ of S_j takes value 1, then assign value 0 to x'.
(3) Let a be the resulting assignment to the variables of $\overline{V}$.

Now, the last claim we prove is the following:

Claim 3.4. *There is at least one variable $z \in \overline{V}$ that does not appear in a bad scc of G if and only if ϕ' is satisfiable.*

Proof of Claim. ($\Rightarrow$) We prove that every clause of type (i)–(v) is satisfied. First, by the construction of a, every clause $\neg x_i \vee \neg x'_i$, $i = 1, \ldots, n$, of type (iv) is obviously satisfied. Also, since by the hypothesis, z is not in a bad scc, it holds, by step 2 above, that either z or z' are set to 1. Thus, the clause $\bigvee_{x \in \overline{V}} x$ of type (v) is also satisfied.

Now, suppose some clause $(x \vee \neg y')$ (type (i)) of ϕ' is not satisfied. Then $a(x) = 0$ and $a(y') = 1$. Furthermore, there is a vertex C such that (y', C) and (C, x) are edges of G. By the construction of G, (x', C) and (C, y) are also edges of G.

Since $a(x) = 0$, it must hold either that x is in a bad scc of G, or that $a(x') = 1$. In the former case, we have that $x \to x'$, which, together with (y', C, x) and (x', C, y) gives us that $y' \to y$. Contradiction, since then $a(y')$ should be 0. In the latter case, we have that there are two scc's S_p, S_r of G such that $x \in S_p$, $x' \in S_r$ and $p < r$ in their topological order. But then, there is some $q : p \leq q \leq r$ such that C in S_q. Now, if $p = q$, we obtain a contradiction due to the existence of (x', C), else, due to (C, x).

The proof for the rest of the clauses of types (i)–(iii) are left to the reader.

($\Leftarrow$) First, recall that for two propositional formulas ϕ, ψ, we say that ϕ *logically entails* ψ, and write $\phi \models \psi$, if any assignment that satisfies ϕ, satisfies ψ too.

Now observe that, if x, y are two vertices in $\overline{V}$ such that $x \to y$, then $\phi' \models (\neg x \vee y)$. Indeed, suppose β is an assignment of values that satisfies ϕ'. If $\beta(y) = 1$, we have nothing to prove. Thus, assume that $\beta(y) = 0$. By Claim 3.1, if $(x, C_1, z_1, C_2, z_2, \ldots, z_k, C_{k+1}, y)$ is the path $x \to y$, then $(\neg x \vee z_1)$, $(\neg z_i \vee z_{i+1})$, $i = 1, \ldots, k-1$ and $(\neg z_k \vee y)$ are all clauses of ϕ' and are thus satisfied by β. Since $\beta(y) = 0$, we have $\beta(z_k) = 0$. Continuing in this way, $\beta(z_i) = 0$, $i = 1, \ldots, k$ and thus $\beta(x) = 0$ too, which implies that $\beta(\neg x \vee y) = 1$.

Now, for the proof of the claim, suppose again that ϕ' is satisfiable, and let β be an assignment (possibly different than α) that satisfies ϕ'. Since β satisfies ϕ', it satisfies $\bigvee_{x \in \overline{V}} x$. This means that there exists some $x \in \overline{V}$ such that $\beta(x) = 1$. But β also satisfies $(\neg x \vee \neg x')$, so we get that $\beta(x') = 0$. Thus $\beta((\neg x \vee x')) = 0$, which means that ϕ' does not logically entail $\neg x \vee x'$. By the discussion above, there exists no path from x to x', so x is not in a bad scc of G. □

By Proposition 2, we have seen that ϕ is renamable partially Horn if and only if ϕ' is satisfiable. Also, in case ϕ' is satisfiable, a variable $x \in V$ is renamed if and only if $a(x) = 1$.

Thus, by the above and Claim 3.4, ϕ is renamable partially Horn if and only if there is some variable x that does not appear in a bad scc of G. Furthermore, the process described in order to obtain assignment a is linear in the length of the input formula, and a provides the information about which variables to rename. □

Because checking whether a formula is affine can be trivially done in linear time, we get:

Theorem 4. *There is an algorithm that, on input a formula ϕ, halts in linear time in the length of ϕ and either returns that ϕ is not a possibility integrity constraint, or alternatively, (i) either it returns that ϕ is affine or (ii) in case ϕ is separable, it produces two non-empty and disjoint subsets $V_1, V_2 \subseteq V$ such that no clause of ϕ contains variables from both V_1 and V_2 and (iii) in case ϕ is renamable partially Horn, it produces a subset $V^* \subseteq V$ such that the formula ϕ^* obtained from ϕ by renaming the literals of variables in V^* is partially Horn.*

We proceed by showing that we can recognize lpic's efficiently.

Theorem 5. *There is an algorithm that, on input a formula ϕ, halts in linear time in the length of ϕ and either returns that ϕ is not a local possibility constraint, or alternatively, produces the sets V_0, V_1 described in Definition 6.*

Proof. We describe the algorithm in a high level way.

(1) Read once each clause C of ϕ. If C does not contain any $\oplus$ connectives, put the variables of $\mathrm{vbl}(C)$ to V_0. Else, put the variables connected by $\oplus$ to V_1.
(2) Check the variables in V_0 and V_1. If $V_1 = V$, ϕ is affine and thus an lpic. Else, if $V_0 \cap V_1 \neq \emptyset$ or $V_0 \cup V_1 \neq V$, ϕ is not an lpic. Also, if $V_0 = V$, ϕ is an lpic if and only if it is a renamable Horn formula, something that can be checked in linear time by the algorithm of del Val [7].
(3) Assume that (V_0, V_1) is a partition of V. We use the following variation of the linear algorithm presented in Theorem 3:
   (a) Build the graph $G = (\overline{V}, E)$ in the same way and compute its scc. Let $\overline{V}_0 = V_0 \cup \{x' \mid x \in V_0\}$ and $\overline{V}_1 = V_1 \cup \{x' \mid x \in V_1\}$.
   (b) If there is any variable of $\overline{V}_0$ in a bad scc of G, G is not an lpic. Else, set all variables of $\overline{V}_1$ to 0.
   (c) Let the scc's of G, in reverse topological order, be $S_1, \ldots . S_t$. For each $j = 1, \ldots, t$ assign value 1 to every variable of $S_j \cap \overline{V}_0$ that has not already received one (if S_j is bad no such variable exists). If some $x \in \overline{V}_0$ of S_j takes value 1, then assign value 0 to x' (recall that $(x')' = x$).
(4) Let a be the resulting assignment to the variables of $\overline{V}$. By renaming all variables $x \in V_0$ such that $a(x) = 1$, we obtain a partially Horn formula, whose non-admissible clauses are (V_0, V_1)-generalized. Thus, ϕ is an lpic.

□

Remark 3. *Recall that we have assumed that all the formulas we consider have non-degenerate domains. Note that the above algorithms cannot distinguish such formulas from other formulas of the same form that have degenerate domains. An algorithm that could efficiently decide that, would effectively be (due e.g. to the syntactic form of separable formulas) an algorithm that could decide on input any given formula, which variables are satisfied by exactly one Boolean value and which admit both.*

The consequences of the existence of such an algorithm would be critical. For example, consider the known coNP*-hard problem of* unique satisfiability*, where we are interested in whether a propositional formula ϕ is satisfied by exactly one assignment of values. We could use the aforementioned algorithm for each variable of the formula consecutively. If, at any point, we find a variable that is satisfied by both* 0 *and* 1*, then ϕ has more that one satisfying assignment. Else, it doesn't.*

3.1. **Semantic equivalence.** We end this section with a sort discussion concerning formulas that are not (local) possibility integrity constraints, but are semantically equivalent to such formulas, in the sense that they have the same set of models.

Recall that the *polynomial hierarchy* consists of the complexity classes Σ_k^P, Π_k^P and Δ_k^P, $k \in \mathbb{N}$, which are recursively defined as follows:

- $\Sigma_0^P = \Sigma_0^P = \Delta_0^P = \mathsf{P}$ and
- Σ_{k+1}^P is NP with oracle Σ_k^P, Π_{k+1}^P is coNP with oracle Σ_k^P and Δ_{k+1}^P is P with oracle Σ_k^P, $k \in \mathbb{N}$,

where it can be shown that:

$$\Sigma_k^P \cup \Pi_k^P \subseteq \Delta_{k+1}^P \subseteq \Sigma_{k+1}^P \cap \Pi_{k+1}^P, \ \forall k \in \mathbb{N}.$$

We refer the interested reader to Stockmeyer's work [32] for a more in depth discussion of the subject.

In a, yet unpublished, extended version of [21], Kirousis et al. show that the problem of deciding whether the domain of a given formula is a possibility domain, is in $\Sigma_2^{\mathrm{P}} \cap \Pi_2^{\mathrm{P}}$ and that of whether it is a local possibility domain, is in Σ_2^{P} and coNP-hard. Thus, we immediately obtain the following results.

Corollary 1. *Deciding, on input ϕ, whether there exists a possibility integrity constraint ψ such that $Mod(\phi) = Mod(\psi)$ is in $\Sigma_2^{\mathrm{P}} \cap \Pi_2^{\mathrm{P}}$.*

Furthermore, deciding, on input ϕ, whether there exists an lpic ψ such that $Mod(\phi) = Mod(\psi)$ is in Σ_2^{P} and coNP-hard.

4. Syntactic Characterization of (local) possibility domains

In this section, we provide syntactic characterizations for (local) possibility domains, by proving they are the models of (local) possibility integrity constraints. Furthermore, we show that given a (local) possibility domain D, we can produce a (local) possibility integrity constraint, whose set of models is D, in time polynomial in the size of D. To obtain the characterization for possibility domains, we proceed as follows. We separately show that each type of a possibility integrity constraint of Definition 5 corresponds to one of the conditions of Theorem 1: (i) Domains admitting non-dictatorial binary projection aggregators are the sets of models of separable formulas, those admitting non-projection binary aggregators are the sets of models of renamable partially Horn formulas and (iii) affine domains are the sets of models of affine formulas. For local possibility domains, we directly show they are the models of local possibility integrity constraints.

We will need some additional notation. For a set of indices I, let $D_I := \{(a_i)_{i\in I} \mid a \in D\}$ be the projection of D to the indices of I and $D_{-I} := D_{\{1,\ldots,n\}\setminus I}$. Also, for two (partial) vectors $a = (a_1, \ldots, a_k) \in D_{\{1,\ldots,k\}}$, $k < n$ and $b = (b_1, \ldots, b_{n-k}) \in D_{\{k+1,\ldots,n\}}$, we define their *concatenation* to be the vector $ab = (a_1, \ldots, a_k, b_1, \ldots, b_{n-k})$. Finally, given two subsets $D, D' \subseteq \{0,1\}^n$, we write that $D \approx D'$ if we can obtain D by *permuting* the coordinates of D', i.e. if $D = \{(d_{j_1}, \ldots, d_{j_n}) \mid (d_1, \ldots, d_n) \in D'\}$, where $\{j_1, \ldots, j_n\} = \{1, \ldots, n\}$.

4.1. Syntactic characterizations.

We begin with characterizing the domains closed under a non-dictatorial projection aggregator as the models of separable formulas.

Proposition 3. *$D \subseteq \{0,1\}^n$ admits a binary non-dictatorial projection aggregator $(f_1, \ldots, f_n)$ if and only if there exists a separable formula ϕ whose set of models equals D.*

We will first need the following lemma:

Lemma 1. *D is closed under a binary non-dictatorial projection aggregator* if and only if *there exists a partition (I, J) of $\{1, \ldots, n\}$ such that $D \approx D_I \times D_J$.*

Proof. ($\Rightarrow$) Let $F = (f_1, \ldots, f_n)$ be a binary non-dictatorial projection aggregator for D. Assume, without loss of generality, that $f_i = \mathrm{pr}_1^2$, $i = 1, \ldots, k < n$ and $f_j = \mathrm{pr}_2^2$, $j = k+1, \ldots, n$. Let also $I := \{1, \ldots, k\}$ and $J := \{k+1, \ldots, n\}$. Since $k < n$, (I, J) is a partition of $\{1, \ldots, n\}$. To prove that $D = D_I \times D_J$, it suffices to prove that $D_I \times D_J \subseteq D$ (the reverse inclusion is always true).

Let $a \in D_I$ and $b \in D_J$. It holds that there exists an $a' \in D_I$ and a $b' \in D_J$ such that both $ab', a'b \in D$. Thus:

$$F(ab', a'b) = ab \in D,$$

since $F = (f_1, \ldots, f_n)$ is an aggregator for D, $f_i = \mathrm{pr}_1^2$, $i \in I$ and $f_j = \mathrm{pr}_2^2$, $j \in J$.

($\Leftarrow$) Suppose that $D \approx D_I \times D_J$, where I, J is a partition of $\{1, \ldots, n\}$. Assume, without loss of generality, that $I = \{1, \ldots, k\}$, $k < n$ and $J = \{k+1, \ldots, n\}$ (thus $D = D_I \times D_J$). Let also $ab', a'b \in D$, where $a, a' \in D_I$ and $b, b' \in D_J$.

Obviously, if $F = (f_1, \ldots, f_n)$ is an n-tuple of projections, such that $f_i = \mathrm{pr}_1^2$, $i \in I$ and $f_j = \mathrm{pr}_2^2$, $j \in J$, then $F(ab', a'b) = ab \in D$, since $a \in D_I$ and $b \in D_J$. Thus $F = (f_1, \ldots, f_n)$ is a non-dictatorial projection aggregator for D. □

Proof of Proposition 3. ($\Rightarrow$) Since D admits a binary non-dictatorial projection aggregator $(f_1, \ldots, f_n)$, by Lemma 1, $D \approx D_I \times D_J$, where (I, J) is a partition of $\{1, \ldots, n\}$ such that $I = \{i \mid f_i = \mathrm{pr}_1^2\}$ and $J = \{j \mid f_j = \mathrm{pr}_2^2\}$. Let ϕ_1 and ϕ_2 defined on $\{x_i \mid i \in I\}$ and $\{x_j \mid j \in J\}$ respectively, such that $\mathrm{Mod}(\phi_1) = D_I$ and $\mathrm{Mod}(\phi_2) = D_J$. Let also $\phi = \phi_1 \wedge \phi_2$. It is straightforward to observe that, since ϕ_1 and ϕ_2 contain no common variables:

$$\mathrm{Mod}(\phi) \approx \mathrm{Mod}(\phi_1) \times \mathrm{Mod}(\phi_2) = D_I \times D_J \approx D.$$

($\Leftarrow$) Assume that ϕ is separable and that $\mathrm{Mod}(\phi) = D$. Since ϕ is separable, we can find a partition (I, J) of $\{1, \ldots, n\}$, a formula ϕ_1 defined on $\{x_i \mid i \in I\}$ and a ϕ_2 defined on $\{x_j \mid j \in J\}$, such that $\phi = \phi_1 \wedge \phi_2$. Easily, it holds that:

$$\mathrm{Mod}(\phi) \approx \mathrm{Mod}(\phi_1) \times \mathrm{Mod}(\phi_2) = D_I \times D_J \approx D.$$

The required now follows by Lemma 1. □

We now turn our attention to domains closed under binary non projection aggregators.

Theorem 6. *D admits a binary aggregator $(f_1, \ldots, f_n)$ which is not a projection aggregator if and only if there exists a renamable partially Horn formula ϕ whose set of models equals D.*

We will first need two lemmas.

Lemma 2. *Suppose D admits a binary aggregator $F = (f_1, \ldots, f_n)$ such that there exists a partition (H, I, J) of $\{1, \ldots, n\}$ where f_h is symmetric for all $h \in H$, $f_i = \mathrm{pr}_s^2$, for all $i \in I$ and $f_j = \mathrm{pr}_t^2$, with $t \neq s$, for all $j \in J$. Then, D also admits a binary aggregator $G = (g_1, \ldots, g_n)$, such that $g_h = f_h$, for all $h \in H$ and $g_i = \mathrm{pr}_s^2$, for all $i \in I \cup J$.*

Proof. Without loss of generality, assume that there exist $1 \leq k < l < n$ such that $H = \{1, \ldots, k\}$, $I = \{k+1, \ldots, l\}$ and $J = \{l+1, \ldots, n\}$ and that $s = 1$ (and thus $t = 2$). It suffices to prove that, for two arbitrary vectors $a, b \in D$, $G(a, b) \in D$, where $(g_1, \ldots, g_n)$ is defined as in the statement of the lemma.

Assume that for all $i \in H$, $f_i(a_i, b_i) = c_i$. Since F is an aggregator for D, it holds that $F(a, b)$ and $F(b, a)$ are both vectors in D. By the same token, so is $F(F(a, b), F(b, a))$. The result is now obtained by noticing that:

$$F(a, b) = (c_1, \ldots, c_k, a_{k+1}, \ldots, a_l, b_{l+1}, \ldots, b_n),$$
$$F(b, a) = (c_1, \ldots, c_k, b_{k+1}, \ldots, b_l, a_{l+1}, \ldots, a_n),$$

and thus: $F(F(a, b), F(b, a)) = (c_1, \ldots, c_k, a_{k+1}, \ldots, a_n) = G(a, b)$. □

Lemma 3. *Suppose D admits a binary aggregator $(f_1, \ldots, f_n)$ such that, for some $J \subseteq \{1, \ldots, n\}$, f_j is symmetric for all $j \in J$. For each $d = (d_1, \ldots, d_n) \in D$, let $d^* = (d_1^*, \ldots, d_n^*)$ be such that:*

$$d_j^* = \begin{cases} 1 - d_j & \text{if } j \in J, \\ d_j & \text{else,} \end{cases}$$

for $j = 1, \ldots, n$ and set $D^* = \{d^* \mid d \in D\}$. Then D^* admits the binary aggregator $(g_1, \ldots, g_n)$, where: (i) $g_j = \wedge$ for all $j \in J$ such that $f_j = \vee$, (ii) $g_j = \vee$ for all $j \in J$ such that $f_j = \wedge$ and (iii) $g_j = f_j$ for the rest.

Furthermore, if there are two formulas ϕ and ϕ^* such that ϕ^* is obtained from ϕ by renaming all x_j, $j \in J$, then $D = Mod(\phi)$ if and only if $D^* = Mod(\phi^*)$.

Note that we do not assume that the set $J \subseteq \{1, \ldots, n\}$ includes every coordinate j such that f_j is symmetric.

Proof. The former statement follows from the fact that $\wedge(1 - d_j, 1 - d'_j) = 1 - \vee(d_j, d'_j)$ (resp. $\vee(1 - d_j, 1 - d'_j) = 1 - \wedge(d_j, d'_j)$), for any $d, d' \in D$. For the latter, observe that by renaming x_j, $j \in J$, in ϕ, we cause all of its literals to be satisfied by the opposite value. Thus, d^* satisfies ϕ^* if and only if d satisfies ϕ. □

For two vectors $a, b \in D$, we define $a \leq b$ to mean that if $a_i = 1$ then $b_i = 1$, for all $i \in \{1, \ldots, n\}$ and $a < b$ when $a \leq b$ and $a \neq b$.

Proof of Theorem 6. ($\Rightarrow$) We will work with the corresponding domain D^* of Lemma 3 that admits an aggregator $(g_1, \ldots, g_n)$ whose symmetric components, corresponding to the symmetric components of $(f_1, \ldots, f_n)$, are all equal to $\wedge$. Suppose that $V_0 = \{x_i \mid g_i = \wedge\}$. For D^*, we compute a formula $\phi = \phi_0 \wedge \phi_1$, where ϕ_0 is defined on the variables of V_0 and is Horn and where ϕ_1 has only negative appearances of variables of V_0. The result is then derived by renaming all the variables x_j, where j is such that $f_j = \vee$.

Let $I := \{i \mid f_i \text{ is symmetric}\}$ (by the hypothesis, $I \neq \emptyset$). Let also $J := \{j \mid f_j = \vee\}$ (J might be empty). Obviously $J \subseteq I$. For each $d = (d_1, \ldots, d_n) \in D$, let $d^* = (d_1^*, \ldots, d_n^*)$, where $d_j^* = 1 - d_j$ if $j \in J$ and $d_i^* = d_i$ else. Easily, if $D^* = \{d^* \mid d \in D\}$, by Lemma 3 it admits an aggregator $(g_1, \ldots, g_n)$ such that $g_i = \wedge$, for all $i \in I$ and $g_j = \mathrm{pr}_1^2$, $j \notin I$. Thus, there is a Horn formula ϕ_0 on $\{x_i \mid i \in I\} := V_0$, such that $\mathrm{Mod}(\phi_0) = D_I^*$.

If $I = \{1, \ldots, n\}$, we have nothing to prove. Thus, suppose, without loss of generality, that $I = \{1, \ldots, k\}$, $k < n$. For each $a = (a_1, \ldots, a_k) \in D_I^*$, let $B_a := \{b \in D_{-I}^* \mid ab \in D^*\}$ be the set containing all partial vectors that can extend a. For each $a \in D_I^*$, let ψ_a be a formula on $\{x_j \mid j \notin I\}$, such that $\mathrm{Mod}(\psi_a) = B_a$. Finally, let $I_a := \{i \in I \mid a_i = 1\}$ and define:

$$\phi_a := \left(\bigwedge_{i \in I_a} x_i \right) \rightarrow \psi_a,$$

for all $a \in D_I^*$.

Consider the formula:

$$\phi = \phi_0 \wedge \left(\bigwedge_{a \in D_I^*} \phi_a \right).$$

We will prove that ϕ is partially Horn and that $\mathrm{Mod}(\phi) = D^*$. By Lemma 3, the renamable partially Horn formula for D can be obtained by renaming in ϕ the variables x_i such that $i \in J$.

We have already argued that ϕ_0 is Horn. Also, since ϕ_a is *logically equivalent* to (has exactly the same models as):

$$\left(\bigvee_{i \in I_a} \neg x_i \right) \vee \psi_a,$$

any variable of V_0 that appears in the clauses of some ϕ_a, does so negatively. It follows that ϕ is partially Horn.

Next we show that $D^* \subseteq \mathrm{Mod}(\phi)$ and that $\mathrm{Mod}(\phi) \subseteq D^*$. For the former inclusion, let $ab \in D^*$, where $a \in D^*_I$ and $b \in B_a$. Then, it holds that a satisfies ϕ_0 and b satisfies ψ_a. Thus ab satisfies ϕ_a.

Now, let $a' \in D^*_I : a \not\geq a'$. Then, a does not satisfy $\bigwedge_{i \in I_{a'}} x_i$, since there exists some coordinate $i \in I_{a'}$ such that $a_i = 0$ and $a'_i = 1$. Thus, ab satisfies $\phi_{a'}$. Finally, let $a'' \in D^*_I : a'' < a$. Then, a satisfies $\bigwedge_{i \in I_{a''}} x_i$ and thus we must prove that b satisfies $\psi_{a''}$.

Since $a'' \in D^*_I$, there exists a $c \in D^*_{-I}$ such that $a''c \in D^*$. Then, since $(g_1, \ldots, g_n)$ is an aggregator for D^*:

$$(g_1, \ldots, g_n)(ab, a''c) = \\ (\wedge(a_1, a''_1), \ldots, \wedge(a_k, a''_k), \mathrm{pr}^2_1(b_1, c_1), \ldots, \mathrm{pr}^2_1(b_{n-k}, c_{n-k})) = a''b \in D^*,$$

since $a'' < a$. Thus, $b \in B(a'')$ and, consequently, it satisfies $\psi_{a''}$.

We will prove the opposite inclusion by showing that an assignment not in D^* cannot satisfy ϕ. Let $ab \notin D^*$. If $a \notin D^*_I$, we have nothing to prove, since a does not satisfy ϕ_0 and thus $ab \notin \mathrm{Mod}(\phi)$. So, let $a \in D^*_I$. Then, $b \notin B_a$, lest $ab \in D^*$. But then, b does not satisfy ψ_a and thus ab does not satisfy ϕ_a. Consequently, $ab \notin \mathrm{Mod}(\phi)$.

Thus, by renaming the variables x_i, $i \in J$, we produce a renamable partially Horn formula, call it ψ, such that $\mathrm{Mod}(\psi) = D$.

($\Leftarrow$) Let ψ be a renamable partially Horn formula with $\mathrm{Mod}(\psi) = D$. Let $J \subseteq \{1, \ldots, n\}$ such that, by renaming all the x_i, $i \in J$, in ψ, we obtain a partially Horn formula ϕ. Let V_0 be the set of variables such that any clause containing only variables from V_0 is Horn, and that appear only negatively in clauses that contain variables from $V \setminus V_0$. By Remark 2, we can assume that $\{x_i \mid i \in J\} \subseteq V_0$. Let also $\mathcal{C}_0$ be the set of admissible Horn clauses of ϕ.

Let again $D^* = \{d^* \mid d \in D\}$, where $d^*_j = 1 - d_j$ if $j \in J$ and $d^*_i = d_i$ else, for all $d \in D$. By Lemma 3, $\mathrm{Mod}(\phi) = D^*$. By the same Lemma, and by noticing that whichever the choice of $J \subseteq \{1, \ldots, n\}$, $(D^*)^* = D$, it suffices to prove that D^* is closed under a binary aggregator $(f_1, \ldots, f_n)$, where $f_i = \wedge$ for all i such that $x_i \in V_0$ and $f_j = \mathrm{pr}^2_1$ for the rest.

Without loss of generality, let $I = \{1, \ldots, k\}$, $k < n$ (lest we have nothing to show) be the set of indices of the variables in V_0. We need to show that if $ab, a'b' \in D$, where $a, a' \in D^*_I$ and $b, b' \in D^*_{-I}$, then $(a \wedge a')b \in D$, where $a \wedge a' = (a_1 \wedge a'_1, \ldots, a_k \wedge a'_k)$.

Let $\phi = \phi_0 \wedge \phi_1$, where ϕ_0 is the conjunction of the clauses in $\mathcal{C}_0$ and ϕ_1 the conjunction of the rest of the clauses of ϕ. By the hypothesis, ϕ_0 is Horn and thus, since a, a' satisfy ϕ_0, so does $a \wedge a'$. Now, let C_r be a clause of ϕ_1. If any literal of C_r that corresponds to a variable not in ϕ_0 is satisfied by b, we have nothing to prove. If there is no such literal, since ab satisfies C_r, it must hold that a negative literal $\bar{x}_i$, $i \in I$, is satisfied by a. Thus, $a_i = 0$, which means that $a_i \wedge a'_i = 0$ too. Consequently, C_r is satisfied by $(a \wedge a')b$. Since C_r was arbitrary, the proof is complete. □

We thus get:

Theorem 7. *D is a possibility domain if and only if there exists a possibility integrity constraint ϕ whose set of models equals D.*

Proof. ($\Rightarrow$) If D is a possibility domain, then, by Theorem 1, it either admits a non-dictatorial binary projection, or a non-projection binary aggregator or a ternary aggregator all components of which are the binary addition $\mod 2$. In the first case, by Proposition 3, D is the model set of a separable formula. In the second, by Theorem 6, it is the model set of a renamable partially

Horn formula and in the third, that of an affine formula. Thus, in all cases, D is the model set of a possibility integrity constraint.

($\Leftarrow$) Let ϕ be a possibility integrity constraint such that $\mathrm{Mod}(\phi) = D$. If ϕ is separable, then, by Proposition 3, D admits a non-dictatorial binary projection aggregator. If ϕ is renamable partially Horn, then, by Theorem 6, D admits a non-projection binary aggregator. Finally, if ϕ is affine, then D admits a ternary aggregator all components of which are the binary addition mod 2. In every case, D is a possibility domain. □

We turn now our attention to local possibility domains. Analogously to the case of possibility domains, we characterize lpd's as the sets of models of lpic's.

Theorem 8. *A domain $D \subseteq \{0,1\}^n$ is a local possibility domain if and only if there is a local possibility integrity constraint ϕ such that Mod(ϕ) = D.*

We will first need two lemmas.

Lemma 4. *Let $D \subseteq \{0,1\}^n$ and $I = \{j_1, \ldots, j_t\} \subseteq \{1, \ldots, n\}$. Then, if $F = (f_1, \ldots, f_n)$ is a k-ary aggregator for D, $(f_{j_1}, \ldots, f_{j_t})$ is a k-ary aggregator for D_I.*

Proof. Without loss of generality, assume $I = \{1, \ldots, s\}$, where $s \leq n$ and let $a^1, \ldots, a^k \in D_I$. It follows that there exist $b^1, \ldots, b^k \in D_{-I}$ such that $c^1, \ldots, c^k \in D$, where $c^i = a^i b^i$, $i = 1, \ldots k$. Since F is an aggregator for D:

$$F(c^1, \ldots, c^k) := (f_1(c_1^1, \ldots, c_1^k), \ldots, f_n(c_n^1, \ldots c_n^k)) \in D.$$

Thus, $(f_1(c_1^1, \ldots, c_1^k), \ldots, f_s(c_s^1, \ldots, c_s^k)) \in D_I$. □

Lemma 5. *Suppose that D admits a ternary aggregator $F = (f_1, \ldots, f_n)$, where $f_j \in \{\wedge^{(3)}, \oplus, \mathrm{maj}\}$, $j = 1, \ldots, n$. Then D admits a binary aggregator $G = (g_1, \ldots, g_n)$ such that $g_i = \wedge$, for all i such that $f_i = \wedge^{(3)}$, $g_j = \mathrm{pr}_2^2$, for all j such that $f_j = \oplus$ and $g_k = \mathrm{pr}_2^2$, for all k such that $f_k = \mathrm{maj}$.*

Proof. The result is immediate, by defining $G = (g_1, \ldots, g_n)$ such that:

$$g_j(x, y) = f_j(x, x, y),$$

for $j = 1, \ldots, n$. □

Proof of Theorem 8. ($\Rightarrow$) The proof will closely follow that of Theorem 6.

Since D is an lpd, by Theorem 2, there is a ternary aggregator $F = (f_1, \ldots, f_n)$ such that every component $f_j \in \{\wedge^{(3)}, \vee^{(3)}, \oplus, \mathrm{maj}\}$, $j = 1, \ldots, n$. Again, let $D^* = \{d^* \mid d \in D\}$, where $d_j^* = 1 - d_j$ if j is such that $f_j = \vee^{(3)}$, and $d_j^* = d_j$ in any other case. Thus, by Lemma 3, D^* admits a ternary aggregator $G = (g_1, \ldots, g_n)$ such that $g_j \in \{\wedge^{(3)}, \oplus, \mathrm{maj}\}$, for $j = 1, \ldots, n$. Thus, by showing that D^* is described by a lpic ϕ, we will obtain the same result for D by renaming all the variables x_j, where j is such that $f_j = \vee^{(3)}$.

Without loss of generality, assume that $I := \{i \mid g_i = \wedge^{(3)}\} = \{1, \ldots, s\}$, $J := \{j \mid g_j = \oplus\} = \{s+1, \ldots, t\}$ and $K := \{k \mid g_k = \mathrm{maj}\} = \{t+1, \ldots, n\}$, where $0 \leq s \leq t \leq n$. Since D_I^* is Horn, there is a Horn formula ϕ_0 such that $\mathrm{Mod}(D_I^*) = \phi_0$.

If $s = t = n$, we have nothing to prove. Thus, suppose $s < t \leq n$. For each $a = (a_1, \ldots, a_s) \in D_I^*$, let $B_a^1 := \{b \in D_J^* \mid ab \in D_{I \cup J}^*\}$ and $B_a^2 := \{c \in D_J^* \mid ac \in D_{I \cup K}^*\}$ be the sets of partial vectors extending a to the indices of J and K respectively.

Claim 8.1. *For each $a \in D_I^*$, B_a^1 and B_a^2 are affine and bijunctive respectively.*

Proof of Claim: We will prove the claim for B^2_a. The proof for B^1_a is the same.

Let $b^1, b^2, b^3 \in B^2_a$. Then $ab^1, ab^2, ab^3 \in D^*_{I\cup K}$. Since, by Lemma 4, $(g_1, \ldots, g_t)$ is an aggregator for $D^*_{I\cup K}$ and by the definition of G, it holds that $ab \in D^*_{I\cup K}$, where $b = \mathrm{maj}(b^1, b^2, b^3)$. Thus, $b \in B^2_a$ and the result follows. □

Thus, for each $a \in D^*_I$, there is an affine formula ψ_a and a bijunctive χ_a, such that $\mathrm{Mod}(\psi_a) = B^1_a$ and $\mathrm{Mod}(\chi_a) = B^2_a$. Let $I_a := \{i \in I \mid a_i = 1\}$ and define:

$$\phi^1_a := \left(\bigwedge_{i\in I_a} x_i\right) \to \psi_a$$

and

$$\phi^2_a := \left(\bigwedge_{i\in I_a} x_i\right) \to \chi_a,$$

for all $a \in D^*_I$.

Consider the formula:

$$\phi = \phi_0 \wedge \left(\bigwedge_{a\in D^*_I} \phi^1_a\right) \wedge \left(\bigwedge_{a\in D^*_I} \phi^2_a\right).$$

Let $V_0 = \{x_i \mid i \in I\}$, $V_1 = \{x_j \mid j \in J\}$ and $V_2 = \{x_k \mid k \in K\}$. That ϕ is partially Horn with admissible set V_0, can be seen in the same way as in Theorem 6. Now, consider ψ_a, for some $a \in D^*_I$. Since it is affine, it is of the form:

$$\psi_a = \bigwedge_{j=1}^{r} \left(l_{j_1} \oplus \cdots \oplus l_{j_t}\right),$$

where l_{j_i} are literals of variables from V_1. Thus, ϕ^1_a is equivalent to:

$$\bigwedge_{j=1}^{r} \left(\left(\bigvee_{i\in I_a} \neg x_i\right) \vee (l_{j_1} \oplus \cdots \oplus l_{j_t})\right).$$

Thus, the clauses of ϕ^1_a are (V_0, V_1)-generalized.

Now consider χ_a, for some $a \in D^*_I$. Since it is bijunctive, it is of the form:

$$\chi_a = \bigwedge_{s=1}^{t} \left(l_{s_1} \vee l_{s_2}\right),$$

where l_{s_1}, l_{s_2} are (not necessarily distinct) literals of variables from V_2. Thus, ϕ^2_a is equivalent to:

$$\bigwedge_{s=1}^{t} \left(\left(\bigvee_{i\in I_a} \neg x_i\right) \vee (l_{s_1} \vee l_{s_2})\right).$$

To prove that ϕ is an lpic, we will show that V_2 can be renamed to that $V_0 \cup V_2$ become a set of admissible variables for the renamed formula. Consider the clauses of

$$\chi := \left(\bigwedge_{a\in D^*_I} \phi^2_a\right) = \bigwedge_{a\in D^*_I} \left(\bigwedge_{s=1}^{t} \left(\left(\bigvee_{i\in I_a} \neg x_i\right) \vee (l_{s_1} \vee l_{s_2})\right)\right).$$

They are comprised of negative appearances of variables from V_0 and of at most two literals from V_2. Since V_0 and V_2 are disjoint, we can project χ to the variables of V_2, to obtain a satisfiable

bijunctive formula. Thus χ_{V_2} is renamable Horn. It is obvious that the same renaming, will result in a renaming for χ such that all its clauses contain at most one positive literal.

What remains now is to show that $\mathrm{Mod}(\phi) = D^*$. By Lemmas 2 and 5, it follows that D^* admits a binary aggregator $H = (h_1, \ldots, h_n)$ such that $h_i = \wedge$, for all $i \in I$ and $h_j = \mathrm{pr}_1^2$, for all $j \in J \cup K$. The proof now is exactly like the one of Theorem 6, by letting $B_a = \{bc \mid b \in B_a^1 \text{ and } c \in B_a^2\}$ and

$$\phi_a = \phi_a^1 \wedge \phi_a^2.$$

($\Leftarrow$) Let ψ be an lpic, with $\mathrm{Mod}(\psi) = D$. Let V_0 and V_1 be subsets of V as in Definition 6. Let also ϕ be the partially Horn formula obtained by ψ by renaming the variables of a subset $V^* \subseteq V_0$. Again, assume $D^* = \{d^* \mid d \in D\}$, where $d_j^* = 1 - d_j$ if $x_j \in V^*$ and $d_i^* = d_i$ else, for all $d \in D$. By Lemma 3, $\mathrm{Mod}(\phi) = D^*$. Thus, by Theorem 2, it suffices to prove that D^* is closed under a ternary aggregator $(f_1, \ldots, f_n)$, where $f_i \in \{\wedge^{(3)}, \mathrm{maj}, \oplus\}$ for $i = 1, \ldots, n$. We will in fact show that we do not need any maj components.

Without loss of generality, let $I = \{1, \ldots, s\}$, be the set of indices of the variables in V_0 and $J = \{s+1 \ldots, n\}$ be that of the indices of variables in V_1. We will show that if $ab, a'b', a''b'' \in D^*$, where $a, a', a'' \in D_I^*$ and $b, b', b'' \in D_J^*$, then

$$d := (\wedge^{(3)}(a, a', a''), \oplus(b, b', b'')) \in D^*.$$

Let $\phi = \phi_0 \wedge \phi_1$, where ϕ_0 is the conjunction of the clauses containing only variables from V_0 and ϕ_1 the conjunction of (V_0, V_1)-generalized clauses. By the hypothesis, ϕ_0 is Horn and thus, since a, a', a'' satisfy ϕ_0, so does $a \wedge a' \wedge a''$.

Now, let C_r be a clause of ϕ_1. Suppose that there is a literal of a variable $x_i \in V_0$ in C_r that is satisfied by a. Since ϕ is partially Horn with respect to V_0, it must hold that this literal was $\neg x_i$. This means that $a_i = 0$ and thus $\wedge^{(3)}(a_i, a_i', a_i'') = 0$. The same holds if $\neg x_i$ is satisfied by a' or a''. Thus, C_r is satisfied.

Now, suppose there is no such literal and that the and that the sub-clause of C_r obtained by deleting the variables of V_0 is:

$$C_r' = (l_1 \oplus \cdots \oplus l_z).$$

Since $ab, a'b', a''b''$ satisfy ϕ, it holds that b, b' and b'' satisfy C_r'. Since C_r' is affine, it holds that $\oplus(b, b', b'')$ satisfies it.

In all cases, we proved that d satisfies ϕ and thus the proof is complete. □

Recall that by Theorem 2, lpd's are characterized as the domains admitting an aggregator $(f_1, \ldots, f_n)$, where $f_i \in \{\wedge^{(3)}, \vee^{(3)}, \oplus, \mathrm{maj}\}$ for $i = 1, \ldots, n$. Observe though that by the proof of Theorem 8, we immediately obtain the following result.

Corollary 2. *$D \subseteq \{0,1\}^n$ is a local possibility domain if and only if it admits a ternary aggregator $(f_1, \ldots, f_n)$ such that $f_j \in \{\wedge^{(3)}, \vee^{(3)}, \oplus\}$, for $j = 1, \ldots, n$.*

The above result is a strengthening of the corresponding characterization of such domains in the non-Boolean framework, where a domain is taken to be a subset of $\Pi_{j=1}^m A_j$, with A_js being arbitrary finite sets with at least two elements. It easily follows from [22, Theorem 5.5] that in this more general framework, a domain is a a local possibility domain[2], if and only if it admits a ternary aggregator $F = (f_1, \ldots, f_n)$ such that, for all $j = 1, \ldots, n$ and all two-element subsets $B_j \subseteq D_j$, taken as $\{0, 1\}$, we have that $f_j \restriction B_j$ is one of the ternary operations $\wedge^{(3)}$, $\vee^{(3)}$, maj, $\oplus$ (to which of these four ternary operations the restriction $f_j \restriction B_j$ is equal to depends on j and B_j).

[2]In [22] the terminology "uniform possibility domain" is used.

Corollary 2 is a strengthening in the sense that in the Boolean framework, the maj case is missing. Interestingly, we were not able to prove this strengthening without the use of our syntactic characterization given in Theorem 8 above. This perhaps explains why Corollary 2 was not previously obtained, despite the fact that for the analogous case of (plain) possibility domains, both a characterization for the non-Boolean framework was given in [22, Theorem 3.1], and a strengthening for the Boolean case, where maj is missing, was already known ([12, Theorem 2.1 and Claim 3.6] or Dietrich and List [10, Theorem 2]).

4.2. **Efficient constructions.** To finish this section, we will use Zanuttini and Hébrard's "unified framework" [38]. Recall the definition of a prime formula (Def. 7) and consider the following proposition:

Proposition 4. *Let ϕ_P be a prime formula and ϕ be a formula logically equivalent to ϕ_P. Then:*

(1) if ϕ is separable, ϕ_P is also separable and

(2) if ϕ is renamable partially Horn, ϕ_P is also renamable partially Horn.

Proof. Let ϕ_P be a prime formula. Quine [30] showed that the prime implicates of ϕ_P can be obtained from any formula ϕ logically equivalent to ϕ_P, by repeated (i) resolution and (ii) omission of the clauses that have sub-clauses already created. Thus, using the procedures (i) and (ii) on ϕ, we can obtain every clause of ϕ_P.

If ϕ is separable, where $(V', V \setminus V')$ is the partition of its vertex set such that no clause contains variables from both V' and $V \setminus V'$, it is obvious that neither resolution or omission can create a clause that destroys that property. Thus, ϕ_P is separable.

Now, let ϕ be a renamable partially Horn formula where, by renaming the variables of $V^* \subseteq V$, we obtain the partially Horn formula ϕ^*, whose admissible set of variables is V_0. Let also ϕ_P^* be the formula obtained by renaming the variables of V^* in ϕ_P. Easily, ϕ_P^* is prime.

Observe that the prime implicates of a partially Horn formula, are also partially Horn. Indeed, it is not difficult to observe that neither resolution, nor omission can cause a variable to seize being admissible: suppose $x \in V_0$. Then, the only way that it can appear in an inadmissible set due to resolution is if there is an admissible Horn clause C containing $\neg x, y$, where $y \in V_0$ too and an inadmissible clause C' containing $\neg y$. But then, after using resolution, x appears negatively to the newly obtained clause. Thus, ϕ_P^* is partially Horn, which means that ϕ_P is renamable partially Horn. □

We are now ready to prove our first main result:

Theorem 9. *There is an algorithm that, on input $D \subseteq \{0,1\}^n$, halts in time $O(|D|^2 n^2)$ and either returns that D is not a possibility domain, or alternatively outputs a possibility integrity constraint ϕ, containing $O(|D|n)$ clauses, whose set of satisfying truth assignments is D.*

Proof. Given a domain D, we first use Zanuttini and Hébrard's algorithm to check if it is affine [38, Proposition 8], and if it is, produce, in time $O(|D|^2 n^2)$ an affine formula ϕ with $O(|D|n)$ clauses, such that $\mathrm{Mod}(\phi) = D$. If it isn't, we use again Zanuttini and Hébrard's algorithm [38] to produce, in time $O(|D|^2 n^2)$, a prime formula ϕ with $O(|D|n)$ clauses, such that $\mathrm{Mod}(\phi) = D$. Then, we use the linear algorithms of Proposition 1 and Theorem 3 to check if ϕ is separable or renamable partially Horn. If it is either of the two, then ϕ is a possibility integrity constraint and, by Theorem 7, D is a possibility domain. Else, by Proposition 4, D is not a possibility domain. □

We end this section by proving our second main result, that given an lpd D, we can efficiently construct an lpic ϕ such that $\mathrm{Mod}(\phi) = D$.

Theorem 10. *There is an algorithm that, on input $D \subseteq \{0,1\}^n$, halts in time $O(|D|^2 n^2)$ and either returns that D is not a local possibility domain, or alternatively outputs a local possibility integrity constraint ϕ, containing $O(|D|n)$ clauses, whose set of satisfying truth assignments is D.*

We first briefly discuss some results of Zanuttini and Hébrard. For a clause $C = l_1 \vee \ldots \vee l_t$, where l_j are literals, $j = 1, \ldots, t$, let $E(C) = l_1 \oplus \ldots \oplus l_t$. For a CNF formula $\phi = \bigwedge_{j=1}^m C_j$, let $A(\phi) = \bigwedge_{j=1}^m E(C)$. In [38, Proposition 8], it is proven that if ϕ is prime, $\mathrm{Mod}(\phi) = D$ and D is affine, then $\mathrm{Mod}(A(\phi)) = D$.

Proof of Theorem 10. Given a domain D, we first use Zanuttini and Hébrard's algorithm [38] to produce, in time $O(|D|^2 n^2)$, a prime formula ϕ with $O(|D|n)$ clauses, such that $\mathrm{Mod}(\phi) = D$. Note that at this point, ϕ does not contain any generalized clauses (see below). We then use the linear algorithm of Theorem 3 to produce a set V_0 such that ϕ is renamable partially Horn with admissible set V_0.

If $V_0 = V$ we have nothing to prove. Thus, suppose that $\phi = \phi_0 \wedge \phi_1$, where ϕ_0 contains only variables from V_0. Let ϕ_1' be the sub-formula of ϕ_1, obtained by deleting all variables of V_0 from ϕ. We then check, with Zanuttini and Hébrard's algorithm, if ϕ_1' is affine. If not, then D is not an lpd. If it is, we construct the formula $A^*(\phi_1)$ as follows. For each clause $C = (l_1 \vee \cdots \vee l_s \vee (l_{s+1} \vee \cdots \vee l_t))$, where $l_1, \ldots, l_s$ are literals of variables in V_0, let:

$$E^*(C) = (l_1 \vee \cdots \vee l_s \vee (l_{s+1} \oplus \cdots \oplus l_t))$$

and $A^*(\phi_1) = \bigwedge_{j=1}^m E^*(C_j)$. Then, the lpic that describes D is $\phi_0 \wedge A^*(\phi_1)$. □

5. Other Forms of non-Dictatorial Aggregation

In this section, we discuss four different notions of non-dictatorial aggregation procedures that have been introduced in the field of judgment aggregation: aggregators that are not generalized dictatorships, and anonymous, monotone and StrongDem aggregators. We prove that pic's, lpic's, a sub-class of pic's, and a subclass of lpic's, respectively, describe domains that admit each of the above four kind of aggregators. Then, we consider the property of systematicity and examine how our results change if the aggregators are required to satisfy it.

5.1. Generalized Dictatorships.

We begin by defining generalized dictatorships.

Definition 10. Let $F = (f_1, \ldots, f_n)$ be an n-tuple of k-ary conservative functions. F is a *generalized dictatorship for a domain* $D \subseteq \{0,1\}^n$, if, for any $x^1, \ldots, x^k \in D$, it holds that:

$$F(x^1, \ldots, x^k) := (f_1(x_1), \ldots, f_n(x_n)) \in \{x^1, \ldots, x^k\}. \tag{1}$$

Much like dictatorial functions, it is straightforward to observe that if F is a generalized dictator for D, then it is also an aggregator for D.

It should be noted here that in the original definition of Grandi and Endriss [18], generalized dictatorships are defined independently of a specific domain. Specifically, condition (1) is required to hold for all $x^1, \ldots, x^k \in \{0,1\}^n$. With this stronger definition, they show that the class of generalized dictators coincides with that of functions that are aggregators for every domain $D \subseteq \{0,1\}^n$.

Remark 4. *The difference in the definition of generalized dictatorships comes from a difference in the framework we use. Here, we opt to consider the aggregators* restricted *in the given domain, in the sense that we are not interested in what they do on inputs that are not allowed by it. The*

implications of this are not very evident in the Boolean framework, especially since we consider aggregators that satisfy IIA, on non-degenerate domains (in fact, this issue will not arise in any other aggregator present in this work, apart from generalized dictatorships). On the other hand, in the non-Boolean framework, using unrestricted aggregators could result in trivial cases of non-dictatorial aggregation, where the aggregator is not a projection only on inputs that are not allowed by the domain.

The following example shows that the result of Grandi and Endriss [18, Theorem 16] does not hold in our setting.

Example 8. Consider the Horn formula:

$$\phi_{11} = (x_1 \vee \neg x_2 \vee \neg x_3) \wedge (\neg x_1 \vee x_2 \vee \neg x_3) \wedge (\neg x_1 \vee \neg x_2 \vee x_3) \wedge (\neg x_1 \vee \neg x_2 \vee \neg x_3),$$

whose set of satisfying assignments is:

$$\mathrm{Mod}(\phi_{11}) = \{(0,0,0), (0,0,1), (0,1,0), (1,0,0)\}.$$

By definition, $\mathrm{Mod}(\phi_{11})$ is a Horn domain and it thus admits the binary symmetric aggregator $\bar{\wedge} = (\wedge, \wedge, \wedge)$. Furthermore, $\bar{\wedge}$ is not a generalized dictatorship for $\mathrm{Mod}(\phi_{11})$, since $\bar{\wedge}((0,0,1),(0,1,0)) = (0,0,0) \notin \{(0,0,1),(0,1,0)\}$.

On the other hand, consider the Horn formula:

$$\phi_{12} = (\neg x_1 \vee x_2) \wedge (x_2 \vee \neg x_3) \wedge (\neg x_1 \vee \neg x_2 \vee x_3).$$

$\bar{\wedge}$ is again an aggregator for the Horn domain:

$$\mathrm{Mod}(\phi_{12}) = \{(0,0,0), (0,1,0), (0,1,1), (1,1,1)\},$$

but, contrary to the previous case, $\bar{\wedge}$ is a generalized dictatorship for $\mathrm{Mod}(\phi_{12})$, since it is easy to verify that for any $x, y \in D'$, $\bar{\wedge}(x,y) \in \{x,y\}$.

Finally, observe that $(\wedge, \vee, \vee)$ is an aggregator for $\mathrm{Mod}(\phi_{12})$ that is not a generalized dictatorship. The latter claim follows from the fact that:

$$(\wedge, \vee, \vee)((0,1,0),(1,1,1)) = (0,1,1) \notin \{(0,1,0),(1,1,1)\},$$

while the former is left to the reader. Thus, interestingly enough, ϕ_{12} describes a domain admitting an aggregator that is not a generalized dictatorship, although it is not the aggregator that "corresponds" to the formula. ◇

It is easy to see that $(\mathrm{pr}_i^k, \ldots, \mathrm{pr}_i^k)$ is a generalized dictatorship of any $D \subseteq \{0,1\}^n$, for all $k \geq 1$ and for all $i \in \{1, \ldots, k\}$. Thus, trivially, every domain admits aggregators which are generalized dictatorships. On the other hand, every domain $D \subseteq \{0,1\}^n$ containing only two elements (a domain cannot contain less than two due to non-degeneracy) admits only generalized dictatorships. Indeed, assume $D = \{x, y\}$, $x \neq y$ and let F be a k-ary aggregator for D. Obviously, $F(x, \ldots, x) = x$ and $F(y, \ldots, y) = y$, since F is unanimous. Also, $F(x, y) \in \{x, y\} = D$ since F is an aggregator.

Our aim is again to find a syntactic characterization for domains that admit aggregators which are not generalized dictatorships. The following result shows that these domains are all the possibility domains with at least three elements, and are thus characterized by possibility integrity constraints.

Theorem 11. *A domain $D \subseteq \{0,1\}^n$, with at least three elements, admits an aggregator that is not a generalized dictatorship if and only if it is a possibility domain.*

Proof. The forward direction is obtained by the trivial fact that an aggregator that is not a generalized dictatorship is also non-dictatorial.

Now, suppose that D is a possibility domain. Then it is either affine or it admits a binary non-dictatorial aggregator. We begin with the affine case. It is a known result that $D \subseteq \{0,1\}^n$ is affine if and only if it is closed under $\oplus$, or, equivalently, if it admits the minority aggregator:

$$\bar{\oplus} = (\underbrace{\oplus, \ldots, \oplus}_{n\text{-times}})$$

Claim 11.1. *Let $D \subseteq \{0,1\}^n$ be an affine domain. Then, the* minority *aggregator:*

$$\bar{\oplus} = (\underbrace{\oplus, \ldots, \oplus}_{n\text{-times}})$$

is not a generalized dictatorship for D.

Proof. Let $x, y, z \in D$ be three pairwise distinct vectors. Since $y \neq z$, there exists a $j \in \{1, \ldots, n\}$ such that $y_j \neq z_j$. It follows that $y_j + z_j \equiv 1(\mod 2)$. This means that $\oplus(x_j, y_j, z_j) \neq x_j$ and thus that $\bar{\oplus}(x, y, z) \neq x$. In the same way we show that $\bar{\oplus}(x, y, z) \notin \{x, y, z\}$, which is a contradiction, since $\bar{\oplus}$ is an aggregator for D. □

Recall that the only binary unanimous functions are $\wedge, \vee, \mathrm{pr}_1^2, \mathrm{pr}_2^2$.

Claim 11.2. *Suppose $D \subseteq \{0,1\}^n$ admits a binary non-dictatorial non-symmetric aggregator $F = (f_1, \ldots, f_n)$. Then F is not a generalized dictatorship.*

Proof. Assume, to obtain a contradiction, that F is a generalized dictatorship for D and let $x, y \in D$. Then, $F(x, y) := z \in \{x, y\}$. Assume that $z = x$. The case where $z = y$ is analogous.

Let $J \subseteq \{1, \ldots, n\}$ such that f_j is symmetric, for all $j \in J$ and f_j is a projection otherwise. Note that $J \neq \{1, \ldots, n\}$. Let also $I \subseteq \{1, \ldots, n\} \setminus J$, such that $f_i = \mathrm{pr}_2^2$, for all $i \in I$ and $f_i = \mathrm{pr}_1^2$ otherwise. If $I \neq \emptyset$, then, for all $i \in I$, it holds that:

$$y_i = pr_2^2(x_i, y_i) = f_i(x_i, y_i) = z_i = x_i.$$

Since x, y were arbitrary, it follows that $D_i = \{x_i\}$, for all $i \in I$. Contradiction, since D is non-degenerate.

If $I = \emptyset$, then $f_j = \mathrm{pr}_1^2$, for all $j \notin J$. Note that in that case, $J \neq \emptyset$, lest F is dictatorial. Now, consider $F(y, x) := w \in \{x, y\}$ since F is a generalized dictatorship. By the definition of F, $w_j = z_j = x_j$, for all $j \in J$, and $w_i = y_i$, for all $i \notin J$. Thus, if $w = x$, D is degenerate on $\{1, \ldots, n\} \setminus J$, whereas if $w = y$, D is degenerate on J. In both cases, we obtain a contradiction. □

The only case left is when $D \subseteq \{0,1\}^n$ admits a binary symmetric aggregator. Contrary to the previous case, where we showed that the respective non-dictatorial aggregators could not be generalized dictatorships, here we cannot argue this way, as Example 8 attests. Interestingly enough, we show that as in Example 8, we can always find some symmetric aggregator for such a domain that is not a generalized dictatorship.

Claim 11.3. *Suppose $D \subseteq \{0,1\}^n$ admits a binary non-dictatorial symmetric aggregator $F = (f_1, \ldots, f_n)$. Then, there is a binary symmetric aggregator $G = (g_1, \ldots, g_n)$ for D (G can be different from F) that is not a generalized dictatorship for D.*

Proof. If F is not a generalized dictatorship for D, we have nothing to prove. Suppose it is and let $J \subseteq \{1, \ldots, n\}$, such that $f_j = \vee$, for all $j \in J$ and $f_i = \wedge$ for all $i \notin J$ (J can be both empty or $\{1, \ldots, n\}$).

Let $D^* = \{d^* = (d_1^*, \ldots, d_n^*) \mid d = (d_1, \ldots, d_n) \in D\}$, where:

$$d_j^* = \begin{cases} 1 - d_j & \text{if } j \in J \\ d_j & \text{else.} \end{cases}$$

By Lemma 3, $H = (h_1, \ldots, h_n)$ is a symmetric aggregator for D if and only if $H^* = (h_1^*, \ldots, h_n^*)$ is an aggregator for D^*, where $h_j^* = h_j$, for all $j \notin J$ and, for all $j \in J$, if $h_j = \vee$, then $h_j^* = \wedge$ and vice-versa. As expected, the property of being a generalized dictatorship carries on this transformation.

Claim 11.4. *H is a generalized dictatorship for D if and only if H^* is a generalized dictatorship for D^*.*

Proof. Let $x = (x_1, \ldots, x_n)$, $y = (y_1, \ldots, y_n) \in D$ and $z := H(x, y)$. Since $\vee(x_j, y_j) = 1 - \wedge(1 - x_j, 1 - y_j)$ and $\wedge(x_j, y_j) = 1 - \vee(1 - x_j, 1 - y_j)$, it holds that $z_j = h_j^*(x_j^*, y_j^*)$, for all $j \notin J$, and $1 - z_j = h_j^*(x_j^*, y_j^*)$, for all $j \in J$. Thus, $z^* = H^*(x^*, y^*)$. It follows that $z \in \{x, y\}$ if and only if $z^* \in \{x^*, y^*\}$. □

Now, since D admits the generalized dictatorship F, it follows that D^* admits the binary aggregator $\bar{\wedge} = (\underbrace{\wedge, \ldots, \wedge}_{n\text{-times}})$, that is also a generalized dictatorship. Our aim is to show that D^* admits a symmetric aggregator that is not a generalized dictatorship. The result will then follow by Claim 11.4.

For two elements $x^*, y^* \in D^*$, we write $x^* \leq y^*$ if, for all $j \in \{1, \ldots, n\}$ such that $x_j^* = 1$, it holds that $y_j^* = 1$.

Claim 11.5. *$\leq$ is a* total ordering *for D^*.*

Proof. To obtain a contradiction, let $x^*, y^* \in D^*$ such that neither $x^* \leq y^*$ nor $y^* \leq x^*$. Thus, there exist $i, j \in \{1, \ldots, n\}$, such that $x_i^* = 1$, $y_i^* = 0$, $x_j^* = 0$ and $y_j^* = 1$. Thus:

$$\wedge(x_i^*, y_i^*) = \wedge(x_j^*, y_j^*) = 0.$$

Then, $\bar{\wedge}(x^*, y^*) \notin \{x^*, y^*\}$. Contradiction, since $\bar{\wedge}$ is a generalized dictatorship. □

Thus, we can write $D^* = \{d^1, \ldots, d^N\}$, where $d^s \leq d^t$ if and only if $s \leq t$. Let $I \subseteq \{1, \ldots, n\}$ be such that, for all $j \in I$: $d_j^s = 0$ for $s = 1, \ldots, N-1$, and $d_j^N = 1$. Observe that I cannot be empty, lest $d^N = d^{N-1}$ and that $I \neq \{1, \ldots, n\}$, since $|D| \geq 3$. Let now $G = (g_1, \ldots, g_n)$ such that $g_j = \wedge$, for all $j \in I$ and $g_j = \vee$, for all $j \notin I$.

G is an aggregator for D^*. Indeed, let $d^s, d^t \in D^*$ with $s \leq t \leq N - 1$. Then, for all $j \notin I$:

$$g_j(d_j^s, d_j^t) = \vee(d_j^s, d_j^t) = d_j^t.$$

Also for all $j \in I$:

$$g_j(d_j^s, d_j^t) = \wedge(d_j^s, d_j^t) = 0 = d_j^t.$$

Thus, $G(d^s, d^t) = d^t \in D^*$. Finally, consider $G(d^s, d^N)$. Again, $g_j(d_j^s, d_j^N) = \wedge(d_j^s, d_j^N) = 0$ for all $j \in I$ and $g_j(d_j^s, d_j^N) = \vee(d_j^s, d_j^N) = d_j^N$, for all $j \notin I$. By definition of I, $G(d^s, d^N) = d^{N-1} \in D^*$. This, last point shows also that G is not a generalized dictatorship, since, for any $s \neq N - 1$, $d^{N-1} \notin \{d^s, d^N\}$. □

This completes the proof of Theorem 11. □

By Theorems 7 and 11 we obtain the following result.

Corollary 3. *A domain $D \subseteq \{0,1\}^n$, with at least three elements, admits an aggregator that is not a generalized dictatorship if and only if there exists a possibility integrity constraint whose set of models equals D.*

Remark 5. *What about knowing if a possibility integrity constraint really describes a domain that admits an aggregator that is not a generalized dictatorship? For the requirement of having a non-degenerate domain, the situation is the same as in Remark 3. For the requirement of the domain having at least three elements, given that it is non-degenerate, it is easy to see that such domains can only arise as the truth sets of possibility integrity constraints that are Horn, renamable Horn or affine. In all these cases, Creignou and Hébrard [5] have devised* polynomial-delay *algorithms that generate all the solutions of such formulas, which can easily be implemented to terminate if they find more than two solutions.*

5.2. **Preliminaries for Anonymous, Monotone and StrongDem Aggregators.** Our final results concern three kinds of non-dictatorial aggregators, whose properties are based on the majority aggregator.

Definition 11. Let $D \subseteq \{0,1\}^n$. A k-ary aggregator $F = (f_1, \ldots, f_n)$ for D is:

(1) Anonymous, if it holds that for all $j \in \{1, \ldots, n\}$ and for any permutation $p : \{1, \ldots, k\} \mapsto \{1, \ldots, k\}$:
$$f_j(a_1, \ldots, a_k) = f_j(a_{p(1)}, \ldots, a_{p(k)}),$$
for all $a_1, \ldots, a_k \in \{0,1\}$.

(2) Monotone, if it holds that for all $j \in \{1, \ldots, n\}$ and for all $i \in \{1, \ldots, k\}$:
$$f_j(a_1, \ldots, a_{i-1}, 0, a_{i+1}, \ldots, a_k) = 1 \Rightarrow f_j(a_1, \ldots, a_{i-1}, 1, a_{i+1}, \ldots, a_k) = 1.$$

(3) StrongDem, if it holds that for all $j \in \{1, \ldots, n\}$ and for all $i \in \{1, \ldots, k\}$, there exist $a_1, \ldots, a_{i-1}, a_{i+1}, \ldots, a_k \in \{0,1\}$:
$$f_j(a_1, \ldots, a_{i-1}, 0, a_{i+1}, \ldots, a_k) = f_j(a_1, \ldots, a_{i-1}, 1, a_{i+1}, \ldots, a_k).$$

Anonymous aggregators ensure that all the voters are treated equally, while monotone that if more voters agree with the aggregator's result, then the outcome does not change. From a Social Theoretic point of view, Nehring and Puppe [27] have argued that "For Arrowian (i.e. independent) aggregators, monotonicity is extremely natural, and it is hard to see how non-monotone Arrowian aggregators could be of interest in practice." StrongDem aggregators were introduced by Szegedy and Xu [33]. The idea here is that there is a way to fix the votes of any $k-1$ voters such that the remaining voter cannot change the outcome of the aggregation. Apart from the interest these aggregators have for Judgement Aggregation, Szegedy and Xu show that they have strong algebraic properties, as they relate to a property of functions called *strong resilience* (see again [23, 33]).

Notationally, since these properties are defined for each component of an aggregator, we will say that a Boolean function f is anonymous or monotone if it satisfies property 1 or 2 of Definition 11 respectively. A Boolean function f that satisfies property 3 of Definition 11 has appeared in the bibliography under the name of 1-*immune* (see [23]). The first immediate consequence of Definition 11, is that an anonymous or StrongDem aggregator is non-dictatorial. On the other hand, projection and binary symmetric functions are easily monotone, thus every

dictatorial and every binary aggregator is monotone. Furthermore, since projections are neither anonymous nor 1-immune and by Theorem 1 it is straightforward to observe the following results.

Corollary 4. *Any possibility domain D either admits a monotone non-dictatorial aggregator or an anonymous one. Furthermore, a domain D admitting an anonymous or StrongDem aggregator is a local possibility domain.*

Regardless of Corollary 4, we can find non-dictatorial aggregators that are neither anonymous, nor monotone, nor StrongDem. An easy such example is an aggregator with at least one component being pr_1^3 and another being $\oplus$, since pr_1^3 is not anonymous, $\oplus$ is not monotone and neither of the two is 1-immune. Corollary 4 implies that a domain admitting such an aggregator, admits also another that is monotone or anonymous.

We proceed now with some examples that highlight the various connections between these types of aggregators.

Example 9. Any renamable Horn or bijunctive formula describes a domain admitting a symmetric or majority aggregator respectively. Such aggregators are anonymous, monotone and StrongDem. For a more complicated example, consider the formula

$$\phi_{13} = (\neg x_1 \vee x_2) \wedge (x_2 \vee x_3 \vee x_4),$$

whose set of satisfying assignments is the local possibility domain:

$$\mathrm{Mod}(\phi_{13}) = \{0,1\}^4 \setminus \Big((\{(1,0)\} \times \{0,1\}^2) \cup \{(0,0,0,0)\} \Big).$$

It is straightforward to check that $\mathrm{Mod}(\phi_{13})$ admits the anonymous, monotone and StrongDem aggregator $(\wedge^{(3)}, \vee^{(3)}, \mathrm{maj}, \mathrm{maj})$.

On the other hand, consider the affine formula

$$\phi_{14} = x_1 \oplus x_2 \oplus x_3,$$

where:

$$\mathrm{Mod}(\phi_{14}) = \{(0,0,1),(0,1,0),(1,0,0),(1,1,1)\}.$$

It can be proven (by a combination of results by Dokow and Holzman [12, Example 3] and Kirousis et al. [22, Example 4.5]) that $\mathrm{Mod}(\phi_{14})$ does not admit any monotone or StrongDem aggregators. On the other hand, it does admit the anonymous aggregator $\bar{\oplus} = (\oplus, \oplus, \oplus)$.

Recall that in Example 6, we argued that the set of satisfying assignments of the formula: $\phi_7 = (\neg x_1 \vee x_2 \vee x_3) \wedge (x_1 \vee \neg x_2 \vee \neg x_3)$ is the impossibility domain $\mathrm{Mod}(\phi_7) = \{0,1\}^3 \setminus \{(1,0,0),(0,1,1)\}$. Consider also, from the same example, the formula $\phi_9 = (\neg x_1 \vee x_2 \vee x_3) \wedge (x_1 \vee \neg x_2 \vee \neg x_3) \wedge (\neg x_4 \vee x_5 \vee x_6) \wedge (x_4 \vee \neg x_5 \vee \neg x_6)$, whose set of satisfying assignments is: $\mathrm{Mod}(\phi_9) = \mathrm{Mod}(\phi_7) \times \mathrm{Mod}(\phi_7)$. $\mathrm{Mod}(\phi_9)$ is a possibility domain admitting the monotone aggregator $(\mathrm{pr}_1^2, \mathrm{pr}_1^2, \mathrm{pr}_1^2, \mathrm{pr}_2^2, \mathrm{pr}_2^2, \mathrm{pr}_2^2)$. On the other hand, $\mathrm{Mod}(\phi_9)$ admits neither anonymous, nor StrongDem aggregators, as it is not a local possibility domain. ◇

We now provide examples of StrongDem aggregators that are either not anonymous or not monotone. Domains admitting such aggregators can be proven to also admit aggregators that are anonymous and monotone (see Theorem 13 below).

Example 10. Let $F = \bar{f}$, where f is a ternary operation defined as follows:

$$f(0,0,0) = f(0,0,1) = f(0,1,1) = f(1,0,1) = 0,$$
$$f(0,1,0) = f(1,0,0) = f(1,1,0) = f(1,1,1) = 1.$$

Obviously, $\bar{f}$ is neither anonymous nor monotone, since e.g. $f(0,0,1) \neq f(0,1,0)$ and $f(0,1,0) = 1$, whereas $f(0,1,1) = 0$. On the other hand, $\bar{f}$ is StrongDem. Indeed, for each component of $\bar{f}$, it holds that:

$$f(x,0,1) = f(0,x,1) = f(0,0,x) = 0,$$

for all $x \in \{0,1\}$.

Now, consider $G = \bar{g}$ where g is a ternary operation defined as follows:

$$g(0,0,0) = g(0,0,1) = g(0,1,0) = g(1,0,0) = g(1,1,0) = 0,$$
$$g(0,1,1) = g(1,0,1) = g(1,1,1) = 1.$$

Again, $\bar{g}$ is easily not anonymous, since $g(1,1,0) \neq g(0,1,1)$. On the other hand, $\bar{g}$ is monotone and StrongDem. For the latter, observe that:

$$g(x,0,0) = g(0,x,0) = g(0,0,x) = 0,$$

for all $x \in \{0,1\}$. The former is very easy to check and is left to the reader.

Finally, let $H = \bar{h}$, where h is a 4-ary operation defined as follows:

$$h(x,y,z,w) = 1 \text{ if and only if exactly two or all of } x,y,z,w \text{ are equal to } 1.$$

Since the output of h does not depend on the positions of the input bits, h is anonymous. Also, h is 1-immune, since:

$$h(x,0,0,0) = h(0,y,0,0) = h(0,0,z,0) = h(0,0,0,w),$$

for all $x,y,z,w \in \{0,1\}$. On the other hand, h is not monotone, since $h(0,0,1,1) = 1$ and $h(0,1,1,1) = 0$. ◇

The only combination of properties from Definition 11 we have not seen, is an anonymous and monotone aggregator that is not Strong Dem. We end this subsection by proving that such aggregators do not exist.

Lemma 6. *Let f be a k-ary anonymous and monotone Boolean function. Then, f is also* 1*-immune.*

Proof. For $k = 2$, the only anonymous functions are $\wedge$ and $\vee$, which are also 1-immune.

Let $k \geq 3$. Since f is anonymous and monotone, it is not difficult to observe that there is some $l \in \{0,\ldots,k\}$, such that the output of f is 0 if and only if there are at most l 1's in the input bits. If $l > 0$, then:

$$f(x,0,0\ldots,0,0) = f(0,x,0,\ldots,0,0) = \cdots = f(0,0,0,\ldots,0,x) = 0,$$

for all $x \in \{0,1\}$. If $l = 0$, then:

$$f(x,1,1\ldots,1,1) = f(1,x,1,\ldots,1,1) = \cdots = f(1,1,1,\ldots,1,x) = 1,$$

for all $x \in \{0,1\}$. In both cases, f is 1-immune. □

5.3. **Characterizations for domains admitting anonymous, monotone and StrongDem aggregators.** We begin with the syntactic characterization of domains admitting anonymous aggregators. Nehring and Puppe [27, Theorem 2] showed that a domain admits a monotone locally non-dictatorial aggregator if and only if it admits a monotone anonymous one. Kirousis et al. [22] strengthened this result by dropping the monotonicity requirement and fixing the arity of the anonymous aggregator, as a direct consequence of Theorem 2.

Corollary 5 (Kirousis et al. [22], Corollary 5.11)**.** *D is a local possibility domain if and only if it admits a ternary anonymous aggregator.*

Proof. Immediate from the fact that any aggregator of the type described in Theorem 2 is anonymous. □

Thus, we obtain the following result.

Corollary 6. *D admits a k-ary anonymous aggregator if and only if there exists a local possibility integrity constraint whose set of models equals D.*

To deal with monotone aggregators we will need some preliminary work. The first fact we will use is that the set of aggregators of a domain D is *closed under superposition*. That is, if $F = (f_1, \ldots, f_m)$ is a k-ary aggregator for D and $G^1, \ldots, G^k$ are l-ary aggregators for D, where $G^i = (g_1^i, \ldots, g_n^i)$, $i = 1, \ldots, k$, then $H := F(G^1, \ldots, G^k)$ is an l-ary aggregator for D, where $H = (h_1, \ldots, h_n)$ and:

$$h_j(x_1, \ldots, x_l) = f_j(g_j^1(x_1, \ldots, x_l), \ldots, g_j^k(x_1, \ldots, x_l)),$$

for all $j = 1, \ldots, n$ and for all $x_1, \ldots, x_l \in \{0, 1\}$. The proof of this is straightforward and can be found in [22, Lemma 5.6].

This will help us employ a result from the field of Universal Algebra. *Clones* are sets of operations that contain all projections and are closed under superposition (see e.g. Szendrei [34]). Post [29] provided a complete classification of clones of Boolean operations. This result has already been effectively used by Kirousis et al. [22] in order to obtain the characterizations of possibility and local possibility domains in the Boolean and non-Boolean framework. Here, we will use it analogously.

By the fact that the set of aggregators of a domain D is closed under superposition, we obtain the following result.

Lemma 7. *For a Boolean domain $D \subseteq \{0, 1\}^n$, let, for all $j \in \{1, \ldots, n\}$:*

$$\mathcal{C}_j := \{f \mid \textit{There exists an aggregator } F = (f_1, \ldots, f_n) \textit{ for } D \textit{ s.t. } f_j = f\},$$

be the set of the j-th components of every aggregator for D. Then, $\mathcal{C}_j$ is a clone.

The main feature of Post's classification result we will use here is the following (see [1] for an easy to follow presentation):

Lemma 8. *Let $\mathcal{C}$ be a clone containing only unanimous functions. Then, either at least one of $\wedge, \vee, \mathrm{maj}, \oplus$ is in $\mathcal{C}$, or $\mathcal{C}$ contains only projections.*

Finally, we will need some definitions. We say that a ternary Boolean operator g is *commutative* if and only if

$$g(x, x, y) = g(x, y, x) = g(y, x, x),$$

for all $x, y \in \{0, 1\}$. It is not difficult to see that a ternary operator g is commutative if and only if $g \in \{\wedge^{(3)}, \vee^{(3)}, \mathrm{maj}, \oplus\}$ (again, see [22, Lemma 5.7]).

Lastly, we say that a k-ary Boolean operation f is *linear*, if there exist constants $c_0, \ldots, c_k \in \{0, 1\}$ such that:

$$f(x_1, \ldots, x_k) = c_0 \oplus c_1 x_1 \oplus \cdots \oplus c_k x_k,$$

where $\oplus$ again denotes binary addition $\mod 2$. We need two facts concerning linear functions.

Lemma 9. *Let $f : \{0, 1\}^k \mapsto \{0, 1\}$ be a linear function and let $c_0, c_1, \ldots, c_k \in \{0, 1\}$ such that:*

$$f(x_1, \ldots, x_k) = c_0 \oplus c_1 x_1 \oplus \cdots \oplus c_k x_k.$$

Then, f is unanimous if and only if $c_0 = 0$ and there is an odd number pairwise distinct indices $i \in \{1, \ldots, k\}$ such that $c_i = 1$.

Proof. The inverse direction is straightforward. For the forward direction, set $x_1 = \cdots = x_k = 0$. Then, $f(0, \ldots, 0) = c_0$ and thus $c_0 = 0$ since f is unanimous. Finally, assume, to obtain a contradiction, that there is an even number of $c_1, \ldots, c_k$ that are equal to 1. Set $x_1 = \cdots = x_k = 1$. Then, it holds that $f(1, \ldots, 1) = 0$ and f is not unanimous. Contradiction. □

Since we work only with unanimous functions, from now on we will assume that a linear function satisfies the conditions of Lemma 9. This implies also that any linear function has odd arity.

Let again $f : \{0,1\}^k \mapsto \{0,1\}$. We say that f is an *essentially unary function*, if there exists a *unary* Boolean function g and an $i \in \{1, \ldots, k\}$, such that:

$$f(x_1, \ldots, x_k) = g(x_i),$$

for all $x_1, \ldots, x_k \in \{0,1\}$ (again, see [1,2,19,20]). Obviously, the only unanimous such functions are the projections. Note also that any k-ary linear functions f, with exactly one $i \in \{1, \ldots, k\}$ such that $c_i = 1$ is an essentially unary function.

Lemma 10. *Let $f : \{0,1\}^k \mapsto \{0,1\}$ be a linear function, $k \geq 3$. Then, either f is an essentially unary function, or it is neither monotone nor 1-immune.*

Proof. Let $c_1, \ldots, c_k \in \{0,1\}$ such that:

$$f(x_1, \ldots, x_k) = c_1 x_1 \oplus \cdots \oplus c_k x_k$$

and assume it is not an essentially unary function. Then, there exist at least three pairwise distinct indices $i \in \{1, \ldots, k\}$ such that $c_i = 1$. If there are exactly three, $f = \oplus$, which is easily neither monotone, nor 1-immune.

Assume now that there are at least five pairwise distinct $i \in \{1, \ldots, k\}$ such that $c_i = 1$. We will need only four of these indices.

Now, let $j_1, j_2, j_3, j_4 \in \{1, \ldots, k\}$ such that $c_{j_1}, c_{j_2}, c_{j_3}, c_{j_4} = 1$. Set $x_{j_1} = x_{j_2} = x_{j_3} = 1$ and $x_i = 0$, for all $i \in \{1, \ldots, k\} \setminus \{j_1, j_2, j_3\}$. Then, $f(x_1, \ldots, x_k) = 1$. By letting $x_{j_4} = 1$ too, we obtain $f(x_1, \ldots, x_k) = 0$, which shows that f is not monotone.

Finally, to obtain a contradiction, suppose f is 1-immune. Then, there exist $d_2, \ldots, d_k \in \{0,1\}$ such that:

$$\begin{aligned} f(0, d_2, \ldots, d_k) &= (1, d_2, \ldots, d_k) \iff \\ c_2 d_2 \oplus \cdots \oplus c_k d_k &= c_1 \oplus c_2 d_2 \oplus \cdots c_k d_k \iff \\ c_1 &= 0. \end{aligned}$$

Continuing in the same way, we can prove that $c_j = 0$, for $j = 1, \ldots, k$, which is a contradiction. □

We are now ready to prove our results concerning monotone and StrongDem aggregators.

Theorem 12. *A domain $D \subseteq \{0,1\}^n$ admits a monotone non-dictatorial aggregator of some arity if and only if it admits a binary non-dictatorial one.*

Proof. That a domain admitting a binary non-dictatorial aggregator, admits also a non-dictatorial monotone one is obvious, since all binary unanimous functions are monotone.

For the forward direction, since D admits a monotone non-dictatorial aggregator, it is a possibility domain. Now, to obtain a contradiction, suppose D does not admit a binary non-dictatorial aggregator. Kirousis et al. [22, Lemma 3.4] showed that in this case, every k-ary

aggregator, $k \geq 2$ for D is systematic (the notion of *local monomorphicity* they use corresponds to systematicity in the Boolean framework).

Now, since D contains no binary non-dictatorial aggregators, $\wedge, \vee \notin \mathcal{C}_j$, for all $j \in \{1, \ldots, n\}$. Thus, by Lemma 8 either maj or $\oplus$ are contained in $\mathcal{C}_j$, for all $j \in \{1, \ldots, n\}$ (since the aggregators must be systematic), lest each $\mathcal{C}_j$ contains only projections.

Assume that $\overline{\text{maj}}$ is an aggregator for D. Then, by Kirousis et al. [22, Theorem 3.7], D admits also a binary non-dictatorial aggregator. Contradiction.

Thus, we also have that maj $\notin \mathcal{C}_j$, $j = 1, \ldots, n$. It follows that only $\oplus \in \mathcal{C}_j$, $j = 1, \ldots, n$. By Post [29], it follows that for all $j \in \{1, \ldots, n\}$, $\mathcal{C}_j$ contains only linear functions (see also [1]). By Lemma 10, we obtain a contradiction. □

Thus, by Proposition 3 and Theorem 6, we obtain the following syntactic characterization.

Corollary 7. *D admits a k-ary non-dictatorial monotone aggregator if and only if there exists a separable or renamable partially Horn integrity constraint whose set of models equals D.*

To end this subsection, we now consider StrongDem aggregators. Kirousis et al. [22] used what they named the "diamond" operator $\diamond$, in order to combine ternary aggregators to obtain new ones whose components are commutative functions (see also Bulatov's [3, Section 4.3] "Three Operations Lemma"). Unfortunately, this operator will not suffice for our purposes, so we will also use a new operator we call the *star* operator $\star$.

Let $F = (f_1, \ldots, f_n)$ and $G = (g_1, \ldots, g_n)$ be two n-tuples of ternary functions. Define $E := F \diamond G$ and $H := F \star G$ to be the n-tuples of ternary functions $E = (e_1, \ldots, e_n)$ and $H = (h_1, \ldots, h_n)$ where:

$$e_j(x, y, z) = f_j(g_j(x, y, z), g_j(y, z, x), g_j(z, x, y)),$$
$$h_j(x, y, z) = f_j(f_j(x, y, z), f_j(x, y, z), g_j(x, y, z)),$$

for all $x, y, z \in \{0, 1\}$. Easily, if F and G are aggregators for a domain D, then so are E and H, since they are produced by a superposition of F and G. It is easy to notice that e_j is commutative if and only if either f_j or g_j are. Also, under some assumptions for F and G, H has no components equal to $\oplus$. In Kirousis et al. [22, Lemma 5.10], it has been proven that, if $F \diamond G = E = (e_1, \ldots, e_n)$, then $e_j \in \{\wedge^{(3)}, \vee^{(3)}, \text{maj}, \oplus\}$ if and only if either f_j or $g_j \in \{\wedge^{(3)}, \vee^{(3)}, \text{maj}, \oplus\}$. Furthermore, if g_j is commutative, then $e_j = g_j$, $j = 1, \ldots, n$. On the other hand, for the $\star$ operation, we have the following result.

Lemma 11. *Let $F = (f_1, \ldots, f_n)$ be an n-tuple of ternary functions, such that $f_j \in \{\wedge^{(3)}, \vee^{(3)}, \text{maj}, \oplus\}$, $j = 1, \ldots, n$, and let $J = \{j \mid f_j = \oplus\}$. Let also $G = (g_1, \ldots, g_n)$ be an n-tuple of ternary functions, such that $g_j \in \{\wedge^{(3)}, \vee^{(3)}, \text{maj}\}$, for all $j \in J$. Then, for the n-tuple of ternary functions $F \star G := H = (h_1, \ldots, h_n)$, it holds that:*

$$h_j \in \{\wedge^{(3)}, \vee^{(3)}, \text{maj}\},$$

for $j = 1, \ldots, n$.

Proof. First, let $j \in \{1, \ldots, n\} \setminus J$. Then, $f_j \in \{\wedge^{(3)}, \vee^{(3)}, \text{maj}\}$ and let $x, y, z \in \{0, 1\}$ that are not all equal (lest we have nothing to show since all f_j, g_j are unanimous). If $f_j = \wedge^{(3)}$, then easily:

$$h_j(x, y, z) = \wedge^{(3)}(\wedge^{(3)}(x, y, z), \wedge^{(3)}(x, y, z), g_j(x, y, z)) = \wedge^{(3)}(0, 0, g_j(x, y, z)) = 0,$$

which shows that $h_j = \wedge^{(3)}$. Analogously, we show that $f_j = \vee^{(3)}$ implies that $h_j = \vee^{(3)}$. Finally, let $f_j = \text{maj}$ and let $\text{maj}(x, y, z) := z \in \{0, 1\}$. Then:

$$h_j(x, y, z) = \text{maj}(\text{maj}(x, y, z), \text{maj}(x, y, z), g_j(x, y, z)) = \text{maj}(z, z, g_j(x, y, z)) = z,$$

which shows that $h_j = \text{maj}$.

Thus, we can now assume that $J \neq \emptyset$. Let $j \in J$. Then, we have that $f_j = \oplus$ and $g_j \in \{\wedge^{(3)}, \vee^{(3)}, \text{maj}\}$. Thus, we have that:

$$h_j(x, y, z) = \oplus(\oplus(x, y, z), \oplus(x, y, z), g_j(x, y, z)) = g_j(x, y, z),$$

from which it follows that $h_j \in \{\wedge^{(3)}, \vee^{(3)}, \text{maj}\}$. The proof is now complete. □

At last, we are ready to prove our final results.

Theorem 13. *A Boolean domain $D \subseteq \{0, 1\}^n$ admits a k-ary StrongDem aggregator if and only if it admits a ternary aggregator $F = (f_1, \ldots, f_n)$ such that $f_j \in \{\wedge^{(3)}, \vee^{(3)}, \text{maj}\}$, for $j = 1, \ldots, n$.*

Proof. It is very easy to see that all the functions in $\{\wedge^{(3)}, \vee, \text{maj}\}$ are 1-immune. Thus, we only need to prove the forward direction of the theorem.

To that end, let $F = (f_1, \ldots, f_n)$ be a k-ary StrongDem aggregator for D. Then, by Theorem 2, there exists a ternary aggregator $G = (g_1, \ldots, g_n)$ such that $g_j \in \{\wedge^{(3)}, \vee^{(3)}, \text{maj}, \oplus\}$ for $j = 1, \ldots, n$. Let $J = \{j \mid f_j = \oplus\}$. If $J = \emptyset$, then we have nothing to prove. Otherwise, consider the clones $\mathcal{C}_j$, for each $j \in J$.

Suppose now that there exists a $j \in J$, such that $\mathcal{C}_j$ contains neither $\wedge$, nor $\vee$, nor maj. By Post's classification of clones of Boolean functions (see [1, 29]) and since $\mathcal{C}_j$ contains $\oplus$ and only unanimous functions, $\mathcal{C}_j$ contains only linear unanimous functions. Again, by Lemma 10, $\mathcal{C}_j$ does not contain any 1-immune function. Contradiction.

Thus, for each $j \in J$, it holds that $\mathcal{C}_j$ contains either $\wedge$, or $\vee$ or maj. In the first two cases, $\mathcal{C}_j$ obviously contains $\wedge^{(3)}$ or $\vee^{(3)}$ too respectively. Then, it holds that for each $j \in J$ there exists an aggregator $H^j = (h_1^j, \ldots, h_n^j)$, such that $h_j^j \in \{\wedge^{(3)}, \vee^{(3)}, \text{maj}\}$. Let $J := \{j_1, \ldots, j_t\}$.

We will now perform a series of iterative combinations between G and the various H^j's, using the $\diamond$ and $\star$ operators, in order to obtain the required aggregator.

First, let $G^j = G \diamond H^j$, for all $j \in J$. By Kirousis et al. [22, Lemma 5.10], we have that

$$G_i^j \in \{\wedge^{(3)}, \vee^{(3)}, \text{maj}, \oplus\},$$

for all $i \in \{1, \ldots, n\}$ and $j \in J$. Furthermore,

$$G_{j_s}^{j_s} \in \{\wedge^{(3)}, \vee^{(3)}, \text{maj}\},$$

for $s = 1, \ldots, t$. Thus for the aggregator:

$$G^* := (\cdots((G \star G^{j_1}) \star G^{j_2}) \star \cdots \star G^{j_t}),$$

we have, by Lemma 11:

$$G_j^* \in \{\wedge^{(3)}, \vee^{(3)}, \text{maj}\},$$

for $j = 1, \ldots, n$, which concludes the proof. □

Recall Definition 6. We say that a local possibility integrity constraint is $\oplus$-*free*, if $V_2 = \emptyset$. Thus, we obtain the following syntactic characterization.

Corollary 8. *A Boolean domain $D \subseteq \{0,1\}^n$ admits a k-ary StrongDem aggregator if and only if there exists an $\oplus$-free local possibility integrity constraint whose set of satisfying assignments equals D.*

5.4. **Systematic Aggregators.** We end this work with a discussion concerning systematic aggregators. This is a natural requirement for aggregators from a Social Choice point of view, given that the issues that need to be decided are of the same nature. Recall that $F = (f_1, \ldots, f_n)$ is systematic if $f_1 = f_2 = \ldots = f_n$.

Definition 12. Let $D \subseteq \{0,1\}^n$ be a Boolean domain and $f : \{0,1\}^k \mapsto \{0,1\}$ a k-ary Boolean operation. f is a *polymorphism* for D (or f *preserves* D or D *is closed under* f) if, for all $x^1, \ldots, x^k \in D$:

$$(f(x_1), \ldots, f(x_n)) \in D,$$

where $x^i = (x_1^i, \ldots, x_n^i)$ and $x_j = (x_j^1, \ldots, x_j^k)$, $i = 1, \ldots, k$, $j = 1, \ldots, n$.

The notion of polymorphisms can be found in many standard texts concerning Abstract and Universal Algebra (see e.g. Szendrei [34]). The following is obvious by considering the definitions of an aggregator and a polymorphism.

Lemma 12. *Let $D \subseteq \{0,1\}^n$ be a Boolean domain and $F = \bar{f}$ a systematic n-tuple of k-ary Boolean functions. Then F is an aggregator for D if and only if f is a polymorphism for D.*

Polymorphisms have been extensively studied in the bibliography and they play a central role in Post's results we discussed in Subsection 5.3. These results where also connected with Complexity Theory, where they can be used to provide an alternative proof to the Dichotomy Theorem in the complexity of the *satisfiability problem* (see [1,2,19,20]). Here, we use a corollary of this Theorem, that can be obtained directly by Post's Lattice, without considering complexity theoretic notions. For an direct algebraic approach, see also Szendrei [34, Proposition 1.12] (by noting that the only Boolean *semi-projections* of arity at least 3 are projections).

Corollary 9. *Let $D \subseteq \{0,1\}^n$ be a Boolean domain. Then, either D admits only essentially unary functions, or it is closed under $\wedge$, $\vee$,* maj *or $\oplus$.*

This directly implies that domains admitting non-dictatorial systematic aggregators are either Horn, dual-Horn, bijunctive or affine. We thus immediately obtain the following characterization.

Corollary 10. *A Boolean domain $D \subseteq \{0,1\}^n$ admits a k-ary non-dictatorial systematic aggregator if and only if there exists an integrity constraint which is either Horn, dual Horn, bijunctive or affine, whose set of satisfying assignments equals D.*

Remark 6. *Why does* $\overline{\text{maj}}$ *appears here, although it did not in the characterization of possibility domains (Theorem 1)? In the Boolean case, a domain admitting* $\overline{\text{maj}}$*, also admits a binary aggregator $F = (f_1, \ldots, f_n)$, such that $f_j \in \{\wedge, \vee\}$, $j = 1, \ldots, n$ (see Kirousis et al. [22, Theorem 3.7]). The problem is that this aggregator need not be systematic. In fact, the proof of the aforementioned theorem would produce a systematic aggregator only if $(0, \ldots, 0)$ or $(1, \ldots, 1) \in D$.*

Now, what if want to characterize domains admitting some of the various non-dictatorial aggregators we discussed, but requiring also that these aggregators satisfy systematicity? By Theorem 11, we know that Corollary 10 works for domains admitting systematic aggregators that are not generalized dictatorships too. Furthermore, all the aggregators (resp. integrity constraints) of Corollary 9 (resp. 10) are locally non-dictatorial and anonymous aggregators

(resp. lpic's), thus we also have characterizations for domains admitting systematic locally non-dictatorial or anonymous aggregators.

For domains admitting monotone or StrongDem systematic aggregators, we will obtain the result by Lemma 10 and Post's Lattice. We will again use the terminology of polymorphisms.

Corollary 11. *A domain $D \subseteq \{0,1\}^n$ admits a k-ary systematic non-dictatorial monotone or StrongDem aggregator if and only if it is closed under $\wedge$, $\vee$ or* maj.

Proof. It is known (and straightforward to see) that the set of polymorphisms of a domain is a clone. Let $\mathcal{C}$ be the Boolean clone of polymorphisms of D. Since it admits a non-dictatorial aggregator, at least one operator from $\wedge, \vee, \mathrm{maj}, \oplus$ is in $\mathcal{C}$. By Lemma 10, this cannot be only $\oplus$. □

Thus, finally, we have the following result.

Corollary 12. *A Boolean domain $D \subseteq \{0,1\}^n$ admits a k-ary systematic non-dictatorial monotone or StrongDem aggregator if and only if there exists an integrity constraint which is either Horn, dual Horn or bijunctive, whose set of satisfying assignments equals D.*

Concluding remarks

It is known that any domain on n issues can be represented either by n formulas $\phi_1, \ldots, \phi_n$ (an agenda), in which case the domain is the set of binary n-vectors, the i-component of which represents the acceptance or rejection of ϕ_i in a consistent way (logic-based approach), or, alternatively, by a single formula ϕ of n variables (an integrity constraint), in which case the domain is the set of models of ϕ. In the former case, there are results, albeit of non-algorithmic nature, that give us conditions on the syntactic form of the ϕ_i's, so that the domain accepts a non-dictatorial aggregator. In this work, we show that domains accepting various kinds of non-dictatorial aggregators can always be described by integrity constraints of specific, easily identifiable, syntactic form. For domains that admit non-dictatorial aggregators, or aggregators that are not generalized dictatorships, we call such formulas possibility integrity constraints and furthermore, we show that a subclass of such formulas, the separable and renamable partially Horn formulas, describe domains admitting monotone non-dictatorial aggregators. Then, we show that local possibility domains and domains admitting anonymous aggregators coincide and are described by local possibility integrity constraints, while domains admitting StrongDem aggregators are described by a subclass of local possibility integrity constraints we called $\oplus$-free. Finally, we discuss the corresponding results for systematic aggregators, which are in fact polymorphisms of the domain. Our results are algorithmic, in the sense that (i) recognizing integrity constraints of the above types can be implemented in time linear in the length of the input formula and (ii) given a domain admitting some of the above non-dictatorial aggregators, a corresponding integrity constraint, whose number of clauses is polynomial in the size of the domain, can be constructed in time polynomial in the size of the domain. Our proofs draw from results in judgment aggregation theory as well from results about propositional formulas and logical relations.

Aknowledgements

We are grateful to Bruno Zanuttini for his comments that improved the presentation and simplified several proofs. Lefteris Kirousis is grateful to Phokion Kolaitis for initiating him to the area of Computational Social Choice Theory. We thank Eirini Georgoulaki for her valuable help in the final stages of writing this paper.

References

[1] Elmar Böhler, Nadia Creignou, Steffen Reith, and Heribert Vollmer. Playing with Boolean blocks, part I: Post's lattice with applications to complexity theory. In *SIGACT News*. Citeseer, 2003.

[2] Elmar Böhler, Nadia Creignou, Steffen Reith, and Heribert Vollmer. Playing with Boolean blocks, part II: Constraint satisfaction problems. In *ACM SIGACT-Newsletter*. Citeseer, 2004.

[3] Andrei A Bulatov. Conservative constraint satisfaction re-revisited. *Journal of Computer and System Sciences*, 82(2):347–356, 2016.

[4] Fabrizio Cariani, Marc Pauly, and Josh Snyder. Decision framing in judgment aggregation. *Synthese*, 163(1):1–24, 2008.

[5] Nadia Creignou and J-J Hébrard. On generating all solutions of generalized satisfiability problems. *RAIRO-Theoretical Informatics and Applications*, 31(6):499–511, 1997.

[6] Rina Dechter and Judea Pearl. Structure identification in relational data. *Artificial Intelligence*, 58(1-3):237–270, 1992.

[7] Alvaro del Val. On 2-sat and renamable Horn. In *Proceedings of the National Conference on Artificial Intelligence*, pages 279–284. Menlo Park, CA; Cambridge, MA; London; AAAI Press; MIT Press; 1999, 2000.

[8] Josep Díaz, Lefteris M. Kirousis, Sofia Kokonezi, and John Livieratos. Algorithmically efficient syntactic characterization of possibility domains. In *46th International Colloquium on Automata, Languages, and Programming, ICALP 2019, July 9-12, 2019, Patras, Greece.*, pages 50:1–50:13, 2019. URL: `https://doi.org/10.4230/LIPIcs.ICALP.2019.50`, `doi:10.4230/LIPIcs.ICALP.2019.50`.

[9] Franz Dietrich. A generalised model of judgment aggregation. *Social Choice and Welfare*, 28(4):529–565, 2007.

[10] Franz Dietrich and Christian List. Arrow's theorem in judgment aggregation. *Social Choice and Welfare*, 29(1):19–33, 2007.

[11] Elad Dokow and Ron Holzman. Aggregation of binary evaluations for truth-functional agendas. *Social Choice and Welfare*, 32(2):221–241, 2009.

[12] Elad Dokow and Ron Holzman. Aggregation of binary evaluations. *Journal of Economic Theory*, 145(2):495–511, 2010.

[13] Ramez Elmasri and Sham Navathe. *Fundamentals of database systems*. Pearson London, 2016.

[14] Herbert B. Enderton. *A mathematical introduction to logic*. Elsevier, 2001.

[15] Ulle Endriss and Ronald de Haan. Complexity of the winner determination problem in judgment aggregation: Kemeny, slater, tideman, young. In *Proceedings of the 2015 International Conference on Autonomous Agents and Multiagent Systems*, pages 117–125. International Foundation for Autonomous Agents and Multiagent Systems, 2015.

[16] Ulle Endriss, Umberto Grandi, Ronald De Haan, and Jérôme Lang. Succinctness of languages for judgment aggregation. In *Fifteenth International Conference on the Principles of Knowledge Representation and Reasoning*, 2016.

[17] Umberto Grandi and Ulle Endriss. Binary aggregation with integrity constraints. In *IJCAI Proceedings-International Joint Conference on Artificial Intelligence*, volume 22, page 204, 2011.

[18] Umberto Grandi and Ulle Endriss. Lifting integrity constraints in binary aggregation. *Artificial Intelligence*, 199:45–66, 2013.

[19] Peter Jeavons and David Cohen. An algebraic characterization of tractable constraints. In *International Computing and Combinatorics Conference*, pages 633–642. Springer, 1995.

[20] Peter Jeavons, David Cohen, and Marc Gyssens. How to determine the expressive power of constraints. *Constraints*, 4(2):113–131, 1999.

[21] Lefteris Kirousis, Phokion G Kolaitis, and John Livieratos. On the computational complexity of non-dictatorial aggregation. In *Proceedings 17th International Conference on Relational and Algebraic Methods in Computer Science*. Springer, 2018. For an extended version, see `https://arxiv.org/abs/1711.01574v2`.

[22] Lefteris Kirousis, Phokion G Kolaitis, and John Livieratos. Aggregation of votes with multiple positions on each issue. *ACM Transactions on Economics and Computation (TEAC)*, 7(1):1, 2019.

[23] Gábor Kun and Mario Szegedy. A new line of attack on the dichotomy conjecture. *European Journal of Combinatorics*, 52:338–367, 2016.

[24] Jérôme Lang, Marija Slavkovik, and Srdjan Vesic. Agenda separability in judgment aggregation. In *Thirtieth AAAI Conference on Artificial Intelligence*, 2016.

[25] Harry R Lewis. Renaming a set of clauses as a Horn set. *Journal of the ACM (JACM)*, 25(1):134–135, 1978.

[26] Christian List. The theory of judgment aggregation: An introductory review. *Synthese*, 187(1):179–207, 2012.
[27] Klaus Nehring and Clemens Puppe. Abstract arrowian aggregation. *Journal of Economic Theory*, 145(2):467–494, 2010.
[28] Gabriella Pigozzi. Belief merging and the discursive dilemma: an argument-based account to paradoxes of judgment aggregation. *Synthese*, 152(2):285–298, 2006.
[29] Emil Leon Post. *The two-valued iterative systems of mathematical logic*. Number 5 in Annals of Mathematics Studies. Princeton University Press, 1941.
[30] Willard V Quine. On cores and prime implicants of truth functions. *The American Mathematical Monthly*, 66(9):755–760, 1959.
[31] Thomas J. Schaefer. The complexity of satisfiability problems. In *Proc. of the 10th Annual ACM Symp. on Theory of Computing*, pages 216–226, 1978.
[32] Larry J Stockmeyer. The polynomial-time hierarchy. *Theoretical Computer Science*, 3(1):1–22, 1976.
[33] Mario Szegedy and Yixin Xu. Impossibility theorems and the universal algebraic toolkit. *CoRR*, abs/1506.01315, 2015. URL: `http://arxiv.org/abs/1506.01315`.
[34] Ágnes Szendrei. *Clones in universal algebra*, volume 99. Presses de l'Université de Montréal, 1986.
[35] Robert Endre Tarjan. Depth-first search and linear graph algorithms. *SIAM J. Comput.*, 1(2):146–160, 1972.
[36] Robert Wilson. On the theory of aggregation. *Journal of Economic Theory*, 10(1):89–99, 1975.
[37] Susumu Yamasaki and Shuji Doshita. The satisfiabilty problem for a class consisting of Horn sentences and some non-Horn sentences in proportional logic. *Information and Control*, 59(1-3):1–12, 1983.
[38] Bruno Zanuttini and Jean-Jacques Hébrard. A unified framework for structure identification. *Information Processing Letters*, 81(6):335–339, 2002.

[1]Computer Science Department, Universitat Politècnica de Catalunya, Barcelona, [2]Department of Mathematics, National and Kapodistrian University of Athens
E-mail address: `diaz@cs.upc.edu, {lkirousis, jlivier89, skoko}@math.uoa.gr`