Algorithmically Efficient Syntactic Characterization of Possibility Domains

Josep Díaz
Computer Science Department, Universitat Politècnica de Catalunya, Barcelona
diaz@cs.upc.edu

Lefteris Kirousis
Department of Mathematics, National and Kapodistrian University of Athens
Computer Science Department, Universitat Politècnica de Catalunya, Barcelona
lkirousis@math.uoa.gr

Sofia Kokonezi
Department of Mathematics, National and Kapodistrian University of Athens
skoko@math.uoa.gr

John Livieratos
Department of Mathematics, National and Kapodistrian University of Athens
jlivier89@math.uoa.gr

Abstract

We call *domain* any arbitrary subset of a Cartesian power of the set $\{0, 1\}$ when we think of it as reflecting abstract rationality restrictions on vectors of two-valued judgments on a number of issues. In Computational Social Choice Theory, and in particular in the theory of judgment aggregation, a domain is called a possibility domain if it admits a non-dictatorial aggregator, i.e. if for some k there exists a unanimous (idempotent) function $F : D^k \to D$ which is not a projection function. We prove that a domain is a possibility domain if and only if there is a propositional formula of a certain syntactic form, sometimes called an integrity constraint, whose set of satisfying truth assignments, or models, comprise the domain. We call *possibility* integrity constraints the formulas of the specific syntactic type we define. Given a possibility domain D, we show how to construct a possibility integrity constraint for D efficiently, i.e, in polynomial time in the size of the domain. We also show how to distinguish formulas that are possibility integrity constraints in linear time in the size of the input formula. Finally, we prove the analogous results for local possibility domains, i.e. domains that admit an aggregator which is not a projection function, even when restricted to any given issue. Our result falls in the realm of classical results that give syntactic characterizations of logical relations that have certain closure properties, like e.g. the result that logical relations component-wise closed under logical AND are precisely the models of Horn formulas. However, our techniques draw from results in judgment aggregation theory as well from results about propositional formulas and logical relations.

2012 ACM Subject Classification Theory of computation → Theory and algorithms for application domains

Keywords and phrases collective decision making, computational social choice, judgment aggregation, logical relations, algorithm complexity

Digital Object Identifier 10.4230/LIPIcs.ICALP.2019.50

Category Track A: Algorithms, Complexity and Games

Related Version A full version of the paper is available at [3], `https://arxiv.org/abs/1901.00138`.

Funding *Josep Díaz*: Research partially supported by TIN2017-86727-C2-1-R, GRAMM.
Lefteris Kirousis: Research carried out while visiting the Computer Science Department of the Universitat Politècnica de Catalunya and supported by TIN2017-86727-C2-1-R, GRAMM.

46th International Colloquium on Automata, Languages, and Programming (ICALP 2019).
Editors: Christel Baier, Ioannis Chatzigiannakis, Paola Flocchini, and Stefano Leonardi;
Article No. 50; pp. 50:1–50:13

Leibniz International Proceedings in Informatics
Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

Acknowledgements 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.

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 [19], and then elaborated by several others (see e.g. the work by Dietrich [4] and Dokow and Holzman [6, 5]). 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 [8, Theorem 15B]). Zanuttini and Hébrard [21] give an algorithm that finds such a formula and runs in polynomial-time with respect to the size of the representation of D as input. Following Grandi and Endriss [11], 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. [7]).

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 and (ii) formulas whose clauses are exclusive OR's of their literals. 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 [21] 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 2.11) 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

[1] A weaker notion of Horn formulas has appeared before in the work of Yamasaki and Doshita [20]; 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.

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. However in Computational Social Choice, domains are considered to be non-empty (see paragraph preceding Example 2.6).

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 [15]). Local non-dictatorial domains were introduced in [12] 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 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 [1]). 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 [18]. 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 [21]).

Our result can be interpreted as verifying that 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 [16] in avoiding the *discursive dilemma*, the characterization of *safe agendas* by Grandi and Endriss [10] and that of Endriss and de Haan [9] concerning the *winner determination problem*. 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 [5] 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. [12] for domains in the non-Boolean framework. Second, a characterization of local possibility domains proven by Kirousis et al. in [12]. Lastly, the "unified framework for structure identification" by Zanuttini and Hébrard [21] (see next section for definitions).

Due to space restrictions, most proofs are omitted and can instead be found in [3].

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, 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 [13]).

▶ **Definition 2.1.** *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.

▶ **Example 2.2.** 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, which will make 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 [2] being a relatively recent such example. The original non-linear one was given by Lewis [13].

We now proceed with introducing several syntactic types of formulas:

▶ **Definition 2.3.** *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 2.4.** 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 3.1 in Subsection 3.1).

We now introduce the following notions:

▶ **Definition 2.5.** *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 2.5 will be called an *admissible set of variables*. 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). However, in Computational Social Choice, 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 2.6.** 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 2.7.** *A formula is called* renamable partially Horn *if some of its variables can be renamed (in the sense of Definition 2.1) 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 2.8.** 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: $\phi_6 = (\neg x_1 \vee x_2 \vee x_3 \vee x_4) \wedge (x_1 \vee \neg x_2 \vee \neg x_3) \wedge \neg x_4$ is easily not partially Horn, but by renaming x_4, we obtain the partially Horn formula $\phi_6^* = (\neg x_1 \vee x_2 \vee x_3 \vee \neg x_4) \wedge (x_1 \vee \neg x_2 \vee \neg x_3) \wedge x_4$, where $V_0 = \{x_4\}$ is the set of admissible variables. One 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 3.3 in Subsection 3.1 that checking whether a formula is renamable partially Horn can be done in linear time in the length of the formula.

▶ Remark 2.9. 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 $\mathcal{C}_0$. ◇

▶ **Definition 2.10.** *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 3.4 in Subsection 3.1, 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, 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 [17] and Zanuttini and Hébrard [21], we find the following definitions:

▶ **Definition 2.11.** *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 sub-section 3.2, we use this notion in order to efficiently construct formulas whose set of models is a possibility domain.

We now come to some notions from Social Choice Theory (for an introduction, see e.g. List [14]). 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 $x_j^i, i = 1, \ldots, k, j = 1, \ldots, n$ 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 Social Choice Theory, $D \subseteq \{0,1\}^n$ is said to have a k-ary (of arity k) unanimous aggregator if there are 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.

An aggregator $(f_1, \ldots, f_n)$ 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.

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 2.12.** *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 [12, Theorem 3.7] (a result that follows from Dokow and Holzman [5], but without being explicitly mentioned there), we can easily get that:

▶ **Theorem 2.13** (Dokow and Holzman [5])**.** *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*.*

Nehring and Puppe [15] 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 $f_j \neq \mathrm{pr}^k_d$, for all $d \in \{1, \ldots, k\}$ and $j = 1, \ldots, n$.

▶ **Definition 2.14.** *D is a* local possibility domain (lpd) *if it admits a locally non-dictatorial aggregator.*

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. In [12], the following characterization of lpd's has been proven:

▶ **Theorem 2.15** (Kirousis et al. [12])**.** *$D \subseteq \{0,1\}$ 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 $j = 1, \ldots, n$.*

3 Syntactic characterization of possibility domains by possibility integrity constraints

3.1 Identifying possibility integrity constraints

In this subsection, we show that identifying possibility integrity constraints can be done in time linear in the length of the input formula. By Definition 2.10, it suffices to show that for separable and renamable partially Horn formulas, 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 3.1.** *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. (Sketch; detailed proof provided in [3].) Let the variables of ϕ be the vertices of a simple graph G. We connect two such vertices 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. ◀

To deal with renamable partially Horn formulas, we will start with Lewis' idea [13] 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 3.2.** *For every formula ϕ, there is a formula ϕ' such that ϕ is renamable partially Horn if and only if ϕ' is satisfiable.*

Proof. (Sketch; detailed proof provided in [3].) For each variable $x \in V$, we introduce a new variable x'. Intuitively, setting $x = 1$ means that x is renamed, 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).

Suppose that a clause C 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 an admissible clause. Also, we should not leave the latter out of V_0, since we would have a variable of V_0 appearing positively in a clause containing a variable not in V_0. Thus, we have that $x' \to y'$, which is expressed by the equivalent clause $(\neg x' \vee y')$. We add this clause to ϕ' and we proceed in this way for any possible combination of literals in a clause of ϕ. We also introduce the clauses $\neg x \vee \neg x'$, for all $x \in V$, in order to exclude the assignment $x = x' = 1$. Finally, we add the clause $\bigvee_{x \in V'} x$, to ensure that at least one variable is admissible. ◀

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 3.3.** *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.*

Proof. (Sketch; detailed proof provided in [3].) To prove Theorem 3.3, 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. One set is comprised of the variables of ϕ', and the other of the clauses of ϕ. Each variable is connected with the clauses it appears in. 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 does not contain both a variable x and x' (and thus allows x to be admissible) *if and only if* ϕ is renamable partially Horn. ◀

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

▶ **Theorem 3.4.** *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.*

3.2 Syntactic Characterization of possibility domains

In this subsection, we provide a syntactic characterization for possibility domains, by proving they are the models of possibility integrity constraints. Furthermore, we show that given a possibility domain D, we can produce a possibility integrity constraint, whose set of models is D, in time polynomial in the size of D. To obtain the characterization, we proceed as follows. We separately show that each type of a possibility integrity constraint of Definition 2.10 corresponds to one of the conditions of Theorem 2.13: (i) Domains admitting non-dictatorial binary projection aggragators 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.

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 $D_{-I} := D_{\{1,\dots,n\}\setminus I}$. Also, for two (partial) vectors $a = (a_1, \dots, a_k) \in D_{\{1,\dots,k\}}$, $k < n$ and $b = (b_1, \dots, b_{n-k}) \in D_{\{k+1,\dots,n\}}$, we define their *concatenation* to be the vector $ab = (a_1, \dots, a_k, b_1, \dots, b_{n-k})$. 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 $D = \{(d_{j_1}, \dots, d_{j_n}) \mid (d_1, \dots, d_n) \in D'\}$, where $\{j_1, \dots, j_n\} = \{1, \dots, n\}$.

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

▶ **Proposition 3.5.** *D admits a binary non-dictatorial projection aggregator $(f_1, \dots, f_n)$ if and only if there exists a separable formula ϕ whose set of models equals D.*

Proof. (Sketch; detailed proof provided in [3].) First prove that D admits a binary non-dictatorial projection aggregator if an only if there exists a partition (I, J) of $\{1, \dots, n\}$ such that $D \approx D_I \times D_J$. To do that, take I to be the indices of the aggregator that are projections to the first coordinate and J that of the indices that are projections to the second coordinate.

Then, take the formulas ϕ_1 and ϕ_2 such that $\mathsf{Mod}(\phi_1) = D_I$ and $\mathsf{Mod}(\phi_2) = D_J$ and prove that $\mathsf{Mod}(\phi_1 \wedge \phi_2) = D$. ◀

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

▶ **Theorem 3.6.** *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.*

Proof. (Sketch; detailed proof provided in [3].) We first show that any domain D admitting a binary aggregator that has some components being projections to different coordinates, also admits one whose projections are all to the same coordinate. Also, given a domain admitting a binary aggregator with some components being $\vee$, we construct a domain D^* admitting a binary aggregator that all of its symmetric components are $\wedge$.

We then proceed to describing a partially Horn formula $\phi = \phi_0 \wedge \phi_1$, such that ϕ_0 is Horn and describes D_I, the projection of D to the indices corresponding to the symmetric components of $(f_1, \ldots, f_n)$. ϕ_1 is then constructed as the conjunction of smaller formulas that describe the sets of partial vectors that extend those of D_I. We also ensure that any variable of ϕ_0 appears only negatively in ϕ_1. ◀

We thus get:

▶ **Theorem 3.7.** *D is a possibility domain if and only if there exists a possibility integrity constraint ϕ whose set of models equals D.*

Proof. (Sketch; detailed proof provided in [3].) The proof follows by combining the characterization of possibility domains of Theorem 2.13, with Proposition 3.5 for separable formulas, Theorem 3.6, for renamable partially Horn formulas and the fact that an affine domain is described by an affine formula. ◀

To finish this section, we will use Zanuttini and Hébrard's "unified framework" [21]. Recall the definition of a prime formula (Def. 2.11) and consider the following proposition:

▶ **Proposition 3.8.** *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. (Sketch; detailed proof provided in [3].) Follows from the fact that neither *resolution* nor *omission* can destroy separability or make an admissible variable non-admissible (see also Quine [17]). ◀

We are now ready to prove our main result:

▶ **Theorem 3.9.** *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 [21, 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 $\mathsf{Mod}(\phi) = D$. If it isn't, we use again Zanuttini and Hébrard's algorithm [21] to produce, in time $O(|D|^2 n^2)$, a prime formula ϕ with $O(|D|n)$ clauses, such that $\mathsf{Mod}(\phi) = D$. Then, we use the linear algorithms of Proposition 3.1 and Theorem 3.3 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 3.7, D is a possibility domain. Else, by Proposition 3.8, D is not a possibility domain. ◀

4 Local possibility domains

We turn now our attention to local possibility domains. As in the case of possibility domains, we want to characterize lpd's with a syntactic type of formulas.

We will first address a technical issue. 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, $j = 1, \ldots, t$, $v_1, \ldots, v_s \in V$, $v_{s+1}, \ldots, v_t \in V'$ and $0 \leq s < t$. Such a clause if 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 definition, which is analogous to Definition 2.10.

▶ **Definition 4.1.** *A formula ϕ is a* local possibility integrity constraint (lpic) *if there are three* pairwise disjoint *subsets $V_0, V_1, V_2 \subseteq V$, with $V_0 \cup V_1 \cup V_2 = V$, where no clause contains variables both from V_1 and V_2 and 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,*
2. *any clause contains at most two variables from V_1 and*
3. *the clauses containing variables from V_2 are (V_0, V_2)-generalized clauses.*

▶ **Example 4.2.** Easily, every (renamable) Horn, bijunctive or affine formula is an lpic. On the other hand, consider the following possibility integrity constraint:

$$\phi = (\neg x_1 \vee x_2 \vee x_3 \vee x_4) \wedge (\neg x_2 \vee \neg x_3 \vee \neg x_4).$$

ϕ is partially Horn, since it has the pure negative literal $\neg x_1$ and thus a possibility integrity constraint. But, it is not an lpic, since however we define V_0, V_1, either there will be a variable of V_0 with a positive appearence in a non-admissible clause (even after any possible renaming of the variables of V_0) and/or there will be a clause with more than two literals from V_1. ◇

By Definition 4.1, we get the following corollary:

▶ **Corollary 4.3.** *If ϕ is a local possibility integrity constraint, then it is also a possibility integrity constraint.*

Proof. Let V_0, V_1 and V_2 be as in Definition 4.1. If $V_0 \neq \emptyset$, ϕ is partially Horn. Else, if $V_0 = V_1 = \emptyset$, then ϕ is affine. On the other hand, if $V_0 = \emptyset$ and V_1 and V_2 are not, ϕ is separable. Finally, if $V_1 = V$, then ϕ is bijunctive and equivalently, 2-SAT. The result now follows by the fact that any 2-SAT formula is renamable Horn. 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. ◀

The first theorem we prove is that we can recognize lpic's efficiently.

▶ **Theorem 4.4.** *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 local possibility constraint, or alternatively, produces the sets V_0, V_1, V_2 described in Definition 4.1.*

Proof. (Sketch; detailed proof provided in [3].) By Theorem 3.3, we check if ϕ is renamable partially Horn in time linear to its length and obtain the set V_0 of admissible variables, in case it is. Then, we can trivially check if any sub-clause, obtained by a non-admissible clause by deleting any variable from V_0 is bijunctive or affine. ◀

We now syntactically characterize lpd's as the sets of models of lpic's.

▶ **Theorem 4.5.** *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 $\mathsf{Mod}(\phi) = D$.*

Proof. (Sketch; detailed proof provided in [3].) The proof is a variation of the one for Theorem 3.6. ◀

We end this section by showing that, given an lpd D, we can efficiently construct an lpic ϕ such that $\mathsf{Mod}(\phi) = D$.

▶ **Theorem 4.6.** *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.*

Proof. (Sketch; detailed proof provided in [3].) Using Zanuttini and Hébrard's unified framework, we compute a prime formula ϕ such that $\mathsf{Mod}(\phi) = D$ in time $O(|D|^2 n^2)$ that contains $O(|D|n)$ clauses. We then check, in linear time to the length of ϕ, whether it is an lpic. ◀

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 give a necessary and sufficient condition on the syntactic form of a formula to be an integrity constraint of a domain that accepts a (locally) non-dictatorial aggregator. We called such formulas, (local) possibility integrity constraints. Our results are algorithmic, in the sense that (i) recognizing a (local) possibility integrity constraint can be implemented in time linear in the length of the input formula and (ii) given a (local) possibility domain, a corresponding (local) possibility 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.

References

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

2 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.

3 Josep Díaz, Lefteris Kirousis, Sofia Kokonezi, and John Livieratos. Algorithmically Efficient Syntactic Characterization of Possibility Domains. *arXiv preprint*, 2019. `arXiv:1901.00138`.
4 Franz Dietrich. A generalised model of judgment aggregation. *Social Choice and Welfare*, 28(4):529–565, 2007.
5 Elad Dokow and Ron Holzman. Aggregation of binary evaluations for truth-functional agendas. *Social Choice and Welfare*, 32(2):221–241, 2009.
6 Elad Dokow and Ron Holzman. Aggregation of binary evaluations. *Journal of Economic Theory*, 145(2):495–511, 2010.
7 Ramez Elmasri and Sham Navathe. *Fundamentals of database systems*. Pearson London, 2016.
8 Herbert Enderton and Herbert B Enderton. *A mathematical introduction to logic*. Elsevier, 2001.
9 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.
10 Umberto Grandi and Ulle Endriss. Binary aggregation with integrity constraints. In *IJCAI Proceedings-International Joint Conference on Artificial Intelligence*, volume 22, page 204, 2011.
11 Umberto Grandi and Ulle Endriss. Lifting integrity constraints in binary aggregation. *Artificial Intelligence*, 199:45–66, 2013.
12 Lefteris Kirousis, Phokion G Kolaitis, and John Livieratos. Aggregation of votes with multiple positions on each issue. In *Proceedings 16th International Conference on Relational and Algebraic Methods in Computer Science*, pages 209–225. Springer, 2017. Expanded version to appear in *ACM Transactions on Economics and Computation*.
13 Harry R Lewis. Renaming a set of clauses as a Horn set. *Journal of the ACM (JACM)*, 25(1):134–135, 1978.
14 Christian List. The theory of judgment aggregation: An introductory review. *Synthese*, 187(1):179–207, 2012.
15 Klaus Nehring and Clemens Puppe. Abstract arrowian aggregation. *Journal of Economic Theory*, 145(2):467–494, 2010.
16 Gabriella Pigozzi. Belief merging and the discursive dilemma: an argument-based account to paradoxes of judgment aggregation. *Synthese*, 152(2):285–298, 2006.
17 Willard V Quine. On cores and prime implicants of truth functions. *The American Mathematical Monthly*, 66(9):755–760, 1959.
18 Thomas J. Schaefer. The complexity of satisfiability problems. In *Proc. of the 10th Annual ACM Symp. on Theory of Computing*, pages 216–226, 1978.
19 Robert Wilson. On the theory of aggregation. *Journal of Economic Theory*, 10(1):89–99, 1975.
20 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.
21 Bruno Zanuttini and Jean-Jacques Hébrard. A unified framework for structure identification. *Information Processing Letters*, 81(6):335–339, 2002.