# Explaining Multivariate Decision Trees: Characterising Tractable Languages

CLÉMENT CARBONNEL, CNRS, University of Montpellier, France
MARTIN C. COOPER*, IRIT, University of Toulouse, France
EMMANUEL HEBRARD, LAAS CNRS, France
DANY MORALES, IRIT, University of Toulouse, France
JOÃO MARQUES-SILVA, ICREA, University of Lleida, Spain

We study multivariate decision trees (MDTs), in particular, classes of MDTs determined by the language of relations that can be used to split feature space. An abductive explanation (AXp) of the classification of a particular instance, viewed as a set of feature-value assignments, is a minimal subset of the instance which is sufficient to lead to the same decision. We investigate when finding a single AXp is tractable. We identify tractable languages for real, integer and boolean features. Indeed, in the case of boolean languages, we provide a P/NP-hard dichotomy. We extend this dichotomy to languages defined by formulas whose literals correspond to splits of ordered domains of arbitrary finite size. Experiments indicate that MDTs can provide more compact models than classical decision trees while conserving accuracy and explainability.

**JAIR Associate Editor:**

**JAIR Reference Format:**
Clément Carbonnel, Martin C. Cooper, Emmanuel Hebrard, Dany Morales, and João Marques-Silva. 2025. Explaining Multivariate Decision Trees: Characterising Tractable Languages. *Journal of Artificial Intelligence Research* 0, Article 0 ( 2025), 27 pages. DOI: 10.1613/jair.1.xxxxx

## 1 Background

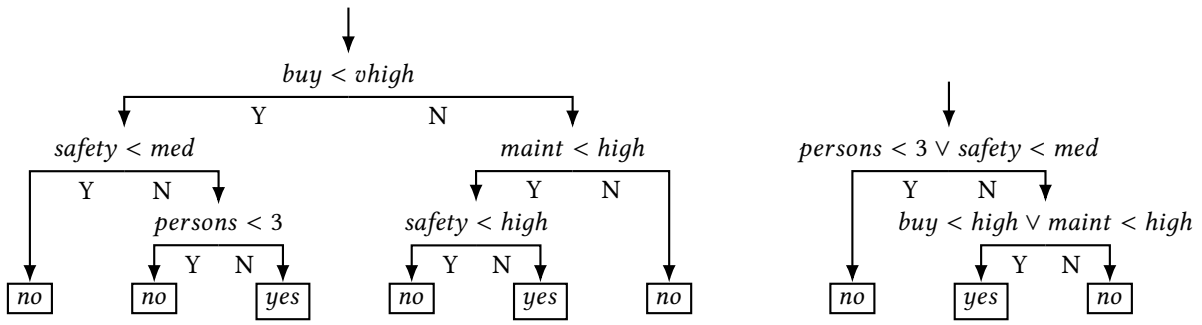
Decision trees (DTs) are a classical family of ML models. Due to their relative simplicity, DTs are often considered to be interpretable (with the implicit assumption that they are shallow enough to allow each decision to be explained by the values of a small number of features). There is considerable interest in their multivariate extension (MDTs) in which feature-space is split according to conditions on several features rather than on a single feature [9, 49, 12, 22, 34]. For example, in oblique DTs these conditions are linear inequalities [25, 41, 5, 48, 16, 24, 44]. Some authors have extended oblique DTs by building DTs with non-linear conditions [21] whereas others have limited the linear conditions to at most two features [7]. In this paper we study families of MDTs, parameterized by the language of possible multivariate conditions, from the point of view of the tractability of explaining decisions.

A (possibly multivariate) DT is a binary tree such that each leaf is labelled by a class and each internal node has two child nodes. At each internal node, the two edges to its child nodes are labelled by a boolean condition $C$

*Corresponding Author.

Authors' Contact Information: Clément Carbonnel, ORCID: 0000-0003-2312-2687, clement.carbonnel@lirmm.fr, CNRS, University of Montpellier, France; Martin C. Cooper, ORCID: 0000-0003-4853-053X, cooper@irit.fr, IRIT, University of Toulouse, France; Emmanuel Hebrard, ORCID: 0000-0003-3131-0709, hebrard@laas.fr, LAAS CNRS, Toulouse, France; Dany Morales, IRIT, University of Toulouse, France; João Marques-Silva, ORCID: 0000-0002-6632-3086, jpms@icrea.cat, ICREA, University of Lleida, Spain.


DOI: 10.1613/jair.1.xxxxx




(a) Optimal DT of depth 3 (b) Optimal 2CNF-MDT of depth 2

Fig. 1. Optimal decision tree of depth 3 and 2CNF-Multivariate decision tree for UCI data set **car**. Features are 'buying price' and 'maintainance price' in {'low', 'med', 'high', 'very high'}; 'doors' and 'persons' in $\{2, 3, 4, \geq 5\}$; 'luggage boot' in {'small', 'med', 'big'}; and 'safety' in {'low', 'med', 'high'}. The classification task is to decide whether we should purchase the car.

and its complement $\neg C$. In classical DTs, conditions are univariate, i.e. $x_i \in S$ for some feature $x_i$ and some subset $S$ of the domain of feature $x_i$. Indeed, for simplicity of learning algorithms, it is often assumed that the domains are totally ordered and the only conditions are splits of the form $x_i \geq a$ for constants $a$.

We suppose a set of features $x_1, \ldots, x_n$ and a set of classes $\mathbb{C}$. Feature space, denoted $\mathbb{F}$ is the cartesian product of the feature domains. An instance (feature-vector) is an element of $\mathbb{F}$ and a classifier is a function $\kappa : \mathbb{F} \to \mathbb{C}$.

A multivariate condition can be seen as a *constraint* which can be decomposed into its *scope* (a list $\ell$ of features) and its *relation* of arity $|\ell|$. This allows us to study multivariate decision trees according to the language of possible constraint relations. For simplicity of presentation, we assume a unique domain $D$ for all features. A relation of arity $k$ is a subset of $D^k$. A *relational language* $\mathcal{L}$ is simply a set of relations.

Definition 1. *A* multivariate decision tree *is a decision tree in which the condition tested at a node is a constraint on any number of features. An $\mathcal{L}$-DT is a multivariate decision tree in which the constraint relations belong to the language $\mathcal{L}$.*

A multivariate DT may be exponentially smaller than a DT. Consider the case of a parity function $\kappa$ on $n$ boolean features: trivially an $\mathcal{L}$-DT of depth one can capture this function provided the relation corresponding to the constraint $\kappa(x) = 1$ belongs to the language $\mathcal{L}$, whereas a classical DT would necessarily be of exponential size. For instance, we give in Figure 1 an optimal decision tree of depth 3 and an optimal multivariate decision of depth 2 (restricted to disjunctions of two features) for UCI's data set "**car**"[1]. Both have similar accuracies (the test accuracy of a depth-2 2CNF-MDT is in average 1% higher than that of a depth-3 DTs on this data set). However, the MDT is shorter and arguably easier to interpret. In particular, it makes it very clear that the most important features are safety and number of passengers.

Tractable constraint languages have been investigated in the context of the Constraint Satisfaction Problem (CSP). A CSP instance consists of a set of $n$ variables, each with its domain, together with a set of constraints, where each constraint is defined by its scope (a list of variables) and the relation that must hold on the variables in this scope. The decision version of the CSP consists in determining whether there exists some assignment to all $n$ variables in the cartesian product of the domains that satisfies all the constraints. Given a language $\mathcal{L}$ of relations, CSP($\mathcal{L}$) is the subproblem of the decision version of the CSP in which all relations belong to the

[1] https://archive.ics.uci.edu/dataset/19/car+evaluation

language $\mathcal{L}$. The languages $\mathcal{L}$ we consider are, as is classical in CSPs, arbitrary sets of relations that can apply to any variables/features.

As we will see later, testing whether a subset of the feature assignments comprising the instance is sufficient to explain the decision involves solving a constraint satisfaction problem consisting of the conditions along each path to a leaf corresponding to a different decision. In classical DT's these conditions are unary and the resulting CSP is trivial, but for multivariate conditions, the resulting CSP is, in general, NP-hard. We will see the close relationship between tractability of explaining $\mathcal{L}$-DTs and the tractability of CSP($\mathcal{L}$). However, there is an important difference. In an MDT, for each edge corresponding to the satisfaction of a relation $R$ there is an alternate edge corresponding to its complement relation $\neg R$. It follows that in the context of MDTs, it is important to study languages *closed under complement*: languages $\mathcal{L}$ such that $R \in \mathcal{L} \Rightarrow \neg R \in \mathcal{L}$. There is large body of work on the characterisation of languages $\mathcal{L}$ for which CSP($\mathcal{L}$) $\in$ P, culminating in a dichotomy theorem in the finite-domain case [10, 50]. This result implies a similar dichotomy for finite languages closed under complement, but the dichotomy criterion does not provide an *explicit* description of the tractable cases.

Although DTs are sometimes considered to be inherently interpretable, it has recently been shown that DT paths can exhibit significant redundancy, both in theory and in practice, when considered as explanations of decisions [29]. In this paper, we therefore study the notion of abductive explanation (AXp) [46, 28] which can provide a more succinct explanation of a particular decision than the (M)DT path corresponding to the decision [29].

Definition 2. *Let $\kappa$ be a classifier and $\mathbf{v}$ a feature-vector. A* weak AXp *(weak abductive explanation) of the decision $\kappa(\mathbf{v}) = c$ is a subset $S$ of the features such that any assignment $\mathbf{y}$ that agrees with $\mathbf{v}$ on the features in $S$ satisfies $\kappa(\mathbf{y}) = c$. An* AXp *of a decision is a subset-minimal weak AXp.*

We study the problem of finding subset-minimal abductive explanations since this is known to be achievable in polynomial time for DTs whereas finding a minimum-cardinality abductive explanation is NP-hard for DTs [4]. Indeed, this latter problem is W[2]-hard when explanation size is the parameter [42].

A related notion is that of contrastive explanation which explains how to change a decision [27].

Definition 3. *Let $\kappa$ be a classifier and $\mathbf{v}$ a feature-vector. A* weak CXp *(weak contrastive explanation) of the decision $\kappa(\mathbf{v}) = c$ is a subset $S$ of the features such that some assignment $\mathbf{y}$ that differs from $\mathbf{v}$ only on features in a subset of $S$ satisfies $\kappa(\mathbf{y}) \neq c$. A* CXp *of a decision is a subset-minimal weak CXp.*

There is a interesting duality between AXps and CXps of a decision: the AXps are the minimal hitting sets of the CXps (and vice versa) [27]. Given the close link between the complexity of finding an AXp or a CXp [19], for simplicity of presentation, we will concentrate on AXps in this paper.

Example 1. *Consider the classifier $\kappa(\mathbf{x}) = \neg x_1 \vee (x_2 \wedge (\neg x_3 \vee x_4))$ where $x_i \in \{0, 1\}$ $(i = 1, \ldots, 4)$ are boolean features. $\kappa$ can be represented by the decision tree in Figure 2. An abductive explanation (AXp) for the decision $\kappa(1, 1, 1, 1) = 1$ is $\{x_2, x_4\}$ since any feature-vector $\mathbf{y}$ with $y_2 = y_4 = 1$ satisfies $\kappa(\mathbf{y}) = 1$ (but neither $y_2 = 1$ nor $y_4 = 1$ alone is sufficient to guarantee $\kappa(\mathbf{y}) = 1$). This AXp is half the length of the path in the DT of Figure 2 corresponding to this decision (i.e. the leftmost path). To see that $\{x_1, x_4\}$ is a weak AXp it suffices to observe that $x_2 = x_4 = 1$ is incompatible with the two paths leading to leaves labelled 0.*

Example 2. *Consider the classifier $\kappa : \{0, 1\}^6 \rightarrow \{0, 1\}$ defined by the MDT in Figure 3. An abductive explanation (AXp) for the decision $\kappa(1, 1, 1, 1, 1, 1) = 1$ is $\{x_1, x_2, x_6\}$ since any feature-vector $\mathbf{y}$ with $y_1 = y_2 = y_6 = 1$ satisfies $\kappa(\mathbf{y}) = 1$ (but none of $y_1 = y_2 = 1$ or $y_1 = y_6 = 1$ or $y_2 = y_6 = 1$ alone is sufficient to guarantee $\kappa(\mathbf{y}) = 1$). This AXp consists of half the features tested along the path in the MDT of Figure 3 corresponding to this decision i.e. the path $(\neg x_1 \vee \neg x_2 \vee \neg x_3 \vee x_6)$, $(\neg x_1 \vee \neg x_4 \vee x_5)$, $(x_5 \wedge x_6)$, $(x_1 \wedge x_2)$. To see that $\{x_1, x_2, x_6\}$ is a weak AXp of $\kappa(1, 1, 1, 1, 1, 1) = 1$, it suffices to verify that $x_1 = x_2 = x_6 = 1$ is incompatible with the three paths leading to leaves*

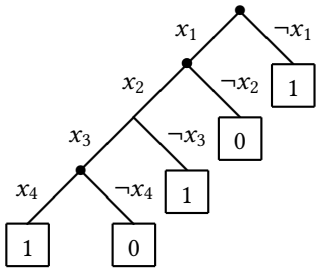



Fig. 2. A decision tree corresponding to the classifier $\kappa(\mathbf{x}) = \neg x_1 \vee (x_2 \wedge (\neg x_3 \vee x_4))$.

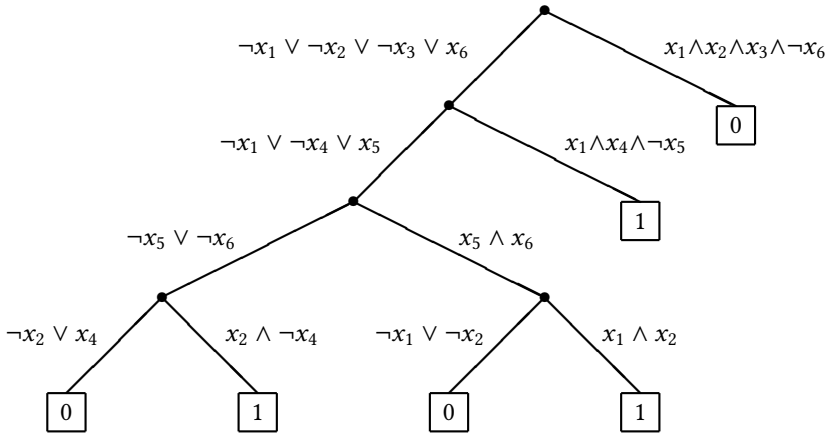



Fig. 3. A multivariate decision tree corresponding to the classifier $\kappa(\mathbf{x}) = (\neg x_1 \vee \neg x_2 \vee \neg x_3 \vee x_6) \wedge ((x_1 \wedge x_4 \wedge \neg x_5) \vee (x_1 \wedge x_2 \wedge x_5 \wedge x_6) \vee (x_2 \wedge \neg x_4 \wedge (\neg x_5 \vee \neg x_6))$

*labelled 0. For example, $x_1 = x_2 = x_6 = 1$ is incompatible with the leftmost path and this can be verified by unit propagation since $(\neg x_1 \vee \neg x_2 \vee \neg x_3 \vee x_6) \wedge (\neg x_1 \vee \neg x_4 \vee x_5) \wedge (\neg x_5 \vee \neg x_6) \wedge (\neg x_2 \vee x_4) \wedge x_1 \wedge x_2 \wedge x_6$ is an instance of HORNSAT.*

Example 3. *Consider the classifier $\kappa : \{0,5\}^5 \to \{0,1\}$ defined by the MDT in Figure 4. An abductive explanation (AXp) for the decision $\kappa(2,2,2,2,2) = 1$ is $\{x_3, x_4\}$ since any feature-vector $\mathbf{y}$ with $y_3 = y_4 = 2$ satisfies $\kappa(\mathbf{y}) = 1$ (but neither $y_3 = 2$ nor $y_4 = 2$ alone is sufficient to guarantee $\kappa(\mathbf{y}) = 1$). This AXp consists of half the features tested along the path in the MDT of Figure 4 corresponding to this decision i.e. the path ($x_1 \geq 3 \vee x_3 < 5$), ($x_4 \geq 3 \vee x_5 < 3$), ($x_5 < 4 \wedge x_1 < 3$), ($x_1 < 4 \wedge x_5 < 3$). To see that $\{x_3, x_4\}$ is a weak AXp of $\kappa(2,2,2,2,2) = 1$, it suffices to verify that $x_3 = x_4 = 2$ is incompatible with the three paths leading to leaves labelled 0. We will see in Example 8 that these incompatibility tests again belong to a tractable class.*

Example 4. *Consider the MDT classifier $\kappa$ depicted in Figure 5. An abductive explanation (AXp) for the decision $\kappa(buy = low, maint = med, persons = 2, safety = med, boot = med) = no$ is $\{persons\}$ since any car with*

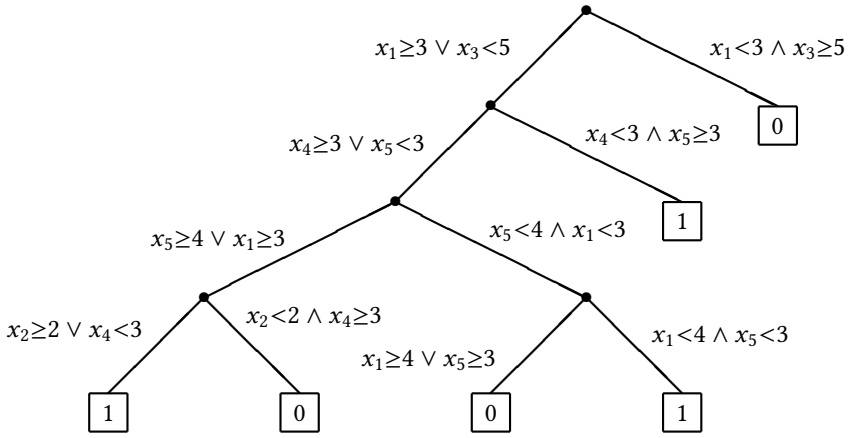



Fig. 4. A multivariate decision tree with conditions on pairs of integer variables.

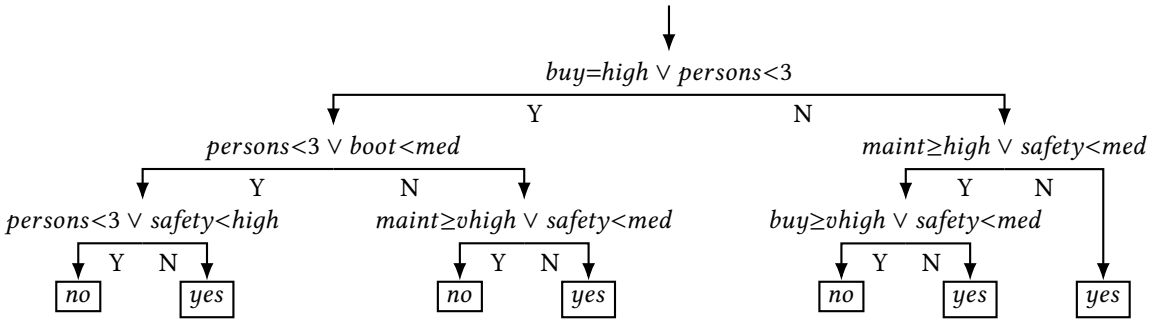



Fig. 5. A multivariate decision tree of depth 3 for a classifier trained on UCI data set **car**.

*only two seats (persons = 2) will follow the leftmost path leading to label 'no'. Similarly, an AXp of the decision* $\kappa(buy = low, maint = med, persons = 4, safety = low, boot = med) = no$ *is* $\{safety\}$ *since any instance with* $safety = low$ *leads to a leaf labelled 'no'. In both cases, the size of the AXp is one whereas the number of features tested along the path to the leaf is four. The required tests to verify that these are indeed AXps belong to the tractable class described in Example 12.*

The need to apply formal reasoning to explainable artificial intelligence (XAI), and in particular to decisions taken by ML models, has been pointed out by many researchers [1, 23, 38, 39, 40]. The computational complexity of finding abductive explanations is an active field of research in the application of formal reasoning to explaining decisions taken by classifiers [2, 3, 4, 6, 11, 19, 26, 28, 47]. Izza et al [29] showed that finding an AXp of a decision taken by a DT is in P. (This corresponds to the case in which all constraints are *unary* , i.e. of the form $x_i \in S$ for some subset $S$ of the domain of $x_i$). In this paper we explore the tractability of this problem for MDTs parameterised by the constraint language $\mathcal{L}$. We show that, in general, this problem is NP-hard, but that there are nonetheless many interesting tractable cases.

Let wAXpDT($\mathcal{L}$) denote the problem of deciding whether a set of features is a weak AXp for a given decision taken by a $\mathcal{L}$-DT, where $\mathcal{L}$ is a language of constraint relations. As we show in Section 2, whenever wAXpDT($\mathcal{L}$) $\in$ P, there is a polynomial-time algorithm to find an AXp: starting with the set of all features, for each feature test whether deleting the feature still leaves a weak AXp [17, 19].

When $\mathcal{L}$ is the set of unary constraints, then an $\mathcal{L}$-DT can be viewed as a classical DT. In this case, wAXpDT($\mathcal{L}$) is known to be tractable [29]. After identifying, in Section 2, several languages $\mathcal{L}$ for which wAXpDT($\mathcal{L}$) is tractable, in Sections 3 and 4, we describe a dichotomy theorem in the case of boolean languages. In Section 5 we extend this dichotomy to formulas based on literals corresponding to cuts in arbitrary finite ordered domains. Finally, in Section 6 we report experiments to compare DTs and MDTs. The conference version of this paper [14] only covered the dichotomy theorem over boolean domains.

## 2 Tractable Explaining of MDT Decisions

We begin by recalling a simple algorithm to find minimal subsets satisfying a monotone property [17]. In this context, a property is a boolean function on the powerset of features. We say that a property $\mathcal{H}$ is *monotone* if for all sets $S \subseteq T$, $\mathcal{H}(S) \Rightarrow \mathcal{H}(T)$.

Lemma 1. *Given a monotone property $\mathcal{H}$ that can be tested in polynomial time and an initial finite set $S_0$ satisfying $\mathcal{H}$, a minimal subset $S$ of $S_0$ satisfying $\mathcal{H}$ can be found in polynomial time.*

Proof. The following so-called 'deletion' algorithm finds a minimal $S \subseteq S_0$ by testing $|S_0|$ times the property $\mathcal{H}$.

for each element $e \in S_0$ :
    if $\mathcal{H}(S \setminus \{e\})$ then $S \leftarrow S \setminus \{e\}$

□

The following corollary follows from the fact that being a weak AXp is a monotone property and that the set of all features is trivially a weak AXp (and hence can be used as the initial set $S_0$ in the deletion algorithm).

Corollary 1. *For any family of classifiers, finding a single AXp is polytime if testing whether a subset of features is a weak AXp is in* P.

We assume that an $\mathcal{L}$-DT is represented as a binary tree in which each leaf node is labelled by a class and each internal node is linked to its two child-nodes by edges labelled respectively by a relation $R \in \mathcal{L}$ and its complement $\neg R \in \mathcal{L}$. The assumption of an explicit representation of $\neg R$ avoids technical issues related to the possible large disparity between the sizes of the explicit representation of $\neg R$ and its implicit representation as the complement of $R$. In the following proposition, we do not impose a fixed representation of relations (as a table of tuples or as a formula) but we do assume the same representation of relations in CSP($\mathcal{L}$) and in $\mathcal{L}$-DTs.

Given an MDT, we use the notation path($\alpha$) to represent the set of conditions satisfied on the path from the root to a leaf $\alpha$. Let *Asst* represent all unary constraints consisting of assignments, i.e. $x_i = u$ for some feature $x_i$ and some constant $u$. We can view a feature-vector $\mathbf{v}$ as a set of literals (i.e. variable-value assignments). For a fixed feature-vector $\mathbf{v}$, it will be convenient to associate a set $X$ of features with the partial assignment $\mathbf{v}_X$, the set of literals corresponding to the subset of $\mathbf{v}$ on these variables.

Proposition 1. *Let $\mathcal{L}$ be a constraint language such that $\mathcal{L}$ is closed under complement. Suppose that $\mathcal{L} \cup Asst \subseteq C$ where CSP(C) $\in$ P. Then* wAXpDT($\mathcal{L}$) $\in$ P *and an AXp of any decision taken by an $\mathcal{L}$-DT can be found in polynomial time.*

Proof. Let $\kappa$ be the classifier defined by an $\mathcal{L}$-DT and consider a decision $\kappa(\mathbf{v}) = c$ to be explained. By Corollary 1, we only need to show that we can test that a set of features $X$ is a weak AXp in polynomial time. Testing whether $X$ is a weak AXp can be achieved by testing whether for all leaves $\alpha$ corresponding to a decision different to $c$, $\mathbf{v}_X$ is incompatible with the set of constraints path($\alpha$). The constraints of path($\alpha$) are in $\mathcal{L}$. Furthermore, the partial assignment $\mathbf{v}_X$ can be viewed as a set of constraints in $Asst$, so this test of incompatibility is a CSP with constraints in $\mathcal{L} \cup Asst$, and hence, by the hypotheses $\mathcal{L} \cup Asst \subseteq C$ and CSP($C$) $\in$ P, is solvable in polynomial time. □

In all the following examples (Examples 5-10), $\mathcal{L}$ is closed under complement, $\mathcal{L} \cup Asst \subseteq C$ where CSP($C$) $\in$ P, and so Proposition 1 applies.

*Boolean domains.* We begin with examples in which features are boolean. Two well-known boolean languages $C$ for which CSP($C$) is tractable are conjunctions of Horn clauses and conjunctions of 2-clauses.

Example 5. *Let $\mathcal{L}$ be the class of Horn clauses and their negations. The complement (negation) of a Horn clause is a conjunction of unary clauses and unary clauses are trivially Horn. $C$ is the class of conjunctions of Horn clauses, and hence CSP($C$) $\in$ P since it corresponds to HORNSAT. We can observe that the constraint relations in the MDT of Figure 3 are all Horn clauses (or their negations) and hence this MDT is an $\mathcal{L}$-DT.*

Note that, in general, the complement of a *conjunction* of Horn clauses is not the conjunction of Horn clauses. In Section 4.1 we identify the maximal generalisation of the class in Example 5. It consists of a specific form of conjunctions of Horn clauses.

Example 6. *Let $\mathcal{L}$ be the class of 2-conjunctions of 2-clauses (i.e. the conjunction of at most two clauses each of which contains at most two literals) together with the complements of such constraints. The complement of a 2-conjunction of 2-clauses is also the conjunction of 2-clauses, since $\neg((a \vee b) \wedge (c \vee d)) \equiv (\neg a \vee \neg c) \wedge (\neg a \vee \neg d) \wedge (\neg b \vee \neg c) \wedge (\neg b \vee \neg d)$. $\mathcal{L} \cup Asst \subseteq C$ where $C$ is the set of conjunctions of 2-clauses. CSP($C$) $\in$ P by tractability of 2SAT.*

In general, the complement of an arbitrary conjunction of 2-clauses is not the conjunction of 2-clauses. We identify the maximal generalisation of this example in Section 4.3.

*Finite domains.* We now consider finite feature-domains of arbitrary size. Define a *two-fan* constraint to be a constraint of the form $x_i = a \vee x_j = b$, where $a, b$ are constants.

Example 7. *Let $\mathcal{L}$ be the class of two-fan constraints and their complements, together with all unary constraints $x_i \in S$ where $S$ is any subset of the domain of $x_i$. The complement of the two-fan $x_i = a \vee x_j = b$ is the constraint $x_i \neq a \wedge x_j \neq b$ which is the conjunction of two unary constraints. Let $maj : D^3 \rightarrow D$ be the function defined by $maj(a, b, c) = b$ if $b = c$ and $maj(a, b, c) = a$ if $b \neq c$. It returns the majority value among its arguments, if it exists, and its first argument otherwise. A binary relation $R$ is maj-closed if $(a_1, a_2), (b_1, b_2), (c_1, c_2) \in R \Rightarrow (maj(a_1, b_1, c_1), maj(a_2, b_2, c_2)) \in R$, and all unary constraints are maj-closed. All two-fan constraints and conjunctions of unary constraints are maj-closed. It is well known that CSP($C$) $\in$ P where $C$ is the set of maj-closed relations [18, 31].*

Now suppose that all domains are totally ordered. Define a unary inequality literal (UI-literal) to be a constraint of the form $x_i \geq a$ or $x_i < a$ where $a$ is any element of the domain of feature $x_i$ except the minimum element.

Example 8. *Let $\mathcal{L}$ be the class of (unary or) binary boolean functions $\phi$ of UI-literals. The complement (negation) of such a function is also a (unary or) binary boolean function of UI-literals. Let med be the ternary function which returns the median value among its three arguments. A binary relation $R$ is med-closed if whenever $(u_1, u_2)$, $(v_1, v_2)$, $(w_1, w_2) \in R$, we have $(med(u_1, v_1, w_1), med(u_2, v_2, w_2)) \in R$, and all unary constraints are med-closed. $med(u, v, w)$ satisfies a UI-literal $\ell$ iff the majority of $u, v, w$ satisfy $\ell$. It follows that, given two UI-literals (such as $\ell_1(x_1) := x_1 \geq a$*

*and $\ell_2(x_2) := x_2 < b$, for example) and three pairs $(u_1, u_2), (v_1, v_2), (w_1, w_2)$, the pair of values $(\ell_1(m_1), \ell_2(m_2))$ where $(m_1, m_2) = (med(u_1, v_1, w_1), med(u_2, v_2, w_2))$ will necessarily coincide with one of $(\ell_1(u_1), \ell_2(u_2))$, $(\ell_1(v_1), \ell_2(v_2))$ or $(\ell_1(w_1), \ell_2(w_2))$. Thus, any boolean function $\phi$ of the literals $\ell_1, \ell_2$ satisfied by all three pairs $(u_1, u_2)$, $(v_1, v_2)$, $(w_1, w_2)$ will also be satisfied by the pair $(med(u_1, v_1, w_1), med(u_2, v_2, w_2))$. It is well known that CSP($C$) $\in$ P where $C$ is the set of med-closed relations [30]. All the constraints in the MDT shown in Figure 4 are binary boolean functions of UI-literals and hence this is an example of an $\mathcal{L}$-DT.*

We will consider a generalisation of both Example 7 and Example 8 in Example 12. Indeed, we identify all tractable languages of boolean functions of UI-literals in Section 5.3.

*Infinite domains.* We now consider infinite domains, firstly integer domains and then real domains.

Example 9. *A unit two variable per inequality (UTVPI) constraint is of the form $ax_i + bx_j \leq d$ where $x_i$ and $x_j$ are integer variables, the coefficients $a, b \in \{-1, 0, 1\}$ and the bound $d$ is an integer constant. The negation of such a constraint is $-ax_i - bx_j \leq -(d + 1)$ and is hence also an UTVPI constraint. A unary assignment $x_i = d$ is equivalent to $x_i \leq d \ \wedge \ -x_i \leq -d$, a conjunction of UTVPI constraints. Let $\mathcal{L}$ be the set of UTVPI constraints and $C$ the class of constraints consisting of conjunctions of UTVPI constraints. Then $\mathcal{L} \cup Asst \subseteq C$ and it is known that CSP($C$) $\in$ P [36].*

Example 10. *Let $\mathcal{L}$ be the class of linear inequalities ($\leq$ or $<$) over the reals. The complement of a linear inequality is again a linear inequality and assignments $x_i = u$ can be viewed as two linear inequalities ($x_i \leq u$ and $-x_i \leq -u$). $C$ is the set of systems of linear inequalities over $\mathbb{R}$. Hence $\mathcal{L} \cup Asst \subseteq C$ and it is well known that CSP($C$) $\in$ P.*

Since an oblique decision tree is an MDT in which all conditions are linear inequalities over $\mathbb{R}$, we can deduce that there is a polynomial-time algorithm to find an AXp of a decision taken by an oblique decision tree. The dual of an abductive explanation is a contrastive explanation, a minimal set of features that, if changed, changes the output of the classifier (Definition 3). It has already been observed that an optimal contrastive explanation, known as a counterfactual explanation or adversarial example, can be found for oblique decision trees in polynomial time for a linear error function, by reduction to Linear Programming [15]. It is important to note that the tractability of finding AXps (or contrastive explanations) for oblique decision trees depends critically on the assumption that all features are real-valued.

It is known that if a family of classifiers is closed under fixing features, then the tractability of the problem of finding an AXp or a CXp coincide (Theorem 4 of [19]). It follows that, for all languages given in the examples listed above (Examples 5 to 10), Proposition 1 also implies that a CXp can be found in polynomial time.

## 3 Tractable Boolean Languages: The Algebraic Approach

We first study the characterisation of tractable languages $\mathcal{L}$ for wAXpDT($\mathcal{L}$) from an abstract algebraic point of view, before looking for a detailed characterisation.

The complexity of wAXpDT($\mathcal{L}$) is closely tied to that of CSP($\mathcal{L}$) so we will use the same machinery to prove our results. Let $f : D^k \to D$ be a function. A relation $R$ has $f$ as a *polymorphism* (we say that $R$ is closed under $f$) if $\forall t_1, \ldots, t_k \in R$, the tuple $f(t_1, \ldots, t_k)$ obtained by applying $f$ componentwise to the $k$ vectors $t_1, \ldots, t_k$ belongs to $R$. We say that a language $\mathcal{L}$ has the polymorphism $f$ if all relations in $\mathcal{L}$ are closed under $f$. The complexity of CSP($\mathcal{L}$) is known to be determined by the polymorphisms of CSP($\mathcal{L}$), up to logspace reductions [32].

For all languages $\mathcal{L}$ over finite domains, CSP($\mathcal{L}$) is either in P or NP-complete [10, 50]. For our purposes, we only need to present the dichotomy criterion in the boolean setting. This particular result is known as Schaefer's Theorem [45], and its modern formulation involves six different boolean functions.

In the following, let $f_0$ and $f_1$ denote the constant unary functions that always return 0 and 1, respectively. The binary functions max and min return respectively the maximum and minimum of their two arguments. The

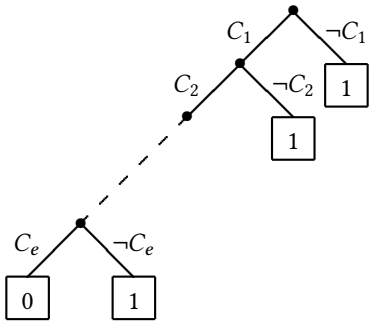



Fig. 6. A decision tree $T_I$ which has a non-empty AXp if and only if the constraints $C_1, \ldots, C_e$ are simultaneously satisfiable.

majority function $\text{maj} : \{0,1\}^3 \to \{0,1\}$ (already seen in Example 7) is given by

$$\text{maj}(x,y,z) = \begin{cases} y & \text{if } y = z \\ x & \text{otherwise} \end{cases}$$

and the minority function $\text{miny} : \{0,1\}^3 \to \{0,1\}$ is given by

$$\text{miny}(x,y,z) = \begin{cases} z & \text{if } x = y \\ \neg z & \text{otherwise.} \end{cases}$$

Theorem 1 (Schaefer [45]). *Let $\mathcal{L}$ be a boolean language. If $\mathcal{L}$ has either* 0, 1, *max, min, maj or miny as a polymorphism, then CSP($\mathcal{L}$) $\in$ P. Otherwise, CSP($\mathcal{L}$) is* NP*-complete.*

Theorem 2. *Let $\mathcal{L}$ be a finite boolean language closed under taking complements. Then, assuming* P $\neq$ NP, wAXpDT($\mathcal{L}$) $\in$ P *iff $\mathcal{L}$ has either max, min, maj or miny as a polymorphism.*

Proof. $\Leftarrow$: Suppose that $\mathcal{L}$ has either max, min, maj or miny as a polymorphism. By Theorem 1, we have CSP($\mathcal{L}$) $\in$ P. Furthermore, all unary relations have these four polymorphisms. Thus, we also have CSP($\mathcal{L} \cup Asst$) $\in$ P, and hence by Proposition 1, wAXpDT($\mathcal{L}$) $\in$ P.

$\Rightarrow$: We first give a polynomial reduction from CSP($\mathcal{L}$) to wAXpDT($\mathcal{L}$). Let $I$ be an instance of CSP($\mathcal{L}$) consisting of constraints $C_1, \ldots, C_e$. We build a DT $T_I$, shown in Figure 6, as a sequence of tests corresponding to these constraints. $C_1$ is the test at the root of $T_I$, and each $C_i$ ($i = 2, \ldots, e$) is the test at the positive child of $C_{i-1}$ (i.e. the node attained after a positive response to the test $C_{i-1}$). The positive child of $C_e$ is a leaf node labelled 0. All negative children of all nodes of $T_I$ are leaf nodes labelled 1. Let $\kappa$ be the function defined by the DT $T_I$. Now consider any decision $\kappa(\mathbf{v}) = 1$. The empty set is a weak AXp of this decision iff it is impossible to simultaneously satisfy the constraints $C_1, \ldots, C_e$, since the only leaf node labelled 0 can only be reached if all these constraints are satisfied. Thus deciding whether $\emptyset$ is a weak AXp amounts to solving $I \in$ CSP($\mathcal{L}$).

Thus, assuming P $\neq$ NP, we deduce from Theorem 1 that CSP($\mathcal{L}$) $\in$ P iff $\mathcal{L}$ has (at least) one of the six functions $f_0$, $f_1$, max, min, maj or miny as a polymorphism. A non-empty relation $R$ has polymorphism $f_a$, where $a \in \{0,1\}$, iff the tuple $(a, \ldots, a)$ (of length the arity of $R$) belongs to $R$. Consequently, a language $\mathcal{L}$ closed under taking complements cannot have $f_a$ as a polymorphism unless all relations in $\mathcal{L}$ are either empty or complete; in this case, $\mathcal{L}$ has all six of Schaefer's polymorphisms. Thus $\mathcal{L}$ has either max, min, maj or miny as a polymorphism. □

In the general (non-boolean) case, the two reductions behind Proposition 1 and Theorem 2 imply that wAXpDT($\mathcal{L}$) is at most as hard as CSP($\mathcal{L} \cup Asst$) and at least as hard as CSP($\mathcal{L}$). However, in contrast to

boolean domains, there exists a wide array of languages $\mathcal{L}$ such that CSP($\mathcal{L}$) $\in$ P and CSP($\mathcal{L} \cup Asst$) is NP-complete. These languages greatly complicate the analysis and make a general-domain analog of Theorem 2 more difficult to prove. For now, we will focus on boolean domains only.

Theorem 2 shows that there is a complexity dichotomy for wAXpDT($\mathcal{L}$) but fails to provide an explicit description of the polynomial-time boolean languages. We address this issue in the next section.

## 4 Characterisation of Tractable Boolean Languages

We now study tractable boolean languages closed under taking complements, in order to gain a better insight into the tractable classes identified in Theorem 2. Let $\mathcal{L}_f$ be the language of boolean relations having the polymorphism $f$. It is well known [31, 33, 32] that

(1) $\mathcal{L}_{\text{min}}$ is the set of conjunctions of Horn clauses.
(2) $\mathcal{L}_{\text{max}}$ is the set of conjunctions of anti-Horn clauses.
(3) $\mathcal{L}_{\text{miny}}$ is the set of conjunctions of affine constraints (i.e. linear equations).
(4) $\mathcal{L}_{\text{maj}}$ is the set of conjunctions of 2-clauses.

In all four cases, $\mathcal{L}_f$ is not closed under complement and so we require extra work to identify the (unique) maximal sublanguage closed under complement.

### 4.1 Horn and Anti-Horn

We start with the language $\mathcal{L}_{\text{min}}$. By the discussion above we need to characterise the maximal sublanguage of $\mathcal{L}_{\text{min}}$ closed under complement, or equivalently the Horn formulas whose negation is expressible by a Horn formula. We will prove that these formulas are exactly those in which the sets of negative literals appearing in clauses are totally ordered with respect to set inclusion. We call such formulas *star-nested*.

Definition 4. *A Horn formula $\psi$ is* star-nested *if and only if there exist sets of literals $L$ and $\emptyset = S_0 \subset S_1 \subset S_2 \subset \ldots \subset S_q$ such that*

- *all literals in $L$ are positive, and*
- *all literals in $S_q$ are negative, and*
- *every clause $C$ in $\psi$ is of the form $C = \bigvee_{s \in S_i} s$ or $C = l \vee \left(\bigvee_{s \in S_i} s\right)$ with $l \in L$.*

To clarify the definition, we point out that each set $S_i$ may occur more than once in the formula (in clauses with different positive literals $l$). In particular, star-nested Horn formulas may contain any number of unit clauses with positive literals (which correspond to the set $S_0 = \emptyset$). Clearly, since the sets $S_i$ are nested, a star-nested formula with no redundant clauses contains at most one clause consisting of only negative literals and at most one clause for each positive literal $l \in L$.

Proposition 2. *Let $\psi$ be a star-nested Horn formula. Then, $\neg\psi$ is equivalent to a star-nested Horn formula.*

Proof. We proceed by induction on the number of sets $S_i$. For $q = 0$, we have $\neg\psi = \bigvee_{l \in L} \neg l$ and hence $\neg\psi$ is a star-nested Horn formula. Now, let $q > 0$ and $\psi$ be a star-nested Horn formula with sets $L, S_0, \ldots, S_q$. Suppose that the claim is true for all formulas with strictly fewer sets. If we denote by $L_0$ the subset of literals in $L$ that appear in unit clauses of $\psi$, then $\psi$ can be rewritten as

$$\psi = \left(\bigwedge_{l \in L_0} l\right) \wedge \left(\left(\bigvee_{s \in S_1} s\right) \vee \phi\right)$$

where $\phi$ is Horn and star-nested with sets $L \setminus L_0, S_1 \setminus S_1, S_2 \setminus S_1, \ldots, S_q \setminus S_1$. In particular, $\phi$ is star-nested with one fewer set than $\psi$. By induction, $\neg\phi$ can be assumed to be Horn and star-nested with sets $L', S'_0, \ldots, S'_p$. Then,

we have

$$\neg\psi = \left(\bigvee_{l\in L_0} \neg l\right) \vee \left((\bigwedge_{s\in S_1} \neg s) \wedge \neg\phi\right)$$

and hence $\neg\psi$ is star-nested with sets $S_0'' = \emptyset, S_1'' = S_0' \cup \{\neg l \mid l \in L_0\}, \ldots, S_{p+1}'' = S_p' \cup \{\neg l \mid l \in L_0\}$, and $L'' = L' \cup \{\neg s \mid s \in S_1\}$. □

Proposition 3. *Let $R$ be a boolean relation such that min is a polymorphism of both $R$ and $\neg R$. Then $R(x_1, \ldots, x_r) \equiv \psi(x_1, \ldots, x_r)$, where $\psi$ is a star-nested Horn formula.*

Proof. We proceed by induction on the arity $r$ of $R$. The claim is true for $r = 1$ since $R$ is either empty, complete, or equivalent to a unit clause; in all cases it is expressible by a star-nested Horn formula. Let $r > 1$ and suppose that the claim is true for all relations whose arity is strictly smaller than $r$. Let $R$ be a relation of arity $r$ such that min is a polymorphism of both $R$ and $\neg R$. We assume without loss of generality that the all-zeroes tuple of length $r$ belongs to $R$. (If this is not the case, then $\neg R$ contains this tuple and we prove the claim on $\neg R$ instead.) If $R$ is complete then we are done. Otherwise, its negation $\neg R = \{t_1, \ldots, t_n\}$ is not empty. Since $\neg R$ has the polymorphism min (which we can assume to be of any arity), we have $t = \min(t_1, \ldots, t_n) \in \neg R$. Note that each $t_i$ is a tuple, so here the operation min is applied componentwise to the set of tuples $t_1, \ldots, t_n$. The tuple $(0, \ldots, 0)$ does not belong to $\neg R$, so the set $P = \{i \leq r \mid t[i] = 1\}$ is not empty. We assume without loss of generality that $P = \{1, \ldots, c\}$. Since $t_j[i] = 1$ for all $j \in \{1, \ldots, n\}$ and $i \in P$, there exists a relation $Q$ such that $\neg R(x_1, \ldots, x_r) \equiv x_1 \wedge \ldots \wedge x_c \wedge Q(x_{c+1}, \ldots, x_r)$. Both $Q$ and $\neg Q$ have the polymorphism min (because $Q$ is a projection of $\neg R$ and $\neg Q$ is a projection of a conjunction of $R$ with unit clauses; the polymorphism min is invariant under these transformations) and the arity of $Q$ is strictly smaller than $r$. By induction, there exists a star-nested Horn formula $\psi$ such that $\neg Q(x_{c+1}, \ldots, x_r) \equiv \psi(x_{c+1}, \ldots, x_r)$. Then, we have

$$\begin{aligned} & R(x_1, \ldots, x_r) \\ & \quad \equiv \neg(x_1 \wedge \ldots \wedge x_c \wedge Q(x_{c+1}, \ldots, x_r)) \\ & \quad \equiv \neg x_1 \vee \ldots \vee \neg x_c \vee \neg Q(x_{c+1}, \ldots, x_r) \\ & \quad \equiv \neg x_1 \vee \ldots \vee \neg x_c \vee \psi(x_{c+1}, \ldots, x_r) \end{aligned}$$

and hence $R$ is equivalent to a star-nested Horn formula by distributivity of $\vee$ over $\wedge$. □

Theorem 3. *Let $\mathcal{L}$ be a boolean constraint language. The following are equivalent:*

*(i) $\mathcal{L}$ has the polymorphism min and is closed under taking complements*
*(ii) Each relation in $\mathcal{L}$ is equivalent to a star-nested Horn formula*

Proof. Follows from Proposition 2 and Proposition 3. □

We also note that, given in input the list of tuples of a relation $R$, star-nested formulas for $R$ and its complement $\neg R$ can be constructed in polynomial time if they exist. The algorithm is given by the recursive constructions used in the proofs of Proposition 2 and Proposition 3.

An anti-Horn formula is *star-nested* if replacing each literal by its negation yields a star-nested Horn formula. The following directly follows from the arguments above, with only slight adaptations.

Theorem 4. *Let $\mathcal{L}$ be a boolean constraint language. The following are equivalent:*

*(i) $\mathcal{L}$ has the polymorphism max and is closed under taking complements*
*(ii) Each relation in $\mathcal{L}$ is equivalent to a star-nested anti-Horn formula*

### 4.2 Affine

We now turn our attention to the case of $\mathcal{L}_{\text{miny}}$, which is straightforward.

Theorem 5. *Let $\mathcal{L}$ be a boolean constraint language. The following are equivalent:*

*(i) $\mathcal{L}$ has the polymorphism miny and is closed under taking complements*
*(ii) Each relation in $\mathcal{L}$ is equivalent to a linear equation over GF(2), the finite field of two elements.*

Proof. The fact that any language satisfying (ii) is closed under taking complements is trivial, as the complement of the equation $a_1x_1 + \ldots + a_rx_r = b$ is $a_1x_1 + \ldots + a_rx_r = 1 - b$. In addition, relations equivalent to linear equations over GF(2) have the minority polymorphism [31]. This establishes (ii) $\Rightarrow$ (i).

Now, let $R$ be a relation of arity $r$ such that both $R$ and $\neg R$ have the minority polymorphism. If $R$ is either empty or complete then it is expressible as a linear equation ($0 = 1$ or $0 = 0$, respectively). Otherwise, both $R$ and $\neg R$ correspond to the solution sets of systems of linear equations over GF(2) that are not degenerate (i.e. at least one equation has a nonzero coefficient). Since any nondegenerate linear equation over GF(2) over $r$ variables has exactly $2^{r-1}$ solutions, we have $|R| = |\neg R| = 2^{r-1}$ and only one equation will remain in both systems after discarding all redundant equations. This establishes (i) $\Rightarrow$ (ii) and concludes the proof. □

### 4.3 Conjunctions of 2-Clauses

As mentioned above, over boolean domains a relation has the polymorphism maj if and only if it is a conjunctions of 2-clauses (clauses containing up to two literals). Thus, to complete the study of tractable cases identified in Theorem 2, we now characterise those formulas $\Phi$ such that both $\Phi$ and $\neg\Phi$ are expressible as conjunctions of 2-clauses.

A *2-clause* is a clause consisting of at most two literals and a *2-term* is a term consisting of at most two literals. The following lemma follows immediately from De Morgan's theorem.

Lemma 2. *A boolean formula $\Phi$ such that $\neg\Phi$ is expressible as conjunction of 2-clauses is expressible as a disjunction of 2-terms.*

Lemma 3. *Suppose that a boolean formula $\Phi$ is such that $\Phi$ is expressible as conjunctions of 2-clauses and also as a disjunction of 2-terms. Suppose, furthermore, that $\Phi \equiv (a \vee b) \wedge \Phi_1$ and $\Phi \equiv (c \wedge d) \vee \Phi_2$. Then there is a non-empty intersection between the two sets of literals $\{a, b\}$ and $\{c, d\}$.*

Proof. With the assignments $a = b = 0$ and $c = d = 1$ we have a contradiction. This can only be avoided if the sets of literals $\{a, b\}$ and $\{c, d\}$ intersect. □

Lemma 4. *Suppose that a boolean formula $\Phi$ is such that $\Phi$ is expressible as a conjunction of 2-clauses and also as a disjunction of 2-terms of the form $\Phi = a \vee \Phi_1$, where $a$ is a literal. Then $\Phi$ is of one of the three forms (1) $a$, (2) $a \vee b$, or (3) $(a \vee b) \wedge (a \vee c)$.*

Proof. Suppose that $\Phi \equiv (b \vee c) \wedge \Phi_2$. Setting $a = 1$ and $b = c = 0$ leads to a contradiction, so to render this impossible we must have $a = b$ or $a = c$. Since this is true for any conjunct, when $\Phi$ is expressed as a conjunction of 2-clauses, we can deduce that $\Phi \equiv \bigwedge_{i=1}^{m}(a \vee b_i)$ for some literals $b_1, \ldots, b_m$. Since $\Phi$ is also expressible as a disjunction of 2-terms, we only need to consider the cases in which $m \leq 2$. When we include the case $\Phi = a$ we have the three cases (1) $a$, (2) $a \vee b$, (3) $(a \vee b) \wedge (a \vee c)$. □

We give without proof the analogous lemma obtained by exchanging conjunction and disjunction.

Lemma 5. *Suppose that a boolean formula $\Phi$ is such that $\Phi$ is expressible as a disjunction of 2-terms and also as a conjunction of 2-clauses of the form $\Phi = a \wedge \Phi_1$, where $a$ is a literal. Then $\Phi$ is of one of the three forms (1) $a$, (2) $a \wedge b$, or (3) $(a \wedge b) \vee (a \wedge c)$.*

Observe that case (3) in Lemma 5 when written as a conjunction of 2-clauses is $a \wedge (b \vee c)$.

A *binary term* is a 2-term that contains exactly two distinct literals.

Lemma 6. *Suppose that a boolean formula $\Phi \neq \bot$ is such that $\Phi$ is expressible as a conjunction of 2-clauses and also as a disjunction of binary terms of the form $\Phi = (a \wedge c) \vee (b \wedge d) \vee \Phi_1$, where $a, b, c, d$ are distinct literals. Then $\Phi$ is of one of the three forms (1) $(a \vee b) \wedge (c \vee d)$, (2) $(a \vee b) \wedge (b \vee c) \wedge (c \vee d)$, or (3) $(a \vee b) \wedge (b \vee c) \wedge (a \vee d) \wedge (c \vee d)$ for distinct literals $a, b, c, d$.*

Proof. Applying Lemma 3 twice, we know that all conjuncts, when $\Phi$ is expressed as a conjunction of 2-clauses, must contain one of $a, c$ and one of $b, d$. Since $a, b, c, d$ are distinct literals, we can deduce that the only possible 2-clauses are $(a \vee b)$, $(b \vee c)$, $(a \vee d)$ and $(c \vee d)$. Eliminating symmetrically equivalent cases, by exhaustive search, we easily obtain only three distinct cases, namely $\Phi$ is of one of the three forms (1) $(a \vee b) \wedge (c \vee d)$, (2) $(a \vee b) \wedge (b \vee c) \wedge (c \vee d)$, or (3) $(a \vee b) \wedge (b \vee c) \wedge (a \vee d) \wedge (c \vee d)$. □

Observe that although $a, b, c, d$ are distinct literals, the variables are not necessarily distinct. For example, if $d = \neg a$ then case (1) becomes $(a \vee b) \wedge (\neg a \vee c)$.

Lemma 7. *Suppose that a boolean formula $\Phi$, expressible as a non-empty conjunction of 2-clauses, is also expressible as a non-empty disjunction of binary terms in which each pair of terms share a literal. Then either $\Phi$ is of the form $\Phi = a \wedge \Phi_1$, where $a$ is a literal, or $\Phi$ is of the form $(a \vee b) \wedge (b \vee c) \wedge (a \vee c)$.*

Proof. If $\Phi$ can be expressed as a disjunction of 2-terms with only one term or two terms (which share a literal), then $\Phi$ is of the form $\Phi = a \wedge \Phi_1$, for some literal $a$. If $\Phi$ can be expressed as a disjunction of three distinct binary terms (where each pair of terms shares a literal), then $\Phi$ is of the form $(a \vee b) \wedge (b \vee c) \wedge (a \vee c)$. There is no set of four distinct binary terms which satisfy the property that each pair shares a literal. □

We now obtain the following characterisation theorem.

Proposition 4. *Let $\Phi$ be a boolean formula such that both $\Phi$ and $\neg\Phi$ are expressible as non-empty conjunctions of 2-clauses. Then $\Phi$ has one of the following forms (in which $a, b, c, d$ are distinct literals):*

*(1)* $a$,
*(2)* $a \vee b$,
*(3)* $a \wedge b$,
*(4)* $a \wedge (b \vee c)$,
*(5)* $(a \vee b) \wedge (a \vee c)$,
*(6)* $(a \vee b) \wedge (c \vee d)$,
*(7)* $(a \vee b) \wedge (b \vee c) \wedge (c \vee d)$,
*(8)* $(a \vee b) \wedge (b \vee c) \wedge (a \vee c)$,
*(9)* $(a \vee b) \wedge (b \vee c) \wedge (a \vee d) \wedge (c \vee d)$.

Proof. By Lemma 2, we are interested in $\Phi$ that can be expressed as a conjunction of 2-clauses and a disjunction of 2-terms. If $\Phi$, when written as a disjunction of 2-terms, has a unary term (i.e. $\Phi$ can be written in the form $a \vee \Phi_1$), then Lemma 4 applies (cases (1), (2), (5)). If $\Phi$ can be expressed as a disjunction of binary terms, two of which share no literals, then Lemma 6 applies (cases (6), (7), (9)). If $\Phi$ can be expressed as a disjunction of binary terms, each pair of which share a literal, then Lemma 7 applies (case (8)). In the subcase of Lemma 7 in which $\Phi$ can be written in the form $a \wedge \Phi_1$, Lemma 5 applies (cases (1), (3), (4)). □

The following corollary is simply a more succinct rewriting of Proposition 4.

Corollary 2. *If $\Phi$ is a boolean formula such that both $\Phi$ and $\neg\Phi$ are expressible as non-empty conjunctions of 2-clauses, then $\Phi$ has one of the three following forms (in which the four literals are not necessarily distinct):*

565 *(i)* $(a \vee b) \wedge (c \vee d)$,
566 *(ii)* $(a \vee b) \wedge (b \vee c) \wedge (c \vee d)$,
567 *(iii)* $(a \vee b) \wedge (b \vee c) \wedge (a \vee d) \wedge (c \vee d)$.
568
569 Proof. We can obtain the nine cases listed in Proposition 4 as follows: (1) set $a = b = c$ in (iii), (2) set $a = c$ and $b = d$ in (iii), (3) set $a = b$ and $c = d$ in (iii), (4) set $a = d$ in (iii), (5) set $a = c$ in (iii), (6) is case (i) (7) is case (ii), (8) set $a = d$ in (ii), (9) is case (iii). □
570
571
572 It is straightforward to verify that the converse to Corollary 2 holds, that is, any formula $\Phi$ satisfying at least one of items (i), (ii) or (iii) is such that both $\Phi$ and $\neg\Phi$ are expressible as conjunctions of 2-clauses. In the following, we use the name *square 2CNF* for formulas that are expressible as both conjunctions of 2-clauses and disjunctions of 2-terms (characterised in Proposition 4 and Corollary 2). The name reflects the fact these formulas are the subformulas of the square given by item (iii) of Corollary 2 (seeing literals $a$,$b$,$c$,$d$ as vertices and clauses as edges).

573
574
575
576
577 Observation 1. *Square 2CNF formulas include all binary relations over boolean domains. For example, the relation $a \neq b$ can be obtained by setting $c = \neg a$ and $d = \neg b$ in $(a \vee b) \wedge (c \vee d)$.*

578
579
580 Theorem 6. *Let $\mathcal{L}$ be a boolean constraint language. The following are equivalent:*

581 *(i) $\mathcal{L}$ has the polymorphism maj and is closed under taking complements*
582 *(ii) Each relation in $\mathcal{L}$ is equivalent to a square 2CNF, i.e. either empty, complete, or expressible in one of the three following forms (in which the four literals are not necessarily distinct): (i) $(a \vee b) \wedge (c \vee d)$, (ii) $(a \vee b) \wedge (b \vee c) \wedge (c \vee d)$, (iii) $(a \vee b) \wedge (b \vee c) \wedge (a \vee d) \wedge (c \vee d)$.*

583
584
585
586
587 ## 4.4 The Dichotomy for Boolean Languages

Bringing together what we have learnt in this section, we have the following theorem.

588
589
590 Theorem 7. *Let $\mathcal{L}$ be a finite boolean language closed under taking complements. Then, assuming* P $\neq$ NP, wAXpDT*($\mathcal{L}$)* $\in$ P *iff at least one of the conditions holds:*

591 *(1) Each relation in $\mathcal{L}$ is equivalent to a star-nested Horn formula*
592 *(2) Each relation in $\mathcal{L}$ is equivalent to a star-nested anti-Horn formula*
593 *(3) Each relation in $\mathcal{L}$ is equivalent to a linear equation over GF(2)*
594 *(4) Each relation in $\mathcal{L}$ is equivalent to a square 2CNF formula.*

595 The requirement that $\mathcal{L}$ is finite in Theorem 7 arises from technicalities related to the representation of
596 infinite languages. Indeed, certain degenerate representations for the relations of an infinite language $\mathcal{L}$ may be
597 problematic from an algorithmic perspective. For example, the promise that the relations of $\mathcal{L}$ are equivalent to
598 star-nested Horn formulas might not be sufficient to ensure tractability (or even membership in NP) if they are
599 encoded in a way that makes even the most elementary relational operations NP-hard. However, this theorem
600 is still true for infinite languages if one makes the mild assumptions that (i) relations equivalent to linear
601 equations are always represented as such, and (ii) the representation used for relations equivalent to star-nested
602 Horn/anti-Horn formulas allows for checking in polynomial time whether a given assignment extends to a tuple.

603 Example 11. *Consider the language $\mathcal{L}$ of Example 6, which consists of all 2-conjunctions of 2-clauses. Now, extend*
604 *$\mathcal{L}$ with pseudo-boolean constraints $a + b + c \geq 2$ for any literals $a, b, c$, where summation is over $\mathbb{Z}$. This larger*
605 *language $\mathcal{L}'$ is closed under taking complements (the complement of $a + b + c \geq 2$ is $\neg a + \neg b + \neg c \geq 2$), and all*
606 *constraints in $\mathcal{L}'$ can be expressed as square 2CNF formulas because $a+b+c \geq 2 \equiv (a \vee b) \wedge (b \vee c) \wedge (c \vee a)$. Therefore,*
607 *by Theorem 7 we have* wAXpDT*($\mathcal{L}'$)* $\in$ P. *However, no quaternary pseudo-boolean constraint $a + b + c + d \geq k$ with*
608 *$1 \leq k < 4$ can be expressed as a square 2CNF formula. In fact, adding any such constraint to $\mathcal{L}$ would cause the*
609 *corresponding* wAXpDT *problem to become NP-complete by Theorem 7 as the resulting language would violate each*
610 *of the four tractability conditions.*
611

## 5 Extending the tractable boolean languages

In this section we extend tractable languages to ordered domains of arbitrary finite size and prove a dichotomy for a restricted form of languages.

Without loss of generality, and for simplicity of presentation, we assume that all features have the same domain $D = \{0, 1, \ldots, d-1\}$.

Recall that we call literals of the form $x_i \geq a$ or $x_i < a$, where $a$ is a constant in $D \setminus \{0\}$, *unary inequality literals* (UI-literals). We choose the convention that literals of the form $x_i \geq a$, where $a \in D \setminus \{0\}$, are *positive* UI-literals, whereas literals of the form $x_i < a$ are *negative* UI-literals. The notions of (anti-)Horn clauses, star-nested (anti-)Horn formulas and square 2CNF all generalise in a natural way. For example, a UI-generalisation of a Horn clause is the disjunction of UI-literals, at most one of which is positive. We show in this section that the star-nested (anti-)Horn and square 2CNF languages generalise to tractable languages over domains of size $d \geq 3$, whereas the affine language does not.

### 5.1 Generalised square 2CNFs

Definition 5. *A constraint relation $R$ (over domains of arbirary finite size) is* generalised square 2CNF *if $R$ has one of the three following forms, where $a, b, c, d$ are (not necessarily distinct) UI-literals:*

*(i)* $(a \vee b) \wedge (c \vee d)$,
*(ii)* $(a \vee b) \wedge (b \vee c) \wedge (c \vee d)$,
*(iii)* $(a \vee b) \wedge (b \vee c) \wedge (a \vee d) \wedge (c \vee d)$

Example 12. *All boolean functions of two UI-literals (i.e. the constraints studied in Example 8) are generalised square 2CNF. This follows directly from Observation 1. All constraints of the form $(x_i \in [p, q]) \vee (x_j \in [r, s])$ are also generalised square 2CNF since they can be obtained by setting $a = (x_i \geq p)$, $b = (x_j \geq r)$, $c = (x_i < q{+}1)$ and $d = (x_j < s{+}1)$ in $(a \vee b) \wedge (b \vee c) \wedge (a \vee d) \wedge (c \vee d)$.*

We denote by $\mathcal{L}_{GS}$ the language of generalised square 2CNF relations together with all unary constraints (i.e. constraints of the form $x_i \in A$ where $\emptyset \subset A \subset D$).

Lemma 8. *$\mathcal{L}_{GS}$ is closed under taking complements.*

Proof. For unary constraints, this follows from the fact that $\neg(x_i \in A) \equiv x_i \in D \setminus A$ and $\emptyset \subset A \subset D$ implies $\emptyset \subset (D \setminus A) \subset D$. For generalised square 2CNF constraints, as in the boolean case, it follows from the identities:

(i) $\neg((a \vee b) \wedge (c \vee d)) \equiv (\neg a \vee \neg c) \wedge (\neg b \vee \neg c) \wedge (\neg a \vee \neg d) \wedge (\neg b \vee \neg d)$,
(ii) $\neg((a \vee b) \wedge (b \vee c) \wedge (c \vee d)) \equiv (\neg a \vee \neg c) \wedge (\neg b \vee \neg c) \wedge (\neg b \vee \neg d)$,
(iii) $\neg((a \vee b) \wedge (b \vee c) \wedge (a \vee d) \wedge (c \vee d)) \equiv (\neg a \vee \neg c) \wedge (\neg b \vee \neg d)$

□

Proposition 5. wAXpDT($\mathcal{L}_{GS}$) $\in$ P.

Proof. Clearly $Asst \subseteq \mathcal{L}_{GS}$ since, by definition, $\mathcal{L}_{GS}$ contains all unary constraints. Given Proposition 1 and Lemma 8, it therefore suffices to show that CSP($\mathcal{L}_{GS}$) $\in$ P.

Consider the ternary function med: $D^3 \to D$ which returns the median value among its three arguments. We will show that med is a polymorphism of $\mathcal{L}_{GS}$. Since med is conservative (that is, it always returns one of its arguments), it is a polymorphism of all unary constraints. It is well known that if two relations $R_1, R_2$ have the same polymorphism $f$, then so does their join $R_1 \bowtie R_2$ [32]. In the definition of generalised square 2CNF relations, the conjunction operation is equivalent to a join. Hence, to show that all generalised square 2CNF relations have the polymorphism med it suffices to show that $a \vee b$ has the polymorphism med where $a$ and $b$ are any UI-literals.

Suppose that $(u_1, u_2), (v_1, v_2), (w_1, w_2) \in R$, where $R = a \vee b$ and $a, b$ are UI-literals on variables $x_1, x_2$. Then at least two of these tuples must satisfy the same UI-literal. Without loss of generality, suppose that $(u_1, u_2), (v_1, v_2)$ both satisfy the literal on the variable $x_1$. Now if this is a positive UI-literal, say $x_1 \geq p$, then $\text{med}(u_1, v_1, w_1) \geq p$. Similarly, if it is a negative literal, say $x_1 < q$, then $\text{med}(u_1, v_1, w_1) < q$. Hence $a \vee b$ has the polymorphism med for any UI-literals $a, b$, and so does $\mathcal{L}_{GS}$.

It is known that for constraint languages $\Gamma$ with a majority polymorphism, such as med, CSP($\Gamma$) can be solved in polynomial time by establishing strong 3-consistency [30]. This thus completes the proof that wAXPDT($\mathcal{L}_{GS}$) $\in$ P. □

## 5.2 Generalised star-nested (anti-)Horn

Definition 6. *A constraint relation $R$ over domains of arbitrary finite size is* generalised star-nested Horn *if and only if there exist sets of UI-literals $L$ and $\emptyset = S_0 \subset S_1 \subset S_2 \subset \ldots \subset S_q$ such that*

- *all UI-literals in $L$ are positive, and*
- *all UI-literals in $S_q$ are negative, and*
- *$R$ is the conjunction of clauses $C$ of the form $C = \bigvee_{s \in S_i} s$ or $C = l \vee \left(\bigvee_{s \in S_i} s\right)$ with $l \in L$.*

*A relation $R$ is generalised star-nested anti-Horn if replacing each UI-literal by its negation yields a generalised star-nested Horn relation.*

Example 13. *A useful constraint that is generalised star-nested Horn is the constraint $x_i \geq x_j$. To see this, observe that $x_i \geq x_j$ is equivalent to the non-existence of a value $a \in D \setminus \{0\}$ such that $x_i < a$ and $x_j \geq a$. Hence*

$$x_i \geq x_j \equiv \bigwedge_{a \in D \setminus \{0\}} \left((x_i \geq a) \vee (x_j < a)\right) \equiv \bigwedge_{a \in D \setminus \{0\}} \left((x_i \geq a) \vee \bigvee_{s \in S_a} s\right)$$

*where $S_a = \{x_j < 1, \ldots, x_j < a\}$. Clearly $S_1 \subset \ldots \subset S_{d-1}$ and so $x_i \geq x_j$ satisfies the definition of a generalised star-nested Horn constraint.*

*This argument can be generalised to a larger family of constraints. Consider $k+1$ features $x_i, x_{j_1}, \ldots, x_{j_k}$ and a constraint of the form $x_i \geq \min\left(f_1(x_{j_1}), \ldots, f_k(x_{j_k})\right)$ where each function $f_q : \mathbb{N} \to \mathbb{N}$ is nondecreasing. As above, this constraint is equivalent to the non-existence of $a \in D \setminus \{0\}$ such that $x_i < a$ and $f_q(x_{j_q}) \geq a$ for all $1 \leq q \leq k$. In order to express this condition as a formula over UI-literals, notice that for all $1 \leq q \leq k$ and $a \in D \setminus \{0\}$ we have*

$$f_q(x_{j_q}) < a \equiv \begin{cases} \textit{true} & \textit{if } f_q(d-1) < a \\ \bigvee_{b \in D \setminus \{d-1\} : f_q(b) < a} (x_{j_q} < b+1) & \textit{otherwise.} \end{cases}$$

*Finally, we obtain*

$$x_i \geq \min\left(f_1(x_{j_1}), \ldots, f_k(x_{j_k})\right) \equiv \bigwedge_{a \in A} \left((x_i \geq a) \vee \bigvee_{s \in S_a} s\right)$$

*where $A = \{a \in D \setminus \{0\} \mid \forall q,\ f_q(d-1) \geq a\}$, $S_a = S_{a1} \cup \ldots \cup S_{ak}$ and $S_{aq} = \{x_{j_q} < b+1 \mid (0 \leq b < d-1) \wedge (f_q(b) < a)\}$ for all $a \in A$ and $1 \leq q \leq k$. Each function is nondecreasing, so $S_p \subset S_q$ whenever $p < q$ and hence this constraint is generalised star-nested Horn.*

We denote by $\mathcal{L}_{GH}$ (respectively $\mathcal{L}_{GA}$) the language of generalised star-nested Horn (resp. anti-Horn) relations together with all unary constraints.

Lemma 9. *$\mathcal{L}_{GH}$ and $\mathcal{L}_{GA}$ are closed under taking complements.*

Proof. For unary constraints, this follows from the fact that $\neg(x_i \in A) \equiv x_i \in D \setminus A$ and $\emptyset \subset A \subset D$ iff $\emptyset \subset (D \setminus A) \subset D$. For generalised star-nested (anti-)Horn constraints, the proof is identical to the proof of Proposition 2. □

Proposition 6. wAXpDT($\mathcal{L}_{GH}$) ∈ P *and* wAXpDT($\mathcal{L}_{GA}$) ∈ P.

Proof. We give the proof only for $\mathcal{L}_{GH}$ since the proof for $\mathcal{L}_{GA}$ is almost identical. Clearly $Asst \subseteq \mathcal{L}_{GH}$ since, by definition, $\mathcal{L}_{GH}$ contains all unary constraints. Given Proposition 1 and Lemma 9, it therefore suffices to show that CSP($\mathcal{L}_{GH}$) ∈ P.

Consider the binary function min: $D^2 \to D$ which returns the minimum value among its two arguments. As min always returns one of its arguments, it is a polymorphism of all unary constraints. Generalised star-nested Horn relations are joins (conjunctions) of Horn formulas on UI-literals. As in the proof of Proposition 5, to show that all generalised star-nested Horn relations have the polymorphism min, it suffices to show that a single Horn clause with UI-literals is min-closed.

Suppose that $(u_1, \ldots, u_r), (v_1, \ldots, v_r) \in R$, where $R$ is given by the following Horn clause on UI-literals:

$$(x_1 < a_1) \vee \ldots \vee (x_{r-1} < a_{r-1}) \vee (x_r \geq a_r)$$

If either of the tuples satisfies a negative literal, say $u_i < a_i$, then $\min(u_i, v_i) < a_i$. The only remaining case to consider is when both tuples satisfy the positive literal: in this case $u_r \geq a_r$ and $v_r \geq a_r$, hence $\min(u_r, v_r) \geq a_r$. The case in which $R$ has no positive literal is immediate. Hence any Horn clause on UI-literals have the polymorphism min, and so does $\mathcal{L}_{GH}$.

It is known that for constraint languages Γ with the polymorphism min, CSP(Γ) can be solved in polynomial time by establishing (generalised) arc consistency [32, 13]. This completes the proof that wAXpDT($\mathcal{L}_{GH}$) ∈ P. □

## 5.3 A dichotomy for UI-generalisations of boolean languages

We now consider a restricted class of languages over non-boolean domains for which we establish a complexity dichotomy. The languages we consider are natural extensions of boolean languages.

Given a formula $\psi(x_1, \ldots, x_r)$ on boolean variables $x_1, \ldots, x_r$, the language obtained from $\psi$ by *UI-generalisation* (of its literals) is

$$\mathcal{L}^{UI}_{\psi} = \{\psi(x_1 \geq a_1, \ldots, x_r \geq a_r) \mid a_1, \ldots, a_r \in D \setminus \{0\}\} \tag{1}$$

i.e. $\mathcal{L}^{UI}_{\psi}$ consists of all formulas obtained by replacing each occurrence of $x_i$ by some UI-literal $x_i \geq a_i$, where each $a_i$ can be any non-zero constant. Note that two logically equivalent formulas $\psi_1, \psi_2$ necessarily give rise to the same languages $\mathcal{L}^{UI}_{\psi_1}, \mathcal{L}^{UI}_{\psi_2}$. Indeed, $\psi_1(x_1 \geq a_1, \ldots, x_r \geq a_r) \equiv \psi_2(x_1 \geq a_1, \ldots, x_r \geq a_r)$ if $\psi_1(x_1, \ldots, x_r) \equiv \psi_2(x_1, \ldots, x_r)$. Hence, we can write $\mathcal{L}^{UI}_{R}$ for any constraint relation $R$ over boolean domains, since it is independent of the boolean formula used to express $R$.

For a boolean language $\mathcal{B}$, the language $\mathcal{L}^{UI}_{\mathcal{B}}$ obtained by UI-generalisation from $\mathcal{B}$ is given by

$$\mathcal{L}^{UI}_{\mathcal{B}} = \bigcup_{R \in \mathcal{B}} \mathcal{L}^{UI}_{R}$$

Clearly, if $\mathcal{B}$ is closed under taking complements, then so is $\mathcal{L}^{UI}_{\mathcal{B}}$: indeed, $\neg(R(x_1 \geq a_1, \ldots, x_r \geq a_r)) \equiv (\neg R)(x_1 \geq a_1, \ldots, x_r \geq a_r)$ and $\mathcal{B}$ closed under taking complements means that $R \in \mathcal{B} \Rightarrow (\neg R) \in \mathcal{B}$.

In the definition of UI-generalisation in equation (1), each occurrence of the argument $x_i$ in the formula $\psi(x_1, \ldots, x_r)$ is replaced by the same UI-literal $x_i \geq a_i$. Observe, however, that we assume (implicitly) in the definition of the problems CSP($\mathcal{L}$) and wAXpDT($\mathcal{L}$) that constraints can be applied with repeated arguments. This allows us in CSP($\mathcal{L}^{UI}_{\mathcal{B}}$) or wAXpDT($\mathcal{L}^{UI}_{\mathcal{B}}$) to apply a constraint of the form $R(x_1 \geq a_1, x_1 \geq a_2, x_3 \geq a_3, \ldots, x_r \geq x_r)$ where $a_1 \neq a_2$, by replacing the second argument $x_2$ of $R$ by the UI-literal $x_1 \geq a_2$.

An important point is that not all languages of relations over a domain $D$ of size greater than 2 can be expressed as a language $\mathcal{L}_{\mathcal{B}}^{UI}$ for some boolean language $\mathcal{B}$. A property of languages $\mathcal{L}_{\mathcal{B}}^{UI}$ obtained by UI-generalisation of boolean languages is that they are value-independent in the sense that $\mathcal{L}_{\mathcal{B}}^{UI}$ necessarily contains all relations $R(x_1 \geq a_1, \ldots, x_r \geq a_r)$ for *all* $(a_1, \ldots, a_r) \in (D \setminus \{0\})^r$ for $R \in \mathcal{B}$. For example, if $\mathcal{B}$ is the language consisting of the single square 2CNF relation $x_1 \vee x_2$, then we do not consider the language consisting of the single relation $(x_1 \geq 1) \vee (x_2 \geq 1)$; instead $\mathcal{L}_R^{UI}$ is $\{(x_1 \geq a_1) \vee (x_2 \geq a_2) \mid a_1, a_2 \in D \setminus \{0\}\}$.

We now study the complexity of CSP($\mathcal{L}_{\mathcal{B}}^{UI}$) for boolean languages $\mathcal{B}$.

Lemma 10. *For boolean languages $\mathcal{B}$, CSP($\mathcal{L}_{\mathcal{B}}^{UI}$) $\in$ P only if CSP($\mathcal{B}$) $\in$ P.*

Proof. Let $\mathcal{L}_{\mathcal{B}}^{1}$ be the subset of $\mathcal{L}_{\mathcal{B}}^{UI}$ consisting of the relations $R(x_1 \geq 1, \ldots, x_r \geq 1)$ for all $R \in \mathcal{B}$. There is a direct polynomial reduction from CSP($\mathcal{B}$) to CSP($\mathcal{L}_{\mathcal{B}}^{1}$) in which each constraint $R(x_{i_1}, \ldots, x_{i_r})$ is replaced by the constraint $R(x_{i_1} \geq 1, \ldots, x_{i_r} \geq 1)$. Hence CSP($\mathcal{L}_{\mathcal{B}}^{UI}$) $\in$ P implies that CSP($\mathcal{B}$) $\in$ P. □

This means that to identify all tractable languages $\mathcal{L}_{\mathcal{B}}^{UI}$ closed under taking complements, we only need to consider the four tractable boolean languages $\mathcal{B}$ closed under taking complements. For square 2CNF and star-nested (anti-)Horn tractability of wAxpDT follows from results we have already proved, as we show in the following proposition.

Proposition 7. *Let $U$ be the set of unary relations (of the form $x_i \in A$ where $\emptyset \subset A \subset D$). Then* wAxpDT($\mathcal{L}_{\mathcal{B}}^{UI}$) $\in$ P *and* wAxpDT($\mathcal{L}_{\mathcal{B}}^{UI} \cup U$) $\in$ P *when $\mathcal{B}$ is any of the following boolean languages:*

*(1) the language of square 2CNF relations,*
*(2) the language of star-nested Horn relations,*
*(3) the language of star-nested anti-Horn relations.*

Proof. We have already seen that $\mathcal{B}$ is closed under taking complements for the language of square 2CNF relations (Lemma 8 in the case of boolean domains), and the language of star-nested (anti-)Horn relations (Proposition 2). As observed above, if $\mathcal{B}$ is closed under taking complements, then so is $\mathcal{L}_{\mathcal{B}}^{UI}$. We have already seen, in the proof of Lemma 8, that $U$ is closed under taking complements. It follows that all languages under consideration are closed under taking complements.

When $\mathcal{B}$ is the language of square 2CNF relations, $\mathcal{L}_{\mathcal{B}}^{UI} \cup U \subseteq \mathcal{L}_{GS}$ and so tractability of wAxpDT($\mathcal{L}_{\mathcal{B}}^{UI} \cup U$) (and wAxpDT($\mathcal{L}_{\mathcal{B}}^{UI}$)) follows from Proposition 5. When $\mathcal{B}$ is the language of star-nested Horn relations, $\mathcal{L}_{\mathcal{B}}^{UI} \cup U \subseteq \mathcal{L}_{GH}$ and so tractability of wAxpDT($\mathcal{L}_{\mathcal{B}}^{UI} \cup U$) (and wAxpDT($\mathcal{L}_{\mathcal{B}}^{UI}$)) follows from Proposition 6. When $\mathcal{B}$ is the language of star-nested anti-Horn relations, $\mathcal{L}_{\mathcal{B}}^{UI} \cup U \subseteq \mathcal{L}_{GA}$ and so tractability of wAxpDT($\mathcal{L}_{\mathcal{B}}^{UI} \cup U$) (and wAxpDT($\mathcal{L}_{\mathcal{B}}^{UI}$)) follows again from Proposition 6. □

The only remaining boolean language to consider is the affine case, i.e. $\mathcal{B}$ the set of linear equations over GF(2). The relation obtained by UI-generalisation of the equation $x_1 + x_2 + \ldots + x_r \equiv c \mod 2$ is

$$[x_1 \geq a_1] + [x_2 \geq a_2] + \ldots + [x_r \geq a_r] \equiv c \mod 2 \tag{2}$$

where $[P]$ equals 1 if $P$ is true, 0 otherwise. We call a relation of the form in equation (2), a UI-generalised affine relation. Recall that all affine constraints can be written in the form of the sum of positive literals $x_1 + x_2 + \ldots + x_r \equiv c \mod 2$. For example, the constraint $(\neg x_1) + x_2 + \ldots + x_r \equiv c \mod 2$ is equivalent to $x_1 + x_2 + \ldots + x_r \equiv c + 1 \mod 2$.

Lemma 11. *Let $\mathcal{B}$ be a boolean language composed of the single affine relation $x_1 + x_2 + \ldots + x_r \equiv c \mod 2$, where $r \geq 3$, and let $\mathcal{L}_{\mathcal{B}}^{UI}$ be the UI-generalisation of $\mathcal{B}$ to a domain of size $d \geq 3$. Then CSP($\mathcal{L}_{\mathcal{B}}^{UI} \cup Asst$) is NP-hard.*

Proof. We demonstrate NP-hardness by reduction from $d$-coloring. To do this, it is sufficient to give a gadget that simulates the constraint $x_i \neq x_j$. Consider the gadget

$$\begin{aligned} \exists y_3, \ldots, y_r \quad & ([x_i \geq a] \ + \ [x_j \geq b] \ + \ [y_3 \geq 1] + \ldots + \ [y_r \geq 1] \equiv c \mod 2 \\ \wedge \quad & [x_i \geq a{+}1] + [x_j \geq b{+}1] + [y_3 \geq 1] + \ldots + [y_r \geq 1] \equiv c \mod 2) \end{aligned}$$

This is equivalent to

$$[x_i \geq a] + [x_j \geq b] \equiv [x_i \geq a{+}1] + [x_j \geq b{+}1] \mod 2$$

which can be simplified to

$$(x_i = a) \equiv (x_j = b)$$

In the above construction, when $a$ is the maximum element in the domain, $x_i \geq a{+}1$ is not a valid UI-literal. But, in this case, we replace it in the above gadget by the literal $[y_1 \geq 2]$ and we add the constraint $y_1 = 1$ (which belongs to *Asst*). This has the same effect as $x_i \geq a{+}1$ which, of course, is always false. A similar remark holds when $a = 0$. In this case, $[x_i \geq a]$ is not a valid UI-literal, but since it is always true we replace it by $y_1 \geq 1$ (where again we impose the assignment constraint $y_1 = 1$). Clearly, similar adjustments will be necessary if $b = d - 1$ or $b = 0$.

Now, for any $a \in D \setminus \{0\}$, we can choose a domain value $a' \neq a$ and we can construct the gadget

$$\begin{aligned} \exists z_k \quad & ((x_i = a) \equiv (z_k = a) \\ \wedge \quad & (x_j = a) \equiv (z_k = a')) \end{aligned}$$

This is equivalent to

$$(x_i \neq a) \ \vee \ (x_j \neq a)$$

The conjunction $\bigwedge_{a \in D}((x_i \neq a) \vee (x_j \neq a))$ is then equivalent to $x_i \neq x_j$. Thus we can construct the constraint $x_i \neq x_j$ and hence there is a polynomial reduction from $d$-coloring to CSP($\mathcal{L}^{UI}_{\mathcal{B}}$). □

This allows us to state the following dichotomy for UI-generalisations of boolean languages,

Theorem 8. *Let $\mathcal{B}$ be a finite boolean language closed under taking complements, and let $U$ be the set of unary relations (of the form $x_i \in A$ where $\emptyset \subset A \subset D$). Assuming* P ≠ NP, wAXpDT($\mathcal{L}^{UI}_{\mathcal{B}} \cup U$) ∈ P *iff $\mathcal{B}$ is a sublanguage of (at least) one of the following boolean languages:*

*(1) the language of square 2CNF relations,*
*(2) the language of star-nested Horn relations,*
*(3) the language of star-nested anti-Horn relations.*

Proof. The 'if' direction follows directly from Proposition 7. For the 'only if' direction, we first show that any binary UI-generalised affine relation is, in fact, equivalent to a generalised square 2CNF. The binary affine constraint $[x_i \geq a] + [x_j \geq b] \equiv 1 \mod 2$ is equivalent to the generalised square 2CNF:

$$((x_i \geq a) \vee (x_j \geq b)) \wedge ((x_i < a) \vee (x_j < b))$$

and the other binary affine constraint $[x_i \geq a] + [x_j \geq b] \equiv 0 \mod 2$ is equivalent to the generalised square 2CNF:

$$((x_i \geq a) \vee (x_j < b)) \wedge ((x_i < a) \vee (x_j \geq b))$$

Unary affine constraints are of course unary constraints. Thus, by Lemma 11, the largest subset $\mathcal{L}$ of UI-generalised affine relations such that CSP($\mathcal{L}$) is not NP-hard, is actually a sublanguange of $\mathcal{L}^{UI}_{\mathcal{B}_S} \cup U$ where $\mathcal{B}_S$ is the boolean language of square 2CNF relations. Thus, the 'only if' direction then follows from Theorem 7 (the dichotomy for boolean languages) together with Lemma 10. □

We now revisit the notion of contrastive explanation (CXp). Recall that a CXp a minimal set of features which can be changed in order to change the class. From Definition 3, it is clear that being a weak CXp is a monotone property. Hence, we can deduce from Lemma 1 that to find a CXp in polynomial time, it is sufficient to be able to test whether a set of features is a weak CXp in polynomial time. We therefore concentrate on this latter problem. Let wCXpDT($\mathcal{L}$) denote the problem of deciding whether a set of features is a weak CXp for a given decision taken by an MDT in $\mathcal{L}$-DT. Let $F$ denote the set of features. It follows immediately from the definitions of weak AXp and weak CXp that a set $S$ is a weak CXp iff $F \setminus S$ is not a weak AXp. This implies that the wAXpDT(L) $\in$ P iff wCXpDT(L) $\in$ P, from which we have the following theorem.

Theorem 9. *Let $\mathcal{B}$ be a finite boolean language closed under taking complements, and let $U$ be the set of unary relations (of the form $x_i \in A$ where $\emptyset \subset A \subset D$). Assuming* P $\neq$ NP, wCXpDT($\mathcal{L}^{UI}_{\mathcal{B}} \cup U$) $\in$ P *iff $\mathcal{B}$ is a sublanguage of (at least) one of the following boolean languages:*

*(1) the language of square 2CNF relations,*
*(2) the language of star-nested Horn relations,*
*(3) the language of star-nested anti-Horn relations.*

## 6 Experimental trials

An important question is whether there is a real advantage in using MDTs rather than DTs. When comparing their relative merits, obvious criteria are the accuracy of the model and the computational resources required to learn the model, to query it and to produce explanations. One should also take into account the interpretability of explanations that are produced for predictions. For example, there may be a trade-off between accuracy and concision of explanations.

In order to guarantee tractability of finding an explanation, we chose to study MDTs whose conditions belong to the language of generalised square 2CNFs. To ensure the feasibility of building a model, we limited ourselves to constraints concerning just one or two finite-domain features. In other words, we consider the language of constraints described in Example 8 and illustrated in Figure 4. In order to perform a fair comparison between MDTs and DTs, we chose to compare the accuracy of MDTs and DTs for which average sizes of explanations are expected to be equal. Our experimental trials indicated that there is no statistically significant difference between the accuracy of depth-5 MDTs and depth-7 DTs. As we explain in detail below, the depths 5 and 7 were chosen so as to have, on average, the same size of explanations. A direct conclusion is that MDTs can provide more compact models with the same performance (in terms of accuracy and explanation-size) as DTs.

We now describe in more detail our experiments. There are exactly ten truly binary boolean functions (i.e. not equivalent to constant or unary functions) on two boolean arguments, namely the following five functions and their complements: $f_1(x, y) = x \vee y$, $f_2(x, y) = \neg x \vee y$, $f_3(x, y) = x \vee \neg y$, $f_4(x, y) = \neg x \vee \neg y$, $f_5(x, y) = x \oplus y$ (i.e. the exclusive or of $x$ and $y$). All these functions are square 2CNFs. Under a uniform distribution of the values of the arguments $x, y$, $f_i(x, y)$ has a probability of $1/4$ of being 0 and $3/4$ of being 1 ($i = 1, \ldots, 4$) whereas $f_5(x, y)$ has a probability of $1/2$ of being either 0 or 1. Consider a node in an MDT whose constraint is $l_1 \vee l_2$, for some literals $l_1, l_2$. It has two child nodes corresponding respectively to the disjunction $l_1 \vee l_2$ and its complement, the conjunction $\neg l_1 \wedge \neg l_2$. The explanation for taking the disjunctive branch is the feature associated with the literal which is true (or an arbitrary choice if both $l_1, l_2$ are true). The explanation for taking the conjunctive branch $\neg l_1 \wedge \neg l_2$ is both features. In the case of an exclusive or, the explanation for taking either branch requires both features. Assuming that the five functions $f_1, \ldots, f_5$ are equally likely to occur as constraints in an MDT, the probability that we require only one feature to explain which branch was taken is $(4/5) \cdot (3/4) = 3/5$ (since 4 out of 5 of the functions $f_i$ have a disjunctive branch, and as observed above there is a $3/4$ chance that this is the branch that is taken).

As a first approximation to compare explanation size for decisions taken by MDTs and DTs, we can simply estimate the size of a weak AXp corresponding to the path from the root to a leaf, assuming that all features encountered along a path are distinct. In a DT the size of this weak AXp is equal to the length of the path. As we have seen above, in an MDT every disjunctive branch along the path contributes one feature and every conjunctive branch or exclusive-or branch contributes two features. An MDT may also contain unary constraints, but since there are much fewer unary constraints than binary constraints (a linear number rather than a quadratic number), we consider that unary constraints are rare (which was confirmed by our experiments) and can hence be ignored in the approximate calculation of expected explanation size. Let $d_{\mathrm{DT}}$ and $d_{\mathrm{MDT}}$ be, respectively, the depths of the DT and MDT. We have seen above that each edge in an MDT contributes one feature with probability 3/5 and two features with probability $1 - 3/5 = 2/5$. This means that the weak AXp corresponding to a path from the root to a leaf in an MDT will contain, on average, $d_{\mathrm{MDT}}(3/5 + 2 \cdot (2/5)) = d_{\mathrm{MDT}} \cdot (7/5)$ features, compared to the $d_{\mathrm{DT}}$ features for a DT. Thus, in order to make a fair comparison, we decided to build MDTs of depth 5 and DTs of depth 7, so that the resulting AXp's can be expected to be of the same average length.

We used Blossom [20], a program that searches for optimal DTs and which importantly has a good anytime performance, in the sense that good-quality trees are found even if the program times out. Blossom is designed to build DTs, so to learn MDTs we opted for a simple solution consisting of effectively adding new features corresponding to $f_i(x, y)$ $(i = 1, \ldots, 5)$ for each pair of distinct features $x, y$. The DTs thus constructed are clearly equivalent to MDTs on the original features. Given the large increase in the number of features (the larger data set have millions of features standing for binary tests), a guarantee of optimality was rarely achieved for the MDTs. Blossom was stopped after 10 minutes of computation time in every case.

The experiments were run on a computing cluster containing 780 CPUs operated with Ubuntu 20.04. LTS and Slurm 20.11.4, although each run was a single thread limited to 10 minutes of CPU time. Blossom was run with its default parameters, in particular, it uses *Gini impurity* [8] to guide the search, and implements preprocessing of the data set which was highly relevant given the size of the expanded data sets. All code and data necessary to reproduce the experiments is available in a repository[2].

We used datasets from the UCI repository [35] (listed in Table 1). The columns "unary" and "binary" features in this table indicate the original number of features and the number of possible binary test in the chosen language. Although the complexity of the method scales exponentially in this parameter[3], its good anytime behavior makes it possible to learn high quality trees. For each data set, we splitted the instances with uniform probability between the training and test set, to a ratio of 4 to 1, respectively. A DT of maximum depth 7 and an MDT of maximum depth 5 were built for each dataset. We then compared accuracy (on the test set) and the size of the weak AXp's corresponding to the path followed by the instance to be explained. This operation was repeated 10 times for every data set with distinct random seeds resulting in 10 distinct partitions, and the values reported in Table 1 are the averages over these 10 runs.

Although the DTs were deeper, the average accuracy for MDTs was slightly better (although the difference was not found to be statistically significant on any particular dataset). Explanation sizes were found to be similar for the DTs and the MDTs, which was to be expected given our choice of depth-5 MDTs and depth-7 DTs, as explained above. However, a definite advantage of the MDTs was their relative compactness: complete depth-5 MDTs have 4 times less nodes than complete depth-7 DTs and in our experiments the depth-5 MDTs had on average 3 times less nodes than the depth-7 DTs. Since there are more opportunities for pruning in depth-7 trees than in depth-5 trees, it is to be expected that the ratio of 4 is not reached.

Thus, our experiments indicate that an MDT can provide a more compact model than a DT with similar accuracy and interpretability (in terms of the number of features required to explain decisions).

[2]https://gitlab.laas.fr/roc/emmanuel-hebrard/mdt-experiments

[3]It scales linearly with the number of instances, which is thus irrelevant here.

| data set | # features | | test accuracy | | tree size | | expl. length | |
|---|---|---|---|---|---|---|---|---|
| | unary | binary | DT | MDT | DT | MDT | DT | MDT |
| car_evaluation-bin | 14 | 455 | 0.9165 | 0.9150 | 35.1 | 13.6 | 1.36 | 1.30 |
| balance-scale-bin | 16 | 600 | 0.7849 | 0.8373 | 76.8 | 23.8 | 2.66 | 3.46 |
| car-un | 21 | 1050 | 0.9668 | 0.9734 | 58.0 | 14.9 | 2.93 | 1.23 |
| compas_discretized | 25 | 1500 | 0.6612 | 0.6667 | 102.1 | 30.9 | 5.40 | 5.09 |
| tic-tac-toe | 27 | 1755 | 0.9705 | 0.9378 | 39.8 | 20.2 | 2.23 | 3.92 |
| banknote-bin | 28 | 1890 | 0.9862 | 0.9876 | 19.7 | 11.8 | 1.23 | 1.89 |
| primary-tumor | 31 | 2325 | 0.7426 | 0.7603 | 41.5 | 21.2 | 1.96 | 4.01 |
| winequality-red-bin | 42 | 4305 | 0.9859 | 0.9884 | 13.5 | 8.9 | 1.25 | 2.44 |
| IndiansDiabetes-bin | 43 | 4515 | 0.6539 | 0.7162 | 114.9 | 27.5 | 5.64 | 5.02 |
| vote | 48 | 5640 | 0.9420 | 0.9489 | 15.9 | 10.8 | 1.73 | 2.11 |
| soybean | 50 | 6125 | 0.9386 | 0.9488 | 27.9 | 11.9 | 1.87 | 3.10 |
| adult_discretized | 59 | 8555 | 0.8540 | 0.8550 | 87.0 | 24.7 | 6.16 | 4.92 |
| hepatitis | 68 | 11390 | 0.7276 | 0.7862 | 12.5 | 8.7 | 1.30 | 1.99 |
| lymph | 68 | 11390 | 0.8194 | 0.8677 | 14.0 | 7.7 | 1.37 | 2.24 |
| HTRU_2-bin | 70 | 12075 | 0.9760 | 0.9778 | 100.2 | 24.1 | 5.28 | 4.33 |
| kr-vs-kp | 73 | 13140 | 0.9836 | 0.9908 | 35.2 | 14.9 | 1.88 | 2.94 |
| messidor-bin | 86 | 18275 | 0.6364 | 0.6450 | 85.1 | 28.3 | 6.54 | 4.91 |
| magic04-bin | 86 | 18275 | 0.8457 | 0.8453 | 125.0 | 31.0 | 7.49 | 5.56 |
| hypothyroid | 88 | 19140 | 0.9743 | 0.9775 | 52.5 | 19.1 | 3.51 | 2.69 |
| breast-cancer-un | 89 | 19580 | 0.9380 | 0.9438 | 25.4 | 12.8 | 1.07 | 1.45 |
| yeast | 89 | 19580 | 0.6980 | 0.7208 | 87.4 | 26.5 | 4.45 | 4.87 |
| seismic_bumps-bin | 91 | 20475 | 0.8992 | 0.9153 | 102.8 | 21.3 | 5.17 | 4.41 |
| anneal | 93 | 21390 | 0.8681 | 0.8687 | 51.1 | 15.2 | 4.25 | 1.77 |
| heart-cleveland | 95 | 22325 | 0.7367 | 0.7650 | 31.1 | 18.5 | 2.59 | 3.89 |
| german-credit | 112 | 31080 | 0.6805 | 0.6950 | 96.0 | 24.8 | 5.87 | 4.73 |
| diabetes | 112 | 31080 | 0.6630 | 0.7221 | 107.4 | 26.1 | 6.57 | 4.62 |
| mushroom | 119 | 35105 | 1.0000 | 1.0000 | 7.8 | 4.0 | 0.29 | 1.00 |
| breast-wisconsin | 120 | 35700 | 0.9328 | 0.9526 | 18.3 | 11.1 | 1.82 | 2.93 |
| australian-credit | 125 | 38750 | 0.8220 | 0.8265 | 59.7 | 19.9 | 3.66 | 4.07 |
| audiology | 148 | 54390 | 0.9295 | 0.9227 | 7.4 | 4.3 | 1.19 | 0.96 |
| taiwan_binarised | 205 | 104550 | 0.8130 | 0.8167 | 121.7 | 30.3 | 8.09 | 5.30 |
| bank_conv-bin | 212 | 111830 | 0.8961 | 0.8966 | 88.0 | 28.3 | 5.46 | 5.31 |
| pendigits | 216 | 116100 | 0.9970 | 0.9963 | 19.9 | 14.7 | 2.09 | 3.27 |
| letter | 224 | 124880 | 0.9915 | 0.9891 | 81.1 | 17.7 | 4.74 | 3.29 |
| segment | 235 | 137475 | 0.9991 | 0.9998 | 5.0 | 3.0 | 1.69 | 1.12 |
| letter_recognition-bin | 240 | 143400 | 0.9941 | 0.9949 | 50.9 | 13.9 | 3.89 | 3.33 |
| vehicle | 252 | 158130 | 0.9482 | 0.9524 | 26.2 | 13.2 | 2.41 | 3.38 |
| splice-1 | 287 | 205205 | 0.9447 | 0.9589 | 57.5 | 21.8 | 4.60 | 4.77 |
| biodeg-bin | 304 | 230280 | 0.8354 | 0.8340 | 72.4 | 23.2 | 5.22 | 3.82 |
| titanic-un | 333 | 276390 | 0.7904 | 0.8163 | 66.8 | 21.5 | 3.98 | 4.35 |
| spambase-bin | 386 | 371525 | 0.8772 | 0.8765 | 75.1 | 21.8 | 6.58 | 4.26 |
| ionosphere | 445 | 493950 | 0.8789 | 0.8732 | 19.8 | 12.2 | 1.37 | 2.29 |
| Statlog_satellite-bin | 539 | 724955 | 0.9660 | 0.9694 | 58.4 | 19.0 | 3.57 | 3.54 |
| forest-fires-un | 989 | 2442830 | 0.5269 | 0.5635 | 31.2 | 12.3 | 1.39 | 1.49 |
| wine1-un | 1276 | 4067250 | 0.6417 | 0.6361 | 12.0 | 6.2 | 0.07 | 0.28 |
| wine2-un | 1276 | 4067250 | 0.6730 | 0.6703 | 12.8 | 7.3 | 0.25 | 0.73 |
| wine3-un | 1276 | 4067250 | 0.7278 | 0.6722 | 13.7 | 6.9 | 0.45 | 0.61 |
| average | | | 0.8518 | 0.8612 | 52.4 | 17.3 | 3.29 | 3.17 |

Table 1. Comparison of DTs and MDTs learnt by Blossom.

## 7 Conclusion

We have shown the close link between classes of multivariate decision trees for which decisions can be explained in polynomial time and tractable constraint languages closed under complement. We have shown that tractable explainability applies to existing and well-studied classes of MDTs, such as oblique DTs over real domains, but also to novel classes of MDTs over finite domains. Such novel classes provide generalisations of classical DTs in that branching is possible not only on the value of a single variable but also according to specific (non-linear) conditions on two or more variables.

Interesting open questions concern the continued evaluation of the practical utility [12, 37] as well as the theoretical computational power of such generalised DTs. There is a rich history of the study of MDTs with linear conditions as a computational model, such as bounds on the depth of such decision trees to test the equality of two sets [43]. An avenue of future work is a similar theoretical study of the computational power of MDTs with (generalised) star-nested Horn constraints (studied in Section 4.1 and Section 5.1), or (generalised) square 2CNF formulas (studied in Section 4.3 and Section 5.2) to determine whether there is a substantial theoretical gain in depth or size when compared with classical DTs. Our experiments on a sublanguage of generalised square 2CNFs indicate that MDTs are more compact than DTs with the same accuracy and average explanation-lengths.

Another avenue of future research is the investigation of algorithms for learning MDTs with (generalised) star-nested Horn constraints or (generalised) square 2CNF formulas. Examples 12 and 13 show that we can extend classical univariate DTs by adding bivariate constraints either of the form $x_i \in [p, q] \vee x_j \in [r, s]$ or of the form $x_i \geq x_j$, while retaining tractable explainability. This may be a useful compromise between the expressive power of the constraint language and the learnability of the corresponding class of MDTs. In our experiments we used a learning algorithm based on a complete search for an optimal tree of bounded depth [20]. Alternative approaches are possible, inspired by methods for learning oblique DTs, such as a classical top-down greedy approach for recursively splitting nodes (possibly followed by a pruning step) [25, 41, 48], a bottom-up approach using clustering [5] or the successive optimisation of the split conditions at nodes [16, 24]. It is known that finding an optimal splitting hyperplane at a node of an oblique DT is NP-hard [25, 41]. An open theoretical question is the complexity of finding an optimal split at a node among (generalised) star-nested (anti-)Horn relations.

Our P/NP-hard dichotomy for boolean languages closed under complement is an interesting theoretical result which may find applications in other domains. Our generalisation of this dichotomy to boolean formulas of UI-literals can be seen as a foundation on which to build a future characterisation of all tractable finite-domain languages closed under complement.

An independent question is the so-called recognition problem: given an arbitrary multivariate DT, determine whether the set of constraints it uses is a sublanguage of one of the tractable languages we have identified. It is reasonable to assume that this problem would be solved off-line, if at all.

## Acknowledgments

This work was supported by the AI Interdisciplinary Institute ANITI, funded by the French program “Investing for the Future – PIA3” under grant agreement no. ANR-19-PI3A-0004 and by the ForML research project ANR-23-CE25-0009.

## References

[1] L. Amgoud and J. Ben-Naim. 2022. Axiomatic foundations of explainability. In *IJCAI*. L. D. Raedt, (Ed.) ijcai.org, 636–642. DOI: 10.24963/ijcai.2022/90.

[2] M. Arenas, P. Barceló, M. A. R. Orth, and B. Subercaseaux. 2022. On computing probabilistic explanations for decision trees. In *NeurIPS*. S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, and A. Oh, (Eds.)

[3] G. Audemard, S. Bellart, L. Bounia, F. Koriche, J. Lagniez, and P. Marquis. 2022. On the explanatory power of boolean decision trees. *Data Knowl. Eng.*, 142, 102088. DOI: 10.1016/j.datak.2022.102088.

[4] P. Barceló, M. Monet, J. Pérez, and B. Subercaseaux. 2020. Model interpretability through the lens of computational complexity. In *NeurIPS*. H. Larochelle, M. Ranzato, R. Hadsell, M. Balcan, and H. Lin, (Eds.) https://proceedings.neurips.cc/paper/2020/hash/b1adda14824f50ef24ff1c05bb66faf3-Abstract.html.

[5] R. C. Barros, P. A. Jaskowiak, R. Cerri, and A. C. P. de Leon Ferreira de Carvalho. 2014. A framework for bottom-up induction of oblique decision trees. *Neurocomputing*, 135, 3–12. DOI: 10.1016/j.neucom.2013.01.067.

[6] S. Bassan, G. Amir, and G. Katz. 2024. Local vs. global interpretability: A computational complexity perspective. In *Forty-first International Conference on Machine Learning, ICML*. OpenReview.net. https://openreview.net/forum?id=veEjiN2w9F.

[7] F. Bollwein and S. Westphal. 2021. A branch & bound algorithm to determine optimal bivariate splits for oblique decision tree induction. *Appl. Intell.*, 51, 10, 7552–7572. https://doi.org/10.1007/s10489-021-02281-x.

[8] L. Breiman, J. Friedman, R. Olshen, and C. Stone. 1984. *Classification and Regression Trees*. Wadsworth & Brooks.

[9] C. E. Brodley and P. E. Utgoff. 1995. Multivariate decision trees. *Mach. Learn.*, 19, 1, 45–77. DOI: 10.1007/BF00994660.

[10] A. A. Bulatov. 2017. A dichotomy theorem for nonuniform csps. In *FOCS*. C. Umans, (Ed.) IEEE Computer Society, 319–330. DOI: 10.1109/FOCS.2017.37.

[11] M. Calautti, E. Malizia, and C. Molinaro. 2025. On the complexity of global necessary reasons to explain classification. *CoRR*, abs/2501.06766.

[12] L. Cañete-Sifuentes, R. Monroy, and M. A. Medina-Pérez. 2021. A review and experimental comparison of multivariate decision trees. *IEEE Access*, 9, 110451–110479. DOI: 10.1109/ACCESS.2021.3102239.

[13] C. Carbonnel and M. C. Cooper. 2016. Tractability in constraint satisfaction problems: a survey. *Constraints An Int. J.*, 21, 2, 115–144. DOI: 10.1007/S10601-015-9198-6.

[14] C. Carbonnel, M. C. Cooper, and J. Marques-Silva. 2023. Tractable explaining of multivariate decision trees. In *KR 2023*. P. Marquis, T. C. Son, and G. Kern-Isberner, (Eds.), 127–135. DOI: 10.24963/KR.2023/13.

[15] M. Á. Carreira-Perpiñán and S. S. Hada. 2021. Counterfactual explanations for oblique decision trees: exact, efficient algorithms. In *AAAI*. AAAI Press, 6903–6911. https://ojs.aaai.org/index.php/AAAI/article/view/16851.

[16] M. Á. Carreira-Perpiñán and P. Tavallali. 2018. Alternating optimization of decision trees, with application to learning sparse oblique trees. In *NeurIPS*. S. Bengio, H. M. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, (Eds.), 1219–1229. https://proceedings.neurips.cc/paper/2018/hash/185c29dc24325934ee377cfda20e414c-Abstract.html.

[17] Z. Chen and S. Toda. 1995. The complexity of selecting maximal solutions. *Inf. Comput.*, 119, 2, 231–239. DOI: 10.1006/inco.1995.1087.

[18] M. C. Cooper, D. A. Cohen, and P. Jeavons. 1994. Characterising tractable constraints. *Artif. Intell.*, 65, 2, 347–361. DOI: 10.1016/0004-3702(94)90021-3.

[19] M. C. Cooper and J. Marques-Silva. 2023. Tractability of explaining classifier decisions. *Artif. Intell.*, 316.

[20] E. Demirovic, E. Hebrard, and L. Jean. 2023. Blossom: an anytime algorithm for computing optimal decision trees. In *International Conference on Machine Learning, ICML 2023, 23-29 July 2023, Honolulu, Hawaii, USA* (Proceedings of Machine Learning Research). A. Krause, E. Brunskill, K. Cho, B. Engelhardt, S. Sabato, and J. Scarlett, (Eds.) Vol. 202. PMLR, 7533–7562. https://proceedings.mlr.press/v202/demirovic23a.html.

[21] Y. Dhebar, K. Deb, S. Nageshrao, L. Zhu, and D. Filev. 2024. Toward interpretable-ai policies using evolutionary nonlinear decision trees for discrete-action systems. *IEEE Transactions on Cybernetics*, 54, 1, 50–62. DOI: 10.1109/TCYB.2022.3180664.

[22] J. H. Good, T. Kovach, K. Miller, and A. Dubrawski. 2023. Feature learning for interpretable, performant decision trees. In *NeurIPS*. A. Oh, T. Naumann, A. Globerson, K. Saenko, M. Hardt, and S. Levine, (Eds.)

[23] R. Guidotti, A. Monreale, S. Ruggieri, F. Turini, F. Giannotti, and D. Pedreschi. 2019. A survey of methods for explaining black box models. *ACM Comput. Surv.*, 51, 5, 93:1–93:42. DOI: 10.1145/3236009.

[24] S. S. Hada, M. Á. Carreira-Perpiñán, and A. Zharmagambetov. 2024. Sparse oblique decision trees: a tool to understand and manipulate neural net features. *Data Min. Knowl. Discov.*, 38, 5, 2863–2902.

[25] D. G. Heath, S. Kasif, and S. Salzberg. 1993. Induction of oblique decision trees. In *IJCAI*. R. Bajcsy, (Ed.) Morgan Kaufmann, 1002–1007.

[26] X. Huang, Y. Izza, A. Ignatiev, M. C. Cooper, N. Asher, and J. Marques-Silva. 2022. Tractable explanations for d-DNNF classifiers. In *AAAI*. AAAI Press, 5719–5728. https://ojs.aaai.org/index.php/AAAI/article/view/20514.

[27] A. Ignatiev, N. Narodytska, N. Asher, and J. Marques-Silva. 2020. From contrastive to abductive explanations and back again. In *AIxIA 2020 - Advances in Artificial Intelligence* (Lecture Notes in Computer Science). M. Baldoni and S. Bandini, (Eds.) Vol. 12414. Springer, 335–355.

[28] A. Ignatiev, N. Narodytska, and J. Marques-Silva. 2019. Abduction-based explanations for machine learning models. In *AAAI*. AAAI Press, 1511–1519. DOI: 10.1609/aaai.v33i01.33011511.

[29] Y. Izza, A. Ignatiev, and J. Marques-Silva. 2022. On tackling explanation redundancy in decision trees. *J. Artif. Intell. Res.*, 75, 261–321. DOI: 10.1613/jair.1.13575.

[30] P. Jeavons, D. A. Cohen, and M. C. Cooper. 1998. Constraints, consistency and closure. *Artif. Intell.*, 101, 1-2, 251–265. DOI: 10.1016/S0004-3702(98)00022-8.

[31] P. Jeavons, D. A. Cohen, and M. Gyssens. 1995. A unifying framework for tractable constraints. In *Principles and Practice of Constraint Programming - CP'95* (LNCS). U. Montanari and F. Rossi, (Eds.) Vol. 976. Springer, 276–291. DOI: 10.1007/3-540-60299-2_17.
[32] P. Jeavons, D. A. Cohen, and M. Gyssens. 1997. Closure properties of constraints. *J. ACM*, 44, 4, 527–548. DOI: 10.1145/263867.263489.
[33] P. Jeavons and M. C. Cooper. 1995. Tractable constraints on ordered domains. *Artif. Intell.*, 79, 2, 327–339. DOI: 10.1016/0004-3702(95)00107-7.
[34] R. Kairgeldin and M. Á. Carreira-Perpiñán. 2024. Bivariate decision trees: smaller, interpretable, more accurate. In *KDD*. R. Baeza-Yates and F. Bonchi, (Eds.) ACM, 1336–1347.
[35] M. Kelly, R. Longjohn, and K. Nottingham. [n. d.] The UCI machine learning repository, https://archive.ics.uci.edu. ().
[36] S. K. Lahiri and M. Musuvathi. 2005. An efficient decision procedure for UTVPI constraints. In *Frontiers of Combining Systems, 5th International Workshop* (Lecture Notes in Computer Science). B. Gramlich, (Ed.) Vol. 3717. Springer, 168–183. DOI: 10.1007/11559306_9.
[37] Y. Li, M. Dong, and R. Kothari. 2005. Classifiability-based omnivariate decision trees. *IEEE Trans. Neural Networks*, 16, 6, 1547–1560. DOI: 10.1109/TNN.2005.852864.
[38] J. Marques-Silva. 2024. Logic-based explainability: past, present and future. In *Leveraging Applications of Formal Methods, Verification and Validation. ISoLA, Proceedings, Part IV* (Lecture Notes in Computer Science). T. Margaria and B. Steffen, (Eds.) Vol. 15222. Springer, 181–204.
[39] J. Marques-Silva and A. Ignatiev. 2022. Delivering trustworthy AI through formal XAI. In *AAAI*. AAAI Press, 12342–12350. https://ojs.aaai.org/index.php/AAAI/article/view/21499.
[40] T. Miller. 2019. Explanation in artificial intelligence: insights from the social sciences. *Artif. Intell.*, 267, 1–38. DOI: 10.1016/j.artint.2018.07.007.
[41] S. K. Murthy, S. Kasif, and S. Salzberg. 1994. A system for induction of oblique decision trees. *J. Artif. Intell. Res.*, 2, 1–32. DOI: 10.1613/jair.63.
[42] S. Ordyniak, G. Paesani, M. Rychlicki, and S. Szeider. 2024. Explaining decisions in ML models: A parameterized complexity analysis. In *KR*. P. Marquis, M. Ortiz, and M. Pagnucco, (Eds.) https://doi.org/10.24963/kr.2024/53.
[43] E. M. Reingold. 1972. On the optimality of some set algorithms. *J. ACM*, 19, 4, 649–659. DOI: 10.1145/321724.321730.
[44] E. G. Rodrigo, J. C. Alfaro, J. A. Aledo, and J. A. Gámez. 2024. Label ranking oblique trees. *Knowl. Based Syst.*, 296, 111882.
[45] T. J. Schaefer. 1978. The complexity of satisfiability problems. In *Proceedings of the 10th Annual ACM Symposium on Theory of Computing*. R. J. Lipton, W. A. Burkhard, W. J. Savitch, E. P. Friedman, and A. V. Aho, (Eds.) ACM, 216–226. DOI: 10.1145/800133.804350.
[46] A. Shih, A. Choi, and A. Darwiche. 2018. A symbolic approach to explaining bayesian network classifiers. In *IJCAI*. J. Lang, (Ed.) ijcai.org, 5103–5111. DOI: 10.24963/ijcai.2018/708.
[47] S. Wäldchen, J. MacDonald, S. Hauch, and G. Kutyniok. 2021. The computational complexity of understanding binary classifier decisions. *J. Artif. Intell. Res.*, 70, 351–387. DOI: 10.1613/jair.1.12359.
[48] D. C. Wickramarachchi, B. L. Robertson, M. Reale, C. J. Price, and J. Brown. 2016. HHCART: an oblique decision tree. *Comput. Stat. Data Anal.*, 96, 12–23. DOI: 10.1016/j.csda.2015.11.006.
[49] H. Zhu, P. Murali, D. T. Phan, L. M. Nguyen, and J. Kalagnanam. 2020. A scalable MIP-based method for learning optimal multivariate decision trees. In *NeurIPS*. H. Larochelle, M. Ranzato, R. Hadsell, M. Balcan, and H. Lin, (Eds.) https://proceedings.neurips.cc/paper/2020/hash/1373b284bc381890049e92d324f56de0-Abstract.html.
[50] D. Zhuk. 2020. A proof of the CSP dichotomy conjecture. *J. ACM*, 67, 5, 30:1–30:78. DOI: 10.1145/3402029.

## A Reproducibility Checklist for JAIR

Select the answers that apply to your research – one per item.

### All articles:

(1) All claims investigated in this work are clearly stated. [yes]
(2) Clear explanations are given how the work reported substantiates the claims. [yes]
(3) Limitations or technical assumptions are stated clearly and explicitly. [yes]
(4) Conceptual outlines and/or pseudo-code descriptions of the AI methods introduced in this work are provided, and important implementation details are discussed. [yes]
(5) Motivation is provided for all design choices, including algorithms, implementation choices, parameters, data sets and experimental protocols beyond metrics. [yes]

### Articles containing theoretical contributions:

Does this paper make theoretical contributions? [yes]
If yes, please complete the list below.

(1) All assumptions and restrictions are stated clearly and formally. [yes]
(2) All novel claims are stated formally (e.g., in theorem statements). [yes]
(3) Proofs of all non-trivial claims are provided in sufficient detail to permit verification by readers with a reasonable degree of expertise (e.g., that expected from a PhD candidate in the same area of AI). [yes]
(4) Complex formalism, such as definitions or proofs, is motivated and explained clearly. [yes]
(5) The use of mathematical notation and formalism serves the purpose of enhancing clarity and precision; gratuitous use of mathematical formalism (i.e., use that does not enhance clarity or precision) is avoided. [yes]
(6) Appropriate citations are given for all non-trivial theoretical tools and techniques. [yes]

### Articles reporting on computational experiments:

Does this paper include computational experiments? [yes]
If yes, please complete the list below.

(1) All source code required for conducting experiments is included in an online appendix or will be made publicly available upon publication of the paper. The online appendix follows best practices for source code readability and documentation as well as for long-term accessibility. [yes]
(2) The source code comes with a license that allows free usage for reproducibility purposes. [yes]
(3) The source code comes with a license that allows free usage for research purposes in general. [yes]
(4) Raw, unaggregated data from all experiments is included in an online appendix or will be made publicly available upon publication of the paper. The online appendix follows best practices for long-term accessibility. [yes]
(5) The unaggregated data comes with a license that allows free usage for reproducibility purposes. [yes]
(6) The unaggregated data comes with a license that allows free usage for research purposes in general. [yes]
(7) If an algorithm depends on randomness, then the method used for generating random numbers and for setting seeds is described in a way sufficient to allow replication of results. [yes]
(8) The execution environment for experiments, the computing infrastructure (hardware and software) used for running them, is described, including GPU/CPU makes and models; amount of memory (cache and RAM); make and version of operating system; names and versions of relevant software libraries and frameworks. [yes]
(9) The evaluation metrics used in experiments are clearly explained and their choice is explicitly motivated. [yes]
(10) The number of algorithm runs used to compute each result is reported. [yes]
(11) Reported results have not been "cherry-picked" by silently ignoring unsuccessful or unsatisfactory experiments. [yes (results have not been cherry-picked)]
(12) Analysis of results goes beyond single-dimensional summaries of performance (e.g., average, median) to include measures of variation, confidence, or other distributional information. [no] (but not relevant since we claim equality rather than difference)
(13) All (hyper-) parameter settings for the algorithms/methods used in experiments have been reported, along with the rationale or method for determining them. [yes]
(14) The number and range of (hyper-) parameter settings explored prior to conducting final experiments have been indicated, along with the effort spent on (hyper-) parameter optimisation. [yes]

(15) Appropriately chosen statistical hypothesis tests are used to establish statistical significance in the presence of noise effects. [NA]

### Articles using data sets:

Does this work rely on one or more data sets (possibly obtained from a benchmark generator or similar software artifact)? [yes]

If yes, please complete the list below.

(1) All newly introduced data sets are included in an online appendix or will be made publicly available upon publication of the paper. The online appendix follows best practices for long-term accessibility with a license that allows free usage for research purposes. [NA (no new datasets)]
(2) The newly introduced data set comes with a license that allows free usage for reproducibility purposes. [NA]
(3) The newly introduced data set comes with a license that allows free usage for research purposes in general. [NA]
(4) All data sets drawn from the literature or other public sources (potentially including authors' own previously published work) are accompanied by appropriate citations. [yes]
(5) All data sets drawn from the existing literature (potentially including authors' own previously published work) are publicly available. [yes]
(6) All new data sets and data sets that are not publicly available are described in detail, including relevant statistics, the data collection process and annotation process if relevant. [NA]
(7) All methods used for preprocessing, augmenting, batching or splitting data sets (e.g., in the context of hold-out or cross-validation) are described in detail. [yes]

Received 29 July 2025; accepted