Journal of Artificial Intelligence Research 72 (2021) 285-328 Submitted 02/2021; published 10/2021

On Quantifying Literals in Boolean Logic and Its Applications to Explainable AI

Adnan Darwiche DARWICHE@CS.UCLA.EDU
Computer Science Department, UCLA,
Los Angeles, CA 90095 USA

Pierre Marquis MARQUIS@CRIL.FR
CRIL, Université d'Artois & CNRS,
Institut Universitaire de France
F-62307, Lens Cedex, France

Abstract

Quantified Boolean logic results from adding operators to Boolean logic for existentially and universally quantifying variables. This extends the reach of Boolean logic by enabling a variety of applications that have been explored over the decades. The existential quantification of literals (variable states) and its applications have also been studied in the literature. In this paper, we complement this by studying universal literal quantification and its applications, particularly to explainable AI. We also provide a novel semantics for quantification, discuss the interplay between variable/literal and existential/universal quantification. We further identify some classes of Boolean formulas and circuits on which quantification can be done efficiently. Literal quantification is more fine-grained than variable quantification as the latter can be defined in terms of the former. This leads to a refinement of quantified Boolean logic with literal quantification as its primitive.

1. Introduction

To quantify a variable from a Boolean formula is to eliminate that variable from the formula. This quantification process can be performed either existentially or universally, each leading to a different semantics and a different set of applications. The use of quantification in Boolean logic dates back at least to George Boole's work (Boole, 1854), where he used universal quantification to perform some forms of logical deduction. Existential quantification has a particularly intuitive interpretation as it can be viewed as a process of removing from a formula all and only the information which pertains to the quantified variable. Due to this semantics, which is referred to as forgetting (Lin & Reiter, 1994), existential quantification has received much attention in AI and database theory, particularly in the management of inconsistent information. This use dates back at least to (Weber, 1986) who employed existential quantification to combine pieces of information that may contradict each other. Variable quantification plays a prominent role in complexity theory too since deciding the validity of quantified Boolean formulas (QBFs) is the canonical PSPACE-complete problem; see, for example, (Papadimitriou, 1994). The validity of QBFs has also been used to characterize the polynomial hierarchy (Stockmeyer, 1977). In contrast to AI where existential quantification has had a more dominant role than universal quantification, the role of these two forms of quantification has been more symmetrical in other areas of computer science particularly in complexity theory.

The interpretation of existential variable quantification as a process of forgetting information prompted a refinement in which literals (states of variables) are also existentially quantified (Lang et al., 2003). With this advance, one can now remove information that pertains to a literal from a formula. Existential literal quantification is more fine-grained than existential variable quantification as the latter can be formulated in terms of the former. This has led to further applications which we review later.

The dominance of existential quantification within AI is worth noting. We attribute this to the lack of an intuitive enough interpretation of universal quantification, which is a gap that we aim to address in this work. We started our study by an observation that some recent work on explainable AI (Darwiche & Hirth, 2020) can be formulated in terms of universal quantification, particularly universal literal quantification which has not been discussed in the literature before this work. This prompted us to formalize this notion and to elaborately investigate its connections to explainable AI. Our investigation led us to interpret universal quantification as a *selection* process (in contrast to a forgetting process), which gave rise to many implications. It also led us to some results on the efficient computation of universal literal quantification (e.g., being tractable on CNFs) which has further implications on the efficient computation of explainable AI queries.

In explainable AI, one is typically interested in reasoning about the behavior of classifiers which make decisions on instances. For example, one may wish to understand why a classifier made a particular decision. One may also wish to determine whether a decision is biased (i.e., would be different if we were to only change some protected features of the instance). Such classifiers can be represented using Boolean formulas, even in some cases where they have a numeric form that is learned from data.[1] In particular, one can use a Boolean formula to represent positive instances and use its negation to represent negative instances. A major insight underlying our results is that many explainable AI queries correspond to a process of selecting instances with particular properties, where the Boolean formula characterizing such instances constitutes the answer to the explainable AI query. Moreover, the universal quantification of literals (and variables) can be used to select such instances. We use this to formulate and generalize some of the recently introduced notions in explainable AI. We also give examples of additional queries that have not been treated earlier to emphasize the open ended applications that could be enabled by the new results.

Our formulation of universal literal quantification implies an alternate treatment of existential literal quantification compared to the existing literature (due to the duality between them). It also implies a new treatment of variable quantification as this is subsumed by literal quantification. Our treatment is based on the novel notion of boundary models. These are models of a Boolean formula that can become models of its negation if we flip a single variable. In the context of classifiers, these correspond to instances whose label may change if we flip a single feature. We also provide a wealth of results on boundary models, which furthers our understanding of Boolean logic; for example, that the boundary models of a Boolean formula characterize all its models and can be exponentially fewer in count.

1. One can capture the input-output behavior of machine learning classifiers with discrete features using Boolean formulas, either through an encoding or a compilation process. This has been shown for Bayesian network classifiers, random forests and some types of neural networks. For some example works along these lines, see (Chan & Darwiche, 2003; Narodytska et al., 2018; Shih et al., 2019; Ignatiev et al., 2019a, 2019b; Shi et al., 2020; Audemard et al., 2020; Choi et al., 2020).

Notation	**Description**
Σ	A finite set of Boolean variables
ω	A truth assignment (world) which maps Σ to $\{0,1\}$
x, X	Positive literals of Boolean variable X
$\bar{x}, \neg X$	Negative literals of Boolean variable X
$\ell, \bar{\ell}$	A literal (positive or negative) and its negation
$\omega[\ell]$	If $\bar{\ell} \in \omega$, $\omega[\ell]$ is the result of replacing $\bar{\ell}$ with ℓ in ω; otherwise, $\omega[\ell] = \omega$
$\omega \models \varphi$	World ω satisfies formula φ
$\varphi \models \psi$	Formula φ implies formula ψ
$M(\varphi)$	The set of models of formula φ
$BM(\varphi)$	The set of boundary models of formula φ
$R(\varphi)$	The set of b-rules for formula φ

Table 1: Some of the key notation adopted in this paper.

We start in Section 2 with some preliminaries on Boolean logic, where we also introduce the notion of boundary models. We then review variable quantification and some of its applications in Section 3. We study boundary models further in Section 4 where we develop a number of additional results. Some of these results are used in our upcoming treatment while others are of a more general interest to the study of Boolean logic. We then formalize and study universal literal quantification in Section 5, where we also review existential literal quantification and some of its applications. We then turn our attention in Section 6 to the computation of literal quantification on various types of formulas and circuits, including CNFs, DNFs, Decision-DNNFs (Huang & Darwiche, 2007) and SDDs (Darwiche, 2011), which include OBDDs (Bryant, 1986) as a special case. Section 7 constitutes a significant portion of the paper and is dedicated to the interpretation of universal quantification as a selection process and its applications to explainable AI. We finally close with some concluding remarks in Section 8. Proofs of all results can be found in Appendixes B and C.

2. Boolean Logic Preliminaries

We start with some notational conventions; see also Table 1. We assume a finite set of Boolean variables Σ where x and $\bar{x}$ denote the positive and negative literals of Boolean variable X. We may also use X and $\neg X$ to denote these literals. We use $\top$ and $\bot$ to denote the Boolean constants true and false. A *world,* typically denoted by ω, is a truth assignment (a mapping from Σ to $\{0,1\}$), often represented as a set of literals that contains exactly one literal for each Boolean variable in Σ. Alternatively, a world can be represented by a sequence of literals, sometimes using commas as separators. When a world ω satisfies a Boolean formula φ we say it is a *model* of the formula and write $\omega \models \varphi$. Otherwise, ω is a model of $\neg\varphi$, aka a *counter model* of φ. We use $M(\varphi)$ to denote the models of formula φ. Whenever $M(\varphi) \subseteq M(\psi)$ holds, formula ψ is said to be a *logical consequence* of formula φ (or equivalently, φ implies ψ). Furthermore, when $M(\varphi) = M(\psi)$ holds, φ and ψ are said to be *logically equivalent.* For a literal ℓ of variable X, we use $\omega[\ell]$ to denote the world that results from replacing the literal of variable X in ω with literal ℓ. For example, if $\omega = xy\bar{z}$, then $\omega[\bar{x}] = \bar{x}y\bar{z}$ and $\omega[x] = xy\bar{z}$.

The following novel definition is fundamental as it will play a critical role when defining the semantics of literal quantification.

Definition 1. *A world ω is said to be an* ℓ-boundary model *of Boolean formula φ iff ω contains literal ℓ, ω is a model of φ and $\omega[\bar{\ell}]$ is a model of $\neg\varphi$.*

That is, model ω of formula φ becomes a model of $\neg\varphi$ once we flip literal ℓ in ω to $\bar{\ell}$. Consider the formula $\varphi = (x \Rightarrow y) \wedge (y \Rightarrow x)$. World $\omega = \{x, y\}$ is an x-boundary model for formula φ as it contains literal x, is a model of φ but becomes a model of its negation $\neg\varphi$ once we flip literal x to $\bar{x}$. World ω is also a y-boundary model for formula φ. We will use $BM(\varphi)$ to denote the set of all boundary models of formula φ.

Boundary models will be used to define the semantics of single-literal quantification. The next, novel notion will be used to define the semantics of multiple-literal quantification.

Definition 2. *Let α be a set of literals. A world ω is said to be an* α-independent model *of formula φ iff $\alpha \subseteq \omega$ and $\omega \setminus \alpha \models \varphi$.*

Consider a world ω and literals α such that $\alpha \subseteq \omega$. If world ω is an α-independent model of formula φ, we can flip any literals of α in world ω while maintaining the world as a model of φ. In this case, the model ω cannot be ℓ-boundary for any literal $\ell \in \alpha$. The converse is not true though. Flipping a single literal $\ell \in \alpha$ may maintain ω as a model of φ for all $\ell \in \alpha$, yet flipping multiple literals in α may not. Consider formula $\varphi = (x \vee y) \wedge z$ and its model $\omega = \{x, y, z\}$. This model is not x-boundary since $\omega[\bar{x}] = \{\bar{x}, y, z\}$ is also a model of φ. It is also not y-boundary since $\omega[\bar{y}] = \{x, \bar{y}, z\}$ is also a model of φ. However, ω is not an $\{x, y\}$-independent model of φ since $\omega[\bar{x}, \bar{y}] = \{\bar{x}, \bar{y}, z\}$ is not a model of φ. On the other hand, world $\{x, y, \bar{z}\}$ is an $\{x, y\}$-independent model of $\neg\varphi$ as we can flip literals x and y in any manner while maintaining the world as a model for $\neg\varphi$.

We next review a number of classical notions from Boolean logic. A *term* is a set of literals over distinct variables and represents the conjunction of these literals. A term is therefore consistent and the empty term represents $\top$. We will sometimes denote a term or a world such as $\{x, \bar{y}\}$ using the tuple $x\bar{y}$ or $(x, \bar{y})$. A *clause* is a set of literals over distinct variables and represents the disjunction of these literals. A clause is therefore non-valid and the empty clause represents $\bot$. A *Disjunctive Normal Form (DNF)* is a set of terms representing the disjunction of these terms. A *Conjunctive Normal Form (CNF)* is a set of clauses representing the conjunction of the clauses. A *Negation Normal Form (NNF)* is a formula which contains only the constants $\top$, $\bot$, the connectives $\wedge$, $\vee$, $\neg$ and where negations appear only next to constants or variables. DNF and CNF are subsets of NNF. An *NNF circuit* is a Boolean circuit satisfying the conditions of an NNF formula. A formula/circuit is *monotone* iff it is equivalent to a formula in which literals x and $\bar{x}$ cannot both occur for any variable X.

An *implicate* of a formula φ is a clause implied by φ. A *prime implicate* of φ is an implicate of φ that is not a superset of another implicate of φ. A *implicant* of φ is a term that implies φ. A *prime implicant* of φ is an implicant of φ that is not a superset of another implicant of φ. The *resolution* of clauses $x \vee \alpha$ and $\bar{x} \vee \beta$ on variable X leads to clause $\alpha \vee \beta$ when α and β share no complementary literals. Adding clauses to a CNF using resolution does not change the models of the CNF. A CNF is *closed* under resolution on variable X iff the result of each resolution on X is in the CNF. The *consensus* of terms $x \wedge \alpha$ and $\bar{x} \wedge \beta$

on variable X leads to term $\alpha \wedge \beta$ when α and β share no complementary literals. Adding terms to a DNF using consensus does not change the models of the DNF. A DNF is *closed* under consensus on variable X iff the result of each consensus on X is in the DNF.

The occurrence of a variable in a Boolean formula is well defined, but the occurrence of a literal in a formula can be ambiguous (e.g., whether the positive literal x occurs in $\neg(\bar{x} \vee y)$). We will therefore refer to literal occurrence only with respect to NNF since this is well defined. We conclude this section with three additional notions.

- The *conditioning* of formula φ on positive literal x, denoted $\varphi|x$, is obtained by replacing every occurrence of variable X in φ with $\top$. The conditioning of formula φ on negative literal $\bar{x}$, denoted $\varphi|\bar{x}$, is obtained by replacing every occurrence of variable X in φ with $\bot$. Hence, variable X does not occur in $\varphi|x$ or in $\varphi|\bar{x}$.
- A formula φ is *independent of variable* X iff φ is equivalent to some formula in which variable X does not occur. For instance, $\varphi = (x \wedge y) \vee (\bar{x} \wedge y)$ is independent of variable X since φ is equivalent to y and the latter does not mention variable X.
- A formula φ is *independent of literal* ℓ iff φ is equivalent to some NNF in which literal ℓ does not occur (Lang et al., 2003). Thus, $\varphi = (x \wedge y) \vee (\bar{x} \wedge y)$ is independent of literal $\bar{x}$ since φ is equivalent to NNF y which does not mention literal $\bar{x}$.

It follows that φ is independent of variable X iff φ is independent of literals x and $\bar{x}$.

3. Variable Quantification

A variable X can be quantified either existentially or universally from a Boolean formula, leading to another formula that does not depend on variable X. The result of existentially quantifying variable X from formula φ is denoted by $\exists X \cdot \varphi$ and the result of universally quantifying it is denoted by $\forall X \cdot \varphi$. Existential quantification received more attention in the AI literature due to its more intuitive semantics which facilitated applications. According to this semantics, the existential quantification of variable X from formula φ removes information from φ but it only removes information that depends on variable X. Hence, every logical consequence of formula φ is also a logical consequence of its quantification $\exists X \cdot \varphi$ as long as that consequence does not depend on variable X. We next provide the formal definition of existential variable quantification and its semantics.

Definition 3. *The existential quantification of variable X from Boolean formula φ, denoted $\exists X \cdot \varphi$, is defined as any Boolean formula that is logically equivalent to $(\varphi|x) \vee (\varphi|\bar{x})$.*

Proposition 1. *$\exists X \cdot \varphi$ is a logically strongest formula that is both independent of variable X and implied by formula φ. $\exists X \cdot \varphi$ is unique up to logical equivalence.*

Consider the formula $\varphi = (x \Rightarrow y) \wedge (y \Rightarrow z)$. Existentially quantifying variable Y yields $\exists Y \cdot \varphi = (x \Rightarrow z)$ which removes only the information that φ has about variable Y. This is why existential variable quantification is typically referred to as "forgetting" in the AI literature, following (Lin & Reiter, 1994) who used existential quantification in a first-order setting to forget facts and relations; see also (Eiter & Kern-Isberner, 2019). Existential variable quantification was employed earlier in (Weber, 1986) to maintain consistency between two conflicting formulas (e.g., forgetting some of our current beliefs when they conflict with new observations). A version of Proposition 1 appeared in (Katsuno & Mendelzon, 1989) who showed that existential variable quantification (referred to as *elimination*) is the only operator satisfying the properties stated in this proposition. Existential variable quantification corresponds to a projection operation since the models of $\exists X \cdot \varphi$ are precisely the models of φ projected onto $\Sigma \setminus X$. This operation has a number of applications in AI. For example, it is a key mechanism for implementing elementary tasks (progression, regression) when reasoning about actions via transition formulas so it has been considered for decades in planning within nondeterministic domains (e.g., Cimatti et al., 1998).

We next define the dual notion of universal variable quantification and its semantics.

Definition 4. *The universal quantification of variable X from Boolean formula φ, denoted $\forall X \cdot \varphi$, is defined as any Boolean formula that is logically equivalent to $(\varphi|x) \wedge (\varphi|\bar{x})$.*

Proposition 2. *$\forall X \cdot \varphi$ is a logically weakest formula that is both independent of variable X and implies formula φ. $\forall X \cdot \varphi$ is unique up to logical equivalence.*

The following duality relates existential and universal variable quantification.

Proposition 3. *$\exists X \cdot \varphi = \neg(\forall X \cdot \neg\varphi)$ and $\forall X \cdot \varphi = \neg(\exists X \cdot \neg\varphi)$.*

George Boole used universal variable quantification in Chapter VII of his book (Boole, 1854), which he also referred to as *elimination* (the chapter was titled "On Elimination"). Boole employed the property that φ is inconsistent only if $\forall X \cdot \varphi$ is inconsistent to devise an inference rule which allowed him, for example, to infer $y = y \wedge z$ from $y = x \wedge z$.[2] He also observed that the order in which we quantify multiple variables does not matter.

As mentioned earlier, existential quantification received more attention than universal quantification in the AI literature. This is largely due to its semantics as a forgetting operator and the central role that forgetting plays in managing inconsistent information and other areas such as planning in nondeterministic domains. For example, a fairly general framework for reasoning with inconsistent information was proposed in (Lang & Marquis, 2010). The framework was based on the notion of *recoveries,* which are sets of variables whose forgetting enables one to restore consistency. Several criteria for defining preferred recoveries were proposed, depending on whether the focus is laid on the relative relevance of variables or the relative entrenchment of certain information (or both). Forgetting has also been employed to resolve conflicts in logic programs including classical logic programs with negation as failure (Zhang et al., 2005; Zhang & Foo, 2006). Notions and techniques for forgetting in logic programs were also adapted to forgetting concepts in ontologies (Eiter et al., 2006). Forgetting has also been widely used for defining update operators which incorporate the effect of an action (expressed as a "change formula") into a base formula that represents current beliefs. Many update operators use the "forget-then-conjoin" scheme where one forgets every variable that the change formula depends on and then conjoins the

2. Boole operated in an algebraic setting where the equivalence between y and $x \wedge z$ would be written as $y = xz$ and $y - xz = 0$. He viewed such an expression as a function of x, $f(x) = y - xz = 0$, and utilized universal quantification to write: $f(x) = 0$ only if $f(0)f(1) = 0$. Applying this to the previous expression gives $f(0)f(1) = (y - 0 \times z)(y - 1 \times z) = 0$. This simplifies to $y(y - z) = 0$ and can be expressed as $y = yz$, which exemplifies how Boole used universal quantification to perform deduction.

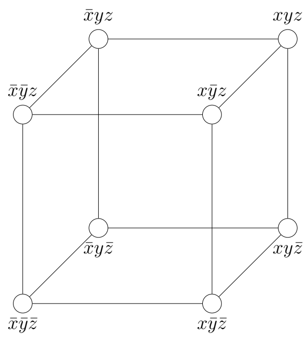
resulting base formula with the change formula (e.g., Winslett, 1990; Doherty et al., 1998, 2000; Herzig, 1996; Herzig & Rifi, 1998, 1999). Just as existential quantification can be understood as a forgetting operator, we will later show that universal quantification can be understood as a *selection* operator. We will also argue that the notion of selection is central to explainable AI as forgetting is central to belief revision and update.

Variable quantification in Boolean logic can be considered from two distinct points of view. According to one view, a quantifier is an *elimination operator* which transforms one Boolean formula into another. According to the second view, quantifiers are *connectives* which lead to a more general and succinct logical representation, known as *Quantified Boolean Formulas (QBFs)*. While quantifiers have been mostly used as elimination operators in AI, they have been mostly used as connectives in complexity theory (e.g., Papadimitriou, 1994). Quantification plays a key role in this context as the validity problem for QBFs is the canonical PSPACE-complete problem: there is a polynomial-space algorithm for deciding the validity of a QBF, and every decision problem which has a polynomial-space algorithm can be reduced efficiently into the validity problem for QBFs. QBFs are particularly important for characterizing the polynomial hierarchy (Stockmeyer, 1977), where the notion of a prenex and closed QBF plays a central role. Consider a standard Boolean formula φ and let $\mathbb{X}_1, \ldots, \mathbb{X}_n$ be a partition of its variables. A (prenex and closed) QBF has the form $Q_1\mathbb{X}_1, \ldots, Q_n\mathbb{X}_n \cdot \varphi$ where $Q_1, \ldots, Q_n$ are alternating quantifiers in $\{\forall, \exists\}$. This formula is said to be *prenex* because φ does not contain any quantifiers. Moreover, it is said to be *closed* because every variable of φ belongs to some $\mathbb{X}_i$. Since every variable is quantified in a closed QBF, the QBF is either valid (equivalent to true) or inconsistent (equivalent to false).[3] A standard assumption of complexity theory is that the polynomial hierarchy does not collapse. This is equivalent to saying that each addition of a block of quantifiers makes the validity problem computationally harder since the validity problem for prenex and closed QBFs is Σ_n^p-complete if $Q_1 = \exists$ and Π_n^p-complete if $Q_1 = \forall$.

While QBFs can be harder to decide compared to standard Boolean formulas, they do allow for exponentially more succinct encodings of problems in domains such as planning, non-monotonic reasoning, formal verification and the synthesis of computing systems (e.g., Giunchiglia et al., 2009; Shukla et al., 2019). This explains the extensive efforts dedicated to developing QBF solvers in the past few decades (see `http://www.qbflib.org`). QBFs with free variables are also a valuable knowledge representation language for Boolean logic. As such, some specific automated reasoning techniques have been designed for dealing with such formulas (e.g., Klieber et al., 2013; Fargier & Marquis, 2014).

We will next present our treatment of literal quantification in Boolean logic which subsumes variable quantification. The results we shall present on literal quantification immediately translate into results on variable quantification as we can quantify a variable by quantifying its two literals. Our treatment of literal quantification is based on the novel notion of boundary models (Definition 1), which we investigate in the next section before using it to define the semantics of literal quantification in later sections.

3. For instance, the QBF $\forall X \cdot (\exists Y \cdot (X \Leftrightarrow Y))$ is valid: whatever truth value is given to X, one can find a truth value for Y that satisfies $X \Leftrightarrow Y$ (we can give Y the same truth value given to X). Contrastingly, $\exists X \cdot (\forall Y \cdot (X \Leftrightarrow Y))$ is inconsistent: whatever truth value we give to X, there is a truth value for Y that falsifies $X \Leftrightarrow Y$ (we can give Y the value not given to X).



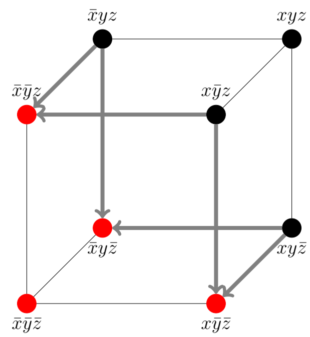


Figure 1: Left: Visualizing the worlds over variables X, Y, Z. Two worlds are connected by an edge iff they disagree on a single variable. Right: Visualizing the models and b-rules of formula $\varphi = (x \vee y) \wedge (x \vee z) \wedge (y \vee z)$. Black nodes are models of φ and red nodes are models of $\neg\varphi$. Each highlighted edge corresponds to a b-rule for φ.

4. Boundary Models and Rules

According to Definition 1, an ℓ-boundary model of formula φ becomes a model for its negation $\neg\varphi$ once we flip its literal ℓ. As such, a model ω of φ can be boundary with respect to multiple literals $\ell \in \omega$. We will next introduce the notion of a *boundary rule* to describe both a model and a particular literal that it is boundary on. We will then show that boundary rules encode certain knowledge that a formula φ has about literals. In fact, we will show that such rules characterize all models of φ and hence they characterize its logical content. In later sections, we will show that quantifying a literal ℓ from a formula φ erases the knowledge that formula φ has about literal ℓ, either by strengthening the formula (universal quantification) or by weakening it (existential quantification).

Definition 5. *Let $\Sigma = \{X_1, \ldots, X_n\}$ be the set of all Boolean variables. A* boundary rule *has the form $\ell_1, \ldots, \ell_{i-1}, \ell_{i+1}, \ldots, \ell_n \rightarrow \ell_i$ where ℓ_i is a literal for variable X_i.*

Boundary rules will be referred to as *b-rules* for short. We will say that a b-rule *infers* the literal appearing in its consequent (ℓ_i). We will also say that a b-rule *uses* the literals appearing in its antecedent ($\ell_1, \ldots, \ell_{i-1}, \ell_{i+1}, \ldots, \ell_n$). The set of legitimate b-rules depends on the set of Boolean variables Σ. In the following examples, and unless stated otherwise, we will assume that Σ is the set of Boolean variables in the formulas under consideration.

Definition 6. *We say formula φ has b-rule $\alpha \rightarrow \ell$ iff α, ℓ is an ℓ-boundary model of φ.*

This definition establishes b-rules as descriptors of boundary models. We will use $R(\varphi)$ to denote the set of b-rules for formula φ. Recall Definition 1 which says that α, ℓ is an ℓ-boundary model of φ precisely when $\alpha, \ell \models \varphi$ and $\alpha, \bar{\ell} \models \neg\varphi$. Consider now the formula $\varphi = (x \vee y) \wedge (x \vee z) \wedge (y \vee z)$ which has four models $M(\varphi) = \{\bar{x}yz, x\bar{y}z, xy\bar{z}, xyz\}$ and six b-rules $R(\varphi) = \{\bar{x}z \rightarrow y, \bar{y}z \rightarrow x, \bar{x}y \rightarrow z, x\bar{y} \rightarrow z, y\bar{z} \rightarrow x, x\bar{z} \rightarrow y\}$. Figure 1 (left) visualizes all eight worlds over variables X, Y, Z using a hypercube. Figure 1 (right) visualizes the models of formula φ and its b-rules. Consider model $\omega = \bar{x}yz$. This is a y-boundary model and a z-boundary model, so it is described by two b-rules: $\bar{x}z \rightarrow y$ and $\bar{x}y \rightarrow z$. The first b-rule tells us that ω will become a model of $\neg\varphi$ if we flip literal y ($\bar{x}\bar{y}z \models \neg\varphi$). The second b-rule tells us that ω will become a model of $\neg\varphi$ if we flip literal z ($\bar{x}y\bar{z} \models \neg\varphi$). Since $yz \rightarrow \bar{x}$ is not a b-rule of formula φ we know that flipping literal $\bar{x}$ will maintain ω as a model ($xyz \models \varphi$). Interestingly, xyz is not a boundary model of formula φ yet we were able to conclude that it is a model of φ using only the b-rules of φ. The next theorem provides a stronger result: one can compute all models of a consistent formula using its b-rules. That is, the b-rules of a consistent formula characterize its logical content.

Theorem 1. *Two consistent formulas φ_1 and φ_2 are equivalent iff they have the same set of b-rules: $M(\varphi_1) = M(\varphi_2)$ iff $R(\varphi_1) = R(\varphi_2)$.*

According to this result, the boundary models of a formula, as described by its b-rules, form a *generating set* from which all models can be constructed. The proof of Theorem 1 provides an explicit construction.

The following proposition shows that b-rules capture certain knowledge about literals.

Proposition 4. *Formula φ has b-rule $\alpha \rightarrow \ell$ iff $\varphi \wedge \alpha$ is consistent and $\varphi \models \alpha \Rightarrow \ell$.*

That is, a b-rule $\alpha \rightarrow \ell$ can be viewed as encoding a feasible scenario (α) under which literal ℓ can be inferred. Formula $\varphi = x \Leftrightarrow y$ has four b-rules $R(\varphi) = \{x \rightarrow y, y \rightarrow x, \bar{x} \rightarrow \bar{y}, \bar{y} \rightarrow \bar{x}\}$ and formula $\phi = (x \vee y) \wedge z$ has five b-rules $R(\phi) = \{\bar{y}z \rightarrow x, \bar{x}z \rightarrow y, xy \rightarrow z, x\bar{y} \rightarrow z, \bar{x}y \rightarrow z\}$. Even though literal z is implied by formula ϕ and literals $\bar{x}\bar{y}$, the formula does not have $\bar{x}\bar{y} \rightarrow z$ as a b-rule since $\bar{x}\bar{y}$ is not consistent with ϕ. Similarly, literal $\bar{z}$ is implied by formula ϕ and literals $\bar{x}\bar{y}$ but the formula does not have $\bar{x}\bar{y} \rightarrow \bar{z}$ as a b-rule.

A formula is independent of a literal (variable) precisely when the formula has no b-rules for that literal (variable).

Proposition 5. *Formula φ is independent of literal ℓ iff it has no b-rules of the form $\alpha \rightarrow \ell$. It is independent of variable X iff it has no b-rules of the form $\alpha \rightarrow x$ or $\alpha \rightarrow \bar{x}$.*

Recall that formula φ is independent of literal ℓ precisely when it can be expressed as an NNF that does not mention literal ℓ. Similarly, formula φ is independent of variable X precisely when it can expressed in a form that does not mention variable X.

While models are compositional, b-rules are not. However, the b-rules of a formula are determined by the b-rules of its negation. This is shown by the next proposition.

Proposition 6. *For formulas φ and ψ, we have:*

(a) $R(\neg\varphi) = \{\alpha \rightarrow \ell \mid \alpha \rightarrow \bar{\ell} \in R(\varphi)\}$.

(b) $R(\varphi) \cap R(\psi) \subseteq R(\varphi \wedge \psi) \subseteq R(\varphi) \cup R(\psi)$.

(c) $R(\varphi) \cap R(\psi) \subseteq R(\varphi \vee \psi) \subseteq R(\varphi) \cup R(\psi)$.

According to this proposition, conjoining or disjoining two formulas φ and ψ preserves only their common b-rules. Moreover, $R(\varphi \wedge \psi)$ and $R(\varphi \vee \psi)$ are not connected. Neither is contained in the other and their intersection may be empty. Consider the formulas $\varphi = x$ and $\psi = y$. We have $R(\varphi \wedge \psi) = \{x \rightarrow y, y \rightarrow x\}$ and $R(\varphi \vee \psi) = \{\bar{x} \rightarrow y, \bar{y} \rightarrow x\}$.

We will conclude this section with a result that relates the number of b-rules for a Boolean formula to the number of its models, counter models and boundary models.

Proposition 7. *For formula φ and $n = |\Sigma|$, we have*

$$|R(\varphi)| \leq n \cdot |BM(\varphi)| \leq n \cdot min(|M(\varphi)|, |M(\overline{\varphi})|).$$

The bound $|R(\varphi)| \leq n \cdot |BM(\varphi)|$ is tight. For instance, $\varphi = x_1 \wedge \ldots \wedge x_n$ has a single boundary model but n b-rules, $R(\varphi) = \{(\bigwedge_{j=1,\ldots,n|j \neq i} x_j) \rightarrow x_i \mid i = 1, \ldots, n\}$. We also remark that $|R(\varphi)|$ and $|BM(\varphi)|$ can be exponentially smaller than $|M(\varphi)|$. Consider the formula $\varphi = x_1 \vee \ldots \vee x_n$ which has $2^n - 1 = |M(\varphi)|$ models. This formula has only n b-rules, $R(\varphi) = \{(\bigwedge_{j=1,\ldots,n|j \neq i} \bar{x}_j) \rightarrow x_i \mid i = 1, \ldots, n\}$, and n boundary models, $BM(\varphi) = \{x_i \wedge \bigwedge_{j=1,\ldots,n|j \neq i} \bar{x}_j \mid i = 1, \ldots, n\}$. Finally, the number of b-rules for a formula can be exponential in the number of its variables. This also holds for the number of its models and the number of its counter models. For instance, the formula $\varphi = \oplus_{i=1}^{n} x_i$ has 2^{n-1} models, 2^{n-1} counter models, and $2^{n-1} \cdot (n-1)$ b-rules.

5. Literal Quantification

We next discuss existential and universal literal quantification. We start by introducing and studying universal literal quantification and then review and study further existential literal quantification. The latter type of quantification was first introduced and studied in (Lang et al., 2003) under the name of *literal forgetting.* Some of the results we shall present on universal literal quantification follow from known results on existential literal quantification due to a duality between the two notions. We will also present new and fundamental results, based on boundary and independent models, which apply to both types of literal quantification, again due to the duality between them. Our main goal of the upcoming study is to develop an intuitive semantics for universal quantification (as a selection process) and then use this semantics to show its central role in explainable AI. Additionally, our treatment will reveal new results on the computation of universal quantification which have complexity implications for explainable AI queries.

5.1 Universal Literal Quantification

Before we define the universal quantification of literal ℓ from a formula φ, we note that φ can be expanded as $\varphi = (\ell \vee (\varphi|\bar{\ell})) \wedge (\bar{\ell} \vee (\varphi|\ell))$, which is equivalent to what is known as Boole's or Shannon's expansion, $\varphi = (\ell \wedge (\varphi|\ell)) \vee (\bar{\ell} \wedge (\varphi|\bar{\ell}))$.

Definition 7. *Universally quantifying literal ℓ from formula φ is defined as follows:*

$$\forall \ell \cdot \varphi \stackrel{def}{=} (\ell \vee (\varphi|\bar{\ell})) \wedge (\varphi|\ell).$$

That is, the operator $\forall \ell$ drops literal $\bar{\ell}$ from the expansion $\varphi = (\ell \vee (\varphi|\bar{\ell})) \wedge (\bar{\ell} \vee (\varphi|\ell))$. Consider the formula $\varphi = (x \Rightarrow y) \wedge (y \Rightarrow x)$ which says that variables X and Y are equivalent. We have $\forall x \cdot \varphi = x \wedge y$ and $\forall \bar{x} \cdot \varphi = \bar{x} \wedge \bar{y}$. Note, however, that $\forall X \cdot \varphi = \bot$. Moreover, $\forall x (\forall \bar{x} \cdot \varphi) = \forall \bar{x} (\forall x \cdot \varphi) = \bot$. Consider now the formula $\phi = (x \Rightarrow y)$. We now have $\forall x \cdot \phi = y$, $\forall \bar{x} \cdot \phi = (x \Rightarrow y)$ and $\forall X \cdot \phi = y$. Moreover, we have $\forall x (\forall \bar{x} \cdot \phi) = \forall \bar{x} (\forall x \cdot \phi) = y$.

Since $(\ell \vee (\varphi|\bar{\ell})) \wedge (\varphi|\ell)$ is equivalent to $(\ell \wedge (\varphi|\ell)) \vee ((\varphi|\bar{\ell}) \wedge (\varphi|\ell))$, and since $(\varphi|\bar{\ell}) \wedge (\varphi|\ell)$ is equivalent to $\forall X \cdot \varphi$, where X is the variable of literal ℓ, we get $\forall \ell \cdot \varphi = (\ell \wedge (\varphi|\ell)) \vee (\forall X \cdot \varphi)$. This shows that $\forall \ell \cdot \varphi$ can be obtained by adding to $\forall X \cdot \varphi$ all models of φ that contain literal ℓ and that are ruled out by variable quantification.

We next provide a number of results on universal literal quantification which have novel counterparts for existential literal quantification that we present in Section 5.2. The first result provides a *semantical characterization* of universal literal quantification based on the notion of boundary models.

Theorem 2. *For formula φ and literal ℓ, we have $M(\forall \ell \cdot \varphi) \subseteq M(\varphi)$. Moreover, $\omega \in M(\varphi)$ and $\omega \notin M(\forall \ell \cdot \varphi)$ iff ω is an $\bar{\ell}$-boundary model of φ.*

That is, universally quantifying literal ℓ from formula φ strengthens the formula by dropping its $\bar{\ell}$-boundary models. These models contain and depend on literal $\bar{\ell}$: they cease to be models of φ if we were to flip literal $\bar{\ell}$. Hence, we can view the operator $\forall \ell$ as a selection operator which picks models of φ that do not depend on literal $\bar{\ell}$. The selected models either contain literal ℓ or an irrelevant literal $\bar{\ell}$. We will later present a more general result that provides selection semantics when universally quantifying multiple literals.

Our second result provides a *syntactic characterization* of universal literal quantification based on the notion of b-rules. In particular, it characterizes what gets added to a formula upon strengthening it by universal quantification.

Theorem 3. *For formula φ and literal ℓ, we have $\forall \ell \cdot \varphi = \varphi \wedge \bigwedge_{\alpha \rightarrow \bar{\ell} \in R(\varphi)} \neg \alpha$.*

That is, the operator $\forall \ell$ adds $\neg \alpha$ to formula φ for each b-rule $\alpha \rightarrow \bar{\ell}$ of the formula. This erases all these b-rules as their antecedents α will no longer be consistent with the quantified formula, therefore erasing the knowledge that formula φ has about literal $\bar{\ell}$. This also makes the quantified formula independent of literal $\bar{\ell}$, as shown by Proposition 5, which leads to the elimination of literal $\bar{\ell}$.

The third result says that the universal quantification of literal ℓ preserves implicants that contain literal ℓ.

Proposition 8. *For formula φ, term $\gamma \models \varphi$ and literal $\ell \in \gamma$, we have $\gamma \models \forall \ell \cdot \varphi$.*

The next result says that universal literal quantification preserves logical implication.

Proposition 9. *For formulas φ, ϕ and literal ℓ, we have $\varphi \models \phi$ only if $\forall \ell \cdot \varphi \models \forall \ell \cdot \phi$.*

The following three results (Propositions 10-12) parallel ones that are known for existential quantification. The first result provides a semantical characterization of universal literal quantification based on the notion of literal independence.

Proposition 10. *$\forall \ell \cdot \varphi$ is the logically weakest formula that is independent of literal $\bar{\ell}$ and that also implies formula φ.*

The second result shows that a set of literals can be universally quantified in any order, therefore justifying the notation $\forall \{\ell_1, \ldots, \ell_n\} \cdot \varphi$.

Proposition 11. *For literals ℓ_1, ℓ_2 and formula φ, we have $\forall \ell_1 (\forall \ell_2 \cdot \varphi) = \forall \ell_2 (\forall \ell_1 \cdot \varphi)$.*

We should note that literals ℓ_1 and ℓ_2 can be for the same variable and hence conflicting.

The third result shows that universal literal quantification is more fine-grained than universal variable quantification as we can universally quantify variable X by universally quantifying literals x and $\bar{x}$ in any order.

Proposition 12. *For variable X and formula φ, we have $\forall X \cdot \varphi = \forall\{x, \bar{x}\} \cdot \varphi$.*

Together with Proposition 11, this result shows that we can universally quantify a set of literals and variables in any order. For example, the following quantifications are all legitimate and equivalent: $\forall x, \bar{x}, Y \cdot \varphi$, $\forall x, Y, \bar{x} \cdot \varphi$ and $\forall Y, X \cdot \varphi$.

We are now ready to present our *selection semantics* for universally quantifying a set of (possibly conflicting) literals. This result will play a major role when discussing the applications of universal quantification to explainable AI in Section 7. It invokes the notion of an α-independent model introduced in Definition 2. This is a model that contains term α and that remains a model if we were to flip any literals of α.

Theorem 4. *Let φ be a formula, $\ell_1, \ldots, \ell_n$ be literals, ω be a world and $\alpha = \omega \cap \{\bar{\ell}_1, \ldots, \bar{\ell}_n\}$. Then $\omega \models \forall \ell_1, \ldots, \ell_n \cdot \varphi$ iff ω is an α-independent model of φ.*

Consider $\varphi = (x \vee y \vee z) \wedge (\bar{x} \vee y \vee t)$, literals x, y, z and world $\omega = \bar{x}\bar{y}zt$, leading to $\alpha = \omega \cap \{\bar{x}, \bar{y}, \bar{z}\} = \bar{x}\bar{y}$. Then ω is an α-independent model of φ since $zt \models \varphi$. Hence, $\omega \models \forall x, y, z \cdot \varphi$ which can be verified since $\forall x, y, z \cdot \varphi = (x \vee y \vee z) \wedge (y \vee t)$. For literals $x, \bar{y}, \bar{z}$, we get $\alpha = \omega \cap \{\bar{x}, y, z\} = \bar{x}z$ so ω is not an α-independent model of φ since $\bar{y}t \not\models \varphi$ (indeed, $\bar{x}\bar{y}\bar{z}t \models \overline{\varphi}$). Hence, $\omega \not\models \forall x, \bar{y}, \bar{z} \cdot \varphi$ which can be verified since $\forall x, \bar{y}, \bar{z} \cdot \varphi = x \wedge t$.

According to Theorem 4, when universally quantifying literals $\ell_1, \ldots, \ell_n$ from formula φ, we are "selecting" all (and only) models of φ that do not depend on literals $\bar{\ell}_1, \ldots, \bar{\ell}_n$. That is, the models of $\forall \ell_1, \ldots, \ell_n \cdot \varphi$ are precisely the models of φ which continue to be models of φ if we were to flip any literals they may have in $\bar{\ell}_1, \ldots, \bar{\ell}_n$.[4] We will revisit Theorem 4 in detail when we discuss explainable AI in Section 7, but we note here that the presence of α-independent models indicates the absence of certain b-rules.[5]

Proposition 13. *Let $\omega = \alpha, \beta$ be a model of formula φ where α and β are disjoint terms. Then ω is an α-independent model of φ iff φ has no b-rules of the form $\beta, \gamma \rightarrow \ell$.*

In Appendix A, we provide a complete characterization of which b-rules are deleted, introduced or preserved when universally quantifying a literal. This allows us to define quantification as a process of b-rule transformation.

5.2 Existential Literal Quantification

We next review and study further the existential quantification of literals which was first introduced and studied in (Lang et al., 2003) under the name of *literal forgetting.* Before we define the existential quantification of literal ℓ from a formula φ, we recall again the well known Boole's or Shannon's expansion, $\varphi = (\ell \wedge (\varphi|\ell)) \vee (\bar{\ell} \wedge (\varphi|\bar{\ell}))$.

Definition 8. *Existentially quantifying literal ℓ from formula φ is defined as follows:*

$$\exists \ell \cdot \varphi \quad \stackrel{def}{=} \quad (\varphi|\ell) \vee (\bar{\ell} \wedge (\varphi|\bar{\ell})).$$

4. Each model of $\forall \ell_1, \ldots, \ell_n \cdot \varphi$ is also an α-independent model of $\forall \ell_1, \ldots, \ell_n \cdot \varphi$; see Lemma 9.
5. Let us say that term γ is an α-independent implicant of formula φ iff $\gamma \models \varphi$, $\alpha \subseteq \gamma$ and $\gamma \setminus \alpha \models \varphi$. Then Theorem 4 will hold if we replace "world" by "term" and "model" by "implicant."

That is, the operator $\exists \ell$ drops literal ℓ from the expansion $\varphi = (\ell \wedge (\varphi|\ell)) \vee (\bar{\ell} \wedge (\varphi|\bar{\ell}))$.

Consider the formula $\varphi = (x \Rightarrow y) \wedge (y \Rightarrow x)$ which says that variables X and Y are equivalent. We have $\exists x \cdot \varphi = (x \Rightarrow y) = (\bar{x} \vee y)$ which is independent of literal x. We also have $\exists \bar{x} \cdot \varphi = (y \Rightarrow x) = (\bar{y} \vee x)$ which is independent of literal $\bar{x}$. Note, however, that $\exists X \cdot \varphi = \top$. We finally have $\exists x(\exists \bar{x} \cdot \varphi) = \exists \bar{x}(\exists x \cdot \varphi) = \top$.

Existential and universal literal quantification are related by the following duality.

Theorem 5. *For literal ℓ and formula φ, $\exists \ell \cdot \varphi = \neg(\forall \ell \cdot \neg\varphi)$ and $\forall \ell \cdot \varphi = \neg(\exists \ell \cdot \neg\varphi)$.*

This is symmetric to the variable quantification duality: $\exists X \cdot \varphi = \neg(\forall X \cdot \neg\varphi)$ and $\forall X \cdot \varphi = \neg(\exists X \cdot \neg\varphi)$. In both cases, pushing a negation through a quantifier flips that quantifier.

The next three results come from (Lang et al., 2003). The first result provides a semantical characterization of existential literal quantification using the notion of independence.

Proposition 14. *$\exists \ell \cdot \varphi$ is the logically strongest formula that is independent of literal ℓ and that is implied by formula φ.*

Similar to existential variable quantification, existentially quantifying literal ℓ removes information from formula φ but it only removes information that depends on literal ℓ. That is, every consequence of φ is also a consequence of $\exists \ell \cdot \varphi$ as long as that consequence does not depend on literal ℓ. Literal quantification provides a more fine-grained notion of forgetting, which enables more refined treatments. This is particularly the case when managing inconsistent information as we can now forget less information from one formula to make it consistent with another. Literal forgetting was employed in extended logic programs (Wang, Sattar, & Su, 2005) and disjunctive logic programs (Eiter & Wang, 2006, 2008). It was also used to define more refined update operators using the "forget-then-conjoin" scheme that we discussed earlier (Herzig, Lang, & Marquis, 2013).

The second results from (Lang et al., 2003) shows that the order in which literals are existentially quantified does not matter, which justifies the notation $\exists\{\ell_1, \ldots, \ell_n\} \cdot \varphi$.

Proposition 15. *For literals ℓ_1, ℓ_2 and formula φ, we have $\exists \ell_1(\exists \ell_2 \cdot \varphi) = \exists \ell_2(\exists \ell_1 \cdot \varphi)$.*

The third result shows that existential literal quantification is more primitive than existential variable quantification as we can existentially quantify a variable X by existentially quantifying its two literals x and $\bar{x}$ in any order.

Proposition 16. *For variable X and formula φ, we have $\exists X \cdot \varphi = \exists\{x, \bar{x}\} \cdot \varphi$.*

Together with Proposition 15, this result shows that we can existentially quantify a set of literals and variables in any order.

The next set of results are new and follow from results we presented for universal literal quantification using the duality between existential and universal literal quantification. The first result provides a *semantical characterization* of existential literal quantification.

Theorem 6. *For formula φ and literal ℓ, we have $M(\varphi) \subseteq M(\exists \ell \cdot \varphi)$. Moreover, $\omega \in M(\exists \ell \cdot \varphi)$ and $\omega \notin M(\varphi)$ iff ω is an $\bar{\ell}$-boundary model of $\neg\varphi$.*

That is, existentially quantifying literal ℓ from formula φ adds the $\bar{\ell}$-boundary models of $\neg\varphi$. These models must contain literal $\bar{\ell}$ so no model that contains literal ℓ is added.

The second result provides a *syntactic characterization* of existential literal quantification using the notion of b-rules.

Theorem 7. *For formula φ and literal ℓ, we have $\exists \ell \cdot \varphi = \varphi \vee \bigvee_{\alpha \rightarrow \ell \in R(\varphi)} \alpha$.*

That is, existentially quantifying literal ℓ from formula φ amounts to disjoining φ with the antecedent α of each of its b-rules $\alpha \rightarrow \ell$. Given b-rule $\alpha \rightarrow \ell$, world α, ℓ is a model of φ but world $\alpha, \bar{\ell}$ is not a model. Disjoining with the antecedent α adds $\alpha, \bar{\ell}$ as a model. This erases b-rules $\alpha \rightarrow \ell$ and the knowledge that φ has about literal ℓ. This also makes the quantified formula independent of literal ℓ, therefore eliminating this literal.

The third result says that existentially quantifying literal ℓ preserves implicates that contain literal $\neg\ell$.

Proposition 17. *If formula φ implies clause β and literal $\bar{\ell} \in \beta$, then $\exists \ell \cdot \varphi \models \beta$.*

The last result says that existential literal quantification preserves logical implication.

Proposition 18. *For formulas φ, ϕ and literal ℓ, we have $\varphi \models \phi$ only if $\exists \ell \cdot \varphi \models \exists \ell \cdot \phi$.*

One can use the duality theorem to state further results such as the parallel of Theorem 4. We will refrain from discussing these additional results though as our goal is to focus more on universal literal quantification and its applications to explainable AI.

5.3 Is it Quantification or Elimination?

As mentioned earlier, variable quantification in Boolean logic can be viewed as an elimination process that transforms a formula so it becomes independent of the quantified variable. In particular, quantifying a variable leads to a new formula that can be expressed without mentioning the variable. In fact, George Boole used the term "elimination" but the term "quantification" is now more commonly used. Moreover, "variable elimination" has become a synonym for "existential variable quantification" in some parts of the literature, including many works on satisfiability where existential variable quantification is used as a preprocessing technique (e.g., Eén & Biere, 2005; Subbarayan & Pradhan, 2005). The distinction between elimination and quantification becomes more relevant though when treating literals instead of variables as we show next. Let us consider variables first. $\exists X.\varphi$ eliminates variable X by weakening formula φ: It *adds* world ω as a model iff $\omega[\ell]$ is a model of φ for *some* literal ℓ of variable X. $\forall X.\varphi$ eliminates variable X by strengthening formula φ: It *keeps* world ω as a model iff $\omega[\ell]$ is a model of φ for *all* literals ℓ of X. The fundamental operation here is one of elimination, which can be achieved in a unique way (up to logical equivalence) through either weakening (adding models) or strengthening (dropping models). Moreover, the conditions for deciding which models to add or drop are based on the two states of variable X, which are considered either existentially or universally. Eliminating literals is also done by weakening or strengthening a formula in a unique way, but the conditions for deciding which models to add or drop consider only one state of the variable. In particular, $\exists \ell.\varphi$ eliminates literal ℓ by adding world ω as a model iff $\omega[\ell]$ is a model of φ. Moreover, $\forall \ell.\varphi$ eliminates literal $\bar{\ell}$ by keeping world ω as a model iff $\omega[\ell]$ is a model of φ. In a sense, the term elimination is perhaps more appropriate than quantification in this case, and the symbols $\exists$ and $\forall$ are more indicative of elimination by weakening and strengthening than anything else. One could have adopted different symbols and terminology for literals, but we opted to keep the same ones used for variables as this emphasizes the symmetries between variable and literal elimination, as revealed by the many results we discussed earlier.

6. Tractable Literal Quantification

We identify in this section some classes of Boolean formulas and circuits that allow one to quantify literals efficiently. We consider CNFs, DNFs and some of their subsets. We also consider two circuit types: Decision-DNNFs (Huang & Darwiche, 2007) and SDDs (Darwiche, 2011), which are strict supersets of OBDDs (Bryant, 1986). All of these circuit types are subsets of NNF circuits (Darwiche & Marquis, 2002).

We start with two results which provide the foundation of further results. The first shows how to quantify literals out of constants and literals. The second identifies conditions under which literal quantification can be distributed through conjunctions and disjunctions. When these conditions are met, quantification can be done in linear time on NNF formulas and circuits. The second result corresponds to a well-known result for variable quantification which we show does extend to the more fine-grained notion of literal quantification.

Proposition 19. *For literal ℓ, we have $\exists \ell \cdot \top = \forall \ell \cdot \top = \top$ and $\exists \ell \cdot \bot = \forall \ell \cdot \bot = \bot$. Moreover, for literals ℓ_1 and ℓ_2, we have:*

$$\exists \ell_1 \cdot \ell_2 = \begin{cases} \top, & \text{if } \ell_1 = \ell_2 \\ \ell_2, & \text{otherwise.} \end{cases} \qquad \text{and} \qquad \forall \ell_1 \cdot \ell_2 = \begin{cases} \bot, & \text{if } \ell_1 = \bar{\ell}_2 \\ \ell_2, & \text{otherwise.} \end{cases}$$

Proposition 20. *Consider literal ℓ and formulas α and β. We then have*

(a) $\exists \ell \cdot (\alpha \vee \beta) = (\exists \ell \cdot \alpha) \vee (\exists \ell \cdot \beta)$

(b) $\forall \ell \cdot (\alpha \wedge \beta) = (\forall \ell \cdot \alpha) \wedge (\forall \ell \cdot \beta)$

Moreover, if the variable of literal ℓ is not shared between α and β, then

(c) $\exists \ell \cdot (\alpha \wedge \beta) = (\exists \ell \cdot \alpha) \wedge (\exists \ell \cdot \beta)$

(d) $\forall \ell \cdot (\alpha \vee \beta) = (\forall \ell \cdot \alpha) \vee (\forall \ell \cdot \beta)$

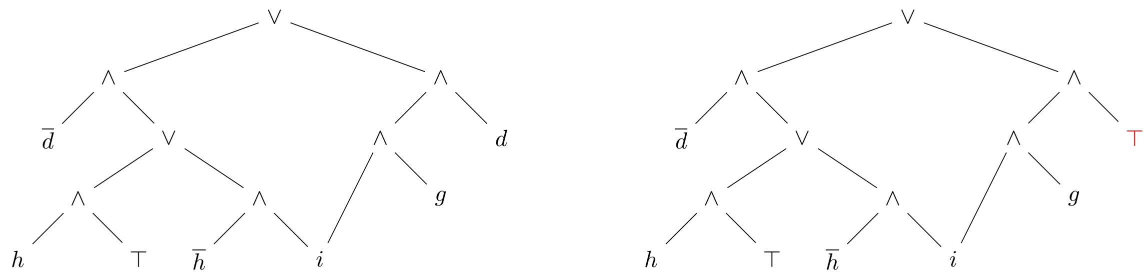
This proposition holds also for variable quantification as it can be implemented through successive literal quantification as shown by Propositions 12 and 16.

These propositions imply direct methods for quantifying literals out of clauses and terms which can be useful when working with CNFs and DNFs. Existentially quantifying literal ℓ from a clause leads to $\top$ if literal ℓ appears in the clause, otherwise the clause is left intact. Universally quantifying literal ℓ drops literal $\bar{\ell}$ from the clause. Existentially quantifying a literal ℓ from a term drops literal ℓ from the term. Universally quantifying literal ℓ leads to $\bot$ if $\bar{\ell}$ appears in the term, otherwise the term is left intact.

We now consider some classes of Boolean formulas and circuits that allow one to quantify literals efficiently. We start by considering CNF, DNF and some of their subsets. Recall that we do not allow trivial clauses or trivial terms that include complementary literals. Hence, clauses cannot be valid and terms cannot be inconsistent.

We first consider CNFs while noting that the procedures discussed below guarantee that the result of quantifying literals is also a CNF.

Proposition 21 (CNF)**.** *Consider a CNF $\Delta = \alpha_1 \wedge \ldots \wedge \alpha_n$. We can obtain $\forall \ell \cdot \Delta$ by removing literal $\bar{\ell}$ from every clause α_i in Δ. Moreover, if Δ is closed under resolution on the variable of literal ℓ, then we can obtain $\exists \ell \cdot \Delta$ by removing from Δ every clause α_i that contains literal ℓ.*



(a) Decision-DNNF Δ (b) $\exists d \cdot \Delta$

Figure 2: Existentially quantifying literal d from Decision-DNNF Δ. This amounts to replacing every occurrence of literal d in the Decision-DNNF by $\top$. The result is a DNNF.

Corollary 1.[6] *Consider a CNF $\Delta = \alpha_1 \wedge \ldots \wedge \alpha_n$ where each clause α_i is a prime implicate of Δ. We can obtain $\exists \ell \cdot \Delta$ by removing from Δ clauses that contain literal ℓ.*

We next consider DNFs. The procedures discussed below also guarantee that the result of quantifying literals is a DNF.

Proposition 22 (DNF)**.** *Consider a DNF $\Delta = \alpha_1 \vee \ldots \vee \alpha_n$. We can obtain $\exists \ell \cdot \Delta$ by removing literal ℓ from every term α_i in Δ. Moreover, if Δ is closed under consensus on the variable of literal ℓ, then we can obtain $\forall \ell \cdot \Delta$ by removing from Δ every term α_i that contains literal $\overline{\ell}$.*

Corollary 2. *Consider a DNF $\Delta = \alpha_1 \vee \ldots \vee \alpha_n$ where each term α_i is a prime implicant of Δ. We can obtain $\forall \ell \cdot \Delta$ by removing from Δ terms that contain literal $\overline{\ell}$.*

To summarize, the following quantifications can be performed in linear time: universal literal quantification on CNF, universal literal quantification on prime implicants, existential literal quantification on DNF and existential literal quantification on prime implicates. These results parallel ones that are well-known for variable quantification. To be more precise, the elimination of universal variable quantifiers from a CNF is a special case of the so-called universal reduction rule for QBFs (Kleine Büning et al., 1995). Moreover, the elimination of existential variable quantifiers from a DNF is a special case of the so-called existential reduction rule for QBFs. Both rules are used in search-based QBF solvers. Similarly, the use of consensus and resolution to eliminate universal and existential quantifiers in CNFs and DNFs, respectively, has been known for a while.

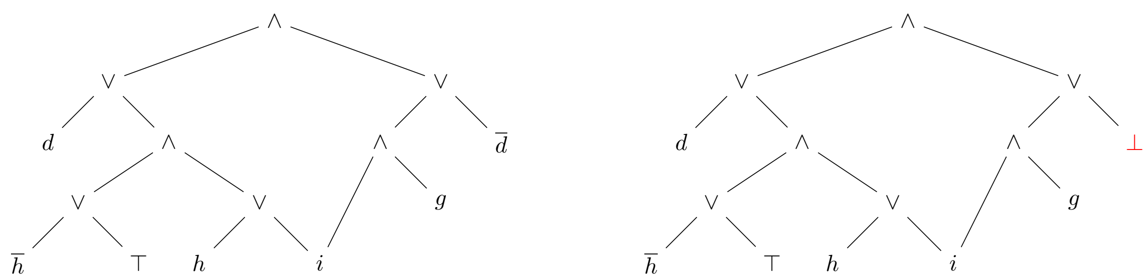
We next show that existential and universal literal quantification can be performed in linear time on Decision-DNNF circuits (Huang & Darwiche, 2007). These are NNF circuits which satisfy two properties: *decision* and *decomposability.* The decision property says that every or-node has the form $(\ell \wedge \alpha) \vee (\overline{\ell} \wedge \beta)$, where ℓ is a literal. The decomposability property says that for every and-node, the sets of variables of its children $\alpha_1, \ldots, \alpha_n$ are pairwise disjoint. Figure 2a depicts an example Decision-DNNF Δ for the CNF $(h \vee i) \wedge (\overline{d} \vee g) \wedge (\overline{d} \vee i)$ over variables D, G, H, I. Decision-DNNF circuits are a strict superset of OBDD circuits.

6. This corollary slightly extends Proposition 19 in (Lang et al., 2003), which focuses on the case when Δ is given by the set of all its prime implicates.

We first discuss existential literal quantification which yields DNNF circuits (Darwiche, 2001). These are NNF circuits that satisfy only the decomposability property.

Proposition 23 ($\exists$, Decision-DNNF)**.** *Literals can be existentially quantified from a Decision-DNNF circuit in time linear in the circuit size while yielding a DNNF circuit.*

The quantification algorithm follows directly from Propositions 19 and 20(a,c) since existential quantification distributes through both the conjunctions and disjunctions of a Decision-DNNF. Figure 2b depicts the DNNF circuit which results from existentially quantifying literal d from the Decision-DNNF Δ of Figure 2a.



(a) NNF circuit Γ (b) $\forall d \cdot \Gamma$

Figure 3: Universally quantifying literal d from the NNF circuit Γ, which is obtained from the Decision-DNNF Δ of Figure 2a using Proposition 24. The quantification process amounts to replacing every occurrence of literal $\bar{d}$ in Γ with $\bot$. Proposition 24 guarantees that the two circuits Δ and Γ are equivalent.

We next show a similar result for universally quantifying literals in linear time, except that the output is only guaranteed to be an NNF circuit in this case (i.e., not necessarily decomposable). This is a two-step procedure, where each step requires linear time processing. The first step is described by the following proposition. It transforms the Decision-DNNF into a form that allows us to invoke Proposition 20(d) so we can distribute universal quantification through disjunctions (this quantification always distributes through conjunctions).

Proposition 24. *Let Δ be a Decision-DNNF circuit and let Γ be the result of replacing every fragment $(\ell \wedge \alpha) \vee (\bar{\ell} \wedge \beta)$ in Δ with $(\ell \vee \beta) \wedge (\bar{\ell} \vee \alpha)$. Then (1) Γ is an NNF circuit that is equivalent to Δ; (2) Γ can be obtained from Δ in time linear in the size of Δ; and (3) for every disjunction $\alpha \vee \beta$ in Γ, the disjuncts α and β do not share variables.*

Figure 3a depicts an example of this first step. It shows an NNF circuit Γ which is obtained by transforming the Decision-DNNF Δ of Figure 2a. The resulting circuit Γ is not a Decision-DNNF, yet no variables are shared between disjuncts in this circuit.

The second step directly applies Propositions 19 and 20(b,d) to the result of the first step, now that disjuncts no longer share variables.

Proposition 25 ($\forall$, Decision-DNNF)**.** *Literals can be universally quantified from a Decision-DNNF circuit in time linear in the circuit size while yielding an NNF circuit.*

Figure 3b depicts an example of this second step. It shows the result of universally quantifying literal d from the NNF circuit Γ of Figure 3a. Given Propositions 19 and 20(b,d), we obtain $\forall d \cdot \Gamma$ by simply replacing every occurrence of literal $\bar{d}$ in Γ with $\bot$ as shown in the figure. We finally note that $\forall d \cdot \Gamma = i \wedge g$ in this case.

The two-step, linear-time procedure suggested by Propositions 24 and 25 was actually used implicitly in (Darwiche & Hirth, 2020) for explaining the decisions made by Boolean classifiers on instances—a process which corresponds to universally quantifying all literals in the instance (see Section 7). The procedure proposed in (Darwiche & Hirth, 2020) transformed fragments $(\ell \wedge \alpha) \vee (\bar{\ell} \wedge \beta)$ in the Decision-DNNF into $(\ell \wedge \alpha) \vee (\bar{\ell} \wedge \beta) \vee (\alpha \wedge \beta)$, calling this a *consensus* operation as it resembles the consensus operation on DNF. It then transformed this further into $(\ell \wedge \alpha) \vee (\alpha \wedge \beta)$ when literal ℓ appeared in the instance, calling this a *filtering* operation. This is equivalent to $(\ell \vee \beta) \wedge \alpha$, which is the result of universally quantifying literal ℓ from fragment $(\ell \vee \beta) \wedge (\bar{\ell} \vee \alpha) = (\ell \wedge \alpha) \vee (\bar{\ell} \wedge \beta)$. These transformations correspond to the two-step procedure described above except that (Darwiche & Hirth, 2020) did not realize that their algorithm was actually universally quantifying literals as such quantification was not introduced yet. But (Darwiche & Hirth, 2020) did observe that their procedure yields a monotone circuit, a property which is guaranteed to hold when universally quantifying a literal for each variable.[7]

We next show that similar linear time algorithms can be used on SDD circuits (Darwiche, 2011). These are NNF circuits which are composed of fragments having the form $(p_1 \wedge s_1) \vee \ldots \vee (p_n \wedge s_n)$, where p_i are called *primes* and s_i are called *subs.* SDDs satisfy the decomposability property. Moreover, the primes $p_1, \ldots, p_n$ form a partition: $p_i \neq \bot$, $p_i \wedge p_j = \bot$ for $i \neq j$ and $p_1 \vee \ldots \vee p_n = \top$. SDDs actually satisfy a stronger version of decomposability, called *structured decomposability* (Pipatsrisawat & Darwiche, 2008), but we do not need this stronger property for the following results. Like Decision-DNNFs, SDDs are a strict superset of OBDDs. We note, however, that Decision-DNNF and SDD circuits are not comparable in terms of succinctness; that is, neither is strictly more succinct than the other (Bollig & Buttkus, 2019; Beame & Liew, 2015).

Proposition 26 ($\exists$, SDD)**.** *One can existentially quantify a set of literals from an SDD circuit in time linear in the circuit size, with the result being a DNNF circuit.*

Again, the algorithm follows directly from Propositions 19 and 20(a,c) since existential quantification can be distributed through both conjunctions and disjunctions in this case.

Universally quantifying literals from an SDD circuit is also based on a two-step procedure, where each step requires linear time processing. Similar to Decision-DNNF circuits, the first step transforms the SDD into an equivalent NNF circuit in which disjuncts do not share variables, therefore making it directly amenable to Propositions 19 and 20(b,d).

Proposition 27. *Let Δ be an SDD circuit and Γ be the result of replacing every fragment $(p_1 \wedge s_1) \vee \ldots \vee (p_n \wedge s_n)$ in Δ with $(\neg p_1 \vee s_1) \wedge \ldots \wedge (\neg p_n \vee s_n)$. Then (1) Γ is an NNF circuit that is equivalent to Δ; (2) Γ can be obtained from Δ in time linear in the size of Δ; and (3) disjuncts in Γ do not share variables.*

Proposition 28 ($\forall$, SDD)**.** *One can universally quantify a set of literals from an SDD circuit in time linear in the circuit size, with the result being an NNF circuit.*

7. The circuit is monotone since for every variable X, the literals x and $\bar{x}$ cannot both appear in the circuit.

In summary, the existential and universal quantification of multiple literals can be performed in linear time on Decision-DNNF and SDD circuits. Existential literal quantification yields DNNF circuits, while universal literal quantification yields NNF circuits that may not be decomposable. Hence, the above procedures on Decision-DNNF and SDD circuits cannot be used to interleave literal quantifications of different types. This should not be surprising though since the existence of a polynomial-time algorithm for applying a sequence of quantifiers of different types to a tractable circuit (even an OBDD) would imply $\mathsf{P} = \mathsf{NP}$; see (Coste-Marquis et al., 2006). Nonetheless, it is worth noting that for both types of literal quantification, the NNF circuit that results from applying the above procedures is monotone, if we quantify a literal for each variable that appears in the input circuit.[8] Hence, one can test efficiently whether the resulting circuit is satisfiable, or whether it is valid, which does not hold for either CNF or DNF unless $\mathsf{P} = \mathsf{NP}$ (deciding the satisfiability of a CNF is NP-complete and deciding the validity of a DNF is coNP-complete).

7. Universal Quantification for Explainable AI

We now consider a number of questions that arise in explainable AI and show how they can be answered using universal literal and variable quantification. These questions include (1) finding the *culprit* behind a decision (a minimal set of characteristics that can trigger the decision); (2) assessing whether a decision is *biased* (depends on protected features); and (3) identifying instances with some *irrelevant* features or characteristics (do not play a role in the decisions made on these instances). Some of these questions have been treated in the literature (e.g., Shih et al., 2018; Ignatiev et al., 2019a, 2019b, 2019c; Darwiche & Hirth, 2020; Audemard et al., 2020), but we provide new formulations based on quantification which allow a more refined and general treatment. Moreover, given the results in Section 6, our treatment will shed light on the syntactic forms of classifiers that facilitate the computation of explainable AI queries.

7.1 Classifiers and Decisions

Our focus is on Boolean classifiers, which correspond to Boolean functions $f(X_1, \ldots, X_n)$ that map literals $\ell_1, \ldots, \ell_n$ of variables $X_1, \ldots, X_n$ into $\{0, 1\}$. A set of literals $\ell_1, \ldots, \ell_n$ will be called an *instance*, which is *positive* when $f(\ell_1, \ldots, \ell_n) = 1$ and *negative* when $f(\ell_1, \ldots, \ell_n) = 0$. A term γ over some variables in $X_1, \ldots, X_n$ will be called a *population* as it characterizes a set of instances (those compatible with term γ). An instance corresponds to a singleton population, so claims about populations apply to instances but the converse is not true. For example, a classifier will always make a decision on an instance but it may not be able to make a collective decision on a population as the population may contain both positive and negative instances.

We will represent a Boolean classifier by a Boolean formula Δ, where the models of Δ correspond to positive instances and the models of $\neg\Delta$ correspond to negative instances. Hence, the syntactic forms of both Δ and its negation $\neg\Delta$ are relevant when computing explainable AI queries.

8. This follows because Proposition 19 tells us that $\exists \ell \cdot \ell = \top$ and $\exists \ell \cdot \bar{\ell} = \bar{\ell}$; similarly $\forall \ell \cdot \ell = \ell$ and $\forall \ell \cdot \bar{\ell} = \bot$. Hence, the quantified NNF circuit will not contain complementary literals for any variable.

Definition 9. *Let Δ be a Boolean formula over variables $X_1, \ldots, X_n$. We call Δ a* classifier*, variable X_i a* feature*, literal ℓ_i a* characteristic*, and $\delta = \{\ell_1, \ldots, \ell_n\}$ an* instance*. The* decision *of classifier Δ on instance δ is denoted $\Delta(\delta)$. It is defined as $\Delta(\delta) = 1$ if $\delta \models \Delta$ (positive decision) and $\Delta(\delta) = 0$ if $\delta \models \neg\Delta$ (negative decision). We further define $\Delta_\delta = \Delta$ when the instance δ is positive and $\Delta_\delta = \neg\Delta$ when the instance δ is negative.*

Since Δ captures positive instances and its negation $\neg\Delta$ captures negative instances, we usually work with Δ when reasoning about positive decisions and with $\neg\Delta$ when reasoning about negative decisions. This explains the significance of the notation Δ_δ.

Consider the following classifier which decides whether an applicant should be granted a loan based on four features: whether they defaulted on a previous loan (D), have a guarantor (G), own a home (H) or have a high income (I). This classifier is specified by the following formula Δ and its negation $\neg\Delta$, both in CNF:

$$\Delta = (h \vee i) \wedge (\bar{d} \vee g) \wedge (\bar{d} \vee i) \text{ and } \neg\Delta = (d \vee \bar{i}) \wedge (d \vee \bar{h}) \wedge (\bar{g} \vee \bar{i}). \tag{1}$$

Classifier Δ will grant a loan to an applicant who never defaulted on a previous loan, owns a home, has a high income but does not have a guarantor, $\Delta(\bar{d}, \bar{g}, h, i) = 1$. But it will not grant a loan to such an applicant if they defaulted on a previous loan, $\Delta(d, \bar{g}, h, i) = 0$. Instance $\delta_1 = \bar{d}, \bar{g}, h, i$ is positive and instance $\delta_2 = d, \bar{g}, h, i$ is negative. Using our notation, we have $\Delta_{\delta_1} = \Delta$ since $\delta_1 \models \Delta$ and $\Delta_{\delta_2} = \neg\Delta$ since $\delta_2 \models \neg\Delta$.

The notion of a decision in Definition 9 can be extended to populations.

Definition 10. *A population γ is* decided *by a classifier Δ iff $\gamma \models \Delta$ or $\gamma \models \neg\Delta$. In the first case, the decision is positive and we write $\Delta(\gamma) = 1$. In the second case, the decision is negative and we write $\Delta(\gamma) = 0$. Otherwise, the decision $\Delta(\gamma)$ is undefined.*

Contrary to instances, it is possible that no decision is made on a population γ as we may have neither $\gamma \models \Delta$ nor $\gamma \models \neg\Delta$. Consider the population of applicants who have guarantors and a high income, $\gamma_1 = g, i$. All members of this population will be granted loans since $\gamma_1 \models \Delta$, and hence $\Delta(\gamma_1) = 1$, making this a positive population. The population of applicants who defaulted on a previous loan and do not have a high income, $\gamma_2 = d, \bar{i}$, is negative. No member of this population will be granted a loan since $\gamma_2 \models \neg\Delta$ and hence $\Delta(\gamma_2) = 0$. Consider now the population of applicants who defaulted on a loan and are not home owners, $\gamma_3 = d, \bar{h}$. This population cannot be decided as it contains some members who will be granted loans (e.g., $d, g, \bar{h}, i$) and others who will not (e.g., $d, g, \bar{h}, \bar{i}$). We then have $\gamma_3 \not\models \Delta$ and $\gamma_3 \not\models \neg\Delta$, causing the decision $\Delta(\gamma_3)$ to be undefined. This will never happen for an instance δ as we must have either $\delta \models \Delta$ or $\delta \models \neg\Delta$ since the instance δ must contain a characteristic for each feature.

We will find it useful to talk about containment when analyzing the decisions made on populations. We will say that population γ *contains* population β iff $\beta \models \gamma$. For example, the population of home owners ($\gamma = h$) contains the population of home owners with a high income ($\beta = h, i$). We say in this case that γ is a *super-population* of β and β is a *sub-population* of γ. For example, when explaining the decision on an instance δ, one is typically interested in finding all maximal super-populations of instance δ that are decided similarly as δ. As we show later, this is precisely the approach proposed in (Shih et al., 2018), which we will generalize to explain decisions on populations as well.

7.2 Decision Making and Universal Quantification

Before we start discussing explainable AI queries in Section 7.3, we will first provide some insights on the fundamental role that universal literal quantification plays when reasoning about decisions. We will interpret literal quantification as *characteristic quantification* and show how such quantification *selects* instances in ways that can be useful for answering various queries of interest to explainable AI.

When universally quantifying characteristic ℓ from classifier Δ, we are *filtering out* all positive instances for which characteristic $\bar{\ell}$ is essential for the decisions on these instances. We call these positive *$\bar{\ell}$-boundary instances* as they are positive instances with characteristic $\bar{\ell}$ but will become negative if this characteristic is flipped (Theorem 2). Universally quantifying characteristic ℓ will not filter out any instance with characteristic ℓ (Proposition 8). We can therefore view the application of operator $\forall\ell$ to Δ as a process of *selecting* all positive instances that do not require characteristic $\bar{\ell}$ for their positiveness. If any of these instances has characteristic $\bar{\ell}$ then that characteristic is irrelevant to the decision made on the instance. These are precisely the instances characterized by $\forall\ell \cdot \Delta$. A complementary situation arises when universally quantifying characteristic ℓ from $\neg\Delta$. The instances characterized by $\forall\ell \cdot \neg\Delta$ are precisely the negative instances characterized by $\neg\Delta$ which do not require characteristic $\bar{\ell}$ for their negativeness.

Consider again the classifier defined by Equation 1. The positive instance $\delta_1 = \bar{d}, \bar{g}, h, i$ is not a model of $\forall d \cdot \Delta$ (which is equivalent to $i \wedge g$ as discussed in Section 6). Thus, characteristic $\bar{d}$ of δ_1 is essential for the positiveness of δ_1 (instance $d, \bar{g}, h, i$ is negative). Contrastingly, the positive instance $\delta_2 = \bar{d}, g, \bar{h}, i$ is a model of $\forall d \cdot \Delta$ so characteristic $\bar{d}$ is not essential for the positiveness of δ_2 (instance $d, g, \bar{h}, i$ is also positive).

We now turn to interpreting b-rules as descriptors of *boundary instances* that get filtered out by universal quantification. Each b-rule $\alpha \rightarrow \ell$ for classifier Δ identifies a positive instance $\delta = \alpha, \ell$ which becomes negative if we flip its characteristic ℓ (Definition 6). That is, b-rules $\alpha \rightarrow \ell$ for Δ identify positive ℓ-boundary instances. Similarly, b-rules $\alpha \rightarrow \ell$ for $\neg\Delta$ identify negative ℓ-boundary instances. Hence, b-rules characterize instances that are selected (or filtered out) when universally quantifying a characteristic.

Suppose now that we are universally quantifying a set of characteristics $\ell_1, \ldots, \ell_n$ from Δ to yield $\forall\ell_1, \ldots, \ell_n \cdot \Delta$. In this case, $\forall\ell_1, \ldots, \ell_n \cdot \Delta$ captures all positive instances δ where the characteristics they have in $\bar{\ell}_1, \ldots, \bar{\ell}_n$ are irrelevant to how these instances are decided. That is, we can flip any of the characteristics in $\delta \cap \{\bar{\ell}_1, \ldots, \bar{\ell}_n\}$ without changing the decision (Theorem 4). As we shall see later, when characteristics $\ell_1, \ldots, \ell_n$ define a positive instance, $\forall\ell_1, \ldots, \ell_n \cdot \Delta$ will characterize all super-populations of this instance that are decided similarly to the instance. Moreover, these super-populations can be viewed as explanations of the decision made on instance $\ell_1, \ldots, \ell_n$. We will also see how selecting instances based on quantifying both characteristics and features can be used to answer further queries such as those relating to decision and classifier bias.

7.3 Irrelevant Features

The first application we shall consider for universal quantification concerns the selection of instances which can be decided independently of a given set of features. For example, we may wish to identify all individuals who will be granted a loan regardless of their income and home ownership. These features may be relevant to some applicants but not others. Our interest is therefore in identifying all instances that can be decided independently of some given features while capturing these instances using a Boolean formula. We next show how universal quantification can be used to achieve this.

Definition 11. *To* erase *variables $X_1, \ldots, X_n$ from term γ is to remove the literals of variables $X_1, \ldots, X_n$ from term γ. The resulting term is denoted $\gamma_{\uparrow X_1,\ldots,X_n}$.*[9]

For example, if $\gamma = x\bar{y}\bar{z}w$ then $\gamma_{\uparrow Z,W} = x\bar{y}$. When term γ represents an instance or a population, the erase operator creates a super-population of γ.

Definition 12. *Let γ be a population decided by classifier Δ. We say that decision $\Delta(\gamma)$ is* independent *of features $X_1, \ldots, X_n$ precisely when $\Delta(\gamma) = \Delta(\gamma_{\uparrow X_1,\ldots,X_n})$.*

We also say in this case that features $X_1, \ldots, X_n$ are *irrelevant* to the decision $\Delta(\gamma)$.

Consider again the loan classifier in (1) and the population of home owners who have a high income but never defaulted on previous loan, $\gamma = \bar{d}, h, i$. This is a positive population, $\Delta(\gamma) = 1$, as all its members will be granted loans. The decision on this population is independent of feature I though since $\Delta(\gamma_{\uparrow I}) = 1$ where $\gamma_{\uparrow I} = \bar{d}h$.

If our goal is to check whether a decision $\Delta(\gamma)$ is independent of features $X_1, \ldots, X_n$ then we can simply check whether $\Delta(\gamma) = \Delta(\gamma_{\uparrow X_1,\ldots,X_n})$. But universal quantification can be used to characterize all instances that are decided independently of some features.

Theorem 8. *Let Δ be a classifier and δ be an instance. Decision $\Delta(\delta)$ is independent of features $X_1, \ldots, X_n$ iff $\delta \models \forall X_1, \ldots, X_n \cdot \; \Delta_\delta$.*

According to this result, $\forall X_1, \ldots, X_n \cdot \Delta$ characterizes all instances that will be decided positively independently of features $X_1, \ldots, X_n$ and $\forall X_1, \ldots, X_n \cdot \neg\Delta$ characterizes all instances that will be decided negatively independently of these features. For example, the following Boolean formula characterizes all applicants who will be granted a loan independently of whether they own a home or have defaulted on a previous loan:

$$\forall D, H \cdot \Delta = g \wedge i, \text{ where } \Delta = (h \vee i) \wedge (\bar{d} \vee g) \wedge (\bar{d} \vee i).$$

This formula captures applicants who have a guarantor and a high income. Each member of this population will be granted a loan regardless of their features D and H. The expression $\forall D, H \cdot \Delta$ can be easily evaluated since Δ is given as a CNF: we just remove literals d, $\bar{d}$, h and $\bar{h}$ from every clause of the CNF (see Proposition 12 and 21).

It is possible that $\forall X_1, \ldots, X_n \cdot \; \Delta = \bot$, which means that every positive decision must depend on features $X_1, \ldots, X_n$. Similarly, it is possible that $\forall X_1, \ldots, X_n \cdot \neg\Delta = \bot$, which means that every negative decision must depend on these features. Moreover, it is possible that one decision type is independent of some features but the other is not. This is illustrated by the following example:

$$\begin{array}{ll} \forall D, G \cdot \Delta = \bot, & \text{where } \Delta = (h \vee i) \wedge (\bar{d} \vee g) \wedge (\bar{d} \vee i) \\ \forall D, G \cdot \neg\Delta = \bar{h} \wedge \bar{i}, & \text{where } \neg\Delta = (d \vee \bar{i}) \wedge (d \vee \bar{h}) \wedge (\bar{g} \vee \bar{i}). \end{array}$$

9. We can also define this operator using variable quantification since $\gamma_{\uparrow X_1,\ldots,X_n} = \exists X_1, \ldots, X_n \cdot \gamma$. We use Definition 11 instead as it is more direct.

No applicant will be granted a loan without considering whether they defaulted on a previous loan and whether they have a guarantor (features D and G). However, some applicants will be denied a loan without considering these features. In particular, any applicant who does not own a home and does not have a high income will be declined.

In contrast, we may have $\forall X_1, \ldots, X_n \cdot \Delta = \Delta$ indicating that every positive decision is independent of features $X_1, \ldots, X_n$. This is equivalent to $\forall X_1, \ldots, X_n \cdot \neg\Delta = \neg\Delta$, which means that all negative decisions will also be independent of these features.[10] The loan classifier depends on all its features.

7.4 Irrelevant Characteristics

We may also be interested in instances whose classification does not depend on some characteristics (in contrast to features). For example, we may be interested in applicants who will be granted a loan but not due to their high income or home ownership. These applicants may not have any of these characteristics but if they do then these characteristics are irrelevant to how their application is decided. We next show how universal literal quantification can be used to select instances with irrelevant characteristics and then further contrast irrelevant characteristics with irrelevant features.

Definition 13. *Let γ be a population decided by classifier Δ and α be a set of characteristics. We say that decision $\Delta(\gamma)$ is* independent *of characteristics α precisely when $\Delta(\gamma) = \Delta(\gamma \setminus \alpha)$.*

We also say in this case that characteristics α are *irrelevant* to decision $\Delta(\gamma)$. This definition does not require every characteristic of α to appear in population γ. Moreover, a characteristic ℓ and its negation $\bar{\ell}$ may both appear in α.

Consider again the loan classifier and an applicant who defaulted on a previous loan, owns a home, has a guarantor but does not have a high income, $\delta = d, h, g, \bar{i}$. This applicant will be denied a loan, $\Delta(\delta) = 0$, but the decision is independent of characteristics $\bar{d}$ and h since $\Delta(\delta \setminus \{\bar{d}, h\}) = 0$. The decision is independent of characteristic $\bar{d}$ since the applicant did default on a previous loan. It is independent of characteristic h since the applicant will still be denied a loan if they did not own a home, $\Delta(d, \bar{h}, g, \bar{i}) = 0$. Note, however, that this decision is not independent of features D and H. For example, the applicant will be granted a loan if they did not default on a previous loan, $\Delta(\bar{d}, h, g, \bar{i}) = 1$.

Theorem 9. *Let Δ be a classifier, $\ell_1, \ldots, \ell_n$ be characteristics and δ be an instance. Then $\delta \models \forall \ell_1, \ldots, \ell_n \cdot \Delta_\delta$ iff decision $\Delta(\delta)$ is independent of characteristics $\bar{\ell}_1, \ldots, \bar{\ell}_n$.*

According to this result, $\forall \ell_1, \ldots, \ell_n \cdot \Delta$ characterizes all instances that are decided positively but not due to any characteristic they may have in $\bar{\ell}_1, \ldots, \bar{\ell}_n$. An instance captured by $\forall \ell_1, \ldots, \ell_n \cdot \Delta$ may not have any characteristic in $\bar{\ell}_1, \ldots, \bar{\ell}_n$. But if it does, then those characteristics are irrelevant: we can change them in any manner without changing the decision. Similarly, $\forall \ell_1, \ldots, \ell_n \cdot \neg\Delta$ captures all instances that are decided negatively but not due to any characteristic they may have in $\bar{\ell}_1, \ldots, \bar{\ell}_n$.

10. To show this, observe that $\forall X_1, \ldots, X_n \cdot \Delta = \Delta$ is equivalent to Δ being independent of $X_1, \ldots, X_n$. This is equivalent to $\neg\Delta$ being independent of $X_1, \ldots, X_n$, which is equivalent to $\forall X_1, \ldots, X_n \cdot \neg\Delta = \neg\Delta$.

For the loan classifier, there are four applicants who will be denied a loan independently of characteristics d and h. These applicants are characterized by the following CNF:

$$\forall \bar{d}, \bar{h} \cdot \neg\Delta = \bar{h} \wedge \bar{i}, \text{ where } \neg\Delta = (d \vee \bar{i}) \wedge (d \vee \bar{h}) \wedge (\bar{g} \vee \bar{i}).$$

None of these applicants owns a home. Moreover, if any of them defaulted on a previous loan they will still be denied if they did not default. But there are no applicants who will be denied a loan independently of features D and H:

$$\forall D, H \cdot \neg\Delta = \bot.$$

That is, these features are relevant for every negative decision.

Again, expressions $\forall \bar{d}, \bar{h} \cdot \neg\Delta$ and $\forall D, H \cdot \neg\Delta$ can be easily evaluated using Propositions 12 and 21 since $\neg\Delta$ is given as a CNF. For the first expression, we just remove literals d and h from all clauses. For the second expression, we remove literals d, $\bar{d}$, h and $\bar{h}$.

It is possible that $\forall \ell_1, \ldots, \ell_n \cdot \Delta = \bot$. This indicates that every positive instance with some characteristics in $\bar{\ell}_1, \ldots, \bar{\ell}_n$ can become negative if we flip some of these characteristics so these characteristics are relevant for all positive instances. Similarly, if $\forall \ell_1, \ldots, \ell_n \cdot \neg\Delta = \bot$, then any negative instance with some characteristics in $\bar{\ell}_1, \ldots, \bar{\ell}_n$ may become positive if we flip some of these characteristics. A set of characteristics may be relevant for one type of decision but irrelevant for the other type since $\forall \ell_1, \ldots, \ell_n \cdot \Delta$ may be inconsistent but $\forall \ell_1, \ldots, \ell_n \cdot \neg\Delta$ may be consistent (and vice versa).

The following result relates irrelevant characteristics and features. It shows that if some features are irrelevant to a decision then any corresponding characteristics are also irrelevant to the decision. The converse is not true though as we have seen earlier.

Proposition 29. *Let Δ be a classifier, $X_1, \ldots, X_n$ be features and $\ell_1, \ldots, \ell_n$ be corresponding characteristics. Then $\forall X_1, \ldots, X_n \cdot \Delta \models \forall \ell_1, \ldots, \ell_n \cdot \Delta$.*

7.5 Explaining Decisions

We will now consider the application of universal literal and variable quantification to explaining the decisions of classifiers on both instances and populations.

Consider a decision $\Delta(\delta)$ made by classifier Δ on instance δ. One way to explain this decision is to identify a minimal set of characteristics $\gamma \subseteq \delta$ that is sufficient to trigger the decision, $\Delta(\gamma) = \Delta(\delta)$. This notion of explanation was introduced in (Shih et al., 2018) under the name of a *PI-explanation* and was later called a *sufficient reason* for the decision in (Darwiche & Hirth, 2020). We will next provide some examples of this notion using a classifier from (Darwiche & Hirth, 2020) for admitting students into an academic program.

This classifier makes its decision based on five features: whether an applicant passed the entrance exam (E), is a first time applicant (F), has good grades (G), has work experience (W) and comes from a rich hometown (R). The classifier is specified by the following CNFs:

$$\begin{aligned} \Delta &= (e \vee g) \wedge (e \vee r) \wedge (e \vee w) \wedge (f \vee r) \wedge (\bar{f} \vee g \vee w) \\ \neg\Delta &= (\bar{e} \vee f \vee \bar{r}) \wedge (\bar{e} \vee \bar{f} \vee \bar{g}) \wedge (\bar{e} \vee \bar{f} \vee \bar{w}) \wedge (\bar{g} \vee \bar{r} \vee \bar{w}) \end{aligned}$$

Consider now an applicant who does not come from a rich hometown but satisfies all other requirements. This applicant will be admitted, $\Delta(e, f, g, w, \bar{r}) = 1$, and there are two sufficient reasons for this decision, (e, f, g) and (e, f, w). Passing the entrance exam (e), being a first time applicant (f) and having good grades (g) will guarantee admission but no strict subset of these characteristics will. Having work experience instead of good grades will also guarantee admission but again no strict subset of characteristics (e, f, w) provides such a guarantee. If this applicant were to come from a rich hometown, then there would be more sufficient reasons for the admission decision, $\Delta(e, f, g, w, r) = 1$:

$$(e, f, g) \quad (e, f, w) \quad (e, g, r) \quad (e, r, w) \quad (g, r, w)$$

For example, passing the entrance exam and having good grades will guarantee admission for an applicant who comes from a rich hometown.

A decision may have an exponential number of sufficient reasons. Moreover, some explainable AI queries, such as ones relating to decision bias, are based on checking whether the sufficient reasons for a decision satisfy some properties. These sufficient reasons can sometimes be represented compactly using the notion of a *complete reason* introduced in (Darwiche & Hirth, 2020), which showed a number of results relating to this notion. These results include (1) The complete reason for a decision can be computed efficiently when the classifier is represented using a tractable circuit of appropriate type; (2) The sufficient reasons for a decision correspond to the prime implicants of its complete reason; and (3) Some properties of sufficient reasons can be checked in time linear in the size of a complete reason, again, when it is represented using a tractable circuit of appropriate type.

We will next show how the notion of a complete reason can be formulated using universal quantification, particularly the selection semantics of such quantification. This formulation has a number of implications, which include generalizing this notion to decisions on populations and opening new pathways for the efficient computation of complete reasons.

The main insight behind our formulation is to try to find a necessary and sufficient condition for why a decision was made. Consider a decision on population $\gamma = \ell_1, \ldots, \ell_m$ and let $X_{m+1}, \ldots, X_n$ be all features not mentioned in population γ. We will do this by finding all instances that are decided similarly to population γ but based only on the information used to decide γ; that is, characteristics $\ell_1, \ldots, \ell_m$. We will first select instances that are decided similarly to γ but independently of features $X_{m+1}, \ldots, X_n$ as these features did not play a role in the decision on population γ. These instances are characterized by the formula $\forall X_{m+1}, \ldots, X_n \cdot \Delta_\gamma$ as shown in Section 7.3. From these instances, we will now select those that are decided similarly to γ but independently of characteristics $\bar{\ell}_1, \ldots, \bar{\ell}_m$ as these characteristics did not play a role in the decision either. As shown in Section 7.4, these instances are characterized by the formula $\forall \ell_1, \ldots, \ell_m(\forall X_{m+1}, \ldots, X_n \cdot \Delta_\gamma)$. This formula can be thought of as a necessary and sufficient reason for the decision on population γ so we shall call it the complete reason for the decision (more on this later).

Definition 14 (Complete Reason)**.** *Let $\gamma = \ell_1, \ldots, \ell_m$ be a population decided by classifier Δ and $X_{m+1}, \ldots, X_n$ be all classifier features not mentioned in γ. The* complete reason *for decision $\Delta(\gamma)$ is defined as the formula $\forall \ell_1, \ldots, \ell_m, X_{m+1}, \ldots, X_n \cdot \Delta_\gamma$.*

Definition 15 (Sufficient Reason)**.** *The* sufficient reasons *for a decision are defined as the prime implicants of its complete reason.*

We now have the following result, which establishes our definition of complete reason as a generalization of the one given in (Darwiche & Hirth, 2020) and our definition of a sufficient reason as a generalization of the PI-explanation introduced in (Shih et al., 2018).

Theorem 10. *Let γ be a population decided by classifier Δ. Then $\gamma^\star$ is a sufficient reason for decision $\Delta(\gamma)$ iff $\gamma^\star$ is a minimal subset of γ that satisfies $\Delta(\gamma^\star) = \Delta(\gamma)$.*

According to this result, the sufficient reasons for decision $\Delta(\gamma)$ are the maximal super-populations of γ that are decided similarly to population γ. Moreover, these super-populations are precisely the prime implicants of the complete reason for the decision. As mentioned earlier, we may have an exponential number of such super-populations but they are now encoded by the complete reason which may not be exponentially sized (depending on its syntactic form). When the population is a singleton (instance), the complete reason for decision $\Delta(\gamma)$ reduces to $\forall \ell_1, \ldots, \ell_m \cdot \Delta_\gamma$ which provides an alternative definition to the one given in (Darwiche & Hirth, 2020). This is an expression that we studied in Section 7.4, except that we now have the condition $\ell_1, \ldots, \ell_m \models \Delta_\gamma$ which we did not assume in that section. This additional condition provides further selection semantics for universal literal quantification: the complete reason $\forall \ell_1, \ldots, \ell_m \cdot \Delta_\gamma$ characterizes all instances δ that are decided similarly to γ but due only to the characteristics they have in common with γ; that is, independently of their characteristics $\delta \cap \{\bar{\ell}_1, \ldots, \bar{\ell}_m\}$.

Consider the admission classifier and an applicant who passed the entrance exam, has good grades and work experience, comes from a rich hometown but is not a first time applicant. The classifier will admit this applicant, $\Delta(e, \bar{f}, g, r, w) = 1$, due to the following complete reason expressed as a CNF:

$$\forall e, \bar{f}, g, r, w \cdot \Delta = (e \vee g) \wedge (e \vee w) \wedge (r) \wedge (\bar{f} \vee g \vee w)$$

This formula has four prime implicants representing the sufficient reasons for this decision:

$$(e, g, r) \quad (e, r, w) \quad (e, \bar{f}, r) \quad (g, r, w)$$

Consider now the population of applicants who are applying again but did not pass the entrance exam and do not come from a rich hometown. Members of this population will be denied admission, $\Delta(\bar{e}, \bar{f}, \bar{r}) = 0$, for the following complete reason

$$\forall \bar{e}, \bar{f}, \bar{r}, G, W \cdot \neg\Delta = (\bar{e} \vee \bar{f}) \wedge (\bar{r})$$

which has two sufficient reasons $(\bar{e}, \bar{r})$ and $(\bar{f}, \bar{r})$.

When a classifier Δ and its negation $\neg\Delta$ are represented by CNFs, the complete reason for a decision $\Delta(\gamma)$ can be computed in linear time using Propositions 12 and 21: We just drop every characteristic from the CNF Δ_γ if that characteristic does not appear in population (term) γ. As a result, the complete reason will be a *monotone* CNF, allowing one to enumerate sufficient reasons using a quasi-polynomial time algorithm (Gurvich & Khachiyan, 1999).[11] This also provides another characterization of the complete reason for

11. (Gurvich & Khachiyan, 1999) presented a quasi-polynomial time algorithm for the incremental enumeration of the prime implicants of a monotone CNF formula. This algorithm is based on the algorithm for the dualization problem (i.e., testing the duality of a pair of monotone DNF formulas) reported in (Fredman & Khachiyan, 1996). These dualization algorithms have been implemented (Khachiyan et al., 2006), evaluated (Hagen et al., 2009), and improved recently (Sedaghat et al., 2018).

decision $\Delta(\gamma)$ as the weakest CNF Γ that contains only characteristics that appears in γ and that satisfies $\gamma \models \Gamma \models \Delta_\gamma$. In this sense, the complete reason is the most general abstraction of population γ that justifies the decision made on γ. Every aspect (set of characteristics) of γ that can trigger decision $\Delta(\gamma)$ is an implicant of the complete reason Γ. Moreover, every prime implicant of Γ is an aspect of γ. This justifies viewing the complete reason for a decision as a necessary and sufficient condition for explaining the decision.

7.6 Decision Bias

We next show how universal quantification can be used to characterize biased decisions. Following (Darwiche & Hirth, 2020), these are decisions that are made based on features some of which are designated as *protected.* A classifier is biased precisely when it makes at least one biased decision. Hence, a biased classifier may still make some unbiased decisions.

Definition 16. *Let Δ be a classifier where some of its features are designated as* protected. *A decision $\Delta(\delta)$ on instance δ is* biased *iff $\Delta(\delta) \neq \Delta(\delta^\star)$ for some instance $\delta^\star$ obtained from δ by changing only the values of some protected features.*

We next show how variable quantification can be used to characterize all biased decisions that a classifier may make.

Theorem 11. *Let Δ be a classifier and let $X_1, \ldots, X_n$ be its protected features. Then $\Delta \wedge \neg(\forall X_1, \ldots, X_n \cdot \Delta)$ characterizes all positive instances on which Δ will make a biased decision. Moreover, $\neg\Delta \wedge \neg(\forall X_1, \ldots, X_n \cdot \neg\Delta)$ characterizes all negative instances on which Δ will make a biased decision.*

(Darwiche & Hirth, 2020) provided an efficient procedure for deciding whether a decision is biased, assuming the classifier is represented using an appropriate tractable circuit. The above theorem suggests an efficient procedure for detecting decision bias assuming classifier Δ and its negation $\neg\Delta$ are in CNF. To check whether a biased, positive decision is made on instance δ, we just need to check whether $\delta \models \Delta \wedge \neg(\forall X_1, \ldots, X_n \cdot \Delta)$. Since Δ is a CNF, $\forall X_1, \ldots, X_n \cdot \Delta$ can be computed in linear time so the previous test can be performed efficiently. A similar procedure can be used to detect biased negative decisions.

Our treatment of decision bias extends the one in (Darwiche & Hirth, 2020) not only computationally, but also in terms of scope. Instead of only testing whether a particular decision is biased, we can now characterize all biased decisions which allows us to entertain further questions relating to bias. Consider again the admission classifier and suppose that feature R (rich hometown) is protected. The following expression will then characterize all applicants on whom a biased, positive decision will be made:

$$\Delta \wedge \neg(\forall R \cdot \Delta) = (e \vee g) \wedge (e \vee w) \wedge (r) \wedge (\bar{f} \vee g) \wedge (\bar{f} \vee w) \wedge (\bar{e} \vee \bar{f})$$

There are six classes of applicants that satisfy the above formula. All will be admitted but they will be denied admission if they were not to come from a rich hometown. We can now find out if any of these applicants could have failed the entrance exam by computing:

$$\bar{e} \wedge (\Delta \wedge \neg(\forall R \cdot \Delta)) = \bar{e} \wedge g \wedge r \wedge w$$

The answer is affirmative. Moreover, the above result tells us that admitted applicants who fail the entrance exam and whose admission depends critically on coming from a rich hometown must have good grades and a work experience.

8. Concluding Remarks

We formalized and studied the universal quantification of literals in Boolean logic, together with its applications to explainable AI. Our treatment was based on the novel notion of boundary models, which stands to have implications on the study of Boolean logic beyond quantification. A major contribution of our work is the interpretation of universal quantification as a selection process, which we hope will expand the applications of this form of quantification in AI and beyond. Another major contribution is the complexity results relating to the computation of existential and universal quantification on various logical forms. While we were driven by understanding universal literal quantification, our findings have furthered our understanding of Boolean logic quantification more broadly.

As to explainable AI, we have shown how to analyze classifiers and their decisions through the systematic construction of Boolean formulas, using universal literal and variable quantification. We provided some prototypical queries in earlier sections but one can go further beyond them. In a sense, the combination of universal quantifiers with classical Boolean connectives provides a query language for interrogating classifiers and for gathering various insights about how they make decisions and why they make these decisions.

Our treatment of literal quantification can be extended to propositional formulas over discrete variables (Miller & Thornton, 2008), allowing one to reason about the behavior of discrete classifiers which arise in a number of contexts including decision trees and random forests (e.g., Ignatiev et al., 2019c; Audemard et al., 2020; Choi et al., 2020). Consider a propositional formula φ over discrete variables and let X be a variable that has values $x_1, \ldots, x_n$. We can generalize Definitions 7 and 8 to quantify a literal x_i as follows:[12]

$$\begin{aligned}
\forall x_i \cdot \varphi &= (\varphi|x_i) \wedge \bigwedge_{j \neq i} (x_i \vee (\varphi|x_j)) \\
\exists x_i \cdot \varphi &= (\varphi|x_i) \vee \bigvee_{j \neq i} (x_j \wedge (\varphi|x_j)).
\end{aligned}$$

These quantifiers are also dual and have selection and forgetting semantics as in the Boolean setting, therefore expanding the utility of our treatment beyond Boolean logic.

We close this section by a remark on further connections of our work to QBFs. Let φ be a Boolean formula and $Ł_1, \ldots, Ł_n$ be a partition of all literals. One can define another class of prenex and closed QBFs, $Q_1 Ł_1, \ldots, Q_n Ł_n \cdot \varphi$, where $Q_1, \ldots, Q_n$ are alternating quantifiers in $\{\forall, \exists\}$. These QBFs are also guaranteed to be either valid or inconsistent. It remains to be explored though whether these QBFs will also admit the notion of a solution that is normally defined for classical, prenex and closed QBFs. This is a subject for future work.

Acknowledgments

We thank the anonymous reviewers for their valuable feedback. This work has benefited from the support of the International Research Project MAKC ("Modern Approaches to

12. The operator $\exists x_i$ drops literal x_i from the expansion $\varphi = \bigvee_j (x_j \wedge (\varphi|x_j))$. The formula φ can also be expanded as $\varphi = \bigwedge_j (\bigvee_{k \neq j} x_k \vee (\varphi|x_j))$; see the proof of Proposition 27. The operator $\forall x_i$ drops all literals but x_i from this expansion.

Knowledge Compilation") shared between the Automated Reasoning Group of the University of California at Los Angeles (UCLA) and the Centre de Recherche en Informatique de Lens (CRIL UMR 8188 CNRS - Artois University). The work has been partially supported by NSF grant IIS-1910317, DARPA grant N66001-17-2-4032, AI Chair EXPE**KC**TATION (ANR-19-CHIA-0005-01) of the French National Research Agency (ANR) and by TAILOR, a project funded by EU Horizon 2020 research and innovation programme under GA No 952215.

Appendix A. Characterizing the Dynamics of Boundary Rules

We provide in this appendix a complete characterization of which b-rules are deleted, introduced or preserved when universally quantifying a literal. This allows us to define universal quantification as a process of b-rule transformation (one can use the duality theorem to prove similar results for existential literal quantification). These results are mostly meant for completeness as they are not essential for the main storyline in the paper.

To simplify notation, we will use the symbol _ in b-rules to denote any term such that the result is a b-rule according to Definition 6. The operator $\forall \ell_i$ erases every b-rule of the form $\_ \rightarrow \bar{\ell}_i$. It also preserves all b-rules of the form $\_ \rightarrow \ell_i$ or $\_, \ell_i \rightarrow \_$. Finally, it preserves b-rules of the form $\_, \bar{\ell}_i \rightarrow \ell_j$ as long as there are no b-rules of the form $\_, \ell_j \rightarrow \bar{\ell}_j$. We present this result formally next and then interpret it using boundary models.

Theorem 12. *For formula φ and literals ℓ_i and ℓ_j where $i \neq j$, we have*

(a) If $\alpha \rightarrow \ell_i \in R(\varphi)$, then $\alpha \rightarrow \ell_i \in R(\forall \ell_i \cdot \varphi)$.

(b) $\alpha \rightarrow \bar{\ell}_i \notin R(\forall \ell_i \cdot \varphi)$.

(c) If $\alpha, \ell_i \rightarrow \ell_j \in R(\varphi)$, then $\alpha, \ell_i \rightarrow \ell_j \in R(\forall \ell_i \cdot \varphi)$.

(d) If $\alpha, \bar{\ell}_i \rightarrow \ell_j \in R(\varphi)$, then $\alpha, \bar{\ell}_i \rightarrow \ell_j \in R(\forall \ell_i \cdot \varphi)$ iff $\alpha, \ell_j \rightarrow \bar{\ell}_i \notin R(\varphi)$.

According to this result, (a) ℓ_i-boundary models of φ are ℓ_i-boundary models of $\forall \ell_i \cdot \varphi$, (b) $\forall \ell_i \cdot \varphi$ does not have any $\bar{\ell}_i$-boundary models and (c,d) ℓ_j-boundary models of φ are ℓ_j-boundary models of $\forall \ell_i \cdot \varphi$ iff they are not $\bar{\ell}_i$-boundary models of φ.

We now turn to the introduction of new b-rules which leads to introducing boundary models. The following result says that the operator $\forall \ell_i$ can only introduce b-rules of the form $\_, \bar{\ell}_i \rightarrow \_$ so it will only introduce ℓ_j-boundary models that contain literal $\bar{\ell}_i$ ($i \neq j$).

Theorem 13. *If $r \notin R(\varphi)$ and $r \in R(\forall \ell_i \cdot \varphi)$, then b-rule r has the form $\alpha, \bar{\ell}_i \rightarrow \ell_j$.*

The next result provides a complete characterization for when universal literal quantification will introduce new b-rules (of the above form).

Theorem 14. *Let b-rule $r = \alpha, \bar{\ell}_i \rightarrow \ell_j$. We then have $r \notin R(\varphi)$ and $r \in R(\forall \ell_i \cdot \varphi)$ iff $\alpha, \ell_i \rightarrow \ell_j \in R(\varphi)$, $\alpha, \bar{\ell}_j \rightarrow \bar{\ell}_i \in R(\varphi)$, $\alpha, \ell_j \rightarrow \ell_i \notin R(\varphi)$ and $\alpha, \bar{\ell}_i \rightarrow \bar{\ell}_j \notin R(\varphi)$.*

The proof of this theorem provides a characterization for when a model of φ that is not ℓ_j-boundary will become an ℓ_j-boundary model of $\forall \ell_i \cdot \varphi$: Flipping either literal $\bar{\ell}_i$ or literal ℓ_j will preserve it as a model of φ but flipping both will not.

Appendix B. Lemmas

The following lemmas are used in proofs of propositions and theorems. The first lemma says that an ℓ-boundary model for a formula is preserved when dropping other models of the formula (but some non-boundary models may become boundary).

Lemma 1. *Let φ and ϕ be formulas and ω be a world. If $\omega \models \varphi$ and $\varphi \models \phi$ and ω is an ℓ-boundary model for ϕ, then ω is also an ℓ-boundary model for φ.*

Proof. Suppose $\omega \models \varphi$ and $\varphi \models \phi$ and ω is an l-boundary model for ϕ. Then ω has the form α, ℓ where $\alpha, \bar{\ell} \models \neg\phi$. Then $\phi \models \alpha \Rightarrow \ell$ and hence $\varphi \models \alpha \Rightarrow \ell$ and $\varphi \wedge \alpha \wedge \bar{\ell}$ is inconsistent. Therefore $\alpha, \bar{\ell}$ is not a model of φ and hence $\omega = \alpha, \ell$ is an l-boundary model for φ. □

This lemma says that universally quantifying a literal from a term keeps it intact if the negation of that literal does not appear in the term.

Lemma 2. *Let γ be a term and ℓ be a literal such that $\bar{\ell} \notin \gamma$. Then $\forall \ell \cdot \gamma = \gamma$.*

Proof. By Defintion 7, $\forall \ell \cdot \gamma = (\ell \vee (\gamma|\bar{\ell})) \wedge (\gamma|\ell)$. If $\ell \notin \gamma$, $(\ell \vee (\gamma|\bar{\ell})) \wedge (\gamma|\ell) = (\ell \vee \gamma) \wedge \gamma = \gamma$. If $\ell \in \gamma$, $(\ell \vee (\gamma|\bar{\ell})) \wedge (\gamma|\ell) = (\ell \vee \bot) \wedge (\gamma \setminus \{\ell\}) = \gamma$. Hence, $\forall \ell \cdot \gamma = \gamma$. □

This lemma provides another syntactic characterization of universal literal quantification.

Lemma 3. *For variable X and formula φ, we have $\forall x \cdot \varphi = (\forall X \cdot \varphi) \vee (x \wedge \varphi)$ and $\forall \bar{x} \cdot \varphi = (\forall X \cdot \varphi) \vee (\bar{x} \wedge \varphi)$.*

Proof. $\forall x \cdot \varphi = (x \vee (\varphi|\bar{x})) \wedge (\varphi|x) = (x \wedge (\varphi|x)) \vee ((\varphi|\bar{x}) \wedge (\varphi|x)) = (x \wedge \varphi) \vee (\forall X \cdot \varphi)$. We can similarly show $\forall \bar{x} \cdot \varphi = (\forall X \cdot \varphi) \vee (\bar{x} \wedge \varphi)$. □

This lemma says that the α-independent models of a formula are preserved when adding more models to the formula.

Lemma 4. *Let φ and ϕ be formulas and ω be a world such that $\omega \models \phi \models \varphi$. If ω is an α-independent model of ϕ, then ω is also an α-independent model of φ.*

Proof. Suppose $\omega \models \phi \models \varphi$ and ω is an α-independent model of ϕ. Then $\alpha \subseteq \omega$ and $\omega \setminus \alpha \models \phi$. Hence, $\omega \setminus \alpha \models \varphi$ which establishes ω as an α-independent model of φ. □

Appendix C. Proofs

The next three results are folklore. We provide proof sketches for the sake of completeness.

Proof of Proposition 1. This result has been known. See for example Theorem 6.1 in (Katsuno & Mendelzon, 1989) and Corollary 6 in (Lang et al., 2003). Since $\exists X \cdot \varphi = (\varphi|x) \vee (\varphi|\bar{x})$, we get that $\exists X \cdot \varphi$ is independent of X since literals x and $\bar{x}$ do not occur in $(\varphi|x) \vee (\varphi|\bar{x})$. Suppose now that there exists a formula ψ such that $\varphi \models \psi$, $\exists X \cdot \varphi \not\models \psi$ and variable X does not occur in ψ. We will next show a contradiction. Since any ψ can be put into an equivalent CNF, we can assume without loss of generality that ψ is a clause. Since $\varphi \models \psi$, every model of φ must satisfy a literal ℓ of ψ. Since $\exists X \cdot \varphi \not\models \psi$, there must be a model ω of $\exists X \cdot \varphi$ that does not satisfy any literal of ψ. The models of $\exists X \cdot \varphi$ are the models of φ plus all those worlds that differ from a model of φ on X only. Since ω cannot be a model of φ, there must be a model ω' of φ such that ω and ω' coincide on every variable but X. Since ψ does not contain any occurrence of X, the variable of ℓ is different from X, and as a consequence, since ω' is a model of φ, ω satisfies φ as well. This is a contradiction. □

Proof of Proposition 2. Follows directly from Propositions 1 and 3. □

Proof of Proposition 3. Follows from Definitions 3 and 4 using De Morgan's law. □

Proof of Theorem 1.

($\Rightarrow$) Suppose $M(\varphi_1) = M(\varphi_2)$. Then $R(\varphi_1) = R(\varphi_2)$ by Definition 6 (and Definition 1).

($\Leftarrow$) It suffices to show that for a consistent formula φ, its models $M(\varphi)$ are fully characterized by its b-rules $R(\varphi)$. We will show this by defining an operator $\mathcal{L}$ that depends only on b-rules $R(\varphi)$ and then show that $M(\varphi)$ is the stationary point of the sequence $(\mathcal{L}^i(BM(\varphi)))_{i\in\mathbb{N}}$, where $\mathcal{L}^0(W) = W$ and $\mathcal{L}^{i+1}(W) = \mathcal{L}(\mathcal{L}^i(W))$. Note that the boundary models of φ also depend only on b-rules $R(\varphi)$ since $BM(\varphi) = \{\alpha, \ell \mid \alpha \to \ell \in R(\varphi)\}$. The operator $\mathcal{L} : \mathcal{S} \mapsto \mathcal{S}$ is defined as follows, where $\mathcal{S}$ consists of all sets of worlds which contain boundary models $BM(\varphi)$:

$$\begin{aligned} \mathcal{S} &= \{W \mid BM(\varphi) \subseteq W \subseteq 2^\Sigma\} \\ \mathcal{L}(W) &= W \cup \{\alpha, \bar{\ell} \mid \alpha, \ell \in W \text{ and } \alpha \to \ell \notin R(\varphi)\} \end{aligned}$$

The operator $\mathcal{L}$ grows the set of worlds W as follows. For each world $\alpha, \ell \in W$, it adds world $\alpha, \bar{\ell}$ in case $\alpha \to \ell \notin R(\varphi)$. This condition is equivalent to: $\alpha, \ell \in M(\varphi)$ only if $\alpha, \bar{\ell} \in M(\varphi)$. If we apply the operator $\mathcal{L}$ to the boundary models $BM(\varphi)$, it will infer additional models of φ and add them to W. Applying $\mathcal{L}$ again to the result will infer/add more models and so on. This is precisely what the sequence $(\mathcal{L}^i(BM(\varphi)))_{i\in\mathbb{N}}$ does. It suffices now to show that $M(\varphi)$ is the least fixed point of operator $\mathcal{L}$. We need a few lemmas for this, the first shows that $\mathcal{L}$ must have a least fixed point.

The pair $(\mathcal{S}, \subseteq)$ forms a complete lattice with $BM(\varphi)$ as the least element and 2^Σ as the greatest element. If $\mathcal{L}$ is monotonic, then by the Knaster–Tarski Theorem (Knaster, 1928; Tarski, 1955), the set of fixed points of $\mathcal{L}$ is not empty and also forms a complete lattice so it does have a least element.

Lemma 5. *$\mathcal{L} : \mathcal{S} \mapsto \mathcal{S}$ is monotonic: $W \subseteq W'$ only if $\mathcal{L}(W) \subseteq \mathcal{L}(W')$ for every $W, W' \in \mathcal{S}$.*

Proof. Suppose $W \subseteq W'$. If $\omega \in \mathcal{L}(W)$, then by definition of $\mathcal{L}$ we have either (1) $\omega \in W$ or (2) $\omega = \alpha, \bar{\ell}$ where $\alpha, \ell \in W$ and $\alpha \to \ell \notin R(\varphi)$. Case (1) implies $\omega \in W'$ and hence $\omega \in \mathcal{L}(W')$. Case (2) implies $\alpha, \ell \in W'$ and hence $\omega = \alpha, \bar{\ell} \in \mathcal{L}(W')$ since $\alpha \to \ell \notin R(\varphi)$. □

The next two lemmas use the notion of a *Hamming path (H-path).* An H-path from world ω_1 to world ω_d is a sequence of worlds $\omega_1, \ldots, \omega_d$ where world ω_i, $i > 1$, is obtained from world ω_{i-1} by flipping the value of a single variable. This H-path has length d.

Lemma 6. *If $\omega_1, \ldots, \omega_d$ is a shortest H-path from $BM(\varphi)$ to $\omega_d \in M(\varphi)$, then $\omega_i \in M(\varphi)$ for $i = 1, \ldots, d$.*

Proof. The lemma holds trivially for $d \in \{1, 2\}$. The proof for $d \geq 3$ is by contradiction. Let k be the largest index such that $\omega_k \notin M(\varphi)$. Then $1 < k < d$. Since $\omega_{k+1} \in M(\varphi)$ and ω_k is obtained by flipping a single variable in ω_{k+1}, we have $\omega_{k+1} \in BM(\varphi)$. Moreover, $\omega_{k+1}, \ldots, \omega_d$ is an H-path from $BM(\varphi)$ to ω_d which has length $d - k < d$, a contradiction. Hence, $\omega_i \in M(\varphi)$ for $i = 1, \ldots, d$. $\square$

Lemma 7. *If $\omega_1, \ldots, \omega_d$ is a shortest H-path from $BM(\varphi)$ to $\omega_d \in M(\varphi)$, then $\omega_d \in \mathcal{L}^{d-1}(BM(\varphi))$.*

Proof. By induction on d. For $d = 1$ (base case), we have $\omega_1 = \omega_d \in BM(\varphi)$, $\mathcal{L}^0(BM(\varphi)) = BM(\varphi)$ and hence $\omega_d \in \mathcal{L}^{d-1}(BM(\varphi))$. For $d > 1$ (inductive step), the subsequence $\omega_1, \ldots, \omega_{d-1}$ is a shortest H-path from $BM(\varphi)$ to ω_{d-1}. Moreover, $\omega_{d-1} \in M(\varphi)$ by Lemma 6. Hence, $\omega_{d-1} \in \mathcal{L}^{d-2}(BM(\varphi))$ by the induction hypothesis. Worlds ω_{d-1} and ω_d must have the forms α, ℓ and $\alpha, \bar{\ell}$. Since both are in $M(\varphi)$, then $\alpha \rightarrow \ell \notin R(\varphi)$. Hence $\omega_d \in \mathcal{L}^{d-1}(BM(\varphi))$ since $\mathcal{L}^{d-1}(BM(\varphi)) = \mathcal{L}(\mathcal{L}^{d-2}(BM(\varphi)))$. $\square$

We will now finish the proof by showing that $M(\varphi)$ is the least fixed point of $\mathcal{L}$. By Lemma 7, $M(\varphi) \subseteq \mathcal{L}^k(BM(\varphi))$ for some $k \geq 0$. Suppose that $BM(\varphi) \subseteq W$ and $\mathcal{L}(W) = W$ (a fixed point). Then $\mathcal{L}^k(W) = W$. Since $\mathcal{L}$ is monotonic (Lemma 5), we have $\mathcal{L}^k(BM(\varphi)) \subseteq \mathcal{L}^k(W)$ and hence $M(\varphi) \subseteq \mathcal{L}^k(BM(\varphi)) \subseteq \mathcal{L}^k(W) = W$. This implies $M(\varphi) \subseteq W$ so $M(\varphi)$ is the least fixed point of $\mathcal{L}$ and therefore the stationary point of the sequence $(\mathcal{L}^i(BM(\varphi)))_{i \in \mathbb{N}}$. $\square$

Proof of Proposition 4.

($\Rightarrow$) Suppose φ has b-rule $\alpha \rightarrow \ell$. By Definition 6, α, ℓ is an ℓ-boundary model of φ. By Definition 1, $\alpha, \ell \models \varphi$ and $\alpha, \bar{\ell} \models \neg\varphi$. From $\alpha, \ell \models \varphi$, we conclude that $\varphi \wedge \alpha$ is consistent. From $\alpha, \bar{\ell} \models \neg\varphi$, we conclude that $\alpha \wedge \bar{\ell} \wedge \varphi$ is inconsistent and hence $\varphi \models \neg(\alpha \wedge \bar{\ell})$ and further $\varphi \models \alpha \Rightarrow \ell$.

($\Leftarrow$) Suppose $\varphi \wedge \alpha$ is consistent and $\varphi \models \alpha \Rightarrow \ell$. From $\varphi \wedge \alpha$ being consistent, we conclude that $\alpha, \ell \models \varphi$ or $\alpha, \bar{\ell} \models \varphi$. From $\varphi \models \alpha \Rightarrow \ell$, we conclude $\alpha, \bar{\ell} \models \neg\varphi$ and hence $\alpha, \ell \models \varphi$. We now have that α, ℓ is an ℓ-boundary model of φ by Definition 1, and hence φ has b-rule $\alpha \rightarrow \ell$ by Definition 6. $\square$

Proof of Proposition 5.

The literal case. ($\Rightarrow$) Suppose φ is independent of literal ℓ. There must exist some NNF $\psi = \varphi$ that does not contain literal ℓ. Let α be a term that does not mention the variable X of ℓ and is such that $\psi \wedge \alpha$ is consistent. Then $\psi \wedge \alpha$ is a consistent NNF that does not mention literal ℓ so $(\psi \wedge \alpha)|\bar{\ell}$ is also consistent and hence $\psi \wedge \alpha \not\models \ell$. This means φ cannot have a b-rule of the form $\alpha \rightarrow \ell$.

($\Leftarrow$) Suppose φ has no b-rule of the form $\alpha \rightarrow \ell$. If φ has a model of the form α, ℓ, it also has $\alpha, \bar{\ell}$ as a model; otherwise, φ will have rule $\alpha \rightarrow \ell$. Since $(\alpha, \ell) \vee (\alpha, \bar{\ell}) = \alpha$, we can express φ as a DNF that does not contain literal ℓ. Hence, φ is independent of literal ℓ.

The variable case. Follows from the above case given that φ is independent of variable X iff it is independent of literal x and independent of literal $\bar{x}$. □

Proof of Proposition 6. This proof invokes Definition 6 frequently.

(a) We have $\alpha \rightarrow \ell \in R(\overline{\varphi})$ iff $\alpha, \ell \models \overline{\varphi}$ and $\alpha, \bar{\ell} \models \overline{\overline{\varphi}}$ iff $\alpha \rightarrow \bar{\ell} \in R(\varphi)$.

(b)
- If $\alpha \rightarrow \ell \in R(\varphi) \cap R(\psi)$, then $\alpha, \ell \models \varphi$, $\alpha, \bar{\ell} \models \overline{\varphi}$, $\alpha, \ell \models \psi$ and $\alpha, \bar{\ell} \models \overline{\psi}$. Thus, $\alpha, \ell \models \varphi \wedge \psi$ and $\alpha, \bar{\ell} \models \overline{\varphi} \vee \overline{\psi}$. Since $\overline{\varphi} \vee \overline{\psi} = \overline{\varphi \wedge \psi}$, we get $\alpha \rightarrow \ell \in R(\varphi \wedge \psi)$ and therefore $R(\varphi) \cap R(\psi) \subseteq R(\varphi \wedge \psi)$.
- If $\alpha \rightarrow \ell \in R(\varphi \wedge \psi)$, then $\alpha, \ell \models \varphi \wedge \psi$ and $\alpha, \bar{\ell} \models \overline{\varphi} \vee \overline{\psi}$. Hence, $\alpha, \ell \models \varphi$ and $\alpha, \ell \models \psi$. Moreover, $\alpha, \bar{\ell} \models \overline{\varphi}$ or $\alpha, \bar{\ell} \models \overline{\psi}$. This gives $\alpha \rightarrow \ell \in R(\varphi)$ or $\alpha \rightarrow \ell \in R(\psi)$, and therefore $R(\varphi \wedge \psi) \subseteq R(\varphi) \cup R(\psi)$.

(c)
- If $\alpha \rightarrow \ell \in R(\varphi) \cap R(\psi)$, then $\alpha, \ell \models \varphi$, $\alpha, \bar{\ell} \models \overline{\varphi}$, $\alpha, \ell \models \psi$ and $\alpha, \bar{\ell} \models \overline{\psi}$. Thus, $\alpha, \ell \models \varphi \vee \psi$ and $\alpha, \bar{\ell} \models \overline{\varphi} \wedge \overline{\psi}$. Since $\overline{\varphi} \wedge \overline{\psi} = \overline{\varphi \vee \psi}$, we get $\alpha \rightarrow \ell \in R(\varphi \vee \psi)$ and therefore $R(\varphi) \cap R(\psi) \subseteq R(\varphi \vee \psi)$.
- If $\alpha \rightarrow \ell \in R(\varphi \vee \psi)$, then $\alpha, \ell \models \varphi \vee \psi$ and $\alpha, \bar{\ell} \models \overline{\varphi} \wedge \overline{\psi}$. Hence, $\alpha, \ell \models \overline{\varphi}$ and $\alpha, \ell \models \overline{\psi}$. Moreover, $\alpha, \ell \models \varphi$ or $\alpha, \ell \models \psi$. This gives $\alpha \rightarrow \ell \in R(\varphi)$ or $\alpha \rightarrow \ell \in R(\psi)$, and therefore $R(\varphi \vee \psi) \subseteq R(\varphi) \cup R(\psi)$. □

Proof of Proposition 7. By Definition 6, a boundary model can generate at most n b-rules, hence $|R(\varphi)| \leq n \cdot |BM(\varphi)|$. The bound $n \cdot |BM(\varphi)| \leq n \cdot |M(\varphi)|$ is trivial since $BM(\varphi) \subseteq M(\varphi)$. Considering $\overline{\varphi}$ instead of φ, we have $|R(\overline{\varphi})| \leq n \cdot |BM(\overline{\varphi})| \leq n \cdot |M(\overline{\varphi})|$. Finally, Proposition 6(a) shows that $|R(\overline{\varphi})| = |R(\varphi)|$. □

Proof of Theorem 2. By Boole's expansion, we have $\varphi = (\ell \wedge (\varphi|\ell)) \vee (\bar{\ell} \wedge (\varphi|\bar{\ell}))$, which can be expanded using consensus into $\varphi = (\ell \wedge (\varphi|\ell)) \vee (\bar{\ell} \wedge (\varphi|\bar{\ell})) \vee ((\varphi|\ell) \wedge (\varphi|\bar{\ell}))$. By Definition 7, we have $\forall \ell \cdot \varphi = (\ell \vee (\varphi|\bar{\ell})) \wedge (\varphi|\ell)$, which can be expanded into $\forall \ell \cdot \varphi = (\ell \wedge (\varphi|\ell)) \vee ((\varphi|\ell) \wedge (\varphi|\bar{\ell}))$. This gives $\forall \ell \cdot \varphi \models \varphi$ and hence $M(\forall \ell \cdot \varphi) \subseteq M(\varphi)$. We next prove the two directions of the second part of the theorem using the expansions:

$$\begin{aligned}
\varphi &= (\ell \wedge (\varphi|\ell)) \vee (\bar{\ell} \wedge (\varphi|\bar{\ell})) && (2)\\
\varphi &= (\ell \wedge (\varphi|\ell)) \vee (\bar{\ell} \wedge (\varphi|\bar{\ell})) \vee ((\varphi|\ell) \wedge (\varphi|\bar{\ell})) && (3)\\
\forall \ell \cdot \varphi &= (\ell \wedge (\varphi|\ell)) \vee ((\varphi|\ell) \wedge (\varphi|\bar{\ell})) && (4)
\end{aligned}$$

($\Rightarrow$) Suppose $\omega \in M(\varphi)$ and $\omega \notin M(\forall \ell \cdot \varphi)$; that is, $\omega \models \varphi$ and $\omega \not\models \forall \ell \cdot \varphi$. Given Expansions (2) and (4), this implies $\omega \models \bar{\ell} \wedge (\varphi|\bar{\ell})$. Suppose ω is not an $\bar{\ell}$-boundary model of φ. Then $\omega[\ell] \models \varphi$ and hence $\omega[\ell] \models \ell \wedge \varphi$ and $\omega[\ell] \models \ell \wedge (\varphi|\ell)$. We now have $\omega \models \varphi|\bar{\ell}$ and $\omega[\ell] \models \varphi|\ell$. Since $\varphi|\ell$ does not mention the variable of literal ℓ, we also have $\omega \models (\varphi|\ell) \wedge (\varphi|\bar{\ell})$ which implies $\omega \models \forall \ell \cdot \varphi$ by Expansion (4). This is a contradiction with the supposition $\omega \notin M(\forall \ell \cdot \varphi)$ so ω must be an $\bar{\ell}$-boundary model of φ.
($\Leftarrow$) Suppose ω is an $\bar{\ell}$-boundary model of φ. Then $\bar{\ell} \in \omega$, $\omega \models \varphi$ and $\omega[\ell] \not\models \varphi$. We then have $\omega \in M(\varphi)$ so we just need to show that $\omega \notin M(\forall \ell \cdot \varphi)$. Suppose $\omega \in M(\forall \ell \cdot \varphi)$. Since $\bar{\ell} \in \omega$, we have $\omega \models (\varphi|\ell) \wedge (\varphi|\bar{\ell})$ by Expansion (4). Since $\varphi|\ell$ and $\varphi|\bar{\ell}$ do not mention the

variable of literal $\bar{\ell}$, we also have $\omega[\ell] \models (\varphi|\ell) \wedge (\varphi|\bar{\ell})$ and hence $\omega[\ell] \models \varphi$ by Expansion (3). This is a contradiction so $\omega \notin M(\forall \ell \cdot \varphi)$. $\square$

Proof of Theorem 3. We first prove $\forall \ell.\varphi \models \varphi \wedge \bigwedge_{\alpha \to \bar{\ell} \in R(\varphi)} \overline{\alpha}$. By Theorem 2, $\forall \ell.\varphi \models \varphi$. We next show that $\forall \ell.\varphi \models \overline{\alpha}$ whenever $\alpha \to \bar{\ell} \in R(\varphi)$. If $\alpha \to \bar{\ell} \in R(\varphi)$, we have $\alpha, \ell \models \overline{\varphi}$ by Definition 6 and hence $\varphi \models \overline{\alpha} \vee \bar{\ell}$. By Proposition 9, $\forall \ell.\varphi \models \forall \ell.(\overline{\alpha} \vee \bar{\ell})$. Since α does not contain the variable of literal ℓ by construction, $\forall \ell.(\overline{\alpha} \vee \bar{\ell}) = (\forall \ell.\overline{\alpha}) \vee (\forall \ell.\bar{\ell})$ by Proposition 20(d). Since $\forall \ell.\bar{\ell} = \perp$ we get $\forall \ell.\varphi \models \forall \ell.\overline{\alpha}$. By Theorem 2, $\forall \ell.\overline{\alpha} \models \overline{\alpha}$ so $\forall \ell.\varphi \models \overline{\alpha}$ and therefore $\forall \ell.\varphi \models \varphi \wedge \bigwedge_{\alpha \to \bar{\ell} \in R(\varphi)} \overline{\alpha}$.

We now prove $\varphi \wedge \bigwedge_{\alpha \to \bar{\ell} \in R(\varphi)} \overline{\alpha} \models \forall \ell.\varphi$. By Definition 7, $\forall \ell.\varphi = (\ell \vee (\varphi \mid \bar{\ell})) \wedge (\varphi \mid \ell)$. Consider a model $\omega \models \varphi \wedge \bigwedge_{\alpha \to \bar{\ell} \in R(\varphi)} \overline{\alpha}$. We will next show that $\omega \models \forall \ell.\varphi$. If $\omega \models \ell$, then $\omega \models \varphi \wedge \ell$. Since $\varphi \wedge \ell \models \varphi \mid \ell$, we now have $\omega \models \varphi \mid \ell$ and hence $\omega \models \forall \ell.\varphi$. If $\omega \models \bar{\ell}$, then $\omega \models \varphi \wedge \bar{\ell}$ and hence $\omega \models \varphi \mid \bar{\ell}$. We next show $\omega \models \varphi \mid \ell$ which gives us $\omega \models \forall \ell.\varphi$, therefore concluding the proof. Let $\omega = \alpha', \bar{\ell}$. We must have $\alpha', \ell \models \varphi$, otherwise $\alpha' \to \bar{\ell} \in R(\varphi)$ and then $\omega \models \overline{\alpha'}$ which is a contradiction. We now have $\alpha' \models \varphi$. Since $\varphi = (\ell \vee (\varphi \mid \bar{\ell})) \wedge (\bar{\ell} \vee (\varphi \mid \ell))$, we have $\alpha' \models \bar{\ell} \vee (\varphi \mid \ell)$. Since $\bar{\ell} \vee (\varphi \mid \ell)$ is independent of ℓ, we also have $\alpha' \models \forall \ell.(\bar{\ell} \vee (\varphi \mid \ell))$. Since $\bar{\ell}$ and $\varphi \mid \ell$ do not share variables, $\forall \ell.(\bar{\ell} \vee (\varphi \mid \ell)) = (\forall \ell.\bar{\ell}) \vee (\forall \ell.(\varphi \mid \ell))$ by Proposition 20(d). Moreover, $(\forall \ell.\bar{\ell}) \vee (\forall \ell.(\varphi \mid \ell)) = \varphi \mid \ell$ since $\forall \ell.\bar{\ell} = \perp$ and $\varphi \mid \ell$ is independent of ℓ. We now have $\alpha' \models \varphi \mid \ell$ and hence $\omega \models \varphi \mid \ell$. Therefore $\varphi \wedge \bigwedge_{\alpha \to \bar{\ell} \in R(\varphi)} \overline{\alpha} \models \forall \ell.\varphi$. $\square$

Proof of Proposition 8. By Lemma 3, we have $\forall \ell \cdot \varphi = (\forall X \cdot \varphi) \vee (\ell \wedge \varphi)$, where X is the variable of literal ℓ. Any implicant of φ that contains literal ℓ will also be an implicant of $\ell \wedge \varphi$ and hence an implicant of $\forall \ell \cdot \varphi$. $\square$

Proof of Proposition 9. By Theorem 4, $\forall \ell \cdot \varphi \models \varphi$. Given $\varphi \models \phi$, we now have $\forall \ell \cdot \varphi \models \phi$. Let ω be a model of $\forall \ell \cdot \varphi$. Then $\omega \models \forall \ell \cdot \varphi \models \varphi \models \phi$. If $\omega \not\models \forall \ell \cdot \phi$ then ω is an $\bar{\ell}$-boundary model of ϕ by Theorem 4 and hence ω is an $\bar{\ell}$-boundary model of φ by Lemma 1. By Theorem 4, $\omega \not\models \forall \ell \cdot \varphi$ which is a contradiction. We then have $\omega \models \forall \ell \cdot \phi$ and therefore $\forall \ell \cdot \varphi \models \forall \ell \cdot \phi$. $\square$

Proof of Proposition 10. Proposition 2(4) in (Lang et al., 2003) shows that φ is independent of literal ℓ if and only if $\neg\varphi$ is independent of literal $\bar{\ell}$. Hence, the result follows directly from Proposition 14 (Proposition 16 of (Lang et al., 2003)) and Theorem 5 (duality). $\square$

Proof of Proposition 11. Follows directly from Proposition 15 and Theorem 5. $\square$

Proof of Proposition 12. Follows directly from Proposition 16 and Theorem 5. $\square$

Proof of Theorem 4. The proof of this theorem is based on two lemmas that we state and prove next. The first lemma says that the result of universally quantifying literals is independent of their complements.

Lemma 8. *Let φ be a formula, $\ell_1, \ldots, \ell_n$ be literals and ω be a world. If $\omega \models \forall \ell_1, \ldots, \ell_n \cdot \varphi$, then $\omega \setminus \{\bar{\ell}_1, \ldots, \bar{\ell}_n\} \models \forall \ell_1, \ldots, \ell_n \cdot \varphi$.*

Proof. Suppose $\omega \models \forall \ell_1, \ldots, \ell_n \cdot \varphi$. By Proposition 10, $\forall \ell_1, \ldots, \ell_n \cdot \varphi$ is independent of literals $\bar{\ell}_1, \ldots, \bar{\ell}_n$. Thus, by Proposition 14, $\exists \bar{\ell}_1, \ldots, \bar{\ell}_n \cdot \omega \models \forall \ell_1, \ldots, \ell_n \cdot \varphi$. Since $\exists \bar{\ell}_1, \ldots, \bar{\ell}_n \cdot \omega = \omega \setminus \{\bar{\ell}_1, \ldots, \bar{\ell}_n\}$, we finally have $\omega \setminus \{\bar{\ell}_1, \ldots, \bar{\ell}_n\} \models \forall \ell_1, \ldots, \ell_n \cdot \varphi$. $\square$

The second lemma identifies a class of α-independent models that are preserved by universal literal quantification.

Lemma 9. *Let φ be a formula, $\ell_1, \ldots, \ell_n$ be literals, ω be a world and $\alpha = \omega \cap \{\bar{\ell}_1, \ldots, \bar{\ell}_n\}$. If ω is an α-independent model of φ, then ω is an α-independent model of $\forall \ell_1, \ldots, \ell_n \cdot \varphi$.*

Proof. Suppose ω is an α-independent model of φ. Then $\alpha \subseteq \omega$ and $\omega \setminus \alpha \models \varphi$. By Proposition 9, $\forall \ell_1, \ldots, \ell_n \cdot (\omega \setminus \alpha) \models \forall \ell_1, \ldots, \ell_n \cdot \varphi$. Since $\bar{\ell}_i \notin (\omega \setminus \alpha)$ for $i = 1, \ldots, n$, we get $\forall \ell_1, \ldots, \ell_n \cdot (\omega \setminus \alpha) = \omega \setminus \alpha$ by Lemma 2. Hence, $\omega \setminus \alpha \models \forall \ell_1, \ldots, \ell_n \cdot \varphi$ so ω is an α-independent model of $\forall \ell_1, \ldots, \ell_n \cdot \varphi$. $\square$

We are now ready to prove the theorem. Let ω be a world and $\alpha = \omega \cap \{\bar{\ell}_1, \ldots, \bar{\ell}_n\}$.

($\Rightarrow$) Suppose $\omega \models \forall \ell_1, \ldots, \ell_n \cdot \varphi$. By Lemma 8, $\omega \setminus \{\bar{\ell}_1, \ldots, \bar{\ell}_n\} \models \forall \ell_1, \ldots, \ell_n \cdot \varphi$ and hence $\omega \setminus \alpha \models \forall \ell_1, \ldots, \ell_n \cdot \varphi$. Since $\alpha \subseteq \omega$, we get that ω is an α-independent model of $\forall \ell_1, \ldots, \ell_n \cdot \varphi$. By Theorem 2, $\forall \ell_1, \ldots, \ell_n \cdot \varphi \models \varphi$. By Lemma 4 and $\omega \models \forall \ell_1, \ldots, \ell_n \cdot \varphi \models \varphi$, we get that ω is an α-independent model of φ.

($\Leftarrow$) Suppose ω is an α-independent model of φ. By Lemma 9, ω is an α-independent model of $\forall \ell_1, \ldots, \ell_n \cdot \varphi$ and hence $\omega \models \forall \ell_1, \ldots, \ell_n \cdot \varphi$. $\square$

Proof of Proposition 13. Let $\omega = \alpha, \beta$ be a model of formula φ where α and β are disjoint terms.

($\Rightarrow$) Suppose ω is an α-independent model of φ. Then $\beta \models \varphi$. Moreover, for every world β, γ, ℓ (where β, γ and ℓ are disjoint) we must have $\beta, \gamma, \ell \models \varphi$ and $\beta, \gamma, \bar{\ell} \models \varphi$. By Definition 6, φ cannot then have a b-rule of the form $\beta, \gamma \to \ell$ or $\beta, \gamma \to \bar{\ell}$.

($\Leftarrow$) Suppose φ has no b-rules of the form $\beta, \gamma \to \ell$. By Definition 6, for every world of the form β, γ, ℓ, we have $\beta, \gamma, \ell \models \varphi$ only if $\beta, \gamma, \bar{\ell} \models \varphi$. Any world $\omega^\star \supseteq \beta$ can be obtained from world $\omega = \beta, \alpha$ by a sequence of single flips to the variables of α. Hence, any such world $\omega^\star$ is a model of φ, which implies $\beta \models \varphi$. Therefore, ω must be an α-independent model of φ. $\square$

Proof of Theorem 5. We have $\neg(\forall \ell \cdot \neg\varphi) = \neg((\ell \vee (\neg\varphi|\bar{\ell})) \wedge (\neg\varphi|\ell)) = (\bar{\ell} \wedge (\varphi|\bar{\ell})) \vee (\varphi|\ell) = \exists \ell \cdot \varphi$. We can similarly show $\forall \ell \cdot \varphi = \neg(\exists \ell \cdot \neg\varphi)$. $\square$

Proof of Proposition 14. See Proposition 16 of (Lang et al., 2003). $\square$

Proof of Proposition 15. See footnote 4 in (Lang et al., 2003). $\square$

Proof of Proposition 16. See Proposition 20 of (Lang et al., 2003). $\square$

Proof of Theorem 6. By Proposition 14, $M(\varphi) \subseteq M(\exists \ell \cdot \varphi)$. By Theorem 5 (duality), $\omega \in M(\exists \ell \cdot \varphi)$ iff $\omega \notin M(\forall \ell \cdot \overline{\varphi})$. Moreover, $\omega \notin M(\varphi)$ iff $\omega \in M(\overline{\varphi})$. Thus, $\omega \in M(\exists \ell \cdot \varphi)$ and $\omega \notin M(\varphi)$ iff $\omega \notin M(\forall \ell \cdot \overline{\varphi})$ and $\omega \in M(\overline{\varphi})$ iff ω is an $\bar{\ell}$-boundary model of $\overline{\varphi}$ (by second part of Theorem 2). $\square$

Proof of Theorem 7. By Theorem 3, $\forall \ell.\overline{\varphi} = \overline{\varphi} \wedge \bigwedge_{\alpha \to \bar{\ell} \in R(\overline{\varphi})} \overline{\alpha}$. Negating the two sides, we get $\neg(\forall \ell.\overline{\varphi}) = \varphi \vee \bigvee_{\alpha \to \bar{\ell} \in R(\overline{\varphi})} \alpha$. By Theorem 5 (duality), $\neg(\forall \ell.\overline{\varphi}) = \exists \ell.\varphi$. By Proposition 6, $\alpha \to \bar{\ell} \in R(\overline{\varphi})$ is equivalent to $\alpha \to \ell \in R(\varphi)$. This concludes the proof. □

Proof of Proposition 17. Suppose $\varphi \models \beta$ and $\bar{\ell} \in \beta$. Let γ be the term containing the complements of literals in clause β ($\gamma = \overline{\beta}$). Then $\gamma \models \overline{\varphi}$ and $\ell \in \gamma$. Moreover, $\gamma \models \forall \ell \cdot \overline{\varphi}$ by Proposition 8, and $\exists \ell \cdot \varphi \models \beta$ by contraposition and Theorem 5 (duality). □

Proof of Proposition 18. From $\varphi \models \phi$, we get $\overline{\phi} \models \overline{\varphi}$. By Proposition 9, $\forall \ell \cdot \overline{\phi} \models \forall \ell \cdot \overline{\varphi}$. Finally, $\exists \ell \cdot \varphi \models \exists \ell \cdot \phi$ by contraposition and Theorem 5 (duality). □

Proof of Proposition 19. The results for $\top/\bot$ follow directly from Definitions 7 and 8.

Using Definition 8, $\exists \ell_1.\ell_2 = (\ell_2|\ell_1) \vee (\bar{\ell}_1 \wedge (\ell_2|\bar{\ell}_1))$. If $\ell_1 = \ell_2$, we have $\ell_2|\ell_1 = \top$ and then $\exists \ell_1.\ell_2 = \top$. If $\ell_1 \neq \ell_2$, then either $\ell_1 = \bar{\ell}_2$ or $\ell_1 \neq \bar{\ell}_2$. If $\ell_1 = \bar{\ell}_2$, then $\ell_2|\ell_1 = \bot$ and $\ell_2|\bar{\ell}_1 = \top$ leading to $\exists \ell_1.\ell_2 = \bar{\ell}_1 = \ell_2$. If $\ell_1 \neq \bar{\ell}_2$, then $\ell_2|\ell_1 = \ell_2|\bar{\ell}_1 = \ell_2$ and hence $\exists \ell_1.\ell_2 = \ell_2 \vee (\bar{\ell}_1 \wedge \ell_2) = \ell_2$.

The results for $\forall \ell_1.\ell_2$ follow from the results for $\exists \ell_1.\ell_2$ using Theorem 5 (duality). □

Proof of Proposition 20.

(a) By Definition 8, we have

$$\begin{aligned}
\exists \ell(\alpha \vee \beta) &= ((\alpha \vee \beta)|\ell) \vee (\bar{\ell} \wedge ((\alpha \vee \beta)|\bar{\ell})) \\
&= (\alpha|\ell) \vee (\beta|\ell) \vee (\bar{\ell} \wedge ((\alpha|\bar{\ell}) \vee (\beta|\bar{\ell}))) \\
&= (\alpha|\ell) \vee (\beta|\ell) \vee (\bar{\ell} \wedge (\alpha|\bar{\ell})) \vee (\bar{\ell} \wedge (\beta|\bar{\ell})) \\
&= (\exists \ell.\alpha) \vee (\exists \ell.\beta)
\end{aligned}$$

(b) Follows from Part (a) and Theorem 5 (duality).

(c) Suppose literals ℓ and $\bar{\ell}$ do not appear in β. By Definition 8, $\exists \ell.\beta = \beta$ and also:

$$\begin{aligned}
\exists \ell(\alpha \wedge \beta) &= ((\alpha \wedge \beta)|\ell) \vee (\bar{\ell} \wedge ((\alpha \wedge \beta)|\bar{\ell})) \\
&= ((\alpha|\ell) \wedge \beta) \vee (\bar{\ell} \wedge (\alpha|\bar{\ell}) \wedge \beta) \\
&= ((\alpha|\ell) \vee (\bar{\ell} \wedge (\alpha|\bar{\ell}))) \wedge \beta \\
&= (\exists \ell.\alpha) \wedge (\exists \ell.\beta)
\end{aligned}$$

(d) Follows from Part (c) and Theorem 5 (duality). □

Proof of Proposition 21. The first statement follows directly from Propositions 19 and 20(b,d). By Proposition 20(b), universal literal quantification distributes over conjuncts (clauses). By Proposition 20(d), it also distributes over disjuncts (literals) when they do not share variables (the literals of a clause are over distinct variables). Hence, we just need to replace each literal ℓ' in the CNF with $\forall \ell \cdot \ell'$. By Proposition 19, $\forall \ell \cdot \ell' = \bot$ if $\bar{\ell} = \ell'$ and $\forall \ell \cdot \ell' = \ell'$.

As to the second statement, for literal ℓ, the clauses of CNF Δ can be partitioned into:

$$\begin{aligned}
\Delta_a &= \{\alpha \mid \alpha \in \Delta \text{ and } \ell \in \alpha\} \\
\Delta_b &= \{\alpha \mid \alpha \in \Delta \text{ and } \bar{\ell} \in \alpha\} \\
\Delta_c &= \{\alpha \mid \alpha \in \Delta \text{ and } \ell \notin \alpha, \bar{\ell} \notin \alpha\}
\end{aligned}$$

By Definition 8, $\exists \ell.\Delta = (\Delta|\ell) \vee (\bar{\ell} \wedge (\Delta|\bar{\ell})) = (\bar{\ell} \vee (\Delta|\ell)) \wedge ((\Delta|\ell) \vee (\Delta|\bar{\ell}))$ so we get CNFs:

$$\begin{aligned}
\Delta|\ell &= \{\alpha \setminus \{\bar{\ell}\} \mid \alpha \in \Delta_b\} \cup \Delta_c \\
\Delta|\bar{\ell} &= \{\alpha \setminus \{\ell\} \mid \alpha \in \Delta_a\} \cup \Delta_c \\
\Delta_A &= \{\bar{\ell} \vee \alpha \mid \alpha \in \Delta|\ell\} \text{ represents } \bar{\ell} \vee (\Delta|\ell) \\
\Delta_B &= \{\alpha \vee \beta \mid \alpha \in \Delta|\ell \text{ and } \beta \in \Delta|\bar{\ell}\} \text{ represents } (\Delta|\ell) \wedge (\Delta|\bar{\ell}) \\
\exists \ell.\Delta &= \Delta_A \cup \Delta_B
\end{aligned}$$

CNF Δ_B does not contain the variable of literal ℓ and can be expressed as follows:

$$\begin{aligned}
\Delta_B &= \Delta_c \cup \Delta_d \\
\Delta_d &= \{(\alpha \setminus \{\ell\}) \cup (\beta \setminus \{\bar{\ell}\}) \mid \alpha \in \Delta, \beta \in \Delta, \ell \in \alpha \text{ and } \bar{\ell} \in \beta\}
\end{aligned}$$

where Δ_d contains the resolvents of Δ on the variable of literal ℓ. We can express CNF Δ_A as:

$$\Delta_A = \{\bar{\ell} \vee (\alpha \setminus \{\bar{\ell}\}) \mid \alpha \in \Delta_b\} \cup \{\bar{\ell} \vee \alpha \mid \alpha \in \Delta_c\}.$$

The first part is equivalent to Δ_b and the second part is subsumed by Δ_c so we now have:

$$\exists \ell.\Delta = \Delta_A \cup \Delta_B = \Delta_b \cup \Delta_c \cup \Delta_d$$

That is, $\exists \ell.\Delta$ consists of all clauses of Δ that do not contain literal ℓ (Δ_b and Δ_c) in addition to all resolvents of Δ on the variable X of literal ℓ (Δ_d). When Δ is closed under resolution on variable X, we have $\Delta_d \subseteq \Delta_c$. In this case, $\exists \ell \cdot \Delta$ can be obtained from Δ by keeping only its clauses Δ_b and Δ_c, which is equivalent to removing clauses Δ_a (containing literal ℓ) as claimed by the theorem. $\square$

Proof of Corollary 1. Follows directly from Proposition 21 given that a CNF Δ in prime implicate form is such that the resolvent of any two clauses of Δ is subsumed by a clause of Δ (Quine, 1955). See also Proposition 19 in (Lang et al., 2003). $\square$

Proof of Proposition 22. Follows directly from Proposition 21 and Theorem 5 (duality). $\square$

Proof of Corollary 2. Follows directly from Proposition 22 given that a DNF Δ in prime implicant form is such that the consensus of any two terms of Δ is subsumed by a term of Δ (Quine, 1955). $\square$

Proof of Proposition 23. By Propositions 19 and 20(a,c), one can existentially quantify literals from a Decision-DNNF circuit in linear time since conjuncts in these circuits do not share variables. The result is a DNNF circuit as one would only be replacing some literals with $\top$, therefore preserving the decomposability property. $\square$

Proof of Proposition 24. (1) $(\ell \vee \beta) \wedge (\bar{\ell} \vee \alpha)$ is equivalent to $(\ell \wedge \alpha) \vee (\bar{\ell} \wedge \beta) \vee (\alpha \wedge \beta)$, which is equivalent to $(\ell \wedge \alpha) \vee (\bar{\ell} \wedge \beta)$ since $\alpha \wedge \beta$ is subsumed by $(\ell \wedge \alpha) \vee (\bar{\ell} \wedge \beta)$. (2) Γ can be obtained from Δ by flipping some $\vee$ to $\wedge$ and vice versa. (3) Every disjunction in Γ appears in a fragment $(\ell \vee \beta) \wedge (\bar{\ell} \vee \alpha)$. By definition of a Decision-DNNF circuit, we know that ℓ shares no variables with β, and $\bar{\ell}$ share no variables with α. $\square$

Proof of Proposition 25. Using Proposition 24, we can in linear time transform a Decision-DNNF circuit into an NNF circuit in which disjuncts do not share variables. Using Propositions 19 and 20(b,d), we can then universally quantify literals in linear time, leading to an NNF circuit since we only replace some literals with constants. □

Proof of Proposition 26. By Propositions 19 and 20(a,c), one can existentially quantify literals from an SDD circuit in linear time since conjuncts in these circuits do not share variables. The result is guaranteed to be a DNNF circuit as one would only be replacing some literals with $\top$, therefore preserving the decomposability property. □

Proof of Proposition 27.

(1) We first observe that primes $p_1, \ldots, p_n$ form a partition. Let $N = \{1, \ldots, n\}$. Then $\bigvee_{i \in N} p_i = \top$. Moreover, for $S \subseteq N$, $\bigvee_{i \in S} p_i = \bigwedge_{i \in N \setminus S} \neg p_i$. We now have:

$$\begin{aligned}(\neg p_1 \vee s_1) \wedge \ldots \wedge (\neg p_n \vee s_n) &= \bigvee_{S \subseteq N} \left[\left[\bigwedge_{i \in N \setminus S} \neg p_i \right] \wedge \left[\bigwedge_{j \in S} s_j \right] \right] \\ &= \bigvee_{S \subseteq N} \left[\left[\bigvee_{i \in S} p_i \right] \wedge \left[\bigwedge_{j \in S} s_j \right] \right].\end{aligned}$$

We will consider the above disjuncts according to the set S. When $S = \{\}$, the disjunct is $\bot$. When $S = \{i\}$, the disjunct is $(p_i \wedge s_i)$. When $S = N$, the disjunct is $(s_1 \wedge \ldots \wedge s_n)$. Otherwise, $1 < |S| < N$ and the disjunct is equivalent to $\bigvee_{i \in S}(p_i \wedge \bigwedge_{j \in S} s_j)$. Each term $(p_i \wedge \bigwedge_{j \in S} s_j)$ is subsumed by the disjunct $(p_i \wedge s_i)$ generated by $S = \{i\}$. Moreover, the term $(s_1 \wedge \ldots \wedge s_n)$ is subsumed by $(p_1 \wedge s_1) \vee \ldots \vee (p_n \wedge s_n)$ since primes p_i form a partition. Hence, $(\neg p_1 \vee s_1) \wedge \ldots \wedge (\neg p_n \vee s_n) = (p_1 \wedge s_1) \vee \ldots \vee (p_n \wedge s_n)$.

(2) An SDD circuit is an NNF circuit. We can construct a negation for each node in an NNF circuit while at most doubling the size of the NNF circuit, a process that can be done in time linear in the NNF circuit size. This can be done by traversing the NNF circuit bottom, while constructing a negation for each encountered node. The process is trivial for constants and literals. For a disjunction $\alpha_1 \vee \ldots \vee \alpha_n$, the negation is $\neg\alpha_1 \wedge \ldots \wedge \neg\alpha_n$ and we already have nodes for all $\neg\alpha_i$. A similar process is applied to conjunctions. We can therefore replace each fragment $(p_1 \wedge s_1) \vee \ldots \vee (p_n \wedge s_n)$ by the fragment $(\neg p_1 \vee s_1) \wedge \ldots \wedge (\neg p_n \vee s_n)$ in time linear in the size of SDD circuit.

(3) Every disjunction in Γ appears in a fragment $(\neg p_1 \vee s_1) \wedge \ldots \wedge (\neg p_n \vee s_n)$. By definition of SDDs, prime p_i shares no variables with sub s_i so $\neg p_i$ shares no variables with s_i. □

Proof of Proposition 28. Using Proposition 27, we can in linear time transform an SDD circuit into an NNF circuit in which disjuncts do not share variables. Using Propositions 19 and 20(b,d), we can then universally quantify literals in linear time, leading to an NNF circuit since we only replace some literals with constants. □

Proof of Theorem 8. Given Definition 12, we need to show $\Delta(\delta) = \Delta(\delta_{\uparrow X_1,\ldots,X_n})$ iff $\delta \models \forall X_1, \ldots, X_n \cdot \Delta_\delta$. We next show both directions of the theorem.

($\Rightarrow$) Suppose $\Delta(\delta) = \Delta(\delta_{\uparrow X_1,\ldots,X_n})$. Then $\delta_{\uparrow X_1,\ldots,X_n} \models \Delta_\delta$. By Propositions 9 and 12, we have $\forall X_1,\ldots,X_n \cdot \delta_{\uparrow X_1,\ldots,X_n} \models \forall X_1,\ldots,X_n \cdot \Delta_\delta$. Since variables $X_1,\ldots,X_n$ are not mentioned in $\delta_{\uparrow X_1,\ldots,X_n}$, we also have $\delta_{\uparrow X_1,\ldots,X_n} \models \forall X_1,\ldots,X_n \cdot \Delta_\delta$. Since $\delta \models \delta_{\uparrow X_1,\ldots,X_n}$, we finally get $\delta \models \forall X_1,\ldots,X_n \cdot \Delta_\delta$.

($\Leftarrow$) Suppose $\delta \models \forall X_1,\ldots,X_n \cdot \Delta_\delta$. Then $\exists X_1,\ldots,X_n \cdot \delta \models \exists X_1,\ldots,X_n(\forall X_1,\ldots,X_n \cdot \Delta_\delta)$ by Proposition 18. Moreover, $\forall X_1,\ldots,X_n \cdot \Delta_\delta$ is independent of variables $X_1,\ldots,X_n$ by Proposition 1, so we have $\exists X_1,\ldots,X_n(\forall X_1,\ldots,X_n \cdot \Delta_\delta) = \forall X_1,\ldots,X_n \cdot \Delta_\delta$. Since $\exists X_1,\ldots,X_n \cdot \delta = \delta_{\uparrow X_1,\ldots,X_n}$, we get $\delta_{\uparrow X_1,\ldots,X_n} \models \forall X_1,\ldots,X_n \cdot \Delta_\delta$. Since $\forall X_1,\ldots,X_n \cdot \Delta_\delta \models \Delta_\delta$, we have $\delta_{\uparrow X_1,\ldots,X_n} \models \Delta_\delta$. Hence, $\Delta(\delta) = \Delta(\delta_{\uparrow X_1,\ldots,X_n})$. □

Proof of Theorem 9. Let $\alpha = \delta \cap \{\bar{\ell}_1,\ldots,\bar{\ell}_n\}$. It suffices to show that decision $\Delta(\delta)$ is independent of characteristics $\bar{\ell}_1,\ldots,\bar{\ell}_n$ iff δ is an α-independent model of Δ_δ. If this holds, Theorem 9 will then follow directly from Theorem 4.

($\Rightarrow$) Suppose decision $\Delta(\delta)$ is independent of characteristics $\bar{\ell}_1,\ldots,\bar{\ell}_n$. By Definition 13, $\Delta(\delta) = \Delta(\delta \setminus \{\bar{\ell}_1,\ldots,\bar{\ell}_n\})$ and hence $\delta \setminus \{\bar{\ell}_1,\ldots,\bar{\ell}_n\} \models \Delta_\delta$. By definition of α, we now have $\delta \setminus \alpha \models \Delta_\delta$. Since $\alpha \subseteq \delta$, then δ is an α-independent model of Δ_δ.

($\Leftarrow$) Suppose δ is an α-independent model of Δ_δ. Then $\delta \setminus \alpha \models \Delta_\delta$ and hence $\delta \setminus \{\bar{\ell}_1,\ldots,\bar{\ell}_n\} \models \Delta_\delta$. By definition, $\delta \models \Delta_\delta$ so $\Delta(\delta) = \Delta(\delta \setminus \{\bar{\ell}_1,\ldots,\bar{\ell}_n\})$. By Definition 13, decision $\Delta(\delta)$ is independent of characteristics $\bar{\ell}_1,\ldots,\bar{\ell}_n$. □

Proof of Proposition 29. By Proposition 12, $\forall X_1,\ldots,X_n \cdot \Delta = \forall \ell_1,\bar{\ell}_1,\ldots,\ell_n,\bar{\ell}_n \cdot \Delta$. By Proposition 11, $\forall \ell_1,\bar{\ell}_1,\ldots,\ell_n,\bar{\ell}_n \cdot \Delta = \forall \bar{\ell}_1,\ldots,\bar{\ell}_n(\forall \ell_1,\ldots,\ell_n \cdot \Delta)$. By Theorem 2, $\forall \bar{\ell}_1,\ldots,\bar{\ell}_n(\forall \ell_1,\ldots,\ell_n \cdot \Delta) \models \forall \ell_1,\ldots,\ell_n \cdot \Delta$. Hence, $\forall X_1,\ldots,X_n \cdot \Delta \models \forall \ell_1,\ldots,\ell_n \cdot \Delta$. □

Proof of Theorem 10. Let $\gamma = \{\ell_1,\ldots,\ell_m\}$, $X_{m+1},\ldots,X_n$ be all classifier features not mentioned in γ and let Γ be the complete reason $\forall \ell_1,\ldots,\ell_m,X_{m+1},\ldots,X_n \cdot \Delta_\gamma$. By definition, $\Delta(\gamma^\star) = \Delta(\gamma)$ is equivalent to $\gamma^\star \models \Delta_\gamma$. Moreover, $\Gamma \models \Delta_\gamma$ by Proposition 2 and Theorem 2. We next prove both directions of the theorem.

($\Rightarrow$) Suppose $\gamma^\star$ is a sufficient reason for decision $\Delta(\gamma)$. By definition, $\gamma^\star$ is a prime implicant of the complete reason Γ: $\gamma^\star \models \Gamma$ and no strict subset of $\gamma^\star$ satisfies this property. We now have $\gamma^\star \models \Gamma \models \Delta_\gamma$ and need to show that no strict subset of $\gamma^\star$ satisfies $\gamma^\star \models \Delta_\gamma$. Suppose to the contrary: $\alpha \subset \gamma^\star$ and $\alpha \models \Delta_\gamma$. By Propositions 9 and 12, we have $\forall \ell_1,\ldots,\ell_m,X_{m+1},\ldots,X_n \cdot \alpha \models \forall \ell_1,\ldots,\ell_m,X_{m+1},\ldots,X_n \cdot \Delta_\gamma$. Since variables $X_{m+1},\ldots,X_n$ do not appear in α and $\alpha \subseteq \{\ell_1,\ldots,\ell_m\}$, we have $\forall \ell_1,\ldots,\ell_m,X_{m+1},\ldots,X_n \cdot \alpha = \alpha$ and then $\alpha \models \forall \ell_1,\ldots,\ell_m,X_{m+1},\ldots,X_n \cdot \Delta_\gamma$. Since $\alpha \subset \gamma^\star$, then $\gamma^\star$ cannot be a prime implicant of Γ which is a contradiction. Hence, $\gamma^\star$ is a minimal subset of γ that satisfies $\gamma^\star \models \Delta_\gamma$ (and $\Delta(\gamma^\star) = \Delta(\gamma)$).

($\Leftarrow$) Suppose $\gamma^\star$ is a minimal subset of γ that satisfies $\Delta(\gamma^\star) = \Delta(\gamma)$. Then $\gamma^\star$ is a prime implicant of Δ_γ. We need to show that $\gamma^\star$ is a sufficient reason for decision $\Delta(\gamma)$ which by definition is equivalent to $\gamma^\star$ being a prime implicant of the complete reason Γ. Since $\gamma^\star \models \Delta_\gamma$, we can use Propositions 9 and 12 as in the first part to get $\gamma^\star \models \Gamma$ ($\gamma^\star$ is an implicant of Γ). Since $\gamma^\star \models \Gamma \models \Delta_\gamma$, then $\gamma^\star$ must be a prime implicant of Γ, otherwise it cannot be a prime implicant of Δ_γ. □

Proof of Theorem 11. By Theorem 8, $\forall X_1,\ldots,X_n \cdot \Delta$ characterizes all positive instances with decisions that are independent of protected features $X_1,\ldots,X_n$; that is, positive instances with unbiased decisions. Therefore, $\Delta \wedge \neg(\forall X_1,\ldots,X_n \cdot \Delta)$ characterizes all positive instances with biased decisions. One can similarly show the second part of the theorem, which characterizes negative instances with biased decisions. □

Proof of Theorem 12. The following proof invokes Definition 6 frequently.

(a) Suppose $\alpha \to \ell_i \in R(\varphi)$. Then $\alpha, \ell_i \models \varphi$ and $\alpha, \bar{\ell}_i \not\models \varphi$. We have $\alpha, \ell_i \models \forall \ell_i \varphi$ by Proposition 8 and $\alpha, \bar{\ell}_i \not\models \forall \ell_i \cdot \varphi$ by Theorem 2. Hence, $\alpha \to \ell_i \in R(\forall \ell_i \cdot \varphi)$.

(b) Suppose $\alpha \to \bar{\ell}_i \in R(\forall \ell_i \cdot \varphi)$. Then $\alpha, \bar{\ell}_i \models \forall \ell_i \cdot \varphi$ and $\alpha, \ell_i \not\models \forall \ell_i \cdot \varphi$. Since $\alpha, \bar{\ell}_i \models \forall \ell_i \cdot \varphi$, we get $\alpha, \bar{\ell}_i \models \varphi$ by Theorem 2. Consider two cases: $\alpha, \ell_i \models \varphi$ or $\alpha, \ell_i \not\models \varphi$. The first case is impossible since it implies $\alpha, \ell_i \models \forall \ell_i \cdot \varphi$ by Proposition 8 so we must have $\alpha, \ell_i \not\models \varphi$. Since $\alpha, \bar{\ell}_i \models \varphi$, then $\alpha, \bar{\ell}_i$ is an $\bar{\ell}_i$-boundary model for φ so it cannot be a model of $\forall \ell_i \cdot \varphi$ by Theorem 2, which is a contradiction. We must therefore have $\alpha \to \bar{\ell}_i \notin R(\forall \ell_i \cdot \varphi)$.

(c) Similar to Part (a).

(d) Suppose $\alpha, \bar{\ell}_i \to \ell_j \in R(\varphi)$. Then $\alpha, \bar{\ell}_i, \bar{\ell}_j \not\models \varphi$ and hence $\alpha, \bar{\ell}_i, \bar{\ell}_j \not\models \forall \ell_i \cdot \varphi$ by Theorem 2.
(⇒) Suppose $\alpha, \bar{\ell}_i \to \ell_j \in R(\forall \ell_i \cdot \varphi)$. Then $\alpha, \bar{\ell}_i, \ell_j \models \forall \ell_i \cdot \varphi$ and $\alpha, \bar{\ell}_i, \bar{\ell}_j \not\models \forall \ell_i \cdot \varphi$. To show $\alpha, \ell_j \to \bar{\ell}_i \notin R(\varphi)$, it suffices to show $\alpha, \ell_i, \ell_j \models \varphi$. Suppose $\alpha, \ell_i, \ell_j \not\models \varphi$. Then $\alpha, \ell_i, \ell_j \not\models \forall \ell_i \cdot \varphi$ by Theorem 2. Since $\alpha, \bar{\ell}_i, \ell_j \models \forall \ell_i \cdot \varphi$, we have $\alpha, \ell_j \to \bar{\ell}_i \in R(\forall \ell_i \cdot \varphi)$ which contradicts (b). Hence, $\alpha, \ell_i, \ell_j \models \varphi$ and $\alpha, \ell_j \to \bar{\ell}_i \notin R(\varphi)$.
(⇐) Suppose $\alpha, \ell_j \to \bar{\ell}_i \notin R(\varphi)$. To show $\alpha, \bar{\ell}_i \to \ell_j \in R(\forall \ell_i \cdot \varphi)$, we need to show that we have $\alpha, \bar{\ell}_i, \ell_j \models \forall \ell_i \cdot \varphi$ and $\alpha, \bar{\ell}_i, \bar{\ell}_j \not\models \forall \ell_i \cdot \varphi$. The latter follows from the supposition $\alpha, \bar{\ell}_i \to \ell_j \in R(\varphi)$. The former also holds since supposition $\alpha, \ell_j \to \bar{\ell}_i \notin R(\varphi)$ implies that $\alpha, \bar{\ell}_i, \ell_j$ is not an $\bar{\ell}_i$-boundary model for φ (Definition 6) so it is not dropped when universally quantifying literal ℓ_i (Theorem 2). Hence, $\alpha, \bar{\ell}_i \to \ell_j \in R(\forall \ell_i \cdot \varphi)$. □

Proof of Theorem 13. Let b-rule $r = \beta \to \ell_k$, world $\omega = \beta, \ell_k$ and suppose $r \notin R(\varphi)$ and that $r \in R(\forall \ell_i \cdot \varphi)$. We next show that $k \neq i$ and $\bar{\ell}_i \in \beta$, which is sufficient to prove the theorem. By Theorem 12(b), $\ell_k \neq \bar{\ell}_i$. By $r \in R(\forall \ell_i \cdot \varphi)$ and Definition 6, $\omega \in M(\forall \ell_i \cdot \varphi)$ and $\omega[\bar{\ell}_k] \notin M(\forall \ell_i \cdot \varphi)$. By $\omega \in M(\forall \ell_i \cdot \varphi)$ and Theorem 2, $\omega \in M(\varphi)$. By $\omega \in M(\varphi)$, $r \notin R(\varphi)$ and Definition 6, $\omega[\bar{\ell}_k] \in M(\varphi)$. By $\omega[\bar{\ell}_k] \in M(\varphi)$, $\omega[\bar{\ell}_k] \notin M(\forall \ell_i \cdot \varphi)$ and Theorem 2, world $\omega^\star = \omega[\bar{\ell}_k]$ must be an $\bar{\ell}_i$-boundary model of φ; that is, $\bar{\ell}_i \in \omega^\star$ and $\omega^\star[\ell_i] \notin M(\varphi)$. To show $i \neq k$, suppose the contrary $i = k$. Then $\omega^\star[\ell_i] = (\omega[\bar{\ell}_k])[\ell_i] = (\omega[\bar{\ell}_k])[\ell_k] = \omega[\ell_k] = \omega$. This conflicts with $\omega \in M(\varphi)$ and $\omega^\star[\ell_i] \notin M(\varphi)$ so $i \neq k$. Since $\bar{\ell}_i \in \omega^\star$ we get $\bar{\ell}_i \in \beta$. □

Proof of Theorem 14. Let world $\omega = \alpha, \bar{\ell}_i, \ell_j$ and let

(A) $\alpha, \ell_i \to \ell_j \in R(\varphi)$

(B) $\alpha, \bar{\ell}_j \to \bar{\ell}_i \in R(\varphi)$

(C) $\alpha, \ell_j \to \ell_i \notin R(\varphi)$

(D) $\alpha, \bar{\ell}_i \to \bar{\ell}_j \notin R(\varphi)$

($\Rightarrow$) Suppose $r \notin R(\varphi)$ and $r \in R(\forall \ell_i \cdot \varphi)$. By $r \in R(\forall \ell_i \cdot \varphi)$ and Definition 6, $\omega \in M(\forall \ell_i \cdot \varphi)$ and $\omega[\bar{\ell}_j] \notin M(\forall \ell_i \cdot \varphi)$. By $\omega \in M(\forall \ell_i \cdot \varphi)$ and Theorem 2, $\omega \in M(\varphi)$. By $\omega \in M(\varphi)$, $r \notin R(\varphi)$ and Definition 6, $\omega[\bar{\ell}_j] \in M(\varphi)$. By $\omega[\bar{\ell}_j] \in M(\varphi)$, $\omega[\bar{\ell}_j] \notin M(\forall \ell_i \cdot \varphi)$ and Theorem 2, world $\omega^\star = \omega[\bar{\ell}_j]$ must be an $\bar{\ell}_i$-boundary model of φ; that is, $\bar{\ell}_i \in \omega^\star$ and $\omega^\star[\ell_i] \notin M(\varphi)$. By $\omega \in M(\varphi)$, $\omega \in M(\forall \ell_i \cdot \varphi)$ and Theorem 2, ω is not an $\bar{\ell}_i$-boundary model of φ and hence $\omega[\ell_i] \in M(\varphi)$. We now have

(1) $\alpha, \bar{\ell}_i, \ell_j = \omega \in M(\varphi)$,

(2) $\alpha, \ell_i, \ell_j = \omega[\ell_i] \in M(\varphi)$,

(3) $\alpha, \bar{\ell}_i, \bar{\ell}_j = \omega[\bar{\ell}_j] = \omega^\star \in M(\varphi)$ and

(4) $\alpha, \ell_i, \bar{\ell}_j = \omega^\star[\ell_i] \notin M(\varphi)$.

By Definition 6, (2) and (4) imply (A); (3) and (4) imply (B); (1) implies (C) and (D).

($\Leftarrow$) Suppose (A), (B), (C) and (D). By Definition 6, (A) implies (2) and (4); (B) implies (3) and (4); (C) and (2) imply (1); (D) and (3) imply (4). We have now established (1), (2), (3) and (4)—we only need (A) and (B) together with either (C) or (D) to establish this. By (3) and Definition 6, we get $r \notin R(\varphi)$. By (1) and (2), $\alpha, \bar{\ell}_i, \ell_j$ is not an $\bar{\ell}_i$-boundary model of φ and hence $\alpha, \bar{\ell}_i, \ell_j \in M(\forall \ell_i \cdot \varphi)$ by Theorem 2. By (3) and (4), $\alpha, \bar{\ell}_i, \bar{\ell}_j$ is an $\bar{\ell}_i$-boundary model of φ and hence $\alpha, \bar{\ell}_i, \bar{\ell}_j \notin M(\forall \ell_i \cdot \varphi)$ by Theorem 2. By $\omega = \alpha, \bar{\ell}_i, \ell_j \in M(\forall \ell_i \cdot \varphi)$, $\alpha, \bar{\ell}_i, \bar{\ell}_j \notin M(\forall \ell_i \cdot \varphi)$ and Definition 6, we get $r \in R(\forall \ell_i \cdot \varphi)$. $\square$

References

Audemard, G., Koriche, F., & Marquis, P. (2020). On tractable XAI queries based on compiled representations. In *Proc. of KR'20*, pp. 838–849.

Beame, P., & Liew, V. (2015). New limits for knowledge compilation and applications to exact model counting. In *Proc. of UAI'15*, pp. 131–140. UAI Press.

Bollig, B., & Buttkus, M. (2019). On the relative succinctness of sentential decision diagrams. *Theory Comput. Syst.*, *63*(6), 1250–1277.

Boole, G. (1854). *An investigation of the laws of thought*. Walton and Maberley, London.

Bryant, R. E. (1986). Graph-based algorithms for Boolean function manipulation. *IEEE Trans. Computers*, *35*(8), 677–691.

Chan, H., & Darwiche, A. (2003). Reasoning about Bayesian network classifiers. In *Proc. of UAI'03*, pp. 107–115.

Choi, A., Shih, A., Goyanka, A., & Darwiche, A. (2020). On symbolically encoding the behavior of random forests. In *Proc. of FoMLAS'20, 3rd Workshop on Formal Methods for ML-Enabled Autonomous Systems*.

Cimatti, A., Roveri, M., & Traverso, P. (1998). Strong planning in non-deterministic domains via model checking. In *Proc. of AIPS'98*, pp. 36–43.

Coste-Marquis, S., Berre, D. L., Letombe, F., & Marquis, P. (2006). Complexity results for quantified Boolean formulae based on complete propositional languages. *J. Satisf. Boolean Model. Comput.*, *1*(1), 61–88.

Darwiche, A. (2001). Decomposable negation normal form. *J. ACM*, *48*(4), 608–647.

Darwiche, A. (2011). SDD: A new canonical representation of propositional knowledge bases. In *Proc. of IJCAI'11*, pp. 819–826.

Darwiche, A., & Hirth, A. (2020). On the reasons behind decisions. In *Proc. of ECAI'20*, pp. 712–720.

Darwiche, A., & Marquis, P. (2002). A knowledge compilation map. *J. Artif. Intell. Res.*, *17*, 229–264.

Doherty, P., Lukasziewicz, W., & Madalińska-Bugaj, E. (1998). The PMA and relativizing change for action update. In *Proc. of KR'98*, pp. 258–269.

Doherty, P., Lukasziewicz, W., & Madalińska-Bugaj, E. (2000). The PMA and relativizing change for action update. *Fundamenta Informaticae*, *44*(1-2), 95–131.

Eén, N., & Biere, A. (2005). Effective preprocessing in SAT through variable and clause elimination. In *SAT*, Vol. 3569 of *Lecture Notes in Computer Science*, pp. 61–75. Springer.

Eiter, T., Ianni, G., Schindlauer, R., Tompits, H., & Wang, K. (2006). Forgetting in managing rules and ontologies. In *Proc. of WI'06, 2006 IEEE / WIC / ACM International Conference on Web Intelligence*, pp. 411–419.

Eiter, T., & Wang, K. (2006). Forgetting and conflict resolving in disjunctive logic programming. In *Proc. of AAAI'06*.

Eiter, T., & Wang, K. (2008). Semantic forgetting in answer set programming. *Artif. Intell.*, *172*(14), 1644–1672.

Eiter, T., & Kern-Isberner, G. (2019). A brief survey on forgetting from a knowledge representation and reasoning perspective. *Künstliche Intell.*, *33*(1), 9–33.

Fargier, H., & Marquis, P. (2014). Disjunctive closures for knowledge compilation. *Artif. Intell.*, *216*, 129–162.

Fredman, M. L., & Khachiyan, L. (1996). On the complexity of dualization of monotone disjunctive normal forms. *J. Algorithms*, *21*(3), 618–628.

Giunchiglia, E., Marin, P., & Narizzano, M. (2009). Reasoning with quantified Boolean formulas. In Biere, A., Heule, M., van Maaren, H., & Walsh, T. (Eds.), *Handbook of Satisfiability*, Vol. 185 of *Frontiers in Artificial Intelligence and Applications*, pp. 761–780. IOS Press.

Gurvich, V., & Khachiyan, L. (1999). On generating the irredundant conjunctive and disjunctive normal forms of monotone Boolean functions. *Discret. Appl. Math.*, *96-97*, 363–373.

Hagen, M., Horatschek, P., & Mundhenk, M. (2009). Experimental comparison of the two Fredman-Khachiyan-algorithms. In *In Proc. of ALENEX'09, 11th Workshop on Algorithm Engineering and Experiments*, pp. 154–161.

Herzig, A. (1996). The PMA revisited. In *Proc. of KR'96*, pp. 40–50.

Herzig, A., Lang, J., & Marquis, P. (2013). Propositional update operators based on formula/literal dependence. *ACM Transactions on Computational Logic*, *14*(3), 24.

Herzig, A., & Rifi, O. (1998). Update operations: a review. In *Proc. of ECAI'98*, pp. 13–17.

Herzig, A., & Rifi, O. (1999). Propositional belief update and minimal change. *Artif. Intell.*, *115*, 107–138.

Huang, J., & Darwiche, A. (2007). The language of search. *J. Artif. Intell. Res.*, *29*, 191–219.

Ignatiev, A., Narodytska, N., & Marques-Silva, J. (2019a). Abduction-based explanations for machine learning models. In *Proc. of AAAI'19*, pp. 1511–1519.

Ignatiev, A., Narodytska, N., & Marques-Silva, J. (2019b). On relating explanations and adversarial examples. In *Proc. of NeurIPS'19*, pp. 15857–15867.

Ignatiev, A., Narodytska, N., & Marques-Silva, J. (2019c). On validating, repairing and refining heuristic ML explanations. *CoRR*, *abs/1907.02509*.

Katsuno, H., & Mendelzon, A. O. (1989). A unified view of propositional knowledge base updates. In *Proc. of IJCAI'89*, pp. 1413–1419.

Khachiyan, L., Boros, E., Elbassioni, K. M., & Gurvich, V. (2006). An efficient implementation of a quasi-polynomial algorithm for generating hypergraph transversals and its application in joint generation. *Discret. Appl. Math.*, *154*(16), 2350–2372.

Kleine Büning, H., Karpinski, M., & Flögel, A. (1995). Resolution for quantified Boolean formulas. *Inf. Comput.*, *117*(1), 12–18.

Klieber, W., Janota, M., Marques-Silva, J., & Clarke, E. M. (2013). Solving QBF with free variables. In *Proc. of CP'13*, Vol. 8124, pp. 415–431.

Knaster, B. (1928). Un théorème sur les fonctions d'ensembles. *Ann. Soc. Polon. Math*, *6*, 133–134.

Lang, J., & Marquis, P. (2010). Reasoning under inconsistency: A forgetting-based approach. *Artif. Intell.*, *174*(12-13), 799–823.

Lang, J., Liberatore, P., & Marquis, P. (2003). Propositional independence: Formula-variable independence and forgetting. *J. Artif. Intell. Res.*, *18*, 391–443.

Lin, F., & Reiter, R. (1994). Forget it!. In *Proc. of AAAI Fall Symposium on Relevance*, pp. 154–159.

Miller, D. M., & Thornton, M. A. (2008). *Multiple Valued Logic: Concepts and Representations*, Vol. 12 of *Synthesis lectures on digital circuits and systems*. Morgan & Claypool Publishers.

Narodytska, N., Kasiviswanathan, S. P., Ryzhyk, L., Sagiv, M., & Walsh, T. (2018). Verifying properties of binarized deep neural networks. In *Proc. of AAAI'18*, pp. 6615–6624.

Papadimitriou, C. H. (1994). *Computational complexity*. Addison–Wesley.

Pipatsrisawat, K., & Darwiche, A. (2008). New compilation languages based on structured decomposability. In *Proc. of AAAI'08*, pp. 517–522.

Quine, W. V. (1955). A way to simplify truth functions. *American Mathematical Monthly*, *62*, 627–631.

Sedaghat, N., Stephen, T., & Chindelevitch, L. (2018). Speeding up dualization in the Fredman-Khachiyan algorithm B. In *In Proc of SEA'18, 17th International Symposium on Experimental Algorithms*, Vol. 103 of *LIPIcs*, pp. 6:1–6:13. Schloss Dagstuhl - Leibniz-Zentrum für Informatik.

Shi, W., Shih, A., Darwiche, A., & Choi, A. (2020). On tractable representations of binary neural networks. In *Proc. of KR'20*, pp. 882–892.

Shih, A., Choi, A., & Darwiche, A. (2018). A symbolic approach to explaining Bayesian network classifiers. In *Proc. of IJCAI'18*, pp. 5103–5111.

Shih, A., Choi, A., & Darwiche, A. (2019). Compiling Bayesian network classifiers into decision graphs. In *Proc. of AAAI'19*, pp. 7966–7974.

Shukla, A., Biere, A., Pulina, L., & Seidl, M. (2019). A survey on applications of quantified Boolean formulas. In *Proc. of ICTAI'19*, pp. 78–84.

Stockmeyer, L. J. (1977). The polynomial-time hierarchy. *Theoretical Computer Science*, *3*, 1–22.

Subbarayan, S., & Pradhan, D. K. (2005). NiVER: non-increasing variable elimination resolution for preprocessing sat instances. In Hoos, H. H., & Mitchell, D. G. (Eds.), *Theory and Applications of Satisfiability Testing*, pp. 276–291, Berlin, Heidelberg. Springer Berlin Heidelberg.

Tarski, A. (1955). A lattice-theoretical fixpoint theorem and its applications. *Pacific Journal of Mathematics*, *5*(2), 285–309.

Wang, K., Sattar, A., & Su, K. (2005). A theory of forgetting in logic programming. In *Proc. of AAAI'05*, pp. 682–688.

Weber, A. (1986). Updating propositional formulas. In *Proc. of EDS'86, 1st Int. Conf. on Expert Database Systems*, pp. 487–500.

Winslett, M. (1990). *Updating Logical Databases*. Cambridge University Press, Cambridge, England.

Zhang, Y., & Foo, N. Y. (2006). Solving logic program conflict through strong and weak forgettings. *Artif. Intell.*, *170*(8-9), 739–778.

Zhang, Y., Foo, N. Y., & Wang, K. (2005). Solving logic program conflict through strong and weak forgettings. In *Proc. of IJCAI'05*, pp. 627–634.