Disjunctive Closures for Knowledge Compilation

Hélène Fargier\textsuperscript{a}, Pierre Marquis\textsuperscript{b}
\textsuperscript{a}IRIT - CNRS / Université Paul Sabatier, Toulouse - France
fargier@irit.fr
\textsuperscript{b}CRIL - CNRS / Université d’Artois, Lens - France
marquis@cril.univ-artois.fr

Abstract

The knowledge compilation (KC) map [1] can be viewed as a multi-criteria evaluation of a number of target classes of representations for propositional KC. Using this map, the choice of a class for a given application can be made, considering both the space efficiency of it (i.e., its ability to represent information using little space), and its time efficiency, i.e., the queries and transformations which can be achieved in polynomial time, among those of interest for the application under consideration. When no class of propositional representations offers all the transformations one would expect, some of them can be left implicit. This is the key idea underlying the concept of closure introduced here: instead of performing computationally expensive transformations, one just remembers that they have to be done. In this paper, we investigate the disjunctive closure principles, i.e., disjunction, existential quantification, and their combinations. We provide several characterization results concerning the corresponding closures. We also extend the KC map with new propositional languages obtained as disjunctive closures of several incomplete propositional languages, including the well-known KROM (the CNF formulae containing only binary clauses), HORN (the CNF formulae containing only Horn clauses), and AFF (the affine language, which is the set of conjunctions of XOR-clauses). Each introduced language is evaluated along the lines of the KC map.

Keywords: Knowledge representation, knowledge compilation, computational complexity.

\textsuperscript{*}This paper is an extended and revised version of [2, 3]. It also elaborates on some of the results presented in [4].
1. Introduction

Knowledge compilation (KC) consists of a family of approaches which aim at improving the efficiency of some computational tasks – typically, saving online computation time – via pre-processing. The pre-processing step consists in turning some pieces of available information into a compiled form, during an offline compilation phase.

KC gathers a number of research lines focusing on different problems, [5, 6, 7, 8, 9, 10, 11, 12]), ranging from theoretical ones, where the key question is the compilability issue, i.e., determining whether pre-processing can lower the computational complexity of some tasks, to more practical ones, especially the design of compilation algorithms for some specific tasks like clausal entailment. An important research line [13, 1] is concerned with the issue of choosing a target class of representations for KC. In [1], the authors argue that the choice of a target class for a compilation purpose must be based both on its time efficiency, defined as the set of queries and transformations which can be achieved in polynomial time when the pieces of data to be exploited are represented in the class, as well as the space efficiency of the class, i.e., its ability to represent data using little space. Thus, the KC map [1] is an evaluation of a dozen of significant propositional classes \( \mathcal{L} \), also called propositional fragments, w.r.t. several dimensions: the space efficiency (aka succinctness) of the fragment and its time efficiency (aka tractability), i.e., the class of queries and transformations it supports (or not) in polynomial time, often under standard assumptions from complexity theory. The KC map is intended to serve as a guide for selecting the "right" target class given the requirements imposed by the application under consideration.

The KC map reported in [1] has already been extended to other propositional classes, queries and transformations in a number of subsequent papers, including [14, 15, 16, 17, 18, 2, 4, 19, 20, 21, 3, 22]. In all those papers, queries and transformations are also viewed as properties of classes of propositional representations \( \mathcal{L} \). One says that \( \mathcal{L} \) offers or satisfies a given query or a transformation when there exists a polynomial-time algorithm to achieve it, provided that the input representations are from \( \mathcal{L} \). When such an algorithm does not exist for sure or unless \( P = NP \), one says that \( \mathcal{L} \) does not offer the query or the transformation.

When no class of propositional representations offers all the transformations one would expect, an approach consists in leaving some of them implicit. This is the key idea underlying closure principles as introduced in this paper: instead of performing some computationally expensive transformations on representations, one just remembers in the representations that the transformations have to be done.
This leads to extend the previous classes to new ones, which are at least as succinct, and for which implicit transformations are for free. Another nice effect of some implicit transformations on incomplete propositional languages is to recover completeness, i.e., the ability to represent any Boolean function.

In this paper, we investigate the disjunctive closure principles, i.e., disjunction $[\lor]$, existential quantification $[\exists]$, and their combinations. The disjunction principle $[\lor]$ when applied to a class $L$ of representations leads to a class $L[\lor]$, the disjunction closure of $L$, which qualifies disjunctions of representations from $L$, while the existential quantification principle $[\exists]$ applied to a class $L$ leads to a class $L[\exists]$, the existential closure of $L$, which qualifies existentially quantified representations from $L$. $L[\lor, \exists]$, the full disjunctive closure of $L$, is obtained by applying both disjunctive closure principles to $L$. We provide a number of characterization results concerning the corresponding closures. Especially, we show that applying at most once each disjunctive closure principle on $L$ is enough, in the sense that applying one of them twice or more leads to classes polynomially equivalent to $L$. We also identify the queries and transformations which are preserved by applying disjunctive closure principles.

In addition, we extend the KC map with new classes of propositional representations obtained as disjunctive closures of several incomplete propositional languages, namely the well-known Krom CNF fragment $KROM$ (also known as the bijunctive fragment) [23] the Horn CNF fragment $HORN$ [24], and the affine fragment $AFF$ (also known as the biconditional fragment) [25], as well as $K/H$ (Krom or Horn CNF formulae) and $renH$, the language of renamable Horn CNF formulae [26]. Each of these languages is a well-known polynomial class for the satisfiability problem $SAT$ (i.e., it offers $CO$), but none of them is fully expressive w.r.t. propositional logic (there exist propositional formulae which cannot be represented in any of them), which drastically restricts their attractiveness for the KC purpose. Importantly, switching from any of those languages to its disjunction closure or to its full disjunctive closure leads to recover a fully expressive propositional language. This is crucial for many applications.

The rest of the paper is organized as follows. In Section 2, some formal preliminaries about graph-based, quantified, propositional representations are provided. In Section 3, we make precise the queries and transformations of interest, and extend the notions of expressiveness, succinctness and polynomial translations to any subsets of the class of graph-based, quantified, propositional representations. In Section 4, the concepts of disjunctive closures of a class of propositional representations are defined and we derive a number of characterization results about them. In Section 5, the disjunctive closures of $KROM$, $HORN$, $K/H$, $renH$, and
AFF are considered and we analyze them along the lines of the KC map. Finally, Section 6 concludes the paper by discussing the results, pointing out the disjunctive closures which appear as the best target classes for the KC purpose; it also gives some perspectives for further research.

2. Quantified Propositional Representations

2.1. Syntax

In this paper, we consider subsets of the class $C\overline{\text{QDAG}}$ of quantified propositional representations over a countably infinite set $PS$ of propositional variables, given a finite set $C$ of propositional connectives. Each connective $c \in C$ is supposed to have a fixed, finite arity. Leaf nodes of such DAGs are labeled by literals, where a literal (over $V \subseteq PS$) is an element $x \in V$ (a positive literal) or a negated one $\neg x$ (a negative literal), or a Boolean constant ($\top$ and $\bot$). $L_V$ is the set of all literals over $V$. Literal $l$ is the complementary literal of literal $\overline{l}$, so that $\top = \bot$, $\bot = \top$, $x = \neg x$ and $\neg \neg x = x$. For a literal $l$ different from a Boolean constant, $\text{var}(l)$ denotes the corresponding variable: for $x \in PS$, we have $\text{var}(x) = x$ and $\text{var}(\neg x) = x$.

Formally, $C\overline{\text{QDAG}}$ is given by:

**Definition 1 ($C\overline{\text{QDAG}}$).** $C\overline{\text{QDAG}}$ is the set of all finite, single-rooted DAGs (also referred to as "representations") $\alpha$ where:

- each leaf node of $\alpha$ is labeled by a literal $l$ over $PS$,
- each internal node of $\alpha$ is labeled by a connective $c \in C$ and has as many children as required by $c$ (it is then called a $c$-node), or is labeled by a quantification $\exists x$ or $\forall x$ (where $x \in PS$) and has a single child.$^1$

The size $|\alpha|$ of a $C\overline{\text{QDAG}}$ representation $\alpha$ is the number of nodes plus the number of arcs in the DAG. $\text{Var}(\alpha)$ denotes the set of free variables of $\alpha$, i.e., those variables $x$ for which there exists a leaf node $N_x$ of $\alpha$ labelled by a literal $l$ such that $\text{var}(l) = x$ and there is a path from the root of $\alpha$ to $N_x$ such that no node from it is labelled by $\exists x$ or $\forall x$. Clearly enough, determining whether a given $x \in$

$^1$Each binary connective $c$ which is associative (like $\land$, $\lor$, $\oplus$) corresponds to a family of connectives (with the same name $c$) of arity $i$ with $i \geq 2$. For each $i \geq 2$, the connective $c$ of arity $i$ is defined by: for every $i$-tuple $(x_1, \ldots, x_i)$ of Boolean values, $c(x_1, \ldots, x_i) = c(x_1, c(x_2, c(\ldots, c(x_{i-1}, x_i)) \ldots))$. 

PS belongs to $\text{Var}(\alpha)$ can be done in time polynomial in the size of $\alpha$; similarly, computing $\text{Var}(\alpha)$ can also be achieved in time polynomial in the size of $\alpha$.

Figure 1: A $C$-QDAG representation with $C = \{\land, \lor, \neg, \oplus\}$. Its set of free variables is $\text{Var}(\alpha) = \{q, r\}$.

As Figure 1 exemplifies it, a $C$-QDAG mainly corresponds to a Quantified Boolean Circuit [27]. Abusing words, such DAG-based representations are also referred to as "formulae" in the KC literature, and classes of such representations are called "languages". In the following, we will only use the term "formula" for designating a tree-shaped representation of a Boolean function, and the term "language" for sets of formulae. Figure 2 gives a $C$-QDAG formula with $C = \{\land, \lor, \neg\}$.

Many classes of propositional representations considered so far as target classes for KC are subsets of $C$-QDAG with $C = \{\land, \lor, \neg, \oplus\}$, and typically subsets of $C$-DAG, the subset of $C$-QDAG with $C = \{\land, \lor, \neg\}$ where no node labeled by a quantification is allowed. Especially, the propositional DAGs considered in [14] are $C$-DAG representations with $C = \{\land, \lor, \neg\}$, and the classes considered in [1]

---

2 The algorithm consists in labeling each node $N$ of $\alpha$ by a set of variables $V_N$; the nodes are considered in inverse topological ordering, $V_N = \text{var}(l)$ when $N$ is a leaf node labeled by $l$, $V_N = V_M \setminus \{x\}$ when $N$ is an internal node labeled by $\exists x$ or $\forall x$ and $M$ is the child of $N$, $V_N = \cup_M \text{child of } V_N$ when $N$ is an internal node labeled by a connective $c \in C$; $\text{Var}(\alpha)$ is equal to $V_{\alpha,\alpha}$ where $\alpha$ is the root of $\alpha$. 
are subsets of $\text{DAG-NNF}$ (the non-quantified DAGs with $C = \{\land, \lor\}$). Clearly enough, for each non-quantified representation $\alpha$ from $C-$DAG, $\text{Var}(\alpha)$ coincides with the set of variables occurring in $\alpha$.

In Figure 1, the DAG rooted at the $\land$ node is a $C-$DAG representation with $C = \{\land, \lor, \lnot\}$ and the DAG rooted at the $\lor$ node is a $\text{DAG-NNF}$ representation. $\text{DNNF}$ is the subset of $\text{DAG-NNF}$ consisting of DAGs where each $\land$-node $\land(\alpha_1, \ldots, \alpha_k)$ is decomposable, which means that $\forall i, j \in \{1, \ldots, k\}$, if $i \neq j$ then $\text{Var}(\alpha_i) \cap \text{Var}(\alpha_j) = \emptyset$. $d-$DNNF is the subset of $\text{DNNF}$ where every $\lor$-node $\lor(\alpha_1, \ldots, \alpha_k)$ is deterministic, which means that $\forall i, j \in \{1, \ldots, k\}$, if $i \neq j$ then $\alpha_i \land \alpha_j$ is inconsistent. $\text{BDD}$ is the subset of $C-$DAG with $C = \{\text{ite}\}$ which consists of DAGs $\alpha$ such that every leaf node is labelled by a Boolean constant, $\top$ or $\bot$. $\text{ite}$ is a ternary connective ("ite" stands for "if ... then ... else ..."). Usually, instead of labeling a decision node $N = (x, N_+, N_-)$ of a $\text{BDD}$ formula by the name of the connective used (i.e., "ite") and considering three children for it (one for $x$, one for $N_+$ and one for $N_-$), $N$ is labelled by $x$ and has only two children (one for $N_+$ and one for $N_-$. Given a total, strict ordering $<$ over $PS$, the class $\text{OBDD}_<$ is the subset of $\text{BDD}$ which consists of DAGs $\alpha$ such that every path from the root of $\alpha$ to a leaf node is compatible with $<$.

As usual, a clause (resp. a term) is a finite disjunction (resp. conjunction) of literals. $\text{CLAUSE}$ is the subset of $\text{DAG-NNF}$ consisting of all clauses, and $\text{TERM}$ is the subset of $\text{DAG-NNF}$ consisting of all terms. $\text{NNF}$ is the subset of
DAG-NNF consisting of formulae (i.e., tree-shaped representations). CNF is the subset of NNF consisting of all conjunctions of clauses, while DNF is the subset of NNF consisting of all disjunctions of terms. PI is the subset of CNF consisting of prime implicates formulae (also known as Blake formulae); a PI formula is a CNF formula, the conjunction of all clauses from the set \( PI(\alpha) \) for some \( C-QDAG \) representation \( \alpha \); \( PI(\alpha) \) contains the prime implicates of \( \alpha \), i.e., the logically strongest clauses which are implied by \( \alpha \) (one representative per equivalence class is considered, only). An essential prime implicate of \( \alpha \) is a prime implicate \( \delta \) of \( \alpha \) such that if the clause equivalent to \( \delta \) is removed from \( PI(\alpha) \), the conjunction of the clauses from the resulting set is no longer equivalent to \( \alpha \). For instance, if \( \alpha = (p \Rightarrow q) \land (q \Rightarrow r) \land (p \Rightarrow (r \lor s)) \), then \( PI(\alpha) = \{ \neg p \lor q, \neg q \lor r, \neg p \lor r \} \). \( \neg p \lor q \) and \( \neg q \lor r \) are essential prime implicates of \( \alpha \), while \( \neg p \lor r \) is not. An important point is that any CNF formula equivalent to a propositional representation \( \alpha \) contains (up to logical equivalence) every essential prime implicate of \( \alpha \).

For space reasons, we do not provide hereafter the definitions of the propositional classes of representations DNNF\(_T\) and IP (see [1, 18] for formal definitions).

### 2.2. Semantics

Let us recall that an interpretation (or world) over \( V \subseteq PS \) is a mapping \( \omega \) from \( V \) to \( BOOL = \{0, 1\} \). Interpretations are sometimes viewed as subsets of \( PS \), consisting of all the variables that are set to 1 by the interpretations. When a total, strict ordering \( < \) over \( PS \) is considered, the restriction of an interpretation \( \omega \) over a finite subset \( \{x_1, \ldots, x_n\} \) of \( PS \) can also be represented as a bit vector; for instance, the restriction of \( \omega \) over \( \{a, b, c\} \) such that \( \omega(a) = 1, \omega(b) = 0, \) and \( \omega(c) = 0 \) can be represented as 100 when \( a, b, c \) are such that \( a < b < c \). For any \( x \in V \), \( \omega_{-x} \) is the interpretation over \( V \) which coincides with \( \omega \) on every variable of \( V \), except on \( x \); formally, \( \omega_{-x}(y) = \omega(y) \) if \( y \neq x \), \( = 1 - \omega(x) \) if \( y = x \).

We are now ready to define the semantics of \( C-QDAG \) representations in an interpretation \( \omega \) over \( PS \):

**Definition 2 (semantics of \( C-QDAG \) representations).** The semantics of a \( C-QDAG \) representation \( \alpha \) in an interpretation \( \omega \) over \( PS \) is the truth value \( [\alpha](\omega) \) from \( BOOL \) defined inductively as follows:

- if \( \alpha = \top \), then \( [\alpha](\omega) = 1 \).
- if \( \alpha = \bot \), then \( [\alpha](\omega) = 0 \).
- if \( \alpha \) is a positive literal \( x \), then \( [\alpha](\omega) = \omega(x) \).
• if \( \alpha \) is a negative literal \( \neg x \), then \( \llbracket \alpha \rrbracket(\omega) = 1 - \omega(x) \).

• if \( \alpha = c(\beta_1, \ldots, \beta_n) \), where \( c \in C \) has arity \( n \), then \( \llbracket \alpha \rrbracket(\omega) = \llbracket c \rrbracket(\llbracket \beta_1 \rrbracket(\omega), \ldots, \llbracket \beta_n \rrbracket(\omega)) \), where \( \llbracket c \rrbracket \) is the Boolean function from \( \text{BOOL}^n \) to \( \text{BOOL} \), which is the semantics of \( c \).

• if \( \alpha = \exists x.\beta \), then \( \llbracket \alpha \rrbracket(\omega) = 1 \) iff \( \llbracket \beta \rrbracket(\omega) = 1 \) or \( \llbracket \beta \rrbracket(\omega{-}x) = 1 \).

• if \( \alpha = \forall x.\beta \), then \( \llbracket \alpha \rrbracket(\omega) = 1 \) iff \( \llbracket \beta \rrbracket(\omega) = 1 \) and \( \llbracket \beta \rrbracket(\omega{-}x) = 1 \).

An interpretation \( \omega \) over \( PS \) is said to be a model of \( \alpha \in C{-}\text{QDAG} \), noted \( \omega \models \alpha \), if and only if \( \llbracket \alpha \rrbracket(\omega) = 1 \). If \( \alpha \) has a model, then it is consistent; if every interpretation over \( PS \) is a model of \( \alpha \), then \( \alpha \) is valid. \( \text{Mod}(\alpha) \) denotes the set of models of \( \alpha \) over \( \text{Var}(\alpha) \). Furthermore, when both \( \alpha \models \beta \) and \( \beta \models \alpha \) hold, \( \alpha \) and \( \beta \) are logically equivalent, noted \( \alpha \equiv \beta \).

For instance, with \( C = \{\land, \lor, \neg, \oplus\} \), the \( C{-}\text{QDAG} \) representation given in Figure 1 is equivalent to the \( C{-}\text{QDAG} \) formula given in Figure 2.

By structural induction one can easily show that the semantics of any \( C{-}\text{QDAG} \) representation \( \alpha \) depends only on its free variables, in the sense that, for any interpretation \( \omega' \) over \( PS \) which coincides with a given interpretation \( \omega \) on all the free variables of \( \alpha \), \( \omega \) is a model of \( \alpha \) if and only if \( \omega' \) is a model of \( \alpha \). Accordingly, the semantics of a \( C{-}\text{QDAG} \) representation \( \alpha \) in an interpretation \( \omega \) over \( PS \) is fully determined by \( \alpha \) and the restriction of \( \omega \) over \( \text{Var}(\alpha) \).

Clearly enough, renaming at the same time a quantified occurrence of a variable \( x \) in a quantification \( \exists x.\beta \) or \( \forall x.\beta \) occurring in a \( C{-}\text{QDAG} \) formula \( \alpha \), and every occurrence of \( x \) in \( \alpha \) which depends on the quantification leads to a \( C{-}\text{QDAG} \) formula equivalent to \( \alpha \). Furthermore, such a renaming process can be achieved in time linear in the size of \( \alpha \).

However, things are much more tricky when general \( C{-}\text{QDAG} \) representations (not reduced to formulae) are considered. Consider for instance the quantification \( \exists q \) occurring in the \( C{-}\text{QDAG} \) representation \( \alpha \) reported at Figure 1, where \( C = \{\land, \lor, \neg, \oplus\} \). The occurrence of variable \( q \) in the leaf of \( \alpha \) labelled with literal \( q \) depends on this quantification. Replacing \( q \) by the fresh variable \( s \) in \( \exists q \) and at this occurrence would not lead to a representation equivalent to \( \alpha \) since \( s \) would be a free variable of the resulting representation. Indeed, there exist four paths from the root of \( \alpha \) to that leaf, and three of them do not contain any quantified occurrence of \( q \). This is salient on the \( C{-}\text{QDAG} \) formula equivalent to \( \alpha \) reported...
at Figure 2, and obtained by "unfolding" $\alpha$. Thus, when some variable occurrence can be both free and bound, renaming quantified variables while preserving equivalence can be a computationally demanding task (the unfolding process may easily lead to an exponential blow-up of the input representation). Actually, when $C \supseteq \{\wedge, \vee\}$, the possibility of having some variable occurrences both free and bound (or to depend on different existential quantifications) in $C$-$QDAG$ representations not containing universal quantifications is enough to simulate universal quantifications in them (see [27]). As a consequence, the corresponding class of DAGs is strictly more succinct than the corresponding language of formulae. On the other hand, some problems are computationally easier when formulae (and not DAGs) are considered; for instance, when universal quantifications are disabled, the model checking problem for $C$-$QDAG$ formulae with $C \supseteq \{\wedge, \vee\}$ is "only" NP-complete, while it is PSPACE-complete when the full class of $C$-$QDAG$ representations without universal quantifications is considered.

Conventionally, the representation $\alpha_N$ rooted at a decision node $N = \langle x, N_+, N_- \rangle$ over $x \in PS$ in the standard representation of an ordered binary decision diagram (i.e., an OBDD$<$ representation) is such that $\alpha_N \equiv \text{ite}(x, \alpha_{N_+}, \alpha_{N_-}) \equiv (x \land \alpha_{N_+}) \lor (\neg x \land \alpha_{N_-})$. $\alpha_{N_+}$ (resp. $\alpha_{N_-}$), the representation associated with node $N_+$ (resp. $N_-$), is the conditioning of $\alpha$ by $x$ (resp. $\neg x$), i.e., the representation obtained by replacing every occurrence of $x$ in $\alpha_N$ by $\top$ (resp. $\bot$).

Finally, we consider the following notations. If $\alpha \in C$-$QDAG$ and $X = \{x_1, \ldots, x_n\} \subseteq PS$, then $\exists X. \alpha$ is a short for

$$\exists x_1.(\exists x_2.(\ldots \exists x_n.\alpha)\ldots)$$

and $\forall X. \alpha$ is a short for

$$\forall x_1.(\forall x_2.(\ldots \forall x_n.\alpha)\ldots)$$

(these notations are well-founded since whatever the chosen ordering on $X$, the resulting representations are logically equivalent).

2.3. KROM, HORN, AFF, K/H, and renH

In the following, we will focus on several well-known propositional languages, namely the Krom CNF language KROM (also known as the bijunctive fragment) [23], the Horn CNF language HORN [24], and the affine language AFF (also known as the biconditional fragment) [25], as well as K/H (Krom or Horn CNF formulae) and renH, the language of renamable Horn CNF formulae [26].

The languages KROM, HORN, AFF, K/H, and renH are formally defined as follows:
Definition 3 (KROM, HORN, AFF, K/H, and renH).

- The language KROM is the subset of all CNF formulae in which each clause is binary, i.e., it contains at most two literals.
- The language HORN is the subset of all CNF formulae in which each clause is Horn, i.e., it contains at most one positive literal.
- The language K/H is the union of KROM and HORN.
- The language renH is the subset of all CNF formulae \( \alpha \) for which there exists a subset \( V \) of \( \text{Var}(\alpha) \) (called a Horn renaming for \( \alpha \)) such that the formula noted \( V(\alpha) \) obtained by substituting in \( \alpha \) every literal \( l \) of \( L_V \) by its complementary literal \( \overline{l} \) is a HORN formula.
- The language AFF is the subset of \( \mathcal{CQDAG} \) with \( \mathcal{C} = \{\land, \neg, \oplus\} \), consisting of conjunctions of XOR-clauses where a XOR-clause is a finite XOR-disjunction of literals (the XOR connective is denoted by \( \oplus \)).

Here are some examples of formulae from KROM, HORN, renH, and AFF:

- \((x \lor y) \land (\neg y \lor z)\) is a KROM formula.
- \((\neg x \lor \neg y \lor z) \land (\neg y \lor z)\) is a HORN formula.
- \((x \lor y \lor z) \land (\neg x \lor \neg y \lor z)\) is a renH formula.
  \(V = \{x, z\}\) is a Horn renaming for it.
- \((x \oplus \neg y \oplus \top) \land (\neg x \oplus y \oplus z)\) is an AFF formula.

Clearly enough, determining whether a given \( \mathcal{CQDAG} \) representation \( \alpha \) (for any fixed \( \mathcal{C} \)) is a KROM (resp. HORN, K/H, AFF) formula can be easily achieved in time polynomial in the size of \( \alpha \). Note also that there exists linear time algorithms for recognizing renH formulae (see e.g. [28, 29]); furthermore, such recognition algorithms typically give a Horn renaming when it exists.

KROM, HORN, AFF, K/H, and renH are known as polynomial classes for the SAT problem (i.e., the restriction of SAT to any of them is in polynomial time – stated otherwise, each of them satisfies \( \text{CO} \)). However, none of them is fully expressive w.r.t. propositional logic (there exist propositional formulae, like \((x \lor y \lor z) \land (\neg x \lor \neg y \lor \neg z)\), which cannot be represented in any of them); this severely restricts their attractiveness for the KC purpose.

Interestingly, KROM, HORN, and AFF have semantical characterizations in terms of closures of sets of models:
A set \( S \) of interpretations over a finite \( V \subseteq PS \) is the set of models of a KROM formula \( \alpha \) such that \( \text{Var}(\alpha) = V \) if and only if it is closed for majority [30, 25], i.e., \( \forall \omega_1, \omega_2, \omega_3 \in S \), the interpretation \( \text{maj}(\omega_1, \omega_2, \omega_3) \) over \( V \) belongs to \( S \) as well. Here \( \text{maj}(\omega_1, \omega_2, \omega_3) \) is defined by \( \forall x \in V, \text{maj}(\omega_1, \omega_2, \omega_3)(x) = 1 \) if at least two interpretations among \( \omega_1, \omega_2, \omega_3 \) give the truth value 1 to \( x \).

A set \( S \) of interpretations over a finite \( V \subseteq PS \) is the set of models of a HORN formula \( \alpha \) such that \( \text{Var}(\alpha) = V \) if and only if it is closed for the bitwise AND connective [31, 32], i.e., \( \forall \omega_1, \omega_2 \in S \), the interpretation \( \text{and}(\omega_1, \omega_2) \) over \( V \) belongs to \( S \). Here \( \text{and}(\omega_1, \omega_2) \) is defined by \( \forall x \in V, \text{and}(\omega_1, \omega_2)(x) = 1 \) if \( \omega_1(x) = \omega_2(x) = 1 \).

A set \( S \) of interpretations over a finite \( V \subseteq PS \) is the set of models of an AFF formula \( \alpha \) such that \( \text{Var}(\alpha) = V \) if and only if \( S \) is closed for the ternary \( \oplus \) connective [30, 25], i.e., \( \forall \omega_1, \omega_2, \omega_3 \in S \), the interpretation \( \oplus(\omega_1, \omega_2, \omega_3) \) over \( V \) belongs to \( S \). Here \( \oplus(\omega_1, \omega_2, \omega_3) \) is defined by \( \forall x \in V, \oplus(\omega_1, \omega_2, \omega_3)(x) = \omega_1(x) \oplus \omega_2(x) \oplus \omega_3(x) \).

These characterization results can be exploited to show that some propositional formulae cannot be expressed as KROM (resp. HORN, AFF) formulae.

3. Queries, Transformations, Expressiveness and Succinctness

Let us now briefly recall the sets of queries and transformations used for comparing propositional languages in [1], as well as the notions of expressiveness and succinctness; their importance is discussed in depth in [1], so we refrain from recalling it here.

3.1. Queries

The basic queries considered in [1] and subsequent papers concern \( \text{DAG-NNF} \) representations; they include tests for consistency (CO), validity (VA), clausal entailment (CE), implicants (IM), equivalence (EQ), sentential entailment (SE), counting (CT) and enumerating theory models (ME). We extend them to \( C-\text{QDAG} \) representations and add to them MC, the model checking query, which is trivially offered by unquantified representations, but not by quantified representations in the general case.

**Definition 4 (queries).** Let \( \mathcal{L} \) denote any subset of \( C-\text{QDAG} \).
• \( \mathcal{L} \) satisfies \textbf{CO}, the consistency query (resp. \textbf{VA}, the validity query) if there exists a polynomial-time algorithm that maps every representation \( \alpha \) from \( \mathcal{L} \) to 1 if \( \alpha \) is consistent (resp. valid), and to 0 otherwise.

• \( \mathcal{L} \) satisfies \textbf{CE}, the clausal entailment query, if there exists a polynomial-time algorithm that maps every pair \( (\alpha, \delta) \), where \( \alpha \) is a representation from \( \mathcal{L} \) and \( \delta \) is a clause, to 1 if \( \alpha \models \delta \) holds, and to 0 otherwise.

• \( \mathcal{L} \) satisfies \textbf{EQ}, the equivalence query (resp. \textbf{SE}, the sentential entailment query) if there exists a polynomial-time algorithm that maps every pair \( (\alpha, \beta) \) of representations from \( \mathcal{L} \) to 1 if \( \alpha \equiv \beta \) (resp. \( \alpha \models \beta \)) holds, and to 0 otherwise.

• \( \mathcal{L} \) satisfies \textbf{IM}, the implicant query, if there exists a polynomial-time algorithm that maps every pair \( (\alpha, \gamma) \), where \( \alpha \) is a representation from \( \mathcal{L} \) and \( \gamma \) is a term, to 1 if \( \gamma \models \alpha \) holds, and to 0 otherwise.

• \( \mathcal{L} \) satisfies \textbf{CT}, the model counting query, if there exists a polynomial-time algorithm that maps every representation \( \alpha \) from \( \mathcal{L} \) to a nonnegative integer that represents the number of models of \( \alpha \) over \( \text{Var}(\alpha) \) (in binary notation).

• \( \mathcal{L} \) satisfies \textbf{ME}, the model enumeration query, if there exists a polynomial \( p(\ldots) \) and an algorithm that outputs all models of an arbitrary representation \( \alpha \) from \( \mathcal{L} \) in time \( p(n, m) \), where \( n \) is the size of \( \alpha \) and \( m \) is the number of its models over \( \text{Var}(\alpha) \).

• \( \mathcal{L} \) satisfies \textbf{MC}, the model checking query, if there exists a polynomial-time algorithm that maps every pair \( (\alpha, \omega) \), where \( \alpha \) is a representation from \( \mathcal{L} \) and \( \omega \) is an interpretation over \( \text{Var}(\alpha) \), to 1 if \( I \) is a model of \( \alpha \), and to 0 otherwise.

3.2. Transformations

The basic transformations considered in [1] are conditioning (\textbf{CD}), (possibly bounded) closures under the connectives\(^3\) \( \land, \lor \), and \( \neg (\land C, \land BC, \lor C, \lor BC, \neg C) \) and (possibly bounded) forgetting which can be viewed as a closure operation under existential quantification (\textbf{FO}, \textbf{SFO}). Forgetting is an important transformation as it allows us to focus/project a representation on a set of variables, which

\(^3\)Closures under other connectives could also be easily defined but seem to be less significant.
proves helpful in many applications, including model-based diagnosis [33], reasoning about actions [34], and reasoning under inconsistency [35, 36]. All those transformations concern DAG–NNF representations. We extend them to C–QDAG representations and enrich the list with two additional transformations, which are dual to (FO, SFO), namely ”ensuring” (EN) and the bounded restriction of it (SEN). Ensuring amounts to eliminating universal quantifications and allows us to project a representation on a set of variables in a robust way, i.e., independently of the values of the removed variables. This transformation is central in decision making under uncertainty and non-deterministic planning, see e.g. [37].

Definition 5 (transformations). Let \( \mathcal{L} \) denote any subset of C–QDAG.

- \( \mathcal{L} \) satisfies CD, the conditioning transformation, if there exists a polynomial-time algorithm that maps every pair \( \langle \alpha, \gamma \rangle \), where \( \alpha \) is a representation from \( \mathcal{L} \) and \( \gamma \) is a consistent term, to a representation from \( \mathcal{L} \) that is logically equivalent to \( \exists \text{Var}(\gamma). (\alpha \land \gamma) \).

- \( \mathcal{L} \) satisfies FO, the forgetting transformation, if there exists a polynomial-time algorithm that maps every pair \( \langle \alpha, X \rangle \), where \( \alpha \) is a representation from \( \mathcal{L} \) and \( X \) is a set of variables from PS, to a representation from \( \mathcal{L} \) equivalent to \( \exists X. \alpha \). If the property holds for each singleton \( X \), we say that \( \mathcal{L} \) satisfies SFO (singleton forgetting).

- \( \mathcal{L} \) satisfies EN, the ensuring transformation, if there exists a polynomial-time algorithm that maps every pair \( \langle \alpha, X \rangle \), where \( \alpha \) is a representation from \( \mathcal{L} \) and \( X \) is a set of variables from PS, to a representation from \( \mathcal{L} \) equivalent to \( \forall X. \alpha \). If the property holds for each singleton \( X \), we say that \( \mathcal{L} \) satisfies SEN (singleton ensuring).

- \( \mathcal{L} \) satisfies \( \land C \), the closure under conjunction transformation (resp. \( \lor C \), the closure under disjunction transformation) if there exists a polynomial-time algorithm that maps every finite set of representations \( \alpha_1, \ldots, \alpha_n \) from \( \mathcal{L} \) to a representation of \( \mathcal{L} \) that is equivalent to \( \alpha_1 \land \ldots \land \alpha_n \) (resp. \( \alpha_1 \lor \ldots \lor \alpha_n \)).

- \( \mathcal{L} \) satisfies \( \land BC \), the bounded closure under conjunction transformation (resp. \( \lor BC \), the bounded closure under disjunction transformation), if there exists a polynomial-time algorithm that maps every pair of representations \( \alpha \) and \( \beta \) from \( \mathcal{L} \) to a representation of \( \mathcal{L} \) that is equivalent to \( \alpha \land \beta \) (resp. \( \alpha \lor \beta \)).
• \( L \) satisfies \( \neg C \), the closure under negation transformation, if there exists a polynomial-time algorithm that maps every representation \( \alpha \) from \( L \) to a representation of \( L \) which is equivalent to \( \neg \alpha \).

When \( \alpha \) is a \( C\text{-DAG} \) representation (i.e., a non-quantified representation), the conditioning of \( \alpha \) by \( \gamma \) can be defined in an equivalent, yet simpler way, as the representation \( \alpha|_{\gamma} \) obtained by replacing in \( \alpha \) every occurrence of variable \( x \) by \( \top \) (resp. \( \bot \)) when \( x \) (resp. \( \neg x \)) is a literal of \( \gamma \). Such a characterization cannot be extended to \( C\text{-QDAG} \) representations in the general case. Especially, considering only those variables \( x \) occurring free in \( \alpha \) as candidates for the replacement is not enough. Indeed, since DAG-based representations are considered, it can be the case that in \( \alpha \) one can find a leaf node \( N \) labeled by \( x \) such that one path from the root of \( \alpha \) to this leaf node does not contain any node labeled by a quantification on \( x \), while other paths from the root to \( N \) contain such quantifications (see [27]).

3.3. Expressiveness, Succinctness, and Polynomial Translations

We consider three notions of translations on classes of propositional representations (here, subsets of \( C\text{-QDAG} \)), starting from the less demanding one, namely expressiveness:

**Definition 6 (expressiveness).** Let \( L_1 \) and \( L_2 \) be two subsets of \( C\text{-QDAG} \). \( L_1 \) is at least as expressive as \( L_2 \), denoted \( L_1 \leq_e L_2 \), if for every representation \( \alpha \in L_2 \), there exists an equivalent representation \( \beta \in L_1 \).

A first refinement of such a notion of translatability consists in considering only polynomial-space translations, i.e., the size of the translated representation must remain polynomial in the size of the input representation:

**Definition 7 (succinctness).** Let \( L_1 \) and \( L_2 \) be two subsets of \( C\text{-QDAG} \). \( L_1 \) is at least as succinct as \( L_2 \), denoted \( L_1 \leq_s L_2 \), if there exists a polynomial \( p \) such that for every representation \( \alpha \in L_2 \), there exists an equivalent representation \( \beta \in L_1 \) where \( |\beta| \leq p(|\alpha|) \).

Finally, we consider still more demanding translations, namely polynomial-time translations:

**Definition 8 (polynomial translation).** Let \( L_1 \) and \( L_2 \) be two subsets of \( C\text{-QDAG} \). \( L_2 \) is said to be polynomially translatable into \( L_1 \), noted \( L_1 \leq_p L_2 \), if there exists a (deterministic) polynomial-time algorithm \( f \) such that for every \( \alpha \in L_2 \), we have \( f(\alpha) \in L_1 \) and \( f(\alpha) \equiv \alpha \). We also say that \( \alpha \) is polynomially translatable into \( f(\alpha) \).
Clearly enough, \( \leq_e \), \( \leq_s \), and \( \leq_p \) are pre-orders (i.e., reflexive and transitive relations) over the subsets of \( C^{-\text{QDAG}} \). Furthermore, we have the inclusions:

\[
\leq_p \subset \leq_s \subset \leq_e
\]

For each relation \( \leq_\ast \) among \( \leq_e \), \( \leq_s \), and \( \leq_p \), the relation \( \sim_\ast \) denotes the symmetric part of \( \leq_\ast \), defined by \( L_1 \sim_\ast L_2 \) if \( L_1 \leq_\ast L_2 \) and \( L_2 \leq_\ast L_1 \). By construction, each \( \sim_\ast \) is an equivalence relation (i.e., a reflexive, symmetric and transitive relation). On the other hand, the relation \( <_\ast \) denotes the asymmetric part of \( \leq_\ast \), defined by \( L_1 <_\ast L_2 \) if \( L_1 \leq_\ast L_2 \) and \( L_2 \not\leq_\ast L_1 \). By construction, each \( <_\ast \) is a strict order (i.e., an irreflexive and transitive relation).

In the following, \( L_1 \not\leq_s L_2 \) means that \( L_1 \not\leq_s L_2 \) unless the polynomial hierarchy \( \text{PH} \) collapses (which is considered very unlikely in complexity theory).

When \( L_1 \leq_e L_2 \) holds, every representation from \( L_2 \) can be translated into an equivalent representation from \( L_1 \). The minimal elements w.r.t. \( \leq_e \) (i.e., the most expressive elements) of the set of all subsets of \( C^{-\text{QDAG}} \) when \( C \) is any functionally complete set of connectives (especially, as soon as \( C \) contains \( \lor \) and \( \land \) since leaf nodes of \( C^{-\text{QDAG}} \) representations are labeled by literals) are called complete propositional classes: they can provide a representation (up to logical equivalence) of any Boolean function.

When \( L_1 \sim_\ast L_2 \) (resp. \( L_1 \sim_s L_2 \), \( L_1 \sim_p L_2 \)), \( L_1 \) and \( L_2 \) are said to be equally expressive (resp. equally succinct, polynomially equivalent).

Whenever \( L_1 \) is polynomially translatable into \( L_2 \), every query which can be answered in polynomial time in \( L_2 \) can also be answered in polynomial time in \( L_1 \); and conversely, every query which cannot be answered in polynomial time in \( L_1 \) unless \( P = \text{NP} \) cannot be answered in polynomial time in \( L_2 \), unless \( P = \text{NP} \). Furthermore, polynomially equivalent classes are equally efficient in the sense that they possess the same set of tractable queries and transformations.

4. On Closures of Propositional Representations

Intuitively, a closure principle applied to a class \( \mathcal{L} \) of propositional representations defines a new class, called a closure of \( \mathcal{L} \), through the (implicit) application of “operators” (i.e., connectives from \( C \) or quantifications) to the representations from \( \mathcal{L} \). Formally:
Definition 9 (closure). Let $\mathcal{L} \subseteq C-\text{QDAG}$ and $\Delta \subseteq C \cup \{\forall, \exists\}$. The closure $\mathcal{L}[\Delta]$ of $\mathcal{L}$ by $\Delta$ is the subset of $C-\text{QDAG}$ inductively defined as follows:\footnote{In order to alleviate the notations, when $\Delta = \{\delta_1, \ldots, \delta_n\}$, we write $\mathcal{L}[\delta_1, \ldots, \delta_n]$ instead of $\mathcal{L}[[\delta_1, \ldots, \delta_n]]$.}

1. if $\alpha \in \mathcal{L}$, then $\alpha \in \mathcal{L}[\Delta]$,

2. if $c \in \Delta$ is an $n$-ary connective and $\alpha_1, \ldots, \alpha_n$ are elements of $\mathcal{L}[\Delta]$ such that $\forall i, j \in \{1, \ldots, n\}$, if $i \neq j$ then $\alpha_i$ and $\alpha_j$ do not share any common (nonempty) subgraphs, then $c(\alpha_1, \ldots, \alpha_n) \in \mathcal{L}[\Delta]$,

3. if $c \in \Delta$ is a quantifier, $x \in PS$, and $\alpha \in \mathcal{L}[\Delta]$, then $cx.\alpha \in \mathcal{L}[\Delta]$.

Each element of $\mathcal{L}[\Delta]$ can be viewed as a “tree” which “internal nodes” are labeled by connectives from $C$ or quantifications and its “leaf nodes” correspond to “independent” representations from $\mathcal{L}$. Accordingly, the representations $\alpha_i$ considered in item 2. of Definition 9 do not share any common subgraphs.

Clearly, if there exists a polynomial-time algorithm for determining whether a given representation $\alpha \in C-\text{QDAG}$ belongs or not to $\mathcal{L}$, then there also exists a polynomial-time algorithm for determining whether a given representation $\alpha \in C-\text{QDAG}$ belongs or not to the closure $\mathcal{L}[\Delta]$ of $\mathcal{L}$ by $\Delta$.

We have derived the following (easy) proposition, which rules the inclusions between closures depending on the way their sets of connectives are related by set inclusion:

Proposition 1. For every subset $\mathcal{L}$, $\mathcal{L}'$ of $C-\text{QDAG}$ and every subset $\Delta_1$, $\Delta_2$ of $C \cup \{\exists, \forall\}$, we have:

0. $\mathcal{L} \subseteq \mathcal{L}[\Delta_1]$, and if $\mathcal{L} \subseteq \mathcal{L}'$, then $\mathcal{L}[\Delta_1] \subseteq \mathcal{L}'[\Delta_1]$.

1. $(\mathcal{L}[\Delta_1])[\Delta_2] \subseteq \mathcal{L}[\Delta_1 \cup \Delta_2]$.

2. $(\mathcal{L}[\Delta_1])[\Delta_1] = \mathcal{L}[\Delta_1]$.

3. If $\Delta_1 \subseteq \Delta_2$ then $\mathcal{L}[\Delta_1] \subseteq \mathcal{L}[\Delta_2]$.

4. If $\Delta_1 \subseteq \Delta_2$ then $(\mathcal{L}[\Delta_1])[\Delta_2] = \mathcal{L}[\Delta_2]$ and $(\mathcal{L}[\Delta_2])[\Delta_1] = \mathcal{L}[\Delta_2]$. 


Some additional properties stating how some closures of a class \( \mathcal{L} \) can be composed, can be derived when bound variables can be “freely” renamed in the \( \mathcal{L} \) representations. The property of stability by uniform renaming, given at Definition 10, characterizes the subsets of \( C\text{-QDAG} \) for which, intuitively, the choice of variable names in the \( \mathcal{L} \) representations does not really matter:

**Definition 10 (stability by uniform renaming).** Let \( \mathcal{L} \) be any subset of \( C\text{-QDAG} \). \( \mathcal{L} \) is stable by uniform renaming if for every \( \alpha \in \mathcal{L} \), for every non-empty subset \( V \) of variables occurring in \( \alpha \), there exist arbitrarily many distinct bijections \( r_i \) from \( V \) to subsets \( V_i \) of fresh variables from \( \text{PS} \) (i.e., for each \( i, j \in \mathbb{N} \) with \( i \neq j \), we have \( V_i \cap V_j = V_i \cap V = \emptyset \)) such that the representation \( r_i(\alpha) \) obtained by replacing in \( \alpha \) (in a uniform way) every occurrence of \( x \in V \) (either quantified or non-quantified) by \( r_i(x) \) belongs to \( \mathcal{L} \) as well.

This condition is not very demanding: all the ”standard” classes of propositional representations (quantified or not) are stable by uniform renaming (when based on a countably infinite set \( \text{PS} \) as this is the case here). Special attention must nevertheless be paid to the \( \text{OBDD}_< \) language, and more generally to every class based on an ordered set of propositional variables. For the \( \text{OBDD}_< \) case where \( < \) is a strict and complete ordering over \( \text{PS} \) we may assume the ordered set \( (\text{PS}, <) \) to be of order type \( \eta \) (\( \eta \) is the order type of the set of rational numbers equipped with its usual ordering [38]). This restriction is harmless since the set of variables occurring in any \( \text{OBDD}_< \) representation is finite. In a nutshell, whatever the way the variables occurring in a given \( \text{OBDD}_< \) representation \( \alpha \) are ordered w.r.t. \( < \), one must be able to ”insert” in this ordering arbitrarily many fresh variables between two variables of \( \alpha \) while preserving the way other variables are ordered. Order type \( \eta \) clearly allows it (between two distinct rational numbers one can find countably many rationals). To make things clearer, let us give a counter-example: let \( \text{PS}= \{x_i \mid i \in \mathbb{N}\} \) ordered in such a way that for every \( i \in \mathbb{N}, x_i < x_{i+1} \). Consider an \( \text{OBDD}_< \) representation of \( x_0 \lor x_1 \) as given in Figure 3. \( < \) is not of type \( \eta \). Take \( V = \{x_0\} \): \( x_0 \) cannot be renamed into a different variable from \( \text{PS} \) without questioning the ordering requirement over \( \text{OBDD}_< \), which shows that \( \text{OBDD}_< \) is not stable by uniform renaming in this case.

Straightforwardly, the closure by any set of connectives/quantifiers of any class of propositional representations, which is stable by uniform renaming, also is stable by uniform renaming.

We are now ready to present more specific results. The following polynomial (dual) equivalences, showing that existential quantifications (resp. universal
quantifications) when viewed as “operators” ”distribute” over disjunctions (resp. conjunctions), are well-known:

$$\forall x. (\alpha_1 \land \ldots \land \alpha_n) \equiv (\forall x. \alpha_1) \land \ldots \land (\forall x. \alpha_n),$$
$$\exists x. (\alpha_1 \lor \ldots \lor \alpha_n) \equiv (\exists x. \alpha_1) \lor \ldots \lor (\exists x. \alpha_n),$$

It can then be shown that:

**Proposition 2.** Let $L$ be any subset of $C-$QDAG s.t. $L$ is stable by uniform renaming. We have:

- $(L[\exists])[\lor] \sim_p (L[\lor])[\exists] \sim_p L[\lor, \exists].$
- $(L[\forall])[\land] \sim_p (L[\land])[\forall] \sim_p L[\land, \forall].$

Figure 4 illustrates the polynomial equivalences between disjunctive closures given at Proposition 2.

Proposition 1 and Proposition 2 show together that when $\Delta = \{\lor, \exists\}$ (resp. $\{\land, \forall\}$) closing $L[\Delta]$ by subsets of $\Delta$ in an iterative fashion does not lead to a "new" class, i.e., a class which is not polynomially equivalent to $L$. Especially, we have

$$(L[\lor, \exists])[\exists] \sim_p (L[\lor, \exists])[\lor] \sim_p L[\lor, \exists].$$

This shows, so to say, that the “sequential” closure of a propositional class, stable by uniform renaming, by a set of operators among $\{\lor, \exists\}$ (resp. among $\{\land, \forall\}$) is polynomially equivalent to its “parallel” closure. No similar result can be systematically guaranteed for arbitrary choices of classes and operators. For instance, if $L$ is the set of literals over $PS$, then the ”sequential” closure $(L[\lor])[\land]$ is the set of all \textsc{cnf} formulae, the ”sequential” closure $(L[\land])[\lor]$ is the set of all \textsc{dnf} formulae, and the ”parallel” closure $L[\lor, \land]$ is the set of all \textsc{nfn} representations. It is
Figure 4: Polynomial equivalences between disjunctive closures. The representation $\alpha$ at the top of the picture belongs to $\mathcal{L}[\lor, \exists]$. $\beta_1[X, Y, Z, T]$ and $\beta_2[X, Y, Z, T]$ denote propositional representations (not necessarily tree-structured ones) from $\mathcal{L}$ such that $\text{Var}(\beta_1) = \text{Var}(\beta_2) = X \cup Y \cup Z \cup T$, where $X$, $Y$, $Z$, and $T$ are pairwise disjoint, finite subsets of $PS$. The representation $\alpha_{\exists \lor}$ at the bottom, left-hand side of the picture is a $(\mathcal{L}[\lor])[\exists]$ representation into which $\alpha$ can be polynomially translated. The representation $\alpha_{\lor \exists}$ at the bottom, right-hand side of the picture is a $(\mathcal{L}[\exists])[\lor]$ representation into which $\alpha$ can be polynomially translated.
well-known that those three languages are not pairwise polynomially equivalent (indeed, we have \( \text{CNF} \not\leq_s \text{DNF} \), \( \text{DNF} \not\leq_s \text{CNF} \), and \( \text{NNF} \not<_s \text{DNF} \), see e.g. [39]). Similarly, if \( \mathcal{L} = \text{CLAUSE} \), then \( (\mathcal{L}[\land])[\exists] \) and \( \mathcal{L}[\land, \exists] \) are polynomially equivalent to \( \text{CNF}[\exists] \), but \( (\mathcal{L}[\exists])[\land] \) is polynomially equivalent to \( \text{CNF} \), which is not polynomially equivalent to \( \text{CNF}[\exists] \). Indeed, whatever \( C \), \( C\text{-DAG} \) is polynomially translatable into \( \text{CNF}[\exists] \) using Tseitin’s extension principle [40], while \( \text{CNF} \) is not at least as succinct as \( C\text{-DAG} \) as soon as \( C \supseteq \{\land, \lor, \neg\} \) (indeed, \( \text{CNF} \) is not at least as succinct as the subset \( \text{DNF} \) of \( \text{NNF} \), which is itself a subset of \( C\text{-DAG} \) in this case).

We have derived the following proposition, which relates the queries and the transformations offered by \( \mathcal{L} \), with the queries and transformations offered by its disjunctive closures \( \mathcal{L}[\lor] \) (the disjunction closure of \( \mathcal{L} \)), \( \mathcal{L}[\exists] \) (the existential closure of \( \mathcal{L} \)), and \( \mathcal{L}[\lor, \exists] \) (the full disjunctive closure of \( \mathcal{L} \)).

**Proposition 3.** Let \( \mathcal{L} \) be any subset of \( C\text{-QDAG} \) s.t. \( \mathcal{L} \) is stable by uniform renaming.

- If \( \mathcal{L} \) satisfies \( \text{CO} \) (resp. \( \text{CD} \)), then \( \mathcal{L}[\lor], \mathcal{L}[\exists] \) and \( \mathcal{L}[\lor, \exists] \) satisfy \( \text{CO} \) (resp. \( \text{CD} \)).
- If \( \mathcal{L} \) satisfies \( \text{CO} \) and \( \text{CD} \), then \( \mathcal{L} \) satisfies \( \text{CE} \) and \( \text{ME} \).
- If \( \mathcal{L} \) satisfies \( \text{CO} \) and \( \text{CD} \), then \( \mathcal{L}, \mathcal{L}[\lor], \mathcal{L}[\exists] \) and \( \mathcal{L}[\lor, \exists] \) satisfy \( \text{MC} \).
- \( \mathcal{L}[\lor] \) and \( \mathcal{L}[\lor, \exists] \) satisfy \( \lor \text{C} \) (hence \( \lor \text{BC} \)) and \( \mathcal{L}[\exists] \) and \( \mathcal{L}[\lor, \exists] \) satisfy \( \text{FO} \) (hence \( \text{SFO} \)).
- If \( \mathcal{L} \) satisfies \( \text{FO} \) (resp. \( \text{SFO} \)), then \( \mathcal{L}[\lor] \) satisfies \( \text{FO} \) (resp. \( \text{SFO} \)).
- If \( \mathcal{L} \) satisfies \( \land \text{C} \) (resp. \( \land \text{BC}, \lor \text{C}, \lor \text{BC} \)), then \( \mathcal{L}[\exists] \) satisfies \( \land \text{C} \) (resp. \( \land \text{BC}, \lor \text{C}, \lor \text{BC} \)).

Note that applying disjunctive closures do not preserve other queries or transformations in the general case. Thus:

- if \( \mathcal{L} \) satisfies \( \text{VA} \) (resp. \( \text{IM}, \text{CT}, \text{EQ}, \) and \( \text{SE} \)), then it can be the case that \( \mathcal{L}[\lor] \) does not satisfy it. For instance, \( \text{TERM} \) satisfies each of \( \text{VA}, \text{IM}, \text{CT}, \text{EQ}, \) and \( \text{SE} \), but \( \text{TERM}[\lor] = \text{DNF} \) does not satisfy any of them unless \( \text{P} = \text{NP} \) [1].
• if \( \mathcal{L} \) satisfies \( \text{VA} \) (resp. \( \text{IM}, \text{CT}, \text{EQ}, \) and \( \text{SE} \)), then it can be the case that \( \mathcal{L}[\exists] \) does not satisfy it. Thus, \( \mathcal{L} = \text{CNF} \) satisfies \( \text{VA} \) and \( \text{IM} \) but \( \text{CNF}[\exists] \) does not satisfy any of them unless \( P = NP \); indeed, \( \text{DNF} \) (which does not offer any of them) is polynomially translatable into \( \text{CNF}[\exists] \) using Tseitin’s transformation [40]. Similarly, \( \mathcal{L} = \text{HORN} \) satisfies both \( \text{EQ} \) and \( \text{SE} \), but \( \text{HORN}[\exists] \) does not offer any of them (see Proposition 5). Finally, the subset \( \mathcal{L} = \text{d-DNF} \) of \( \text{DNF} \) consisting of deterministic \( \text{DNF} \) formulae (i.e., the \( \text{DNF} \) formulae \( \alpha = \bigvee_{i=1}^{n} \gamma_i \) such that for each \( i, j \in 1, \ldots, n \), if \( i \neq j \), then the terms \( \gamma_i \) and \( \gamma_j \) are such that \( \gamma_i \land \gamma_j \) is inconsistent) satisfies \( \text{CT} \), but \( \text{d-DNF}[\exists] \) does not. Indeed, \( \text{DNF} \) is polynomially translatable into \( \text{d-DNF}[\exists] \): with each \( \text{DNF} \) formula \( \alpha = \bigvee_{i=1}^{n} \gamma_i \) we can associate in polynomial time the equivalent \( \text{d-DNF}[\exists] \) formula \( \exists \{ y_1, \ldots, y_n \}. \bigwedge_{i=1}^{n} (y_i \lor \bigwedge_{\substack{j=1 \atop j \neq i}}^{n} \neg y_j \land \gamma_i) \), where \( \{ y_1, \ldots, y_n \} \) is a set of fresh variables (disjoint from \( \text{Var}(\alpha) \)).

• if \( \mathcal{L} \) satisfies \( \forall \), then it can be the case that \( \mathcal{L}[\forall] \) does not satisfy it. Thus, \( \mathcal{L} = \text{TERM} \) satisfies \( \forall \), but \( \text{TERM}[\forall] = \text{DNF} \) does not, unless \( P = NP \) [1].

• if \( \mathcal{L} \) satisfies \( \neg \forall \), then it can be the case that none of \( \mathcal{L}[\forall] \) and \( \mathcal{L}[\exists] \) satisfies it. Thus, \( \mathcal{L} = \text{OBDD}_\prec \) satisfies \( \neg \forall \), but none of \( \text{OBDD}_\prec[\forall] \) and \( \text{OBDD}_\prec[\exists] \) satisfies \( \neg \forall \) unless \( P = NP \). As to \( \text{OBDD}_\prec[\forall] \), this comes from the fact that \( \text{TERM} \geq_p \text{OBDD}_\prec \), which implies that \( \text{DNF} \geq_p \text{OBDD}_\prec[\forall] \). Since every \( \text{CNF} \) formula \( \alpha \) is polynomially translatable into the negation of a \( \text{DNF} \) formula \( \beta \), if \( \text{OBDD}_\prec[\forall] \) would satisfy \( \neg \forall \), then the consistency of \( \alpha \) could be tested in polynomial time by computing first an \( \text{OBDD}_\prec[\forall] \) representation equivalent to \( \beta \), then ”negating” it to reach an \( \text{OBDD}_\prec[\forall] \) representation equivalent to \( \alpha \). Indeed, since \( \text{OBDD}_\prec \) satisfies \( \text{CO} \), \( \text{OBDD}_\prec[\forall] \) also satisfies \( \text{CO} \) (see Proposition 3). As to \( \text{OBDD}_\prec[\exists] \), we can make a rather similar proof given that \( \text{OBDD}_\prec[\exists] \) also satisfies \( \text{CO} \) (see again Proposition 3). To get the proof, it is enough to show that \( \text{OBDD}_\prec[\forall] \geq_p \text{OBDD}_\prec[\exists] \): let \( \alpha = \bigvee_{i=1}^{n} \alpha_i \) be an \( \text{OBDD}_\prec[\forall] \) representation; let \( y_1, \ldots, y_n \) be variables from \( PS \setminus \text{Var}(\alpha) \) such that each \( y_i \) (\( i \in 1, \ldots, n \)) precedes every variable from \( \text{Var}(\alpha) \). From \( \alpha \), we can generate in polynomial time the \( \text{OBDD}_\prec \) representation

\[
\beta = \langle y_1, \alpha_1, \langle y_2, \alpha_2, \ldots, \langle y_n, \alpha_n, \bot, \rangle \ldots \rangle \rangle.
\]

To conclude the proof it is enough to observe that \( \alpha \) is equivalent to the \( \text{OBDD}_\prec[\exists] \) representation \( \exists \{ y_1, \ldots, y_n \}. \beta \).
• if $L$ satisfies $EN$, then it can be the case that $L[\forall]$ does not satisfy it. Thus, $L = \text{TERM}$ satisfies $EN$, but $\text{TERM}[\forall] = \text{DNF}$ does not, unless $P = \text{NP}$, since a DNF formula $\alpha$ is valid iff its universal closure $\forall \text{Var}(\alpha).\alpha$ is valid iff $\forall \text{Var}(\alpha).\alpha$ is consistent (since $\forall \text{Var}(\alpha).\alpha$ has no free variable, it is equivalent to $\top$ or to $\bot$, hence it is consistent precisely when it is valid), and DNF satisfies CO.

• if $L$ satisfies $EN$, then it can be the case that $L[\exists]$ does not satisfy it. Thus, $L = \text{CNF}$ satisfies $EN$, but $\text{CNF}[\exists] = \text{DNF}$ does not, unless $\text{PH}$ collapses. This comes easily from the fact that the validity problem for $\text{CNF}[\exists]$ formulae of the form $\exists X.\alpha$ is $\Pi^p_2$-complete. Indeed, $\exists X.\alpha$ is valid iff the (closed) quantified Boolean formula $\forall \text{Var}(\alpha) \setminus X.(\exists X.\alpha)$ is valid.

5. On the Disjunctive Closures of $\text{KROM}$, $\text{HORN}$, $\text{AFF}$, $\text{K/H}$, and $\text{renH}$

Let us now focus on the disjunctive closures of $\text{KROM}$, $\text{HORN}$, $\text{AFF}$, $\text{K/H}$, and $\text{renH}$.

First of all, it is obvious that the four languages $\text{KROM}$, $\text{HORN}$, $\text{K/H}$, and $\text{AFF}$ are stable by uniform renaming. This is also the case for $\text{renH}$: if $V$ is a Horn renaming for a $\text{renH}$ formula $\alpha$, and if $\alpha'_X$ is the CNF formula obtained by substituting in a uniform way in $\alpha$ every occurrence of a variable $v$ from $X \subseteq \text{PS}$ by the fresh variable $v'$, then $\alpha'_X$ also is a $\text{renH}$ formula and $V' = \{v \in \text{Var}(\alpha') \mid v \in V \setminus X\} \cup \{v' \in \text{Var}(\alpha') \mid v \in V \cap X\}$ is a Horn renaming for it.

Now, thanks to Propositions 1 and 2, it is enough to consider the three disjunctive closures $L[\exists], L[\forall], L[\exists, \forall]$ with $L$ being any of the five above languages. Clearly enough, the disjunction (resp. existential, full disjunctive) closure of any language among $\text{KROM}$, $\text{HORN}$, $\text{K/H}$, $\text{renH}$, and $\text{AFF}$ is also stable by uniform renaming.

Applying the disjunctive closure principles $[\forall], [\exists], [\forall, \exists]$ to the five languages $\text{KROM}$, $\text{HORN}$, $\text{K/H}$, $\text{renH}$, and $\text{AFF}$ leads to consider fifteen additional languages. The following easy result shows that some of the resulting languages do not need to be considered separately, because they are polynomially equivalent.

**Proposition 4.**

- $\text{KROM} \sim_p \text{KROM}[\exists]$.
- $\text{KROM}[\forall] \sim_p \text{KROM}[\forall, \exists]$.
• $\text{AFF} \sim_p \text{AFF}[\exists]$.

• $\text{AFF}[v] \sim_p \text{AFF}[v, \exists]$.

As a direct consequence, we have that $KROM$ and $KROM[\exists]$ (resp. $AFF$ and $AFF[\exists]$, $KROM[v]$ and $KROM[v, \exists]$, $AFF[v]$ and $AFF[v, \exists]$) are both (pairwise) equally succinct and equally expressive.


From Proposition 1, we get that:

\[
\begin{align*}
KROM & \subseteq KROM[v] \subseteq KROM[v, \exists] \\
HORN & \subseteq HORN[\exists] \subseteq HORN[v, \exists] \\
HORN & \subseteq HORN[v] \subseteq HORN[v, \exists] \\
K/H & \subseteq K/H[\exists] \subseteq K/H[v, \exists] \\
K/H & \subseteq K/H[v] \subseteq K/H[v, \exists] \\
\text{renH} & \subseteq \text{renH}[\exists] \subseteq \text{renH}[v, \exists] \\
\text{renH} & \subseteq \text{renH}[v] \subseteq \text{renH}[v, \exists] \\
\text{AFF} & \subseteq \text{AFF}[v] 
\end{align*}
\]

Obviously enough, from the definitions of the languages $KROM$, $HORN$, $K/H$, and $\text{renH}$, we also have the following inclusions:

\[
\begin{align*}
KROM & \subseteq K/H \\
HORN & \subseteq K/H \\
HORN & \subseteq \text{renH} 
\end{align*}
\]

From such results, we immediately derive that:

\[
\begin{align*}
KROM[v] & \subseteq K/H[v] \\
HORN[v] & \subseteq K/H[v] \\
HORN[v] & \subseteq \text{renH}[v] \\
HORN[\exists] & \subseteq K/H[\exists] \\
HORN[\exists] & \subseteq \text{renH}[\exists] \\
HORN[v, \exists] & \subseteq K/H[v, \exists] \\
HORN[v, \exists] & \subseteq \text{renH}[v, \exists] 
\end{align*}
\]
In addition, since every consistent \textsc{Krom} formula is a \textsc{renH} formula\textsuperscript{5} and since \textsc{Krom} satisfies \textsc{co}, with every \textsc{k/h} formula we can associate in polynomial time an equivalent \textsc{renH} formula, i.e., \textsc{k/h} \geq_p \textsc{renH}. As a consequence, we also get that
\[
\text{K/H}[V] \geq_p \text{renH}[V] \\
\text{K/H}[\exists] \geq_p \text{renH}[\exists] \\
\text{K/H}[V, \exists] \geq_p \text{renH}[V, \exists]
\]

Finally, since for every subset \( \mathcal{L} \) of \( \textsc{c-qdag} \), \( \mathcal{L} \subseteq \mathcal{L}[V] \subseteq \mathcal{L}[V, \exists] \), \( \subseteq \) is included into \( \geq_p \), and both \( \subseteq \) and \( \geq_p \) are transitive relations, a number of additional inclusions/polynomial translatability results can be directly obtained from the results above; they will be exploited in some forthcoming proofs.

5.1. Queries and Transformations

As to queries, we have obtained the following results:

**Proposition 5.** The results in Table 1 hold.

As to transformations, we have obtained the following results:

**Proposition 6.** The results in Table 2 hold.

5.2. Expressiveness

It is well-known that none of the languages \textsc{Krom}, \textsc{horn}, \textsc{k/h}, \textsc{renH}, or \textsc{aff} is complete for propositional logic. For instance, there is no formula from any of these languages which is equivalent to the \textsc{cnf} formula \((x \lor y \lor z) \land (\neg x \lor \neg y \lor \neg z)\). This is problematic for many applications; indeed, what can be done when the available information cannot be represented in the targeted language? Approximating it is not always an option, especially because the best approximation of the available information can be rough and the missing pieces of information in the approximation can be crucial ones for reasoning and/or decision making. In the following, we are going to prove that while considering the existential closure of

\textsuperscript{5}This is a direct consequence of the fact that a \textsc{cnf} formula \( \alpha \) is renamable Horn precisely when there exists an interpretation \( \omega \) such that at most one literal per clause of \( \alpha \) is false in \( \omega \) [26]; indeed, when \( \alpha \) is a \textsc{Krom} formula, this last statement exactly means that \( \alpha \) is consistent.
any of those languages does not increase its expressiveness, switching to its disjunction closure (or to its full disjunctive closure) is enough to recover a complete propositional language, thus escaping from the above mentioned expressiveness problem.

Let us start with the existential closures of KROM, HORN, K/H, renH, and AFF. First, since KROM (resp. AFF) is polynomially equivalent to KROM[∃] (resp. AFF[∃]), it turns out that those languages are (pairwise) equally expressive: KROM[∃] \sim_e KROM, and AFF[∃] \sim_e AFF. Similarly, we have derived the following expressiveness results, showing that the existential closure of any language \( \mathcal{L} \) among HORN, K/H, and renH is not more expressive than \( \mathcal{L} \) itself.

**Proposition 7.**

- HORN[∃] \sim_e HORN.
- K/H[∃] \sim_e K/H.
- renH[∃] \sim_e renH.

<table>
<thead>
<tr>
<th></th>
<th>CO</th>
<th>VA</th>
<th>CE</th>
<th>IM</th>
<th>EQ</th>
<th>SE</th>
<th>CT</th>
<th>ME</th>
<th>MC</th>
</tr>
</thead>
<tbody>
<tr>
<td>renH[V,∃]</td>
<td>√</td>
<td>o</td>
<td>√</td>
<td>o</td>
<td>o</td>
<td>o</td>
<td>o</td>
<td>√</td>
<td>√</td>
</tr>
<tr>
<td>K/H[V,∃]</td>
<td>√</td>
<td>o</td>
<td>√</td>
<td>o</td>
<td>o</td>
<td>o</td>
<td>o</td>
<td>√</td>
<td>√</td>
</tr>
<tr>
<td>HORN[V,∃]</td>
<td>√</td>
<td>o</td>
<td>√</td>
<td>o</td>
<td>o</td>
<td>o</td>
<td>o</td>
<td>√</td>
<td>√</td>
</tr>
<tr>
<td>AFF[V]</td>
<td>√</td>
<td>o</td>
<td>√</td>
<td>o</td>
<td>o</td>
<td>o</td>
<td>o</td>
<td>√</td>
<td>√</td>
</tr>
<tr>
<td>renH[∃]</td>
<td>√</td>
<td>√</td>
<td>√</td>
<td>o</td>
<td>o</td>
<td>o</td>
<td>o</td>
<td>√</td>
<td>√</td>
</tr>
<tr>
<td>K/H[∃]</td>
<td>√</td>
<td>√</td>
<td>√</td>
<td>o</td>
<td>o</td>
<td>o</td>
<td>o</td>
<td>√</td>
<td>√</td>
</tr>
<tr>
<td>HORN[∃]</td>
<td>√</td>
<td>√</td>
<td>√</td>
<td>o</td>
<td>o</td>
<td>o</td>
<td>o</td>
<td>√</td>
<td>√</td>
</tr>
</tbody>
</table>

Table 1: KROM, HORN, K/H, AFF, renH, their disjunction, existential and full disjunctive closures and the corresponding polynomial-time queries. √ means "satisfies" and o means "does not satisfy unless \( P = NP \)."
<table>
<thead>
<tr>
<th></th>
<th>CD</th>
<th>FO</th>
<th>SFO</th>
<th>EN</th>
<th>SEN</th>
<th>∧C</th>
<th>∧BC</th>
<th>∨C</th>
<th>∨BC</th>
<th>¬C</th>
</tr>
</thead>
<tbody>
<tr>
<td>renH[V,Ξ]</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>o</td>
<td>o</td>
<td>o</td>
<td>o</td>
<td>✓</td>
<td>✓</td>
<td>o</td>
</tr>
<tr>
<td>K/H[V,Ξ]</td>
<td>✓</td>
<td>●</td>
<td>✓</td>
<td>o</td>
<td>o</td>
<td>o</td>
<td>o</td>
<td>✓</td>
<td>✓</td>
<td>o</td>
</tr>
<tr>
<td>HORN[V,Ξ]</td>
<td>✓</td>
<td>●</td>
<td>✓</td>
<td>o</td>
<td>o</td>
<td>o</td>
<td>o</td>
<td>✓</td>
<td>✓</td>
<td>o</td>
</tr>
<tr>
<td>AFF[V]</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>o</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>o</td>
</tr>
<tr>
<td>renH[Ξ]</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>!</td>
<td>!</td>
<td>!</td>
<td></td>
<td></td>
</tr>
<tr>
<td>K/H[Ξ]</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>!</td>
<td>!</td>
<td>!</td>
<td></td>
<td></td>
</tr>
<tr>
<td>HORN[Ξ]</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>!</td>
<td></td>
<td></td>
</tr>
<tr>
<td>AFF</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td></td>
<td></td>
</tr>
<tr>
<td>renH</td>
<td>✓</td>
<td>●</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td></td>
<td></td>
</tr>
<tr>
<td>K/H</td>
<td>✓</td>
<td>●</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td></td>
<td></td>
</tr>
<tr>
<td>HORN</td>
<td>✓</td>
<td>●</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td></td>
<td></td>
</tr>
<tr>
<td>KROM</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Table 2: KROM, HORN, K/H, AFF, renH, their disjunction, existential and full disjunctive closures and the corresponding polynomial-time transformations. ✓ means “satisfies,” ● means “does not satisfy,” while ◦ means “does not satisfy unless P=NP.” ! means that the transformation is not always feasible within the language.

Now, from the definitions of KROM, HORN, K/H, renH, the fact that K/H \geq_p renH, and the fact that \( x \lor y \) is a KROM formula, which is not equivalent to a HORN one, \( \neg x \lor \neg y \lor \neg z \) is a HORN formula which is not equivalent to a KROM one, \( x \lor y \lor z \) is a renH formula which is not equivalent to a K/H one, we easily get that:

\[
\text{KROM} \not<_{e} \text{HORN and HORN} \not<_{e} \text{KROM} \\
\text{renH} <_{e} K/H <_{e} \text{HORN} \\
K/H <_{e} \text{KROM}
\]

In addition, AFF and any of KROM, HORN, K/H, renH are incomparable w.r.t. \( \leq_e \). Indeed, there is no renH formula equivalent to the AFF formula \( x \oplus y \oplus z \). This comes from the fact that every CNF formula equivalent to \( x \oplus y \oplus z \) must contain the four clauses \( x \lor y \lor z, \neg x \lor \neg y \lor z, x \lor \neg y \lor \neg z, \neg x \lor y \lor \neg z \) since those clauses are essential prime implicates of \( x \oplus y \oplus z \), plus the fact that by construction, every CNF formula containing the clauses \( x \lor y \lor z, \neg x \lor \neg y \lor z, x \lor \neg y \lor \neg z, \neg x \lor y \lor \neg z \) is not renamable Horn (renaming at least two variables...
in the first clause to make it a Horn clause also changes one of the remaining three clauses into a non-Horn one. Conversely, there is no AFF formula equivalent to \( \neg x \lor \neg y \), which is both in KROM and in HORN. This is a direct consequence of the semantical characterization result concerning AFF recalled in Section 2.3: with \( x < y \), \( \omega_1 = 00 \), \( \omega_2 = 01 \), and \( \omega_3 = 10 \) are models of \( \neg x \lor \neg y \), but \( \oplus(\omega_1, \omega_2, \omega_3) = 11 \) is not a model of \( \neg x \lor \neg y \).

Let us finally switch to the disjunction closures and the full disjunctive closures of KROM, HORN, K/H, renH, or AFF; interestingly, the eight languages defined as such are equally, and fully, expressive:

**Proposition 8.** KROM[\lor], HORN[\lor], K/H[\lor], renH[\lor], AFF[\lor], HORN[\lor, \exists], K/H[\lor, \exists], renH[\lor, \exists] are complete propositional languages.

Figure 5 depicts the expressiveness relationships identified in the above propositions.

![Diagram](image)

Figure 5: The expressiveness picture for disjunctive closures. An arrow from \( \mathcal{L}_1 \) to \( \mathcal{L}_2 \) means that \( \mathcal{L}_1 \) is strictly more expressive than \( \mathcal{L}_2 \), so that a lack of arrow means that the expressiveness of \( \mathcal{L}_1 \) and the expressiveness of \( \mathcal{L}_2 \) are incomparable).

### 5.3. Succinctness

As to incomplete languages, since KROM (resp. AFF) is polynomially equivalent to KROM[\exists] (resp. AFF[\exists]), those languages are (pairwise) equally succinct:
KROM[∃] \sim_s KROM, and AFF[∃] \sim_s AFF. More interestingly, we have obtained the following succinctness results, showing that the existential closure of any language \( \mathcal{L} \) among \textsc{horn}, \( K/H \), and renH is strictly more succinct than \( \mathcal{L} \) itself.

**Proposition 9.**

- \( \text{HORN[∃]} <_s \text{HORN} \).
- \( \text{K/H[∃]} <_s \text{K/H} \).
- \( \text{renH[∃]} <_s \text{renH} \).
- renH and \( K/H[∃] \) are incomparable w.r.t. \( \leq_s \).
- \( K/H \) and \( \text{HORN[∃]} \) are incomparable w.r.t. \( \leq_s \).

Figure 6 summarizes the succinctness relationships among incomplete languages identified in Proposition 9. We observe that it does not coincide with the corresponding expressiveness picture, restricted to incomplete languages (see Figure 5).

---

Figure 6: The succinctness picture for incomplete languages. An arrow from \( \mathcal{L}_1 \) to \( \mathcal{L}_2 \) means that \( \mathcal{L}_1 \) is strictly more succinct than \( \mathcal{L}_2 \), i.e., \( \mathcal{L}_1 <_s \mathcal{L}_2 \). The arrow is thick in the specific case when the fact that \( \mathcal{L}_2 \nleq_s \mathcal{L}_1 \) comes from the fact that \( \mathcal{L}_2 \nleq_e \mathcal{L}_1 \). A lack of arrow means that the succinctness of \( \mathcal{L}_1 \) and the succinctness of \( \mathcal{L}_2 \) are incomparable.
As to complete languages, our succinctness results mainly focus on the five languages \( \text{KROM}[\lor] \), \( \text{HORN}[\lor, \exists] \), \( \text{K/H}[\lor, \exists] \), \( \text{renH}[\lor, \exists] \), \( \text{AFF}[\lor] \), the full disjunctive closures of the incomplete languages \( \text{KROM} \), \( \text{HORN} \), \( \text{K/H} \), \( \text{renH} \), \( \text{AFF} \) considered at start.\(^6\)

There are several reasons for this focus:

- \( \text{KROM}[\lor] \), \( \text{HORN}[\lor, \exists] \), \( \text{K/H}[\lor, \exists] \), \( \text{renH}[\lor, \exists] \), \( \text{AFF}[\lor] \) are complete languages, while \( \text{KROM} \), \( \text{HORN} \), \( \text{K/H} \), \( \text{renH} \), \( \text{AFF} \) and their existential closures are not (see Proposition 7 and Proposition 8 above).

- \( \text{HORN}[\lor, \exists] \), \( \text{K/H}[\lor, \exists] \), \( \text{renH}[\lor, \exists] \) satisfy the same queries as the corresponding disjunction closures, namely \( \text{HORN}[\lor] \), \( \text{K/H}[\lor] \), \( \text{renH}[\lor] \) (see Proposition 5), and more transformations than them (see Proposition 6), since they offer FO "for free".

- due to the obvious inclusion \( \text{HORN}[\lor, \exists] \subseteq \text{HORN}[\lor] \), we have that \( \text{HORN}[\lor, \exists] \) is at least as succinct as \( \text{HORN}[\lor] \); and similarly for \( \text{K/H}[\lor, \exists] \) and \( \text{renH}[\lor, \exists] \).

Actually, we can strengthen this point by proving that the full disjunctive closure of \( \text{HORN} \) (resp. \( \text{K/H} \), \( \text{renH} \)) is strictly more succinct than the corresponding disjunction closure:

**Proposition 10.**

\[
\begin{align*}
\text{HORN}[\lor, \exists] & <_s \text{HORN}[\lor], \\
\text{K/H}[\lor, \exists] & <_s \text{K/H}[\lor], \\
\text{renH}[\lor, \exists] & <_s \text{renH}[\lor].
\end{align*}
\]

Thus, the full disjunctive closures of the incomplete languages \( \text{KROM} \), \( \text{HORN} \), \( \text{K/H} \), \( \text{renH} \), \( \text{AFF} \) are either equally succinct as the corresponding disjunction closures (this is the case for the closures of \( \text{KROM} \) and of \( \text{AFF} \)), or strictly more succinct than them (for the three remaining languages).

Let us now provide the remaining succinctness results we got. We split our results into two propositions (and two tables). In the first table, we compare \( \text{KROM}[\lor] \), \( \text{HORN}[\lor, \exists] \), \( \text{K/H}[\lor, \exists] \), \( \text{renH}[\lor, \exists] \), \( \text{AFF}[\lor] \) w.r.t. spatial efficiency \( \leq_s \).

**Proposition 11.** The results in Table 3 hold.
Table 3: The relative succinctness of the full disjunctive closures of KROM, HORN, AFF, K/H, and renH.

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>AFF[V]</td>
<td>∼ₘ</td>
<td>∼ₘ</td>
<td>∼ₘ</td>
<td>∼ₘ</td>
<td>∼ₘ</td>
</tr>
<tr>
<td>renH[V, ξ]</td>
<td>∼ₘ</td>
<td>∼ₘ</td>
<td>∼ₘ</td>
<td>∼ₘ</td>
<td>∼ₘ</td>
</tr>
<tr>
<td>K/H[V, ξ]</td>
<td>∼ₘ</td>
<td>∼ₘ</td>
<td>∼ₘ</td>
<td>∼ₘ</td>
<td>∼ₘ</td>
</tr>
<tr>
<td>HORN[V, ξ]</td>
<td>∼ₘ</td>
<td>∼ₘ</td>
<td>∼ₘ</td>
<td>∼ₘ</td>
<td>∼ₘ</td>
</tr>
<tr>
<td>KROM[V]</td>
<td>∼ₘ</td>
<td>∼ₘ</td>
<td>∼ₘ</td>
<td>∼ₘ</td>
<td>∼ₘ</td>
</tr>
</tbody>
</table>

As a direct consequence of Proposition 11, we have that

renH[V, ξ] <ₘ K/H[V, ξ] <ₘ HORN[V, ξ]
K/H[V, ξ] <ₘ KROM[V, ξ]

One can observe that the resulting succinctness picture is similar to the expressiveness picture for the corresponding incomplete fragments AFF, renH, K/H, HORN, KROM.

In Proposition 12, we compare w.r.t. ≤ₘ the languages KROM[V], HORN[V, ξ], K/H[V, ξ], renH[V, ξ] and AFF[V] with several classes of propositional representations for KC which have been introduced so far, and with CNF. We specifically focus on those target classes for which compilers have been developed, i.e., PI, IP, DNF, OBDDₜ, d-DNNF, and DNNFₜ.

Proposition 12. The results in Table 4 hold.

Figure 7 depicts the succinctness relationships reported mainly in Proposition 11 and Proposition 12. The closure languages considered in this paper are underlined.

5.4. Discussion

Let us now compare in more details the five languages KROM, HORN, K/H, renH, and AFF, with their closures and with other classes of propositional representations considered so far for knowledge compilation.

We start with the existential closures. Since applying the existential closure principle to any of KROM, HORN, K/H, renH, and AFF does not change its expressiveness, the existential closures of KROM, HORN, K/H, renH, and AFF are

6Remember that KROM[V, ξ] ∼ₚ KROM[V] and that AFF[V, ξ] ∼ₚ AFF[V], see Proposition 4.
Figure 7: The succinctness picture for complete classes. An arrow from $\mathcal{L}_1$ to $\mathcal{L}_2$ means that $\mathcal{L}_1$ is strictly more succinct than $\mathcal{L}_2$, so that a lack of arrow means that the succinctness of $\mathcal{L}_1$ and the succinctness of $\mathcal{L}_2$ are incomparable for sure (or, under the assumption that the polynomial hierarchy does not collapse when $d$-DNNF is concerned). For a clarity sake, there are two exceptions to this notation, which also concern $d$-DNNF: it is unknown whether $d$-DNNF $\leq_s$ PI, and whether $d$-DNNF $\leq_s$ IP [1].
incomplete languages as well. KROM[\exists] (resp. AFF[\exists]) is polynomially equivalent to KROM (resp. AFF) since KROM and AFF satisfy FO. As a consequence, KROM[\exists] and AFF[\exists] satisfy the same queries and transformations as their underlying language, and KROM[\exists] and AFF[\exists] are equally succinct as KROM and AFF, respectively. For the remaining existential closures (namely, HORN[\exists], K/H[\exists], renH[\exists]), all the transformations already offered by HORN, K/H, renH, are preserved and FO is obtained "for free". However, some queries offered by HORN, K/H, and renH (EQ, SE) are not preserved. This seems to be the price to be paid for the gain in succinctness the existential closures offer. Indeed, we have HORN[\exists] <_s HORN, K/H[\exists] <_s K/H, and renH[\exists] <_s renH. Thus, for applications where their expressiveness proves enough and EQ and SE are not expected but FO is, renH[\exists] (resp. HORN[\exists]) appears as a better choice than renH (resp. HORN) as a target language for KC.

Unlike existential closures, the disjunction closures and the full disjunctive closures of KROM, HORN, K/H, renH, and AFF are complete propositional languages, i.e., fully expressive ones. Furthermore, switching from any of KROM, HORN, K/H, renH, or AFF to its disjunction closure or its full disjunctive closure leads to get VC (hence VBC) "for free" and FO, when it was not already offered. Conversely, some queries and transformations primarily offered are then lost; as to queries, this is the case for VA, IM, EQ, SE for the five languages, plus CT satisfied by AFF but not by any of its disjunction closure or its full disjunctive closure; as to transformations, this is the case for EN and \land C (and even SEN, which is satisfied by renH but not satisfied by renH[\lor] or renH[\lor, \exists], unless P = NP). Just like considering the existential closures of HORN, K/H,
renH leads to strictly more succinct languages, considering the existential closures of HORN[v], K/H[v], renH[v] leads as well to strictly more succinct languages since HORN[v, ‖] <s HORN[v], K/H[v, ‖] <s K/H[v], and renH[v, ‖] <s renH[v]. Thus, it turns out that the full disjunctive closures of KROM, HORN, K/H, renH, and AFF are always at least as interesting as the corresponding disjunction closures from a KC perspective: as to KROM and AFF, those closures are polynomially equivalent, hence equally interesting; as to HORN, K/H, renH, both closures satisfy the same queries, while each full disjunctive closure offers FO (not satisfied by the corresponding disjunction closure) and is strictly more succinct than the underlying language.

Comparing now one another the full disjunctive closures of KROM, HORN, K/H, renH, AFF it turns out that none of them is strictly dominated by another one from the KC point of view. All of them are equally expressive, and they satisfy precisely the same queries CO, CE, ME, MC. As to transformations, KROM[v], HORN[v, ‖], and AFF[v] satisfy CD, FO, SFO, SEN, ^BC, vC and vBC. They are pairwise incomparable w.r.t. succinctness. While K/H[v, ‖] is strictly more succinct than each of KROM[v], or HORN[v, ‖], it does not offer SEN, and while renH[v, ‖] is strictly more succinct than K/H[v, ‖], it does not offer ^BC.

Finally, it is interesting to compare the full disjunctive closures of KROM, HORN, K/H, renH, AFF, with previous complete classes of representations for propositional logic, which have been considered as target classes for KC. One focuses on IP, DNF, PI, OBDD<, DNNF_T, and d-DNNF:

- IP satisfies all the queries but CT, and no transformation but CD, EN, SEN and ^BC. Hence each of the full disjunctive closures of KROM, HORN, AFF satisfies less queries than IP but they are incomparable w.r.t. transformations. Furthermore, IP is strictly less succinct than any of the full disjunctive closures of KROM, HORN, K/H, renH, AFF.

- DNF satisfies the same queries as any of the full disjunctive closures of KROM, HORN, K/H, renH, AFF, and the same transformations as KROM[v], HORN[v, ‖], and AFF[v]. Since it is strictly less succinct than any of the full disjunctive closures of KROM, HORN, K/H, renH, AFF, it appears as dominated by KROM[v], HORN[v, ‖], and AFF[v].

- PI satisfies all the queries but CT, and the transformations CD, FO, SFO, vBC. Hence, any of the full disjunctive closures of KROM, HORN, K/H, renH, AFF satisfies more transformations than PI. In addition, PI is incomparable w.r.t. succinctness with any of them.
• $\text{OBDD}_<$ satisfies all the queries and the transformations $\text{CD}$, $\text{SFO}$, $\text{SEN}$, $\wedge \text{BC}$, $\vee \text{BC}$, and $\neg \text{C}$. Hence it offers more queries than the full disjunctive closures of $\text{KROM}$, $\text{HORN}$, $\text{K/H}$, $\text{renH}$, $\text{AFF}$, but it is incomparable with any of them when transformations are considered. $\text{OBDD}_<$ is also incomparable w.r.t. succinctness with any of the full disjunctive closures of $\text{KROM}$, $\text{HORN}$, $\text{K/H}$, $\text{renH}$, $\text{AFF}$.

• $\text{DNNF}_T$ satisfies the same queries as any of the full disjunctive closures of $\text{KROM}$, $\text{HORN}$, $\text{K/H}$, $\text{renH}$, $\text{AFF}$, and the same transformations as $\text{KROM}[\lor]$, $\text{HORN}[\lor, \exists]$, and $\text{AFF}[\lor]$. It is incomparable w.r.t. succinctness with any of them.

• $d$-$\text{DNNF}$ satisfies all the queries but $\text{SE}$, and it is unknown whether it offers $\text{EQ}$. As to transformations, it offers only $\text{CD}$ (it is unknown whether it satisfies $\text{EN}$, $\text{SEN}$, or $\neg \text{C}$, but it is known that it does not satisfy the other transformations). Thus, the full disjunctive closures of $\text{KROM}$, $\text{HORN}$, $\text{K/H}$, $\text{renH}$, $\text{AFF}$ satisfy less queries than $d$-$\text{DNNF}$ but offer additional transformations. Furthermore, $d$-$\text{DNNF}$ is incomparable w.r.t. succinctness with any of the full disjunctive closures of $\text{KROM}$, $\text{HORN}$, $\text{K/H}$, $\text{renH}$, $\text{AFF}$ (unless the polynomial hierarchy collapses).

Thus, none of the full disjunctive closures of $\text{KROM}$, $\text{HORN}$, $\text{K/H}$, $\text{renH}$, and $\text{AFF}$ is strictly dominated by any of $\text{IP}$, $\text{DNF}$, $\text{PI}$, $\text{OBDD}_<$, $\text{DNNF}_T$, and $d$-$\text{DNNF}$, viewing the set of queries, the set of transformations and the succinctness relation as comparison criteria.

6. Conclusion and Perspectives

6.1. Conclusion

In the light of the results reported in the previous sections, the following conclusions can be drawn.

Generally speaking, the disjunctive closures of classes $\mathcal{L}$ of propositional representations appear as interesting target classes for KC when the application under consideration expects tractability for the queries and transformations $\text{CO}$, $\text{CD}$ and their consequences (e.g., $\text{CE}$, $\text{ME}$), as well as $\text{FO}$ and/or $\lor \text{C}$ (depending on the type of closure which is considered). Especially, as soon as $\mathcal{L}$ is stable by uniform renaming, the transformations $\text{FO}$, $\lor \text{C}$ are offered ”for free” by the full disjunctive closure $\mathcal{L}[\lor, \exists]$ (even if the underlying class $\mathcal{L}$ does not offer any of them),
while CO, CD are preserved by the closure. The other queries and transformations considered in the KC map are not guaranteed to be offered or to survive a disjunctive closure operation in the general case.

Considering specific disjunctive closures may allow for preserving additional queries or transformations, and for increasing the expressiveness of the underlying language. Thus, the disjunction closure and the full disjunctive closure of any language $L$ containing TERM are complete propositional languages, even if $L$ is not (KROM, HORN, K/H, renH, and AFF are such languages). Clearly enough, fully expressive propositional languages are highly expected by many applications.

Of course, it cannot be guaranteed in the general case that the size of a compiled form remains ”small enough” when a disjunctive closure is targeted. Nevertheless, every disjunctive closure of a class $L$ includes $L$ as a subset, hence applying a disjunctive closure principle to a class $L$ decreases neither the expressiveness nor the succinctness of $L$. Actually, applying any/both of those two principles may lead to new classes, which can prove strictly more expressive and strictly more succinct than the underlying class $L$. Thus, each of the disjunction closure and the full disjunctive closure of any of KROM, HORN, K/H, renH, and AFF is strictly more expressive than the underlying language. Furthermore, the full disjunctive closure $\text{HORN}[\forall, \exists]$ (resp. $\text{K/H}[\forall, \exists]$, $\text{renH}[\forall, \exists]$) is strictly more succinct than the corresponding disjunction closure $\text{HORN}[\forall]$ (resp. $\text{K/H}[\forall]$, $\text{renH}[\forall]$).

Now, from the application point of view, there are many important problems in AI and in other fields of Computer Science, where one is interested in encoding some pieces of information using representations for which CO, CD, FO and ME are computationally easy.

For instance, in model-based diagnosis, it makes sense to compile the description of the system to be diagnosed (during an off-line phase) in order to be able to generate efficiently consistency-based diagnoses, for a number of observations available on-line only [33, 41, 42]. Such diagnoses are the models of the system description, once conditioned by the given observation and then projected onto the variables expressing the components statuses (in the simplest case, faulty or not). Accordingly, if the system description has been compiled first into a representation which satisfies CO, CD, FO and ME, then the diagnoses can be computed in input-output polynomial time. Our results thus show full disjunctive closures of languages $L$ satisfying the stability by uniform renaming condition as valuable target languages for the compilation, as soon as $L$ satisfies CO and CD (which is the case for KROM, HORN, K/H, renH, and AFF).

In product configuration and interactive recommendation, it is also important
to offer some response-time guarantees to the front-end user, especially when the interaction is Web-based. In order to achieve this goal, an approach consists in compiling the product catalog into a propositional representation (the models of it representing the feasible products). Among the operations required by the configuration process are propagating the user's choices (the CD transformation), testing whether at least one feasible product is compatible with the user's choices (the CO query), and listing a fixed number of feasible products compatible with the user's choices (see e.g. [43, 44]). Often, the feasible products are described using two types of variables (or "codes", [45]): the customer variables – the variables the user controls – and the manufacturer control variables – which express some information related to the factory or to the distribution of the product, and are not available to the user. Thus, the manufacturer control variables must be forgotten from the representation before listing the solutions. Our results show that those operations can be achieved efficiently when the catalog has been compiled into a full disjunctive closure of a class $L$ of propositional representations, stable by uniform renaming, and satisfying CO and CD. In particular, the task of enumerating a preset number of solutions is feasible in polynomial time in this case (Algorithm 1 given in Section 6.2 is a polynomial delay enumeration procedure).

Beyond AI applications, enumerating models once projected on a given set of variables appears as a fundamental issue for a number of problems considered in software engineering and formal methods. Thus, in the setting of automatic case generation based on propositional logic, such models correspond to test cases [46]. The problem ALL-SAT (or "all-solutions" SAT) which consists in enumerating the assignments to "important" variables of a propositional representation, which can be extended to models, turns out to be very significant in symbolic model checking [47], which explains that dedicated algorithms have been developed for solving it [48]. Indeed, this problem is considered for predicate abstraction [49], and re-parameterization in symbolic simulation [50]. In reachability analysis, one is interested in computing the set of states reachable from (resp. leading to) a given set of states under a transition relation; this is called the image (resp. pre-image) computation problem. The transition relation $T$ can be modeled as a Boolean function $T$ over $X \cup Y \cup X'$, complete terms $\gamma_X$ over $X$ (resp. $X'$) are used to denote states before (resp. after) a transition and complete terms $\gamma_Y$ over $Y$ represent inputs making precise the transition. By construction, the models of $\exists Y. T|_{\gamma_X}$ (resp. $\exists Y. T|_{\gamma_X'}$) represent the image of $\gamma_X$ (resp. the pre-image of $\gamma_X'$) by $T$. The "important" variables are those of $X'$ (resp. $X$). Accordingly, many SAT solvers have been customized into ALL-SAT solvers precisely for computing images or pre-images (see e.g. [51, 52]) from CNF representations of transition relations. In
practice, such SAT-based approaches to symbolic model checking can prove much more efficient than OBDD\_<_based approaches on some instances, which coheres with the fact that the succinctness of OBDD\_<_ and the succinctness of CNF\_<_ are incomparable [53]. Interestingly, when $T$ is represented as a full disjunctive closure of a class $\mathcal{L}$ of propositional representations, stable by uniform renaming, and satisfying CO and CD, both the computation of $\exists Y. T|_{\gamma X}$ (resp. $\exists Y. T|_{\gamma X'}$) and the enumeration of its models can be achieved in polynomial time (in the size of the input plus the size of the output). Contrastingly, no response-time guarantee can be ensured in the general case for computing a single model when $T$ is represented as a CNF formula.

Thus, for each of the applications above, considering full disjunctive closures for the representation purpose can prove to be a reasonable choice.

6.2. Perspectives

This work calls for several perspectives.

One of them concerns the problem of closed-world reasoning. Indeed, the disjunction covers of HORN and renH are known as interesting target languages when propositional formulae are to be interpreted under some form of the closed-world assumption, like the extended closed-world assumption (ECWA) [54], the extended generalized closed-world assumption (EGCWA) [55], the generalized closed-world assumption (GCWA) [56] or the careful closed-world assumption (CCWA) [57]. To be more precise, though inference from a propositional formula interpreted under ECWA, EGCWA, GCWA or CCWA is $\Pi^p_2$-hard, its complexity is at most at the first level of the polynomial hierarchy when the formula belongs to HORN[\_V] or to renH[\_V] [58]. Furthermore, the complexity of inference under EGCWA falls down to P when HORN[\_V] formulae are considered, or when GCWA is considered and queries are limited to CNF formulae. Finally, it turns out that the complexity of closed-world reasoning is the same one for HORN[\_V] formulae and for DNF formulae, despite the fact that DNF is strictly less succinct than HORN[\_V]. It would be interesting to identify the complexity of closed-world reasoning for full disjunctive closures, especially those of HORN and renH.

Another important issue for further research is the design and the evaluation of compilers targeting the disjunctive closures introduced in the paper. Actually, compilers targeting some of those closures considered here do exist. Thus, Boufkhad et al. [6] present some compilation algorithms targeting KROM[\_V], HORN[\_V], K/H[\_V], and renH[\_V], and evaluate them on a number of benchmarks. While the obtained results show the feasibility of computing disjunction closure compilations, we can hardly use them to compare the practical significance of
the corresponding closures with \( \text{OBDD}_< \) and \( \text{DNNF}_+ \) for which some experimental results are also available. Indeed, the compilation algorithms given in [6] are based on an old-style DPLL SAT solver, and the performances of such solvers are dramatically overtaken by those of modern SAT solvers, based on a CDCL architecture.

Interestingly, Nishimura et al. [59] have shown that the problem of determining whether a given CNF formula \( \alpha \) has a strong KROM-backdoor set (resp. a strong HORN-backdoor set) containing at most \( k \) variables is fixed-parameter tractable with parameter \( k \). Similarly, Samer and Szeider [60]) have shown that the problem of determining whether a given CNF formula \( \alpha \) has a KROM-backdoor tree (resp. a HORN-backdoor tree) containing at most \( k \) leaves is fixed-parameter tractable with parameter \( k \). The algorithm given in [60] can be used to determine "efficiently" (i.e., for sufficiently "small" \( k \)) whether a KROM[\( \lor \)] compilation or a HORN[\( \lor \)] compilation of "reasonable" size (i.e., linear in \( k \) and the size of \( \alpha \)) exists. As mentioned in [60]: "There is some empirical evidence that real-world instances actually have small backdoor sets". Such instances also have "small" HORN[\( \lor \)] representations (hence, "small" K/H[\( \lor \)] and "small" renH[\( \lor \)] representations). This explains why it makes sense to develop new compilation algorithms targeting disjunction closures.

Incorporating existentially quantified variables in the representations during the compilation phase in order to generate full disjunctive closures also appears as an interesting perspective. Indeed, new variables can be introduced as "names" given to arbitrarily complex subformulae of the input formula (using equivalences); the point is that equivalence w.r.t. the input formula is preserved when such variables are existentially quantified. Taking advantage of it can dramatically reduce the size of the compiled forms (our succinctness results show that exponential gaps in the representation size can be achieved thanks to existential closure); the difficulty is to determine when introducing new variables (this is reminiscent to the general problem of lemmatization in automated reasoning).

Finally, the fact that each of KROM[\( \lor \)], HORN[\( \lor , \exists \)], and AFF[\( \lor \)] satisfies \( \land \text{BC} \) paves the way for bottom-up compilation algorithms for those classes. As noted in [18], this is important for applications from formal verification based on unbounded model checking which require bottom-up, incremental compilation of

---

Note by the way that determining whether "small" renH[\( \lor \)] representations which are not HORN[\( \lor \)] representations exist can be computationally demanding since the detection of a strong renH-backdoor set is W[2]-hard [61].
formulae, where pieces of information are compiled independently and then con-
joined together. This explains why \( \text{HORN}[\lor, \exists] \), and \( \text{AFF}[\lor] \) which offers \( \text{CD} \), \( \text{FO} \), \( \land BC \), and \( \text{ME} \) appear as valuable candidates for the image/pre-image com-
putation problem considered in reachability analysis, as discussed above. Indeed, \( \text{OBDD}_< \), which offers \( \text{CD} \), \( \text{FO} \), \( \land BC \), and \( \text{ME} \) as well, has been extensively used for the purpose of symbolic model checking [62]; furthermore, we have shown that the succinctness of \( \text{HORN}[\lor, \exists] \), and of \( \text{AFF}[\lor] \) are incomparable with the succinctness of \( \text{OBDD}_< \). Since each of \( \text{KROM}[\lor] \), \( \text{HORN}[\lor, \exists] \), and \( \text{AFF}[\lor] \) satisfies \( \text{CO} \) and includes \( \text{CLAUSE} \), getting \( \land BC \) is optimal in the sense that no class of propositional representations containing \( \text{CLAUSE} \) can satisfy both \( \land C \) and \( \text{CO} \), unless \( P = \text{NP} \).

Acknowledgments

We would like to thank the anonymous reviewers for many helpful comments.

References


[2] H. Fargier, P. Marquis, Extending the knowledge compilation map: Krom, 


[5] M. Cadoli, F. Donini, A survey on knowledge compilation, AI Communica-


[7] B. Selman, H. Kautz, Knowledge compilation and theory approximation, 


[38] G. Cantor, Beiträge zur begründung der transfiniten mengenlehre, Mathematische Annalen 49 (1897) 207–246.


Appendix: proofs

Proposition 1 For every subset $\mathcal{L}$, $\mathcal{L}'$ of $C$-$Q$DAG and every subset $\triangle_1$, $\triangle_2$ of $C \cup \{\exists, \forall\}$, we have:

0. $\mathcal{L} \subseteq \mathcal{L}[\triangle_1]$, and if $\mathcal{L} \subseteq \mathcal{L}'$, then $\mathcal{L}[\triangle_1] \subseteq \mathcal{L}'[\triangle_1]$.

1. $(\mathcal{L}[\triangle_1])[\triangle_2] \subseteq \mathcal{L}[\triangle_1 \cup \triangle_2]$.

2. $(\mathcal{L}[\triangle_1])[\triangle_1] = \mathcal{L}[\triangle_1]$.

3. If $\triangle_1 \subseteq \triangle_2$ then $\mathcal{L}[\triangle_1] \subseteq \mathcal{L}[\triangle_2]$.

4. If $\triangle_1 \subseteq \triangle_2$ then $(\mathcal{L}[\triangle_1])[\triangle_2] = \mathcal{L}[\triangle_2]$ and $(\mathcal{L}[\triangle_2])[\triangle_1] = \mathcal{L}[\triangle_2]$.

Proof:

0. Obvious.

1. $(\mathcal{L}[\triangle_1])[\triangle_2] \subseteq \mathcal{L}[\triangle_1 \cup \triangle_2]$ is immediate from Definition 9; indeed, the construction of any representation from $(\mathcal{L}[\triangle_1])[\triangle_2]$ requires only representations from $\mathcal{L}[\triangle_1]$ and operators in $\triangle_2$; and the construction of any representation from $\mathcal{L}[\triangle_1]$ requires only representations from $\mathcal{L}$ and operators in $\triangle_1$. Thus, if representations from $\mathcal{L}$ and operators in $\triangle_1 \cup \triangle_2$ are available, then any representation from $(\mathcal{L}[\triangle_1])[\triangle_2]$ can be generated.

2. $(\mathcal{L}[\triangle_1])[\triangle_1] = \mathcal{L}[\triangle_1]$: considering the inclusion reported at item 1. in this proof with $\triangle_2 = \triangle_1$, we get $(\mathcal{L}[\triangle_1])[\triangle_1] \subseteq \mathcal{L}[\triangle_1]$; the converse inclusion $\mathcal{L}[\triangle_1] \subseteq (\mathcal{L}[\triangle_1])[\triangle_1]$ follows from the inclusion at item 0. in this proof.

3. If $\triangle_1 \subseteq \triangle_2$ then $\mathcal{L}[\triangle_1] \subseteq \mathcal{L}[\triangle_2]$: the inclusion at item 0. in this proof shows that $\mathcal{L}[\triangle_1] \subseteq (\mathcal{L}[\triangle_1])[\triangle_2]$, and item 1. in this proof shows that $(\mathcal{L}[\triangle_1])[\triangle_2] \subseteq \mathcal{L}[\triangle_1 \cup \triangle_2]$. The fact that $\mathcal{L}[\triangle_1 \cup \triangle_2]$ is equal to $\mathcal{L}[\triangle_2]$ when $\triangle_1 \subseteq \triangle_2$ completes the proof.

4. Suppose that $\triangle_1 \subseteq \triangle_2$. Then, from the equality reported at item 2. in this proof, since $\triangle_1 \cup \triangle_2 = \triangle_2$, we have $(\mathcal{L}[\triangle_1])[\triangle_2] \subseteq \mathcal{L}[\triangle_2]$ and $(\mathcal{L}[\triangle_2])[\triangle_1] \subseteq \mathcal{L}[\triangle_2]$. Conversely, the inclusion at item 0. in this proof shows that $\mathcal{L}[\triangle_2] \subseteq (\mathcal{L}[\triangle_2])[\triangle_1]$; finally, $\mathcal{L}[\triangle_2] \subseteq (\mathcal{L}[\triangle_1])[\triangle_2]$ derives from the fact that $\mathcal{L} \subseteq \mathcal{L}[\triangle_1]$ (which is again ensured the inclusion at item 0. in this proof), and the implication reported at item 0. as well.
Proposition 2 Let \( L \) be any subset of \( C-\text{QDAG} \) s.t. \( L \) is stable by uniform renaming. We have:

- (\( L[\exists] \)[\( \lor \)] \( \sim_p \) \( (L[\forall])[\exists] \) \( \sim_p \) \( L[\forall, \exists] \)).
- (\( L[\forall] \)[\( \land \)] \( \sim_p \) \( (L[\forall])[\forall] \) \( \sim_p \) \( L[\land, \forall] \)).

Proof: We just prove the first point of the proposition; the second one is similar (by duality). The facts that \( L[\lor, \exists] \leq_p (L[\forall])[\exists] \) and \( L[\forall, \exists] \leq_p (L[\exists])[\forall] \) come immediately from the inclusions \( L[\lor, \exists] \supseteq (L[\forall])[\exists] \) and \( L[\forall, \exists] \supseteq (L[\exists])[\forall] \) (see item 1. in Proposition 1). It remains to show that \( L[\lor, \exists] \geq_p (L[\forall])[\exists] \) and \( L[\forall, \exists] \geq_p (L[\exists])[\forall] \).

- \( L[\lor, \exists] \geq_p (L[\forall])[\exists] \). This comes from the possibility to turn in polynomial time any \( L[\lor, \exists] \) representation into an equivalent, prenex, one while preserving the set of free variables. The proof is by structural induction. Let \( \alpha \) be any representation from \( L[\forall, \exists] \):
  - If \( \alpha \) is an \( L \) representation, then it is also an \( (L[\forall])[\exists] \) representation due to the inclusion \( (L[\forall])[\exists] \supseteq L \) which comes from item 0. in Proposition 1.
  - If \( \alpha \) is not an \( L \) representation and \( \alpha = \lor(\beta_1, \ldots, \beta_n) \) with \( \beta_i \in L[\forall, \exists] \) for \( i = 1, \ldots, n \), then by induction hypothesis, one can compute in polynomial time \( n \) representations \( \beta'_1, \ldots, \beta'_n \in (L[\forall])[\exists] \) such that for \( i = 1, \ldots, n \), we have \( \beta'_i \equiv \beta_i \). Hence, for \( i = 1, \ldots, n \), we can compute in polynomial time \( X_i \subseteq PS \) and \( \beta''_i \in L[\forall] \) such that \( \beta''_i \equiv \exists X_i, \beta'_i \) (especially, if \( \beta'_i \) is an \( L[\forall] \) representation, then we take \( X_i = \emptyset \)). For \( i = 1, \ldots, n \), let \( X'_i \) be a set of variables of \( PS \) which is disjoint with the set of variables occurring in \( \alpha \) and such that there exists a bijection between \( X'_i \) and \( X_i \). One can always find such a bijection since each \( \beta_i \) \( (i = 1, \ldots, n) \) belongs to \( L[\forall, \exists] \), which is stable by uniform renaming since \( L \) is so. Furthermore, since \( PS \) is countably infinite, we can always find sets \( X'_i \)

\footnote{A key observation here is that all \( \beta_i \) \( (i = 1, \ldots, n) \) are pairwise independent, i.e., they do not share any node and for \( i = 1, \ldots, n \), every arc reaching a node of \( \beta_i \) comes from a node of \( \beta_i \); if this was not the case, such a proof by structural induction would not work.}
so that for $i, j \in 1, \ldots, n$, if $i \neq j$ then $X_i \cap X'_j = \emptyset$. Now, for $i \in 1, \ldots, n$, let $\beta_i''[X_i \leftarrow X_i']$ be the representation obtained by replacing in a uniform way in $\beta_i''$ every occurrence of $x \in X_i$ by the corresponding variable $x' \in X'_i$. Clearly enough, such representations can be computed in polynomial time. Since quantified variables are dummy ones, we have $\exists x. \beta_i'' \equiv \exists x'. \beta_i''[X_i \leftarrow X_i']$. Hence, we have $\alpha \equiv \bigvee (\beta_1', \ldots, \beta_n') \equiv \bigvee (\exists X_1. \beta_1'', \ldots, \exists X_n. \beta_n'') \equiv \bigvee (\exists X_1. \beta_1'[X_1 \leftarrow X_1'], \ldots, \exists X_n. \beta_n'[X_n \leftarrow X_n'])$. Since for each $i \in 1, \ldots, n$, we have $\text{Var}(\beta_i'')[X_i \leftarrow X_i'] \cap \bigcup_{j=1}^{n} X_j' = \emptyset$, each $\exists X_i. \beta_i''[X_i \leftarrow X_i']$ is equivalent to $\exists \bigcup_{j=1}^{n} X_j'. \beta_i''[X_i \leftarrow X_i']$. Thus we get that $\alpha \equiv \bigvee (\exists \bigcup_{j=1}^{n} X_j'. \beta_1'[X_1 \leftarrow X_1'], \ldots, \exists \bigcup_{j=1}^{n} X_j'. \beta_n'[X_n \leftarrow X_n']) \equiv \exists \bigcup_{j=1}^{n} X_j'. \bigvee (\beta_1''[X_1 \leftarrow X_1'], \ldots, \beta_n''[X_n \leftarrow X_n'])$. Since $\forall (\beta''[X_1 \leftarrow X_1'], \ldots, \beta_n''[X_n \leftarrow X_n'])$ is an $L[\forall]$ representation, the conclusion follows. Note that the set of free variables of $\alpha$ is preserved by the translation.

- If $\alpha$ is not an $L$ representation and $\alpha = \exists x. \beta$ with $\beta \in L[\exists, \forall]$, then by induction hypothesis, one can compute in polynomial time a representation $\beta' \in (L[\forall])[\exists]$ such that $\beta' \equiv \beta$. Since $\exists x. \beta'$ is an $(L[\forall])[\exists]$ representation equivalent to $\alpha$, the conclusion follows. Again, the set of free variables of $\alpha$ is preserved by the translation.

- $L[\forall, \exists] \supseteq_p (L[\exists])[\forall]$. Again, the proof is by structural induction. Let $\alpha$ be any representation from $L[\forall, \exists]$:

  - If $\alpha$ is an $L$ representation, then it is also an $(L[\exists])[\forall]$ representation due to the inclusion $(L[\exists])[\forall] \supseteq L$ which comes from item 0. in Proposition 1.

  - If $\alpha = \bigvee (\beta_1, \ldots, \beta_n)$ with $\beta_i \in L[\forall, \exists]$ ($i \in 1, \ldots, n$), then by induction hypothesis, one can compute in polynomial time a representation $\beta'_i \in (L[\exists])[\forall]$ ($i \in 1, \ldots, n$) such that for each $i \in 1, \ldots, n$, $\beta'_i \equiv \beta_i$. Since $\forall (\beta'_1, \ldots, \beta_n)$ is an $(L[\exists])[\forall]$ representation equivalent to $\alpha$, the conclusion follows.

  - If $\alpha = \exists x. \beta$ with $\beta \in L[\forall, \exists]$, then by induction hypothesis, one can compute in polynomial time a representation $\beta' \in (L[\exists])[\forall]$ such that $\beta' \equiv \beta$. If $\beta'$ is an $L[\exists]$ representation, then $\exists x. \beta'$ also is an $L[\exists]$ representation; since it is equivalent to $\alpha$ and since $(L[\exists])[\forall] \supseteq L[\exists]$ (see item 0. in Proposition 1), the conclusion follows. Otherwise we have
\[ \beta' = \vee(\beta'_1, \ldots, \beta'_n) \] where \( \beta'_i \) is an \( \mathcal{L}[\exists] \) representation \( (i \in 1, \ldots, n) \).

By replacement, \( \alpha \) is equivalent to \( \exists x. \vee (\beta'_1, \ldots, \beta'_n) \), which is equivalent to \( \vee((\exists x. \beta'_1), \ldots, (\exists x. \beta'_n)) \). Since the latter representation is an \( (\mathcal{L}[\exists])[\vee] \) representation, the conclusion follows.

\[ \square \]

**Proposition 3** Let \( \mathcal{L} \) be any subset of \( C-QDAG \) s.t. \( \mathcal{L} \) is stable by uniform renaming.

- If \( \mathcal{L} \) satisfies \( CO \) (resp. \( CD \)), then \( \mathcal{L}[\vee], \mathcal{L}[\exists] \) and \( \mathcal{L}[\vee, \exists] \) satisfy \( CO \) (resp. \( CD \)).
- If \( \mathcal{L} \) satisfies \( CO \) and \( CD \), then \( \mathcal{L} \) satisfies \( CE \) and \( ME \).
- If \( \mathcal{L} \) satisfies \( CO \) and \( CD \), then \( \mathcal{L}, \mathcal{L}[\vee], \mathcal{L}[\exists] \) and \( \mathcal{L}[\vee, \exists] \) satisfy \( MC \).
- \( \mathcal{L}[\vee] \) and \( \mathcal{L}[\vee, \exists] \) satisfy \( \forall C \) (hence \( \forall BC \)) and \( \mathcal{L}[\exists] \) and \( \mathcal{L}[\vee, \exists] \) satisfy \( FO \) (hence \( SFO \)).
- If \( \mathcal{L} \) satisfies \( FO \) (resp. \( SFO \)), then \( \mathcal{L}[\vee] \) satisfies \( FO \) (resp. \( SFO \)).
- If \( \mathcal{L} \) satisfies \( \wedge C \) (resp. \( \wedge BC, \forall C, \forall BC \)), then \( \mathcal{L}[\exists] \) satisfies \( \wedge C \) (resp. \( \wedge BC, \forall C, \forall BC \)).

**Proof:**

- As to \( CO \), since \( \mathcal{L}[\vee] \subseteq \mathcal{L}[\vee, \exists] \) and \( \mathcal{L}[\exists] \subseteq \mathcal{L}[\vee, \exists] \), it is enough to show that \( \mathcal{L}[\vee, \exists] \) satisfies \( CO \). Let \( \alpha \) be any representation from \( \mathcal{L}[\vee, \exists] \); since \( \mathcal{L}[\vee, \exists] \sim_p (\mathcal{L}[\vee])[\exists] \) (cf. Proposition 2), we can compute in time polynomial in the size of \( \alpha \) an equivalent representation \( \beta = \exists X. \vee (\beta_1, \ldots, \beta_n) \) where \( X \) is a finite subset of \( PS \) and each \( \beta_i (i \in 1, \ldots, n) \) is an \( \mathcal{L} \) representation. We have that \( \alpha \) is consistent iff \( \beta \) is consistent iff \( \vee(\beta_1, \ldots, \beta_n) \) is consistent iff at least one \( \beta_i (i \in 1, \ldots, n) \) is consistent. Since the latter can be decided in polynomial time, the conclusion follows.

As to \( CD \), let \( \gamma \) be any consistent term. Let \( \alpha \) be an \( \mathcal{L}[\vee] \) representation; we have \( \alpha = \vee(\beta_1, \ldots, \beta_n) \) where each \( \beta_i (i \in 1, \ldots, n) \) is an \( \mathcal{L} \) representation. Since \( \wedge \) distributes over \( \vee \) and existential quantifications "distribute" over \( \vee \) as well, we have \( \exists \text{Var}(\gamma). (\alpha \wedge \gamma) \equiv \exists \text{Var}(\gamma). (\vee(\beta_1, \ldots, \beta_n) \wedge \gamma) \equiv \exists \text{Var}(\gamma). \wedge(\beta_1 \wedge \gamma, \ldots, \beta_n \wedge \gamma) \equiv \vee(\exists \text{Var}(\gamma). (\beta_1 \wedge \gamma), \ldots, \exists \text{Var}(\gamma). (\beta_n \wedge \gamma)) \).
\( \exists \) \( L \) associated in polynomial time with an equivalent \( \gamma \) which can be computed in time polynomial in the size of the input. Now let \( L \) be any \( L[\exists] \) representation; we have \( \alpha = \exists X.\beta \) where \( X \) is a finite subset of \( PS \) and \( \beta \) is an \( L \) representation.

We have \( \exists \text{Var}(\gamma).(\alpha \land \gamma) \equiv \exists \text{Var}(\gamma[X \leftarrow X']).((\exists X.\beta) \land \gamma[X \leftarrow X']) \) where \( \gamma[X \leftarrow X'] \) is the representation obtained by replacing in \( \gamma \) every variable \( x \in \text{Var}(\gamma) \cap X \) by a fresh variable \( x' \), not occurring in \( \beta \) or \( \gamma \). Since \( \text{Var}(\gamma[X \leftarrow X']) \cap X = \emptyset \), we have that \( \exists \text{Var}(\gamma[X \leftarrow X']).((\exists X.\beta) \land \gamma[X \leftarrow X']) \equiv \exists \text{Var}(\gamma[X \leftarrow X']) \cup X.(\beta \land \gamma[X \leftarrow X']) \equiv \exists X.(\exists \text{Var}(\gamma[X \leftarrow X']).(\beta \land \gamma[X \leftarrow X'])). \) If \( L \) satisfies \( CD \), then \( \exists \text{Var}(\gamma[X \leftarrow X']).(\beta \land \gamma[X \leftarrow X']) \) can be associated in polynomial time with an equivalent \( L \) representation \( \beta' \). Hence \( \exists \text{Var}(\gamma).(\alpha \land \gamma) \) is equivalent to the \( L[\exists] \) representation \( \exists X.\beta' \) which can be computed in time polynomial in the size of the input. Finally, let \( \alpha \) be an \( L[\lor, \exists] \) representation; since \( L[\lor, \exists] \sim_p (L[\lor])[\exists] \) (cf. Proposition 2), we can compute in time polynomial in the size of \( \alpha \) an equivalent representation \( \beta = \exists X.\lor(\beta_1, \ldots, \beta_n) \) where \( X \) is a finite subset of \( PS \) and each \( \beta_i \) \((i \in 1, \ldots, n)\) is an \( L \) representation. Then it is enough to combine the two previous proofs to get the desired result.

- We generalize some easy lemmata from [1] to the \( C-QDAG \) case. As to \( CE \), it is enough to observe that for any \( C-QDAG \) representation \( \alpha \) and any non-valid clause \( \delta \), we have \( \alpha \models \delta \) iff \( \alpha \land \lnot \delta \) is inconsistent iff \( \exists \text{Var}(\lnot \delta).(\alpha \land \lnot \delta) \) is inconsistent.

As to \( ME \), let \( \alpha \) be any \( L \) representation. Procedure 1 enumerates the models of \( \alpha \) over \( \text{Var}(\alpha) \). It amounts to searching a decision tree \( T \) in a depth-first manner. Each branch of \( T \) corresponds either to a model of \( \alpha \) over \( \text{Var}(\alpha) \), or to an implicant of \( \lnot \alpha \). Each model is represented as a set of literals over \( \text{Var}(\alpha) \). The procedure is called with \( \gamma = \emptyset \). Given a total, strict ordering over the variables of \( \text{Var}(\alpha) \), the function \( \text{first}(\alpha) \) at Line 4 returns the first variable of \( \alpha \) w.r.t. this ordering.

Procedure 1 first consists in testing whether \( \alpha \) is consistent (Line 1). If \( \alpha \) is inconsistent, then the procedure stops; otherwise, one checks whether \( \text{Var}(\alpha) \) is empty or not (Line 2). If this set is empty, then one returns the model of \( \alpha \) stored in the accumulator \( \gamma \) (Line 3). In the remaining case,
Algorithm 1: enumerate($\alpha, \gamma$)

input : an $L$ representation $\alpha$, and a set $\gamma$ of literals over $Var(\alpha)$

1 if $\alpha$ is consistent then
2   if $Var(\alpha) = \emptyset$ then
3     write($\gamma$)
4   else
5     $x \leftarrow \text{first}(Var(\alpha))$
6     enumerate($\alpha \mid x$, $\gamma \cup \{x\}$)
7     enumerate($\alpha \mid \neg x$, $\gamma \cup \{\neg x\}$)

one computes the first variable $x$ of $\alpha$ (Line 5). Afterwards, the procedure enumerates recursively all the models of $\alpha_{\mid x}$ by adding $x$ to the accumulator $\gamma$ (Line 6), then all the models of $\alpha_{\mid \neg x}$ by adding $\neg x$ to the accumulator $\gamma$ (Line 7). In both cases, a variable is removed (since $x \not\in Var(\alpha_{\mid x}) \cup Var(\alpha_{\mid \neg x})$), hence the number of recursive calls for each branch of $T$ cannot exceed the number of variables of $\alpha$. Furthermore, since $L$ satisfies CO and CD, the time spent between two successive calls is polynomial in the input size.

Procedure 1 is thus a polynomial delay model enumeration algorithm: a first model of $\alpha$ (when it exists) is generated in time polynomial in the size of the input, and after each model generation, the time needed to generate a further model (or to determine that no more models exist) also is polynomial in the size of the input. As a consequence, it runs in time polynomial in the size of the input plus the size of the output.

- Due to the inclusions $L \subseteq L[\lor], L \subseteq L[\exists] \subseteq L[\lor, \exists]$ (see Proposition 1), it is enough to show that $L[\lor, \exists]$ satisfies MC. Let $\alpha$ be any $L[\lor, \exists]$ representation. Since $L[\lor, \exists] \sim_p (L[\lor])[\exists]$ (cf. Proposition 2), we can compute in time polynomial in the size of $\alpha$ an equivalent representation $\beta = \exists X \lor (\beta_1, \ldots, \beta_n)$ where $X$ is a finite subset of $PS$ and each $\beta_i (i \in 1, \ldots, n)$ is an $L$ representation. Furthermore, we have $Var(\beta) = Var(\alpha)$. Let $\omega$ be any interpretation over $Var(\alpha)$ and let $\gamma$ be the consistent term (unique up to logical equivalence) such that $Var(\gamma) = Var(\alpha)$ and $\omega$ is a model of $\gamma$. We have $\omega \models \alpha$ iff $\gamma \land \beta$ is consistent iff $\gamma \land \lor(\beta_1, \ldots, \beta_n)$ is consistent iff there exists $i \in 1, \ldots, n$ such that $\gamma \land \beta_i$ is consistent iff
there exists $i \in 1, \ldots, n$ such that $\exists \text{Var}(\gamma). (\beta_i \land \gamma)$ is consistent. Since $\mathcal{L}$ satisfies CO and CD, the conclusion follows.

- The fact that $\mathcal{L}[\lor]$ and $\mathcal{L}[\lor, \exists]$ satisfy $\lor$C and $\mathcal{L}[\exists]$ and $\mathcal{L}[\lor, \exists]$ satisfy FO is obvious (by construction).

- We prove the FO case (for SFO just take $X$ as a singleton). Let $\alpha$ be a representation from $\mathcal{L}[\lor]$ and $X \subseteq PS$. By construction, $\alpha = \lor(\beta_1, \ldots, \beta_n)$ where each $\beta_i (i \in 1, \ldots, n)$ is an $\mathcal{L}$ representation. Since existential quantifications "distribute" over $\lor$, we have $\exists X. \alpha \equiv \lor(\exists X. \beta_1, \ldots, \exists X. \beta_n)$. Now, since $\mathcal{L}$ satisfies FO, with each $\exists X. \beta_i (i \in 1, \ldots, n)$ we can associate in polynomial time an equivalent $\mathcal{L}$ representation $\beta'_i$. Applying the replacement metatheorem, we get that $\exists X. \alpha \equiv \lor(\beta'_1, \ldots, \beta'_n)$. Since the $\mathcal{L}[\lor]$ representation $\lor(\beta'_1, \ldots, \beta'_n)$ can be computed in polynomial time in the size of $\alpha$ plus the size of $X$, the conclusion follows.

- We prove the $\land$C case. Let us consider $n$ representations $\alpha_1, \ldots, \alpha_n$ from $\mathcal{L}[\exists]$ where $\mathcal{L}$ satisfies $\land$C. By construction, for each $i \in 1, \ldots, n$, $\alpha_i$ is of the form $\exists X_i. \beta_i$ where $X_i$ is a finite subset of $PS$ and $\beta_i \in \mathcal{L}$. With each $\exists X_i. \beta_i$ we can associate in polynomial time the equivalent representation $\exists X_i^i. \beta_i[X_i \leftarrow X_i^i]$ obtained by renaming in a uniform way every occurrence of variable $x \in X_i$ by the fresh variable $x^i$. Whenever $\beta_i$ belongs to $\mathcal{L}$, $\beta_i[X_i \leftarrow X_i^i]$ belongs to $\mathcal{L}$ as well (due to the stability condition). From the replacement metatheorem, we get that $\land^n_{i=1} \alpha_i \equiv \land^n_{i=1} (\exists X_i. \beta_i) \equiv \land^n_{i=1} (\exists X_i^{i^i}. \beta_i[X_i \leftarrow X_i^i])$. By construction, we have $X_i^i \cap X_j^j = \emptyset$ when $i \neq j$. As a consequence, we have $\land^n_{i=1} (\exists X_i^{i^i}. \beta_i[X_i \leftarrow X_i^i]) \equiv \exists \bigcup^n_{i=1} X_i^{i^i}. (\land^n_{i=1} \beta_i[X_i \leftarrow X_i^i])$ Since $\mathcal{L}$ satisfies $\land$C, we can turn in polynomial time the representation $\land^n_{i=1} \beta_i[X_i \leftarrow X_i^i]$ into an equivalent representation $\beta$ from $\mathcal{L}$. Since $\land^n_{i=1} \alpha_i \equiv \exists \bigcup^n_{i=1} X_i^i. \beta$ and $\bigcup^n_{i=1} X_i^i. \beta$ is

---

9In classical propositional logic, this metatheorem states that if $\beta$ is a subformula of a propositional formula $\alpha$ and $\beta'$ is a formula equivalent to $\beta$, then the formula obtained by replacing in $\alpha$ the subformula $\beta$ by $\beta'$ is a formula equivalent to $\alpha$ [63] (this comes directly from the truth-functionality of the connectives); this metatheorem also holds for quantified formulae and can be generalized to the case of DAG-based representations (under some conditions); more precisely, given any node $N$ of a $C$-$\text{QDAG}$ representation $\alpha$ let $\beta_N$ be the subgraph of $\alpha$ given by the set $S_N$ of nodes of $\alpha$ reachable from $N$; if every arc of $\alpha$ having its extremity in $S_N \setminus \{N\}$ also has its origin in $S_N$, then for every $C$-$\text{QDAG}$ representation $\beta'$ equivalent to $\beta$, the $C$-$\text{QDAG}$ representation obtained by removing in $\alpha$ every node and every arc of $\beta$, and redirecting the arcs entering $N$ to the root of $\beta'$ is a representation equivalent to $\alpha$. 

51
an $\mathcal{L}[\exists]$ representation, the conclusion follows. The proof is similar for the remaining cases ($\land BC$, $\lor C$, $\lor BC$).

\begin{proof}
These polynomial equivalences come easily from the fact that each of KROM and AFF satisfies FO (cf. Proposition 6), plus the fact that existential quantifications "distribute" over disjunctions.
\end{proof}

\begin{proposition}
The results in Table 1 hold.
\end{proposition}

\begin{proof}
CO It is well-known that each of KROM, HORN, renH, AFF satisfies CO (cf. [64, 65, 66, 67, 25]). Since deciding whether a $C$-QDAG representation is in KROM (resp. HORN) can be done in polynomial time, we get that K/H satisfies CO. Then point 1. of Proposition 3 allows to conclude that each of the $[\lor, \exists]$ and $[\lor, \exists]$ closures of those languages satisfies CO as well.

VA KROM, HORN, K/H and renH satisfy VA since they are subsets of CNF and CNF satisfies VA. AFF satisfies VA since it satisfies CT (indeed, an AFF formula $\alpha$ is valid if and only if it has $2^n$ models where $n$ is the cardinality of $\text{Var}(\alpha)$).

As to renH[$\exists$], K/H[$\exists$] and HORN[$\exists$], the results hold since each of these languages satisfies IM. Obviously, every subset $\mathcal{L}$ of $C$-QDAG which satisfies IM satisfies VA as well (indeed, $\alpha \in \mathcal{L}$ is valid iff it is implied by the term $\top$). Since the proof that each of renH[$\exists$], K/H[$\exists$] and HORN[$\exists$] satisfies IM relies on the fact that HORN[$\exists$] satisfies VA, it just remains to show it. This is easy since a formula $\alpha$ from HORN[$\exists$] is valid if and only if its universal closure is valid. The fact that the validity problem for closed, prenex
KROM, HORN, K/H, AFF, renH, their disjunction, existential and full disjunctive closures and the corresponding polynomial-time queries. √ means “satisfies” and ◦ means “does not satisfy unless P = NP.”

quantified Boolean formulae with a HORN matrix is in P [68] concludes the proof.

Finally, none of KROM[V], HORN[V], K/H[V], renH[V], AFF[V], HORN[V, ∃], K/H[V, ∃], renH[V, ∃] satisfies VA unless P = NP since each of those languages includes DNF as a subset and DNF does not satisfy VA unless P = NP.

CE, ME The results come directly from the second item of Proposition 3, given that each of the sixteen languages considered here satisfies both CO and CD.

IM As to KROM, HORN, K/H, renH, and AFF, the results come from the fact that if a subset of C-QDAG satisfies VA and CD, then it satisfies IM (this slightly extends Lemma A.7 from [1] to C-QDAG representations).

Consider now the case of renH[∃], K/H[∃] and HORN[∃]. Since each of K/H[∃] and HORN[∃] is polynomially translatable into renH[∃], it is enough to prove the result for renH[∃]. We first show that the implicant problem for renH[∃] formulae can be reduced in polynomial time into the
implicant problem for HORN[∃] formulae. Let ∃X.α be a renH[∃] formula such that α is a renH formula, and let γ be a term. Let V be any Horn renaming for α. We have γ ⊨ ∃X.α iff γ ⇒ (∃X.α) is valid.

Now, viewing V as a substitution, one can take advantage of the substitution metatheorem for propositional logic. This theorem (see e.g., [63]) states that for any propositional formula Σ and any substitution σ (a mapping which replaces each variable by a formula), if Σ is valid, then σ(Σ) is valid. With Σ = γ ⇒ (∃X.α) and σ = V, we get that if γ ⇒ (∃X.α) is valid, then V(γ) ⇒ (V(∃X.α)) is valid. Since for every formula β, V(V(β)) ≡ β, we also get that if V(γ ⇒ (∃X.α)) is valid, then γ ⇒ (∃X.α) is valid. Altogether, we get that γ ⇒ (∃X.α) is valid iff V(γ ⇒ (∃X.α)) is valid. Now, V(γ ⇒ (∃X.α)) is valid iff V(γ) ⇒ V(∃X.α) is valid iff V(γ) ⊨ V(∃X.α).

Let ω be any interpretation over Var(α) ∪ X. Since for every variable x, V(x) is equal to x or is equal to ¬x, V(ω) can be viewed as well as an interpretation over Var(α) ∪ X. We have ω ⊨ V(∃X.α) iff V(ω) ⊨ ∃X.α (using the substitution theorem and the fact that for every formula β, V(V(β)) ≡ β) iff there exists an interpretation ω′ over Var(α) ∪ X such that ω′ ⊨ α and ∀y ∈ (Var(α) ∪ X) \ X, V(ω)(y) = ω′(y) (by definition of ∃X.α) iff there exists an interpretation V(ω′) over Var(α) ∪ X such that V(ω′) ⊨ V(α) and ∀y ∈ (Var(α) ∪ X) \ X, V(V(ω′))(y) = V(V(ω′))(y). Since V(V(ω)) = ω, this is equivalent to state that ω is a model of ∃X.V(α). As a consequence, we have V(∃X.α) ≡ ∃X.V(α).

Accordingly, γ is an implicant of the renH[∃] formula ∃X.α iff the term V(γ) is an implicant of the HORN[∃] formula ∃X.V(α). As explained above (see the VA point in the proof), the fact that HORN[∃] satisfies CD and VA shows that it satisfies IM as well. Given that a Horn renaming V for α can be computed in polynomial time given α, and that V(γ) (resp. V(α)) can be computed in polynomial time from γ (resp. α) once V has been computed, the fact that HORN[∃] satisfies IM shows that renH[∃] satisfies IM as well.

Finally, none of KROM[∨], HORN[∨], K/H[∨], renH[∨], AFF[∨], HORN[∨, ∃], K/H[∨, ∃], renH[∨, ∃] satisfies IM unless P = NP, since none of them satisfies VA unless P = NP.

SE Determining whether a KROM (resp. HORN, K/H, renH) formula β is a logical consequence of a KROM (resp. HORN, K/H, renH) formula α amounts to determining whether every clause of β is a logical consequence of α. The fact that each of KROM, HORN, K/H and renH satisfy CE completes
the proof for those four languages. As to AFF, determining whether an AFF formula $\beta$ is a logical consequence of an AFF formula $\alpha$ amounts to determining whether every XOR-clause of $\beta$ is a logical consequence of $\alpha$. Now, a XOR-clause $l_1 \oplus \cdots \oplus l_n$ is a logical consequence of an AFF formula $\alpha$ if and only if the AFF formula $\alpha \land (l_1 \oplus \cdots \oplus l_n \oplus \top)$ is inconsistent. The fact that AFF satisfies CO concludes the proof for AFF.

As to renH[\exists], K/H[\exists] and HORN[\exists], it is enough to prove the result for HORN[\exists] since this language is included in the two remaining ones. Let $\alpha$ be a CNF formula over $n$ variables $x_1, \ldots, x_n$. Let $\alpha'$ be the HORN formula obtained by replacing each positive literal $x_i$ in $\alpha$ by the negative literal $\neg x'_i$ (where each $x'_i$ is a fresh variable), conjoined with $n$ additional clauses $\neg x_i \lor \neg x'_i$ ($i \in 1, \ldots, n$). Let $\beta'$ be the KROM formula $\land_{i=1}^{n} (x_i \lor x'_i)$. By construction, $\alpha$ is inconsistent iff $\alpha' \land \beta'$ is inconsistent iff $\alpha' \models \neg \beta'$. $\neg \beta'$ is equivalent to $\lor_{i=1}^{n} (\neg x_i \land \neg x'_i)$, which in turn is equivalent to the formula $\gamma' = \exists \{y_1, \ldots, y_n\} ((\neg y_1 \lor \cdots \lor \neg y_n) \land \lor_{i=1}^{n} ((y_i \lor \neg x_i) \land (y_i \lor \neg x'_i)))$ (where each $y_i$ is a fresh variable). The fact that $\alpha'$ and $\gamma'$ are HORN[\exists] formulae which can be computed in time polynomial in the size of $\alpha$ shows the coNP-hardness of the sentential entailment problem for HORN[\exists] formulae and concludes the proof.

Finally, none of KROM[\lor], HORN[\lor], K/H[\lor], renH[\lor], AFF[\lor], HORN[\lor, \exists], K/H[\lor, \exists], renH[\lor, \exists] satisfies SE unless P = NP; the fact that $\top$ is a formula from each of these languages and that $\alpha \in C$ is a logical consequence of an AFF formula $\alpha$ that $\top$ is a formula from each of these languages and that $\alpha \in C$-QDAG is valid iff $\top \models \alpha$ concludes the proof.

**EQ** Each of KROM, HORN, K/H, renH, and AFF satisfies EQ since it satisfies SE.

As to renH[\exists], K/H[\exists] and HORN[\exists]: for every formulae $\alpha'$ and $\gamma'$ from C-QDAG we have that $\alpha' \models \gamma'$ iff $\alpha' \land \gamma' \equiv \alpha'$. Consider now the formulae $\alpha'$ and $\gamma'$ used for proving that none of renH[\exists], K/H[\exists] and HORN[\exists] satisfies SE unless P = NP (see the item SE in this proof). Since none of the $y_i$ variables occurs in $\alpha'$, the formula $\alpha' \land \gamma'$ can be turned in linear time into the equivalent formula $\exists \{y_1, \ldots, y_n\}. (\alpha' \land ((\neg y_1 \lor \cdots \lor \neg y_n) \land \lor_{i=1}^{n} ((y_i \lor \neg x_i) \land (y_i \lor \neg x'_i))))$, which is a HORN[\exists] formula. This concludes the proof.

Finally, none of KROM[\lor], HORN[\lor], K/H[\lor], renH[\lor], AFF[\lor], HORN[\lor, \exists], K/H[\lor, \exists], renH[\lor, \exists] satisfies EQ unless P = NP; the fact that $\top$ is a formula from each of these languages and that $\alpha \in C$-QDAG is valid iff $\top \models \alpha$ concludes the proof.
languages and that $\alpha \in C^\neg \text{QDAG}$ is valid iff $\top \equiv \alpha$ concludes the proof.

**CT** The result for **AFF** is proven in [69]. The results for all the remaining languages come from the fact that the language of negative Krom formulae (i.e., the set of all conjunctions of negative, binary clauses) is included into each language among **KROM**, **HORN**, **K/H**, **renH**, **KROM[∨]**, **HORN[∨]**, **K/H[∨]**, **renH[∨]**, **HORN[∨,∃]**, **K/H[∨,∃]**, **renH[∨,∃]**; furthermore, DNF is included in **AFF[∨]** since each term is an **AFF** formula. The fact that none of the language of negative Krom formulae and DNF satisfies CT [70] concludes the proof.

**MC** The results come directly from the third item of Proposition 3, given that each of **KROM**, **HORN**, **K/H**, **renH**, and **AFF** satisfies both **CO** and **CD**.

**Proposition 6** The results in Table 2 hold.

<table>
<thead>
<tr>
<th></th>
<th>CD</th>
<th>FO</th>
<th>SFO</th>
<th>EN</th>
<th>SEN</th>
<th>∧C</th>
<th>∧BC</th>
<th>∨C</th>
<th>∨BC</th>
<th>¬C</th>
</tr>
</thead>
<tbody>
<tr>
<td>renH[∨,∃]</td>
<td>√</td>
<td>√</td>
<td>√</td>
<td>o</td>
<td>o</td>
<td>o</td>
<td>o</td>
<td>√</td>
<td>√</td>
<td>o</td>
</tr>
<tr>
<td>K/H[∨,∃]</td>
<td>√</td>
<td>√</td>
<td>√</td>
<td>o</td>
<td>√</td>
<td>o</td>
<td>o</td>
<td>√</td>
<td>√</td>
<td>o</td>
</tr>
<tr>
<td>HORN[∨,∃]</td>
<td>√</td>
<td>√</td>
<td>√</td>
<td>o</td>
<td>√</td>
<td>o</td>
<td>√</td>
<td>√</td>
<td>o</td>
<td></td>
</tr>
<tr>
<td><strong>AFF[∨]</strong></td>
<td>√</td>
<td>√</td>
<td>√</td>
<td>o</td>
<td>√</td>
<td>o</td>
<td>√</td>
<td>√</td>
<td>o</td>
<td></td>
</tr>
<tr>
<td>renH[∨]</td>
<td>√</td>
<td>●</td>
<td>√</td>
<td>o</td>
<td>o</td>
<td>o</td>
<td>o</td>
<td>√</td>
<td>√</td>
<td>o</td>
</tr>
<tr>
<td>K/H[∨]</td>
<td>√</td>
<td>●</td>
<td>√</td>
<td>o</td>
<td>√</td>
<td>o</td>
<td>√</td>
<td>√</td>
<td>●</td>
<td></td>
</tr>
<tr>
<td>HORN[∨]</td>
<td>√</td>
<td>●</td>
<td>√</td>
<td>o</td>
<td>√</td>
<td>o</td>
<td>√</td>
<td>√</td>
<td>●</td>
<td></td>
</tr>
<tr>
<td>KROM[∨]</td>
<td>√</td>
<td>√</td>
<td>√</td>
<td>o</td>
<td>√</td>
<td>o</td>
<td>√</td>
<td>√</td>
<td>●</td>
<td></td>
</tr>
<tr>
<td>renH[∃]</td>
<td>√</td>
<td>√</td>
<td>√</td>
<td>√</td>
<td>√</td>
<td>√</td>
<td>√</td>
<td>!</td>
<td>!</td>
<td>!</td>
</tr>
<tr>
<td>K/H[∃]</td>
<td>√</td>
<td>√</td>
<td>√</td>
<td>√</td>
<td>√</td>
<td>√</td>
<td>!</td>
<td>!</td>
<td>!</td>
<td>!</td>
</tr>
<tr>
<td>HORN[∃]</td>
<td>√</td>
<td>√</td>
<td>√</td>
<td>√</td>
<td>√</td>
<td>√</td>
<td>!</td>
<td>!</td>
<td>!</td>
<td>!</td>
</tr>
<tr>
<td><strong>AFF</strong></td>
<td>√</td>
<td>√</td>
<td>√</td>
<td>√</td>
<td>√</td>
<td>√</td>
<td>!</td>
<td>!</td>
<td>!</td>
<td>!</td>
</tr>
<tr>
<td>renH</td>
<td>√</td>
<td>●</td>
<td>√</td>
<td>√</td>
<td>!</td>
<td>!</td>
<td>!</td>
<td>!</td>
<td>!</td>
<td>!</td>
</tr>
<tr>
<td>K/H</td>
<td>√</td>
<td>●</td>
<td>√</td>
<td>√</td>
<td>!</td>
<td>!</td>
<td>!</td>
<td>!</td>
<td>!</td>
<td>!</td>
</tr>
<tr>
<td>HORN</td>
<td>√</td>
<td>●</td>
<td>√</td>
<td>√</td>
<td>!</td>
<td>!</td>
<td>!</td>
<td>!</td>
<td>!</td>
<td>!</td>
</tr>
<tr>
<td>KROM</td>
<td>√</td>
<td>√</td>
<td>√</td>
<td>√</td>
<td>!</td>
<td>!</td>
<td>!</td>
<td>!</td>
<td>!</td>
<td>!</td>
</tr>
</tbody>
</table>

KROM, HORN, K/H, AFF, renH, their disjunction, existential and full disjunctive closures and the corresponding polynomial-time transformations. √ means “satisfies,” ● means “does not satisfy,” while o means “does not satisfy unless P=NP.”! means that the transformation is not always feasible within the language.
Proof:

**CD** When \( \alpha \) is a CNF formula and \( \gamma \) is a term, a CNF formula \( \beta \) equivalent to \( \alpha \mid \gamma \) can be computed in time polynomial in the size of \( \alpha \) plus the size of \( \gamma \) by removing from \( \alpha \) every clause containing a literal \( l \) from \( \gamma \) while removing the complementary literal \( \overline{l} \) from every clause of \( \alpha \) containing it. Obviously enough, removing clauses and shortening clauses are two internal laws in the languages \( \text{KROM} \) and \( \text{HORN} \). This shows that \( \text{KROM}, \text{HORN} \) and \( \text{K/H} \) satisfy **CD**. Similarly, when \( \alpha \) is an AFF formula and \( \gamma \) is a term (viewed as a set of literals), an AFF formula \( \beta \) equivalent to \( \alpha \mid \gamma \) can be computed in time polynomial in the size of \( \alpha \) plus the size of \( \gamma \) by replacing in \( \alpha \) every occurrence of a literal \( l \) by \( \top \) when \( l \) belongs to \( \gamma \) and by \( \bot \) when \( \overline{l} \) belongs to \( \gamma \). As to \( \text{renH} \), it is not hard to see that if \( V \) is a Horn renaming for a \( \text{renH} \) formula \( \alpha \) then for any term \( \gamma \), \( V \) also is Horn renaming for the formula \( \beta \) as defined above. Hence \( \text{renH} \) also satisfies **CD**.

Then point 1. of Proposition 3 allows to conclude that each of the \( [\lor], [\exists], \) and \( [\lor, \exists] \) closures of those languages satisfies **CD** as well.

**FO** Each of \( \text{HORN}[\exists], \text{K/H}[\exists], \text{renH}[\exists], \text{HORN}[\lor, \exists], \text{K/H}[\lor, \exists], \text{renH}[\lor, \exists] \) obviously satisfies **FO** since such a transformation can be done in an implicit way in each of those languages.

As to \( \text{KROM} \), it is well-known that the set of prime implicates of a \( \text{KROM} \) formula \( \alpha \) can be computed in time polynomial in the size of \( \alpha \) and that each such prime implicate is a binary clause (see [71]). Furthermore, the prime implicates of \( \exists X.\alpha \) with \( X \subseteq PS \) are the prime implicates of \( \alpha \) which do not contain any atom from \( X \) (Proposition 55 in [71]), showing in particular that \( \text{PI} \) satisfies **FO**. Together, this shows that \( \text{KROM} \) satisfies **FO**.

The fact that \( \text{AFF} \) satisfies **FO** is given by Lemma 1 from [72].

Now, taking advantage of the fact that for any \( C\text{-QDAG} \) representation of the form \( \lor(\alpha_1, \ldots, \alpha_n) \) and any finite subset \( X \) of \( PS \) \( \exists X.\lor(\alpha_1, \ldots, \alpha_n) \) is logically equivalent to \( \lor(\exists X.\alpha_1, \ldots, \exists X.\alpha_n) \), we get that each of \( \text{KROM}[\lor] \) and \( \text{AFF}[\lor] \) satisfies **FO**.

It remains to consider the cases of \( \text{HORN}, \text{K/H}, \text{renH} \) and of their disjunction closures. Consider the \( \text{HORN} \) formula \( \alpha_n = (\lor_{i=1}^n \neg x_i) \land \land_{i=1}^n (x_i \lor \neg y_i) \land (x_i \lor \neg z_i) \) and the set \( X_n = \{x_1, \ldots, x_n\} \) of atoms. Every clause of the form \( \lor_{i=1}^n \neg l_i \) where \( l \) is \( y \) or \( z \) is an essential prime implicate of \( \exists X_n.\alpha_n \) and there are \( 2^n \) such clauses. This shows that \( \exists X_n.\alpha_n \) has only exponential
size CNF representations. Thus HORN does not satisfy FO. Since \( \alpha_n \) also is a K/H formula and a renH formula, we also get that none of K/H and renH satisfies FO.

Finally, the fact that HORN[\( \lor \)] (resp. K/H[\( \lor \)], renH[\( \lor \)]) does not satisfy FO comes from the fact that HORN[\( \lor \), \( \exists \)] \( \preceq_s \) HORN[\( \lor \)] (resp. K/H[\( \lor \), \( \exists \)] \( \preceq_s \) renH[\( \lor \)]). Let us consider the HORN case (the other cases are similar): forgetting a set of variables \( X \) in a HORN[\( \lor \)] formula \( \alpha \) amounts to computing a HORN[\( \lor \)] formula equivalent to the HORN[\( \lor \), \( \exists \)] formula \( \exists X.\alpha \). If HORN[\( \lor \)] would satisfy FO, then every HORN[\( \lor \), \( \exists \)] formula \( \exists X.\alpha \) could be turned in polynomial time into an equivalent HORN[\( \lor \)] formula. Since HORN[\( \lor \), \( \exists \)] \( \not\preceq_p \) HORN[\( \lor \)], we would have HORN[\( \lor \), \( \exists \)] \( \not\preceq_s \) HORN[\( \lor \)]. But this conflicts with the fact that HORN[\( \lor \)] \( \not\preceq_s \) HORN[\( \lor \), \( \exists \)] (in a nutshell, if no polynomial-space translation exists, then no polynomial-time translation exists).

**SFO** Obviously, every language satisfying FO satisfies SFO as well. Hence it is enough to consider the cases of HORN, K/H, renH and of their disjunction closures.

Let us consider first the HORN and renH cases. For any CNF formula \( \alpha \) (viewed as a set of clauses) and a propositional variable \( x \in PS \), one can compute from \( \alpha \) in polynomial time the following three sets of clauses \( \alpha^* \), \( \alpha^+ \), and \( \alpha^- \): first remove from \( \alpha \) every valid clause to get a set of clauses \( \alpha' \); now, compute \( \alpha^* \) as the set of clauses of \( \alpha' \) not containing \( x \) as a variable, \( \alpha^+ \) as the set of clauses of \( \alpha' \) containing \( x \) as a (positive) literal, from which \( x \) is removed, and compute \( \alpha^- \) as the set of clauses of \( \alpha' \) containing \( \neg x \) as a (negative) literal from which \( \neg x \) is removed. By construction, the conjunction \( \beta \) of clauses from \( \alpha^* \cup \{ \delta^+ \lor \delta^- \mid \delta^+ \in \alpha^+, \delta^- \in \alpha^- \} \) is a CNF formula equivalent to \( (\alpha \mid \neg x) \lor (\alpha \mid x) \), hence equivalent to \( \exists x.\alpha \). Since none of \( \alpha^+ \) and \( \alpha^- \) can contain more clauses or more literals than \( \alpha \), it comes that \( \beta \) can be computed in time polynomial in the size of \( \alpha \). It remains to show that if \( \alpha \) is HORN (resp. renH) then the corresponding \( \beta \) is HORN (resp. renH). Assume that \( \alpha \) is HORN. Then every clause from \( \alpha^* \) is a Horn clause; furthermore, by construction every clause \( \delta^+ \in \alpha^+ \) is a negative clause and every clause \( \delta^- \in \alpha^- \) is a Horn clause; hence, every clause of the form \( \delta^+ \lor \delta^- \) is a Horn clause. Similarly, if \( \alpha \) is renH and \( V \) is any Horn renaming for it, then \( V \) also is a Horn renaming for the corresponding \( \beta \). Hence HORN and renH satisfy SFO.
Since both KROM and HORN satisfy SFO, K/H satisfies SFO as well.

Finally, given that for any $C\neg Dag$ representation $\alpha$ and any atom $x \in PS$, we have $\exists x.\alpha \equiv (\exists x.(\alpha \land \neg x)) \lor \exists x.(\alpha \land x)$, the results for HORN[$\forall$], K/H[$\forall$], and renH[$\forall$] come that each of these languages satisfies CD and $\lor$ BC.

EN For any $C\neg Dag$ representations $\alpha$ and $\beta$ and any finite subset $X$ of $PS$ we have the equivalence $\forall X.(\alpha \land \beta) \equiv (\forall X.\alpha) \land (\forall X.\beta)$. Furthermore, when $\delta$ is a clause, $\forall X.\delta$ is equivalent to the clause obtained by removing from $\delta$ every literal $l$ such that $\text{var}(l) \in X$. Since removing literals from a KROM (resp. HORN) clause leads to a KROM (resp. HORN) clause, altogether we get that each of KROM and HORN satisfies EN, and this shows that K/H satisfies EN as well. Now, if $\alpha$ is a renH formula and $V$ is a Horn renaming for it, then the formula obtained by removing in every clause of $\alpha$ every literal built up from a variable of $X$ still is a renH formula (indeed, $V$ is still a Horn renaming for it). Hence, renH also satisfies EN. Let us consider now the case of an AFF formula $\alpha$. We assume w.l.o.g. that $\alpha$ is simplified, i.e., for every XOR-clause $\delta = l_1 \oplus \ldots \oplus l_k$ of $\alpha$, either $\delta$ reduces to $\bot$, or every literal in $\delta$ is positive or equal to $T$ and $\delta$ does not contain more than one occurrence of any variable and of $T$ (if this is not the case it is sufficient to exploit the equivalences $\neg x \equiv x \oplus T$, $\beta \oplus \beta \equiv \bot$, $\beta \oplus \bot \equiv \beta$ to render $\alpha$ simplified while preserving logical equivalence); it is easy to check that if $\alpha$ is a simplified AFF formula containing a variable from $X$, then $\forall X.\alpha$ is equivalent to $\bot$, otherwise $\forall X.\delta$ is equivalent to $\alpha$. Hence, AFF satisfies EN.

The fact that HORN[$\exists$] satisfies EN is a consequence of Corollary 11 from [73]. Since KROM[$\exists$] $\sim_p$ KROM and KROM satisfies EN, as a consequence, we also have that K/H[$\exists$] satisfies EN.

As to the case of renH[$\exists$], let us consider a renH[$\exists$] formula $\alpha = \exists X.\beta$. Let $V$ be a Horn renaming for $\beta$. Since HORN[$\exists$] satisfies EN, for every finite subset $Y$ of $PS$, the formula $\forall Y.(\exists X.V(\beta))$ can be turned in polynomial time into an equivalent formula $\exists Z.\gamma$ from HORN[$\exists$]. From the substitution metatheorem, we have $V (\forall Y.(\exists X.V(\beta))) \equiv V (\exists Z.\gamma)$. Hence, we have $\forall Y.(\exists X.V(\beta)) \equiv \exists Z.V(\gamma)$. Since $V(\beta) = \beta$, we get that $\forall Y.(\exists X.\beta) \equiv \exists Z.V(\gamma)$. Clearly, $\exists Z.V(\gamma)$ is a renH[$\exists$] formula; indeed, $V(\gamma)$ is a renH formula since $V(V(\gamma)) = \gamma$ is a HORN formula. Since $\exists Z.V(\gamma)$ can be computed in polynomial time from $\forall Y.(\exists X.\beta)$, we get that
\text{renH}[\exists] \text{satisfies EN.}

Finally, for any $C \text{-QDAG}$ representation $\alpha$ and any finite subset $X$ of $PS$ we have that $\alpha$ is valid iff $\forall \text{Var}(\alpha).\alpha$ is valid iff $\forall \text{Var}(\alpha).\alpha$ is consistent (since $\forall \text{Var}(\alpha).\alpha$ has no free variable, it is equivalent to $\top$ or to $\bot$, hence it is consistent precisely when it is valid). Hence every language satisfying CO but not satisfying VA unless $P = NP$ cannot satisfy EN unless $P = NP$. This is the case for each language among $\text{KROM}[\lor]$, $\text{HORN[\lor]}$, $\text{K/H[\lor]}$, $\text{renH}[\lor]$, $\text{AFF[\lor]}$, $\text{HORN[\lor, \exists]}$, $\text{K/H[\lor, \exists]}$, $\text{renH[\lor, \exists]}$.

\textbf{SEN} Every language satisfying EN also satisfies SEN. Hence, each of $\text{KROM}$, $\text{HORN}$, $\text{K/H}$, $\text{renH}$, $\text{HORN[\exists]}$, $\text{K/H[\exists]}$, $\text{renH[\exists]}$, $\text{AFF}$ satisfies SEN. Furthermore, since for any $C \text{-QDAG}$ representation $\alpha$ and a variable $x \in PS$, we have $\forall x.\alpha \equiv \alpha_{|x} \land \alpha_{|\neg x}$, every language satisfying both CD and $\land \text{ABC}$ also satisfies SEN. Hence each of $\text{HORN[\lor]}$, $\text{KROM[\lor]}$, $\text{HORN[\lor, \exists]}$ satisfies SEN. Since each of $\text{HORN[\lor]}$, $\text{KROM[\lor]}$ satisfies SEN, we also have that $\text{K/H[\lor]}$ satisfies SEN. Similarly, since each of $\text{HORN[\lor, \exists]}$, $\text{KROM[\lor, \exists]}$ ($\sim_p \text{KROM[\lor]}$) satisfies SEN, we have that $\text{K/H[\lor, \exists]}$ satisfies SEN.

Finally, as to $\text{renH[\lor]}$ and $\text{renH[\lor, \exists]}$, let $\alpha$ be a CNF formula over $n$ variables $x_1, \ldots, x_n$. Let $\alpha'$ be the $\text{HORN}$ formula obtained by replacing every positive literal $x_i$ in $\alpha$ by the negative literal $\neg x_i'$ (where each $x_i'$ is a fresh variable), conjoined with $n$ additional clauses $\neg x_i \lor \neg x_i'$ ($i = 1, \ldots, n$). Let $\beta'$ be the $\text{KROM}$ formula $\bigwedge_{i=1}^n (x_i \lor x_i')$. By construction, $\alpha$ is inconsistent iff $\alpha' \land \beta'$ is inconsistent. Now, we associate $\alpha$ in polynomial time with the $\text{renH[\lor]}$ formula $\gamma = (\alpha' \land \neg y) \lor (\beta' \land y)$ where $y$ is a fresh variable. $\gamma$ also is a $\text{renH[\lor, \exists]}$ formula. We can easily check that $\forall y.\gamma$ is equivalent to $\alpha' \land \beta'$. If $\text{renH[\lor]}$ (resp. $\text{renH[\lor, \exists]}$) would satisfy SEN, then we could compute in time polynomial in the size of $\alpha$ a $\text{renH[\lor]}$ (resp. $\text{renH[\lor, \exists]}$) formula equivalent to $\forall y.\gamma$. Since each of $\text{renH[\lor]}$ and $\text{renH[\lor, \exists]}$ satisfies CO, we would have a polynomial time algorithm for deciding the satisfiability of $\alpha$, hence we would have $P = NP$.

$\land C$ It is obvious that each of $\text{KROM}$, $\text{HORN}$, and $\text{AFF}$ satisfies $\land C$.

For $\text{K/H}$, $\text{renH}$, $\text{K/H[\exists]}$, $\text{renH[\exists]}$, the non-representability results (!) holds already in the bounded case ($\land \text{ABC}$).

For $\text{K/H[\lor]}$, $\text{renH[\lor]}$, $\text{K/H[\lor, \exists]}$, $\text{renH[\lor, \exists]}$, the results comes from the fact that none of these languages satisfies $\land \text{ABC}$, unless $P = NP$. 

60
Consider now the cases of \(\text{KROM}[\lor]\), \(\text{HORN}[\lor]\), \(\text{AFF}[\lor]\) and \(\text{HORN}[\lor, \exists]\). Observe that every clause is a formula from any of those languages since every literal is a \(\text{KROM}\) formula, a \(\text{HORN}\) formula, and an \(\text{AFF}\) formula. Determining whether a conjunction of clauses is consistent cannot be achieved in (deterministic) polynomial time unless \(P = \text{NP}\) (this is the famous \(\text{SAT}\) problem). Since each of \(\text{KROM}[\lor]\), \(\text{HORN}[\lor]\), \(\text{AFF}[\lor]\) and \(\text{HORN}[\lor, \exists]\) satisfies \(\text{CO}\), none of them can also satisfy \(\land \text{C}\) unless \(P = \text{NP}\).

Finally, let us consider the case of \(\text{HORN}[\exists]\): let \(\exists X_1.\alpha_1, \ldots, \exists X_n.\alpha_n\) be \(n\) \(\text{HORN}[\exists]\) formulae where each \(\alpha_i\) \((i \in 1, \ldots, n)\) is a \(\text{HORN}\) formula. For each \(i \in 1, \ldots, n\), let \(\alpha_i\) be the \(\text{HORN}\) formula obtained by replacing in \(\alpha_i\) every occurrence of \(x \in X_i\) by a fresh variable \(x_i\), and let \(X_i\) be the set of all the variables \(x_i\) generated in the construction of \(\alpha_i\). By construction, every variable from \(X_i\) does not occur in any \(\alpha_j\) when \(j \neq i\). Hence, \(\bigwedge_{i=1}^n \exists X_i.\alpha_i\) is equivalent to \(\exists \bigcup_{i=1}^n X_i.\bigwedge_{i=1}^n \alpha_i\). Clearly enough, \(\exists \bigcup_{i=1}^n X_i.\bigwedge_{i=1}^n \alpha_i\) is a \(\text{HORN}[\exists]\) formula, and it can be generated in polynomial time from \(\exists X_1.\alpha_1, \ldots, \exists X_n.\alpha_n\).

\(\land \text{BC}\) Each of \(\text{KROM}\), \(\text{HORN}\), \(\text{AFF}\) and \(\text{HORN}[\exists]\) satisfies \(\land \text{BC}\) since it satisfies \(\land \text{C}\).

As to \(\text{K/H}\) and \(\text{K/H}[\exists]\), consider the \(\text{K/H}\) formulae \(x \lor y \land \neg x \lor \neg y \lor \neg z\). They are also \(\text{K/H}[\exists]\) formulae. The conjunction of them neither is equivalent to a \(\text{KROM}\) formula, nor is equivalent to a \(\text{HORN}\) formula, hence it is not equivalent to a \(\text{K/H}\) formula. From Proposition 7, we know that \(\text{K/H} \sim_e \text{K/H}[\exists]\), hence this conjunction is not equivalent to a \(\text{K/H}[\exists]\) formula.

As to \(\text{renH}\) and \(\text{renH}[\exists]\), consider the two \(\text{renH}\) formulae \(\alpha = x \lor y \lor z\) and \(\beta = \neg x \lor \neg y \lor \neg z\). They are also \(\text{renH}[\exists]\) formulae. There is no \(\text{renH}\) formula logically equivalent to the conjunction \(\alpha \land \beta\). From Proposition 7, we know that \(\text{renH} \sim_e \text{renH}[\exists]\), hence this conjunction is not equivalent to a \(\text{renH}[\exists]\) formula.

Let us now consider the cases of \(\text{KROM}[\lor]\), \(\text{HORN}[\lor]\), and \(\text{AFF}[\lor]\). Let \(\alpha = \lor(\alpha_1, \ldots, \alpha_n)\) and \(\beta = \lor(\beta_1, \ldots, \beta_m)\) be two \(\text{KROM}[\lor]\) (resp. \(\text{HORN}[\lor]\), \(\text{AFF}[\lor]\)) formulae. Then the formula \(\bigvee_{i=1}^n \bigvee_{j=1}^m (\alpha_i \land \beta_j)\) can be computed in time polynomial in the size of \(\alpha\) plus the size of \(\beta\), and is a \(\text{KROM}[\lor]\) (resp. \(\text{HORN}[\lor]\), \(\text{AFF}[\lor]\)) formula logically equivalent to the conjunction \(\alpha \land \beta\).

Let us focus on the case of \(\text{HORN}[\lor, \exists]\). Let \(\alpha\) and \(\beta\) be two \(\text{HORN}[\lor, \exists]\) formulae. From Proposition 2, since \(\text{HORN}\) is stable by uniform renaming,
Consider the two BC\(C\) The non-representability results for \(\beta\) Let \(\exists\) one can compute in polynomial time a HORN\([\forall]\exists\) formula \(\exists X.\alpha'\) (resp. \(\exists Y.\beta'\)) equivalent to \(\alpha\) (resp. \(\beta\)) where \(\alpha'\) (resp. \(\beta'\)) is a HORN\([\forall]\) formula. Let \(\alpha''\) (resp. \(\beta''\)) be the HORN\([\forall]\) formula obtained by replacing in \(\alpha'\) (resp. \(\beta'\)) every occurrence of \(x \in X\) (resp. \(x \in Y\)) by a fresh variable \(x'\) and let \(X'\) (resp. \(Y'\)) be the set of all variables \(x'\) generated in the construction of \(\alpha''\) (resp. \(\beta''\)). By construction \(\alpha \land \beta\) is equivalent to \((\exists X.\alpha') \land (\exists Y.\beta')\), which is in turn equivalent to \(\exists X' \cup Y'.(\alpha'' \land \beta'')\). Now, HORN\([\forall]\) satisfies \(\land BC\). Hence, a HORN\([\forall]\) formula \(\gamma\) equivalent to \(\alpha'' \land \beta''\) can be generated in time polynomial in the size of \(\alpha''\) plus the size of \(\beta''\). Accordingly, \(\exists X' \cup Y'.\gamma\) is a HORN\([\forall],\exists\) formula equivalent to \(\alpha \land \beta\), and it can be computed in time polynomial in the size of \(\alpha\) plus the size of \(\beta\).

Finally, let us consider the cases of \(K/H[\forall]\), ren\(H[\forall],\exists\), ren\(H[\forall]\). Let \(\alpha\) be a CNF formula over \(n\) variables \(x_1,\ldots, x_n\). Let \(\alpha'\) be the HORN formula obtained by replacing every positive literal \(x_i\) in \(\alpha\) by the negative literal \(\neg x_i'\) (where each \(x_i'\) is a fresh variable), conjoined with \(n\) additional clauses \(\neg x_i \lor \neg x_i'\) (\(i \in 1,\ldots, n\)). Let \(\beta'\) be the KROM formula \(\bigwedge_{i=1}^{n}(x_i \lor x_i')\). Observe that \(\beta'\) is consistent, hence each of \(\alpha'\) and \(\beta'\) is a K/H formula and ren\(H\) formula. As a consequence, each of them also belongs to \(K/H[\forall]\), ren\(H[\forall]\), K/H[\forall], \exists, and ren\(H[\forall],\exists\). Furthermore, both \(\alpha'\) and \(\beta'\) can be computed in time polynomial in the size of \(\alpha\). By construction, \(\alpha\) is consistent iff \(\alpha' \land \beta'\) is consistent. If any of \(K/H[\forall]\), ren\(H[\forall]\), K/H[\forall], \exists, or ren\(H[\forall],\exists\) would satisfy \(\land BC\), since each of these languages satisfy \(CO\), we would have \(P = NP\).

\(\forall C\) The non-representability results for KROM, HORN, K/H, ren\(H, AFF, HORN[\exists], K/H[\exists], ren\(H[\exists]\) come directly from the corresponding non-representability results for \(\forall BC\). The fact that each of KROM\([\forall]\), HORN\([\forall]\), K/H\([\forall]\), ren\(H[\forall]\), AFF\([\forall]\), HORN\([\forall],\exists\), K/H\([\forall],\exists\), ren\(H[\forall],\exists\) satisfies \(\forall C\) is immediate from their definitions.

\(\forall BC\) Consider the two KROM formulae \(\alpha = (x \lor y) \land (\neg y \lor \neg z)\) and \(\beta = \neg x \land z\). Each of \(\alpha\) and \(\beta\) belongs as well to K/H, ren\(H, K/H[\exists], ren\(H[\exists]\). Now, \(\alpha \lor \beta\) is logically equivalent to the formula \((x \lor y \lor z) \land (\neg x \lor \neg y \lor \neg z)\) for which no equivalent KROM formula (resp. K/H formula, ren\(H\) formula) exists. From Proposition 7, we know that KROM \(\sim_e KROM[\exists]\) (resp. K/H \(\sim_e K/H[\exists], ren\(H \sim_e ren\(H[\exists]\), hence there are no KROM[\exists] formula (resp. K/H[\exists] formula, ren\(H[\exists]\) formula) equivalent to \(\alpha \lor \beta\).

As to the HORN case and the AFF case, it is enough to consider \(\alpha = x\) and
\[ \beta = y: \text{no HORN formula and no AFF formula is equivalent to } \alpha \lor \beta. \] From Proposition 7, we know that \( \text{HORN} \sim_e \text{HORN}[\exists] \), hence there is no \( \text{HORN}[\exists] \) formula equivalent to \( \alpha \lor \beta \).

Finally, the fact that each of \( \text{KROM}[\lor], \text{HORN}[\lor], \text{K/H}[\lor], \text{renH}[\lor], \text{AFF}[\lor], \text{HORN}[\lor, \exists], \text{K/H}[\lor, \exists], \text{renH}[\lor, \exists] \) satisfies \( \lor \text{BC} \) comes from the fact that each of them satisfies \( \lor \text{C} \).

\[ \neg C \] Consider the \( \text{KROM} \) formula \( \alpha = (\neg x \lor y) \land (\neg z \lor y) \land (x \lor \neg y \lor z) \land (x \lor \neg z) \land (y \lor \neg z) \). \( \alpha \) also is a \( \text{HORN} \) formula, an \( \text{K/H} \) formula, a \( \text{renH} \) formula, an \( \text{HORN}[\exists] \) formula, an \( \text{K/H}[\exists] \) formula, and a \( \text{renH}[\exists] \) formula. But \( \neg \alpha \) is equivalent to the formula \( (x \lor y \lor z) \land (\neg x \lor \neg y \lor \neg z) \) for which no equivalent \( \text{KROM} \) formula (resp. \( \text{HORN} \) formula, \( \text{K/H} \) formula, \( \text{renH} \) formula) exists. From Proposition 7, we know that \( \text{HORN} \sim_e \text{HORN}[\exists], \text{K/H} \sim_e \text{K/H}[\exists], \text{and renH} \sim_e \text{renH}[\exists] \). Accordingly, the \( \neg C \) transformation is not always feasible in any of \( \text{KROM}, \text{HORN}, \text{K/H}, \text{renH}, \text{HORN}[\exists], \text{K/H}[\exists], \text{renH}[\exists] \).

Similarly, consider the \( \text{AFF} \) formula \( \alpha = \neg x \land \neg y \). No \( \text{AFF} \) formula is equivalent to \( \neg \alpha \), hence the \( \neg C \) transformation is not always feasible in \( \text{AFF} \).

As to \( \text{KROM}[\lor], \text{HORN}[\lor], \text{K/H}[\lor] \), let us consider the DNF formula \( \alpha_n = \bigvee_{i=1}^n (\neg x_i \land \neg y_i \land \neg z_i) \); \( \alpha_n \) is a \( \text{KROM}[\lor] \) formula, a \( \text{HORN}[\lor] \) formula and a \( \text{K/H}[\lor] \) formula. Now, in the proof of Proposition 11 (see Table 9), we show that the \( \text{renH} \) formula \( \bigwedge_{i=1}^n (x_i \lor y_i \lor z_i) \) equivalent to \( \neg \alpha_n \) has no polynomial-size representation in \( \text{K/H}[\lor] \), hence the conclusion follows.

Finally, let us consider the cases of \( \text{AFF}[\lor], \text{renH}[\lor], \text{renH}[\lor, \exists], \text{K/H}[\lor, \exists] \) and \( \text{HORN}[\lor, \exists] \). DNF is a subset of each of these languages. Now, a DNF formula \( \alpha \) is valid iff \( \neg \alpha \) is inconsistent. The fact that each of \( \text{AFF}[\lor], \text{renH}[\lor], \text{renH}[\lor, \exists], \text{K/H}[\lor, \exists] \) and \( \text{HORN}[\lor, \exists] \) satisfies \( \lor \text{CO} \) completes the proof.

\( \square \)

**Proposition 7**

- \( \text{HORN}[\exists] \sim_e \text{HORN} \).
- \( \text{K/H}[\exists] \sim_e \text{K/H} \).
• renH[∃] ∼ₚ renH.

Proof:

• HORN, K/H: Every prime implicate of a HORN formula (resp. a KROM formula) α is a Horn clause (resp. a binary clause). Since the prime implicates of ∃X.α for a finite subset X of PS and a C-DAG representation α are the prime implicates δ of α such that Var(δ) ∩ X = ∅, we get that ∃X.α is equivalent to a HORN formula (resp. a KROM formula) when α is a HORN formula (resp. a KROM formula).

• renH: Let α be a renH formula. A PL formula equivalent to ∃X.α is given by the conjunction β of all prime implicates of α not containing any variable from X. If V is a Horn renaming for α, then V(α) is a HORN formula. Since V(β) is equivalent to ∃X.V(α) and since HORN[∃] ∼ₚ HORN, V(β) is equivalent to a HORN formula. This shows that β is a renH formula (V is a Horn renaming for it) and this concludes the proof.

Proposition 8 KROM[V], HORN[V], K/H[V], renH[V], AFF[V], HORN[V, ∃], K/H[V, ∃], renH[V, ∃] are complete propositional languages.

Proof: This comes easily from the fact that TERM is included in each of KROM, HORN and AFF; as a consequence, its disjunction closure TERM[V] is included into each of eight closures above; the fact that DNF = TERM[V] is complete ends up the proof.

Proposition 9

• HORN[∃] <ₚ HORN.
• K/H[∃] <ₚ K/H.
• renH[∃] <ₚ renH.
• renH and K/H[∃] are incomparable w.r.t. ≤ₚ.
• K/H and HORN[∃] are incomparable w.r.t. ≤ₚ.
Proof: Let us consider first the three first items. For $\mathcal{L} \in \{\textsc{horn}, \textsc{k/h}, \textsc{renH}\}$, we have to prove that $\mathcal{L}[\exists] \preceq_s \mathcal{L}$, i.e., $\mathcal{L}[\exists] \preceq \mathcal{L}$ and $\mathcal{L} \not\preceq_s \mathcal{L}[\exists]$. That $\mathcal{L}[\exists] \preceq_s \mathcal{L}$ comes immediately from the inclusion $\mathcal{L}[\exists] \supseteq \mathcal{L}$ (cf. item 0. of Proposition 1). The other way around, consider the $\textsc{horn}[\exists]$ formula $\alpha_n = \exists\{y_1, \ldots, y_n\}.((\lor_{i=1}^n \neg y_i) \land \land_{i=1}^n ((\neg x_i \lor y_i) \land (\neg z_i \lor y_i)))$. Since $\textsc{horn} \subseteq \textsc{k/h}$ and $\textsc{horn} \subseteq \textsc{renH}$, this is also a $\textsc{k/h}\exists$ formula and a $\textsc{renH}\exists$ formula (cf. item 0. of Proposition 1). Since $\alpha_n$ has $2^n$ essential prime implicates, it does not have a CNF representation of size polynomial in $n$. Since $\textsc{horn}$, $\textsc{k/h}$, and $\textsc{renH}$ are subsets of $\textsc{cnf}$, the language $\textsc{cnf}$ is at least as succinct as any of them, so $\alpha_n$ does not have a representation of size of polynomial in $n$ as a $\textsc{horn}$ formula, a $\textsc{k/h}$ formula or a $\textsc{renH}$ formula.

For the last two items, we have to prove that $\textsc{renH} \not\preceq_s \textsc{k/h}\exists$, $\textsc{k/h}\exists \not\preceq_s \textsc{renH}$, $\textsc{k/h} \not\preceq\textsc{horn}\exists$, $\textsc{horn}\exists \not\preceq_s \textsc{k/h}$. From Proposition 7, we know that $\textsc{k/h}\exists \sim_e \textsc{horn}$, and that $\textsc{horn}\exists \sim_e \textsc{k/h}$. Furthermore, we know that $\textsc{renH} <_e \textsc{k/h} <_e \textsc{horn}$ (cf. Section 5). Altogether, this shows that $\textsc{renH} <_e \textsc{k/h}\exists$ and both $<_e \textsc{horn}\exists$. Especially, we have $\textsc{k/h}\exists <_e \textsc{renH}$ and $\textsc{horn}\exists <_e \textsc{k/h}$. Due to the fact that the relation $\preceq_s$ is included into the relation $\sim_e$, we have that for any subsets $\mathcal{L}_1$ and $\mathcal{L}_2$ of $\textsc{c-dag}$, if $\mathcal{L}_1 \not\preceq \mathcal{L}_2$, then $\mathcal{L}_1 \not\preceq_s \mathcal{L}_2$. This shows that $\textsc{k/h}\exists \not\preceq_s \textsc{renH}$ and $\textsc{horn}\exists \not\preceq_s \textsc{k/h}$. Finally, in order to prove that $\textsc{renH} \not\preceq_s \textsc{k/h}\exists$ and $\textsc{k/h} \not\preceq_s \textsc{horn}\exists$, it is enough to consider again the $\textsc{horn}\exists$ formula $\alpha_n = \exists\{y_1, \ldots, y_n\}.((\lor_{i=1}^n \neg y_i) \land \land_{i=1}^n ((\neg x_i \lor y_i) \land (\neg z_i \lor y_i)))$. We have shown above that this formula also is a $\textsc{k/h}\exists$ formula but that it does not have a representation of size of polynomial in $n$ as a $\textsc{renH}$ formula or as a $\textsc{k/h}$ formula. This concludes the proof. □

Proposition 10

- $\textsc{horn}[v, \exists] <_s \textsc{horn}[v]$.
- $\textsc{k/h}[v, \exists] <_s \textsc{k/h}[v]$.
- $\textsc{renH}[v, \exists] <_s \textsc{renH}[v]$.

Proof: We focus on $\textsc{ac3}$, the class of propositional representations containing all disjunctions of $\textsc{cnf}$ formulae and all conjunctions of $\textsc{dnf}$ formulae. Since every formula from $\textsc{horn}[v]$, $\textsc{k/h}[v]$ or $\textsc{renH}[v]$ is a disjunction of $\textsc{cnf}$ formulae, each of the languages $\textsc{horn}[v]$, $\textsc{k/h}[v]$, and $\textsc{renH}[v]$ is a subset of $\textsc{ac3}$, hence we have $\textsc{ac3} \subseteq \textsc{horn}[v]$, $\textsc{ac3} \leq_s \textsc{k/h}[v]$, and $\textsc{ac3} \leq_s \textsc{renH}[v]$. In order to prove the proposition, it is thus enough to show that $\textsc{ac3} \not\preceq_s \textsc{horn}[v, \exists]$, $\textsc{ac3} \not\preceq_s \textsc{k/h}[v, \exists]$, and $\textsc{ac3} \not\preceq_s \textsc{renH}[v, \exists]$. Since $\textsc{horn} \subseteq \textsc{k/h}$ and $\textsc{horn} \subseteq \textsc{renH}$,
we have the inclusions $\text{HORN}[\forall, \exists] \subseteq K/H[\forall, \exists]$ and $\text{HORN}[\forall, \exists] \subseteq \text{renH}[\forall, \exists]$ (cf. item 0. of Proposition 1), which imply that $K/H[\forall, \exists] \leq_s \text{HORN}[\forall, \exists]$, and $\text{renH}[\forall, \exists] \leq_s \text{HORN}[\forall, \exists]$. Therefore, in order to show that $\text{AC3} \not\leq_s \text{HORN}[\forall, \exists]$, $\text{AC3} \not\leq_s K/H[\forall, \exists]$, and $\text{AC3} \not\leq_s \text{renH}[\forall, \exists]$, it is enough to show that $\text{AC3} \not\leq_s \text{HORN}[\forall, \exists]$. We do it by exhibiting a $\text{HORN}[\forall, \exists]$ formula which has no polynomial-sized $\text{AC3}$ representation.

The proof is based on a theorem due to Sipser [74]. This theorem can be expressed as follows: consider any Boolean function $\alpha_i^n$ over $n^{2k-2}$ variables, represented by a NNF formula of depth $k > 1$ and such that all the leaves are labeled by variables occurring once in the formula, the $i$th level ($i \in 1, \ldots, k - 1$) from the bottom consists of nodes labeled by $\land$ (resp. $\lor$) when $i$ is even (resp. odd), the outdegree of the root node and the deepest internal nodes (those at depth $k - 1$) is equal to $n^2$. Sipser showed that such an $\alpha_i^n$ cannot be represented by a polynomial-sized circuit over $\{\neg, \lor, \land\}$ of depth at most $k - 1$.

Consider the Boolean function $\alpha_i^n$ over $n^6$ variables. By construction, it can be represented by a disjunction of $n$ conjunctions $\beta_1, \ldots, \beta_n$ of DNF formulae, where each $\beta_i$ ($i \in 1, \ldots, n$) is the conjunction of $n^2$ DNF $\gamma_{i,j}$ ($j \in 1, \ldots, n^2$), each $\text{DNF} \gamma_{i,j}$ ($j \in 1, \ldots, n^2$) consists of the disjunction of $n^2$ terms $\delta_{i,j,k}$ ($k \in 1, \ldots, n^2$), and finally each term $\delta_{i,j,k}$ ($k \in 1, \ldots, n^2$) consists of the conjunction of $n$ negated variables $\neg x_{i,j,k,l}$ ($l \in 1, \ldots, n$) occurring only once in $\alpha_i^n$. For each $i \in 1, \ldots, n$ and $j \in 1, \ldots, n^2$, consider now the $\text{HORN}$ formula $h_{i,j}$ such that $h_{i,j} =$

\[
\left( \bigvee_{k=1}^{n^2} \neg y_{i,j,k} \right) \land \bigwedge_{k=1}^{n^2} \bigwedge_{l=1}^{n} (y_{i,j,k} \lor \neg x_{i,j,k,l}).
\]

$h_{i,j}$ contains $n^3 + 1$ clauses of size at most $n^2$, hence the $\text{HORN}$ formula $\bigwedge_{j=1}^{n^2} h_{i,j}$ contains $n^5 + n^2$ clauses of size at most $n^2$. Let $Y = \bigcup_{j=1}^{n^2} \left( \bigcup_{k=1}^{n^2} \{y_{i,j,k}\} \right)$. By construction, the $\text{HORN}[\forall, \exists]$ formula $\exists Y. \left( \bigwedge_{i=1}^{n^2} \left( \bigwedge_{j=1}^{n^2} h_{i,j} \right) \right)$ can be generated in time polynomial in $n$. From Sipser theorem, $\alpha_i^n$ has no polynomial-sized $\text{AC3}$ representation. It remains to show that $\alpha_i^n$ is equivalent to $\exists Y. \left( \bigwedge_{i=1}^{n^2} \left( \bigwedge_{j=1}^{n^2} h_{i,j} \right) \right)$. First of all, since existential quantifications "distribute" over disjunctions and since each $y_{i,j,k}$ ($i \in 1, \ldots, n$, $j \in 1, \ldots, n^2$, $k \in 1, \ldots, n^2$) does not occur in $h_{i,j}$ ($i' \in 1, \ldots, n$, $j \in 1, \ldots, n^2$) unless $i' = i$ and $j' = j$, we have that $\exists Y. \left( \bigwedge_{i=1}^{n^2} \left( \bigwedge_{j=1}^{n^2} h_{i,j} \right) \right)$ is equivalent to $\bigvee_{i=1}^{n^2} \left( \bigwedge_{j=1}^{n^2} \exists \bigcup_{k=1}^{n^2} \{y_{i,j,k}\} h_{i,j} \right)$. Finally, by construction, for each $i \in 1, \ldots, n$ and $j \in 1, \ldots, n^2$, $\gamma_{i,j} = \bigvee_{k=1}^{n^2} \delta_{i,j,k}$ is the $\text{IP}$
representation of \( \exists \bigcup_{k=1}^{n^2} \{y_{i,j,k}\}.h_{i,j} \), hence it is equivalent to it. The replacement metatheorem for propositional logic concludes the proof. \( \square \)

**Proposition 11** The results in Table 3 hold.

<table>
<thead>
<tr>
<th>AFF[V]</th>
<th>renH[V,\exists]</th>
<th>K/H[V,\exists]</th>
<th>HORN[V,\exists]</th>
<th>KROM[V]</th>
</tr>
</thead>
<tbody>
<tr>
<td>AFF[V]</td>
<td>~s</td>
<td>~s</td>
<td>~s</td>
<td>~s</td>
</tr>
<tr>
<td>renH[V,\exists]</td>
<td>~s</td>
<td>~s</td>
<td>~s</td>
<td>~s</td>
</tr>
<tr>
<td>K/H[V,\exists]</td>
<td>~s</td>
<td>~s</td>
<td>~s</td>
<td>~s</td>
</tr>
<tr>
<td>HORN[V,\exists]</td>
<td>~s</td>
<td>~s</td>
<td>~s</td>
<td>~s</td>
</tr>
<tr>
<td>KROM[V]</td>
<td>~s</td>
<td>~s</td>
<td>~s</td>
<td>~s</td>
</tr>
</tbody>
</table>

The relative succinctness of the full disjunctive closures of KROM, HORN, AFF, K/H, and renH.

**Proof:** The proof is broken into six steps, where we prove some succinctness relationships between languages, and then apply transitivity of \( \leq_s \) to possibly infer new relationships. Associated with each step of the proof is a table in which we mark all relationships proved at the step.

**Table 5:** From the obvious equalities and inclusions \( \text{HORN}[V, \exists] \subseteq K/H[V, \exists], \text{HORN}[V, \exists] \subseteq \text{renH}[V, \exists], \text{KROM}[V] \subseteq K/H[V, \exists] \), we get the results given in Table 5.

**Table 6:** Since \( K/H[V, \exists] \sim_p K/H[\exists][V] \) (cf. Proposition 1), every \( K/H[V, \exists] \) formula can be associated in polynomial time with an equivalent disjunction \( \bigvee_{i=1}^{n} \exists X_i . \beta_i \) of \( K/H[\exists] \) formulae. Since KROM satisfies CO, we can easily determine in polynomial time which \( \beta_i \) (\( i \in 1, \ldots, n \)) which are consistent. All the \( \beta_i \) (\( i \in 1, \ldots, n \)) which are inconsistent

<table>
<thead>
<tr>
<th>AFF[V]</th>
<th>renH[V,\exists]</th>
<th>K/H[V,\exists]</th>
<th>HORN[V,\exists]</th>
<th>KROM[V]</th>
</tr>
</thead>
<tbody>
<tr>
<td>AFF[V]</td>
<td>\sim_s \underline{1}</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>renH[V,\exists]</td>
<td>\sim_s \underline{1}</td>
<td>\sim_s \underline{1}</td>
<td>\sim_s \underline{1}</td>
<td>\sim_s \underline{1}</td>
</tr>
<tr>
<td>K/H[V,\exists]</td>
<td>\sim_s \underline{1}</td>
<td>\sim_s \underline{1}</td>
<td>\sim_s \underline{1}</td>
<td>\sim_s \underline{1}</td>
</tr>
<tr>
<td>HORN[V,\exists]</td>
<td>\sim_s \underline{1}</td>
<td>\sim_s \underline{1}</td>
<td>\sim_s \underline{1}</td>
<td>\sim_s \underline{1}</td>
</tr>
<tr>
<td>KROM[V]</td>
<td>\sim_s \underline{1}</td>
<td>\sim_s \underline{1}</td>
<td>\sim_s \underline{1}</td>
<td>\sim_s \underline{1}</td>
</tr>
</tbody>
</table>

Table 5: The relative succinctness of the full disjunctive closures of KROM, HORN, AFF, K/H, and renH.
can be removed from the disjunction without questioning equivalence (if they are all inconsistent, the input formula is associated with ⊥, which is a renH[\forall, 3] formula). In the remaining case, since every consistent KROM formula is a renH formula, the resulting disjunction is a renH[\forall, 3] formula equivalent to the input formula. Hence we get the results given in Table 6.

Table 7: Let us now show that HORN[\forall, 3] \not\leq_s KROM[\forall, 3], HORN[\forall, 3] \not\leq_s K/H[\forall, 3], and HORN[\forall, 3] \not\leq_s renH[\forall, 3]. To do so, it is enough to prove that HORN[\forall, 3] \not\leq_s KROM. Consider the KROM formula \alpha_n = \bigwedge_{i=1}^n (x_i \lor y_i) for any n. Towards a contradiction, suppose that there exists in HORN[\forall, 3] a formula equivalent to \alpha_n and whose size is polynomial in n; since HORN[\forall, 3] \sim_p HORN[\exists][\forall] (cf. Proposition 1), there exists as well a HORN[\exists][\forall] formula \beta = \bigvee_{i=1}^m \exists x_i \beta_i equivalent to \alpha_n and whose size is polynomial in n. Especially, m must remain polynomial in n. We also know that HORN[\exists] \sim_e HORN (cf. Proposition 7). Hence, if \beta exists, then there also exists a HORN[\forall] formula \gamma = \bigvee_{i=1}^m \gamma_i equivalent to \alpha_n and with m polynomial in n. Note that the size of \gamma_i (i \in 1, \ldots, m) can be exponential in the size of \beta_i (this does not matter for the remaining part of the proof).

By construction, \alpha_n has 2^n minimal models \omega over Var(\alpha_n), where for each i \in 1, \ldots, n, exactly one of the two variables x_i and y_i are set to 1 by \omega. Consider now any pair \omega, \omega' of distinct minimal models of \alpha_n; by construction, and(\omega, \omega') maps each variable to 0, hence it is not a model of \alpha_n. Thus, as a consequence of the characterization of HORN by closure of models, \omega and \omega' cannot be models of the same formula \gamma_i. Therefore, every HORN[\forall] formula \gamma = \bigvee_{i=1}^m \gamma_i equivalent to \alpha_n must be such that m \geq 2^n. This shows that there is no HORN[\forall, 3] formula equivalent to \alpha_n and whose size is polynomial in n. Thus, we get the results given in Table 7.
We also have to show that

\[
\begin{array}{|c|c|c|c|c|}
\hline
& \text{AFF[\(\top\)]} & \text{renH[\(\top, \exists\)]} & \text{K/H[\(\top, \exists\)]} & \text{HORN[\(\top, \exists\)]} & \text{KROM[\(\top\)]} \\
\hline
\text{AFF[\(\top\)]} & \sim_s & & & \\
\hline
\text{renH[\(\top, \exists\)]} & \sim_s & \leq_s & \leq_s & \leq_s \\
\hline
\text{K/H[\(\top, \exists\)]} & & \sim_s & \leq_s & \leq_s \\
\hline
\text{HORN[\(\top, \exists\)]} & \leq_s & \leq_s & \sim_s & \leq_s \\
\hline
\text{KROM[\(\top\)]} & & & & \sim_s \\
\hline
\end{array}
\]

Table 7: The relative succinctness of the full disjunctive closures of \text{KROM}, \text{HORN}, \text{AFF}, \text{K/H}, and \text{renH}.

\textbf{Table 8:} Let us now show that \text{KROM[\(\top\)]} \not\leq_s \text{HORN[\(\top, \exists\)]}, \text{KROM[\(\top\)]} \not\leq_s \text{K/H[\(\top, \exists\)]}, and \text{KROM[\(\top\)]} \not\leq_s \text{renH[\(\top, \exists\)]}. To do so, it is enough to prove that \text{KROM[\(\top\)]} \not\leq_s \text{HORN}. Consider the \text{HORN} formula \(\alpha_n = \bigwedge_{i=1}^n (\neg x_i \lor \neg y_i \lor \neg z_i)\) for any \(n\). Towards a contradiction, suppose that there exists in \text{KROM[\(\top\)]} a formula \(\gamma = \bigvee_{i=1}^m \gamma_i\) equivalent to \(\alpha_n\) and whose size is polynomial in \(n\); then \(m\) must remain polynomial in \(n\).

By construction, \(\alpha_n\) has \(7^n\) models over \(\text{Var}(\alpha_n)\). How many models of \(\alpha_n\) can be models of the same \(\gamma_i\) \((i \in 1, \ldots, m)\)? Let us consider any \(\omega_1, \omega_2, \omega_3 \in \text{Mod}(\gamma_i)\), it cannot be the case that for any \(i \in 1, \ldots, n\), we have \(\omega_1(x_i) = 0, \omega_1(y_i) = 1, \omega_1(z_i) = 1, \omega_2(x_i) = 1, \omega_2(y_i) = 0, \omega_2(z_i) = 1, \omega_3(x_i) = 1, \omega_3(y_i) = 1, \omega_3(z_i) = 0\). Indeed, if this were the case, we would have \(\text{maj}(\omega_1, \omega_2, \omega_3)(x_i) = \text{maj}(\omega_1, \omega_2, \omega_3)(y_i) = \text{maj}(\omega_1, \omega_2, \omega_3)(z_i) = 1\). If \(\gamma_i\) is a \text{KROM} formula, then \(\text{maj}(\omega_1, \omega_2, \omega_3)\) should also be a model of \(\gamma_i\). But \(\text{maj}(\omega_1, \omega_2, \omega_3)\) is not a model of \(\alpha_n\). Thus, each \(\gamma_i\) cannot have more than \(6^n\) models of \(\alpha_n\) over \(\text{Var}(\alpha_n)\). Subsequently, the pigeon/hole principle shows that at least \(\lceil \frac{7}{6} \rceil^n\) \text{KROM formulae} \(\gamma_i\) are required to cover the models of \(\alpha_n\). The fact that \(\lceil \frac{7}{6} \rceil^n\) is exponential in the size of \(\alpha_n\) concludes the proof. By transitivity of \(\leq_s\), we get the results given in Table 8.

\textbf{Table 9:} We also have to show that \(\text{K/H[\(\top, \exists\)]} \not\leq_s \text{renH[\(\top, \exists\)]}\). To do so, it is enough to prove that \(\text{K/H[\(\top, \exists\)]} \not\leq_s \text{renH}\). Consider the \text{renH} formula \(\alpha_n = \bigwedge_{i=1}^n (x_i \lor y_i \lor z_i)\) for any \(n\) (\(\text{Var}(\alpha_n)\) is a possible Horn renaming for it, since if one replaces in \(\alpha_n\) every literal from \(L_{\text{Var}(\alpha_n)}\) by its complementary literal, one gets a \text{HORN} formula).

Towards a contradiction, suppose that there exists in \(\text{K/H[\(\top, \exists\)]}\) a formula equivalent to \(\alpha_n\) and whose size is polynomial in \(n\); since \(\text{K/H[\(\top, \exists\)]} \sim_p \text{K/H[\(\exists\)][\(\top\)]}\) (cf. Proposition 1), there exists as well a \(\text{K/H[\(\exists\)][\(\top\)]}\) formula \(\beta = \bigvee_{i=1}^m \exists X_i \beta_i\)
Table 8: The relative succinctness of the full disjunctive closures of \(KROM, HORN, AFF, K/H, \) and \(renH\).

<table>
<thead>
<tr>
<th></th>
<th>(AFF[V])</th>
<th>(renH[V, \exists])</th>
<th>(K/H[V, \exists])</th>
<th>(HORN[V, \exists])</th>
<th>(KROM[V])</th>
</tr>
</thead>
<tbody>
<tr>
<td>(AFF[V])</td>
<td>(\sim_s)</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>(renH[V, \exists])</td>
<td>(\sim_s)</td>
<td>(\leq_s)</td>
<td>(\leq_s)</td>
<td>(\leq_s)</td>
<td></td>
</tr>
<tr>
<td>(K/H[V, \exists])</td>
<td>(\leq_s)</td>
<td>(\sim_s)</td>
<td>(\leq_s)</td>
<td>(\leq_s)</td>
<td></td>
</tr>
<tr>
<td>(HORN[V, \exists])</td>
<td>(\nleq_s)</td>
<td>(\nleq_s)</td>
<td>(\sim_s)</td>
<td>(\nleq_s)</td>
<td></td>
</tr>
<tr>
<td>(KROM[V])</td>
<td>(\nleq_s)</td>
<td>(\leq_s)</td>
<td>(\nleq_s)</td>
<td>(\leq_s)</td>
<td>(\sim_s)</td>
</tr>
</tbody>
</table>

Table 9: The relative succinctness of the full disjunctive closures of \(KROM, HORN, AFF, K/H, \) and \(renH\).

<table>
<thead>
<tr>
<th></th>
<th>(AFF[V])</th>
<th>(renH[V, \exists])</th>
<th>(K/H[V, \exists])</th>
<th>(HORN[V, \exists])</th>
<th>(KROM[V])</th>
</tr>
</thead>
<tbody>
<tr>
<td>(AFF[V])</td>
<td>(\sim_s)</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>(renH[V, \exists])</td>
<td>(\sim_s)</td>
<td>(\leq_s)</td>
<td>(\leq_s)</td>
<td>(\leq_s)</td>
<td></td>
</tr>
<tr>
<td>(K/H[V, \exists])</td>
<td>(\nleq_s)</td>
<td>(\sim_s)</td>
<td>(\leq_s)</td>
<td>(\leq_s)</td>
<td></td>
</tr>
<tr>
<td>(HORN[V, \exists])</td>
<td>(\nleq_s)</td>
<td>(\nleq_s)</td>
<td>(\sim_s)</td>
<td>(\nleq_s)</td>
<td></td>
</tr>
<tr>
<td>(KROM[V])</td>
<td>(\nleq_s)</td>
<td>(\nleq_s)</td>
<td>(\nleq_s)</td>
<td>(\sim_s)</td>
<td></td>
</tr>
</tbody>
</table>

equivalent to \(\alpha_n\) and whose size is polynomial in \(n\). Especially, \(m\) must remain polynomial in \(n\). We also know that \(K/H[\exists] \sim_e K/H\) (cf. Proposition 7). Hence, if \(\beta\) exists, then there also exists a \(K/H[V]\) formula \(\gamma = \bigvee_{i=1}^{m} \gamma_i\) equivalent to \(\alpha_n\) and with \(m\) polynomial in \(n\).

Let us now prove that if such a \(\gamma\) exists, then there also exists a \(KROM[V]\) formula \(\delta = \bigvee_{i=1}^{m} \delta_i\) equivalent to \(\alpha_n\) and with \(m\) polynomial in \(n\). Consider any \(K/H\) formula \(\gamma_i\) \((i \in 1, \ldots, m)\) and suppose that it is a \(HORN\) formula. Then \(\gamma_i\) is equivalent to an implicant of \(\alpha_n\). This is obvious if \(\gamma_i\) is inconsistent. In the remaining case, every clause \(x_i \lor y_i \lor z_i\) \((i \in 1, \ldots, n)\) of \(\alpha_n\) must be implied by a prime implicate of \(\gamma_i\), which must be a Horn clause; hence this prime implicate must be equivalent to \(x_i, y_i\) or \(z_i\). Accordingly, \(\gamma_i\) must be equivalent to a term, hence to a \(KROM\) formula.

It remains to show that \(\alpha_n\) has no polyspace representation in \(KROM[V]\). The proof is similar to the one used for showing that \(KROM[V, \exists] \nleq_s HORN\) (this is not surprising since \(\alpha_n\) is a reverse Horn \text{CNF} formula). We get the results given in Table 9.
Table 10: Finally, we show that $\text{AFF}[\exists]$ is incomparable w.r.t. any of $\text{renH}[\exists, \forall], K/H[\exists, \forall], \text{HORN}[\exists, \forall]$ and $\text{KROM}[\forall]$. We first show that $\text{renH}[\exists, \forall] \not\leq_s \text{AFF}$, which proves enough to conclude that $\text{renH}[\exists, \forall] \not\leq_s \text{AFF}[\forall], K/H[\forall, \exists] \not\leq_s \text{AFF}[\forall]$, and $\text{KROM}[\forall] \not\leq_s \text{AFF}[\forall]$.

Let $\alpha_n = \bigwedge_{i=1}^n (x_i \oplus y_i \oplus z_i \oplus \top)$. By construction $\alpha_n$ is an $\text{AFF}$ formula. Furthermore, the restriction of any model of $\alpha_n$ over any $\{x_i, y_i, z_i\}$ for $i \in \{1, \ldots, n\}$ is of the form 000, 011, 101 or 110. Thus, $\alpha_n$ has $4^n$ models over $\text{Var}(\alpha_n)$. Let $\beta$ be a $\text{renH}[\exists, \forall]$ formula equivalent to $\alpha_n$. Since $\text{renH}[\exists, \forall] \sim_p \text{renH}[\exists][\forall]$ (cf. Proposition 1), $\beta$ is polynomially translatable into a formula $\bigvee_{i=1}^m \beta_i$ from $\text{renH}[\exists][\forall]$. Therefore, if $\beta$ has a polynomial-sized representation of $\alpha_n$, then $\bigvee_{i=1}^m \beta_i$ also is a polynomial-sized representation of $\alpha_n$, which implies that $m$ must not be exponential in $n$.

Since $\text{renH}[\exists] \sim_e \text{renH}$ (cf. Proposition 7), each $\beta_i$ for $i \in \{1, \ldots, m\}$ can be translated into an equivalent $\text{renH}$ formula $\gamma_i$ (which size can be exponential in the size of $\beta_i$, but this does not matter). The point is that if $\beta$ has a $\text{renH}[\exists][\forall]$ representation as a disjunction of $m \text{renH}[\exists]$ formulae, then it also has a $\text{renH}[\forall]$ representation as a disjunction of $m \text{renH}$ formulae.

Since each $\gamma_i$ for $i \in \{1, \ldots, m\}$ is a $\text{renH}$ formula which entails $\alpha_n$, from [75], there exists a model $V_i$ of $\alpha_n$ such that $V_i$ is a Horn renaming for $\gamma_i$, and $V_i(\gamma_i) \models V_i(\alpha_n)$. As explained above, the restriction of $V_i$ over any $\{x_i, y_i, z_i\}$ for $i \in \{1, \ldots, n\}$ is of the form 000, 011, 101 or 110. Since applying $V_i$ leads to renaming an even number of variables in each set $\{x_i, y_i, z_i\}$ for $i \in \{1, \ldots, n\}$, we necessarily have $V_i(x_i \oplus y_i \oplus z_i \oplus \top) \equiv x_i \oplus y_i \oplus z_i \oplus \top$, and subsequently $V_i(\alpha_n) \equiv \alpha_n$.

Thus, we get that $\bigvee_{i=1}^m V_i(\gamma_i)$ is a $\text{HORN}[\forall]$ formula equivalent to $\alpha_n$. At this stage, we have shown that if $\alpha_n$ has a polynomial-sized representation as a $\text{renH}[\forall, \exists]$ formula, then it must also have a $\text{HORN}[\forall]$ representation with a number of disjoints that is polynomial in $n$.

We are now going to prove that this is not the case, i.e., the number of disjoints in any $\text{HORN}[\forall]$ formula $\bigvee_{i=1}^m \delta_i$ equivalent to $\alpha_n$ actually is exponential in $n$. Consider the subset $S$ of models $\omega$ of $\alpha_n$ over $\text{Var}(\alpha_n)$ such that for each $i \in \{1, \ldots, n\}$ the restriction of $\omega$ over $\{x_i, y_i, z_i\}$ is of the form 011, 101 or 110. Every pair of distinct models $\omega$ and $\omega'$ from $S$ is such that $\text{and}(\omega, \omega')$ is not a model of $\alpha_n$; indeed, there must exist $i \in \{1, \ldots, n\}$ such that the restrictions of $\omega$ and $\omega'$ over $\{x_i, y_i, z_i\}$ differ, and $\text{and}(\omega, \omega')$ is not a model of $x_i \oplus y_i \oplus z_i \oplus \top$ (its restriction over $\{x_i, y_i, z_i\}$ is of the form 001, 010 or 100). Thus, because of the closure property of $\text{HORN}$ formulae, every pair of distinct models in $S$ cannot be models of the same $\text{HORN}$ formula $\delta_i$. Since $S$ contains $3^n$ models, the number
Table 10: The relative succinctness of the full disjunctive closures of KROM, HORN, AFF, K/H, and renH.

<table>
<thead>
<tr>
<th></th>
<th>AFF[∨]</th>
<th>renH[∨, ∃]</th>
<th>K/H[∨, ∃]</th>
<th>HORN[∨, ∃]</th>
<th>KROM[∨]</th>
</tr>
</thead>
<tbody>
<tr>
<td>AFF[∨]</td>
<td>∼s</td>
<td>≤s</td>
<td>≤s</td>
<td>≤s</td>
<td>≤s</td>
</tr>
<tr>
<td>renH[∨, ∃]</td>
<td>≤s</td>
<td>∼s</td>
<td>≤s</td>
<td>≤s</td>
<td>≤s</td>
</tr>
<tr>
<td>K/H[∨, ∃]</td>
<td>≤s</td>
<td>≤s</td>
<td>∼s</td>
<td>≤s</td>
<td>≤s</td>
</tr>
<tr>
<td>HORN[∨, ∃]</td>
<td>≤s</td>
<td>≤s</td>
<td>≤s</td>
<td>∼s</td>
<td>≤s</td>
</tr>
<tr>
<td>KROM[∨]</td>
<td>≤s</td>
<td>≤s</td>
<td>≤s</td>
<td>≤s</td>
<td>∼s</td>
</tr>
</tbody>
</table>

$m$ of disjoints in $\bigvee_{i=1}^{m} \delta_i$ is lower bounded by $3^n$. This shows that $\alpha_n$ has no polynomial-sized representation as a $\text{renH}[\vee, \exists]$ formula.

Conversely, let us show that $\text{AFF}[\vee]$ is not at least as succinct as any of $\text{renH}[\vee, \exists]$, $\text{K/H}[\vee, \exists]$, $\text{HORN}[\vee, \exists]$ and $\text{KROM}[\vee]$. Consider the formula $\alpha_n = \bigwedge_{i=1}^{n} (\neg x_i \vee \neg y_i)$ for any $n$. It is a KROM formula and a HORN formula. Hence, it is also a K/H formula and a renH formula. Since $\text{Var}(\alpha_n)$ contains $2^n$ atoms, $4^n$ interpretations over $\text{Var}(\alpha_n)$ have to be considered. Among them, one can find $3^n$ models of $\alpha_n$, only, since for each $i \in 1, \ldots, n$, there are only 3 truth assignments of $x_i$ and $y_i$ (over the four possible assignments of those two variables) which satisfy $\neg x_i \vee \neg y_i$. Now, there is no AFF formula $\beta$ implying $\alpha_n$ and with strictly more than $2^n$ models (taken in the set of models of $\alpha_n$ since $\beta \models \alpha_n$ must hold). By reductio ad absurdum: if this were the case, then one could find $i \in 1, \ldots, n$ and $\omega_1, \omega_2, \omega_3 \in \text{Mod}(\alpha_n)$ such that $\omega_1(x_i) = 0$, $\omega_1(y_i) = 0$, $\omega_2(x_i) = 0$, $\omega_2(y_i) = 1$, $\omega_3(x_i) = 1$, $\omega_3(y_i) = 0$. If $\omega_1, \omega_2, \omega_3 \in \text{Mod}(\beta)$ and $\beta$ is an AFF formula, then the affine closure property requires $\oplus(\omega_1, \omega_2, \omega_3)$ to be a model of $\beta$, hence a model of $\alpha_n$. But $\oplus(\omega_1, \omega_2, \omega_3)$ falsifies $\neg x_i \vee \neg y_i$. Subsequently, from the pigeon/hole principle, every AFF[∨] formula equivalent to $\alpha_n$ must contain at least $\lceil (\frac{3}{2})^n \rceil$ AFF formulae as disjuncts. The fact that $\lceil (\frac{3}{2})^n \rceil$ is exponential in the size of $\alpha_n$ concludes the proof.

\(\square\)

**Proposition 12** The results in Table 4 hold.

**Proof:** Again, the proof is broken in a number of steps, where we prove some succinctness relationships between languages, and then apply transitivity of $\leq_s$ to possibly infer new relationships. Associated with each step of the proof is a table in which we mark all relationships proved at the step.
Comparing w.r.t. succinctness the full disjunctive closures of $\text{KROM}$, $\text{HORN}$, $\text{K/H}$, $\text{renH}$, and $\text{AFF}$, with $\text{OBDD}_<$, $\text{IP}$, $\text{DNF}$, $\text{d-DNNF}$, $\text{DNNF}_T$, $\text{PI}$, and $\text{CNF}$. * means that the result holds unless the polynomial hierarchy collapses.

<table>
<thead>
<tr>
<th></th>
<th>$\text{AFF}^\lor$</th>
<th>$\text{renH}_k$</th>
<th>$\text{K/H}^\lor$</th>
<th>$\text{HORN}^\lor$</th>
<th>$\text{KROM}^\lor$</th>
</tr>
</thead>
<tbody>
<tr>
<td>CNF</td>
<td>$\preceq_s$</td>
<td>$\preceq_s$</td>
<td>$\preceq_s$</td>
<td>$\preceq_s$</td>
<td>$\preceq_s$</td>
</tr>
<tr>
<td>$\text{PI}$</td>
<td>$\preceq_s$</td>
<td>$\preceq_s$</td>
<td>$\preceq_s$</td>
<td>$\preceq_s$</td>
<td>$\preceq_s$</td>
</tr>
<tr>
<td>$\text{DNNF}_T$</td>
<td>$\preceq_s$</td>
<td>$\preceq_s$</td>
<td>$\preceq_s$</td>
<td>$\preceq_s$</td>
<td>$\preceq_s$</td>
</tr>
<tr>
<td>$\text{d-DNNF}$</td>
<td>$\preceq_s$</td>
<td>$\preceq_s$</td>
<td>$\preceq_s$</td>
<td>$\preceq_s$</td>
<td>$\preceq_s$</td>
</tr>
<tr>
<td>DNF</td>
<td>$\preceq_s$</td>
<td>$\preceq_s$</td>
<td>$\preceq_s$</td>
<td>$\preceq_s$</td>
<td>$\preceq_s$</td>
</tr>
<tr>
<td>IP</td>
<td>$\preceq_s$</td>
<td>$\preceq_s$</td>
<td>$\preceq_s$</td>
<td>$\preceq_s$</td>
<td>$\preceq_s$</td>
</tr>
<tr>
<td>$\text{OBDD}_&lt;$</td>
<td>$\preceq_s$</td>
<td>$\preceq_s$</td>
<td>$\preceq_s$</td>
<td>$\preceq_s$</td>
<td>$\preceq_s$</td>
</tr>
</tbody>
</table>

Table 11: Let $\mathcal{L}$ be any language among $\text{AFF}^\lor$, $\text{renH}_k$, $\text{K/H}^\lor$, $\text{HORN}^\lor$, $\text{KROM}^\lor$, and the corresponding existential closures $\text{renH}_k$, $\text{K/H}^\lor$, $\text{HORN}^\lor$. Since $\text{TERM} \subseteq \text{AFF}$, $\text{TERM} \subseteq \text{HORN}$, $\text{TERM} \subseteq \text{KROM}$, $\text{HORN} \supseteq \text{renH}$ and $\text{KROM} \supseteq \text{renH}$, we obviously have $\text{DNF} \supseteq \text{L}$, hence we have $\text{DNF} \supseteq \text{L}$. In [1], it is proven that $\text{PI} \equiv_s \text{CNF}$, $\text{DNF} \not\equiv_s \text{CNF}$, $\text{DNF} \not\equiv_s \text{OBDD}_<$, and $\text{IP} \equiv_s \text{DNF}$. By transitivity of $\leq_s$, we get the results given in Table 11.

<table>
<thead>
<tr>
<th></th>
<th>$\text{AFF}^\lor$</th>
<th>$\text{renH}_k$</th>
<th>$\text{K/H}^\lor$</th>
<th>$\text{HORN}^\lor$</th>
<th>$\text{KROM}^\lor$</th>
</tr>
</thead>
<tbody>
<tr>
<td>CNF</td>
<td>$\leq_s$</td>
<td>$\leq_s$</td>
<td>$\leq_s$</td>
<td>$\leq_s$</td>
<td>$\leq_s$</td>
</tr>
<tr>
<td>$\text{PI}$</td>
<td>$\leq_s$</td>
<td>$\leq_s$</td>
<td>$\leq_s$</td>
<td>$\leq_s$</td>
<td>$\leq_s$</td>
</tr>
<tr>
<td>$\text{DNNF}_T$</td>
<td>$\leq_s$</td>
<td>$\leq_s$</td>
<td>$\leq_s$</td>
<td>$\leq_s$</td>
<td>$\leq_s$</td>
</tr>
<tr>
<td>$\text{d-DNNF}$</td>
<td>$\leq_s$</td>
<td>$\leq_s$</td>
<td>$\leq_s$</td>
<td>$\leq_s$</td>
<td>$\leq_s$</td>
</tr>
<tr>
<td>DNF</td>
<td>$\geq_s$</td>
<td>$\geq_s$</td>
<td>$\geq_s$</td>
<td>$\geq_s$</td>
<td>$\geq_s$</td>
</tr>
<tr>
<td>IP</td>
<td>$\geq_s$</td>
<td>$\geq_s$</td>
<td>$\geq_s$</td>
<td>$\geq_s$</td>
<td>$\geq_s$</td>
</tr>
<tr>
<td>$\text{OBDD}_&lt;$</td>
<td>$\geq_s$</td>
<td>$\geq_s$</td>
<td>$\geq_s$</td>
<td>$\geq_s$</td>
<td>$\geq_s$</td>
</tr>
</tbody>
</table>

Table 11: Comparing w.r.t. succinctness the full disjunctive closures of $\text{KROM}$, $\text{HORN}$, $\text{K/H}$, $\text{renH}$, and $\text{AFF}$, with $\text{OBDD}_<$, $\text{IP}$, $\text{DNF}$, $\text{d-DNNF}$, $\text{DNNF}_T$, $\text{PI}$, and $\text{CNF}$. * means that the result holds unless the polynomial hierarchy collapses.

Table 12: Consider the following consistent $\text{KROM}$ formula $\alpha_n = \bigwedge_{i=1}^n (\neg x_i \lor \neg y_i)$; it is also a $\text{HORN}$ formula, hence it belongs to the disjunction closure and to the full disjunctive closure of each language among $\text{KROM}$, $\text{HORN}$, $\text{K/H}$, and
renH. \( \alpha_n \) has \( 2^n \) essential prime implicants\(^\text{10}\), hence there is no polynomial-sized IP formula and no polynomial-sized DNF formula equivalent to it. Similarly, the AFF formula \( \beta_n = \bigoplus_{i=1}^{n} x_i \) (which is also an AFF[∨] formula) has \( 2^{n-1} \) essential prime implicants, hence there is no polynomial-sized IP formula and no polynomial-sized DNF formula equivalent to it. We get the results given in Table 12.

<table>
<thead>
<tr>
<th>Class</th>
<th>AFF[∨]</th>
<th>renH[∨, ∃]</th>
<th>K/H[∨, ∃]</th>
<th>HORN[∨, ∃]</th>
<th>KROM[∨]</th>
</tr>
</thead>
<tbody>
<tr>
<td>CNF</td>
<td>( \not\in \mathcal{S} )</td>
<td>( \not\in \mathcal{S} )</td>
<td>( \not\in \mathcal{S} )</td>
<td>( \not\in \mathcal{S} )</td>
<td>( \not\in \mathcal{S} )</td>
</tr>
<tr>
<td>PI</td>
<td>( \not\in \mathcal{S} )</td>
<td>( \not\in \mathcal{S} )</td>
<td>( \not\in \mathcal{S} )</td>
<td>( \not\in \mathcal{S} )</td>
<td>( \not\in \mathcal{S} )</td>
</tr>
<tr>
<td>DNNF(_T)</td>
<td>( \not\in \mathcal{S} )</td>
<td>( \not\in \mathcal{S} )</td>
<td>( \not\in \mathcal{S} )</td>
<td>( \not\in \mathcal{S} )</td>
<td>( \not\in \mathcal{S} )</td>
</tr>
<tr>
<td>d-DNNF</td>
<td>( \not\in \mathcal{S} )</td>
<td>( \not\in \mathcal{S} )</td>
<td>( \not\in \mathcal{S} )</td>
<td>( \not\in \mathcal{S} )</td>
<td>( \not\in \mathcal{S} )</td>
</tr>
<tr>
<td>DNF</td>
<td>( \lfloor \mathcal{S} \rfloor \geq s )</td>
<td>( \lfloor \mathcal{S} \rfloor \geq s )</td>
<td>( \lfloor \mathcal{S} \rfloor \geq s )</td>
<td>( \lfloor \mathcal{S} \rfloor \geq s )</td>
<td>( \lfloor \mathcal{S} \rfloor \geq s )</td>
</tr>
<tr>
<td>IP</td>
<td>( \lfloor \mathcal{S} \rfloor \geq s )</td>
<td>( \lfloor \mathcal{S} \rfloor \geq s )</td>
<td>( \lfloor \mathcal{S} \rfloor \geq s )</td>
<td>( \lfloor \mathcal{S} \rfloor \geq s )</td>
<td>( \lfloor \mathcal{S} \rfloor \geq s )</td>
</tr>
<tr>
<td>OBDD(_&lt;)</td>
<td>( \not\in \mathcal{S} )</td>
<td>( \not\in \mathcal{S} )</td>
<td>( \not\in \mathcal{S} )</td>
<td>( \not\in \mathcal{S} )</td>
<td>( \not\in \mathcal{S} )</td>
</tr>
</tbody>
</table>

Table 12: Comparing w.r.t. succinctness the full disjunctive closures of KROM, HORN, K/H, renH, and AFF, with other classes of propositional representations.

Table 13: In the proof of Proposition 11, we have shown that the AFF formula \( \alpha_n = \bigwedge_{i=1}^{n} (x_i \oplus y_i \oplus z_i \oplus \top) \) has no polynomially-sized renH[∨, ∃] representation. The point is that \( \alpha_n \) has a polynomially-sized PI representation (consisting in \( 4n \) clauses: \( \neg x_i \lor \neg y_i \lor \neg z_i \lor x_i \lor y_i \lor z_i \lor x_i \lor y_i \lor \neg z_i \) for each \( i \in 1, \ldots, n \)), and a polynomially-sized OBDD\(_<\) representation for every ordering \( < \) which is such that \( x_i, y_i, z_i \) (\( i \in 1, \ldots, n \)) are successive elements. Indeed, for each \( i \in 1, \ldots, n \), one can generate in constant time an OBDD\(_<\) representation equivalent to each \( x_i \oplus y_i \oplus z_i \oplus \top \) and then, starting with the OBDD\(_<\) representation of \( x_1 \oplus y_1 \oplus z_1 \oplus \top \), in an iterative way, replace the \( \top \) sink of the current OBDD\(_<\) representation by the root of the next OBDD\(_<\) representation.

Furthermore, in the proof of Proposition 11, we proved that the formula \( \alpha_n = \bigwedge_{i=1}^{n} (\neg x_i \lor \neg y_i) \) (for any \( n \)) does not have a polynomial-size AFF[∨] representation. The point is that \( \alpha_n \) is a PI formula, and it also has a polynomially-sized OBDD\(_<\) representation for every ordering \( < \) which is such that \( x_i, y_i \) (\( i \in 1, \ldots, n \)) are successive elements. Indeed, for each \( i \in 1, \ldots, n \), one can generate in con-

\(^{10}\)A prime implicant \( \gamma \) of a formula \( \alpha \) is **essential** iff the disjunction of all prime implicants of \( \alpha \) except \( \gamma \) (up to logical equivalence) is not equivalent to \( \alpha \).
Table 13: Comparing w.r.t. succinctness the full disjunctive closures of KROM, HORN, K/H, renH, and AFF, with other classes of propositional representations.

<table>
<thead>
<tr>
<th></th>
<th>AFF[∨]</th>
<th>renH[∨, ∃]</th>
<th>K/H[∨, ∃]</th>
<th>HORN[∨, ∃]</th>
<th>KROM[∨]</th>
</tr>
</thead>
<tbody>
<tr>
<td>CNF</td>
<td>Zs, Zs</td>
<td>Zs, Zs</td>
<td>Zs, Zs</td>
<td>Zs, Zs</td>
<td>Zs, Zs</td>
</tr>
<tr>
<td>PI</td>
<td>Zs, Zs</td>
<td>Zs, Zs</td>
<td>Zs, Zs</td>
<td>Zs, Zs</td>
<td>Zs, Zs</td>
</tr>
<tr>
<td>DNNF_T</td>
<td>Zs</td>
<td>Zs</td>
<td>Zs</td>
<td>Zs</td>
<td>Zs</td>
</tr>
<tr>
<td>d-DNNF</td>
<td>Zs</td>
<td>Zs</td>
<td>Zs</td>
<td>Zs</td>
<td>Zs</td>
</tr>
<tr>
<td>DNF</td>
<td>Zs, ≥s</td>
<td>Zs, ≥s</td>
<td>Zs, ≥s</td>
<td>Zs, ≥s</td>
<td>Zs, ≥s</td>
</tr>
<tr>
<td>IP</td>
<td>Zs, ≥s</td>
<td>Zs, ≥s</td>
<td>Zs, ≥s</td>
<td>Zs, ≥s</td>
<td>Zs, ≥s</td>
</tr>
<tr>
<td>OBDD&lt;</td>
<td>Zs, Zs</td>
<td>Zs, Zs</td>
<td>Zs, Zs</td>
<td>Zs, Zs</td>
<td>Zs, Zs</td>
</tr>
</tbody>
</table>

Table 13: Comparing w.r.t. succinctness the full disjunctive closures of KROM, HORN, K/H, renH, and AFF, with other classes of propositional representations.

Given that PI ≥_p CNF, OBDD< ≥_p DNNF, OBDD< ≥_p DNNF_T, and the succinctness relationships given in Proposition 11, by transitivity of ≤_s, we get the results given in Table 13.

Table 14: As to DNNF_T, it is enough to show that the family of circular bit shift functions cbs_m have polynomially-sized representations in KROM[∨], HORN[∨], K/H[∨], and AFF[∨]. Indeed, it has been proven that such functions do not have polynomially-sized SDNNF representations, where SDNNF is the union of DNNF_T for all vtrees T [20].

For any positive integer m, consider the following Boolean function over 2^{m+1} + m variables cbs_m(x_0, ..., x_{2^m-1}, y_0, ..., y_{2^m-1}, i_0, ..., i_{m-1}) which is the semantics of the formula α_m =

\[ \bigvee_{b_0, ..., b_{m-1} \in \{0, 1\}} \left( \bigwedge_{j=0}^{m-1} i_j^{b_j} \land \bigwedge_{j=0}^{2^m-1} x_j \Leftrightarrow y(j + \text{num}(b_0, ..., b_{m-1})) \mod 2^m \right), \]

whose size is linear in the number of variables of cbs_m. In this formula, i_j^{b_j} denotes the literal i_j when b_j = 0 and the literal \neg i_j when b_j = 1; num is the mapping from \{0, 1\}^m to the set of natural numbers which gives the integer represented
by the binary string $b_0 \ldots b_{m-1}$. Thus, the variables $i_0, \ldots, i_{m-1}$ make precise how the bits of the binary string $y_0 \ldots y_{2^m-1}$ must be (circularly) shifted, and $cbs_m(x_0, \ldots, x_{2^m-1}, y_0, \ldots, y_{2^m-1}, i_0, \ldots, i_{m-1}) = 1$ exactly when the variables $x_0, \ldots, x_{2^m-1}$ and the shifted variables $y_0, \ldots, y_{2^m-1}$ are pairwise equal.

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>CNF</td>
<td>$\xi_s, \xi_s$</td>
<td>$\xi_s, \xi_s$</td>
<td>$\xi_s, \xi_s$</td>
<td>$\xi_s, \xi_s$</td>
<td>$\xi_s, \xi_s$</td>
</tr>
<tr>
<td>PI</td>
<td>$\xi_s, \xi_s$</td>
<td>$\xi_s, \xi_s$</td>
<td>$\xi_s, \xi_s$</td>
<td>$\xi_s, \xi_s$</td>
<td>$\xi_s, \xi_s$</td>
</tr>
<tr>
<td>DNNF</td>
<td>$\gamma_s, \gamma_s$</td>
<td>$\gamma_s, \gamma_s$</td>
<td>$\gamma_s, \gamma_s$</td>
<td>$\gamma_s, \gamma_s$</td>
<td>$\gamma_s, \gamma_s$</td>
</tr>
<tr>
<td>DNF</td>
<td>$\gamma_s, \gamma_s$</td>
<td>$\gamma_s, \gamma_s$</td>
<td>$\gamma_s, \gamma_s$</td>
<td>$\gamma_s, \gamma_s$</td>
<td>$\gamma_s, \gamma_s$</td>
</tr>
<tr>
<td>IP</td>
<td>$\gamma_s, \gamma_s$</td>
<td>$\gamma_s, \gamma_s$</td>
<td>$\gamma_s, \gamma_s$</td>
<td>$\gamma_s, \gamma_s$</td>
<td>$\gamma_s, \gamma_s$</td>
</tr>
<tr>
<td>OBDD</td>
<td>$\gamma_s, \gamma_s$</td>
<td>$\gamma_s, \gamma_s$</td>
<td>$\gamma_s, \gamma_s$</td>
<td>$\gamma_s, \gamma_s$</td>
<td>$\gamma_s, \gamma_s$</td>
</tr>
</tbody>
</table>

Table 14: Comparing w.r.t. succinctness the full disjunctive closures of KROM, HORN, K/H, renH, and AFF, with other classes of propositional representations.

For each $b_0, \ldots, b_{m-1} \in \{0, 1\}$, the formula $\beta_{b_0,\ldots,b_{m-1}} =$

$$\bigwedge_{j=0}^{m-1} i_j^{b_j} \land \bigwedge_{j=0}^{2^m-1} x_j \Leftrightarrow y(j + \text{num}(b_0,\ldots,b_{m-1})) \mod 2^m$$

is equivalent to the KROM formula $\gamma_{b_0,\ldots,b_{m-1}} =$

$$\bigwedge_{j=0}^{m-1} i_j^{b_j} \land \bigwedge_{j=0}^{2^m-1} (\neg x_j \lor y(j + \text{num}(b_0,\ldots,b_{m-1})) \mod 2^m)$$

$$\land \bigwedge_{j=0}^{2^m-1} (x_j \lor \neg y(j + \text{num}(b_0,\ldots,b_{m-1})) \mod 2^m).$$

Clearly enough, $\gamma_{b_0,\ldots,b_{m-1}}$ also is a HORN formula, hence it is a K/H formula and a renH formula. Similarly, $\beta_{b_0,\ldots,b_{m-1}}$ is also equivalent to the AFF formula $\delta_{b_0, \ldots, b_{m-1}} =$

$$\bigwedge_{j=0}^{m-1} \text{lit}(i_j, b_j) \land \bigwedge_{j=0}^{2^m-1} x_j \oplus y(j + \text{num}(b_0,\ldots,b_{m-1})) \mod 2^m \oplus T$$

76
where \( \text{lit}(i_j, b_j) = i_j \) when \( b_j = 0 \) and \( \text{lit}(i_j, b_j) = i_j \oplus \top \) when \( b_j = 1 \). Both \( \gamma_{b_0, \ldots, b_{m-1}} \) and \( \delta_{b_0, \ldots, b_{m-1}} \) can be computed in time linear in the size of \( \beta_{b_0, \ldots, b_{m-1}} \), hence linear in the number of variables of \( \text{cbs}_m \).

As a consequence, \( \bigvee_{b_0, \ldots, b_{m-1} \in \{0, 1\}} \gamma_{b_0, \ldots, b_{m-1}} \) is a \( \text{KROM} [\lor] \) (and a \( \text{HORN} [\lor] \), a \( \text{K/H} [\lor] \), a \( \text{renH} [\lor] \)) formula equivalent to \( \alpha_m \), and \( \bigvee_{b_0, \ldots, b_{m-1} \in \{0, 1\}} \delta_{b_0, \ldots, b_{m-1}} \) is a \( \text{AFF} [\lor] \) formula equivalent to \( \alpha_m \). The fact that the size of any of

\[
\bigvee_{b_0, \ldots, b_{m-1} \in \{0, 1\}} \gamma_{b_0, \ldots, b_{m-1}}
\]

and

\[
\bigvee_{b_0, \ldots, b_{m-1} \in \{0, 1\}} \delta_{b_0, \ldots, b_{m-1}}
\]

is linear in the number of variables of \( \text{cbs}_m \) completes the proof.

As to \( d\text{-DNNF} \), the result comes easily from the fact that \( d\text{-DNNF} \) is not at least as succinct as \( \text{DNF} \), unless the polynomial hierarchy collapses \([1]\), plus the fact that \( \text{DNF} \) is polynomially translatable into the disjunction closure and into the full disjunctive closure of each of \( \text{KROM}, \text{HORN}, \text{K/H}, \) and \( \text{renH} \).

We finally get the results given in Table 14.