Existential Closures for Knowledge Compilation

Pierre Marquis
CRIL-CNRS, Université d’Artois,
Lens, France
marquis@cril.univ-artois.fr

Abstract
We study the existential closures of several propositional languages \( L \) considered recently as target languages for knowledge compilation (KC), namely the incomplete fragments KROM-C, HORN-C, K/H-C, renH-C, AFF, and the corresponding disjunction closures KROM-C[\( \lor \)], HORN-C[\( \lor \)], K/H-C[\( \lor \)], renH-C[\( \lor \)], and AFF[\( \lor \)]. We analyze the queries, transformations, expressiveness and succinctness of the resulting languages \( L[\exists] \) in order to locate them in the KC map. As a by-product, we also address several issues concerning disjunction closures that were left open so far. From our investigation, the language HORN-C[\( \lor, \exists \)] (where disjunctions and existential quantifications can be applied to Horn CNF formulae) appears as an interesting target language for the KC purpose, challenging the influential DNNF languages.

1 Introduction
Knowledge compilation (KC) is concerned with preprocessing for improving the efficiency of computational tasks (see among others [Darwiche, 2001; Cadoli and Donini, 1998; Selman and Kautz, 1996; Schrag, 1996; del Val, 1994]). An important issue within this research area is the choice of a target language into which some pieces of data are to be translated during the off-line compilation phase [Gogic et al., 1995; Darwiche and Marquis, 2002]. In [Darwiche and Marquis, 2002], the authors argue that such a choice must be based both on the set of queries and transformations which can be achieved in polynomial time once the pre-processed pieces of data are represented in the target language, as well as the succinctness of the language (i.e., its ability to represent data using few space.) The basic queries considered in [Darwiche and Marquis, 2002] include tests for consistency, validity, implicates (clausal entailment), implicants, equivalence, sentential entailment, counting and enumerating theory models (CO, VA, CE, IM, EQ, SE, CT, ME.) The basic transformations are conditioning (CD), closures under the connectives (\( \land, \land \lor, \land \lor \lor, \land \lor \land \lor, \land \neg \)), and forgetting (FO, SFO.) The KC map given in [Darwiche and Marquis, 2002] is an evaluation of a dozen of significant propositional languages w.r.t. several criteria: the succinctness of the language and the set of queries and transformations it supports in polynomial time. Such a map can be used as a guide for targeting the “right language” given the requirements imposed by the application under consideration.

The KC map provided in [Darwiche and Marquis, 2002] has been extended to incorporate further propositional languages (also referred to as “fragments”), queries and transformations, see among others [Wachter and Haenni, 2006], [Fargier and Marquis, 2006], [Subbarayan et al., 2007], [Fargier and Marquis, 2008b], [Fargier and Marquis, 2008a], [Fargier and Marquis, 2009]. In [Fargier and Marquis, 2008b; 2008a] an approach to define new target languages for KC has been pointed out. It consists in applying closure principles to previous languages. [Fargier and Marquis, 2008a] consider two disjunctive closure principles: disjunction (\( \lor \)) and existential closure (\( \exists \)). Intuitively, the disjunction principle when applied to a propositional language \( L \) leads to a language \( L[\lor] \) which qualifies disjunctions of formulae from \( L \), while existential closure applied to \( L \) leads to a language \( L[\exists] \) which qualifies existentially quantified formulae from \( L \). Whatever \( L \), \( L[\lor] \) satisfies polytime closure under \( \lor \) and \( L[\exists] \) satisfies polytime forgetting. Applying any/both of those two principles to \( L \) may lead to new fragments, which can prove strictly more succinct than \( L \). Thus, [Fargier and Marquis, 2008a] locate on the KC map all languages obtained by applying those closure principles to some complete languages, i.e., languages \( L \) for which every propositional formula has an equivalent in \( L \). Thus, the disjunctive closures of the languages OBDD, \( \land \lor \land \lor \land \lor \land \lor \land \neg \) (ordered binary decision diagrams), DNF (disjunctive normal forms), DNNF (decomposable negation normal forms), CNF (conjunctive normal forms), PI (prime implicates), IP (prime implicants), MODS (models), considered in [Darwiche and Marquis, 2002], have been studied in [Fargier and Marquis, 2008a]. On the other hand, [Fargier and Marquis, 2008b] consider the disjunction closures KROM-C[\( \lor \)], HORN-C[\( \lor \)], K/H-C[\( \lor \)], renH-C[\( \lor \)], and AFF[\( \lor \)], composed respectively of disjunctions of Krom CNF formulae, disjunctions of Horn CNF formulae, disjunctions of Krom or Horn CNF formulae, disjunctions of renamable Horn CNF formulae, and disjunctions of affine formulae.

*This research was supported by the project TUPLES-ANR-10-BLAN-0210 of the French National Agency for Research - Agence Nationale de la Recherche
Each of these languages is complete, unlike the underlying languages KROM-C (also known as the bijunctive fragment) [Krom, 1970], HORN-C [Horn, 1951], AFF (also known as the biconditional fragment) [Schaefer, 1978], K/H−C (Krom or Horn CNF formulae) and renH−C.

In the following, we complete the results reported in [Fargier and Marquis, 2008b; 2008a] by focusing on the existential closures of the ten languages KROM-C, HORN-C, K/H−C, renH−C, AFF, KROM−C[V], HORN−C[V], K/H−C[V], renH−C[V], and AFF[V]. We evaluate each of them along the lines considered in [Darwiche and Marquis, 2002]. The contribution of the paper is mainly as follows:

- For each existential closure of KROM−C, HORN−C, K/H−C, renH−C, AFF, KROM−C[V], HORN−C[V], K/H−C[V], renH−C[V], and AFF[V], we identify the queries and transformations which are feasible in polynomial time, and those which are not (possibly under some standard assumptions of complexity theory).
- We demonstrate that the existential closure of each language \( L \) among the ten languages above is just as expressive as \( L \). As to succinctness, we prove that \( L[\emptyset] \) is strictly more succinct than \( L \), except for \( L = KROM−C \) and \( L = AFF \) since for those two languages, \( L[\emptyset] \) and \( L \) are polynomially equivalent.
- We complete the results about disjunction closures provided in [Fargier and Marquis, 2008b]; especially, we show that FO is not satisfied by any of HORN−C[V], K/H−C[V], renH−C[V].
- We show that neither d−DNNF nor SDNNF is strictly more succinct than any disjunction closure of KROM−C, HORN−C, K/H−C, renH−C, AFF; it shows such disjunction closures (and the existential closures of them) as interesting alternatives to DNNF languages.

The rest of the paper is organized as follows: in Section 2, the queries and transformations considered in the KC map, as well as the fundamental notions of expressiveness and succinctness, are recalled. The notions of disjunctive, disjunction and existential closure are also presented. In Section 3, our new results are presented. Section 4 discusses the results and provides some perspectives.

2 The KC Map and Disjunctive Closures

In this paper, we consider subsets of the propositional language \( \text{QDAG} \) of quantified propositional DAGs. \( \text{QDAG} \) is given by:

**Definition 1 (QDAG)** Let \( PS \) be a denumerable set of propositional variables. \( \text{QDAG} \) is the set of all finite, single-rooted DAGs \( \alpha \) where:

- each leaf node of \( \alpha \) is labeled by a literal \( l \) over \( PS \), or by a Boolean constant \( \top \) (always true) or \( \bot \) (always false);
- each internal node of \( \alpha \) is labeled by a connective \( c \in \{\land, \lor, \neg, \oplus\} \) and has as many children as required by \( c \), or is labeled by \( \exists x \) (where \( x \in PS \)) and has a single child.

All the propositional languages considered so far as target languages to KC are subsets of \( \text{QDAG} \), and typically of \( \text{DAG} \), the subset of \( \text{QDAG} \) where no node labeled by a quantification is allowed. Especially, the fragments considered in [Wachter and Haenni, 2006] are included in the subset of \( \text{DAG} \) where \( c \) is among \( \{\land, \lor, \neg\} \), and the fragments considered in [Darwiche and Marquis, 2002] (especially OBDD\(<\), DNF, DNNF, CNF, PI, IP, MODS) are subsets of \( \text{DAG−NNF} \) (the subset of the latter when \( \neg \) is disallowed.)

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. \( T \) is the complementary literal of literal \( l \), so that \( T = \bot, \bot = T, x = \neg x \) and \( \neg \neg x = x \). For a literal \( l \) different from a Boolean constant, \( var(l) \) denotes the corresponding variable: for \( x \in PS \), we have \( var(x) = x \) and \( var(\neg x) = x \). A clause (resp. a term) is a finite disjunction (resp. conjunction) of literals. An XOR-clause is a finite XOR-disjunction of literals (the XOR connective is denoted by \( \oplus \)).

Each element \( \alpha \) of \( \text{QDAG} \) is called a QDAG formula. \( Var(\alpha) \) denotes the set of free variables \( x \) 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 \( var(l) = x \) and there is a path from the root of \( \alpha \) to \( n_x \) such that no node from it is labelled by \( \exists x \). The size \( |\alpha| \) of a QDAG formula \( |\alpha| \) is the number of nodes plus the number of arcs in the DAG.

Figure 1 presents a QDAG formula. Its set of free variables is \( \{q, r\} \). The DAG rooted at the \( \Delta \) node is a DAG formula and the DAG rooted at the \( \lor \) node is a DAG−NNF formula.

In the following, we consider the next queries and transformations:

**Definition 2 (queries)** Let \( L \subseteq \text{QDAG} \).

- \( L \) satisfies CO (resp. VA) iff there exists a polytime algorithm that maps every formula \( \alpha \) from \( L \) to 1 if \( \alpha \) is consistent (resp. valid), and to 0 otherwise.
- \( L \) satisfies CE iff there exists a polytime algorithm that maps every formula \( \alpha \) from \( L \) and every clause \( \gamma \) to 1 if \( \alpha \models \gamma \) holds, and to 0 otherwise.
\begin{itemize}
  \item $\mathcal{L}$ satisfies **EQ** (resp. **SE**) if there exists a polytime algorithm that maps every pair of formulae $\alpha, \beta$ from $\mathcal{L}$ to $1$ if $\alpha \equiv \beta$ (resp. $\alpha \models \beta$) holds, and to $0$ otherwise.
  
  \item $\mathcal{L}$ satisfies **IM** if there exists a polytime algorithm that maps every formula $\alpha$ from $\mathcal{L}$ to a nonnegative integer that represents the number of models of $\alpha$ over $\text{Var}(\alpha)$ (in binary notation.)
  
  \item $\mathcal{L}$ satisfies **ME** if there exists a polynomial $p(\ldots)$ and an algorithm that outputs all models of an arbitrary formula $\alpha$ from $\mathcal{L}$ in time $p(n, m)$, where $n$ is the size of $\alpha$ and $m$ is the number of its models (over $\text{Var}(\alpha)$.)
  
  \item $\mathcal{L}$ satisfies **MC** (model checking) if there exists a polytime algorithm that maps every formula $\alpha$ from $\mathcal{L}$ and every interpretation $\omega$ over $\text{Var}(\alpha)$ (represented as a term) to $1$ if $\omega$ is a model of $\alpha$, and to $0$ otherwise.

**Definition 3 (transformations)** Let $\mathcal{L} \subseteq \text{QDAG}$.

- $\mathcal{L}$ satisfies **CD** (resp. **VC**) if there exists a polytime algorithm that maps every formula $\alpha$ from $\mathcal{L}$ and every consistent term $\gamma$ to a formula from $\mathcal{L}$ that is logically equivalent to the conditioning $\alpha \mid \gamma$ of $\alpha$ on $\gamma$, i.e., the formula obtained by replacing each free occurrence of variable $x$ of $\alpha$ by $\top$ (resp. $\bot$) if $x$ (resp. $\neg x$) is a positive (resp. negative) literal of $\gamma$.
  
- $\mathcal{L}$ satisfies **FO** if there exists a polytime algorithm that maps every formula $\alpha$ from $\mathcal{L}$ and every subset $X$ of variables from $\text{PS}$ to a formula from $\mathcal{L}$ equivalent to $\exists X. \alpha$. If the property holds for each singleton $X$, we say that $\mathcal{L}$ satisfies **SFO**.
  
- $\mathcal{L}$ satisfies $\land C$ (resp. $\lor C$) if there exists a polytime algorithm that maps every finite set of formulae $\alpha_1, \ldots, \alpha_n$ from $\mathcal{L}$ to a formula of $\mathcal{L}$ that is logically equivalent to $\alpha_1 \land \ldots \land \alpha_n$ (resp. $\alpha_1 \lor \ldots \lor \alpha_n$.)
  
- $\mathcal{L}$ satisfies $\land BC$ (resp. $\lor BC$) if there exists a polytime algorithm that maps every pair of formulae $\alpha$ and $\beta$ from $\mathcal{L}$ to a formula of $\mathcal{L}$ that is logically equivalent to $\alpha \land \beta$ (resp. $\alpha \lor \beta$.)
  
- $\mathcal{L}$ satisfies $\sim C$ if there exists a polytime algorithm that maps every formula $\alpha$ from $\mathcal{L}$ to a formula of $\mathcal{L}$ logically equivalent to $\sim \alpha$.

We also consider the following notions of expressiveness and succinctness:

**Definition 4 (expressiveness)** Let $\mathcal{L}_1$ and $\mathcal{L}_2$ be two subsets of QDAG. $\mathcal{L}_1$ is at least as expressive as $\mathcal{L}_2$, denoted $\mathcal{L}_1 \leq_c \mathcal{L}_2$, iff for every formula $\alpha \in \mathcal{L}_2$, there exists an equivalent formula $\beta \in \mathcal{L}_1$.

**Definition 5 (succinctness)** Let $\mathcal{L}_1$ and $\mathcal{L}_2$ be two subsets of QDAG. $\mathcal{L}_1$ is at least as succinct as $\mathcal{L}_2$, denoted $\mathcal{L}_1 \leq_s \mathcal{L}_2$, iff there exists a polynomial $p$ such that for every formula $\alpha \in \mathcal{L}_2$, there exists an equivalent formula $\beta \in \mathcal{L}_1$ where $|\beta| \leq p(|\alpha|)$.

Finally, we take advantage of the following restriction of the succinctness relation:

**Definition 6 (polynomial translation)** Let $\mathcal{L}_1$ and $\mathcal{L}_2$ be two subsets of QDAG. $\mathcal{L}_1$ is said to be polynomially translatable into $\mathcal{L}_2$, noted $\mathcal{L}_1 \triangledown_p \mathcal{L}_2$, iff there exists a polytime algorithm $f$ such that for every $\alpha \in \mathcal{L}_1$, we have $f(\alpha) \in \mathcal{L}_2$ and $f(\alpha) \equiv \alpha$.

Whenever $\mathcal{L}_1$ is polynomially translatable into $\mathcal{L}_2$, every query which is supported in polynomial time in $\mathcal{L}_2$ also is supported in polynomial time in $\mathcal{L}_1$; and conversely, every query which is not supported in polynomial time in $\mathcal{L}_1$ unless $P = NP$ (resp. unless the polynomial hierarchy PH collapses) cannot be supported in polynomial time in $\mathcal{L}_2$, unless $P = NP$ (resp. unless PH collapses.)

$\sim_c$ is the symmetric part of $\leq_c$ defined by $\mathcal{L}_1 \sim_c \mathcal{L}_2$ iff $\mathcal{L}_1 \leq_c \mathcal{L}_2$ and $\mathcal{L}_2 \leq_c \mathcal{L}_1$. $\sim_c$ is the asymmetric part of $\leq_c$ defined by $\mathcal{L}_1 \triangleleft_c \mathcal{L}_2$ iff $\mathcal{L}_1 \leq_c \mathcal{L}_2$ and $\mathcal{L}_2 \not\leq_c \mathcal{L}_1$. Similarly, $\sim_s$ (resp. $\triangleleft_s$) is the symmetric part of $\leq_s$ (resp. $\leq_p$) when the operators are among $\land$, $\lor$, or $\sim$. When $\mathcal{L}_1 \sim_s \mathcal{L}_2$, $\mathcal{L}_1$ and $\mathcal{L}_2$ are said to be polynomially equivalent. Obviously enough, polynomially equivalent fragments are equally efficient (and succinct) and possess the same set of tractable queries and transformations.

We are now ready to present disjunctive closures. Intuitively, a closure principle applied to a propositional fragment $\mathcal{L}$ defines a new propositional language, called a closure of $\mathcal{L}$, through the application of “operators” (i.e., connectives or quantifications.) The resulting closure is said to be disjunctive when the operators are among $\lor$ and $\exists x$ with $x \in \text{PS}$. Formally:

**Definition 7 (disjunctive closures)** Let $\mathcal{L} \subseteq \text{QDAG}$ and $\Delta \subseteq \{\lor, \exists\}$. The closure $\mathcal{L}[\Delta]$ of $\mathcal{L}$ by $\Delta$ is the subset of $\text{QDAG}$ inductively defined as follows:

1. if $\alpha \in \mathcal{L}$, then $\alpha \in \mathcal{L}[\Delta]$.
2. if $\forall \in \Delta$ and $\alpha_i \in \mathcal{L}[\Delta]$ for each $i = 1, \ldots, n$, then $\forall (\alpha_1, \ldots, \alpha_n) \in \mathcal{L}[\Delta]$.
3. if $\exists \in \Delta$, $x \in \text{PS}$, and $\alpha \in \mathcal{L}[\Delta]$, then $\exists x. \alpha \in \mathcal{L}[\Delta]$.

Thus, an element of $\mathcal{L}[\Delta]$ can be viewed as a “tree” which “internal nodes” are labelled by quantifications of the form $\exists x$ or by $\forall$ and its “leaf nodes” are labelled by elements of $\mathcal{L}$. Accordingly, the formulae $\alpha_i$ considered in item 2. of Definition 7 do not share any common subgraphs.

The set $D(\mathcal{L})$ (resp. $\forall(\mathcal{L}), \exists(\mathcal{L})$) of all disjunctive (resp. disjunction, existential) closures of a subset $\mathcal{L}$ of QDAG is defined inductively by:

- $\mathcal{L}$ belongs to $D(\mathcal{L})$ (resp. $\forall(\mathcal{L}), \exists(\mathcal{L})$);
- If $\mathcal{L}$ belongs to $D(\mathcal{L})$ (resp. $\forall(\mathcal{L}), \exists(\mathcal{L})$) and $\Delta \subseteq \{\forall, \exists\}$ (resp. $\Delta \subseteq \{\forall\}$, $\Delta \subseteq \{\exists\}$), then $\mathcal{L}[\Delta]$ belongs to $D(\mathcal{L})$ (resp. $\forall(\mathcal{L}), \exists(\mathcal{L})$).

[Farfarg and Marquis, 2008a] provide several general-scope characterization results for disjunctive closures. Especially, they show that for a given $\mathcal{L} \subseteq \text{DAG}$, only three disjunctive closures are worth to be considered since $\mathcal{L}[\exists[\exists]] = \mathcal{L}[\forall[\forall]]$.  

\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 $\mathcal{L}[\{\delta_1, \ldots, \delta_n\}]$.}
\[\mathcal{L} \setminus \mathcal{L}_3, \mathcal{L}[v] = \mathcal{L}[v] \text{ and } \mathcal{L}[\exists] \sim_p \mathcal{L}[\exists], \mathcal{L}[\exists] \sim_p \mathcal{L}[v, \exists] \]

(see item 4. of Proposition 1 and item 1. of Proposition 2 in [Fargier and Marquis, 2008a].) They also study the disjunctive closures of the languages OBDD, DNF, DNNF, CNF, PI, IP, MODS considered in [Darwiche and Marquis, 2002].

3 Analyzing New Existential Closures

[Fargier and Marquis, 2008b] focus on the disjunctive closures of the following incomplete fragments:

**Definition 8 (some incomplete fragments)**

- **KROM-C** is the subset of CNF consisting of formulae in which each clause is binary, i.e., it contains at most two literals.
- **HORN-C** is the subset of CNF consisting of formulae in which each clause is Horn, i.e., it contains at most one positive literal.
- **K/H-C** is the union of KROM-C and HORN-C.
- **renH-C** is the subset of CNF consisting of formulae \( \alpha \) for which there exists a subset \( V \) of \( \text{Var}(\alpha) \) (called a Horn renaming for \( \alpha \)) such that the formula \( V(\alpha) \) obtained by substituting in \( \alpha \) every literal \( l \) over \( V \) by its complementary literal \( \bar{l} \) is a HORN-C formula.
- **AFF** is the subset of DAG consisting of conjunctions of XOR clauses.

In the following, we complete the results provided in [Fargier and Marquis, 2008b] by studying the queries, transformations, expressiveness and succinctness of the existential closures of the propositional fragments considered in [Fargier and Marquis, 2008b], i.e., we consider the languages KROM-C, HORN-C, K/H-C, renH-C, AFF, KROM-C, HORN-C, K/H-C, renH-C, AFF.

\[
\begin{array}{c}
V \\
\exists q \\
\wedge \\
\neg r \\
\neg q \\
\neg r \\
\neg s \\
\neg q \\
q \\
\neg p
\end{array}
\]

Figure 2: A HORN – C formula.

Figure 2 presents a HORN – C formula equivalent to the CNF formula \((\neg p \lor \neg r) \land (\neg r \lor \neg s) \land (\neg q \lor \neg s)\).

We know that each of KROM-C and AFF satisfies FO [Fargier and Marquis, 2008b]. Furthermore, for any \( \mathcal{L} \subseteq \text{DAG} \), if \( \mathcal{L} \) satisfies FO, then \( \mathcal{L}[v] \) satisfies FO as well. As a consequence, we immediately get that

- **KROM-C** is a subset of each of them.) Conversely, each of HORN-C, K/H-C, renH-C is strictly less expressive than any complete language.

Furthermore, we have:

\[
\begin{align*}
\text{renH-C} & < \text{K/H-C} \sim \text{HORN-C} \\
\text{K/H-C} & \sim \text{renH-C}
\end{align*}
\]

Table 1: Queries. \( \sqrt{\text{meas}} \) means “satisfies” and \( \circ \) means “does not satisfy unless \( P = \text{NP} \).”

<table>
<thead>
<tr>
<th>( \mathcal{L} )</th>
<th>( \mathcal{L} \setminus \mathcal{L}_3 )</th>
<th>( \mathcal{L}_3 )</th>
<th>( \mathcal{L}[v] )</th>
<th>( \mathcal{L}[\exists] )</th>
<th>( \mathcal{L}[\exists], \mathcal{L}[\exists] )</th>
<th>( \mathcal{L}[v, \exists] )</th>
<th>( \mathcal{L}[\exists] )</th>
<th>( \mathcal{L}[\exists] )</th>
</tr>
</thead>
<tbody>
<tr>
<td>renH-C</td>
<td>( \vee )</td>
<td>( \circ )</td>
<td>( \circ )</td>
<td>( \circ )</td>
<td>( \circ )</td>
<td>( \circ )</td>
<td>( \circ )</td>
<td>( \circ )</td>
</tr>
<tr>
<td>K/H-C</td>
<td>( \vee )</td>
<td>( \circ )</td>
<td>( \circ )</td>
<td>( \circ )</td>
<td>( \circ )</td>
<td>( \circ )</td>
<td>( \circ )</td>
<td>( \circ )</td>
</tr>
<tr>
<td>HORN-C</td>
<td>( \vee )</td>
<td>( \circ )</td>
<td>( \circ )</td>
<td>( \circ )</td>
<td>( \circ )</td>
<td>( \circ )</td>
<td>( \circ )</td>
<td>( \circ )</td>
</tr>
<tr>
<td>renH-C</td>
<td>( \vee )</td>
<td>( \circ )</td>
<td>( \circ )</td>
<td>( \circ )</td>
<td>( \circ )</td>
<td>( \circ )</td>
<td>( \circ )</td>
<td>( \circ )</td>
</tr>
<tr>
<td>K/H-C</td>
<td>( \vee )</td>
<td>( \circ )</td>
<td>( \circ )</td>
<td>( \circ )</td>
<td>( \circ )</td>
<td>( \circ )</td>
<td>( \circ )</td>
<td>( \circ )</td>
</tr>
<tr>
<td>HORN-C</td>
<td>( \vee )</td>
<td>( \circ )</td>
<td>( \circ )</td>
<td>( \circ )</td>
<td>( \circ )</td>
<td>( \circ )</td>
<td>( \circ )</td>
<td>( \circ )</td>
</tr>
</tbody>
</table>

Table 2: Transformations. \( \sqrt{\text{meas}} \) means “satisfies,” while \( \circ \) means “does not satisfy unless \( P = \text{NP} \).”

As to queries and transformations, we have obtained the following results:

**Proposition 1** The results given in Table 1 and in Table 2 hold.

As to expressiveness, it turns out that for any language \( \mathcal{L} \) among K/H-C, HORN-C, renH-C, \( \text{K/H-C} \), \( \text{HORN-C} \), \( \text{renH-C} \), going from \( \mathcal{L} \) to \( \mathcal{L}[\exists] \) does not lead to a shift:

**Proposition 2** For every \( \mathcal{L} \) among K/H-C, HORN-C, renH-C, K/H-C, HORN-C, we have \( \mathcal{L} \sim \mathcal{L}[\exists] \).

As a consequence (see [Fargier and Marquis, 2008b]), we have \( \text{HORN-C} \sim \text{renH-C} \); these three languages are just equally expressive since they are complete for propositional logic (just observe that the complete language DNF is a subset of each of them.) Contrastingly, each of HORN-C, K/H-C, renH-C is strictly less expressive than any complete language.

Furthermore, we have:

\[
\begin{align*}
\text{renH-C} & < \text{K/H-C} \sim \text{HORN-C} \\
\text{K/H-C} & \sim \text{renH-C}
\end{align*}
\]
while HORN−C[3] and KROM−C[3] are incomparable w.r.t. \( \leq_c \) (and AFF[3], which is polynomially equivalent to AFF, is incomparable w.r.t. \( \leq_c \) with any of the three incomplete fragments above.)

As to succinctness, the picture is rather different. Typically going from \( L \) to \( L[3] \) leads to a succinctness increase:

**Proposition 3**

1. None of CNF or DNF is at least at succinct as any of HORN−C[3] \( \leq_h C[H−C[3]] \), or renH−C[3] \( \leq_h C[H−C[3]] \).

2. \( \forall \alpha \), the language containing all disjunctions of CNF formulae and all conjunctions of DNF formulae, is not at least as succinct as any of HORN−C[3] \( \leq_h C[H−C[3]] \) and renH−C[3] \( \leq_h C[H−C[3]] \).

Since HORN−C (resp. K/H−C, renH−C) is a subset of both CNF and HORN−C[3] (resp. K/H−C[3], renH−C[3]), item 1. of Proposition 3 shows that going from \( L \) to \( L[3] \) for any of these three languages leads to a strictly more succinct language. Furthermore, since HORN−C[3] (resp. K/H−C[3], renH−C[3]) is a subset of both \( \forall \alpha \) and HORN−C[3] (resp. K/H−C[3], renH−C[3]), item 2. of Proposition 3 shows that a similar conclusion can be drawn for those three languages, namely HORN−C[3] (resp. K/H−C[3], renH−C[3]) is strictly less succinct than \( \forall \alpha \leq_h C[H−C[3]] \) (resp. K/H−C[3], renH−C[3]). As a by-product, this shows that for sure, none of HORN−C[3], K/H−C[3], renH−C[3] satisfies FO, an issue left open in [Fargier and Marquis, 2008b].

Finally, we have also compared the disjunction closures of KROM−C, HORN−C, K/H−C, renH−C, AFF, PI with the two main subsets of DNNF, namely the set \( d−DNNF \) of deterministic DNNFs and the set \( SDNNF \) of structured DNNFs (the union for all \( T \) of all DNNF languages \( DNNF_T \) respecting \( T \)) (see [Pipatsrisawat and Darwiche, 2008] for details). What makes those subsets so significant is that every available compilation algorithm which outputs a DNNF formula actually computes a \( d−DNNF \) formula [Darwiche, 2001; 2004] or a \( SDNNF \) formula [Pipatsrisawat and Darwiche, 2008; 2010b]. First of all, as an easy consequence of the fact that \( d−DNNF \) is not at least as succinct as DNNF, unless PH collapses, we have that:

**Proposition 4** \( d−DNNF \) is not at least as succinct as any language among the existential closures of KROM−C[3], HORN−C[3], K/H−C[3], renH−C[3], AFF[3], PI[3], and OBDD[3], unless PH collapses.

This shows that the existential closures as interesting alternatives to \( d−DNNF \) for applications where FO, \( \forall C \) (and for some of them \( \forall BC \)) are required since such closures offer them while \( d−DNNF \) does not (the price to be paid is that VA, IM and CT are lost while they are offered by \( d−DNNF \)).

As to SDNNF, we have got that:

**Proposition 5** SDNNF is not at least as succinct as any language among the disjunctive closures of KROM−C, HORN−C, K/H−C, renH−C, AFF, PI, and CNF.

Thus, every language that is at most as succinct as SDNNF is not at least as succinct as any disjunctive closure of KROM−C, HORN−C, K/H−C, renH−C, AFF, PI, CNF. This includes in particular OBDD<, OBDD>[v], the language O−DDG of Ordered Decomposable Decision Diagrams [Fargier and Marquis, 2006], and the language AOMDD of AND/OR Multi-Valued Decision Diagrams (restricted to propositional variables) [Mateescu et al., 2008] (or equivalently the language SO−DDG of Strongly Ordered Decomposable Decision Diagrams [Fargier and Marquis, 2006]).

## 4 Conclusion and Perspectives

In this paper, we have studied the existential closures of both incomplete languages (KROM−C, HORN−C, K/H−C, renH−C, and AFF) and complete languages (the corresponding disjunction closures KROM−C[V], HORN−C[V], K/H−C[V], renH−C[V], and AFF[V]). The results given above show that for each complete language \( L \) under consideration, the corresponding existential closure \( L[3] \) is at least as good as \( L \) as a target language for KC. Indeed, \( L \) and \( L[3] \) satisfy the same queries and when \( L \) does not satisfy FO, \( L[3] \) offers it for free and is equally expressive but strictly more succinct than \( L \). For the incomplete languages considered here, namely KROM−C, HORN−C, K/H−C, renH−C, AFF, the same conclusions can be drawn as to transformations, expressiveness and succinctness; the price to be paid for getting FO for free and obtaining a strictly more succinct language is paid by the loss of the EQ and SE queries when the existential closures of KROM−C, K/H−C, renH−C are considered.

Our study also shows HORN−C[V, 3] as a valuable complete target language for the KC purpose. Indeed, HORN−C[V, 3] satisfies the same queries and transformations as DNNF \( T \) or AFF[V] (which is polynomially equivalent to AFF[V, 3].) Especially, HORN−C[V, 3] satisfies \( \forall BC \) and this paves the way for bottom-up compilation algorithms targeting HORN−C[V, 3]. As noted in [Pipatsrisawat and Darwiche, 2008], this is important for applications from formal verification based on unbounded model checking which require bottom-up, incremental compilation of formulae, where pieces of the knowledge base are compiled independently and then conjoined together.\(^3\) Furthermore, HORN−C[V, 3] guarantees some polynomial-sized representations for families of propositional formulae, like the CNF representations of circular bit shift functions (resp. the CNF formulae \( \alpha_n = \bigwedge_{i=1}^n (\neg x_i \lor y_i) \)), which have only exponential-sized SDNNF representations (resp. exponential-sized AFF[V] representations, see [Fargier and Marquis, 2008b]). Compared to \( d−DNNF \), while it does not satisfy the queries VA, IM and CT, HORN−C[V, 3] offers the transformations FO, \( \forall C \) and \( \forall BC \). Finally, \( d−DNNF \) is not at least as succinct as HORN−C[V, 3] unless PH collapses.

From the practical side, it is important to note that some empirical evidence in favour of HORN−C[V, 3] compilations and renH−C[V, 3] already exists. On the one hand, [Nishimura et al., 2004] (resp. [Samer and Szeider, 2008]) have shown that the problem of determining whether a given

\( ^3 \)Since HORN−C[V, 3] satisfies CO and CL (i.e., every propositional clause has a polynomial-sized representation in the language), getting \( \forall BC \) is optimal in the sense that no propositional language can satisfy both \( \forall C, CO \), and CL, unless \( P = NP \).
CNF formula $\alpha$ has a strong HORN–C-backdoor set (resp. a HORN–C-backdoor tree) containing at most $k$ variables (resp. leaves) is fixed-parameter tractable with parameter $k$. Interestingly, the algorithm given in [Samer and Szeider, 2008] can be used to determine "efficiently" (i.e., for sufficiently "small" $k$) whether a HORN $\models \exists \Delta$ compilation of reasonable size (i.e., linear in $k$ and the size of $\alpha$) exists. As mentioned in [Samer and Szeider, 2008]: “There is some empirical evidence that real-world instances actually have small backdoor sets”. On the other hand, [Boufkhad et al., 1997] present some compilation algorithms targeting respectively HORN $\models \exists \Delta$ and reHORN $\models \exists \Delta$, and evaluate them on a number of benchmarks. While such results show the feasibility of HORN $\models \exists \Delta$ compilation and reHORN $\models \exists \Delta$ compilation, we can hardly use those results to compare the two target fragments HORN $\models \exists \Delta$ and reHORN $\models \exists \Delta$ with OBDD and DNNF for which some experimental results are also available. Indeed, the compilation algorithms given in [Boufkhad et al., 1997] 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. Accordingly, the main perspective of this work is to develop and evaluate compilation algorithms based on a CDCL SAT solver, and which target HORN $\models \exists \Delta$ and reHORN $\models \exists \Delta$.

Acknowledgments

The author would like to thank the anonymous reviewers for their helpful comments.

References


Appendix

Proof:[Proposition 1]
We start with the queries:

- **CO:** Direct from item 1. of Proposition 4 from [Fargier and Marquis, 2008a] given the fact that each subset \(\mathcal{L}\) of \(\text{QDAG}\) satisfies \(\text{CO}\), then \(\mathcal{L}[\exists]\) and \(\mathcal{L}[\forall, \exists]\) satisfies \(\text{CO}\) as well, plus the fact that each of \(\text{KROM-}\text{C}\), \(\text{HORN-}\text{C}\), \(\text{renH-}\text{C}\) satisfies \(\text{CO}\).

- **VA:** Obviously, every subset \(\mathcal{L}\) of \(\text{QDAG}\) which satisfies \(\text{IM}\) satisfies \(\text{VA}\) as well (indeed, \(\alpha \in \mathcal{L}\) is valid if it is implied by the term \text{new} where \text{new} \in \text{PS} \setminus \text{Var}(\alpha)\). Furthermore, Lemma A.7 from [Darwiche and Marquis, 2002] can be straightforwardly extended to formulae \(\Sigma\), \(\alpha\) from \(\text{QDAG}\). As a consequence, every subset \(\mathcal{L}\) of \(\text{QDAG}\) satisfies \(\text{IM}\) whenever it satisfies \(\text{CD}\) and \(\text{VA}\).

- **CE, ME:** Direct from item 2. of Proposition 4 from [Fargier and Marquis, 2008a] given the fact that each subset of \(\text{QDAG}\) under consideration satisfies \(\text{CO}\) and \(\text{CD}\), and contains only proper formulae, i.e., those formulae \(\alpha\) from \(\text{QDAG}\) such that for every leaf node \(n\) of \(\alpha\) labeled by a literal of the form \(x \lor \neg x\) (with \(x \in \text{PS}\)), either there is a path from the root of \(\alpha\) to \(n\) which does not contain any node labeled by \(\exists x\) or there is a single path from the root of \(\alpha\) to \(n\) containing a node labeled by \(\exists x\).

- **IM:**
  - \(\text{renH-}\text{C}[\forall, \exists], K/H-\text{C}[\forall, \exists]\) and \(\text{HORN-}\text{C}[\forall, \exists]\): hold because none of these languages satisfies \(\text{VA}\) unless \(\mathcal{P} = \text{NP}\).
  - \(\text{renH-}\text{C}[\exists], K/H-\text{C}[\exists]\) and \(\text{HORN-}\text{C}[\exists]\): Since each of \(K/H-\text{C}[\exists]\) and \(\text{HORN-}\text{C}[\exists]\) is polynomially translatable into \(\text{renH-}\text{C}[\exists]\), it is enough to prove the result for \(\text{renH-}\text{C}[\exists]\). We first show that the implicant problem for \(\text{renH-}\text{C}[\exists]\) formulae can be reduced in polynomial time into the the implicant problem for \(\text{HORN-}\text{C}[\exists]\) formulae. Let \(\exists X.\alpha\) be a \(\text{renH-}\text{C}[\exists]\) formula such that \(\alpha\) is a \(\text{renH-}\text{C}\) formula, and let \(\gamma\) be a term. Let \(V\) be any Horn renaming for \(\alpha\). We have \(\gamma \models \exists X.\alpha\) iff \(\gamma \models (\exists X.\alpha)\) is valid. Now, the substitution metatheorem shows that if \(\gamma \models (\exists X.\alpha)\) is valid then \(V(\gamma) \models (\exists X.\alpha)\) is valid. Since for every formula \(\beta\), \(V(V(\beta)) \equiv \beta\), we also have that if \(V(\gamma) \models (\exists X.\alpha)\) is valid, then \(\gamma \models (\exists X.\alpha)\) is valid. Altogether, we get that \(\gamma \models \exists X.\alpha\) iff \(V(\gamma) \models (\exists X.\alpha)\) is valid. Now, \(V(\gamma) \models (\exists X.\alpha)\) is valid iff \(V(\gamma) \models (\exists X.\alpha)\) is valid iff \(V(\gamma) \models (\exists X.\alpha)\). Since for every \(x \in \text{Var}(\alpha)\), \(V(x)\) is a literal, we have \(V(\exists X.\alpha) \equiv \exists X.V(\alpha)\). Accordingly, \(\alpha\) is an implicant of the \(\text{renH-}\text{C}[\exists]\) formula \(\exists X.\alpha\) iff the term \(V(\gamma)\) is an implicant of the \(\text{HORN-}\text{C}[\exists]\) formula \(\exists X.V(\alpha)\). As explained above (see the \(\text{VA}\) point in the proof), the fact that \(\text{HORN-}\text{C}[\exists]\) satisfies \(\text{CD}\) and \(\text{VA}\) shows that it satisfies \(\text{IM}\) as well. Given that a Horn renaming \(V\) for \(\alpha\) can be computed in polynomial time given \(\alpha\), and that \(V(\gamma)\) (resp. \(V(\alpha)\)) can be computed in polynomial time from \(\gamma\) (resp. \(\alpha\)) once \(V\) has been computed, the fact that \(\text{HORN-}\text{C}[\exists]\) satisfies \(\text{IM}\) shows that \(\text{renH-}\text{C}[\exists]\) satisfies \(\text{IM}\) as well.

- **EQ:**
  - \(\text{renH-}\text{C}[\forall, \exists], K/H-\text{C}[\forall, \exists]\) and \(\text{HORN-}\text{C}[\forall, \exists]\): By reduction from the validity problem for those fragments. Let \(\alpha\) be any formula from one of those languages. \(\alpha\) is valid iff it is equivalent to a valid clause, like \(a \lor \neg a\) which belongs to each of the three languages under consideration here.

- \(\text{renH-}\text{C}[\exists], K/H-\text{C}[\exists]\) and \(\text{HORN-}\text{C}[\exists]\): For every propositional formulae \(\alpha'\) and \(\gamma'\) we have that \(\alpha' \equiv \gamma'\) iff \(\alpha' \land \gamma' \equiv \alpha'\). Consider now the formulae \(\alpha'\) and \(\gamma'\) used for proving that none of \(\text{renH-}\text{C}[\exists], K/H-\text{C}[\exists]\) and \(\text{HORN-}\text{C}[\exists]\) satisfies \(\text{SE}\) unless \(\mathcal{P} = \text{NP}\) (see the item \(\text{SE}\) in this proof). Since none of the \(\text{new}_i\) variables occurs in \(\alpha'\), the formula \(\alpha' \land \gamma'\) can be turned in linear time into the equivalent formula \(\exists \{\text{new}_1, \ldots, \text{new}_n\}\alpha' \land ((\neg \text{new}_1 \lor \ldots \lor \neg \text{new}_n) \land \bigwedge_{i=1}^n ((\text{new}_i \lor \neg \text{not} \lor x_i))\), which is a \(\text{HORN-}\text{C}[\exists]\) formula. This concludes the proof.

- **SE:**
  - \(\text{renH-}\text{C}[\forall, \exists], K/H-\text{C}[\forall, \exists]\) and \(\text{HORN-}\text{C}[\forall, \exists]\): By reduction from the validity problem for those fragments. Let \(\alpha\) be any formula from one of those languages. \(\alpha\) is valid iff it is entail by a valid clause, like \(a \lor \neg a\) which belongs to each of the three languages under consideration here.

- \(\text{renH-}\text{C}[\exists], K/H-\text{C}[\exists]\) and \(\text{HORN-}\text{C}[\exists]\): It is enough to prove the result for \(\text{HORN-}\text{C}[\exists]\) since this language is included in the two remaining ones. Let \(\alpha\) be a \(\text{CNF}\) formula over \(n\) variables \(x_1, \ldots, x_n\). Let \(\alpha'\) be the \(\text{HORN-}\text{C}\) formula obtained by replacing every positive literal \(x_i\) in \(\alpha\) by
the negative literal \(-not - x_i\) (where each \(not - x_i\) is a fresh variable), conjoined with \(n\) additional clauses \(\neg x_i \lor \neg not - x_i\) (\(i = 1, \ldots, n\)). Let \(\beta'\) be the \(\text{KROM-C}\) formula \(\bigwedge_{i=1}^{n} x_i \lor \neg not - x_i\). By construction, \(\alpha\) is consistent iff \(\alpha' \land \beta'\) is consistent iff \(\alpha' \neq \neg \beta'. \neg \beta'\) is equivalent to \(\bigvee_{i=1}^{n} \neg x_i \land \neg not - x_i\), which in turn is equivalent to the formula \(\gamma' = \exists \{\text{new}_1, \ldots, \text{new}_n\} ((\neg \text{new}_1 \lor \ldots \lor \neg \text{new}_n) \land \bigwedge_{i=1}^{n} ((\text{new}_i \lor \neg x_i) \land (\text{new}_i \lor \neg not - x_i))\) (where each \(\text{new}_i\) is a fresh variable). The fact that \(\alpha'\) and \(\gamma'\) are \(\text{HORN-C}[\exists]\) formulae which can be computed in time polynomial in the size of \(\alpha\) shows the \(\text{coNP}\)-hardness of \(\text{SE}\) for \(\text{HORN-C}[\exists]\) formulae and concludes the proof.

- **CT:** Direct from the fact that \(\text{KROM-C}\) does not satisfy \(\text{CT}\) [Roth, 1996], given that \(\text{KROM-C}\) is polynomially translatable into each of the fragments under consideration.

- **MC:** Direct from item 3. of Proposition 4 from [Fargier and Marquis, 2008a] showing that when a subset \(\mathcal{L}\) of \(\text{DAG}\) satisfies \(\text{CO}\) and \(\text{CD}\), then \(\mathcal{L}[\exists]\) and \(\mathcal{L}[\forall, \exists]\) satisfies \(\text{MC}\), plus the fact that each of \(\text{KROM-C}, \text{HORN-C}, \text{renH-C}\) satisfies \(\text{CO}\) and \(\text{CD}\).

As to the transformations:

- **CD:** Direct from item 1. of Proposition 4 from [Fargier and Marquis, 2008a] showing that when a subset \(\mathcal{L}\) of \(\text{DAG}\) satisfies \(\text{CD}\), then \(\mathcal{L}[\exists]\) and \(\mathcal{L}[\forall, \exists]\) satisfies \(\text{CD}\) as well, plus the fact that each of \(\text{KROM-C}, \text{HORN-C}, \text{renH-C}\) satisfies \(\text{CD}\).

- **FO:** Obvious since all the languages considered here are existential closures.

- **SFO:** Obvious since all the languages considered here satisfy \(\text{FO}\).

- **∧C:**
  - \(\text{renH-C}[\forall, \exists], \text{K/H-C}[\forall, \exists]:\) Direct from the fact that none of these three fragments satisfies \(\land\text{BC}\).
  - \(\text{HORN-C}[\forall, \exists]:\) Direct from the fact that each clause is a \(\text{HORN-C}[\forall, \exists]\) formula, plus the fact that \(\text{HORN-C}[\forall, \exists]\) satisfies \(\text{CO}\).
  - \(\text{K/H-C}[\exists], \text{renH-C}[\exists]:\) Direct from the fact that the restricted transformation \(\land\text{BC}\) is not always feasible within the fragment.

- **HORN-C[\exists]:** Let \(\exists x_1, \alpha_1, \ldots, \exists x_n, \alpha_n\) be \(n\) \(\text{HORN-C}\) formulae where each \(\alpha_i\) (\(i \in \{1, \ldots, n\}\)) is a \(\text{HORN-C}\) formula. For each \(i\) \(1, \ldots, n\), let \(\alpha_i'\) be the \(\text{HORN-C}\) formula obtained by replacing in \(\alpha_i\) every occurrence of \(x \in X_i\) by a fresh variable \(x'\), and let \(X_i'\) be the set of all the variables \(x'\) 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 \(\exists \bigcup_{i=1}^{n}\bigcup_{i=1}^{n} X_i' \land \bigwedge_{i=1}^{n} \alpha_i'\). Clearly enough, \(\exists \bigcup_{i=1}^{n}\bigcup_{i=1}^{n} X_i' \land \bigwedge_{i=1}^{n} \alpha_i'\) is a \(\text{HORN-C}[\exists]\) formula, and it can be generated in polynomial time from \(\exists x_1, \alpha_1, \ldots, \exists x_n, \alpha_n\).

- **∧BC:**
  - \(\text{renH-C}[\forall, \exists], \text{K/H-C}[\forall, \exists]:\) Let \(\alpha\) be a \(\text{CNF}\) formula over \(n\) variables \(x_1, \ldots, x_n\). Let \(\alpha'\) be the \(\text{HORN-C}\) formula obtained by replacing every positive literal \(x_i\) in \(\alpha\) by the negative literal \(\neg not - x_i\) (where each \(\neg not - x_i\) is a fresh variable), conjoined with \(n\) additional clauses \(\neg x_i \lor \neg not - x_i\) (\(i = 1, \ldots, n\)). Let \(\beta'\) be the \(\text{KROM-C}\) formula \(\bigwedge_{i=1}^{n} \neg x_i \lor \neg not - x_i\). Observe that \(\beta'\) is consistent, hence each of \(\alpha'\) and \(\beta'\) is a \(\text{renH-C}[\forall, \exists]\) formula and also a \(\text{K/H-C}[\exists]\) formula. By construction, \(\alpha\) is consistent iff \(\alpha' \land \beta'\) is consistent. If any of \(\text{renH-C}[\forall, \exists], \text{K/H-C}[\forall, \exists], \text{K/H-C}[\exists]\) would satisfy \(\land\text{BC}\), since each of these fragments satisfy \(\text{CO}\), we would have \(\text{P} = \text{NP}\).

- **HORN-C[\forall, \exists]:** Let \(\alpha\) and \(\beta\) be two \(\text{HORN-C}[\forall, \exists]\) formulae. From item 1. of Proposition 2 from [Fargier and Marquis, 2008a], one can compute in polynomial time a \(\text{HORN-C}[\forall, \exists]\) formula \(\exists x, \alpha'\) (resp. \(\exists Y, \beta'\)) equivalent to \(\alpha\) (resp. \(\beta\)) where \(\alpha'\) (resp. \(\beta'\)) is a \(\text{HORN-C}[\forall]\) formula. Let \(\alpha''\) (resp. \(\beta''\)) be the \(\text{HORN-C}[\forall]\) 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, \(\text{HORN-C}[\forall]\) satisfies \(\land\text{BC}\) (cf. Proposition 2 from [Fargier and Marquis, 2008b]). Hence, a \(\text{HORN-C}[\forall]\) 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 \(\text{HORN-C}[\forall, \exists]\) formula equivalent to \(\alpha \land \beta\), and it can be computed in time polynomial in the size of \(\alpha\) plus the size of \(\beta\).

- **K/H-C[\exists]:** \(a \lor b \land \neg a \lor \neg b \lor c\) are two \(K/H-C[\exists]\) formulae (the first one is a \(\text{KROM-C}\) formula but not a \(\text{HORN-C}\) one and the second one is a \(\text{HORN-C}\) formula but not a \(\text{KROM-C}\) one.) There does not exist a \(K/H-C[\forall]\) formula equivalent to their conjunction, since \((a \lor b) \land \neg (a \lor b \lor c)\) is neither equivalent to a \(\text{KROM-C}\) formula, nor equivalent to a \(\text{HORN-C}\) formula. The fact that \(K/H-C[\exists] \sim_{\text{e}} K/H-C\) concludes the proof.

- **renH-C[\exists]:** \(a \lor b \land \neg a \lor \neg b \lor c\) are two \(\text{renH-C}\) formulae, but there does not exist a \(\text{renH-C}\) formula equivalent to their conjunction, since \((a \lor b \lor c) \land \neg (a \lor b \lor c)\) is not equivalent to a Horn renamable formula and \(\text{renH-C}[\exists] \sim_{\text{e}} \text{renH-C}\).

- **HORN-C[\exists]:** Direct from the fact that \(\text{HORN-C}[\exists]\) satisfies \(\land\text{C}\.\)

- **∨C:**
  - \(\text{renH-C}[\forall, \exists], \text{K/H-C}[\forall, \exists]\) and \(\text{HORN-C}[\forall, \exists]\): Obvious since all the languages considered here are disjunction closures.
Proof: \[ \text{KROM-C} \] and \( \text{HORN-C} \): Direct from the fact that \( \forall \mathbf{BC} \) is not always feasible within the fragment.

\( \forall \mathbf{BC} \):

- \( \text{renH-C} \) and \( \text{HORN-C} \): Obviously since all languages considered here satisfy \( \forall \mathbf{C} \).

- \( \text{renH-C} \): Consider the two \( \text{renH-C} \) formulae \( \alpha = (a \vee b) \land (\neg a \vee c) \) and \( \beta = \neg a \land \neg c \). \( \alpha \lor \beta \) is equivalent to \( (a \vee b) \land (\neg a \lor \neg b \lor \neg \mathbf{C}) \), which is not equivalent to a \( \text{renH-C} \) formula, hence not to a \( \text{renH-C} [3] \) formula since \( \text{renH-C} \sim \mathbf{C} \).

- \( \text{K/H-C} \) and \( \text{HORN-C} \): \( \exists \alpha \lor b \land c \) are both \( \text{KROM-C} \) and \( \text{HORN-C} \) formulae but their disjunction \( \neg \alpha \lor b \lor c \) neither is equivalent to a \( \text{KROM-C} \) formula nor to a \( \text{HORN-C} \) formula since it is neither a \( \text{KROM-C} \) formula nor a \( \text{HORN-C} \) none and we have \( \text{KROM-C} \sim \mathbf{C} \) and \( \text{HORN-C} \sim \mathbf{C} \).

\( \neg \mathbf{C} \):

- \( \text{renH-C} \) and \( \text{HORN-C} \): \( \forall \mathbf{BC} \) is a subset of each of these fragments. Now, a \( \mathbf{DNF} \) formula \( \alpha \) is valid if \( \neg \alpha \) is inconsistent. The fact that each of \( \text{renH-C} \) and \( \text{HORN-C} \) satisfies \( \mathbf{CO} \) the proof.

- \( \text{renH-C} \) and \( \text{HORN-C} \): Consider the consistent \( \text{KROM-C} \) formula \( \alpha = (\neg a \lor b) \land (\neg \alpha \lor c) \land (a \lor \neg b) \land (b \lor \neg c) \). It belongs to each of the fragments above. \( \neg \alpha \) is equivalent to \( (a \lor b \lor c) \land (\neg a \lor \neg b \lor \neg \mathbf{C}) \), which is not equivalent to a \( \text{renH-C} \) formula, hence not a \( \text{renH-C} [3] \) formula since \( \text{renH-C} \sim \mathbf{C} \). As such, it is not equivalent to a \( \text{K/H-C} \) formula or a \( \text{HORN-C} \) formula.

Proof: \[ \text{Proposition 3} \] Our proof is based on a theorem due to Sipser [Sipser, 1983]. This theorem can be expressed as follows: consider any Boolean function \( \alpha_k \) over \( n \) variables, represented by a \( \mathbf{NNF} \) formula having the shape of a tree of depth \( k > 1 \) and such that all the leaves are labeled by variables occurring once in the formula, the \( i^{\text{th}} \) level \( (i \in 1, \ldots, k) \) 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 \( n > 1 \) and the outdegree of every other internal node is equal to \( n^2 \). Sipser showed that such an \( \alpha_k \) and its dual \( \alpha_k^* \) (whose \( \mathbf{NNF} \) representation can be obtained by replacing the \( \mathbf{NNF} \) representation of \( \alpha_k \) every \( \land \) or \( \lor \) vice-versa cannot be represented by a polynomial-sized circuit over \( \{\land, \lor, \land\} \) of depth at most \( k - 1 \). In this construction, the variables of \( \alpha_k^* \) or its dual can be negated or not. Obviously, by considering \( \neg \alpha_k \) and simplifying, a similar conclusion can be drawn for formulae \( \beta_k^* \) over \( n \) variables, having the shape of a regular tree of depth \( k > 1 \) and such that all the leaves are labeled by negated variables, the \( i^{\text{th}} \) level \( (i \in 1, \ldots, k - 1) \) from the bottom consists of nodes labeled by \( \lor \) (resp. \( \land \)) 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 \( n > 1 \) and the outdegree of every other internal node is equal to \( n^2 \).

- Consider the Boolean function \( \alpha_k^* \) over \( n^4 \) variables. By construction, it can be represented by a conjunction of \( n \) \( \mathbf{DNF} \) formulae \( \beta_1, \ldots, \beta_n \), where each \( \beta_i \) \((i \in 1, \ldots, n)\) consists of \( n^2 \) terms \( \gamma_{i,j} \) \((j \in 1, \ldots, n^2)\), and each term \( \gamma_{i,j} \) \((j \in 1, \ldots, n^2)\) consists of the conjunction of \( n \) negated variables \( \neg x_{i,j,k} \) \((k \in 1, \ldots, n)\) occurring only once in \( \alpha_k^* \). For each \( i \in 1, \ldots, n \), consider now the \( \text{HORN-C} \) formula \( h_i \) such that: 

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

\( h_i \) contains \( n^3 + 1 \) clauses of size at most \( n^2 \), hence \( \bigwedge_{i=1}^{n} h_i \) contains \( n^4 + n \) clauses of size at most \( n^2 \). Let \( Y = \bigcup_{i=1}^{n} \bigcup_{j=1}^{n^2} \{y_{i,j}\} \). By construction, the \( \text{HORN-C} [3] \) formula \( \exists Y. \bigwedge_{i=1}^{n} h_i \) can be gen-
erated in time polynomial in \(n\). By definition, this formula \(\exists Y. \bigwedge_{i=1}^n h_i\) also is a \(K/H-C[\exists]\) formula and a \(renH-C[\exists]\) formula. From Sipser theorem, \(\alpha_3^*\) has no polynomial-sized CNF representation and no polynomial-sized DNF representation. It remains to show that \(\alpha_3^*\) is equivalent to \(\exists Y. \bigwedge_{i=1}^n h_i\). First of all, since each \(y_{i,j}\) \((i \in \{1, \ldots, n\}, j \in \{1, \ldots, n^2\})\) does not occur in \(h_i'\) \((i' \in \{1, \ldots, n\})\) unless \(i' = i\), we have that \(\exists Y. \bigwedge_{i=1}^n h_i\) is equivalent to \(\bigwedge_{i=1}^n (\exists Y_{i,j} \{y_{i,j}\}. h_i)\). Finally, by construction, for each \(i \in \{1, \ldots, n\}\), \(\beta_i = \bigwedge_{j,k=1}^{n^2} \gamma_{i,j,k}\) is the IP representation of \(\exists \bigwedge_{j,k=1}^{n^2} \{y_{i,j,k}\}. h_i\), hence it is equivalent to it. The substitution theorem for propositional logic concludes the proof.

- The proof is similar to the one given in item 1., but lifted to a greater depth. Consider the Boolean function \(\alpha_4^*\) 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 \(\gamma_{i,j,k}\) \((j \in \{1, \ldots, n\})\) consists of the disjunction of \(n^2\) terms \(\delta_{i,j,k}\), 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 \(\alpha_4^*\). For each \(i \in \{1, \ldots, n\}\) and \(j \in \{1, \ldots, n\}\), consider the HORN-C formula \(h_{i,j}\) such that \(h_{i,j} =\)

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

\(h_{i,j}\) contains \(n^3 + 1\) clauses of size at most \(n^2\), hence the HORN-C formula \(\bigwedge_{i,j,k}^{n^2} h_{i,j}\) contains \(n^5 + n^2\) clauses of size at most \(n^2\). Let \(Y = \bigcup_{i=1}^{n^6} (\bigcup_{j,k=1}^{n^2} \{y_{i,j,k}\})\). By construction, the HORN-C\([\forall, \exists]\) formula \(\exists Y. (\bigwedge_{i,j,k}^{n^2} h_{i,j})\) can be generated in time polynomial in \(n\). By definition, this formula \(\exists Y. (\bigwedge_{i,j,k}^{n^2} h_{i,j})\) also is a \(K/H-C[\forall, \exists]\) formula and a \(renH-C[\forall, \exists]\) formula. From Sipser theorem, \(\alpha_4^*\) has no polynomial-sized \(\Delta^3\) representation. It remains to show that \(\alpha_4^*\) is equivalent to \(\exists Y. (\bigwedge_{i,j,k}^{n^2} h_{i,j})\). 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 \(h_{i,j,k}'\) \((i' \in \{1, \ldots, n\}, j \in \{1, \ldots, n\})\) unless \(i' = i\) and \(j' = j\), we have that \(\exists Y. (\bigwedge_{i,j,k}^{n^2} h_{i,j,k})\) is equivalent to \(\bigwedge_{i,j,k}^{n^2} \bigwedge_{i=1}^{n^2} (\bigcup_{k=1}^{n^2} \{y_{i,j,k}\}. h_{i,j})\). Finally, by construction, for each \(i \in \{1, \ldots, n\}\) and \(j \in \{1, \ldots, n\}\), \(\gamma_{i,j,k} = \bigwedge_{k=1}^{n^2} \delta_{i,j,k}\) is the IP representation of \(\exists \bigwedge_{j,k=1}^{n^2} \{y_{i,j,k}\}. h_{i,j}\), hence it is equivalent to it. The substitution theorem for propositional logic concludes the proof.

**Proof:**[Proposition 4] The result comes easily from the fact that \(d\)-DNF is not at least as succinct as DNF, unless the polynomial hierarchy collapses [Darwiche and Marquis, 2002], plus the fact that DNF is polynomially translatable into each of KROM-C\([\forall]\), HORN-C\([\forall]\), K/H-C\([\forall]\), renH-C\([\forall]\), AFF\([\forall]\), \(\PI^1[\forall]\) and \(OBDD\_c[\forall]\) (observe that a term is a KROM-C formula, a HORN-C formula, a K/H-C formula, a renH-C formula, an AFF formula, and a \(\PI^1[\forall]\) formula, and that it can be turned in linear time into an \(OBDD\_c[\forall]\ formula whatever the underlying ordering \(<\)), and finally that any existential clause of a propositional language \(L \subseteq Q\) is at least as succinct as \(L\).

**Proof:**[Proposition 5] Since \(KROM-C \subseteq K/H-C, HORN-C \subseteq renH-C, and KROM-C is polynomially translatable into \(\PI^1[\forall]\), and each of those languages is a subset of \(CNF\), it is enough to show that the family of circular bit shift functions \(CBS_m\) have polynomial-sized representations in KROM-C\([\forall]\), HORN-C\([\forall]\), and AFF\([\forall]\). Indeed, it has been proven that such functions do not have polynomial-sized SDNF representations, where SDNF is the union of \(DNNF\_t\) for all vreeses \(T\) [Pipatsrisawat and Darwiche, 2010a].

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

\[
\bigwedge_{b_0, \ldots, b_{m-1} \in \{0,1\}} \bigwedge_{j=0}^{m-1} x_j \Leftrightarrow y_j + \text{num}(b_0, \ldots, b_{m-1}) \mod 2^m,
\]

whose size is linear in the number of variables of \(CBS_m\). In this formula, \(i_j\) denotes the literal \(i\) when \(b_j = 0\) and the literal \(-i\) when \(b_j = 1\); \(\text{num}\) 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 \(y_0 \ldots y_{2m-1}\) must be (circularly) shifted, and \(CBS_m(x_0, \ldots, x_{2m-1}, y_0, \ldots, y_{2m-1}, i_0, \ldots, i_{m-1}) = 1\) exactly when the variables \(x_0, \ldots, x_{2m-1}\) and the shifted variables \(y_0, y_{2m-1}\) are pairwise equal.

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

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

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

\[
\bigwedge_{j=0}^{m-1} i_j \land \bigwedge_{j=0}^{m-1} (\neg x_j \lor y_j + \text{num}(b_0, \ldots, b_{m-1}) \mod 2^m) \land (x_j \lor \neg y_j + \text{num}(b_0, \ldots, b_{m-1}) \mod 2^m).
\]
Clearly enough, \( \gamma_{b_0, \ldots, b_{m-1}} \) also is a HORN-C formula. Similarly, \( \beta_{b_0, \ldots, b_{m-1}} \) is equivalent to the AFF formula

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

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 \( cbs_m \).

As a consequence, \( \bigvee_{b_0, \ldots, b_{m-1} \in \{0, 1\}} \gamma_{b_0, \ldots, b_{m-1}} \) is a KROM-C[V] (and a HORN-C[V]) formula equivalent to \( \alpha_m \), and \( \bigvee_{b_0, \ldots, b_{m-1} \in \{0, 1\}} \delta_{b_0, \ldots, b_{m-1}} \) is an AFF[V] 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 \( cbs_m \) completes the proof.

As to \( \ddot{d-} \text{DNNF} \), the result comes easily from the fact that \( \ddot{d-} \text{DNNF} \) is not at least as succinct as DNF, unless the polynomial hierarchy collapses [Darwiche and Marquis, 2002], plus the fact that DNF is polynomially translatable into the disjunction closure and into the full disjunctive closure of each of KROM-C, HORN-C, K/H-C, and renH-C. ■