Extending the Knowledge Compilation Map: Krom, Horn, Affine and Beyond

Hélène Fargier
IRIT-CNRS, Université Paul Sabatier,
Toulouse, France
fargier@irit.fr

Pierre Marquis
Université Lille-Nord de France, Artois
CRIL UMR CNRS 8188, Lens, France
marquis@cril.univ-artois.fr

Abstract

We extend the knowledge compilation map introduced by Darwiche and Marquis with three influential propositional fragments, the Krom CNF one (also known as the bijunctive fragment), the Horn CNF fragment and the affine fragment (also known as the biconditional fragment) as well as seven additional languages based on them, and composed respectively of Krom or Horn CNF formulas, renamable Horn CNF formulas, disjunctions of Krom CNF formulas, disjunctions of Horn CNF formulas, disjunctions of Krom or Horn CNF formulas, disjunctions of renamable Horn CNF formulas, and disjunction of affine formulas. Each fragment is evaluated w.r.t. several criteria, including the complexity of basic queries and transformations, and its spatial efficiency is also analyzed.

Introduction

Knowledge compilation (KC) is considered in many AI applications where short on-line response times are expected. It consists in turning (during an off-line phase) the initial data into a form that ensures the tractability of the requests and transformations of interest (see among others (Darwiche 2001; Cadoli & Donini 1998; Boufkhad *et al.* 1997; Selman & Kautz 1996; Schrag 1996; Gogic *et al.* 1995; Marquis 1995; del Val 1994; Dechter & Rish 1994)).

Many languages can be considered as target languages for knowledge compilation. In the propositional case, (Darwiche & Marquis 2002) investigate a dozen such languages, called propositional fragments. In this paper, the authors argue that the choice of a target language for a compilation purpose must be based both on the spatial efficiency of the language (i.e., its ability to represent data using little space) as well as its temporal efficiency, i.e., its ability to enable a set of queries and transformations to be achieved in polynomial time. The basic queries considered in (Darwiche & Marquis 2002) include tests for consistency, validity, implicates (clausal entailment), implicants, equivalence, sentential entailment, counting and enumerating theory models (**CO**, **VA**, **CE**, **EQ**, **SE**, **IM**, **CT**, **ME**). The basic transformations are conditioning (**CD**), closures under the connectives ($\wedge$, $\vee$, $\neg$) ($\wedge\mathbf{C}$, $\wedge\mathbf{BC}$, $\vee\mathbf{C}$, $\vee\mathbf{BC}$, $\neg\mathbf{C}$), and forgetting which can be viewed as a closure operation under existential quantification (**FO**, **SFO**).

The KC map and its extensions (Wachter & Haenni 2006; Fargier & Marquis 2006; Subbarayan, Bordeaux, & Hamadi 2007) have put forward the interest of fragments satisfying the property of decomposability, especially DNNF (Darwiche 2001). Indeed this fragment enables **CO**, **CE**, **CD** and **FO** in polynomial time, while being more succinct than DNF and OBDD. On the other hand, the property of decomposability is seldom compatible with a polynomial handling of the conjunctive transformations $\wedge\mathbf{C}$ or even $\wedge\mathbf{BC}$; thus DNNF does not enable $\wedge\mathbf{BC}$ in polytime unless P = NP.

In this paper, we focus on ten propositional fragments which have not been considered in the above-mentioned papers. We first consider the Krom CNF fragment KROM-C (also known as the bijunctive fragment), the Horn CNF fragment HORN-C, and the affine fragment AFF (also known as the biconditional fragment) (Schaefer 1978), as well as K/H-C (Krom or Horn CNF formulas) and renH-C, the class of renamable Horn CNF formulas. None of these first five fragments is fully expressive w.r.t. propositional logic (there exist propositional formulas which cannot be represented in any of them). But full expressiveness can be recovered by considering disjunctions of such formulas; we thus include in our investigation the following fragments, which are complete for propositional logic: KROM-C[$\vee$], HORN-C[$\vee$], K/H-C[$\vee$], renH-C[$\vee$], and AFF[$\vee$] are composed respectively of disjunctions of Krom CNF formulas, disjunctions of Horn CNF formulas, disjunctions of Krom or Horn CNF formulas, disjunctions of renamable Horn CNF formulas, and disjunctions of affine formulas. Interestingly, each of these classes enables **CE** in polynomial time, just like DNNF does; furthermore, it has been shown that, from the practical side, for some propositional formulas the size of renH-C[$\vee$] compilations can be much smaller than the size of DNF compilations (Boufkhad *et al.* 1997).

The contribution of this paper consists of an evaluation of KROM-C, HORN-C, K/H-C, renH-C, AFF, KROM-C[$\vee$], HORN-C[$\vee$], K/H-C[$\vee$], renH-C[$\vee$], and AFF[$\vee$] following the lines of (Darwiche & Marquis 2002). While (Boufkhad *et al.* 1997) considers only the clausal entailment issue, all the queries and transformations considered in (Darwiche & Marquis 2002) are investigated here for the ten fragments and the spatial efficiency of those fragments is also analyzed. Among other things, our results show that, when $\wedge\mathbf{BC}$ is expected, HORN-C[$\vee$], KROM-C[$\vee$] and AFF[$\vee$] are very interesting alternatives to DNNF, which does not satisfy it. The $\wedge\mathbf{BC}$ transformation is of the utmost value in a number of applications; for instance, it offers the opportunity of incrementally compiling devices for the diagnosis issue: when a device is composed of a small number of components, each component can be compiled separately, and connecting them amounts mainly to conjoin the corresponding compiled forms, which can be done efficiently when a compilation fragment satisfying $\wedge\mathbf{BC}$ is targeted.

The paper is organized as follows. After some formal preliminaries, we recall the languages, queries, transformations and the notion of succinctness considered in the KC map. We then present our results concerning the evaluation of the languages KROM-C, HORN-C, K/H-C, renH-C, AFF, KROM-C[$\vee$], HORN-C[$\vee$], K/H-C[$\vee$], renH-C[$\vee$] and AFF[$\vee$]. Those results are discussed just before the concluding section. For space reasons, proofs are omitted.

Formal Preliminaries

We assume the reader familiar with the basics of propositional logic, including the notion of satisfaction, entailment ($\models$) and equivalence ($\equiv$). All the propositional fragments we consider in this paper are subsets of the following propositional language DAG$_{PS}$:

Definition 1 *Let PS be a set of propositional variables (or atoms).* DAG$_{PS}$ *is the set of all finite, single-rooted DAGs where each leaf node is labeled by a literal over PS or one of the two Boolean constants $\top$ or $\bot$, and each internal node is labeled by $\wedge$, $\vee$ or $\oplus$ and has arbitrarily many children. The elements of* DAG$_{PS}$ *are called formulas.*

The fragment DAG-NNF$_{PS}$, considered in (Darwiche & Marquis 2002) is the set of all DAG$_{PS}$ formulas in which the XOR connective $\oplus$ does not occur. For any formula α, $Var(\alpha)$ denotes the set of atoms of PS occurring in α. The size $\mid \alpha \mid$ of α is the total number of vertices and arcs in its DAG representation.

Distinguished formulas are the literals over PS; for any subset V of PS, L_V denotes the set of all literals built over V, i.e., $\{x, \neg x \mid x \in PS\}$. If a literal l of L_{PS} is an atom x from PS, it is said to be positive; otherwise it has the form $\neg x$ with $x \in PS$ and it is said to be negative. If l is a positive literal x then its complementary literal $\bar{l}$ is the negative literal $\neg x$; if l is a negative literal $\neg x$ then its complementary literal $\bar{l}$ is the positive literal x. Other distinguished formulas are the clauses (resp. the terms) over PS; a clause (resp. a term) is a finite disjunction (resp. conjunction) of literals, or the Boolean constant $\bot$ (resp. $\top$). An XOR-clause is a finite exclusive disjunction of literals or Boolean constants.

The KC Map

Among the propositional fragments considered in (Darwiche & Marquis 2002) are OBDD, DNF, DNNF, CNF, PI, IP. Many fragments among them can be characterized by a number of properties, restricting the admissible formulas:

- **Flatness:** A DAG$_{PS}$ formula satisfies this property iff its height is at most 2.
- **Simple-disjunction:** A DAG$_{PS}$ formula satisfies this property iff the children of each or-node are leaves that share no variables (the node is a clause).
- **Simple-conjunction:** A DAG$_{PS}$ formula satisfies this property iff the children of each and-node are leaves that share no variables (the node is a term).
- **Decomposability:** A DAG$_{PS}$ formula satisfies this property iff for each conjunction C in the formula, the conjuncts of C do not share variables. That is, if $C_1, \ldots, C_n$ are the children of and-node C, then $Var(C_i) \cap Var(C_j) = \emptyset$ for $i \neq j$.
- **Decision:** A DAG$_{PS}$ formula satisfies this property iff its root is a decision node, where a decision node N in a DAG$_{PS}$ formula is one which is labeled with $\top$, $\bot$, or is an or-node having the form $(x \wedge \alpha) \vee (\neg x \wedge \beta)$, where x is a variable, α and β are decision nodes. In the latter case, $dVar(N)$ denotes the variable x.
- **Ordering:** Let $<$ be a total ordering on PS. A DAG$_{PS}$ formula satisfying **Decomposability** and **Decision** satisfies **Ordering** iff whenever N and M are or-nodes in it, if N is an ancestor of node M, then $dVar(N) < dVar(M)$.

Definition 2

- DNNF *is the subset of all* DAG-NNF$_{PS}$ *formulas satisfying* **Decomposability**.
- OBDD$_<$ *is the subset of all* DAG-NNF$_{PS}$ *formulas satisfying* **Decomposability**, **Decision** *and* **Ordering** *(for a given total ordering $<$ on PS).* OBDD *is the union of all* OBDD$_<$ *languages.*
- CNF *is the subset of all* DAG-NNF$_{PS}$ *formulas satisfying* **Flatness** *and* **Simple-disjunction**.
- DNF *is the subset of all* DAG-NNF$_{PS}$ *formulas satisfying* **Flatness** *and* **Simple-conjunction**.
- PI *is the subset of* CNF *in which each clause entailed by the formula is subsumed by a clause that appears in the formula; and no clause in the formula is subsumed by another.*
- IP *is the subset of* DNF *in which each term entailing the formula subsumes some term that appears in the formula; and no term in the formula is subsumed by another term.*

The following queries and transformations have been considered in (Darwiche & Marquis 2002); since their importance has been discussed in depth, we refrain from recalling it here.

Definition 3 *Let $\mathcal{C}$ denote any subset of* DAG$_{PS}$.

- *$\mathcal{C}$ satisfies* **CO** *(resp.* **VA***) iff there exists a polytime algorithm that maps every formula α from $\mathcal{C}$ to 1 if α is consistent (resp. valid), and to 0 otherwise.*
- *$\mathcal{C}$ satisfies* **CE** *iff there exists a polytime algorithm that maps every formula α from $\mathcal{C}$ and every clause γ to 1 if $\alpha \models \gamma$ holds, and to 0 otherwise.*
- *$\mathcal{C}$ satisfies* **EQ** *(resp.* **SE***) iff there exists a polytime algorithm that maps every pair of formulas α, β from $\mathcal{C}$ to 1 if $\alpha \equiv \beta$ (resp. $\alpha \models \beta$) holds, and to 0 otherwise.*

- $\mathcal{C}$ *satisfies* **IM** *iff there exists a polytime algorithm that maps every formula* α *from* $\mathcal{C}$ *and every term* γ *to* 1 *if* $\gamma \models \alpha$ *holds, and to* 0 *otherwise.*
- $\mathcal{C}$ *satisfies* **CT** *iff there exists a polytime algorithm that maps every formula* α *from* $\mathcal{C}$ *to a nonnegative integer that represents the number of models of* α *over* $Var(\alpha)$ *(in binary notation).*
- $\mathcal{C}$ *satisfies* **ME** *iff there exists a polynomial* $p(.,.)$ *and an algorithm that outputs all models of an arbitrary formula* α *from* $\mathcal{C}$ *in time* $p(n, m)$*, where* n *is the size of* α *and* m *is the number of its models (over* $Var(\alpha)$*).*

Definition 4 *Let* $\mathcal{C}$ *denote any subset of* $\texttt{DAG}_{PS}$*.*

- $\mathcal{C}$ *satisfies* **CD** *iff there exists a polytime algorithm that maps every formula* α *from* $\mathcal{C}$ *and every consistent term* γ *to a formula from* $\mathcal{C}$ *that is logically equivalent to the conditioning* $\alpha \mid \gamma$ *of* α *on* γ*, i.e., the formula obtained by replacing each variable* x *of* α *by* $\top$ *(resp.* $\bot$*) if* x *(resp.* $\neg x$*) is a positive (resp. negative) literal of* γ*.*
- $\mathcal{C}$ *satisfies* **FO** *iff there exists a polytime algorithm that maps every formula* α *from* $\mathcal{C}$ *and every subset* $\mathbf{X}$ *of variables from* PS *to a formula from* $\mathcal{C}$ *equivalent to* $\exists \mathbf{X}.\alpha$*.* $\mathcal{C}$ *satisfies* **SFO** *iff the property holds for any singleton.*
- $\mathcal{C}$ *satisfies* $\wedge$**C** *(resp.* $\vee$**C***) iff there exists a polytime algorithm that maps every finite set of formulas* $\alpha_1, \ldots, \alpha_n$ *from* $\mathcal{C}$ *to a formula of* $\mathcal{C}$ *that is logically equivalent to* $\alpha_1 \wedge \ldots \wedge \alpha_n$ *(resp.* $\alpha_1 \vee \ldots \vee \alpha_n$*).*
- $\mathcal{C}$ *satisfies* $\wedge$**BC** *(resp.* $\vee$**BC***) iff there exists a polytime algorithm that maps every pair of formulas* α *and* β *from* $\mathcal{C}$ *to a formula of* $\mathcal{C}$ *that is logically equivalent to* $\alpha \wedge \beta$ *(resp.* $\alpha \vee \beta$*).*
- $\mathcal{C}$ *satisfies* $\neg$**C** *iff there exists a polytime algorithm that maps every formula* α *from* $\mathcal{C}$ *to a formula of* $\mathcal{C}$ *that is logically equivalent to* $\neg\alpha$*.*

Finally, the following notion of succinctness has been considered in (Darwiche & Marquis 2002):

Definition 5 *Let* $\mathcal{C}_1$ *and* $\mathcal{C}_2$ *be two subsets of* $\texttt{DAG}_{PS}$ *which are complete for propositional logic.* $\mathcal{C}_1$ *is at least as succinct as* $\mathcal{C}_2$*, denoted* $\mathcal{C}_1 \leq_s \mathcal{C}_2$*, iff there exists a polynomial* p *such that for every formula* $\alpha \in \mathcal{C}_2$*, there exists an equivalent formula* $\beta \in \mathcal{C}_1$ *where* $|\beta| \leq p(|\alpha|)$*.*

$=_s$ is the symmetric part of $\leq_s$ defined by $\mathcal{C}_1 =_s \mathcal{C}_2$ iff $\mathcal{C}_1 \leq_s \mathcal{C}_2$ and $\mathcal{C}_2 \leq_s \mathcal{C}_1$. $<_s$ is the asymmetric part of $\leq_s$ defined by $\mathcal{C}_1 <_s \mathcal{C}_2$ iff $\mathcal{C}_1 \leq_s \mathcal{C}_2$ and $\mathcal{C}_2 \not\leq_s \mathcal{C}_1$. The succinctness preorder $\leq_s$ can be viewed as a refinement of the expressiveness preorder over all propositional fragments, given by $\mathcal{C}_1$ is at least as expressive as $\mathcal{C}_2$, denoted $\mathcal{C}_1 \leq_e \mathcal{C}_2$, iff for every formula $\alpha \in \mathcal{C}_2$, there exists an equivalent formula $\beta \in \mathcal{C}_1$. $<_e$ is the asymmetric part of $\leq_e$.

Extending the KC Map

Languages

This paper focuses on the following subsets of $\texttt{DAG}_{PS}$:

Definition 6

- *The language* `KROM-C` *is the subset of all* `CNF` *formulas in which each clause is binary, i.e., it contains at most two literals.*
- *The language* `HORN-C` *is the subset of all* `CNF` *formulas in which each clause contains at most one positive literal.*
- *The language* `K/H-C` *is the union of* `KROM-C` *and* `HORN-C`*.*
- *The language* `renH-C` *is the subset of all* `CNF` *formulas* α *for which there exists a subset* V *of* $Var(\alpha)$ *(called a Horn renaming for* α*) such that the formula obtained by substituting in* α *every literal* l *of* L_V *by its complementary literal* $\overline{l}$ *is a Horn CNF formula.*
- *The language* `AFF` *is the subset of* $\texttt{DAG}_{PS}$ *consisting of conjunctions of XOR clauses.*
- *The language* `KROM-C`$[\vee]$ *is the subset of* $\texttt{DAG-NNF}_{PS}$ *consisting of disjunctions of* `KROM-C` *formulas.*
- *The language* `HORN-C`$[\vee]$ *is the subset of* $\texttt{DAG-NNF}_{PS}$ *consisting of disjunctions of* `HORN-C` *formulas.*
- *The language* `K/H-C`$[\vee]$ *is the subset of* $\texttt{DAG-NNF}_{PS}$ *consisting of disjunctions of* `K/H-C` *formulas.*
- *The language* `renH-C`$[\vee]$ *is the subset of* $\texttt{DAG-NNF}_{PS}$ *consisting of disjunctions of* `renH-C` *formulas.*
- *The language* `AFF`$[\vee]$ *is the subset of* $\texttt{DAG}_{PS}$ *consisting of disjunctions of* `AFF` *formulas.*

Example 1

- $(a \vee b) \wedge (\neg b \vee c)$ *is a* `KROM-C` *formula.*
- $(\neg a \vee \neg b \vee c) \wedge (\neg b \vee c \vee \neg d)$ *is a* `HORN-C` *formula.*
- $(a \vee b \vee c) \wedge (\neg a \vee \neg b \vee c)$ *is a* `renH-C` *formula.*
- $((a \vee b) \wedge (\neg b \vee c)) \vee a \vee d$ *is a* `KROM-C`$[\vee]$ *formula.*
- $((\neg a \vee c) \wedge (\neg b \vee c \vee \neg d)) \vee (a \wedge b)$ *is a* `HORN-C`$[\vee]$ *formula.*
- $(a \oplus \neg b \oplus \top) \wedge (\neg a \oplus c)$ *is an* `AFF` *formula.*
- $((\neg a \vee c) \wedge (\neg b \vee c \vee \neg d)) \vee ((a \vee b) \wedge c)$ *is a* `K/H-C`$[\vee]$ *formula.*
- $((a \vee c) \wedge (b \vee c \vee d)) \vee a \vee b$ *is a* `renH-C`$[\vee]$ *formula.*
- $((a \oplus \neg b \oplus \top) \wedge (\neg a \oplus c)) \vee ((b \oplus d) \wedge \neg e)$ *is an* `AFF`$[\vee]$ *formula.*

Obviously enough, we have the following inclusions:

$$\texttt{HORN-C} \subseteq \texttt{K/H-C} \subseteq \texttt{renH-C}$$
$$\texttt{HORN-C} \subseteq \texttt{HORN-C}[\vee] \subseteq \texttt{K/H-C}[\vee]$$
$$\texttt{HORN-C}[\vee] \subseteq \texttt{renH-C}[\vee]$$
$$\texttt{KROM-C} \subseteq \texttt{K/H-C}$$
$$\texttt{KROM-C} \subseteq \texttt{KROM-C}[\vee] \subseteq \texttt{K/H-C}[\vee].$$

Note that there exists linear time algorithms for recognizing renamable Horn CNF formulas (see e.g. (Hébrard 1994; del Val 2000)); furthermore, such recognition algorithms typically give a Horn renaming when it exists. It is also known that every satisfiable `KROM-C` formula is a `renH-C` formula and that `KROM-C` satisfies **CO**; so, to every `K/H-C`$[\vee]$ formula we can associate in polynomial time an equivalent `renH-C`$[\vee]$ formula.

Expressiveness

It is well-known that none of the fragments `KROM-C`, `HORN-C`, `K/H-C`, `renH-C`, or `AFF` is complete for propositional logic. For instance, there is no formula from any of these fragments which is equivalent to the `CNF` formula $(a \vee b \vee c) \wedge (\neg a \vee \neg b \vee \neg c)$.

As to expressiveness, it is easy to see that we have

$$\texttt{renH-C} <_e \texttt{K/H-C} <_e \texttt{HORN-C}$$
$$\texttt{K/H-C} <_e \texttt{KROM-C}$$

while HORN-C and KROM-C are incomparable w.r.t. $\leq_e$, and AFF and any of the other four incomplete fragments are incomparable w.r.t. $\leq_e$.

Queries and Transformations

We have obtained the following results:

Proposition 1 *The results in Table 1 hold.*

$\mathcal{C}$	CO	VA	CE	IM	EQ	SE	CT	ME
AFF[∨]	√	○	√	○	○	○	○	√
renH-C[∨]	√	○	√	○	○	○	○	√
K/H-C[∨]	√	○	√	○	○	○	○	√
HORN-C[∨]	√	○	√	○	○	○	○	√
KROM-C[∨]	√	○	√	○	○	○	○	√
AFF	√	√	√	√	√	√	√	√
renH-C	√	√	√	√	√	√	○	√
K/H-C	√	√	√	√	√	√	○	√
HORN-C	√	√	√	√	√	√	○	√
KROM-C	√	√	√	√	√	√	○	√

Table 1: Subsets of the DAG_{PS} language and their corresponding polytime queries. √ means "satisfies" and ○ means "does not satisfy unless P = NP."

Most of the results given in Proposition 1 are well-known or easy. We mainly provide them for the sake of completeness. As to transformations, we have obtained the following results (which, like succinctness results, are typically less easy):

Proposition 2 *The results in Table 2 hold.*

$\mathcal{C}$	CD	FO	SFO	$\wedge$**C**	$\wedge$**BC**	$\vee$**C**	$\vee$**BC**	$\neg$**C**
AFF[∨]	√	√	√	○	√	√	√	○
renH-C[∨]	√	?	√	○	○	√	√	○
K/H-C[∨]	√	?	√	○	○	√	√	•
HORN-C[∨]	√	?	√	○	√	√	√	•
KROM-C[∨]	√	√	√	○	√	√	√	•
AFF	√	√	√	√	√	!	!	!
renH-C	√	•	√	!	!	!	!	!
K/H-C	√	•	√	○	○	!	!	!
HORN-C	√	•	√	√	√	!	!	!
KROM-C	√	√	√	√	√	!	!	!

Table 2: Subsets of the DAG_{PS} language and their corresponding polytime 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 fragment.

In the light of the results reported in Propositions 1 and 2, we can draw the following remarks:

- Focusing on the queries only, the fragments we have considered can be gathered into two classes: one contains all the incomplete fragments and the other one contains all complete fragments. Within a class, all fragments have the same tractable queries (among those considered here) except AFF in the class of incomplete fragments which satisfies also **MC**. The incomplete fragments satisfy more tractable queries than complete fragments (in some sense, this balances their loss w.r.t. expressiveness).
- Taking transformations into account renders the comparison more complex; except KROM-C and AFF which satisfy the same feasible and tractable transformations (but are incomparable w.r.t. expressiveness), the incomplete fragments exhibit pairwise distinct sets of feasible and tractable transformations, justifying further that each of them has its own interest.

  Notice that KROM-C[∨] and AFF[∨] satisfy **FO**, just like DNNF does (recall that OBDD does not). Each of AFF[∨], HORN-C[∨], and KROM-C[∨] satisfies $\wedge$**BC** while DNNF does not. renH-C[∨] and K/H-C[∨] loose this property; this seems to the price to be paid for the gain in succinctness offered by these fragments, compared to HORN-C[∨] and KROM-C[∨], as shown in the next section.

Succinctness

Because they are incomplete fragments, we do not put KROM-C, HORN-C, K/H-C, renH-C, or AFF into the succinctness picture. We split our results into two propositions (and two tables). In the first table, we compare the complete fragments we have considered w.r.t. spatial efficiency $\leq_s$.

Proposition 3 *The results in Table 3 hold.*

	AFF[∨]	renH-C[∨]	K/H-C[∨]	HORN-C[∨]	KROM-C[∨]
AFF[∨]	$=_s$	$\not\leq_s$	$\not\leq_s$	$\not\leq_s$	$\not\leq_s$
renH-C[∨]	$\not\leq_s$	$=_s$	$\leq_s$	$\leq_s$	$\leq_s$
K/H-C[∨]	$\not\leq_s$	$\not\leq_s$	$=_s$	$\leq_s$	$\leq_s$
HORN-C[∨]	$\not\leq_s$	$\not\leq_s$	$\not\leq_s$	$=_s$	$\not\leq_s$
KROM-C[∨]	$\not\leq_s$	$\not\leq_s$	$\not\leq_s$	$\not\leq_s$	$=_s$

Table 3: Succinctness of target compilation languages.

Proposition 3 shows in particular that

$$\texttt{renH-C}[\vee] <_s \texttt{K/H-C}[\vee] <_s \texttt{HORN-C}[\vee]$$
$$\texttt{K/H-C}[\vee] <_s \texttt{KROM-C}[\vee]^1$$

while HORN-C[∨] and KROM-C[∨] are incomparable w.r.t. $\leq_s$, and AFF[∨] is incomparable w.r.t. succinctness with any fragment among KROM-C[∨], HORN-C[∨], K/H-C[∨], and renH-C[∨].

A direct consequence of Proposition 3 is that it makes sense to consider each of HORN-C[∨], KROM-C[∨], AFF[∨] and renH-C[∨] as a target fragment for knowledge compilation; indeed, HORN-C[∨], KROM-C[∨], AFF[∨] are pairwise incomparable from the point of view of succintness. This tells, from the practical side, that each of them can lead to exponentially smaller compiled forms than the other ones, depending on the instance at hand. renH-C[∨] is shown strictly more succinct than HORN-C[∨] and KROM-C[∨], but, unless P = NP it does not satisfy $\wedge$**BC** while HORN-C[∨] and KROM-C[∨] do. Finally, renH-C[∨] seems to be a better choice than K/H-C[∨] in the sense that it is strictly more succinct than it while it has the same set of tractable queries and transformations.

In the second table, we compare w.r.t. $\leq_s$ the fragments KROM-C[∨], HORN-C[∨], K/H-C[∨], renH-C[∨]

[1] One can observe that this succinctness picture is similar to the expressiveness picture for the "corresponding" incomplete fragments AFF, renH-C, K/H-C, HORN-C, KROM-C.

and AFF[∨] with many of the complete fragments considered in (Darwiche & Marquis 2002).

Proposition 4 *The results in Table 4 hold.*

	AFF[∨]	renH-C[∨]	K/H-C[∨]	HORN-C[∨]	KROM-C[∨]
$\mathtt{DAG}_{PS}$	$\leq_s, \not\geq_s$	$\leq_s, \not\geq_s^*$	$\leq_s, \not\geq_s$	$\leq_s, \not\geq_s$	$\leq_s, \not\geq_s$
CNF	$\not\leq_s, \not\geq_s$	$\not\leq_s, \not\geq_s^*$	$\not\leq_s, \not\geq_s$	$\not\leq_s, \not\geq_s$	$\not\leq_s, \not\geq_s$
PI	$\not\leq_s, \not\geq_s$	$\not\leq_s$, **?**	$\not\leq_s, \not\geq_s$	$\not\leq_s, \not\geq_s$	$\not\leq_s, \not\geq_s$
DNNF	**?**, $\not\geq_s$	**?**, $\not\geq_s$	**?**, $\not\geq_s$	**?**, $\not\geq_s$	**?**, $\not\geq_s$
DNF	$\not\leq_s, \geq_s$	$\not\leq_s, \geq_s$	$\not\leq_s, \geq_s$	$\not\leq_s, \geq_s$	$\not\leq_s, \geq_s$
IP	$\not\leq_s, \geq_s$	$\not\leq_s, \geq_s$	$\not\leq_s, \geq_s$	$\not\leq_s, \geq_s$	$\not\leq_s, \geq_s$
OBDD	$\not\leq_s, \not\geq_s$	$\not\leq_s, \not\geq_s$	$\not\leq_s, \not\geq_s$	$\not\leq_s, \not\geq_s$	$\not\leq_s, \not\geq_s$

Table 4: Succinctness of target compilation languages. $*$ means that the result holds unless the polynomial hierarchy collapses.

The results given in Propositions 3 and 4 are synthesized on Figure 1, which can be interpreted as the Hasse diagram of the set of all fragments given in it, ordered w.r.t. strict succinctness (thus, edges stemming from transitivity of strict succinctness are not explicitly represented). This succinctness picture completes the one given in (Darwiche & Marquis 2002) with the results we obtained.

It appears that each of the five complete fragments we considered (namely AFF[∨], renH-C[∨], K/H-C[∨], HORN-C[∨], KROM-C[∨]) is strictly more succinct than DNF and incomparable w.r.t. $\leq_s$ to OBDD (and possibly to PI – this is known for sure for AFF[∨], K/H-C[∨], HORN-C[∨] and KROM-C[∨]). The main open question concerns the relationships between AFF[∨], renH-C[∨], K/H-C[∨], HORN-C[∨], KROM-C[∨] and DNNF w.r.t. succinctness. We do not know whether DNNF is at least as succinct as any of them. We conjecture that this is not the case.

Discussion

Our results show AFF[∨], renH-C[∨], K/H-C[∨], HORN-C[∨], KROM-C[∨] as interesting alternatives to many of the complete fragments considered in (Darwiche & Marquis 2002) for the knowledge compilation purpose. Indeed, in the light of the results reported in Propositions 1, 2, 3, and 4, the following conclusions can be drawn:

- AFF[∨] and KROM-C[∨] are strictly more succinct than DNF while satisfying the same tractable queries and the same tractable transformations; similarly, renH-C[∨], K/H-C[∨], and HORN-C[∨] are strictly more succinct than DNF[2] while satisfying the same tractable queries, and possibly the same tractable transformations (depending on **FO**). Thus, AFF[∨] and KROM-C[∨] prove better fragments than DNF in a perspective of knowledge compilation, while HORN-C[∨], K/H-C[∨], and renH-C[∨] are at least challenging alternatives to it.
- Each of AFF[∨], renH-C[∨], K/H-C[∨], HORN-C[∨], KROM-C[∨] is strictly more succinct than IP, but satisfies less tractable queries than it. But each of them challenges IP w.r.t. transformations: AFF[∨], HORN-C[∨], and KROM-C[∨] satisfy more tractable transformations than IP, the set of tractable transformations satisfied by renH-C[∨] or K/H-C[∨] and the set of tractable transformations satisfied by IP being incomparable w.r.t. $\subseteq$.
- PI and any of AFF[∨], K/H-C[∨], HORN-C[∨], KROM-C[∨] are incomparable w.r.t. succinctness. PI satisfies more tractable queries than any of them; however it satisfies less tractable transformations than AFF[∨] and KROM-C[∨], and possibly of renH-C[∨], K/H-C[∨], and HORN-C[∨] depending on **FO**.
- Similarly, OBDD and any of AFF[∨], renH-C[∨], K/H-C[∨], HORN-C[∨], KROM-C[∨] are incomparable w.r.t. succinctness, as well as w.r.t. their sets of tractable queries or transformations.
- AFF[∨], renH-C[∨], K/H-C[∨], HORN-C[∨], KROM-C[∨] and DNNF satisfy exactly the same set of tractable queries. DNNF satisfies more tractable transformations than renH-C[∨] or K/H-C[∨], but can be challenged by aff[∨], HORN-C[∨] and KROM-C[∨] when $\wedge\mathbf{BC}$ is required: the latter fragments satisfy this transformation while DNNF does not.

Finally, let us remark that, when they are expressive enough and no transformation is required by the application under consideration, the incomplete fragments renH-C and AFF prove as good choices for the compilation purpose: they are the most expressive fragments among the five incomplete ones (KROM-C, HORN-C, K/H-C, renH-C and AFF) and they offer many tractable queries among those considered in the KC map (all of them for AFF). When transformations are required, the performance of renH-C gets lower; AFF is still a valuable candidate, provided than neither $\neg\mathbf{C}$ nor $\vee\mathbf{C}$ (or $\vee\mathbf{BC}$) are required: all the other transformations (and all the queries) can be performed in polynomial time within this fragment. When it proves expressive enough, AFF may even challenge OBDD since it satisfies **FO** and $\wedge\mathbf{C}$ while OBDD does not (unless P = NP).

Conclusion

In this paper, we have extended the KC map introduced by Darwiche and Marquis with ten propositional fragments based on the Krom CNF one, the Horn CNF one, and the affine one. Each fragment has been evaluated w.r.t. several criteria, including the complexity of basic queries and transformations, and its spatial efficiency has also been analyzed. As discussed in the previous section, AFF[∨], renH-C[∨], HORN-C[∨], KROM-C[∨] prove valuable alternatives to many of the complete fragments considered in (Darwiche & Marquis 2002). AFF[∨], in particular, satisfies both **FO** and $\wedge\mathbf{BC}$, while being strictly more succinct than each of the fragments in (Darwiche & Marquis 2002) offering both transformations. When expressive enough, the incomplete fragments renH-C and AFF are also valuable as target languages for the compilation purpose.

This paper opens a number of perspectives. One of them consists in determining whether DNNF is incomparable or not with AFF[∨], renH-C[∨], K/H-C[∨], HORN-C[∨], KROM-C[∨] w.r.t. succinctness. Another perspective consists in extending further the KC map by applying the prin-

[2] This coheres with the experimental results reported in (Boufkhad *et al.* 1997), where it is shown that from the practical side, for some propositional formulas the size of renH-C[∨] compilations is much smaller than the size of DNF compilations.

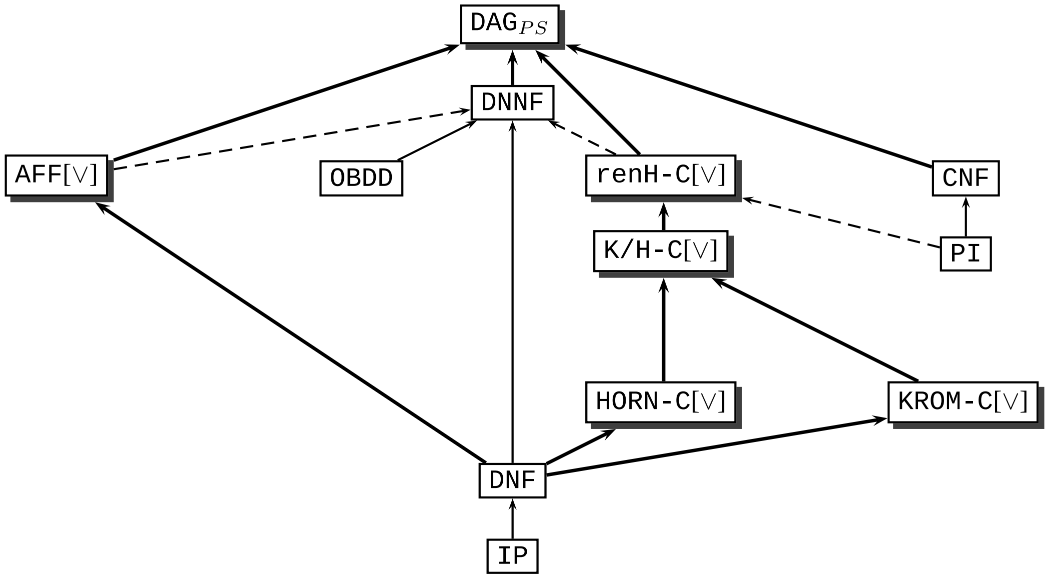

Figure 1: The succinctness picture. A solid edge from $\mathcal{C}_1$ to $\mathcal{C}_2$ indicates that $\mathcal{C}_1$ is strictly more succinct than $\mathcal{C}_2$ (edges stemming from transitivity of strict succinctness are not explicitly represented; some of the edges are conditioned on the polynomial hierarchy not collapsing). The new fragments are in shadowed boxes. Bold edges are new results. Dashed arrows indicate unknown relationships. The lack of edges, up to transitivity, between two fragments reflects the fact that they are incomparable.

ciple of $\vee$ closure at work here to other fragments than KROM-C, HORN-C, or AFF, leading to new fragments (e.g. OBDD[$\vee$] and PI[$\vee$]), and evaluating them. Another line for further research concerns the design of compilation algorithms targeting the AFF[$\vee$] fragment (interestingly, for the four other complete fragments such algorithms are already available (Boufkhad *et al.* 1997)).

Acknowledgements

The authors would like to thank the anonymous reviewers for their helpful comments. They have been partly supported by the ANR project PHAC (ANR-05-BLAN-0384). Pierre Marquis has also been partly supported by the IUT de Lens and the Région Nord/Pas-de-Calais.

References

Boufkhad, Y.; Grégoire, E.; Marquis, P.; Mazure, B.; and Saïs, L. 1997. Tractable cover compilations. In *Proceedings of IJCAI'97*, 122–127.

Cadoli, M., and Donini, F. 1998. A survey on knowledge compilation. *AI Communications* 10(3–4):137–150.

Darwiche, A., and Marquis, P. 2002. A knowledge compilation map. *Journal of Artificial Intelligence Research* 17:229–264.

Darwiche, A. 2001. Decomposable negation normal form. *Journal of the ACM* 48(4):608–647.

Dechter, R., and Rish, I. 1994. Directional resolution: the Davis-Putnam procedure, revisited. In *Proceedings of KR'94*, 134–145.

del Val, A. 1994. Tractable databases: how to make propositional unit resolution complete through compilation. In *Proceedings of KR'94*, 551–561.

del Val, A. 2000. On 2-SAT and Renamable Horn. In *Proceedings of AAAI'00*, 279–284.

Fargier, H., and Marquis, P. 2006. On the use of partially ordered decision graphs in knowledge compilation and quantifi ed Boolean formulae. In *Proceedings of AAAI'06*, 42–47.

Gogic, G.; Kautz, H.; Papadimitriou, C.; and Selman, B. 1995. The comparative linguistics of knowledge representation. In *Proceedings of IJCAI'95*, 862–869.

Hébrard, J. J. 1994. A Linear Algorithm for Renaming a Set of Clauses as a Horn Set. *Theoretical Computer Science* 124(2):343–350.

Marquis, P. 1995. Knowledge compilation using theory prime implicates. In *Proceedings of IJCAI'95*, 837–843.

Schaefer, T. J. 1978. The complexity of satisfi ability problems. In *Proceedings of STOC'78*, 216–226.

Schrag, R. 1996. Compilation for critically constrained knowledge bases. In *Proceedings of AAAI'96*, 510–515.

Selman, B., and Kautz, H. 1996. Knowledge compilation and theory approximation. *Journal of the ACM* 43:193–224.

Subbarayan, S.; Bordeaux, L.; and Hamadi, Y. 2007. Knowledge compilation properties of tree-of-BDDs. In *Proceedings of AAAI'07*, 502–507.

Wachter, M., and Haenni, R. 2006. Propositional DAGs: A new graph-based language for representing Boolean functions. In *Proceedings of KR'06*, 277–285.