Compilation of Propositional Weighted Bases

Adnan Darwiche
Computer Science Department
University of California, Los Angeles
CA 90095, USA
e-mail: darwiche@cs.ucla.edu

Pierre Marquis
CRIL-CNRS
Université d'Artois
F-62307, Lens Cedex, France
e-mail: marquis@cril.univ-artois.fr

Abstract

In this paper, we investigate the extent to which knowledge compilation can be used to improve inference from propositional weighted bases. We present a general notion of compilation of a weighted base that is parametrized by any equivalence–preserving compilation function. Both negative and positive results are presented. On the one hand, complexity results are identified, showing that the inference problem from a compiled weighted base is as difficult as in the general case, when the prime implicates, Horn cover or renamable Horn cover classes are targeted. On the other hand, we show that the inference problem becomes tractable whenever $DNNF$-compilations are used and clausal queries are considered. Moreover, we show that the set of all preferred models of a $DNNF$-compilation of a weighted base can be computed in time polynomial in the output size. Finally, we sketch how our results can be used in model-based diagnosis in order to compute the most probable diagnoses of a system.

1 INTRODUCTION

Penalty logic is a logical framework developed by Pinkas [19; 20] and by Dupin de St Cyr, Lang and Schiex [13], which enables the representation of propositional weighted bases. A *weighted base* is a finite set

$$W = \{\langle \phi_1, k_1 \rangle, \ldots, \langle \phi_n, k_n \rangle\}.$$

Each ϕ_i is a propositional formula, and k_i is its corresponding weight, i.e., the price to be paid if the formula is violated. In penalty logic, weights are positive integers[1] or $+\infty$ and they are *additively* aggregated.

[1] Floating numbers can also be used; what is important is the fact that sum is a total function over the set of (totally ordered) numbers under consideration, and that it can be computed in polynomial time.

A weighted base can be considered as a compact, implicit encoding of a total pre-ordering over a set Ω of propositional worlds. Indeed, given a weighted base W, the weight of each world ω can be defined as follows:

$$K_W(\omega) \stackrel{def}{=} \sum_{\langle \phi_i, k_i \rangle \in W,\ \omega \models \neg \phi_i} k_i.$$

That is, the weight of a world is the sum of all weights associated with sentences violated by the world. One can extend the function K_W to arbitrary sentences α:

$$K_W(\alpha) \stackrel{def}{=} \min_{\omega \models \alpha} K_W(\omega).$$

Finally, $min_W(\Omega)$ denotes the most preferred worlds in Ω, those having minimal weight:

$$min_W(\Omega) \stackrel{def}{=} \{\omega \mid \omega \in \Omega, \forall \omega' \in \Omega\, K_W(\omega) \leq K_W(\omega')\}.$$

The weight of base W, denoted $K(W)$, is the weight of some world in $min_W(\Omega)$. Obviously enough, we have $K(W) = K_W(true)$, and $\omega \in min_W(\Omega)$ if and only if $K_W(\omega) = K(W)$.

Example 1.1 *Let $W = \{\langle a \wedge b, 2 \rangle, \langle \neg b, 1 \rangle\}$ be a weighted base. Let us consider the following four worlds over the variables appearing in W, $Var(W)$:*

- $\omega_1 = (a, b)$
- $\omega_2 = (a, \neg b)$
- $\omega_3 = (\neg a, b)$
- $\omega_4 = (\neg a, \neg b)$

We then have $K_W(\omega_1) = 1$, $K_W(\omega_2) = 2$, $K_W(\omega_3) = 3$, and $K_W(\omega_4) = 2$. Accordingly, we have $K(W) = 1$ and $min_W(\Omega) = \{\omega_1\}$.

All formulas ϕ_i associated with finite weights in a weighted base are called *soft constraints*, while those associated with the weight $+\infty$ are called *hard constraints*.

Penalty logic has some valuable connections with possibilistic logic, as well as with Dempster–Shafer theory (see [13] for details). It is also closely connected to the optimization problem WEIGHTED-MAX-SAT considered in operations research. Accordingly, several proposals for the use of weighted bases can be found in the AI literature.

One of them concerns the compact representation of preferences in a decision making setting. Indeed, in some decision making problems, models (and formulas) can be used to encode decisions. Accordingly, the weight of a model represents the disutility of a decision, and a weighted base can be viewed as an implicit representation of the set of all decisions of an agent, totally ordered w.r.t. their (dis)utility. Lafage and Lang [16] take advantage of such an encoding for group decision making. A key issue here from a computational point of view is the problem consisting in computing (one or all) element(s) from $min_W(\Omega)$.

Another suggested use of penalty logic concerns inference from inconsistent belief bases. Based on the preference information given by K_W, several inference relations from a weighted base W can be defined. Among them is skeptical inference given by $\alpha \mid\!\sim_W \beta$ if and only if every world ω that is of minimal weight among the models of α is a model of β. In this framework, propositional formulas represent pieces of (explicit) belief. The inference relation $\mid\!\sim_W$ is interesting for at least two reasons. On the one hand, it is a *comparative* inference relation, i.e., a rational inference relation satisfying supraclassicality [13]. On the other hand, weighted bases can be used to encode some well-known forms of inference from stratified belief bases $B = (B_1, \ldots, B_k)$ [21; 1; 2]. Especially, the so-called skeptical *lexicographic inference* $B \mid\!\sim_{lex}$ can be recovered as a specific case of $true \mid\!\sim_{W_B}$ for some weighted base W_B.

Example 1.2 *Let $B = (B_1, B_2)$ be a belief base interpreted under lexicographic inference, where $B_1 = \{a \vee b \vee c\}$ (the most reliable stratum) and $B_2 = \{\neg a \wedge c, \neg b \wedge c, \neg c\}$. W can associate with B the weighted base*

$$W_B = \{\langle a \vee b \vee c, 4\rangle, \langle \neg a \wedge c, 1\rangle, \langle \neg b \wedge c, 1\rangle, \langle \neg c, 1\rangle\}.$$

The unique most preferred world for W_B is $(\neg a, \neg b, c)$ that is also the only lexicographically-preferred model of B.

Weighted bases enable more flexibility than stratified belief bases (e.g., violating two formulas of weight 5 is worse than violating a single formula of weight 9, but this cannot be achieved through a simple stratification)[2].

[2]Since lexicographic inference also includes inference from consistent sub–bases that are maximal w.r.t. cardinality as a subcase (to achieve it, just put every formula of the belief base into a single stratum), the latter can also be recovered as a specific case of inference from a weighted base.

The *inference* problem from a weighted base W consists in determining whether $true \mid\!\sim_W \beta$ holds given W and β. Up to now, weighted bases have been investigated from a theoretical point of view, only. Despite their potentialities, we are not aware of any industrial application of weighted bases. There is a simple (but partial) explanation of this fact: inference (and preferred model enumeration) from weighted bases are intractable. Actually, the inference problem is known as Δ_2^p-complete [12] (even in the restricted case where queries are literals). Furthermore, it is not hard to show that computing a preferred world from $min_W(\Omega)$ is $\mathsf{F}\Delta_2^p$-complete. This implies that any of the two problems is very likely to require an unbounded polynomial number of calls to an NP oracle to be solved in polynomial time on a deterministic Turing machine.

In this paper, we investigate the extent to which *knowledge compilation* [4] can be used to improve inference from weighted bases. The key idea of compilation is pre-processing of the fixed part of the inference problem. Several knowledge compilation functions dedicated to the clausal entailment problem have been pointed out so far (e.g., [23; 15; 14; 17; 6; 24; 25; 3; 8]). The input formula is turned into a compiled one during an off-line compilation phase and the compiled form is used to answer the queries on-line. Assuming that the formula does not often change and that answering queries from the compiled form is computationally easier than answering them from the input formula, the compilation time can be balanced over a sufficient number of queries. Thus, when queries are CNF formulas, the complexity of classical inference falls from coNP-complete to P. While none of the techniques listed above can ensure the objective of enhancing inference to be reached in the worst case (because the size of the compiled form can be exponentially larger than the size of the original knowledge base – see [25; 4]), experiments have shown such approaches valuable in many practical situations [24; 3; 10].

In the following, we show how compilation functions for clausal entailment from classical formulas can be extended to clausal inference from weighted bases. Any equivalence–preserving knowledge compilation function can be considered in our framework. Unfortunately, for many target classes for such functions, including the prime implicates, Horn cover and renamable Horn cover classes, we show that the inference problem from a compiled base remains Δ_2^p-complete, even for very simple queries (literals). Accordingly, in this situation, there is no guarantee that compiling a weighted base using any of the corresponding compilation functions may help. Then we focus on $DNNF$-compilations as introduced in [8; 9]. This case is much more favourable since the clausal inference problem becomes tractable. We also show that the preferred models of a $DNNF$-compilation of a weighted base can be enumerated in output polynomial time. Finally, we sketch how our results can be used in the model–based diagnosis framework in order to compute the most probable diagnoses of a system.

2 FORMAL PRELIMINARIES

In the following, we consider a propositional language $PROP_{PS}$ defined inductively from a finite set PS of propositional symbols, the boolean constants $true$ and $false$ and the connectives $\neg$, $\wedge$, $\vee$ in the usual way. L_{PS} is the set of literals built up from PS. For every formula ϕ from $PROP_{PS}$, $Var(\phi)$ denotes the symbols of PS occurring in ϕ. As mentioned before, if $W = \{\langle\phi_1, k_1\rangle, \ldots, \langle\phi_n, k_n\rangle\}$ is a weighted base, then $Var(W) = \bigcup_{i=1}^{n} Var(\phi_i)$.

Formulas are interpreted in a classical way. As evoked before, Ω $(= 2^{PS})$ denotes the set of all interpretations built up from PS. Every interpretation (world) $\omega \in \Omega$ is represented as a tuple of literals. $Mod(\phi)$ is the set of all models of ϕ.

As usual, every finite set of formulas is considered as the conjunctive formula whose conjuncts are the elements of the set. A CNF formula is a (finite) conjunction of clauses, where a clause is a (finite) disjunction of literals. A formula ϕ is Blake if and only if it is a CNF formula where each prime implicate[3] of ϕ appears as a conjunct (one representative per equivalence class). A formula is Horn CNF if and only if it is a CNF formula s.t. every clause in it contains at most one positive literal. A formula ϕ is renamable Horn CNF if and only if $\sigma(\phi)$ is a Horn CNF formula, where σ is a substitution from L_{PS} to L_{PS} s.t. $\sigma(l) = l$ for every literal l of L_{PS} except those of a set L, and for every literal l of L, $\sigma(l) = \neg l$ and $\sigma(\neg l) = l$.

We assume the reader familiar with the complexity classes P, NP, coNP and Δ_2^p of the polynomial hierarchy. $\mathsf{F}\Delta_2^p$ denotes the class of function problems associated to Δ_2^p (see [18] for details).

3 COMPILING WEIGHTED BASES

In this section, we first show how knowledge compilation techniques for improving clausal entailment can be used in order to compile weighted bases. Then, we present some complexity results showing that compiling a weighted base is not always a good idea, since the complexity of inference from a compiled base does not necessarily decrease. We specifically focus on prime implicates [23] and Horn covers and renamable Horn covers compilations [3].

3.1 A FRAMEWORK FOR WEIGHTED BASES COMPILATION

Let $W = \{\langle\phi_1, k_1\rangle, \ldots, \langle\phi_n, k_n\rangle\}$ be a weighted base. In the case where $\bigwedge_{i=1}^{n} \phi_i$ is consistent, then $K(W) = 0$ and $min_W(\Omega)$ is the set of all models of $\bigwedge_{i=1}^{n} \phi_i$. Accordingly, in this situation, inference $\mid\!\sim_W$ is classical entailment, so it is possible to directly use any knowledge compilation function and compiling W comes down to compile $\bigwedge_{i=1}^{n} \phi_i$. However, this situation is very specific and out of the ordinary when weighted bases are considered (otherwise, weights would be useless). A difficulty is that, in the situation where $\bigwedge_{i=1}^{n} \phi_i$ is inconsistent, we cannot compile directly this formula using any equivalence–preserving knowledge compilation function (otherwise, trivialization would not be avoided). Indeed, in this situation, $\mid\!\sim_W$ is not classical entailment any longer, so a more sophisticated approach is needed.

In order to compile weighted bases, it is helpful to consider weighted bases in normal form:

Definition 3.1 (Weighted bases in normal form)
A belief base $W = \{\langle\phi_1, k_1\rangle, \ldots, \langle\phi_n, k_n\rangle\}$ is in normal form *if and only if for every $i \in 1 \ldots n$, either $k_i = +\infty$ or ϕ_i is a propositional symbol.*

Every weighted base can be turned into a query–equivalent base in normal form.

Definition 3.2 (V-equivalence of weighted bases)
Let W_1 and W_2 be two weighted bases and let $V \subseteq PS$. W_1 and W_2 are V-equivalent *if and only if for every pair of sentences α and β in $PROP_V$, we have $\alpha \mid\!\sim_{W_1} \beta$ precisely when $\alpha \mid\!\sim_{W_2} \beta$.*

Accordingly, two V–equivalent weighted bases must agree on queries built up from the symbols in V. Note that a stronger notion of equivalence can be defined by requiring that both bases induce the same weight function, i.e., $K_{W_1} = K_{W_2}$ [13]. Finally, note that if K_{W_1} and K_{W_2} agree on the sentences in $PROP_V$, then W_1 and W_2 must be V–equivalent.

Proposition 3.1
Let $W = \{\langle\phi_1, k_1\rangle, \ldots, \langle\phi_n, k_n\rangle\}$ be a weighted base. If

$$
\begin{aligned}
H &\stackrel{def}{=} \{\langle\phi_i, +\infty\rangle \mid \langle\phi_i, +\infty\rangle \in W\}, \\
S &\stackrel{def}{=} \{\langle holds_i \Rightarrow \phi_i, +\infty\rangle, \langle holds_i, k_i\rangle \mid \\
&\qquad \langle\phi_i, k_i\rangle \in W and\ k_i \neq +\infty\ \},
\end{aligned}
$$

[3] A prime implicate of a formula ϕ is a logically strongest clause entailed by ϕ.

where $\{holds_1, \ldots, holds_n\} \subseteq PS \setminus Var(W)$, then the weighted base $W\!\downarrow \stackrel{def}{=} H \cup S$ is in normal form. Moreover, K_W and $K_{W\downarrow}$ agree on all weights of sentences in $PROP_{Var(W)}$ and, hence, $W\!\downarrow$ is $Var(W)$–equivalent to W.

We will call $W\!\downarrow$ the normal form of W. Intuitively, the variable $holds_i$ is guaranteed to be false in any world that violates the sentence ϕ_i and, hence, that world is guaranteed to incur the penalty k_i.

Example 1.1 (Continued) The weighted base

$$W\!\downarrow = \{\langle holds_1 \Rightarrow (a \wedge b), +\infty\rangle,$$
$$\langle holds_2 \Rightarrow \neg b, +\infty\rangle, \langle holds_1, 2\rangle, \langle holds_2, 1\rangle\}$$

is a normal form of W as given by Proposition 3.1. The normalized weighted base $W\!\downarrow$ induces the following weight function:

World	$K_{W\downarrow}$
$a, b, holds_1, holds_2$	$+\infty$
$a, b, holds_1, \neg holds_2$	1
$a, b, \neg holds_1, holds_2$	$+\infty$
$a, b, \neg holds_1, \neg holds_2$	3
$a, \neg b, holds_1, holds_2$	$+\infty$
$a, \neg b, holds_1, \neg holds_2$	$+\infty$
$a, \neg b, \neg holds_1, holds_2$	2
$a, \neg b, \neg holds_1, \neg holds_2$	3
$\neg a, b, holds_1, holds_2$	$+\infty$
$\neg a, b, holds_1, \neg holds_2$	$+\infty$
$\neg a, b, \neg holds_1, holds_2$	$+\infty$
$\neg a, b, \neg holds_1, \neg holds_2$	3
$\neg a, \neg b, holds_1, holds_2$	$+\infty$
$\neg a, \neg b, holds_1, \neg holds_2$	$+\infty$
$\neg a, \neg b, \neg holds_1, holds_2$	2
$\neg a, \neg b, \neg holds_1, \neg holds_2$	3

We have $K(W\!\downarrow) = 1$ and

$$min_{W\downarrow}(\Omega) = \{(a, b, holds_1, \neg holds_2)\}.$$

Moreover, K_W and $K_{W\downarrow}$ agree on all sentences constructed from the variables in $\{a, b\}$.

We now discuss the compilation of a weighted base in normal form. The basic idea is to combine all hard constraints in the base into a single constraint, which preserves the weight function induced by the base. We then compile that single hard constraint using an equivalence–preserving compilation function $COMP$, that is, a function which maps each sentence α into its compiled form $COMP(\alpha)$. From here on, we will use $\widehat{W}$ to denote the conjunction of all sentences in the weighted base W that have $+\infty$ weights:

$$\widehat{W} \stackrel{def}{=} \bigwedge_{\langle \phi_i, +\infty\rangle \in W} \phi_i.$$

Definition 3.3 (Compilation of a weighted base) *Let $W = \{\langle \phi_1, k_1\rangle, \ldots, \langle \phi_n, k_n\rangle\}$ be a weighted base. Let $COMP$ be any equivalence–preserving knowledge compilation function. The $COMP$-compilation of W is the weighted base*

$$W\!\downarrow_{COMP} \stackrel{def}{=} \{\langle COMP(\widehat{W\!\downarrow}), +\infty\rangle\} \cup$$
$$\{\langle holds_i, k_i\rangle \mid \langle holds_i, k_i\rangle \in W\!\downarrow \textit{ and } k_i \neq +\infty\}.$$

That is, to compile a weighted base W, we perform three steps. First, we compute a normal form $W\!\downarrow$ according to Proposition 3.1, which is guaranteed to be V-equivalent to W, where V are the variables in W. Next, we combine all of the hard constraints of $W\!\downarrow$ into a single hard constraint $\widehat{W\!\downarrow}$. Finally, we compile $\widehat{W\!\downarrow}$ using the function $COMP$.

Example 1.1 (Continued) We have

$$\widehat{W\!\downarrow} = (\neg holds_1 \vee (a \wedge b)) \wedge (\neg holds_2 \vee \neg b).$$

Accordingly, the Blake-compilation of W is

$$\{\langle(\neg holds_1 \vee a) \wedge (\neg holds_1 \vee b) \wedge (\neg holds_2 \vee \neg b)$$
$$\wedge(\neg holds_1 \vee \neg holds_2), +\infty\rangle, \langle holds_1, 2\rangle, \langle holds_2, 1\rangle\}.$$

Given Proposition 3.1, and since $COMP$ is equivalence–preserving, we have:

Corollary 3.1 *Let $W = \{\langle \phi_1, k_1\rangle, \ldots, \langle \phi_n, k_n\rangle\}$ be a weighted base. Let $COMP$ be any equivalence–preserving knowledge compilation function. $K_{W\downarrow_{COMP}}$ and K_W agree on the sentences in $PROP_{Var(W)}$. Moreover, W and $W\!\downarrow_{COMP}$ are $Var(W)$-equivalent.*

3.2 SOME COMPLEXITY RESULTS

In the following, the next tractable classes of formulas, that are target classes for some existing equivalence–preserving compilation functions $COMP$, are considered:

- The *Blake* class is the set of formulas given in prime implicates normal form,
- The *Horn cover* class is the set of disjunctions of Horn CNF formulas,
- The *renamable Horn cover* class (r. Horn cover for short) is the set of the disjunctions of renamable Horn CNF formulas.

The Blake class is the target class of the compilation function $COMP_{Blake}$ described in [23]. The Horn cover class and the renamable Horn cover class are target classes for the tractable covers compilation functions given in

[3]. We shall note respectively $COMP_{Horn\ cover}$ and $COMP_{r.\ Horn\ cover}$ the corresponding compilation functions.

Accordingly, a Blake (resp. Horn cover, r. Horn cover) compiled weighted base W is defined as a weighted base in normal form whose unique hard constraint belongs to the Blake (resp. Horn cover, r. Horn cover) class.

In the next section, we will also focus on the *DNNF* class. We consider it separately because — unlike the other classes — it will lead to render tractable clausal inference from compilations.

Of course, all these compilation functions $COMP$ are subject to the limitation evoked above: in the worst case, the size of the compiled form $COMP(\Sigma)$ is exponential in the size of Σ. Nevertheless, there is some empirical evidence that some of these approaches can prove computationally valuable for many instances of the clausal entailment problem (see e.g., the experimental results given in [24; 3; 10]).

As evoked previously, knowledge compilation can prove helpful only if inference from the compiled form is computationally easier than direct inference. Accordingly, it is important to identify the complexity of inference from a compiled weighted base if we want to draw some conclusions about the usefulness of knowledge compilation in this context. Formally, we are going to consider the following decision problems:

Definition 3.4 (FORMULA $\mid\!\sim_W$)
FORMULA $\mid\!\sim_W$ *is the following decision problem:*

- **Input:** *A weighted base W and a formula β from $PROP_{PS}$.*
- **Query:** *Does $true \mid\!\sim_W \beta$ hold?*

CLAUSE $\mid\!\sim_W$ (resp. LITERAL $\mid\!\sim_W$) is the restriction of FORMULA $\mid\!\sim_W$ to the case where β is required to be a CNF formula (resp. a term).

When no restriction is put on W, FORMULA $\mid\!\sim_W$ is known as Δ_2^p-complete [12], even in the restricted LITERAL $\mid\!\sim_W$ case. Now, what if W is a *compiled* weighted base? We have identified the following results:

Proposition 3.2 (Inference from compiled weighted bases) *The complexity of* CLAUSE $\mid\!\sim_W$ *and of its restrictions to literal inference when W is a $Blake$ (resp. $Horn\ cover$, $r.\ Horn\ cover$) compiled weighted base is reported in Table 1.*

Hardness results can be easily derived from results given in [5] due to the fact that (skeptical) lexicographic inference

$COMP$	CLAUSE / LITERAL $\mid\!\sim_W$
$Blake$	Δ_2^p-complete
$Horn\ cover$	Δ_2^p-complete
$r\ Horn\ cover$	Δ_2^p-complete

Table 1: Complexity of clausal inference from compiled weighted bases.

$\mid\!\sim_{lex}$ from a stratified belief base can be easily encoded as inference from a weighted base. Indeed, if m is the maximum number of formulas belonging to any stratum B_i of $B = (B_1, \ldots, B_k)$, then let $W_B = \{\langle \phi, (m+1)^{k-i} \rangle \mid \phi \in B_i\}$ (see Example 1.2 for an illustration). It is not hard to prove that $B \mid\!\sim_{lex} \beta$ if and only if $true \mid\!\sim_{W_B} \beta$.

The complexity results reported in Table 1 do not give good news: there is no guarantee that compiling a belief base using the $Blake$ (or the $Horn\ cover$ or the $r.\ Horn\ cover$) compilation function leads to improve inference since its complexity from the corresponding compiled bases is just as hard as the complexity of $\mid\!\sim_W$ in the general case.

Fortunately, it is not the case that such negative results hold for any compilation function. As we will see in the next section, $DNNF$-compilations of weighted bases exhibit a much better behaviour.

4 COMPILING WEIGHTED BASES USING DNNF

In this section, we focus on $DNNF$-compilations of weighted bases. After a brief recall of what $DNNF$-compilation is, we show that $DNNF$-compilations support two important computational tasks in polynomial time, especially preferred model enumeration and clausal inference.

4.1 A GLIMPSE AT THE $DNNF$ LANGUAGE

The $DNNF$ language is the set of sentences, defined as follows [8]:

Definition 4.1 (DNNF) *Let PS be a finite set of propositional variables. A sentence in $DNNF$ is a rooted, directed acyclic graph (DAG) where each leaf node is labeled with $true$, $false$, x or $\neg x$, $x \in PS$; each internal node is labeled with $\wedge$ or $\vee$ and can have arbitrarily many children. Moreover, the* decomposability *property is satisfied: for each conjunction C in the sentence, the conjuncts of C do not share variables.*

Figure 1 depicts a $DNNF$ of the hard constraint $\widehat{W\downarrow}$, where W is the weighted base given in Example 1.1. Note here that $W\downarrow$ is the normal form constructed from W ac-

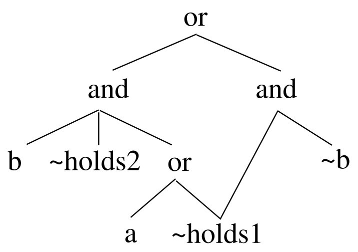

Figure 1: A sentence in $DNNF$.

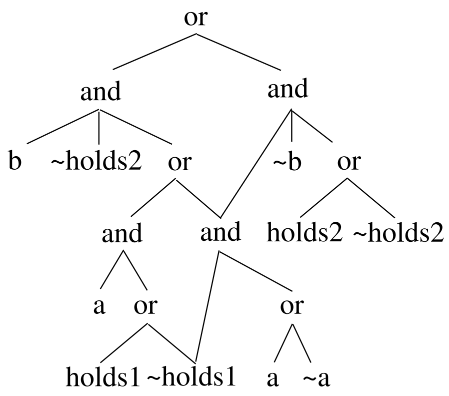

Figure 2: A sentence in smooth $DNNF$.

cording to Proposition 3.1, and $\widehat{W\downarrow}$ is the conjunction of all hard constraints in $W\downarrow$.

An interesting subset of $DNNF$ is the set of *smooth* $DNNF$ sentences [9]:

Definition 4.2 (Smooth DNNF) *A $DNNF$ sentence satisfies the* smoothness *property if and only if for each disjunction C in the sentence, each disjunct of C mentions the same variables.*

Interestingly, every $DNNF$ sentence can be turned into an equivalent, smooth one in polynomial time [9].

For instance, Figure 2 depicts a smooth $DNNF$ which is equivalent to the $DNNF$ in Figure 1. Note that for readibility reasons some leaf nodes are duplicated in the figure.

Among the various tasks that can be achieved in a tractable way from a $DNNF$ sentence are *conditioning*, *clausal entailment*, *forgetting* and *model enumeration* (given that the $DNNF$ is smooth) [9; 11].

4.2 TRACTABLE QUERIES

Given a weighted base W, and given a $DNNF$–compilation of W, we now show how the compilation can be used to represent the preferred models of W as a $DNNF$ in polynomial time.

Definition 4.3 (Minimization of a weighted base) *A* minimization *of a weighted base W is a propositional formula Δ where the models of Δ are $\min_W(\Omega)$.*

Note that this notion generalizes the notion of minimization of a propositional formula ϕ reported in [8], for which the preferred models are those containing a maximal number of variables assigned to true. Such a minimization can be easily achieved in a weighted base setting by considering the base $\{\langle\phi, +\infty\rangle\} \cup \bigcup_{x \in Var(\phi)} \{\langle x, 1\rangle\}$.

Definition 4.4 (Minimization of $DNNF$-compilation) *Let W be a $DNNF$-compilation of a weighted base. Let $\langle\alpha, +\infty\rangle$ be the single hard constraint in W, where α is a $DNNF$ sentence. Suppose that α is also smooth.*

- *We define $k(\alpha)$ inductively as follows:*
  - $k(true) \stackrel{def}{=} 0$ *and* $k(false) \stackrel{def}{=} +\infty$.
  - *If α is a literal, then:*
    - *If $\alpha = \neg holds_i$, then $k(\alpha) \stackrel{def}{=} k_i$, where $\langle holds_i, k_i\rangle \in W$.*
    - *Otherwise, $k(\alpha) \stackrel{def}{=} 0$.*
  - $k(\alpha = \bigvee_i \alpha_i) \stackrel{def}{=} min_i k(\alpha_i)$.
  - $k(\alpha = \bigwedge_i \alpha_i) \stackrel{def}{=} \sum_i k(\alpha_i)$.
- *We define $min(\alpha)$ inductively as follows:*
  - *If α is a literal or a boolean constant, then $min(\alpha) \stackrel{def}{=} \alpha$.*
  - $min(\alpha = \bigvee_i \alpha_i) \stackrel{def}{=} \bigvee_{k(\alpha_i)=k(\alpha)} min(\alpha_i)$.
  - $min(\alpha = \bigwedge_i \alpha_i) \stackrel{def}{=} \bigwedge_i min(\alpha_i)$.

We have the following result:

Proposition 4.1 *Let W be a $DNNF$-compilation of a weighted base. Let $\langle\alpha, +\infty\rangle$ be the single hard constraint in W, where α is a smooth $DNNF$ sentence. Then $min(\alpha)$ is a smooth $DNNF$ and is a minimization of W.*

Figure 3 depicts the weight $k(\alpha)$ of every subformula α of the smooth $DNNF$ sentence given in Figure 2. Figure 4 (left) depicts the minimization of the $DNNF$ in Figure 3. Figure 4 (right) depicts a simplification of this minimized $DNNF$ which has a single model.

Since $min(\alpha)$ can be computed in time polynomial in the size of $DNNF$ α, and since clausal entailment can be done in time linear in the size of α [8], we have:

Corollary 4.1 *The clausal inference problem* CLAUSE $\mid\!\sim_W$ *from $DNNF$-compilations of weighted bases is in* P.

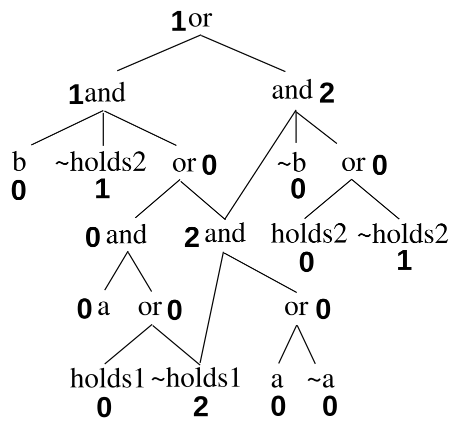

Figure 3: Weights on a smooth $DNNF$ sentence.

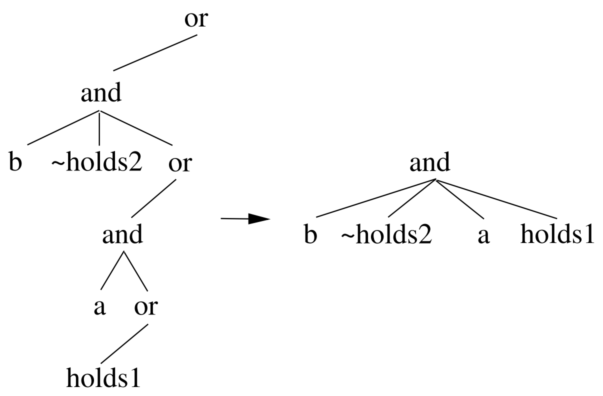

Figure 4: Minimization of a smooth $DNNF$ sentence.

Since model enumeration can be done in output polynomial time from a smooth $DNNF$, we also have:

Corollary 4.2 *The preferred model enumeration problem from $DNNF$-compilations of weighted bases can be solved in output polynomial time.*

5 APPLICATION TO MODEL-BASED DIAGNOSIS

We now briefly sketch how the previous results can be used to compute the set of most probable diagnoses of a system in time polynomial in the size of system description and the output size. The following results generalize those given in [7; 9] to the case where the probability of failure of components is available.

We first need to briefly recall what a consistency-based diagnosis of a system is [22]:

Definition 5.1 (Consistency-based diagnosis)

- *A* diagnostic problem $\mathcal{P} = \langle SD, OK, OBS \rangle$ *is a triple consisting of:*
  - *a formula SD from $PROP_{PS}$, the system description.*
  - *a finite set $OK = \{ok_1, \ldots, ok_n\}$ of propositional symbols. "ok_i is true" means that the component i of the system to be diagnosed is not faulty.*
  - *OBS is a term, typically gathering the inputs and the outputs of the system.*
- *A* consistency-based *diagnosis Δ for $\mathcal{P}$ is a complete OK-term (i.e., a conjunction of literals built up from OK in which every ok_i occurs either positively or negatively) s.t. $\Delta \wedge SD \wedge OBS$ is consistent.*

Because a system can have a number of diagnoses that is exponential in the number of its components, preference criteria are usually used to limit the number of candidates. The most current ones consist in keeping the diagnoses containing as few negative OK-literals as possible (w.r.t. set inclusion or cardinality).

When the *a priori* probability of failure of components is available (and such probabilities are considered independent), the most probable diagnoses for $\mathcal{P}$ can also be preferred. Such a notion of preferred diagnosis generalizes the one based on minimality w.r.t. cardinality (the latter corresponds to the case where the probability of failure of components is uniform and $< \frac{1}{2}$).

Interestingly, the most probable diagnoses for $\mathcal{P}$ can be enumerated in output polynomial time as soon as a smooth $DNNF$-compilation $\mathcal{P}_{DNNF}$ corresponding to $\mathcal{P}$ has been derived first.

Definition 5.2 (Compilation of a diagnostic problem) *Let $\mathcal{P}$ be a diagnostic problem for which the a priori probability of failure p_i of any component i is available.*

$$\mathcal{P}_{DNNF} \stackrel{def}{=} \{\langle DNNF(SD \mid OBS), +\infty \rangle\} \cup \{\langle ok_i, log\ p_i \rangle \mid ok_i \in OK\}$$

is the smooth $DNNF$-compilation associated with $\mathcal{P}$.

In this definition, $SD \mid OBS$ denotes the conditioning of SD by the term OBS, i.e., the formula obtained by replacing in SD every variable x by $true$ (resp. $false$) if x (resp. $\neg x$) is a positive (resp. negative) literal of OBS.

The log transformation performed here enables to compute the log of the probability of a diagnosis Δ as $\sum_{\neg ok_i \in \Delta} log\ p_i$. Because log is strictly nondecreasing, the induced preference ordering between diagnoses is p-reserved.

Proposition 5.1

- $K(\mathcal{P}_{DNNF})$ *is the log of the probability of any most probable diagnosis for* $\mathcal{P}$.
- *The most probable diagnoses for* $\mathcal{P}$ *are the the models of* $Forget(min(DNNF(SD \mid OBS)), PS \setminus OK)$[4].

An important point is that $\mathcal{P}_{DNNF}$ does not have to be re-compiled each time the observations change; indeed, a $DNNF$ sentence $DNNF(SD \mid OBS)$ equivalent to the conditioning of SD by the observations OBS can be computed as $DNNF(SD) \mid OBS$, the conditioning of a $DNNF$ sentence equivalent to SD by OBS. Since conditioning can be achieved in linear time from a $DNNF$ formula, it is sufficient to compile only the system description SD (that is the fixed part of the diagnostic problem) so to compute $DNNF(SD)$ instead of $DNNF(SD \mid OBS)$.

Because (1) forgetting variables in a $DNNF$ formula can be done in polynomial time [8] and (2) the models of a smooth $DNNF$ formula can be generated in time polynomial in the output size [9], we obtain that:

Corollary 5.1 *The most probable diagnoses for a diagnostic problem* $\mathcal{P}$ *can be enumerated in time polynomial in the size of* $\mathcal{P}_{DNNF}$.

6 CONCLUSION

In this paper, we have studied how existing knowledge compilation functions can be used to improve inference from propositional weighted bases. Both negative and positive results have been put forward. On the one hand, we have shown that the inference problem from a compiled weighted base is as difficult as in the general case, when prime implicates, Horn cover or renamable Horn cover target classes are considered. On the other hand, we have shown that this problem becomes tractable whenever $DNNF$-compilations are used. Finally, we have sketched how our results can be used in model-based diagnosis in order to compute the most probable diagnoses of a system.

Acknowledgements

The first author has been partially supported by NSF grant IIS-9988543 and MURI grant N00014-00-1-0617. The second author has been partly supported by the IUT de Lens, the Université d'Artois, the Région Nord/Pas-de-Calais under the TACT-TIC project, and by the European Community FEDER Program.

[4]For every formula ϕ and every set of variables X, $Forget(\phi, X)$ denotes the logically strongest consequence of ϕ that is independent from X, i.e., that can be turned into an equivalent formula in which no variable from X occurs.

References

[1] S. Benferhat, C. Cayrol, D. Dubois, J. Lang, and H. Prade. Inconsistency management and prioritized syntax-based entailment. In *Proc. of the* 13^{th} *International Joint Conference on Artificial Intelligence (IJCAI'93)*, pages 640–645, Chambéry, 1993.

[2] S. Benferhat, D. Dubois, and H. Prade. How to infer from inconsistent beliefs without revising. In *Proc. of the* 14^{th} *International Joint Conference on Artificial Intelligence (IJCAI'95)*, pages 1449–1455, Montreal, 1995.

[3] Y. Boufkhad, E. Grégoire, P. Marquis, B. Mazure, and L. Saïs. Tractable cover compilations. In *Proc. of the* 15^{th} *International Joint Conference on Artificial Intelligence (IJCAI'97)*, pages 122–127, Nagoya, 1997.

[4] M. Cadoli and F.M. Donini. A survey on knowledge compilation. *AI Communications*, 10:137–150, 1997. (printed in 1998).

[5] S. Coste-Marquis and P. Marquis. Compiling stratified belief bases. In *Proc. of the* 14^{th} *European Conference on Artificial Intelligence (ECAI'00)*, pages 23–27, Berlin, 2000.

[6] M. Dalal. Semantics of an anytime family of reasoners. In *Proc. of the* 12^{th} *European Conference on Artificial Intelligence (ECAI'96)*, pages 360–364, Budapest, 1996.

[7] A. Darwiche. Compiling devices: a structure-based approach. In *Proc. of the* 6^{th} *International Conference on Knowledge Representation and Reasoning (KR'98)*, pages 156–166, Trento, 1998.

[8] A. Darwiche. Compiling knowledge into decomposable negation normal form. In *Proc. of the* 16^{th} *International Joint Conference on Artificial Intelligence (IJCAI'99)*, pages 284–289, Stockholm, 1999.

[9] A. Darwiche. Decomposable negation normal form. *Journal of the Association for Computing Machinery*, 48(4):1–42, 2001.

[10] A. Darwiche. A Compiler for Deterministic Decomposable Negation Normal Form. Technical Report D-125, Computer Science Department, UCLA, Los Angeles, Ca 90095, 2002. To appear in AAAI'02.

[11] A. Darwiche and P. Marquis. A perspective on knowledge compilation. In *Proc. of the* 17^{th} *International Joint Conference on Artificial Intelligence (IJCAI'99)*, pages 175–182, Seattle (WA), 2001.

[12] F. Dupin de St Cyr. *Gestion de l'évolutif et de l'incertain en logiques pondérées*. PhD thesis, Université Paul Sabatier, 1996.

[13] F. Dupin de St Cyr, J. Lang, and Th. Schiex. Penalty logic ans its link with Dempster-Shafer theory. In *Proc. of the* 10^{th} *Conference on Uncertainty in Artificial Intelligence (UAI'94)*, pages 204–211, 1994.

[14] R. Dechter and I. Rish. Directional resolution: the Davis-Putnam procedure, revisited. In *Proc. of the* 4^{th} *International Conference on Knowledge Representation and Reasoning (KR'94)*, pages 134–145, Bonn, 1994.

[15] A. del Val. Tractable databases: how to make propositional unit resolution complete through compilation. In *Proc. of the* 4^{th} *International Conference on Knowledge Representation and Reasoning (KR'94)*, pages 551–561, Bonn, 1994.

[16] C. Lafage and J. Lang. Logical representation of preferences for group decision making. In *Proc. of the* 7^{th} *International Conference on Knowledge Representation and Reasoning (KR'00)*, pages 457–468, Breckenridge (CO), 2000.

[17] P. Marquis. Knowledge compilation using theory prime implicates. In *Proc. of the* 14^{th} *International Joint Conference on Artificial Intelligence (IJCAI'95)*, pages 837–843, Montreal, 1995.

[18] Ch. Papadimitriou. *Computational complexity*. Addison–Wesley, 1994.

[19] G. Pinkas. Propositional nonmonotonic reasoning and inconsistency in symmetric neural networks. In *Proc. of the* 12^{th} *International Joint Conference on Artificial Intelligence (IJCAI'91)*, pages 525–530, 1991.

[20] G. Pinkas. Reasoning, nonmonotonicity and learning in connectionnist networks that capture propositional knowledge. *Artificial Intelligence*, 77:203–247, 1995.

[21] G. Pinkas and R.P. Loui. Reasoning from inconsistency: a taxonomy of principles for resolving conflict. In *Proc. of the* 3^{rd} *International Conference on Knowledge Representation and Reasoning (KR'92)*, pages 709–719, Cambridge (MA), 1992.

[22] R. Reiter. A theory of diagnosis from first principles. *Artificial Intelligence*, 32:57–95, 1987.

[23] R. Reiter and J. de Kleer. Foundations of assumption-based truth maintenance systems: preliminary report. In *Proc. of the* 6^{th} *National Conference on Artificial Intelligence (AAAI'87)*, pages 183–188, Seattle (WA), 1987.

[24] R. Schrag. Compilation for critically constrained knowledge bases. In *Proc. of the* 13^{th} *National Conference on Artificial Intelligence (AAAI'96)*, pages 510–515, Portland (OR), 1996.

[25] B. Selman and H.A. Kautz. Knowledge compilation and theory approximation. *Journal of the Association for Computing Machinery*, 43:193–224, 1996.