Tractable Cover Compilations*

Yacine Boufkhad[1], **Éric Grégoire**[2], **Pierre Marquis**[2],
Bertrand Mazure[2], **Lakhdar Saïs**[2,3]

[1] LIP6
Université Paris 6
4, place Jussieu
F-75252 Paris Cedex 05, FRANCE
boufkhad@laforia.ibp.fr

[2] CRIL [3] IUT de Lens
Université d'Artois
Rue de l'Université, S.P. 16
F-62307 Lens Cedex, FRANCE
{gregoire,marquis,mazure,sais}@cril.univ-artois.fr

Abstract

Tractable covers are introduced as a new approach to equivalence-preserving compilation of propositional knowledge bases. First, a general framework is presented. Then, two specific cases are considered. In the first one, partial interpretations are used to shape the knowledge base into tractable formulas from several possible classes. In the second case, they are used to derive renamable Horn formulas. This last case is proved less space-consuming than prime implicants cover compilations for every knowledge base. Finally, experimental results show that the new approaches can prove efficient w.r.t. direct query answering and offer significant time and space savings w.r.t. prime implicants covers.

1 Introduction

Different approaches have been proposed to circumvent the intractability of propositional deduction. Some of them restrict the expressive power of the representation language to tractable classes, like the Horn, reverse Horn, binary, monotone, renamable Horn, q-Horn, nested clauses formulas [Dowling and Gallier, 1984; Lewis, 1978; Boros *et al.*, 1994; Knuth, 1990]. Unfortunately, such classes are not expressive enough for many applications. Contrastingly, *compilation approaches* apply to full propositional-logic knowledge bases (KBs for short). Thanks to an off-line pre-processing step, a KB Σ is compiled into a formula Σ^* so that on-line query answering can be performed tractably from Σ^*. Many approaches to compilation have been proposed so far, mainly [Reiter and De Kleer, 1987; Selman and Kautz, 1991; 1994; del Val, 1994; Dechter and Rish, 1994; Marquis, 1995; del Val, 1995; 1996; Marquis and Sadaoui, 1996; Schrag, 1996].

In this paper, a new approach to equivalence-preserving compilation, called *tractable covers*, is introduced. In short, a tractable cover of Σ is a finite set $\mathcal{T}$ of tractable formulas Φ (disjunctively considered) s.t. $\Sigma \equiv \mathcal{T}$. Tractable covers of Σ are equivalence-preserving compilations of Σ: a clause c is a logical consequence of Σ iff for every Φ in $\mathcal{T}$, c is a logical consequence of Φ. Since Φ is tractable, each elementary test $\Phi \models c$ can be computed in time polynomial in $|\Phi| + |c|$. The point is to find out tractable Φs that concisely represent (i.e. cover) the largest sets of models of Σ, so that $|\mathcal{T}|$ remains limited. To some extent, the present work could then be related to other model-based approaches to knowledge representation and reasoning, like [Khardon and Roth, 1996].

First, a general framework is presented, which can take advantage of most tractable classes, simultaneously. In many respects, it generalizes the prime implicants cover technique recently used for compilation purpose [Schrag, 1996]. Then, the focus is laid on tractable covers that can be computed and intensionally represented thanks to (partial) interpretations. Two specific cases are considered. In the first one, partial interpretations are used to shape the KB into formulas from several possible classes. In the second one, they are used to derive renamable Horn formulas. The last one is proved less space-consuming than prime implicants covers [Schrag, 1996] for every KB. Since tractable covers of Σ are equivalence-preserving compilations, their size may remain exponential in $|\Sigma|$ unless $NP \subseteq P/poly$ [Selman and Kautz, 1994], which is very unlikely. However, experimental results show that the new approaches can prove efficient w.r.t. direct query answering and offer significant time and space savings w.r.t. prime implicants covers.

2 Formal Preliminaries

A literal is a propositional variable or a negated one. A clause (resp. a term) is a finite set of literals, representing their disjunction (resp. conjunction). A Horn (resp. reverse Horn) clause contains at most one literal that is

*This work has been supported in part by the Ganymède II project of the Contrat État/Région Nord–Pas-de-Calais.

positive (resp. negative). A binary clause contains at most two literals. A KB Σ is a finite set of propositional formulas, conjunctively considered. $Var(\Sigma)$ is the set of variables occurring in Σ. Every KB can be rewritten into a conjunction of clauses (into CNF for short) while preserving equivalence. A KB Σ is said renamable Horn iff there exists a substitution σ over literals l built up from $Var(\Sigma)$, s.t. $\sigma(l) = \bar{l}$ and $\sigma(\Sigma)$ is Horn.

Interpretations and models are defined in the usual way. Let us stress that, quite unconventionally, (partial) interpretations will be represented as terms, i.e. as satisfiable sets of literals (vs. sets of variables). An implicant of a KB Σ is a partial interpretation α s.t. $\alpha \models \Sigma$. A prime implicant of Σ is an implicant π of Σ s.t. for all implicants α of Σ s.t. $\pi \models \alpha$, we have $\alpha \models \pi$. The set of all prime implicants of Σ (up to logical equivalence) is noted $PI(\Sigma)$. A prime implicant cover of Σ is any subset S of $PI(\Sigma)$ (disjunctively considered) s.t. $S \equiv \Sigma$.

3 Tractable Cover Compilations

3.1 The General Framework

First, let us make precise what classes of tractable formulas will be considered:

Definition 3.1 *A list Cs of* classes C of tractable propositional formulas *(w.r.t. cover compilation) is s.t.:*

- *There exists a polytime algorithm $TRACTABLE?$ for checking whether any formula Φ belongs to C.*
- *There exists a polytime algorithm $QUERY?$ for checking whether $\Phi \models c$ holds for any Φ in C and any clause c.*
- *For every C in Cs, for every formula Φ of C and every term p, there exists C' in Cs s.t. $(\Phi \wedge p) \in C'$.*

Since classes of tractable formulas are not finite sets in the general case, they are intensionally represented by ordered pairs of decision procedures $\langle TRACTABLE?, QUERY?\rangle$.

Interestingly, the great majority of classes of formulas tractable for SAT (the well-known propositional satisfiability decision problem) are also tractable for cover compilations. Especially, this is the case for the Horn, reverse Horn, binary, renamable Horn, q-Horn and nested clauses classes.

We are now ready to define the notion of a tractable cover of a propositional KB.

Definition 3.2 *Given a CNF-KB Σ and a finite list Cs of classes of tractable formulas w.r.t. cover compilation (represented intensionally), a* tractable cover *of Σ w.r.t. Cs is a finite set $\mathcal{T}$ of satisfiable formulas (disjunctively considered) s.t.:*

- *For every $\Phi_i \in \mathcal{T}$, there exists C in Cs s.t. Φ_i belongs to C, and*
- *$\mathcal{T} \equiv \Sigma$.*

In the following, we will assume that Cs contains at least the class $\{t \text{ s.t. } t \text{ is a term}\}$. This ensures that there always exists at least one tractable cover of Σ, namely a prime implicants one.

In this paper, we focus on tractable covers that can be *intensionally represented*, using (partial) interpretations. The corresponding explicit covers can be generated online from the intensional ones in polynomial time.

3.2 Carver-Based Tractable Covers

A first way to derive covers consists in *simplifying* the KB using partial interpretations. For any partial interpretation $p = \{l_1, \ldots, l_n\}$, the KB Σ simplified w.r.t. p is the set $\Sigma_p = (((\Sigma_{l_1})_{l_2}) \ldots)_{l_n}$, where $\Sigma_l = \{c \setminus \{\neg l\}$ s.t. $c \in \Sigma$ and $c \cap \{l\} = \emptyset\}$.

Definition 3.3 *Given a CNF-KB Σ and a finite list Cs of classes of tractable formulas (represented intensionally):*

- *A* carver *of Σ w.r.t. Cs is a partial interpretation p s.t. there exists C in Cs s.t. Σ_p belongs to C and Σ_p is satisfiable.*
- *A* carver-based tractable cover *of Σ w.r.t. Cs is a set PC of carvers of Σ w.r.t. Cs (disjunctively considered) s.t. $(PC \wedge \Sigma) \equiv \Sigma$.*
- *A* carver-based cover compilation *of Σ is a pair $\mathcal{C} = \langle \Sigma, PC\rangle$, where PC is a carver-based tractable cover of Σ w.r.t. Cs.*

Interestingly, the space required to store a carver p is always lower or equal to the space needed by the corresponding tractable formula $\Phi = p \wedge \Sigma_p$ (with a $O(|\Sigma|)$ factor in the worst case). At the on-line query answering stage, the price to be paid is a $O(|\Sigma \wedge p|)$ time complexity overhead per carver p but this does not question the tractability of query answering.

As an example, let $\Sigma = \{\bar{x}_1 \vee \bar{x}_2 \vee \bar{x}_3 \vee x_4, x_1 \vee x_2 \vee x_4\}$ and let Cs contain the Horn and the binary classes. The set $PC = \{\{x_1\}, \{\bar{x}_1\}\}$ is a carver-based tractable cover of Σ w.r.t. Cs since $\Sigma_{\{x_1\}} = \{\bar{x}_2 \vee \bar{x}_3 \vee x_4\}$ is Horn and $\Sigma_{\{\bar{x}_1\}} = \{x_2 \vee x_4\}$ is binary. If Cs reduces to the renamable Horn class, $\{\emptyset\}$ is a carver-based tractable cover of Σ w.r.t. Cs since Σ belongs to the renamable Horn class.

Clearly enough, carver-based cover compilations are equivalence-preserving compilations:

Proposition 3.1 *Let Σ be a CNF-KB, $\mathcal{C} = \langle \Sigma, PC\rangle$ be a carver-based cover compilation of Σ and let c be a clause. $\Sigma \models c$ iff for every p in PC, $((p \models c)$ or $(\Sigma_p \models c))$, which can be decided in time polynomial in $|\mathcal{C}| + |c|$.*

Although the class(es) to which Σ_p belongs can be polynomially recognized on-line for each carver p of PC, in practice, they are determined once only at the off-line compiling stage. Accordingly, each carver p found during the compiling process is indexed with a reference $label(p)$ to the concerned class in Cs. A significant amount of time in on-line query answering can be saved via such indexing.

Interestingly, carver-based cover compilations can lead to exponential space savings w.r.t. prime implicants cover compilations for some KBs. An extreme case consists of tractable KBs Σ that remain invariant under carver-based cover compilation but are such that every prime implicant cover is exponential in the size of Σ.

3.3 Hyper-Implicant Covers

Formulas in covers must be tractable and exhibit as many models of Σ as possible. Interestingly, several classes C of tractable formulas admit model-theoretic characterizations and features that can be exploited. For the binary, Horn, reverse Horn, renamable Horn classes, among others, class membership is equivalent to the existence of some specific interpretation(s). Formally, a CNF-KB Σ is renamable Horn [Lewis, 1978] iff there exists an interpretation p over $Var(\Sigma)$ s.t. for every clause c of Σ, at most one literal of c does not belong to p. Interestingly, the renamable Horn class includes both the Horn, reverse Horn, and satisfiable binary KBs as proper subsets.

As the semantical characterizations above indicate it, literals that belong both to p and to Σ play a central role that we can take advantage of. In order to derive satisfiable renamable Horn formulas, every clause c of Σ is shortened using a model $p = m$ of Σ: only the literals occurring in m are kept, plus an additional literal of c (when possible). In this way, every shortened clause is such that every literal of m (except possibly one) belongs to it. Accordingly, the resulting set of clauses, called *hyper-implicant* of Σ, is satisfiable and renamable Horn.

Formally, let m be a model of Σ. For every clause c in Σ, let c^m be the clause consisting of the literals common to c and m. Let $P(\Sigma, m)$ denote the set of clauses of Σ s.t. for every clause c in $P(\Sigma, m)$, c and c^m are identical. Let $N(\Sigma, m)$ be $\Sigma \setminus P(\Sigma, m)$. For every clause c in $N(\Sigma, m)$, let l^m_c denote a literal of c s.t. $\overline{l^m_c}$ belongs to m. Obviously, several candidates l^m_c may exist in the general case.

Definition 3.4 *Given a CNF-KB Σ:*

- *A* hyper-implicant *of Σ (w.r.t. to a model) m of Σ is a formula* $\Sigma^m = (\bigwedge_{c \in P(\Sigma,m)} c) \wedge (\bigwedge_{c \in N(\Sigma,m)} (c^m \vee l^m_c))$.
- *A* hyper-implicant cover *$\mathcal{H}$ of Σ is a set of hyper-implicants of Σ (disjunctively considered) s.t. $\mathcal{H} \equiv \Sigma$.*

Importantly, the size of any hyper-implicant Σ^m of Σ is strictly lower than the size of Σ, except when Σ is renamable Horn.

Hyper-implicant covers are equivalence-preserving compilations:

Proposition 3.2 *Let Σ be a CNF-KB, $\mathcal{H}$ be a hyper-implicant cover of Σ and c be a clause. $\Sigma \models c$ iff for every hyper-implicant Σ^m of $\mathcal{H}$, we have $\Sigma^m \models c$. This can be decided in time linear in $|\mathcal{H}| + |c|$.*

Assuming that the clauses in $\Sigma = \{c_1, \ldots, c_n\}$ are totally ordered, hyper-implicant covers can be represented intensionally by Σ and sets of pairs $\langle m, [l^m_{c_1}, \ldots, l^m_{c_n}] \rangle$ where m is a model of Σ and $l^m_{c_i}$ ($i \in 1..n$) is $false$ if $c_i \in P(\Sigma, m)$ and is the literal of $c_i \setminus m$ which is kept, otherwise. Each Σ^m can be derived on-line from Σ and $\langle m, [l^m_{c_1}, \ldots, l^m_{c_n}] \rangle$ in linear time.

As an example, let $\Sigma = \{x_1 \vee x_3 \vee x_4, x_1 \vee \bar{x}_2 \vee x_3, x_1 \vee x_2 \vee \bar{x}_3 \vee \bar{x}_4, \bar{x}_1 \vee \bar{x}_2 \vee \bar{x}_3 \vee \bar{x}_4, \bar{x}_1 \vee x_2 \vee \bar{x}_3 \vee x_4\}$. The hyper implicant cover represented by Σ and the two pairs $\langle \{x_1, \bar{x}_2, \bar{x}_3, x_4\}, [x_3, x_3, x_2, \bar{x}_1, \bar{x}_1] \rangle$ and $\langle \{\bar{x}_1, x_2, x_3, \bar{x}_4\}, [x_1, \bar{x}_2, \bar{x}_3, \bar{x}_2, x_4] \rangle$, contains two hyper-implicants: $\{x_1 \vee x_4 \vee x_3, x_1 \vee \bar{x}_2 \vee x_3, x_1 \vee \bar{x}_3 \vee x_2, \bar{x}_2 \vee \bar{x}_3 \vee \bar{x}_1, \bar{x}_3 \vee x_4 \vee \bar{x}_1\}$ and $\{x_3 \vee x_1, x_3 \vee \bar{x}_2, x_2 \vee \bar{x}_4 \vee \bar{x}_3, \bar{x}_1 \vee \bar{x}_4 \vee \bar{x}_2, \bar{x}_1 \vee x_2 \vee x_4\}$.

Intuitively, any hyper-implicant Σ^m of Σ is a concise representation of some prime implicants of Σ. To be more specific, the prime implicants of Σ which are entailed by m are prime implicants of Σ^m.

Proposition 3.3 *For every model m of Σ, let $PI_m(\Sigma)$ be the set of prime implicants of Σ s.t. for every π in $PI_m(\Sigma)$, we have $m \models \pi$. Then, $PI_m(\Sigma) \subseteq PI(\Sigma^m)$.*

Consequently, Σ^m covers every model of Σ covered by a prime implicant of Σ entailed by m.

Hyper-implicants covers are economical representations w.r.t. prime implicants ones. Not withstanding the fact that hyper-implicants of Σ are smaller than Σ, the number of hyper-implicants in a cover is lower than the number of implicants in a cover:

Proposition 3.4 *For every prime implicant cover of Σ containing t prime implicants there exists a hyper-implicant cover of Σ containing at most $\lfloor \frac{t+1}{2} \rfloor$ hyper-implicants.*

In the example above, the hyper-implicant cover contains 2 formulas, while the smallest prime implicant cover of Σ consists of 6 prime implicants. Actually, experiments show significant savings w.r.t. the sizes of the prime implicants covers for most KBs (cf. Section 5).

4 Computing Compilations

(Partial) interpretations giving rise to intensionally-represented tractable covers are computed using systematic search, thanks to a Davis/Putnam-like procedure `DPTC`. This procedure is closely related to Schrag's `DPPI` algorithm [Schrag, 1996]. `Cs` is empty for the hyper-implicant case.

```
DPTC(Σ,Cs):
  PC ← ∅;
  DP*(Σ,Cs,∅);
  Return(⟨Σ,PC⟩).
```

```
DP*(Σ,Cs,p):
  If not PRUNING(Σ,p) then
     UNIT_PROPAGATE(Σ,p);
     If ∅ ∈ Σ then return;
     If (p ⊨ Σ)
```

```
        then PROCESS_IMPLICANT(p,Σ,Cs)
             return;
    l← CHOOSE_BEST_LITERAL(Σ);
    DP*(Σ, Cs, p ∪ {l});
    DP*(Σ, Cs, p ∪ {¬l}).
```

The CHOOSE_BEST_LITERAL branching rule and UNIT_PROPAGATE procedures are standard Davis/Putnam features. In our experiments, the branching rule by [Dubois *et al.*, 1996] is used. The main role of DP* is to find implicants of Σ in the whole search tree. PRUNING and PROCESS_IMPLICANT depend on the considered approach. From the found implicants, (partial) interpretations (carvers and implicit representations of hyper-implicants) are derived thanks to PROCESS_IMPLICANT; they are collected into the global variable PC.

4.1 The Carver-Based Case

```
PRUNING_C(Σ,new_p):
    If there exists p ∈ PC s.t. (new_p ⊨ p)
       then return(true) else return(false).
```

```
PROCESS_IMPLICANT_C(new_p,Σ,Cs):
    Carver ← DERIVE_C(new_p,Σ,Cs);
    For every p ∈ PC do
        If (p ⊨ Carver) then remove p from PC;
    Put Carver in PC.
```

```
DERIVE_C(p,Σ,Cs):
    Prime ← ONE_PRIME(p,Σ);
    For every literal l ∈ Prime do
     For every ⟨TRACTABLE?, QUERY?⟩ in Cs do
       If TRACTABLE?(Σ_{Prime\{l}})
          then Return(DERIVE_C(Prime\{l}, Σ, Cs));
    Return(Prime).
```

Whenever an implicant p of Σ is found, a prime implicant Prime of Σ s.t. $p \models$ Prime is extracted from p, thanks to the ONE_PRIME procedure. ONE_PRIME considers every literal l of p successively, check whether l is necessary (i.e. if there exists a clause c of Σ s.t. $c \cap p = \{l\}$), and remove l from p if l is not necessary (see [Castell and Cayrol, 1996][Schrag, 1996] for details). These prime implicants are then tentatively simplified; all the literals of each Prime are considered successively; for every literal l of Prime, $\Sigma_{Prime\setminus\{l\}}$ is checked for tractability using the recognition procedures given in Cs. If $\Sigma_{Prime\setminus\{l\}}$ is found tractable, l is ruled out from Prime and is kept otherwise. Once all the literals of Prime have been considered, the resulting simplified Prime (indexed by the label of the corresponding class in Cs) is checked against the set PC of carvers collected so far. Only maximal carvers w.r.t. $\models$ are kept in PC. Interestingly, the set PC of carvers stored during the traversal of the DP search tree is used to prune this tree, thanks to the PRUNING$_C$ procedure; indeed, whenever a candidate (partial) interpretation new_p is elected, it can be immediately removed if new_p entails one of the current carvers.

Clearly enough, both the literal ordering and the recognition procedure ordering used in DERIVE$_C$ can greatly influence the cover generated in this way.

4.2 The Hyper-Implicant Case

```
PRUNING_H(Σ,new_p):
    If there exists p ∈ PC s.t. (Σ^{new_p} ⊨ Σ^p)
       then return(true) else return(false).
```

```
PROCESS_IMPLICANT_H(new_p,Σ,_):
    Model ← DERIVE_H(new_p,Σ);
    For every p ∈ PC do
        If (Σ^{Model} ⊨ Σ^p) then remove p from PC;
    Put Model in PC.
```

```
DERIVE_H(p,Σ):
    Model ← p;
    For every variable v ∈ Var(Σ) do
        If ({v} ∩ p = ∅) and ({¬v} ∩ p = ∅)
           then put v with its most frequent
                sign in Σ to Model;
    Return(Model).
```

Each time an implicant p of Σ is found, a model Model of Σ s.t. Model $\models p$ is derived from p. The variables that do not occur in p are added with the sign occurring the most frequently in Σ. When Σ^{Model} is computed for the first time, a list $[l^m_{c_1}, \ldots, l^m_{c_n}]$ is attached to $Model$. The DERIVE$_H$ procedure is both simpler and more efficient than DERIVE$_C$ (roughly, it is not more time-consuming than the prime implicant extraction achieved by ONE_PRIME). The tractable formulas Σ^p corresponding to the models p collected in PC are used to prune the search tree (PRUNING$_H$). Only the ps giving rise to the logically weakests Σ^p are kept in PC. Since the renamable Horn formulas Σ^{Model} and Σ^p stem from the same KB Σ, checking whether ($\Sigma^{Model} \models \Sigma^p$) can be performed in a very efficient way.

Both the literals chosen to extend the found implicants to models p of Σ, and the literals l^m_c selected in clauses of Σ^p may have a significant impact on the hyper-implicants Σ^p that are generated.

5 Experimental Results

In contrast to SAT, only few benchmarks for knowledge compilation can be found in the literature (with the well-developed experimental framework of [Schrag, 1996] as an exception). Actually, no comprehensive analysis of what should be the nature of meaningful benchmarks for evaluating compilation approaches has ever been conducted. Clearly, benchmarks must be hard for query answering since the goal of knowledge compilation is to overcome its intractability. However, in contrast to [Schrag, 1996], we do not focus on hard SAT instances,

problems	#var	#cla	$\alpha_{\mathcal{P}}$ $\alpha_{\mathcal{C}}$ $\alpha_{\mathcal{H}}$	$\beta_{\mathcal{P}}$ $\beta_{\mathcal{C}}$ $\beta_{\mathcal{H}}$	$\lvert\mathcal{P}\rvert$ $\lvert\mathcal{C}\rvert$ $\lvert\mathcal{H}\rvert$
adder	21	50	5.08	—	5376
			2.09	—	1044
			0.20	78.75	305
history-ex	21	17	2.00	—	172
			0.75	1000.00	234
			0.14	11.66	77
two-pipes	15	54	0.66	125.00	255
			0.60	125.00	234
			0.12	20.00	192
three-pipes	21	82	0.47	45.45	441
			0.42	<1	373
			0.27	17.50	284
four-pipes	27	110	0.36	23.80	675
			0.25	18.51	528
			0.20	29.16	380
regulator	21	106	0.51	26.31	861
			0.29	20.83	530
			0.17	29.99	422
selenoid	11	19	2.16	—	297
			1.40	—	115
			0.20	17.49	94
valve	13	50	0.85	<1	312
			0.66	<1	297
			0.16	21	246

Table 1: Experimental results.

only. Hard SAT instances (with respect to current algorithms) should be considered hard for query answering since at least one query (namely, the empty clause) is difficult. However, easy SAT instances can exhibit hard queries that differ from the empty clause.

Accordingly, we tested the tractable covers and the prime implicants approaches w.r.t. many KBs, including "standard" structured problems, taken from [Forbus and De Kleer, 1993], and random k-SAT problems [Dubois *et al.*, 1996], varying the $\#cla(use)/\#var(iable)$ ratio from the easy to the hard regions. Each KB Σ has been compiled using the 3 techniques. Then, 500 queries have been considered. In order to check the usefulness of the compilation process, we also answered these queries from Σ, using a direct, uncompiled, Davis/Putnam-based approach [Dubois *et al.*, 1996]. For each problem Σ and each compilation technique, the ratios $\alpha = Q_C/Q_U$ and $\beta = C/(Q_U - Q_C)$ have been computed. Q_C (resp. Q_U) is the time needed by the compiled (resp. uncompiled) approach to answer all the queries, and C is the compilation time. α (resp. β) tells us how much query time improvement we get from compilation (resp. how many queries are required to amortize the cost of compilation).

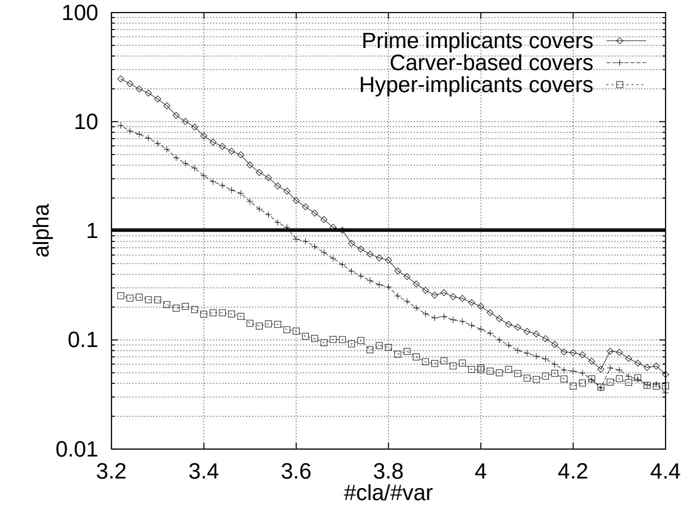

Figure 1: Ratios α.

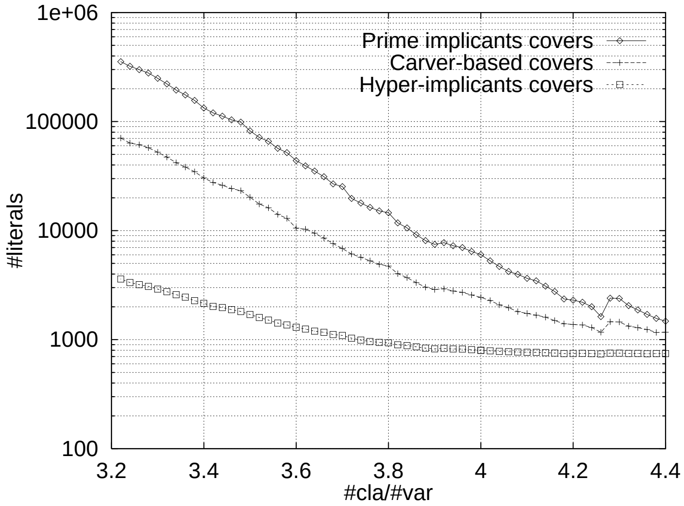

Figure 2: Sizes of the compilations.

Table 1 reports some results of our extensive experiments. For each problem [Forbus and De Kleer, 1993], it lists results for the prime implicants, carvers, and hyper-implicants covers, successively. Especially, it gives the ratios α and β and the size (in literals) of the corresponding cover. The size of any tractable cover compilation is the size of Σ plus the size of the set of (partial) interpretations used as an implicit representation. For the carver-based approach, only the Horn, reverse Horn and binary classes have been considered. For the hyper-implicant approach, simplification of the cover (i.e. lines 2 to 4 of the $\texttt{PROCESS\_IMPLICANT}_{\mathcal{H}}$ procedure) has not been implemented.

Results obtained on 50 variables random 3-SAT problems, where the ratio $\#cla/\#var$ varies from 3.2 to 4.4, are reported on the two next figures. 50 problems have been considered per point and the corresponding scores averaged. Figure 1 (resp. Figure 2) gives aggregate values of ratios α (resp. sizes in literals) obtained for each compilation technique. $\alpha = 1$ separates the region for which compilation is useful from the region for which it is not.

At the light of our experiments, tractable covers prove better than prime implicants covers, both for structured and random k-SAT problems. Significant time savings w.r.t. query answering and significant space savings are obtained. Especially, tractable covers prove useful for many KBs for which prime implicants covers are too large to offer improvements w.r.t. query answering. More, the tractable cover approach allows the compilation of KBs which have so huge prime implicants covers $\mathcal{P}$ that $\mathcal{P}$ cannot be computed and stored. This coheres with the theoretical results reported in [Boufkhad and Dubois, 1996], showing that the average number of prime implicants of k-SAT formulas is exponential in their number of variables.

6 Conclusion

Both theoretical and experimental results show the tractable cover approach promising and encourage us to extend it in several directions. A first issue for further research is how to determine efficiently the best suited classes of tractable formulas for a given KB Σ. On the experimental side, an extensive evaluation of the carver-based technique equipped with more expressive tractable classes must be done. Extending the hyper-implicant approach to other tractable classes, especially the q-Horn one [Boros *et al.*, 1994], is another interesting perspective. Finally, fragments of tractable covers of Σ can serve as approximate compilations (lower bounds) of Σ in the sense of [Selman and Kautz, 1991; 1994; del Val, 1995; 1996]. Since the tractable cover approach allows disjunctions of tractable formulas from several classes, better approximations could be obtained.

References

[Boros *et al.*, 1994] E. Boros, P.L. Hammer, and X. Sun. Recognition of q-horn formulae in linear time. *Discrete Applied Mathematics*, 55(1):1–13, 1994.

[Boufkhad and Dubois, 1996] Y. Boufkhad and O. Dubois. Length of prime implicants and number of solutions of random r-cnf formulas. *(submitted)*, 1996.

[Castell and Cayrol, 1996] T. Castell and M. Cayrol. Computation of prime implicates and prime implicants by the davis and putnam procedure. In *Proc. ECAI'96 Workshop on Advances in Propositional Deduction*, pages 61–64, Budapest, 1996.

[Dechter and Rish, 1994] R. Dechter and I. Rish. Directional resolution: The davis-putnam procedure, revisited. In *Proc. KR'94*, pages 134–145, Bonn, 1994.

[del Val, 1994] A. del Val. Tractable databases: How to make propositional unit resolution complete through compilation. In *Proc. KR'94*, pages 551–561, 1994.

[del Val, 1995] A. del Val. An analysis of approximate knowledge compilation. In *Proc. IJCAI'95*, pages 830–836, Montreal, 1995.

[del Val, 1996] A. del Val. Approximate knowledge compilation: The first-order case. In *Proc. AAAI'96*, pages 498–503, Portland (OR), 1996.

[Dowling and Gallier, 1984] W. Dowling and J. Gallier. Linear time algorithms for testing the satisfiability of propositional horn formulae. *Journal of Logic Programming*, 1(3):267–284, 1984.

[Dubois *et al.*, 1996] O. Dubois, P. Andre, Y. Boufkhad, and J. Carlier. Sat vs. unsat. In 2^{nd} *DIMACS Implementation Challenge*, volume 26 of *DIMACS Series*, pages 415–436. American Mathematical Society, 1996.

[Forbus and De Kleer, 1993] K.D. Forbus and J. De - Kleer. *Building Problem Solvers*. MIT Press, 1993.

[Khardon and Roth, 1996] R. Khardon and D. Roth. Reasoning with models. *Artificial Intelligence*, 87:187–213, 1996.

[Knuth, 1990] D.E. Knuth. Nested satisfiability. *Acta Informatica*, 28:1–6, 1990.

[Lewis, 1978] H.R. Lewis. Renaming a set of clauses as a horn set. *JACM*, 25:134–135, 1978.

[Marquis and Sadaoui, 1996] P. Marquis and S. Sadaoui. A new algorithm for computing theory prime implicates compilations. In *Proc. AAAI'96*, pages 504–509, Portland (OR), 1996.

[Marquis, 1995] P. Marquis. Knowledge compilation using theory prime implicates. In *Proc. IJCAI'95*, pages 837–843, Montreal, 1995.

[Reiter and De Kleer, 1987] R. Reiter and J. De Kleer. Foundations of assumption-based truth maintenance systems: Preliminary report. In *Proc. AAAI'87*, pages 183–188, Seattle (WA), 1987.

[Schrag, 1996] R. Schrag. Compilation for critically constrained knowledge bases. In *Proc. AAAI'96*, pages 510–515, Portland (OR), 1996.

[Selman and Kautz, 1991] B. Selman and H. Kautz. Knowledge compilation using horn approximations. In *Proc. AAAI'91*, pages 904–909, 1991.

[Selman and Kautz, 1994] B. Selman and H. Kautz. Knowledge compilation and theory approximation. *JACM*, 43(2):193–224, 1994.