Knowledge Compilation Using Theory Prime Implicates

Pierre Marquis
CRIN-CNRS and INRIA-Lorraine
Batiment LORIA - B.P. 239
F-54506 Vandceuvre-l&s-Nancy Cedex - FRANCE
e-mail: marquis@loria.fr

Abstract

In this paper, we are mainly concerned with logical compilations of propositional knowledge bases. We propose a new approach to equivalence-preserving knowledge compilation based on a generalization of the standard notion of prime implicate, the *theory prime implicates.* Our approach consists in taking advantage of tractable theories Φ implied by the knowledge base Σ to make Σ local by computing the theory prime implicates of Σ w.r.t. Φ. As a main result, query answering from theory prime implicates compilations can be done in time polynomial in their sizes. While there is no guarantee that the size of a compilation is not exponentially greater than the size of the original knowledge base, we show that it is smaller than the size of the prime implicates compilation in the general case and it can even be exponentially smaller. Additionally, we present some experimental results providing evidence for the substantial space savings achievable with our compilation technique.

1 Introduction

Query answering from a propositional knowledge base is a central concern in artificial intelligence. Unfortunately, its computational complexity is high, both theoretically (co-NP-complete), and practically since every existing algorithm runs in time exponential in the size of the knowledge base in the worst case.

To improve query answering and deal with its intractability, several approaches have been proposed so far. A first approach consists in restricting the representation language (e.g. considering only Horn formulas) or the inference mechanisms (e.g. involving resource-bounded reasoning). Another approach consists in approximating the knowledge base [Selman and Kautz, 1991; Kautz and Selman, 1994] or the logical entailment relation [Cadoli and Schaerf, 1991].

A third family gathers *equivalence-preserving compilation techniques.* Compiling a knowledge base consists in translating (compiling) the base during a preprocessing phase into an equivalent compiled form (a compilation) from which query answering is tractable. In contrast to the two other approaches, equivalence-preserving compilation does not give up expressiveness or equivalence but simply speeds inference by shifting computational costs from the on-line query answering process to an off-line compilation process.

Relevant to this last approach is, among others (e.g. [del Val, 1994; Mathieu and Delahaye, 1994; Dechter and Rish, 1994]), Reiter and de Kleer's technique which consists in turning the knowledge base Σ into the prime implicates normal form [Reiter and de Kleer, 1987].

Interestingly, this last technique relies on a more general and computationally valuable principle of *locality.* Let us say that a knowledge base Σ is local for query answering when, for every formula π, π is a logical consequence of Σ iff there exists a formula π' in Σ s.t. π is a logical consequence of π'. Now, let us assume that Σ contains n formulas π' of size m and that π is of size p. The time cost $c(\Sigma, \pi)$ for answering query π from Σ is (in practice) exponential in $n*m + p$ in the worst case. When Σ is local, the cost comes down to $n*c(\pi', \pi)$, which can easily be exponentially smaller than $c(\Sigma, \pi)$. Additionally, query answering from formulas *rt* can be computationally easier than query answering from the whole knowledge base. This is exactly what happens with prime implicates knowledge bases: they are local for clausal query answering (i.e. considering queries limited to clauses) and, while $c(\Sigma, \pi)$ is exponential in the size of $\Sigma \cup \{\pi\}$, $c(\pi', \pi)$ is polynomial in the size of $\{\pi', \pi\}$ (checking whether a clause π' entails a clause π can be done in time $O(\sup(|\pi'|, |\pi|)^2)$).

In this paper, we propose a new approach to equivalence-preserving knowledge compilation which extends Reiter and de Kleer's technique. Our approach is based on both:
(1) the idea of expressiveness restrictions and/or compilations that make inferences tractable. Thus, we propose a compilation function which takes advantage of tractable theories Φ among those implied by Σ. This leads us to generalize the standard logical entailment relation $\vdash$ to the relation $\vdash_\Phi$ where Φ is built-in.
(2) the idea of locality for clausal query answering. To be more precise, our compilation function aims at making Σ local for clausal query answering by computing the theory prime implicates of Σ w.r.t. Φ; theory prime implicates are to *=(> as prime implicates are to $\vdash$.

The rest of this paper is organized as follows. Some formal preliminaries are given in section 2. In section 3, the concept of theory prime implicate is introduced. Some metatheoretic properties and a procedure for generating theory prime implicates are successively pointed out. In section 4, we present our compilation function which basically consists in computing the theory prime implicates of Σ w.r.t. a tractable theory Φ implied by Σ. As a central result,

we prove that query answering from a compilation can be done in time polynomial in the size of the compilation. In addition, we point out several potential candidates for the notion of "good" tractable theories, i.e. those leading to a reasonable growth of me corresponding compilation. While a compilation which is exponentially larger in size than the original base may easily be produced, the size of the theory prime implicates compilation of a knowledge base is smaller than the size of its prime implicates compilation in the general case and it can even be exponentially smaller. In section 5, we show that our compilation technique can be advantageously compared with other approaches to knowledge compilation in some situations; we present some preliminary experimental results providing evidence for the substantial space savings achievable with this approach. Section 6 concludes the paper with some perspectives. More details about this work can be found in [Marquis, 1995].

2 Formal preliminaries

Let PS be a finite set of propositional symbols (atoms). Let $PROP_{PS}$ denote the propositional language built in the usual way upon PS and the connectives. The elements ψ of $PROP_{PS}$ are called *formulas*. Any finite set of formulas will be identified with the conjunctive formula whose conjuncts are the elements of the set.

An *interpretation* I (on $PROP_{PS}$) is a mapping from PS to {T, F}. The *semantics* of a formula ψ in an interpretation I, noted $[\![\psi]\!](I)$, is an element of {T, F} defined in the usual way. We say that an interpretation I is a *model* of a formula ψ iff $[\![\psi]\!](I) = T$. We will note $\psi \models \phi$ when $[\![\phi]\!](I) = T$ for every model I of ψ, and say that ϕ is a *logical consequence* of ψ. When $\psi \models \phi$ and $\phi \models \psi$, ψ and ϕ are said *equivalent*; this is noted $\psi \equiv \phi$. $PROP_{PS} / \equiv$ denotes the quotient set of $PROP_{PS}$ induced by the equivalence relation $\equiv$.

The *size* $|\psi|$ of a formula ψ is the number of propositional symbols occurring in it. nb_cl(Ψ) will denote the number of clauses occurring in the set of formulas Ψ. We say that a formula ψ is in k-CNF if it is in clausal normal form (CNF), and the size of every clause in ψ is bounded by the constant k. When ψ is in k-CNF, then $|\psi| = \Theta(\text{nb\_cl}(\psi))$.

A *prime implicate* of a finite set Ψ of formulas is a clause π such that:

- $\Psi \models \pi$ holds, and
- for every clause π', if $\Psi \models \pi'$ and $\pi' \models \pi$ hold, then $\pi' \equiv \pi$ holds.

PI(Ψ) will denote the set of prime implicates of Ψ. Clearly enough, the prime implicates of Ψ are the minimal elements w.r.t. $\models$ in the set of all the clauses implied by Ψ (the so-called *implicates* I(Ψ) of Ψ). Formally, we will note it $PI(\Psi) = \min(I(\Psi), \models)$.

Let Φ be a finite set of formulas. Let $\models_\Phi$ denote the relation over $PROP_{PS}$ X $PROP_{PS}$ defined by $\sigma \models_\Phi \tau$ iff $\Phi \cup \{\sigma\} \models \tau$. Clearly enough, $\models_\Phi$ is a partial pre-ordering over $PROP_{PS}$ and $\models_\Phi$ extends $\models$. When $\sigma \models_\Phi \tau$ holds, we say that τ is a *Φ-logical consequence* of σ. Let $\equiv_\Phi$ denote the equivalence relation over $PROP_{PS}$ induced by $\models_\Phi$ and defined by $\sigma \equiv_\Phi \tau$ iff $\sigma \models_\Phi \tau$ and $\tau \models_\Phi \sigma$. When $\sigma \equiv_\Phi \tau$ holds, σ and τ are said *Φ-equivalent*. Let $PROP_{PS} / \equiv_\Phi$ denote the quotient set of $PROP_{PS}$ induced by the equivalence relation $\equiv_\Phi$. By construction, $\models_\Phi$ is a par ordering over $PROP_{PS} / \equiv_\Phi$.

3 Theory prime implicates

3.1 Generalizing prime implicates

Turning the concept of prime implicate into the notion theory prime implicate consists in extending logi entailment by considering some piece of knowledge built-in.

Definition 1 Let Ψ and Φ be finite sets of formulas. clause π is a *theory prime implicate* of Ψ w.r.t. Φ iff:

- $\Psi \models_\Phi \pi$ holds, and
- for every clause π', if $\Psi \models_\Phi \pi'$ and $\pi' \models_\Phi \pi$ ho then $\pi' \equiv_\Phi \pi$ holds.

We note TPI(Ψ, Φ) the set of theory prime implicates Ψ w.r.t. Φ. TPI(Ψ, Φ) is composed of the minin elements w.r.t. $\models_\Phi$ in the set of all the clauses that Φ-logical consequences of Ψ (the *theory implicates* of w.r.t. Φ). Such clauses are considered up to Φ-logi equivalence: only one representative per equivalence clas kept in TPI(Ψ, Φ).

Example 1 Let $\Psi =_{def} \{p \vee q \vee s, p \vee t, q \vee q \vee s \vee u\}$ and $\Phi =_{def} \{\neg p \vee q, \neg q \vee r, \neg t \vee$ Clauses $p \vee t$ and $q \vee s$ are the theory prime implicates Ψ w.r.t. Φ (up to Φ-equivalence).

Clearly enough, the concept of theory prime implic extends the standard notion of prime implicate. Formal $PI(\Psi) = TPI(\Psi, \{\})$.

3.2 Some metatheoretic properties of theory prim implicates

In the following, we point out some metatheoretic propert of theory prime implicates. Thus, we show that TPI(Ψ, is finite and independent of the syntactic nature of formu occurring in Ψ or Φ. We also stress the central role taken theory prime implicates among theory implicates. On t ground, we make the interplay between prime implicates a theory prime implicates precise.

Proposition 1 Let Ψ, Ψ', Φ, Φ' be finite sets formulas. TPI(Ψ, Φ) is a finite set (up to Φ-equivalenc Moreover, if $\Phi \equiv \Phi'$ and $\Psi \equiv_\Phi \Psi'$ hold, th TPI(Ψ', Φ') = TPI(Ψ', Φ') (up to Φ-equivalence).

The next proposition is central for our purpose since shows that TPI(Ψ, Φ) captures in some sense all the clau that are Φ-logical consequences of Ψ.

Proposition 2 Let Ψ and Φ be finite sets of formul and π be a clause. $\Psi \models_\Phi \pi$ holds iff there exists a thec prime implicate π' of Ψ w.r.t. Φ s.t. $\pi' \models_\Phi \pi$ holds.

Example 1 (continued) $r \vee t \vee u$ is a Φ-logic consequence of Ψ since there exists $p \vee t$ in TPI(Ψ, s.t. $p \vee t \models_\Phi r \vee t \vee u$ holds.

As a consequence, turning a set Ψ of formulas into the set of its theory prime implicates w.r.t. Φ is an information-preserving operation given Φ.

Corollary 1 Let Ψ and Φ be finite sets of formulas. $\Psi \equiv_\Phi TPI(\Psi, \Phi)$ holds.

An additional consequence of Proposition 2 is that strengthening Φ can never increase the number of clauses of $TPI(\Psi, \Phi)$, as long as Φ is a logical consequence of Ψ.

Corollary 2 Let Ψ, Φ and Φ' be finite sets of formulas s.t. $\Psi \models \Phi'$ and $\Phi' \models \Phi$ hold. For every $\pi' \in TPI(\Psi, \Phi')$, there exists $\pi \in TPI(\Psi, \Phi)$ s.t. $\pi' \equiv_{\Phi'} \pi$ holds. Consequently, $nb\_cl(TPI(\Psi, \Phi')) \leq nb\_cl(TPI(\Psi, \Phi))$.

Finally, we have shown before that prime implicates can be defined from theory prime implicates. The next proposition shows that the converse holds as well.

Proposition 3 Let Ψ and Φ be finite sets of formulas. $TPI(\Psi, \Phi) = \min(PI(\Psi \cup \Phi), \models_\Phi)$.

3.3 Computing theory prime implicates

Let us now address the problem of computing theory prime implicates. The algorithm TPI below is adaptated from Quine's iterated consensus technique [Quine, 1952; Quine, 1955]; it basically consists in computing the implicates of $\Psi \cup \Phi$ using any resolution-based prime implicates algorithm (e.g. [Kean and Tsiknis, 1990; de Kleer, 1992]), while keeping clauses minimal w.r.t. Φ only (using any procedure for deciding Φ-logical entailment).

```
Function TPI(Ψ, Φ)
% Input  : two sets of formulas Ψ and Φ in CNF.
% Output : the theory prime implicates of Ψ
w.r.t. Φ.
1. Set TPI to Ψ∪Φ.
2. a/ Compute a new implicate C of Ψ∪Φ using
any resolution-based prime implicates algorithm.
   b/ If no new implicate is generated then
return(TPI).
   c/ If C is a Φ-logical consequence of some
clause in TPI then repeat this step ignoring C.
3. a/ Delete from TPI any clause which is a
Φ-logical consequence of C.
   b/ Add C to TPI.
4. Go back to step 2.
```

Clearly enough, the correctness of TPI w.r.t. theory prime implicates computation is a direct consequence of Propositions 1 and 3. Interestingly, TPI is independent of the resolution-based prime implicates procedure used in it; indeed, whatever the resolution-based prime implicates procedure under consideration is, the set of clauses computed by TPI will be *the same* (up to O-equivalence). This allows us to use refinements of Quines iterated consensus technique (in particular, this prevents one from generating redundant implicates thanks to Tison's technique [Tison, 1967] without sacrifying correctness). We can also improve TPI by preventing the implicates of o from being generated (step 2a/).

We developed two additional algorithms for computing theory prime implicates. The first one relies on a **property** relating the theory prime implicates of a disjunction of formulas to the theory prime implicates of its disjuncts (several prime implicates algorithms, including [Slagle *et al,* 1970; Jackson and Pais, 1990], rely on this property restricted to the prime implicates situation). The second one is based on theory resolution [Stickel, 1985]. Due to space limitations, we do not present these algorithms hereafter.

4 Knowledge compilation using theory prime implicates

4.1 Theory prime implicates compilations

Our approach to knowledge compilation basically consists (1) in pointing out tractable theories Φ that are implied by Σ and (2) in making Σ local using Φ while preserving logical equivalence. Formally, (2) is achieved thanks to the computation of the theory prime implicates of Σ w.r.t. Φ.

Definition 2 Let Σ, Φ be finite sets of formulas such that $\Sigma \models \Phi$ holds and Φ is tractable (i.e. clausal queries π can be answered in time polynomial in the size of $\Phi \cup \{\pi\}$). The *theory prime implicates compilation* of Σ w.r.t. Φ is defined by $COMP_\Phi(\Sigma) =_{def} \langle TPI(\Sigma, \Phi), \Phi \rangle$.

From a logical point of view, $COMP_\Phi(\Sigma)$ is equivalent to $TPI(\Sigma, \Phi) \cup \Phi$; however, we do not keep together $TPI(\Sigma, \Phi)$ and Φ since they do not play the same role w.r.t. query answering.

Our objective is to prove that $COMP_\Phi$ is actually an equivalence-preserving compilation function. On the one hand, showing that $COMP_\Phi$ is equivalence-preserving is easy (the next corollary results directly from Propositions 1 and 2).

Corollary 3 $COMP_\Phi(\Sigma) \equiv \Sigma$ holds.

On the other hand, we have to prove that query answering from $COMP_\Phi(\Sigma)$ can be done in time polynomial in the size of $COMP_\Phi(\Sigma)$. For this purpose, let us consider the function query-answering below:

```
Function query-answering(COMPΦ(Σ), π)
% Input: the theory prime implicates compilation
COMPΦ(Σ) of Σ w.r.t. Φ and a clause π.
% Output: true iff Σ⊨ π holds.
1. For every π' in TPI(Σ, Φ)
    If π' ⊨Φ π holds then return(true)
2. Return(false).
```

The correctness of query-answering w.r.t. query answering is a straightforward consequence of Propositions 1 and 2. Now, a way to check whether $\pi' \models_\Phi \pi$ holds consists in determining whether $\Phi \models (\neg l_i \vee \pi)$ holds for every literal l_i of π'. When Φ is tractable, this test can be done in time $O(|\pi'| * |\Phi \cup \{\pi\}|^\alpha)$, where α is a constant. Subsequently, query answering from a compilation can be performed in time polynomial in its size. More precisely:

Proposition 4 Let Σ be a finite set of formulas. Let Φ be a finite set of formulas s.t. $\Sigma \models \Phi$. If, for every clause π, checking whether $\Phi \models \pi$ holds can be done in time $O(|\Phi \cup \{\pi\}|^{\alpha})$ (where α is a constant), then `query-answering` runs in time $O(|TPI(\Sigma, \Phi)|*|\Phi \cup \{\pi\}|^{\alpha})$ in the worst case.

Thus, the crucial complexity factor of query answering from our compilations is the size of $TPI(\Sigma, \Phi)$ (provided that $|\Phi| = O(|\Sigma|^{\beta})$ where β is a constant).

4.2 Main characteristics of theory prime implicates compilations

Our approach to knowledge compilation possesses several interesting features. A strong point is that it is *model-theoretic.* In particular, the number of clauses of COMPo(L) does not depend on any particular ordering of symbols but on the *models* of o only. This is not true for many approaches to knowledge compilation, particularly FPIo [del Val, 1994] and E_d [Dechter and Rish, 1994]. We can also prove that FPIo and FPI2 compilations depend as well on a particular ordering of symbols; particularly, the way in which implicates are generated in FPIo and FPI2 may dramatically bear on the size of the resulting compilation (i.e. by an exponential factor).

Another strong point of our approach is that it is *generic* in essence. It may give rise to many compilations, taking advantage of many tractable classes of theories in propositional logic. For instance, the class of theories which are unit-refutation complete (including the Horn and the reverse Horn theories) and the class of theories consisting of binary clauses (Krom formulas) are polynomial for SAT (see [Jones and Laaser, 1977], [Cook, 1971]). It is easy to prove that they are also *stable by expansion with unit clauses* (i.e. if O belongs to such a class C then O u $\{l_1 \ldots l_n\}$ (where l_j are unit clauses) belongs to C). Consequently, all these classes are tractable. Additionally, the classes of theories *which result from any equivalence-preserving compilation technique* (including FPIo, FPI], FPI2, E_d compilations and all the techniques that remain to be discovered) are tractable (by definition). All these classes of theories can be used to generate theory prime implicates compilations. Incidentally, this shows that our approach can be considered complementary as well as supplementary to the existing (and future!) knowledge compilation techniques.

From a computational point of view, some additional advantages of our approach -that are not shared by many existing techniques- lie in the fact that theory prime implicates compilations are *local.* In particular, it is possible to enhance `query-answering` by checking in parallel whether $\pi' \models_{\Phi} \pi$ holds for every π' of $TPI(\Sigma, \Phi)$. Furthermore, when the procedure used for checking whether $\pi' \models_{\Phi} \pi$ holds consists in determining whether $l_j \models_{\Phi} \pi$ holds for every literal l_j of π', `query-answering` can be further improved by *caching*; caching simply consists in keeping track of the answer "yes" / "no" to the test $l \models_{\Phi} \pi$ for every literal l encountered so far. In this situation, the time complexity of `query-answering` becomes $O(|PS|*|\Phi \cup \{\pi\}|^{\alpha}+|TPI(\Sigma,\Phi)|*\log_2(|PS|))$ in the worst case. Interestingly, caching can be viewed as *an additional compilation* of $TPI(\Sigma, \Phi)$ [Moses and Tennenholtz, 1993]. Indeed, $\pi' \models_{\Phi} \pi$ holds iff $\neg\pi \models_{\Phi} \neg\pi'$ holds; the point is that the set of all the literals of PROPps is an *efficient basis* for the set of all the conjunctions of literals —ℜC* of PROPps (see [Moses and Tennenholtz, 1993] for details).

Another valuable consequence offered by locality is the possibility to *organize* the theory prime implicates of I w.r.t. o. Thus, attaching to each minimal implicate the frequency with which it is used to answer a query, we can order such implicates in a decreasing way. Since the ordering may vary dynamically, the organization of the compilation evolves with the queries and this may result in a an improvement of query answering with time. Interestingly, this is fully compatible with the common point of view stating that every piece of knowledge which is often used must be easy to remember. Clearly enough, such a behaviour is not guaranteed by the existing approaches to compilation (with [Reiter and de Kleer, 1987] as a notable exception) since they do not suggest any way to organize, the knowledge.

Interestingly, our compilation technique can also be shown *incremental* and *anytime.* In fact, the results pointed out in [del Val, 1994] can be easily adaptated to our framework (the structure of TPI is close to the structure of FPIo). Thus, provided that TPI uses an incremental prime implicate procedure inside, each time the knowledge base is revised (i.e. a new clause is added), recompiling the base from scratch is not mandatory. Note however that a small change in the knowledge base may result in a significant (and computationally expensive) change in the compilation (thus, from a computational point of view, incremental prime implicate algorithms do not behave better than non-incremental ones in the worst case [Kean and Tsiknis, 1990]). A further advantage of our compilation technique is that it is anytime; it means that the knowledge base can be queried before compiling ends up. To be more precise, TPI can be interrupted at any time and produce some useful intermediate results nevertheless: while time passes, the number of queries that can be answered successfully increases and when the compiling process ends, every query can be answered. As [del Val, 1994] underlines it, the interest of this "convergent approximation" view of compilation is that it has the potential to greatly decrease the inconvenience of using off-line computation.

Despite many advantages, our approach to knowledge compilation does not overcome the main impediment of existing compilation techniques: there is *no guarantee* that the size of our compilation is not *exponentially greater* than the size of the original knowledge base. For instance, let us consider the empty theory O =$_{def}$ {}. The empty theory is tractable but theory prime implicates compilations with empty theory reduce to prime implicates compilations (cf. § 3). Hence, their size can easily be exponential in I£I [Chandra and Markowsky, 1978; Kean and Tsiknis, 1990]. In this situation, query answering, while tractable w.r.t. COMP<j>(I), is still exponential in the size of I.

However, our approach does not behave computationally worse than its predecessors in this respect since, for all the compilation techniques pointed out so far, an exponential growth of the knowledge base may result from compiling. Moreover, due to the complexity considerations pointed out in [Kautz and Selman, 1992], it is unlikely that a given compilation technique definitely outperforms by an

exponential factor all its challengers *for every possible knowledge base*; indeed, a compilation technique s.t. compiling Σ results in a polynomial growth for every Σ is hardly to be expected.

Additionally, the size of $COMP_\Phi(\Sigma)$ can be advantageously compared with the size of $PI(\Sigma)$ when the size of Φ is small w.r.t. the size of $PI(\Sigma)$ (this condition is often satisfied when theories Φ are taken among subsets of the knowledge base Σ). Indeed, a direct consequence of Corollary 2 is that $nb\_cl(TPI(\Sigma, \Phi)) \leq nb\_cl(PI(\Sigma))$.

Finally, we will show in section 5 that our compilations can also beat their challengers by an exponential factor in some situations. Since $PI(\Sigma)$ is an instance of $TPI(\Sigma, \Phi)$, this shows incidentally that every tractable theory is not a good theory for the compilation purpose and that the choice of a "good" tractable theory may strongly bear on the efficiency of query answering from our compilations.

4.3 Pointing out "good", tractable theories

Since all the theory prime implicates compilations of I do not behave identically w.r.t. query answering from a computational point of view, a way to compare such compilations and a notion of optimal compilation of a knowledge base Σ must be defined. Intuitively, an optimal compilation is one that makes query answering the most efficient. However, we cannot define a unique criterion, significant and computable in practice, for characterizing the notion of optimal compilation in full generality.

In order to assess $COMP_\Phi(\Sigma)$ w.r.t. query answering in the worst case, a possible criterion c is the time complexity of 0 w.r.t. query answering per the size of $TPI(\Sigma, \Phi)$ Unfortunately, given a knowledge base Σ, we did not find a procedure to point out, in time polynomial in $|\Sigma|$, a tractable theory Φ leading to an optimal $COMP_\Phi(\Sigma)$ w.r.t. c (we suspect that there is no such polytime procedure).

Instead, we propose a time-bounded heuristic search strategy through the space of tractable theories. Thus, our strategy basically consists in pointing out a potentially good tractable theory Φ from Σ, compile Σ w.r.t. Φ, then retain $COMP_\Phi(\Sigma)$ if it is better than every compilation encountered so far, reject it otherwise. The search terminates when a satisfying compilation has been found or when the time devoted to search is wasted. Clearly enough, our strategy is fully compatible with del Val's view of compilation as a process of increasing the efficiency and the quality of query answering over the life cycle of the knowledge base [del Val, 1994].

Applying this strategy requires to deai with the size of the space of tractable theories. It may exist a large number of tractable theories which are logical consequences of a given knowledge base Σ (and we surely do not know all of them!). Consequently, we need some criteria to prune the search space and focus on promising candidates Φ only. To this end, we mainly consider the time complexity of Φ w.r.t. query answering and an estimation of the computational resources to be spent to point out Φ; in particular, tractable theories that cannot be put forward in time polynomial in $|\Sigma|$ are considered as a last resort. The logical strength of Φ is taken into account as an additional criterion (when it does not conflict with the two previous criteria). Indeed, in the light of Corollary 2, we know that it is sufficient to consider *logically strongest theories* only in order to minimize the number of clauses in the corresponding compilations. Since checking logical entailment is intractable in the general case, we focus in practice on computationally easier relationships between theories, particularly on set-inclusion.

Thus, given a knowledge base Σ, we will successively take into account as potentially good theories:

1/ The set Φ of all the Horn clauses of Σ,
2/ The set Φ of all the binary clauses of Σ,
3/ Subsets Φ of Σ such that the *tied chain graph* $G_T(\Phi)$ is acyclic (if so, Φ is unit-refutation complete [del Val, 1994]).
4/ Sets $Ed(\Phi)$ where Φ is a subset of Σ such that the *induced width* of 0 is lower or equal to 2 or the *induced diversity* of O is lower or equal to 1 (if so, Φ can be compiled under the form of a directional extension without an exponential growth [Dechter and Rish, 1994]).
5/ Sets $CF(\Phi)$ where CF is an equivalence-preserving compilation function (or CF is Horn_LUB [Selman and Kautz, 1991]) and Φ is a subset of Σ, provided that $|CF(\Phi)|$ does not exceed $|\Phi|$ from more than around one order of magnitude.

When several choices are possible (steps 3/ 4/ 5/), we prefer maximal sets Φ (w.r.t. set-inclusion) if we have a way to point them out in polynomial time in $|\Sigma|$[1]. Otherwise, we try to approximate them. If it is not possible, we pick up randomly any possible choice.

To restrict the search space further, an additional heuristic can be used; it consists in setting aside every theory Φ s.t. $nb\_cl(\Phi) / nb\_cl(\Sigma)$ is lower than a given threshold. This may prevent us from taking into account unpromising theories (e.g. if there is no Horn clause in Σ, we jump directly to the set of all the binary clauses of Σ and we do not generate $TPI(\Sigma, \{\}) = PI(\Sigma)$).

Clearly enough, our strategy is very inefficient in the general case since an experimental *post-hoc* evaluation is necessary to check the choices that have been made. However, existing compilation techniques do not behave better here: in the general case, *there is no way to estimate the size of a compilation before computing it*[2].

5 A comparison with some other approaches to knowledge compilation

Before comparing our approach with some existing techniques, we must define the way in which the comparison will be done. As a matter of fact, a crucial complexity factor of PI, FPI(), FPI_0, FPI_2 and E_d compilations w.r.t. query answering is the size of the compilation (the number of

[1] This is not possible in the general case. For instance, determining maximal subsets of Σ that do not contain a tied-chain by focusing on maximal acyclic subgraphs of $G_T(\Sigma)$ cannot be done in time polynomial in $|\Sigma|$ in the general case since this last problem is NP-complete.

[2] This is not exact for the approach of [Dechter and Rish, 1994] since the size of a directional extension is related to its induced width and to its induced diversity. These factors can be determined without computing the compilation but this does not help so much since determining a best ordering -i.e. one leading to a smallest induced width or a smallest induced diversity- is not tractable in the general case.

clauses is equally significant when k-CNF knowledge bases are considered, cf. § 2).

Since this is true for $COMP_\Phi$ as well, we will compare our approach with PI, FPI_0, FPI_1, FPI_2 and E_d focusing on the number of clauses in the compilations.

Because Σ does not fully characterize $FPI_0(\Sigma)$, $FPI_1(\Sigma)$, $FPI_2(\Sigma)$ and $E_d(\Sigma)$ in the general case, we must choose some FPI_0, FPI_1, FPI_2 and E_d compilations among the possible ones; to be fair, we will compare our compilations *with the best possible choices* (i.e. FPI_0, FPI_1, FPI_2 and E_d compilations containing a minimal number of clauses).

5.1 Some examples for which our technique proves helpful

Our approach clearly outperforms any equivalence-preserving compilation function CF (including PI, FPI_0, FPI_1, FPI_2 and E_d) whenever $\Sigma = \Phi \cup \{c\}$, $|CF(\Sigma)|$ is exponentially greater than $|\Sigma|$ and $|CF(\Phi)|$ is polynomial in $|\Phi|$. Indeed, in such a situation, choosing $CF(\Phi)$ as a theory leads to the compilation $COMP_\Phi(\Sigma) = \langle\{c\}, CF(\Phi)\rangle$ and $|COMP_\Phi(\Sigma)|$ is polynomial in $|\Sigma|$ while $|CF(\Sigma)|$ is exponentially greater than $|\Sigma|$. In particular, when Σ is not known as tractable but Φ is, our approach proves valuable.

Example 2 (adaptated from [Kean and Tsiknis, 1990])

$$\Sigma =_{def} \{\neg p_1 \vee \neg q_{11} \vee r, \ldots, \neg p_1 \vee \neg q_{1m} \vee r,$$
$$\ldots,$$
$$\neg p_k \vee \neg q_{k1} \vee r, \ldots, \neg p_k \vee \neg q_{km} \vee r,$$
$$p_1 \vee p_2 \vee \ldots \vee p_k \vee r\}.$$

With $\Phi =_{def} \Sigma \setminus \{p_1 \vee p_2 \vee \ldots \vee p_k \vee r\}$, we obtain:

$nb\_cl(COMP_\Phi(\Sigma)) = 1 + m^*k.$
$nb\_cl(FPI_1(\Sigma)) = 1 + m^*k.$
$nb\_cl(E_d(\Sigma)) = 1 + m^*k.$
$nb\_cl(PI(\Sigma)) = (m+1)^k + m^*k.$
$nb\_cl(FPI_0(\Sigma)) = (m+1)^k + m^*k.$
$nb\_cl(FPI_2(\Sigma)) = (m+1)^k + m^*k.$

In this example, $COMP_\Phi$ beats PI, FPI_0 and FPI_2 by an exponential factor. If we substitute each p_i with its negation and *vice-versa*, then the resulting Φ is unit-refutation complete (there are no tied literals in it). The results above still hold.

5.2 Preliminary experimental results

As a first step towards a comparison, we performed some preliminary experiments. We implemented our compilation technique and tested it on k-CNF knowledge bases Σ drawn from diagnosis and qualitative physics. These knowledge bases are taken from [Forbus and de Kleer, 1993], allowing us to compare our results with PI and FPI_1 compilations. For each problem under consideration, the theory Φ we have first considered is the subset of all the Horn clauses of Σ (it can easily be found in time linear in the size of Σ).

Table 1 gives the number of clauses of the initial knowledge base, the number of clauses of the corresponding prime implicates knowledge base, the numbers of clauses of the FPI_1 compilation of Σ, and finally the number of clauses of the theory prime implicates compilation of Σ w.r.t. Φ. $FPI_1(\Sigma)$ corresponds to the compilations associated with the orderings of symbols which gave an optimal $FPI_1(\Sigma)$ in each case [del Val, 1994].

Problem	Σ	$PI(\Sigma)$	$FPI_1(\Sigma)$	Φ	$TPI(\Sigma, \Phi)$
2 pipes	54	638	173	*49*	*48*
3 pipes	82	2360	545	*75*	*135*
4 pipes	110	6208	1328	*101*	*298*
Regulator	106	2814	829	*99*	*555*
Adder	50	9700	294	*42*	*706*

Table 1

Table 1 clearly shows that our approach can result in huge savings in the growth of the knowledge base by comparison with the prime implicates compilation. The savings we obtain w.r.t. FPI_1 are often less significant. We do not know the numbers of clauses of the optimal FPIfj compilations associated with these knowledge bases. However, the knowledge base returned by FPI_0 never is smaller than the knowledge base returned by FPI_1 when the same prime implicate algorithm is used inside both procedures. Hence, the savings obtained w.r.t. any optimal FPI_0 are at least as significant as those obtained w.r.t. the corresponding optimal FPI_1 and they can be exponentially larger.

In contrast to our compilation, the performances of FPI_0 and FPI_1 depend on the choice of a good ordering over the propositional symbols and, as [del Val, 1994] points out, we only have a very limited understanding of how to obtain such good orderings (hence, to find optimal FPIfj and FPI_1 compilations). Conversely, the performance of our approach depends on the existence and the choice of a good theory, i.e. a tractable theory such that the absolute growth of the corresponding compilation is reasonable (e.g. no more than around an order of magnitude). As shown in Table 1, this was the case for the theories used in our experiments.

Interestingly, using other theories can lead to still better savings in the growth of the knowledge base. We performed a few additional experiments to this end, focusing on the adder problem (see [Kean and Tsiknis, 1992] for its specification). In the first place, we have considered the set composed of the 16 last clauses of the specification (the "negation laws"). Since this set is composed of binary clauses only, it can be used as a tractable theory. The corresponding set TPI(Z, O) contains 698 clauses. This is a slight improvement w.r.t. the number of clauses obtained by using all the Horn clauses of the specification as a theory. We have also focused on the set composed of the 34 last clauses in the specification (all the clauses of the specification except the description of the behaviour of the two exclusive-or gates). In this set, every merge resolvent is subsumed by a clause of the original set. Accordingly, this set is identical to its FPIfj compilation, hence it is unit-refutation complete [del Val, 1994]. Using it as a theory, we obtained a set TPI(E, O) composed of 112 clauses only. Once more, this example shows that our own compilation technique can take advantage of the derivation of a tractable theory thanks to any other approach to compilation.

6 Conclusion

In this paper, we have been concerned with equivalence-preserving logical compilation of propositional knowledge bases. The main contribution of this paper is a new approach to compilation, based on the concept of theory prime implicates, a generalization of the standard notion of prime implicates. Our approach can be viewed as an additional element in the panoply of techniques designed to deal with intractability of propositional query answering. Interestingly, our approach can also be viewed as complementary to its predecessors. Analytically and experimentally, we have shown that theory prime implicates compilation can achieve substantial space savings w.r.t. some other compilation techniques.

This work must be extended in several directions. A first direction concerns the empirical validation of the approach. More experiments are clearly needed to assess its practical applicability. A second direction is related to the choice of a good theory. We have only a very limited understanding of how to point out such good theories. Accordingly, the naive strategy we have proposed must be improved. More generally, the key problem of predicting whether a given compilation technique is well-suited to a given knowledge base should be addressed in the future.

Acknowledgements

The author would like to thank Alvaro del Val for several fruitful discussions and the anonymous reviewers for their helpful comments.

References

[Cadoli and Schaerf, 1991] M. Cadoli and M. Schaerf. Approximate entailment. In *Proceedings of the Second Conference of the Italian Association for Artificial Intelligence,* pages 68-77, Palermo, 1991.

[Chandra and Markowsky, 1978] A.K. Chandra and G. Markowsky. On the number of prime implicants. ***Discrete Mathematics,* 24:7-11, 1978.**

[Cook, 1971] S.A. Cook. The complexity of theorem-**proving procedures. In *Proceedings of the Third Annual ACM Symposium on the Theory of Computing,* pages** 151-158, 1971.

[Dechter and Rish, 1994] R. Dechter and I. Rish. Directional resolution: the Davis-Putnam procedure, revisited. In ***Proceedings of the Fourth International Conference on Knowledge Representation and Reasoning,* pages** 134-145, Bonn, 1994.

[de Kleer, 1992] J. de Kleer. An improved incremental algorithm for generating prime implicates. In ***Proceedings of the Tenth National Conference on** Artificial Intelligence,* pages 780-785, San Jose (CA), 1992.

[del Val, 1994] A. del Val. Tractable databases: how to make propositional unit resolution complete through **compilation. In *Proceedings of the Fourth International Conference on Knowledge Representation and Reasoning,*** pages 551-561, Bonn, 1994.

[Forbus and de Kleer, 1993] K. Forbus and J. de Kleer. *Building Problem Solvers.* **MIT** Press, Cambridge (MA), 1993.

[Jackson and Pais, 1990] P. Jackson and J. Pais. **Computing prime implicants. In *Proceedings of the Tenth International Conference on Automated Deduction,* pages** 543-557, Kaiserslautern, 1990.

[Jones and Laaser, 1977]. N.D. Jones and W.T. Laaser. Complete problems for determininistic polynomial time. ***Theoretical Computer Science,* 3:105-117, 1977.**

[Kautz and Selman, 1992] H. Kautz and B. Selman. Forming concepts for fast inference. In *Proceedings of **the Tenth National Conference on Artificial Intelligence,*** pages 786-793, San Jose (CA), 1992.

[Kautz and Selman, 1994] H. Kautz and B. Selman. An empirical evaluation of knowledge compilation by theory **approximation. In *Proceedings of the Twelfth National Conference on Artificial Intelligence,* pages 155-161,** Seattle (WA), 1994.

[Kean and Tsiknis, 1990] A. Kean and G. Tsiknis. An incremental method for generating prime implicants/implicates. *Journal of Symbolic Computation,* 9:185-206, 1990.

[Kean and Tsiknis, 1992] A. Kean and G. Tsiknis. Assumption-based reasoning and clause management **systems. *Computational Intelligence,* 8:1-24, 1992.**

[Marquis, 1995] P. Marquis. A logical framework for knowledge compilation based on minimal implicates. In preparation, 1995.

[Mathieu and Delahaye, 1994] Ph. Mathieu and J.-P. Delahaye. A kind of logical compilation for **knowledge bases. *Theoretical Computer Science,*** 131:197-218, 1994.

[Moses and Tennenholtz, 1993] Y. Moses and M. Tennenholtz. Off-line reasoning for on-line efficiency. In ***Proceedings of the Thirteenth International Joint Conference on Artificial Intelligence,* pages 490-495,** Chambery, 1993.

[Quine, 1952] W.V.O. Quine. The problem of simplifying **truth functions. *American Mathematical Monthly,*** 59: 521-531, 1952.

[Quine, 1955] W.V.O. Quine. A way to simplify truth **functions. *American Mathematical Monthly,*** 62:627-631, 1955.

[Reiter and de Kleer, 1987] R. Reiter and J. de Kleer. Foundations of assumption-based truth maintenance systems: preliminary report. In *Proceedings of the Sixth **National Conference on Artificial Intelligence,* pages** 183-188, Seattle (WA), 1987.

[Selman and Kautz, 1991] B. Selman and H. Kautz. Knowledge compilation using Horn approximations. In ***Proceedings of the Ninth National Conference on** Artificial Intelligence,* pages 904-909, Anaheim, 1991.

[Slagle *et ai,* 1970] J.R. Slagle, C.L. Chang and R.C.T. Lee. A new algorithm for generating prime **implicants. *IEEE Transactions on Computers,*** C-19(4):304-310, 1970.

[Stickel, 1985] M.E. Stickel. Automated deduction by **theory resolution. *Journal of Automated Reasoning,*** 1:333-355, 1985.

[Tison, 1967] P. Tison. Generalization of consensus theory and application to the minimization of boolean **functions. *IEEE Transactions on Electronic Computers,*** EC-16(4): 446-456, 1967.