Algorithms for propositional KB approximation

Yacine Boufkhad

CRIL Université d'Artois
Rue de l'université SP-16 62307
Lens cedex 03 FRANCE
email: boufkhad@cril.univ-artois.fr

Abstract

One of the obstacles to the effective compilation of propositional knowledge bases (KBs) using Horn approximations, as introduced by (Selman & Kautz 1991), is the lack of computationally feasible methods for generating Horn bounds. In this paper new algorithms for generating Horn Greatest Lower Bounds (GLB) that can apply to large size KBs, are presented. The approach is extended through a more general target language: the renamable Horn class. The conditions under which a renamable Horn formula is a renamable Horn GLB of a KB are established and algorithms for computing it are derived. These algorithms can be used in the other approaches based on computation of Horn or renamable lower bounds as (Boufkhad *et al.* 1997). The efficiency of these algorithms and the tightness with respect to the KB in terms of number of models of the bounds, are experimentally evaluated. The renamable Horn GLB proves to be closer to the KB than the Horn GLB.

Introduction

Given a satisfiable Knowledge Base (in short; KB) Σ for which the propositional deduction is intractable, a knowledge compilation approach consists in transforming Σ into one or several tractable formulae so that the subsequent queries are answered efficiently from the tractable formulae. Many approaches to knowledge compilation have been proposed (Reiter & De Kleer 1987; Selman & Kautz 1991; del Val 1994; Dechter & Rish 1994; Marquis 1995). In particular, (Selman & Kautz 1991; 1996) have proposed to approximate a KB through a greatest lower bound (GLB), which is the weakest Horn theory that entails Σ, and a lowest upper bound (LUB) which is the strongest Horn theory entailed by Σ. Answering queries from a GLB of Σ (Σ_{glb}) and the LUB (Σ_{lub}) is done in this way: for a clause c, if $\Sigma_{glb} \not\models c$ then $\Sigma \not\models c$ and if $\Sigma_{lub} \models c$ then $\Sigma \models c$ otherwise, the answer is "don't know". The nice feature is that queries $\Sigma_{glb} \models^? c$ and $\Sigma_{lub} \models^? c$ can be answered in linear time since Σ_{glb} and Σ_{lub} are Horn (Dowling & Gallier 1984). The size, the quality and algorithms for generating LUB have been studied in (Cadoli 1993; del Val 1995; Selman & Kautz 1996). It has been shown that the size of the LUB is in general exponential w.r.t. the size of the KB. On the other hand, the size of a GLB is always smaller than the size of the original formula (Selman & Kautz 1996) and the complexity of the problem of generating a GLB is in P^{NP} (Cadoli 1993).

An other approach to compilation using tractable lower bounds is proposed in (Boufkhad *et al.* 1997). It consists in transforming a KB into an equivalent disjunction of tractable formulae. Each formula being a lower bound of the KB falling into a list of target tractable classes.

Although the compilation step is viewed as an off-line process for which time is not a critical parameter, finding comptationally feasible algorithms allowing large size KBs to be handled remains an important open issue. In this paper, new algorithms for generating Horn and renamable Horn GLBs are introduced. The basic algorithm for generating a Horn GLB is based on the Davis and Putnam procedure (Davis & Putnam 1960). When the KBs are too large for a systematic search to be considered, an incomplete method is used to preprocess the KB and make it easier for compilation. Using this method, Horn GLBs of large KBs can be actually computed.

As an even more important contribution, GLBs are then investigated within the more general class of renamable Horn. The conditions under which a renamable Horn formula is a renamable Horn GLB ($\mathcal{RH}$-GLB) of a KB Σ are established. Then an algorithm for generating a $\mathcal{RH}$-GLB is provided. Finally, the tightness of these bounds is explored experimentally with respect to the KB, in terms of the number of models. As can be expected from the generality of the renamable Horn class the $\mathcal{RH}$-GLB proves to be a tighter bound than the Horn GLB.

The above procedures can also be used to compute more efficiently the tractable covers of the KB as defined in

(Boufkhad *et al.* 1997).

In this paper, interpretations and models are represented by the sets of literals they satisfy. Implicants and prime implicants are defined in the usual way. The KB to be compiled is a set Σ of clauses (in CNF). The KB Σ is assumed to be satisfiable.

This paper[1] is organized in three sections. In the first one, a new algorithm for generating Horn GLB is introduced and some improvements w.r.t. the processing of large KBs described. The second section is concerned with the renamable Horn case and the third one presents the experimental evaluation of algorithms and the comparative evaluation of the quality of the two types of GLBs.

Generating Horn GLBs

The basic algorithm

Let us give some useful preliminary definitions. A clause is Horn iff it contains at most one positive literal. A set of clauses is Horn if every clause is Horn. A Horn GLB of a Knowledge base Σ is a Horn formula Σ_{glb} s.t. there is no other Horn formula H s.t. $\Sigma_{glb} \not\equiv H$ and $\Sigma_{glb} \models H \models \Sigma$.

First, Selman and Kautz's approach to compute Horn GLB is based on the *Horn strengthening* concept. A Horn strengthening of a clause C is a clause obtained by removing all but one positive literal of C. Example: the clause $\neg a \vee \neg b \vee c \vee d \vee e$. The Horn strengthenings of this clause are $\neg a \vee \neg b \vee c$, $\neg a \vee \neg b \vee d$ and $\neg a \vee \neg b \vee e$. A Horn strengthening of a set of clauses Σ is a set of Horn strengthenings of clauses of Σ. For every Horn GLB Σ_{glb} of Σ there exists a Horn strengthening H s.t. $H \equiv \Sigma_{glb}$ (Selman & Kautz 1996). The difficulty of finding a Horn GLB comes from the possibly large number of Horn strengthenings of a set of clauses.

To overcome this drawback, implicants of Σ will be used to strengthen Σ so that the search of a GLB of Σ collapses into the search of a GLB of this strengthened Σ. To this end, a concept of *I-Strengthening* is introduced, which itself is based on the Horn strengthening. For a partial interpretation I and a clause C satisfied by I, the I-Strengthening of C is defined as the clause obtained by removing from C every positive literal l s.t. $I \not\models C^H$ where C^H is the Horn strengthening of C containing l. The I-Strengthening of a set of clauses Σ entailed by I is the formula formed by the I-Strengthening of its clauses. Example: the same clause in the previous example is taken. For the partial interpretation $I = \{a, b, \neg d, c\}$ the I-strengthening of C is $\neg a \vee \neg b \vee c$ since I does not satisfy the second and the third Horn strengthening of C (see previous example). For $I' = \{\neg a, b, \neg d\}$ the I'-strengthening of C is C itself since I' satisfies every Horn strengthening of C.

[1]A long version of this paper, including proofs, is available.

Accordingly, any Horn GLB of Σ can be computed as a Horn GLB of the I-strengthening of Σ, provided that I is an implicant of Σ that entails this GLB. Indeed:

Proposition 1 *Let Σ be a set of clauses, I an implicant of Σ and Γ a Horn GLB of Σ. $I \models \Gamma$ iff Γ is a Horn GLB of the I-strengthening of Σ.*

The idea of the generation procedure of a Horn GLB consists in searching for implicants of Γ, after initializing Γ with Σ. Each time an implicant is found, Γ is replaced by its I-strenghtning. Consequently the remaining part of the search is restricted to the Horn Strengthenings that are entailed by I. This process is repeated on Γ until Γ is Horn or all its implicants have been visited. In the latter case, all the Horn strengthenings of Γ are equivalent.

Let us now describe the procedure that generates a Horn GLB of a set Σ of clauses. A systematic search of implicants is done using the DP procedure. DP procedure has already been used in compilation (Schrag 1996; Boufkhad *et al.* 1997). As an important difference with DP, the formula for which DP searches for implicants is not the same all along the tree. Every time an implicant I is found, the current formula is replaced by its I-strengthening.

```
GLB($\Sigma$)
input: a set of clauses $\Sigma = \{C_1, C_2, ..., C_p\}$
output: a Horn GLB.
begin
  $\Gamma \leftarrow$ DP_GLB($\Sigma$,$\emptyset$);
  return any Horn-strengthenings of $\Gamma$;
end

DP_GLB($\Gamma$,$I$)
input: a set of clauses $\Gamma$ and a partial interpretation $I$
output: a strengthening of $\Gamma$.
begin
  if unit propagation shows $\Gamma \wedge I$ is inconsistent
    then return $\Gamma$;
  if ($I \models \Gamma$) then return I-Strengthening of $\Gamma$;
  $l \leftarrow$ ChooseLiteral($\Gamma \wedge I$);
  $\Gamma \leftarrow$ DP_GLB($\Gamma$,$I \cup \{l\}$);
  if $\Gamma$ is Horn then return $\Gamma$;
  $\Gamma \leftarrow$ DP_GLB($\Gamma$,$I \cup \{\neg l\}$);
  return $\Gamma$;
end
```

Theorem 1 *Let Σ be a set of clauses,* **GLB**(Σ) *delivers a Horn GLB of Σ.*

Interestingly enough, **GLB** is an anytime procedure in the sense that if interrupted, it returns a Horn strengthening of Γ which is a Horn LB that includes at least the implicants encountered so far by **DP_GLB**. An other advantage of this procedure is that it allows to choose which models to include in the Horn GLB to be generated.

Let us now compare the efficiency of GLB with the original algorithm **Generate_GLB** by Selman and Kautz. **Generate_GLB** enumerates all possible Horn strengthenings of the set of clauses and chooses the weakest one. The number N of iteration steps needed by **Generate_GLB** is thus the number of Horn strengthenings of a formula, which depends on the numbers of non Horn clauses and of positive literals in these clauses. This is, let $|P(C_i)|$ be the number of positive literals in the clause C_i: $N = \prod_{C_i \in \Sigma} |P(C_i)|$. In general, this number is so huge that this algorithm is actually intractable. Conversely, when the number of non Horn clauses is very small **Generate_GLB** might behave better than **GLB**. An extreme example requires the presence of only one non Horn clause with 2 positive literals and a large number of Horn clauses with an exponential number of models: **Generate_GLB** needs only two steps to decide which Horn strengthening to choose, while **GLB** needs a large amount of time to find a GLB. Because this situation can happen in the course of execution of **GLB**, the number of steps needed by **Generate_GLB** to give a Horn GLB of the current formula Γ is computed before every recursive call to **DP_GLB**. If this number is sufficiently small, **DP_GLB** is interrupted and switches to **Generate_GLB** to finish the computation process.

Preprocessing large formulae

In **DP_GLB**, each time an implicant is found, it strengthens the formula Γ making it more constrained and thus easier for the remaining search process. When the formula is too large, even finding just one model is too hard for this method. To overcome this drawback, a logically incomplete but efficient procedure is used for finding a model and strengthening the formula until it becomes sufficiently constrained for being processed by **DP_GLB**. Generating Horn GLB of large size KBs will consist in looping over the following 3 steps until the formula is sufficiently constrained to be easy for procedure **GLB**. Γ is initialized with the KB Σ :

1. Use a logically incomplete procedure (e.g. a local search one) for finding a model I of Γ (in our experimentations we used the local search method (Mazure, Saïs, & Grégoire 1997)).
2. Drop literals from I successively until I is a prime implicant of Γ.
3. Replace Γ by its I-strengthening.

Many criteria can be used to decide whether a formula is sufficiently constrained for being easy enough for **GLB**. In our experimentations the number of unit and binary clauses in the formula has been chosen. After this preprocessing **GLB** is called with the resulting formula as an input.

Generating Renamable Horn GLBs

The general framework for theory approximation defined in (Selman & Kautz 1996) leaves freedom for choice of representation languages other than Horn. The renamable Horn class (Lewis 1978; Henschen & Wos 1974) is one of these possible target languages. Let us stress that it includes the Horn, reverse Horn and satisfiable binary classes.

Renaming a variable x in a set of clauses Σ consists in replacing every occurrence of x by $\neg x$ and vice versa. A renaming function w.r.t an interpretation r, denoted by $\mathcal{R}_r$, maps the formula Σ into an other formula $\mathcal{R}_r(\Sigma)$ obtained by renaming in Σ every variable having its positive literal in r. When used in a renaming function, the interpretation r is said to be a renaming. It can be noted that $\mathcal{R}_r(\mathcal{R}_r(\Sigma)) = \Sigma$. Example: $\Sigma = \{x \vee \neg y, y \vee z\}$ and $r = \{\neg x, y, z\}$ then $\mathcal{R}_r(\Sigma) = \{x \vee y, \neg y \vee \neg z\}$. A set of clauses Σ is renamable Horn if there exists a renaming r s.t. $\mathcal{R}_r(\Sigma)$ is Horn, in this case r is said to be a Horn renaming of Σ. Testing whether a CNF formula Σ is renamable Horn can be done in time linear in the size of Σ (Aspvall 1980).

A $\mathcal{RH}$-GLB of a set of clauses Σ is a renamable Horn formula Σ_{rh-glb} s.t. $\Sigma_{rh-glb} \models \Sigma$ and there is no other renamable Horn formula H s.t. $H \not\equiv \Sigma_{rh-glb}$ and $\Sigma_{rh-glb} \models H \models \Sigma$. There is a close connection between Horn and renamable Horn LBs. Indeed, considering a set of clause Σ and a renaming r, every horn LB of $\mathcal{R}_r(\Sigma)$ is a renamable Horn LB of Σ. This is why, in the following, there is a need to use the procedure **GLB** to generate a renamable Horn GLB.

Due to the larger scope of the renamable Horn class, generating bounds should be more difficult and expensive than in the Horn case. However, tighter bounds could be expected.

The algorithm for computing a $\mathcal{RH}$-GLB can be summarized in this way: The formula Λ is initialized to $false$. Then the renamings r are successively considered: if there exists H a Horn GLB of $\mathcal{R}_r(\Sigma)$ that improves Λ (i.e. $\Lambda \models \mathcal{R}_r(H)$) then Λ is replaced by $\mathcal{R}_r(H)$.

By these successive improvements of Λ, after testing all the renamings, we are sure that there is no better renamable Horn LB than Λ.

Since for a set of propositional variables V there are $2^{|V|}$ possible renamings, and for each renaming r there is in general an exponential number of Horn GLBs of $\mathcal{R}_r(\Sigma)$, the search space of $\mathcal{RH}$-GLBs of Σ is huge. It is then essential to restrict this search space in order to allow the generation of a $\mathcal{RH}$-GLB to be achieved in reasonable time. The following results established here, give the means for dramatically reducing this search:

- It is sufficient to consider only the renamings that are models of Σ.
- It can be tested in polynomial time, whether renaming w.r.t. a model m can give a better bound than the current one.
- When a model m verifies the latter condition, it is sufficient to generate only one Horn GLB H of $\mathcal{R}_m(\Sigma)$ provided that the current bound entails $\mathcal{R}_m(H)$.

Thanks to the next proposition, the first point of the previous list is established:

Proposition 2 *Let Σ be a set of clauses and K a satifiable renamable Horn LB of Σ. There exists a model m of Σ and a renamable Horn Lower bound Λ of Σ having m as a Horn renaming s.t. $K \equiv \Lambda$.*

If K of the previous proposition has a model among its Horn renamings then the statement is trivial. Otherwise, the formula Λ is obtained by removing from K every opposite occurrence of the literals satisfied by a unit propagation performed on Λ.

The preceding proposition stated for any renamable Horn LB, holds in particular for a $\mathcal{RH}$-GLB. Then every $\mathcal{RH}$-GLB is equivalent to a renamable Horn formula that has a model among its Horn renamings. Thus, in searching for a $\mathcal{RH}$-GLB of Σ it is sufficient to consider only the models of Σ as renamings.

Let us consider a model m for renaming, if H is a Horn LB of $\mathcal{R}_m(\Sigma)$ then $\mathcal{R}_m(H)$ is a renamable Horn LB of Σ. Consequently, only Horn GLBs of $\mathcal{R}_m(\Sigma)$ can correspond to renamable Horn GLBs of Σ. Indeed:

Proposition 3 *Let Σ be a set of clauses and Σ_{rh-glb} one of its renamable Horn GLBs. For every Horn renaming r of Σ_{rh-glb} , $\mathcal{R}_r(\Sigma_{rh-glb})$ is a Horn GLB of $\mathcal{R}_r(\Sigma)$.*

Let us now consider a renamable Horn LB K. Checking whether renaming w.r.t some model m improves K (i.e. whether there is a Horn GLB Γ of $\mathcal{R}_m(\Sigma)$ s.t. $K \models \mathcal{R}_m(\Gamma)$) could require to generate every Horn GLB Γ of $\mathcal{R}_m(\Sigma)$ and to check if $K \models \mathcal{R}_m(\Gamma)$. This is avoided by the next proposition that establishes the conditions under which, renaming w.r.t a model m can improve K. Since a Horn GLB of $\mathcal{R}_m(\Sigma)$ is equivalent to a Horn Strengthening (Selman & Kautz 1996) then K must entail at least one Horn strengthening of $\mathcal{R}_m(\Sigma)$.

Proposition 4 *Let Σ be a set of clauses, m one of its models and K a renamable Horn LB of Σ. There exists a Horn GLB Γ of $\mathcal{R}_m(\Sigma)$ s.t. $K \models \mathcal{R}_m(\Gamma)$ iff for every clause C of $\mathcal{R}_m(\Sigma)$ there is a Horn Strengthening C^H of C s.t. $\mathcal{R}_m(K) \models C^H$.*

Interestingly, since K is renamable Horn, the above condition can be checked in linear time in the size of K and C.

When the conditions defined in the proposition 4 are fulfilled by a model m, the procedure **GLB** defined in the previous section can be used to generate a Horn GLB Λ of $\mathcal{R}_m(\Sigma)$ s.t. $\mathcal{R}_m(K) \models \Lambda$. To ensure that $\mathcal{R}_m(K) \models \Lambda$, the parameter given to **GLB** is a strengthening of $\mathcal{R}_m(\Sigma)$ similar to the I-strengthening defined in the previous section. For this, the definition of the I-strengthening is generalized in this way: for a formula F, the F-strengthening of a clause C is the clause obtained from C by removing every positive literal l s.t. $F \not\models C^H$ where C^H is the Horn strengthening of C containing l. Clearly, and for the same reasons that in proposition 1, every Horn GLB of the $\mathcal{R}_m(K)$-strengthening of $\mathcal{R}_m(\Sigma)$ is entailed by $\mathcal{R}_m(K)$.

To summarize the above results, let us give a partial description of the algorithm for computing a $\mathcal{RH}$-GLB. Let M denotes the set of the models of Σ. The formula Λ is initialized to $false$.

```
for every model $m$ in $M$
  if $\mathcal{R}_m(\Lambda)$ entails some Horn strengthening of $\mathcal{R}_m(\Sigma)$
  then
    $\Sigma' \leftarrow \mathcal{R}_m(\Lambda)$-strengthening of $\mathcal{R}_m(\Sigma)$;
    $\Lambda' \leftarrow$ GLB($\Sigma'$);
    $\Lambda \leftarrow \mathcal{R}_m(\Lambda)$
```

It is assumed in the partial procedure given above, that the set of models of Σ has been computed. In our implementation, a Davis and Putnam procedure is used to sytematically search for models. As models are found the above partial procedure is used to improve the current bound.

By generalizing the condition given in proposition 4, it is possible to prune the DP search tree at a node where the current partial interpretation cannot improve the current bound through one of its models (if any). Let us consider a partial interpretation I and a renamable Horn LB K, we address the question: Is it possible to extend I to a renaming that improves K? A small modification to proposition 4 allows, in some cases, to answer efficiently negatively this question. Let us consider a clause C and rename it using the renaming function $\mathcal{R}'_I$ defined in this way: for every literal l in C if l or $\neg l$ is in I then rename it the usual way, otherwise rename it negatively. Let us denote by C^H the clause s.t. $\mathcal{R}'_I(C^H)$ is a Horn strengthening of $\mathcal{R}'_I(C)$. Example: $C = \neg a \vee \neg b \vee c \vee d$ and $I = \{a, \neg b, \neg c\}$ then $\mathcal{R}'_I(C) = a \vee \neg b \vee c \vee \neg d$, then $C^H = \neg b \vee c \vee d$ is s.t. $\mathcal{R}'_I(C^H)$ is a Horn strengthening of $\mathcal{R}'_I(C)$. Clearly, if there is no Horn strengthening $\mathcal{R}'_I(C^H)$ s.t. $K \models C^H$ then the answer to the above question is no. This is because no full interpretation obtained from I, can rename negatively more literals than does I using the renaming function described above. This method allows us to prune the search by skipping all the models (if any) in the considered partial interpretation.

Let us now describe the complete procedure that generates a $\mathcal{RH}$-GLB of a set Σ of clauses. Similarly to **GLB**, **RH_GLB** performs a single call to a procedure **DP_RH_GLB** that develops a DP like tree. A formula Λ is initialized to $false$ and then is improved everytime a model of Σ that can improve it is found. At each step, **DP_RH_GLB** tests if the current partial interpretation can produce a renaming that improves Λ, in the negative case **DP_RH_GLB** backtracks allowing for a significant pruning (in the next procedures, Λ is a global variable).

```
RH_GLB($\Sigma$)
input: a set of clauses $\Sigma = \{C_1, C_2, ..., C_p\}$
output: a $\mathcal{RH}$-GLB.
begin
  $\Lambda \leftarrow false$;
  DP_RH_GLB($\Sigma$,$\emptyset$);
  return $\Lambda$;
end.

DP_RH_GLB($\Sigma$,$I$);
input: a formula $\Sigma$ and a partial interpretation $I$
output: none.
begin
  if unit propagation shows that $\Sigma \wedge I$ is inconsistent
    then return;
  for every clause $C$
    if there is no Horn strengthening $\mathcal{R}'_I(C^H)$
     of $\mathcal{R}'_I(C)$ s.t. $\Lambda \models C^H$ then return;
  if $I$ is a model of $\Sigma$
    then
     $\Sigma' \leftarrow \mathcal{R}_I(\Lambda)$-strengthening of $\mathcal{R}_I(\Sigma)$;
     $\Lambda' \leftarrow$ GLB($\Sigma'$);
     $\Lambda \leftarrow \mathcal{R}_I(\Lambda')$;
     return;
  $l \leftarrow$ ChooseLiteral;
  DP_RH_GLB($\Sigma$,$I \cup \{l\}$);
  DP_RH_GLB($\Sigma$,$I \cup \{\neg l\}$);
end.
```

Theorem 2 *Let Σ be a set of clauses,* **RH_GLB**(Σ) *delivers a renamable Horn GLB of Σ.*

Interestingly, the procedure **RH_GLB** is anytime. If interrupted, it returns the current formula Λ which is a renamable Horn LB of Σ.

Experimental results

Generation of GLB

In (Kautz & Selman 1994; Selman & Kautz 1996) the experimental evaluation of knowledge compilation using theory approximation has been done using unit clause bounds. To our best knowledge, actual computations of Horn bounds have never been performed. This is why the experimental results given in this section are not compared with other methods. As an illustration of the efficiency of the algorithms for generating GLB on large size KBs, we tested them on large random k-CNF formula on a Pentium 200MHz. These instances have been generated around the hard region (Mitchell, Selman, & Levesque 1992; Dubois *et al.* 1996). Table 1 reports the results on 3-CNF instances of 1000 variables and 4-CNF instances of 300 variables. Up to one hour was allocated for each formula. CPU times are presented separately for preprocessing and for the **GLB** procedure.

k #vars	#clauses	#instances (#fails)	CPU Prep.	CPU **GLB**
3-cnf 1000	3800	100 (0)	0.5s	1s
	4000	100 (0)	0.8s	1s
	50	100 (0)	61s	0.5s
4-cnf 300	2700	100 (23)	2s	324s
	2800	100 (13)	4s	173s
	2900	100 (0)	84s	108s

Table 1: Generation of Horn GLBs of randomly generated 3-CNF formulae of 1000 variables and 4 -CNF formulae of 300 variables.

instance	#variables	#clauses	T_H	T_R
as2-yes.cnf	96	954	0.1s	55s
as3-yes.cnf	96	954	0.1s	26s
as6-yes.cnf	184	2277	12s	-
as8-yes.cnf	84	974	0.2s	140s
as10-yes.cnf	216	2780	265s	-
as11-yes.cnf	112	1312	1s	78s
as12-yes.cnf	72	1012	0.1s	43s
as14-yes.cnf	92	758	0.1s	123s

Table 2: Experiment on Asynchronous circuits design instances.

Table 2 reports the results of CPU computation time of Horn GLBs (T_H) and renamable Horn lower bounds (T_R) on 15 benchmarks coming from asynchronous circuits design (available at dimacs.rutgers.edu/pub/sat-files/ by anonymous ftp). The table reports only the benchmarks for which computation was possible. To let experiments be done in reasonable time, the renamable Horn lower bound is computed using **RH_GLB** which is interrupted, the first time a renamable Horn LB better than the Horn GLB computed is found.

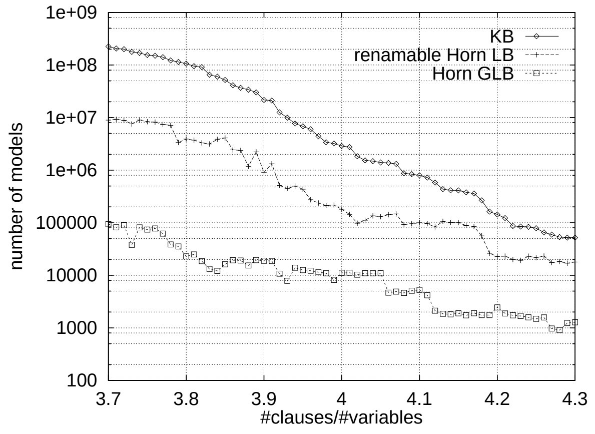

Figure 1: Number of models of Horn LBs, renamable Horn GLBs and KBs.

Quality of the lower bounds

Results obtained on random 3-CNF instances of 100 variables and for ratio #clauses/#variables ranging from 3.7 to 4.3 are represented in Figure 1. Each point of each curve is the mean number of models on a sample of 1000 instances. For the renamable Horn case, only lower bounds have been computed. In spite of that the renamable Horn LBs approximates better the KB than does the Horn GLB.

Conclusion

The renamable Horn class proves to be an interesting target language in propositional KB approximation. It can actually also improve the quality of upper bounds. Let us stress that the algorithms described in this paper can be easily modified to generate more compact tractable covers in the framework defined in (Boufkhad *et al.* 1997). An interesting issue for further research is to investigate the computational complexity of generating the renamable Horn bounds.

Aknowledgment

This work has been supported in part by the Ganymede II project of the ”Contrat de plan Etat/Nord-Pas-de-Calais”. Special thanks to Eric Grégoire for his useful comments that helped to improve this paper.

References

Aspvall, B. 1980. Recognizing disguised nr(1) instances of the satisfiability problem. *J. Algorithms* 1:97–103.

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

Cadoli, M. 1993. Semantical and computational aspects of horn approximations. In *Proc. IJCAI'93*, 39–44.

Davis, M., and Putnam, H. 1960. A computing procedure for quantification theory. *JACM* 7:201–215.

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

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

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

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

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

Henschen, L., and Wos, L. 1974. Unit refutations and horn sets. *JACM* 21:590–605.

Kautz, H., and Selman, B. 1994. An empirical evaluation of knowledge compilation by theory approximation. In *AAAI'94*, 155–161.

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

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

Mazure, B.; Saïs, L.; and Grégoire, E. 1997. Tabu search for SAT. In *AAAI'97*. 281–285.

Mitchell, D.; Selman, B.; and Levesque, H. 1992. Hard and easy distribution of sat problems. In *AAAI'92*, 459–465.

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

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

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

Selman, B., and Kautz, H. 1996. Knowledge compilation and theory approximation. *JACM* 43(2):193–224.