Knowledge Compilation and Theory Approximation

Henry Kautz and Bart Selman

Presented by Kelvin Ku

kelvin@cs.toronto.edu

Overview

- Problem
- Approach
- Horn Theory and Approximation
- Computing Horn Approximations
- Empirical Results
- Generalizations

Problem Formulation

- Goal: Inference in clausal propositional KBs
  - Σ is the KB/theory, α is a CNF formula
  - inference is checking $\Sigma \models \alpha$
- NP-complete (reduction to SAT)
- Alternatives
  - Restricted language: limited expressiveness
  - Incomplete inference, either resource-bounded or non-traditional inference: inconclusive results

Approach

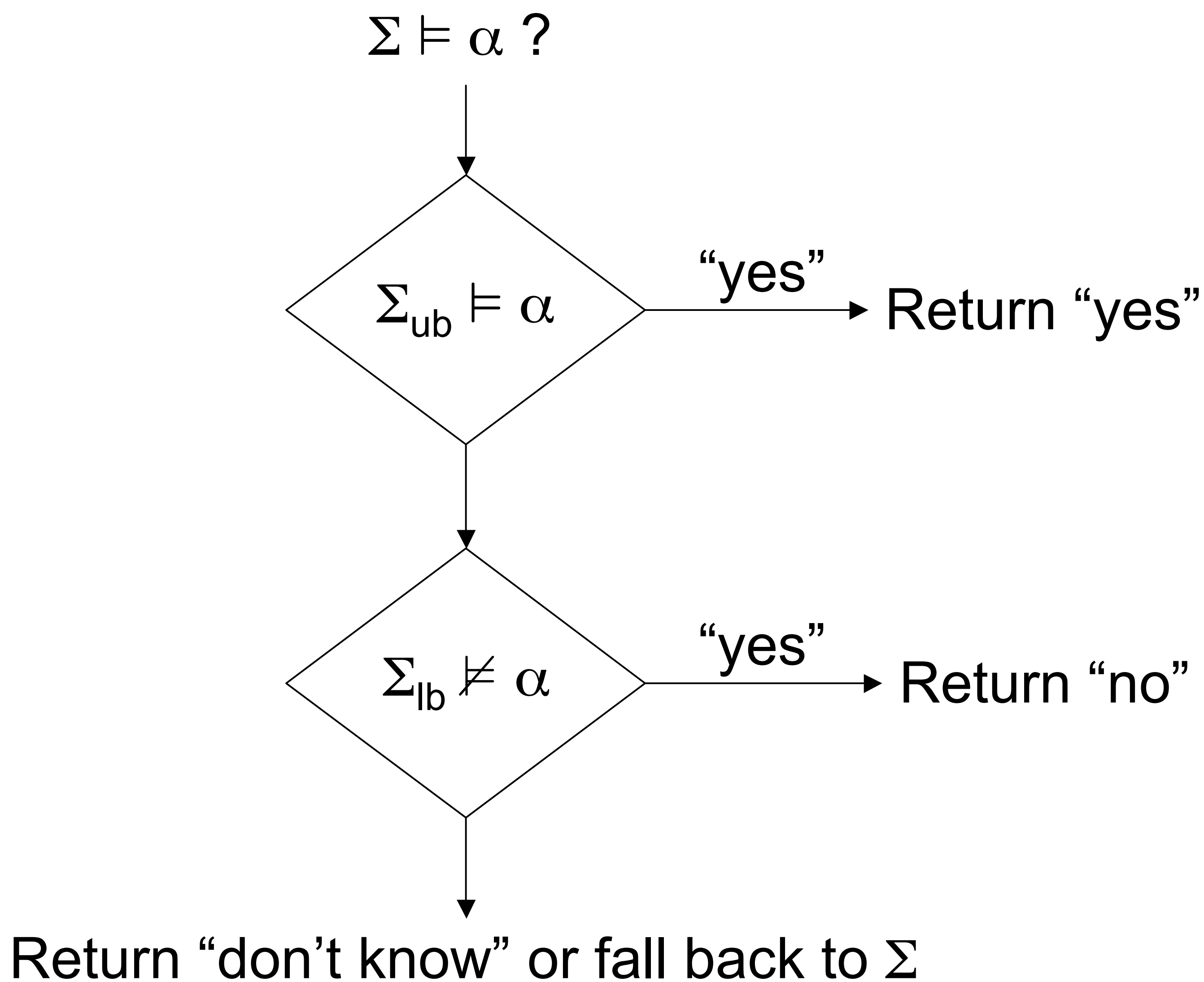
- Basis: Compile (rewrite) KB into tractable language
- Not always possible, e.g. with incomplete languages
- Compromise: <u>approximate</u> KB using tractable language

KB Approximation

- Approximate Σ using Σ_{lb} and Σ_{ub} such that
  - inference in $\Sigma_{lb,ub}$ is fast
  - bound the original theory: $\Sigma_{lb} \models \Sigma \models \Sigma_{ub}$
  - in other words, $\mathcal{M}(\Sigma_{lb}) \subseteq \mathcal{M}(\Sigma) \subseteq \mathcal{M}(\Sigma_{ub})$
    - $\mathcal{M}(\Sigma)$: models of Σ
- Σ_{glb} is greatest/weakest lower-bound
  - $\neg\exists\, \Sigma' \cdot M(\Sigma_{glb}) \subset M(\Sigma') \subseteq M(\Sigma)$
- Σ_{lub} is least/strongest upper-bound
  - $\neg\, \exists\, \Sigma' \cdot M(\Sigma) \subseteq M(\Sigma') \subset M(\Sigma_{lub})$
- Transitivity of $\models$ leads to fast querying scheme

...

Fast Querying Scheme



Horn Theory

- Form: Subset of CNF with at most one positive literal per clause
  - e.g. clause: $(\neg p \vee \neg q \vee r) \equiv (p \wedge q \Rightarrow r)$
  - Natural form for some domains, e.g. Prolog rules
- Inference: Linear in size (number of literals) of KB and query
- Incomplete: e.g. cannot represent $p \vee q$ as a Horn clause (since it has two models)

Horn Approximation Example

- Non-Horn theory Σ:

  $$(\neg a \vee c) \wedge (\neg b \vee c) \wedge (a \vee b)$$

- e.g. Horn LB: $a \wedge b \wedge c$
- e.g. Horn GLBs: $a \wedge c$, $b \wedge c$
- No Horn theory $\Sigma' \neq a \wedge c$ such that $(a \wedge c) \models \Sigma' \models \Sigma$
- e.g. Horn UB: $(\neg a \vee c) \wedge (\neg b \vee c)$
- Horn LUB: c
- No Horn theory $\Sigma' \neq c$ such that $\Sigma \models \Sigma' \models c$

Computing Horn Approximations

- Worst-case approximation time is $O(2^{|\Sigma|})$
  - either |approx| is $O(2^{|\Sigma|})$ or
  - |approx| is $O(|\Sigma|^n)$ and computation time is $O(2^{|\Sigma|})$
- Reasonable trade-off if we do many queries
- Compromise: incremental “anytime” approximation

Computing the Horn GLB

- Basis of method: Horn-strengthening
  - def: weakest Horn-clause subsuming a clause wrt. a positive literal
    - e.g. $\{\neg p, q, r\}$ has Horn-strengthenings $\{\neg p, q\}$ and $\{\neg p, r\}$
  - so remove all but one positive literal
- Lemma 1: If a Horn theory entails clause α, then it entails some Horn-strengthening of α
- Lemma 2: Every GLB of a theory is equivalent to some Horn-strengthening of the theory
- So searching through strengthenings for each clause will obtain the GLB; gives rise to algorithm ...

Generate_GLB

Input: a set of clauses $\Sigma = \{C_1, C_2, \ldots, C_n\}$

Output: a Horn GLB of Σ

```
begin
   L := first Horn-strengthening of Σ
   loop
        L' := next Horn-strengthening of Σ
        if none exists then exit loop
        if L ⊨ L' then L := L' /* found weaker LB */
   end loop
   remove subsumed clauses from L
   return L
end
```

Computing the Horn LUB

- Basis of method: Prime implicate
  - A strongest clause implied by Σ
  - In other words: $\Sigma \models C$ and $\neg\exists\, C' \subset C \cdot \Sigma \models C'$
- Horn LUB $\equiv$ all Horn prime implicates of Σ
- Thus, naive Generate_LUB: Compute all resolutions in Σ, and collect Horn prime implicates
  - resolution is complete

Size of Approximations

- $|GLB| \leq |\Sigma|$ (recall Generate_GLB)
- Theorem: There exist theories Σ such that $|LUB| \in O(2^{|\Sigma|})$, so LUB is EXP in the worst case
- Compromise: Theory Compaction
  - Transform original theory to obtain relatively smaller LUB
  - Theorem: compaction can't always help either

Empirical Results

- Hypothesis: Fast querying with approximations can efficiently answer queries that are intractable in the original theory
- Theoretical Motivation
  - Finding model of theory with unique model is intractable
  - Probablistic analysis of inference in hard random theories

Empirical Results

- Experiment: Compare execution time and coverage against Davis-Putnam on hard random theories
- Results
  - All queries answered
  - Total time (bounds and querying) significantly better than DP

Generalizations

- Parameters: languages of original theory, approximation, and query
- Formally, elements of framework are
  - $\mathcal{L}$, $\models$, $\mathcal{L}_S$, $\mathcal{L}_T$, $\mathcal{L}_Q$, $f_{L,U}$:$\mathcal{L}_S \times \mathbb{N} \rightarrow \mathcal{L}_T$
- For all classes of propositional clauses θ
  - ✓ Generate_GLB if (i) θ is closed under resolution and (ii) every clause is subsumed by a θ clause
  - ✓ Generate_LUB if (iii) θ is closed under subsumption

Alternative Clausal Languages

- Satisfying (i), (ii), and (iii)
  - Reverse-horn: $\{p \vee q \vee \neg r\}$
  - Binary: $\{p \vee \neg q, r\}$
  - Unit: $\{p, \neg q, r\}$
- Requiring modification to compilation algorithms:
  - k-Horn: Horn w/at most k literals per clause (violates (i), LUB has polynomial size)

Alternative Logics

- First-Order Logic
  - In general, <u>G</u>LB in ground clauses is not well-defined
    - e.g. for $\exists x.P(x)$ exists an infinite series of better LBs
  - Special case: first-order clauses (prenex, universal, clausal body) have Horn GLBs
    - e.g. $A(p) \vee B(q) \vee \neg C(r) \Leftarrow A(p) \vee \neg C(r)$
  - Exist finite first-order clausal theories with no finite Horn LUB (even without function symbols)
- Description Logic (concepts, subsumption)
  - Exists a tractable subset of $\mathcal{FL}$, $\mathcal{FL}^{-}$
  - Compute and store tractable subsumption bounds for each concept

Related Work

- Darwiche and Marquis paper considers compiling to complete languages with polytime entailment
  - Horn is incomplete but permits anytime approx
  - Only considering CE here
  - D&M concerned with more general queries and theory transformation