Complexity aspects of CNF to CNF compilation

Ondřej Čepek
Charles University in Prague, Czech Republic
(joint work with M.Babka, T.Balyo, Š.Gurský, P.Kučera, and V.Vlček)

Symposium on New Frontiers in Knowledge Compilation
Vienna Center for Logic and Algorithms, June 4-6, 2015

CNF to CNF compilation

- Input: arbitrary CNF
- Output: logically equivalent CNF with “good” inference properties
- Compilation method: add implicates
- What do we mean by “good” inference properties?

“Good” inference properties

- In general – polynomial time procedure to
  - prove that a given clause is an implicate of the given CNF (clausal entailment)
  - discover all entailed literals after any partial substitution
- In our context – we want unit propagation (UP) to suffice for both tasks

Clausal entailment by UP

- Clause C is 1-provable w.r.t. CNF formula F iff unit propagation on $F \wedge \neg C$ derives $\bot$.
- CNF F is unit refutation complete if every implicate of F is 1-provable w.r.t. F. (definition due to Alvaro del Val 1994)
- URC = class of unit refutation complete CNFs

Literal entailment by UP

- CNF F is propagation complete if for any partial assignment $x_1, \ldots, x_k$ and any literal d : if d is entailed from F by $x_1, \ldots, x_k$ then d is also entailed by unit propagation on F after fixing the values of $x_1, \ldots, x_k$. (definition due to Lucas Bordeaux, Joao Marques-Silva 2012)
- PC = class of propagation complete CNFs

URC versus PC

- Easy to see PC $\subseteq$ URC but not vice versa

$$F = (a \vee b \vee x) \wedge (a \vee c \vee \neg x)$$

- F is not in PC ($b=0$ and $c=0$ entail $a=1$ but unit propagation does not discover this fact)
- F is in URC (the only prime implicate not explicitly present in F is $a \vee b \vee c$ and it is clearly 1-provable)

How to achieve PC?

- Which clauses are worth adding?
- Clause $C = x_1 \vee \ldots \vee x_k$ is an empowering implicate for CNF F if C (after a possible renumbering) contains an empowered literal x_k such that
  - $F \wedge \neg x_1 \wedge \ldots \wedge \neg x_{k-1}$ entails x_k
  - unit propagation run on $F \wedge \neg x_1 \wedge \ldots \wedge \neg x_{k-1}$ entails neither $\bot$ nor x_k

(definition due to Darwiche, Pipatsrisawat 2011)

Notes on empowering implicates

- Asserting clauses learnt by CDCL SAT solvers are empowering implicates for the "current" CNF held by the solver.
- CNF F is propagation complete iff there exists no empowering implicate for F.
- CNF F can be turned into a propagation complete CNF by repeatedly adding empowering implicates (compilation process)

Complexity issues of such KC

- Given CNF F and clause C what is the complexity of deciding whether C is an empowering implicate for F?
- Given CNF F what is the complexity of deciding whether there exists an empowering implicate for F?
- Given CNF F how many empowering implicates need to be added to achieve PC?

Result 1

- Question: Given CNF F and clause C what is the complexity of deciding whether C is an empowering implicate for F?
- Answer: The problem is co-NP-complete.

Result 2

- Question: Given CNF F what is the complexity of deciding whether there exists an empowering implicate for F?
- Answer: The problem is NP-complete.
- Corollary: Testing whether a given CNF is PC is co-NP-complete.

Result 3

- Question: Given CNF F how many empowering implicates need to be added to achieve propagation completeness?
- Answer: There are CNFs for which an exponential number of empowering clauses has to be added to arrive to a PC formula.

Connection to CSP

- In CSP each variable X_i has its finite domain $D(X_i)$.
- Constraint – specifies which combinations of values from domains are allowed.
- Propagator P for a constraint C – an algorithm that restricts the domains of variables appearing in C
- P detects dis-entailment $\leftrightarrow$ P returns an empty domain whenever C has no solution
- P enforces domain consistency $\leftrightarrow$ for every domain value $d \in D(X_i)$ returned by P, there is a solution of C in which $X_i = d$.

Binarization of CSP variables

- Direct encoding – one Boolean variable for every value in every domain

$$x_{ij} = 1 \leftrightarrow X_i = j \text{ for } j \in D(X_i)$$

- ALO clauses

$$\forall i : (x_{i1} \vee x_{i2} \vee \ldots \vee x_{ip})$$

- AMO clauses

$$\forall i\ \forall j \neq k : (\neg x_{ij} \vee \neg x_{ik})$$

CNF decomposition for a propagator

- CNF decomposition F_P for a propagator P is a CNF on variables $(\mathbf{x},\mathbf{y})$ where $\mathbf{x}$ is the set of variables from the direct encoding and $\mathbf{y}$ is a set of auxiliary variables, such that
  - UP derives $\bot$ on $F_P \leftrightarrow P$ returns an empty domain
  - $x_{ij} \leftarrow 0$ by UP on $F_P \leftrightarrow P$ removes j from $D(X_i)$
- P detects dis-entailment $\leftrightarrow F_P$ is URC
- P enforces domain consistency $\leftrightarrow F_P$ is PC

Open problem

- What is the gap between the size of the input and the PC output if both CNFs are compiled into some other representation (e.g. ZBDD)?

Thank you for your attention.