Compilation of CNF-formulas: new algorithms and lower bounds

Florent Capelli

Based on results elaborated with Simone Bova, Johan Brault-Baron, Stefan Mengel, Friedrich Slivovsky

Journées du GT ALGA,
12 Avril 2016.

Knowledge compilation

- F a CNF-formulas represents some knowledge on a system
- we want to query this knowledge many times
- Compilation $=$ translate F into a good data structure that supports queries in PTIME

Knowledge compilation

- F a CNF-formulas represents some knowledge on a system
- we want to query this knowledge many times
- Compilation = translate F into a good data structure that supports queries in PTIME

Without compilation :
Is F satisfiable?
Please wait, an NP-complete problem is being solved... Yes
$\#F[x \mapsto 0, y \mapsto 1]$?
Please wait even longer... 237
Enumerate ∃x.F:
Please wait again... 01100110110
Are you bored?... 01100111111

With compilation :
Please wait, we are compiling F.
Is F satisfiable? YES
$\#F[x \mapsto 0, y \mapsto 1]$? 237
Enumerate ∃x.F ?
01100110110
01100111111
01100111101
...

Which kind of data structure?

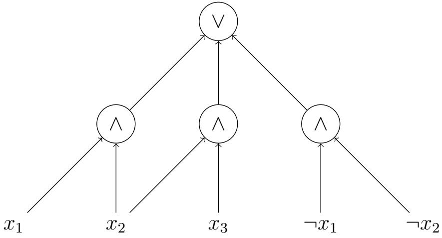
- Rich literature on the subject, numerous target languages exist
- In this talk: only (deterministic) DNNF, one of the most general

Which kind of data structure?

- Rich literature on the subject, numerous target languages exist
- In this talk: only (deterministic) DNNF, one of the most general

A DNNF D is a boolean circuit with gates $\wedge, \vee$ such that:

- inputs are labeled by literal $x, \neg x$
- $\wedge$ are *decomposable*: if α and β are the input of an $\wedge$-gate then $\mathrm{var}(D_\alpha) \cap \mathrm{var}(D_\beta) = \emptyset$



$$(x_1 \wedge x_2) \vee (x_2 \wedge x_3) \vee (\neg x_1 \wedge \neg x_2)$$

Supported PTIME queries

Given a DNNF D, we can in PTIME:

- Find $\tau \in \mathsf{sat}(D)$ in time $O(|D|)$
- Enumerate $\mathsf{sat}(D)$ with delay $O(|D| \cdot |\mathsf{var}(D)|)$
- Project D on partial assignments: $D[x \mapsto 0, y \mapsto 1]$.
- Existentially project D: $\exists x.D$

Supported PTIME queries

Given a DNNF D, we can in PTIME:

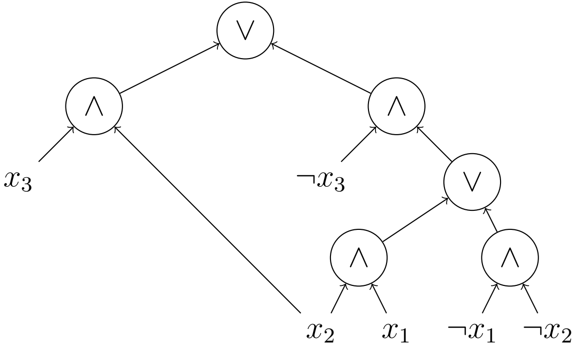
- Find $\tau \in \mathsf{sat}(D)$ in time $O(|D|)$
- Enumerate $\mathsf{sat}(D)$ with delay $O(|D| \cdot |\mathsf{var}(D)|)$
- Project D on partial assignments: $D[x \mapsto 0, y \mapsto 1]$.
- Existentially project D: $\exists x.D$

What about counting?

- #P-hard
- Main problem: overlap in the solution of $\vee$-gates

Deterministic DNNF

- $\vee$-gate with children α, β is *deterministic* if $D_\alpha \wedge D_\beta$ is UNSAT, i.e. $\mathsf{sat}(D_\alpha) \cap \mathsf{sat}(D_\beta) = \emptyset$.
- **deterministic DNNF** = all $\vee$-gates are deterministic
- support model counting in PTIME: replace $\vee$ by $+$ and $\wedge$ by $\times$



Structure based-algorithms

- When can we compile CNF-formula into DNNFs?
- Inspiration: algorithms for #SAT based on the structure of the formula
- Idea: restrict the variables-clauses interaction

Structure based-algorithms

- When can we compile CNF-formula into DNNFs?
- Inspiration: algorithms for #SAT based on the structure of the formula
- Idea: restrict the variables-clauses interaction

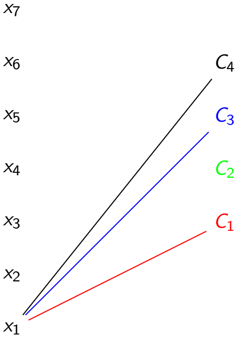


Figure:
$(x_1 \vee x_2 \vee x_3) \wedge (x_3 \vee x_4 \vee \neg x_5) \wedge (x_1 \vee x_5 \vee x_6) \wedge (x_1 \vee \neg x_3 \vee x_5 \vee \neg x_7)$

Structure based-algorithms

- When can we compile CNF-formula into DNNFs?
- Inspiration: algorithms for #SAT based on the structure of the formula
- Idea: restrict the variables-clauses interaction

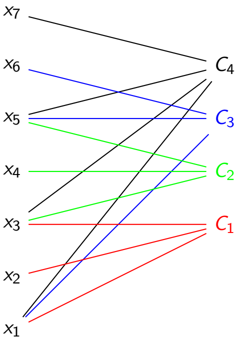


Figure:
$(x_1 \vee x_2 \vee x_3) \wedge (x_3 \vee x_4 \vee \neg x_5) \wedge (x_1 \vee x_5 \vee x_6) \wedge (x_1 \vee \neg x_3 \vee x_5 \vee \neg x_7)$

Structure based-algorithms

- When can we compile CNF-formula into DNNFs?
- Inspiration: algorithms for #SAT based on the structure of the formula
- Idea: restrict the variables-clauses interaction

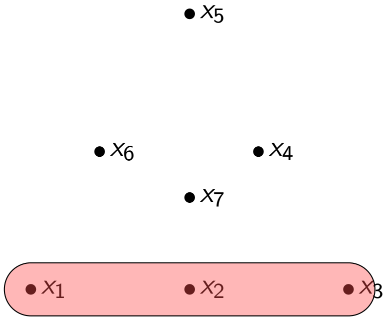


Figure:
$(x_1 \vee x_2 \vee x_3) \wedge (x_3 \vee x_4 \vee \neg x_5) \wedge (x_1 \vee x_5 \vee x_6) \wedge (x_1 \vee \neg x_3 \vee x_5 \vee \neg x_7)$

Structure based-algorithms

- When can we compile CNF-formula into DNNFs?
- Inspiration: algorithms for #SAT based on the structure of the formula
- Idea: restrict the variables-clauses interaction

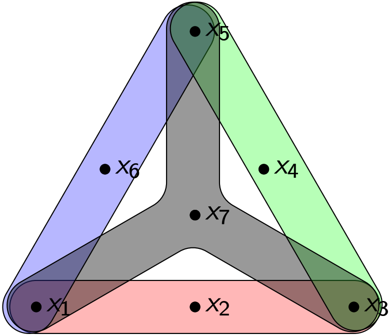


Figure:
$(x_1 \lor x_2 \lor x_3) \land (x_3 \lor x_4 \lor \neg x_5) \land (x_1 \lor x_5 \lor x_6) \land (x_1 \lor \neg x_3 \lor x_5 \lor \neg x_7)$

Structural tractability of #SAT

A class of graphs $\mathcal{C}$ is tractable for #SAT if:

- Given F, one can decide if the graph of F is in $\mathcal{C}$
- If so, one can output $\#F$ in polynomial time

Structural tractability of #SAT

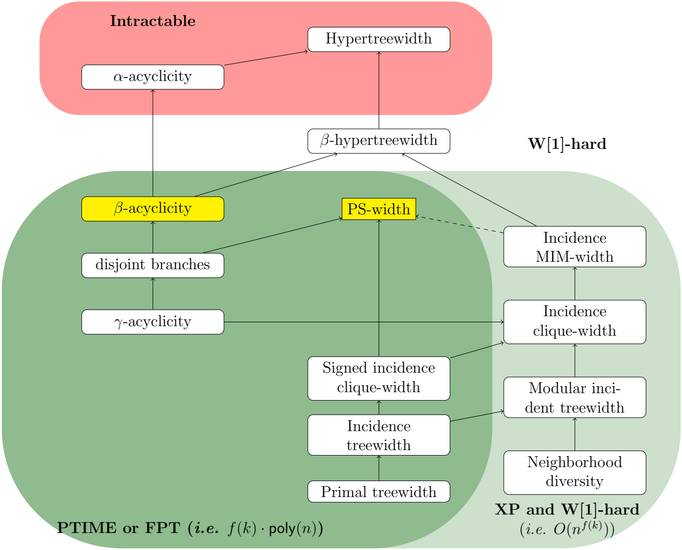
A class of graphs $\mathcal{C}$ is tractable for #SAT if:

- Given F, one can decide if the graph of F is in $\mathcal{C}$
- If so, one can output $\#F$ in polynomial time

Examples:

- #SAT is tractable on trees
- #SAT is tractable on bounded treewidth graphs
- ...

Structural tractability of #SAT



#SAT and knowledge compilation

- Existing tools for #SAT based on *exhaustive DPLL*:

$$\#F = \#F[x \mapsto 0] + \#F[x \mapsto 1]$$

  $+$ caching $+$ heuristics for choosing variables
- Implicitely construct a deterministic DNNF (Huang, Darwiche)

#SAT and knowledge compilation

- Existing tools for #SAT based on *exhaustive DPLL*:

$$\#F = \#F[x \mapsto 0] + \#F[x \mapsto 1]$$

  + caching + heuristics for choosing variables
- Implicitly construct a deterministic DNNF (Huang, Darwiche)
- The same is true for structural restriction based algorithms:

Theorem (Bova, C., Mengel, Slivovsky)

Every known structure-based algorithm for #SAT may be seen as an implicit compilation of the formula into deterministic DNNF.

- In particular, we can: count (with weights), enumerate, projects, find minimal assignments ...

Limitations of structure-based algorithm

- Known (structure-based) algorithms for #SAT = compilation into DNNF
- Hard instances for #SAT = lower bound on the size of equivalent DNNF

Limitations of structure-based algorithm

- Known (structure-based) algorithms for $\#\mathsf{SAT}$ = compilation into DNNF
- Hard instances for $\#\mathsf{SAT}$ = lower bound on the size of equivalent DNNF

Question

Can we always compile a CNF into a small DNNF?

- If NP $\not\subseteq$ P/poly, no...
- Can we prove it unconditionally?

Communication complexity

General model:

- $f : \{0,1\}^A \times \{0,1\}^B \to \{0,1\}$, $|A| \simeq |B|$
- Alice: $\bar{a} \in \{0,1\}^A$, Bob: $\bar{b} \in \{0,1\}^B$
- Complexity of f : how many bits Alice and Bob have to exchange in order to compute $f(\bar{a}, \bar{b})$?

Communication complexity

General model:

- $f : \{0,1\}^A \times \{0,1\}^B \to \{0,1\}$, $|A| \simeq |B|$
- Alice: $\bar{a} \in \{0,1\}^A$, Bob: $\bar{b} \in \{0,1\}^B$
- Complexity of f : how many bits Alice and Bob have to exchange in order to compute $f(\bar{a}, \bar{b})$?

Variations:

1. Complexity of f for a fixed partition A, B.
2. Complexity of f for the best partition A, B with $|A| = |B| \pm 1$

Communication complexity

General model:

- $f : \{0,1\}^A \times \{0,1\}^B \to \{0,1\}$, $|A| \simeq |B|$
- Alice: $\bar{a} \in \{0,1\}^A$, Bob: $\bar{b} \in \{0,1\}^B$
- Complexity of f : how many bits Alice and Bob have to exchange in order to compute $f(\bar{a}, \bar{b})$?

Variations:

1. Complexity of f for a fixed partition A, B.
2. Complexity of f for the best partition A, B with $|A| = |B| \pm 1$
3. Multipartition complexity of f where: an oracle sees the input $\bar{c}$ and choose the best partition A, B with $|A| \simeq |B|$

Lifting lower bounds

- DNNF have small multipartition complexity

Theorem (Bova, C., Mengel, Slivovsky)

Let D be a DNNF. The multipartition complexity of the function computed by D is at most $\log |D|$.

Lifting lower bounds

- DNNF have small multipartition complexity

Theorem (Bova, C., Mengel, Slivovsky)

Let D be a DNNF. The multipartition complexity of the function computed by D is at most $\log |D|$.

- Known lower bound on the multipartition complexity:

Theorem (Jukna, Schnigter)

There exists a family of 3*-CNF having multipartition complexity* $\Omega(n+m)$, **and thus no DNNF of size smaller than** $2^{\Omega(m+n)}$.

- We can actually construct a hard family of *monotone* 2*-CNF*

Conclusion

- Structural restrictions of CNF-formulas = restrict variables-clauses interaction
- Efficient algorithms for #SAT can often be lifted to knowledge compilation
- Hard instances for these algorithms = lower bound for knowledge compilation