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
$\#F[x \leftrightarrow 0, y \leftrightarrow 1]$?
Please wait even longer... 237
Enumerate $\exists x. F$:
Please wait again... 01100110110
Are you bored?... 01100111111

**With compilation:**
Please wait, we are compiling $F$.
Is $F$ satisfiable? YES
$\#F[x \leftrightarrow 0, y \leftrightarrow 1]$? 237
Enumerate $\exists x. F$:
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 $\land, \lor$ such that:

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

\[
(x_1 \land x_2) \lor (x_2 \land x_3) \lor (\neg x_1 \land \neg x_2)
\]
Supported PTIME queries

Given a DNNF $D$, we can in PTIME:

- Find $\tau \in \text{sat}(D)$ in time $O(|D|)$
- Enumerate $\text{sat}(D)$ with delay $O(|D| \cdot |\text{var}(D)|)$
- 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 $\tau \in \text{sat}(D)$ in time $O(|D|)$
- Enumerate $\text{sat}(D)$ with delay $O(|D| \cdot |\text{var}(D)|)$
- 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 $\lor$-gates
Deterministic DNNF

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

```
x_3 \lor \neg x_3 \lor x_2 \land \neg x_1 \land \neg x_2
```
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:**

\[(x_1 \lor x_2 \lor x_3) \land (x_3 \lor x_4 \lor \neg x_5) \land (x_1 \lor x_5 \lor x_6) \land (x_1 \lor \neg x_3 \lor x_5 \lor \neg x_7)\]
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

![Diagram of a formula with variables \( x_1, x_2, x_3, x_4, x_5, x_6, x_7 \) and clauses \( C_1, C_2, C_3, C_4 \).]

Figure:

\[
(x_1 \lor x_2 \lor x_3) \land (x_3 \lor x_4 \lor \neg x_5) \land (x_1 \lor x_5 \lor x_6) \land (x_1 \lor \neg x_3 \lor x_5 \lor \neg x_7)
\]
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:

\[(x_1 \lor x_2 \lor x_3) \land (x_3 \lor x_4 \lor \neg x_5) \land (x_1 \lor x_5 \lor x_6) \land (x_1 \lor \neg x_3 \lor x_5 \lor \neg x_7)\]
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:
\[(x_1 \lor x_2 \lor x_3) \land (x_3 \lor x_4 \lor \neg x_5) \land (x_1 \lor x_5 \lor x_6) \land (x_1 \lor \neg x_3 \lor x_5 \lor \neg x_7)\]
A class of graphs $C$ is tractable for #SAT if:

- Given $F$, one can decide if the graph of $F$ is in $C$
- If so, one can output $\#F$ in polynomial time
A class of graphs \( C \) is tractable for \#SAT if:
- Given \( F \), one can decide if the graph of \( F \) is in \( C \)
- If so, one can output \( \#F \) in polynomial time

Examples:
- \#SAT is tractable on trees
- \#SAT is tractable on bounded treewidth graphs
- \ldots
Structural tractability of #SAT

PTIME or FPT (i.e. $f(k) \cdot \text{poly}(n)$)

XP and W[1]-hard (i.e. $O(n^{f(k)})$)

Intractable

- $\alpha$-acyclicity
- $\beta$-acyclicity
- disjoint branches
- $\gamma$-acyclicity

Hypertreewidth

- $\beta$-hypertreewidth

PS-width

- Incidence clique-width
- Signed incidence clique-width
- Incidence treewidth
- Primal treewidth
- Incidence MIM-width
- Modular incidence treewidth
- Neighborhood diversity

W[1]-hard
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
- Implicitely 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 $\#\text{SAT} =$ compilation into DNNF
- Hard instances for $\#\text{SAT} =$ lower bound on the size of equivalent DNNF
Limitations of structure-based algorithm

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

**Question**

*Can we always compile a CNF into a small DNNF?*

- If $\text{NP} \not\subseteq \text{P/poly}$, 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 $f(\bar{a}, \bar{b})$?

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 chooses the best partition $A$, $B$ with $|A| \simeq |B|$.
Communication complexity

General model:

- $f : \{0, 1\}^A \times \{0, 1\}^B \rightarrow \{0, 1\}, \ |A| \simeq |B|$
- 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 $\log |D|$. 

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 $2^{\Omega(n + m)}$.

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