SAT Solving

Jan Johannsen

Lecture course, winter semester 2018/19
Overview

Introduction

Tractable cases

DPLL algorithms

CDCL solvers

Lookahead-based solvers

Probabilistic algorithms

Certification

Applications
Overview

Introduction
    Propositional Logic
    Normal Forms
    Complexity

Tractable cases

DPLL algorithms

CDCL solvers

Lookahead-based solvers

Probabilistic algorithms

Certification

Applications
Propositional logic: syntax

Let $X$ be a set of variables.

Propositional formulas over $X$ are defined inductively:

- constants 0 and 1 are formulas
- each variable $x \in X$ is a formula
- if $F$ is a formula, then so is $\neg F$
- if $F$ and $G$ are formulas, then so is $(F \land G)$
- if $F$ and $G$ are formulas, then so is $(F \lor G)$
Some definitions

$V(F)$ is the set of variables occurring in $F$:
- $V(0) = V(1) = \emptyset$
- $V(x) = \{x\}$
- $V(\neg F) = V(F)$
- $V(F \land G) = V(F \lor G) = V(F) \cup V(G)$

Size $|F|$ of $F$:
- $|0| = |1| = |x| = 1$
- $|\neg F| = |F| + 1$
- $|F \land G| = |F \lor G| = |F| + |G| + 1$
An assignment for $F$ is a finite partial map $\alpha : V(F) \rightarrow \{0, 1\}$, written as $[x_1 \leftarrow \epsilon_1, \ldots, x_k \leftarrow \epsilon_k]$ where $x_i \in \text{dom } \alpha \subseteq V(F)$ and $\epsilon_i = \alpha(x_i) \in \{0, 1\}$.

Value $F\alpha$ is computed:

- Replace every $x \in \text{dom } \alpha$ by $\alpha(x)$
- Simplify according to the rewrite rules:
  - $\neg 0 \leadsto 1$, $\neg 1 \leadsto 0$
  - $F \land 0$, $0 \land F \leadsto 0$,
  - $F \land 1$, $1 \land F \leadsto F$
  - $F \lor 0$, $0 \lor F \leadsto F$,
  - $F \lor 1$, $1 \lor F \leadsto 1$
Assignment $\alpha$ ist total, if $\text{dom} \, \alpha = V(F)$, otherwise partial.

$\alpha$ total: $F\alpha \in \{0, 1\}$

$\alpha$ partial: $V(F\alpha) \subseteq V(F) \setminus \text{dom} \, \alpha$.

$\alpha$ satisfies $F$, written $\alpha \models F$, if $F\alpha = 1$.

$F$ is satisfiable, if $\alpha \models F$ for some $\alpha$.

$F$ is a tautology, if $\alpha \models F$ for all total $\alpha$.

$F$ and $G$ are equivalent ($F \equiv G$) if $F\alpha = G\alpha$ for all total $\alpha$. 
A literal is a variable $x$ or a negated variable $\neg x$.

Formulas in negation normal form (NNF) are defined by:

- 0, 1 and literals are in NNF,
- if $F$ and $G$ are in NNF, then so are $F \land G$ and $F \lor G$.

Thus: $F$ is in NNF if negations occur only at variables.
Negation

For a formula $F$ in NNF, the formula $\overline{F}$ in NNF is defined by:

- $\overline{0} = 1$, $\overline{1} = 0$
- $\overline{\bar{x}} = x$
- if $F = \neg x$, then $\overline{F} = x$
- if $F = G \land H$, then $\overline{F} = \overline{G} \lor \overline{H}$
- if $F = G \lor H$, then $\overline{F} = \overline{G} \land \overline{H}$

Lemma

For $F$ in NNF, $\overline{F} \equiv \neg F$. 
Construction of NNF

**Theorem**

*For every formula* $F$, there is $n(F)$ in NNF with $n(F) \equiv F$.

**Proof:** By construction:

- $F = x$, $F = 0$ or $F = 1$ are in NNF, so $n(F) = F$.
- For $F = G \land H$, by induction we have $n(G)$ and $n(H)$ in NNF. Let $n(F) := n(G) \land n(H)$.
- Analogously for $F = G \lor H$.
- For $F = \neg G$, by induction we have $H := n(G)$ in NNF. Let $n(F) = \bar{H}$. 
Formulas in disjunctive and conjunctive normal form (DNF and CNF) are defined by:

- A term is a conjunction \( a_1 \land \ldots \land a_k \) of literals.
- A clause is a disjunction \( a_1 \lor \ldots \lor a_k \) of literals.
- A formula in DNF is a disjunction \( T_1 \lor \ldots \lor T_m \) of terms.
- A formula in CNF is a conjunction \( C_1 \land \ldots \land C_m \) of clauses.

Every formula in CNF or DNF is in NNF.

Theorem

For every formula \( F \), there are \( \hat{F} \) in CNF and \( \tilde{F} \) in DNF with \( \hat{F} \equiv \tilde{F} \equiv F \).
## Exponential Blowup

<table>
<thead>
<tr>
<th>Theorem</th>
</tr>
</thead>
<tbody>
<tr>
<td>There are formulas $F_n$ in CNF of size $</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Corollary</th>
</tr>
</thead>
<tbody>
<tr>
<td>There are formulas $F'_n$ in DNF of size $</td>
</tr>
</tbody>
</table>
Width of clauses

The width $w(C)$ of a clause $C = a_1 \lor \ldots \lor a_k$ is $k$.

A formula is in $k$-CNF if every clause $C$ in $F$ is of width $w(C) \leq k$.

**Theorem**

*For every $k$, there is a formula $F$ in $(k+1)$-CNF, for which there is no $F'$ in $k$-CNF with $F' \equiv F$.***
Clause $C = a_1 \lor \ldots \lor a_k$ is identified with the set $\{a_1, \ldots, a_k\}$.

CNF-formula $F = C_1 \land \ldots \land C_m$ is identified with the set $\{C_1, \ldots, C_m\}$.

Let $F \setminus C := F \setminus \{C\}$.

For every formula $F$ in CNF, we denote by

- $n$ the number of variables in $F$
- $m$ the number of clauses in $F$
- $k$ the width of $F$. 

Some definitions

For $F$ in CNF:

$\alpha \models F$ if in every clause $C$ there is literal $a \in C$ with $\alpha(a) = 1$.

A clause of width 1 is called a unit clause.

**Property**

If $a$ is a unit clause in $F$, then $\alpha(a) = 1$ for every $\alpha \models F$.

Literal $a$ is pure in $F$ if $\overline{a}$ does not occur in $F$.

**Property**

If $C \in F$ contains a pure literal, then $F$ is satisfiable iff $F \setminus C$ is satisfiable.
Cook’s Theorem

Problem FSAT

Instance: Formula $F$
Question: Is $F$ satisfiable?

Theorem

FSAT is NP-complete.
The problem SAT

Problem $k$-SAT

Instance: Formula $F$ in $k$-CNF
Question: Is $F$ satisfiable?

Theorem

*For every formula $F$ there is a formula $E(F)$ in 3-CNF s.t. $E(F)$ is satisfiable iff $F$ is satisfiable.*

Corollary

*SAT and $k$-SAT for $k \geq 3$ are NP-complete.*
Limiting occurrences

For a class $\mathcal{F}$ of formulas:
$\mathcal{F}(k)$: formulas in $\mathcal{F}$ with $\leq k$ occurrences of every variable.

**Theorem**

For every formula $F$ in CNF there is a formula $D(F)$ in CNF(3) with $w(D(F)) = w(F)$ s.t. $D(F)$ is satisfiable iff $F$ is satisfiable.

**Corollary**

$3$-SAT(3) is NP-complete.
More on limited occurrences

$E_k$-CNF: formulas with exactly $k$ literals per clause

**Proposition**

E3-SAT(3) is trivial: all formulas in E3-CNF(3) are satisfiable.

OTOH: E3-SAT(4) is NP-complete.

More general: for every $k$, E$k$-SAT($k$) is trivial.
More on limited occurrences

**Theorem**

For every $k \geq 3$, there is $s = s(k) \geq k$ such that

- $E_k\text{-SAT}(s)$ is trivial
- $E_k\text{-SAT}(s+1)$ is NP-complete.

**Bounds on the value $s(k)$:**

<table>
<thead>
<tr>
<th>$k$</th>
<th>$s(k) \geq$</th>
<th>$s(k) &lt;$</th>
</tr>
</thead>
<tbody>
<tr>
<td>3</td>
<td>3</td>
<td>4</td>
</tr>
<tr>
<td>4</td>
<td>4</td>
<td>5</td>
</tr>
<tr>
<td>5</td>
<td>5</td>
<td>8</td>
</tr>
<tr>
<td>6</td>
<td>7</td>
<td>12</td>
</tr>
<tr>
<td>7</td>
<td>13</td>
<td>18</td>
</tr>
<tr>
<td>8</td>
<td>24</td>
<td>30</td>
</tr>
<tr>
<td>9</td>
<td>41</td>
<td>52</td>
</tr>
</tbody>
</table>