SAT Solving

Jan Johannsen

Lecture course, winter semester 2018/19

Overview

Overview

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 $(F \wedge G)$
- if F and G are formulas, then so is $(F \vee G)$

Some definitions

$V(F)$ is the set of variables occurring in F:

- $V(0) = V(1) = \emptyset$
- $V(x) = \{x\}$
- $V(\neg F) = V(F)$
- $V(F \wedge G) = V(F \vee G) = V(F) \cup V(G)$

Size $|F|$ of F:

- $|0| = |1| = |x| = 1$
- $|\neg F| = |F| + 1$
- $|F \wedge G| = |F \vee G| = |F| + |G| + 1$

Semantics

An assignment for F is a finite partial map $\alpha : V(F) \to \{0,1\}$,
written as $[x_1 \leftarrow \epsilon_1, \ldots, x_k \leftarrow \epsilon_k]$
where $x_i \in \operatorname{dom}\alpha \subseteq V(F)$ and $\epsilon_i = \alpha(x_i) \in \{0,1\}$.

Value $F\alpha$ is computed:

- Replace every $x \in \operatorname{dom}\alpha$ by $\alpha(x)$
- Simplify according to the rewrite rules:
  - $\neg 0 \rightsquigarrow 1, \quad \neg 1 \rightsquigarrow 0$
  - $F \wedge 0\,,\, 0 \wedge F \;\rightsquigarrow\; 0,$
  - $F \wedge 1\,,\, 1 \wedge F \;\rightsquigarrow\; F$
  - $F \vee 0\,,\, 0 \vee F \;\rightsquigarrow\; F,$
  - $F \vee 1\,,\, 1 \vee F \;\rightsquigarrow\; 1$

Satisfiability

Assignment α ist total, if $\operatorname{dom}\alpha = V(F)$, otherwise partial.

α total: $\quad F\alpha \in \{0,1\}$

α partial: $\quad V(F\alpha) \subseteq V(F) \setminus \operatorname{dom}\alpha$.

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

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

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

F and G are equivalent ($F \equiv G$) if $F\alpha = G\alpha$ for all total α.

Negation normal form

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 $F \wedge G$ and $F \vee G$.

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

Negation

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

- $\bar{0} = 1$, $\bar{1} = 0$
- $\bar{x} = \neg x$
- if $F = \neg x$, then $\bar{F} = x$
- if $F = G \wedge H$, then $\bar{F} = \bar{G} \vee \bar{H}$
- if $F = G \vee H$, then $\bar{F} = \bar{G} \wedge \bar{H}$

Lemma

For F in NNF, $\bar{F} \equiv \neg F$.

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 $F = G \wedge H$, by induction we have $n(G)$ and $n(H)$ in NNF. Let $n(F) := n(G) \wedge n(H)$.
- Analogously for $F = G \vee H$.
- For $F = \neg G$, by induction we have $H := n(G)$ in NNF. Let $n(F) = \bar{H}$.

Disjunctive and Conjunctive Normal Form

Formulas in disjunctive and conjunctive normal form
(DNF and CNF) are defined by:

- A term is a conjunction $a_1 \wedge \ldots \wedge a_k$ of literals.
- A clause is a disjunction $a_1 \vee \ldots \vee a_k$ of literals.
- A formula in DNF is a disjunction $T_1 \vee \ldots \vee T_m$ of terms.
- A formula in CNF is a conjunction $C_1 \wedge \ldots \wedge C_m$ of clauses.

Every formula in CNF or DNF is in NNF.

Theorem

For every formula F, there are $\hat{F}$ in CNF and $\check{F}$ in DNF with $\hat{F} \equiv \check{F} \equiv F$.

Exponential Blowup

Theorem

There are formulas F_n in CNF of size $|F_n| = O(n)$ s.t. every formula $G_n \equiv F_n$ in DNF is of size $|G_n| \geq n2^n$.

Corollary

There are formulas F'_n in DNF of size $|F'_n| = O(n)$ s.t. every formula $H_n \equiv F'_n$ in CNF is of size $|H_n| \geq n2^n$.

Width of clauses

The width $w(C)$ of a clause $C = a_1 \vee \ldots \vee a_k$ 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 $F' \equiv F$.

Notation

Clause $C = a_1 \vee \ldots \vee a_k$ is identified with the set $\{a_1, \ldots, a_k\}$.

CNF-formula $F = C_1 \wedge \ldots \wedge C_m$ 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 $\bar{a}$ 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

Ek-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, Ek-SAT(k) is trivial.

More on limited occurrences

Theorem

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

- *Ek-SAT(s) is trivial*
- *Ek-SAT(s+1) is NP-complete.*

Bounds on the value $s(k)$:

k	$s(k) \geq$	$s(k) <$
3	3	4
4	4	5
5	5	8
6	7	12
7	13	18
8	24	30
9	41	52