universität innsbruck

Constraint Solving

Cezary Kaliszyk René Thiemann

based on a previous course by Aart Middeldorp

Outline

Important Information

- LVA 703304 (VO 2) + 703305 (PS 2)
- http://cl-informatik.uibk.ac.at/teaching/ws21/cs
- Please register for both VO and PS
- OLAT link for VO

Time and Place

VO	Friday	10:15 – 12:00	SR 13 (except 22.11: HSB 6)	CK & RT
PS	Thursday	10:15 – 12:00	HSB 9	CK & RT

Consultation Hours

Cezary Kaliszyk	3M12	Thursday	12:00–13:00
René Thiemann	3M09 and online	Tuesday	10:00 – 11:00

Schedule

week 1	08.10 & 14.10	week 6	12.11 & 18.11	week 11	17.12 & 13.01
week 2	21.10	week 7	19.11 & 25.11	week 12	14.01 & 20.01
week 3	22.10 & 28.10	week 8	26.11 & 02.12	week 13	21.01 & 27.01
week 4	29.10 & 04.11	week 9	03.12 & 09.12	week 14	28.01 & 03.02
week 5	05.11 & 11.11	week 10	10.12 & 16.12	week 15	04.02 first exam

Grading — Vorlesung

- first (online) exam on February 04
- registration starts 5 weeks before exam and ends 2 weeks before exam
- de-registration is possible until 23:59 on February 3
- second and third exams in March and September (on demand)

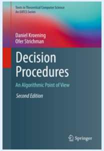
Grading — Proseminar

- solved exercises must be marked in OLAT before 8 am on Thursday (physical marking during physical PS)
- 10 points per PS
- attendance is compulsory; unexcused absence is allowed twice (email solutions to get some points in such cases)
- additional points for presentation of solutions

Literature

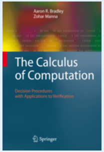
Daniel Kröning and Ofer Strichman
Decision Procedures – An Algorithmic Point of View (Second Edition)
Springer, 2016 online version via Ebook Central

http://www.decision-procedures.org



Aaron Bradley and Zohar Manna
The Calculus of Computation – Decision Procedures with Applications to Verification
Springer, 2007 online version via Ebook Central

http://theory.stanford.edu/~arbrad/book.html



Online Material

slides and additional reading material are available from uibk.ac.at domain

Topics

- propositional logic (SAT)
  binary decision diagrams, cardinality constraints, DPLL, maxSAT, NP-completeness, unsatisfiable cores
- satisfiability modulo theories (SMT)
  DPLL(T), Nelson–Oppen combination method
- equality logic and uninterpreted functions (EUF)
  Ackermann's reduction, Bryant's reduction, congruence closure, graph-based reduction
- linear arithmetic (LIA and LRA)
  branch and bound, cutting planes, difference logic, Fourier–Motzkin variable elimination, omega test, simplex algorithm
- bit vectors (BV) and floating points (FP)
  bit-vector arithmetic, bounded model checking, fixed-point arithmetic, flattening
- arrays (AX) and pointers
  array properties, heap-allocated data structures, lazy encoding, pointer logic
- quantified formulas (QBF)
  Cooper's method, PSPACE-completeness, quantified boolean formulas, QDPLL, Q-resolution

Outline

Concepts of Propositional Logic

- formula
- assignment
- satisfiability
- validity
- negation normal form (NNF)
- conjunctive normal form (CNF)
- disjunctive normal form (DNF)
- literal

Definition (Propositional Logic: Syntax)

- propositional formulas are built from
  - atoms $p, q, r, p_1, p_2, \ldots$
  - bottom, top $\bot, \top$
  - negation $\neg$ $\neg p$ "not p"
  - conjunction $\wedge$ $p \wedge q$ "p and q"
  - disjunction $\vee$ $p \vee q$ "p or q"
  - implication $\rightarrow$ $p \rightarrow q$ "if p then q"
  - equivalence $\leftrightarrow$ $p \leftrightarrow q$ "p if and only if q"

  according to BNF grammar $\phi ::= p \mid \bot \mid \top \mid (\neg\phi) \mid (\phi \wedge \phi) \mid (\phi \vee \phi) \mid (\phi \rightarrow \phi) \mid (\phi \leftrightarrow \phi)$
- notational conventions:
  - binding precedence $\neg > \wedge, \vee > \rightarrow, \leftrightarrow$ omit outer parentheses
  - $\rightarrow, \wedge, \vee$ are right-associative: $p \rightarrow q \rightarrow r$ denotes $p \rightarrow (q \rightarrow r)$

Definition (Propositional Logic: Semantics)

- valuation (truth assignment) is mapping $v\colon \{p \mid p \text{ is atom}\} \to \{\mathsf{T}, \mathsf{F}\}$
  truth values
- extension to formulas:
  - $v(\bot) = \mathsf{F}$
  - $v(\top) = \mathsf{T}$
  - $v(\neg\phi) = \begin{cases} \mathsf{T} & \text{if } v(\phi) = \mathsf{F} \\ \mathsf{F} & \text{otherwise} \end{cases}$
  - $v(\phi \wedge \psi) = \begin{cases} \mathsf{T} & \text{if } v(\phi) = v(\psi) = \mathsf{T} \\ \mathsf{F} & \text{otherwise} \end{cases}$
  - $v(\phi \vee \psi) = \begin{cases} \mathsf{F} & \text{if } v(\phi) = v(\psi) = \mathsf{F} \\ \mathsf{T} & \text{otherwise} \end{cases}$
  - $v(\phi \to \psi) = \begin{cases} \mathsf{F} & \text{if } v(\phi) = \mathsf{T} \text{ and } v(\psi) = \mathsf{F} \\ \mathsf{T} & \text{otherwise} \end{cases}$
  - $v(\phi \leftrightarrow \psi) = \begin{cases} \mathsf{T} & \text{if } v(\phi) = v(\psi) \\ \mathsf{F} & \text{otherwise} \end{cases}$

Definitions

- semantic entailment

$$\phi_1, \phi_2, \ldots, \phi_n \vDash \psi$$

  if $v(\psi) = \mathsf{T}$ whenever $v(\phi_1) = v(\phi_2) = \cdots = v(\phi_n) = \mathsf{T}$, for every valuation v
- formula ϕ is valid if $v(\phi) = \mathsf{T}$ for every valuation v
- formula ϕ is satisfiable if $v(\phi) = \mathsf{T}$ for some valuation v
- formulas ϕ and ψ are equivalent ($\phi \equiv \psi$) if $v(\phi) = v(\psi)$ for every valuation v
- formulas ϕ and ψ are equisatisfiable ($\phi \approx \psi$) if

$$\phi \text{ is satisfiable} \iff \psi \text{ is satisfiable}$$

Theorem

- formula ϕ is valid $\iff$ $\neg\phi$ is unsatisfiable
- validity and satisfiability are decidable

Definitions

- negation normal form (NNF) is formula without implication and equivalence, and with negation only applied to atoms
- literal is atom p or negation $\neg p$ of atom
- clause is disjunction of literals
- conjunctive normal form (CNF) is conjunction of clauses
- disjunctive normal form (DNF) is disjunction of conjunctions of literals

Theorem

$\forall$ formula ϕ $\exists$ CNF ψ $\exists$ DNF χ such that $\phi \equiv \psi \equiv \chi$

Satisfiability (SAT)

instance: (propositional) formula ϕ

question: is ϕ satisfiable?

Theorem

SAT is NP-complete, even for CNF formulas

Remark

most SAT solvers require CNF as input

DIMACS Input Format

`c`	
`c comments`	
`c`	
`p cnf 4 3`	4 atoms and 3 clauses
`1 -2 4 0`	$x_1 \vee \neg x_2 \vee x_4$
`-1 2 -3 -4 0`	$\neg x_1 \vee x_2 \vee \neg x_3 \vee \neg x_4$
`3 -2 0`	$x_3 \vee \neg x_2$

SAT Applications

Applications of SAT

- Encoding games
  `http://cl-informatik.uibk.ac.at/software/puzzles/`
- Strategies and configurations
- Cryptanalysis
- Many graph problems
- Component of reasoning in more complex logics

Outline

Remarks

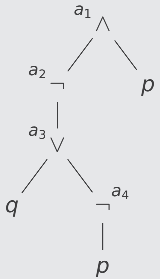
- translation from arbitrary formula to equivalent CNF is expensive
- Tseitin's transformation is linear-time translation to equisatisfiable CNF

Example (Tseitin's Transformation)

- $\phi = \neg(q \vee \neg p) \wedge p$
- introduce new variable for each propositional connective:

a_1	$\neg(q \vee \neg p) \wedge p$	a_3	$q \vee \neg p$
a_2	$\neg(q \vee \neg p)$	a_4	$\neg p$

- $\phi \approx a_1 \wedge (a_1 \leftrightarrow a_2 \wedge p) \wedge (a_2 \leftrightarrow \neg a_3) \wedge (a_3 \leftrightarrow q \vee a_4) \wedge (a_4 \leftrightarrow \neg p)$



Lemma

1. $(\phi \leftrightarrow \neg\psi) \equiv (\phi \vee \psi) \wedge (\neg\phi \vee \neg\psi)$
2. $(\phi \leftrightarrow \psi \wedge \chi) \equiv (\neg\phi \vee \psi) \wedge (\neg\phi \vee \chi) \wedge (\phi \vee \neg\psi \vee \neg\chi)$
3. $(\phi \leftrightarrow \psi \vee \chi) \equiv (\phi \vee \neg\psi) \wedge (\phi \vee \neg\chi) \wedge (\neg\phi \vee \psi \vee \chi)$

Example (cont'd)

$$\phi \approx a_1 \wedge (a_1 \leftrightarrow a_2 \wedge p) \wedge (a_2 \leftrightarrow \neg a_3) \wedge (a_3 \leftrightarrow q \vee a_4) \wedge (a_4 \leftrightarrow \neg p)$$

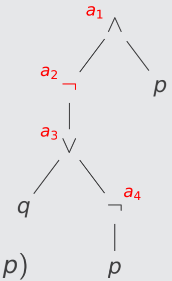
$$\equiv a_1 \wedge (\neg a_1 \vee a_2) \wedge (\neg a_1 \vee p) \wedge (a_1 \vee \neg a_2 \vee \neg p) \wedge (a_2 \vee a_3) \wedge (\neg a_2 \vee \neg a_3)$$
$$\wedge (a_3 \vee \neg q) \wedge (a_3 \vee \neg a_4) \wedge (\neg a_3 \vee q \vee a_4) \wedge (a_4 \vee p) \wedge (\neg a_4 \vee \neg p)$$

Improvement (Plaisted & Greenbaum 1986)

replace equivalence ($\leftrightarrow$) by implication ($\rightarrow$ or $\leftarrow$) based on polarity of subformulas

Example (cont'd)

- $\phi = \neg(q \vee \neg p) \wedge p$
- $\phi \approx a_1 \wedge (a_1 \rightarrow a_2 \wedge p) \wedge (a_2 \rightarrow \neg a_3) \wedge (a_3 \leftarrow q \vee a_4) \wedge (a_4 \leftarrow \neg p)$
- $a_1 \rightarrow a_2 \wedge p \;\equiv\; (\neg a_1 \vee a_2) \wedge (\neg a_1 \vee p) \wedge (a_1 \vee \neg a_2 \vee \neg p)$
- $a_2 \rightarrow \neg a_3 \;\equiv\; (a_2 \vee a_3) \wedge (\neg a_2 \vee \neg a_3)$
- $a_3 \leftarrow q \vee a_4 \;\equiv\; (a_3 \vee \neg q) \wedge (a_3 \vee \neg a_4) \wedge (\neg a_3 \vee q \vee a_4)$
- $a_4 \leftarrow \neg p \;\equiv\; (a_4 \vee p) \wedge (\neg a_4 \vee \neg p)$
- $\phi \approx a_1 \wedge (\neg a_1 \vee a_2) \wedge (\neg a_1 \vee p) \wedge (\neg a_2 \vee \neg a_3) \wedge (a_3 \vee \neg q) \wedge (a_3 \vee \neg a_4) \wedge (a_4 \vee p)$



replace $a \leftrightarrow \psi$ by $a \rightarrow \psi$ if ψ occurs positively and by $a \leftarrow \psi$ otherwise

Definition

subformula ψ occurs **positively** in formula ϕ if number of negations on path from root of ϕ to root of ψ in parse tree of ϕ is even

Outline

Remarks

- most state-of-the-art SAT solvers are based on variations of Davis – Putnam – Logemann – Loveland (DPLL) procedure (1960, 1962)
- abstract version of DPLL described in JACM paper of Nieuwenhuis, Oliveras, Tinelli (2006)

Definition (Abstract DPLL)

- states $M \parallel F$ consist of
  - list M of (possibly annotated) non-complementary literals
  - CNF F
- transition rules

$$M \parallel F \quad \Longrightarrow \quad M' \parallel F' \quad \text{or} \quad \text{fail-state}$$

Example

$$\phi = (\neg 1 \vee \neg 2) \wedge (2 \vee 3) \wedge (\neg 1 \vee \neg 3 \vee 4) \wedge (2 \vee \neg 3 \vee \neg 4) \wedge (1 \vee 4)$$

$$
\begin{array}{lrcll}
 & & \| & \neg 1 \vee \neg 2,\ 2 \vee 3,\ \neg 1 \vee \neg 3 \vee 4,\ 2 \vee \neg 3 \vee \neg 4,\ 1 \vee 4 & \\
\Longrightarrow & \overset{d}{1} & \| & \neg 1 \vee \neg 2,\ 2 \vee 3,\ \neg 1 \vee \neg 3 \vee 4,\ 2 \vee \neg 3 \vee \neg 4,\ 1 \vee 4 & \text{decide} \\
\Longrightarrow & \overset{d}{1}\ \neg 2 & \| & \neg 1 \vee \neg 2,\ 2 \vee 3,\ \neg 1 \vee \neg 3 \vee 4,\ 2 \vee \neg 3 \vee \neg 4,\ 1 \vee 4 & \text{unit propagate} \\
\Longrightarrow & \overset{d}{1}\ \neg 2\ 3 & \| & \neg 1 \vee \neg 2,\ 2 \vee 3,\ \neg 1 \vee \neg 3 \vee 4,\ 2 \vee \neg 3 \vee \neg 4,\ 1 \vee 4 & \text{unit propagate} \\
\Longrightarrow & \overset{d}{1}\ \neg 2\ 3\ 4 & \| & \neg 1 \vee \neg 2,\ 2 \vee 3,\ \neg 1 \vee \neg 3 \vee 4,\ 2 \vee \neg 3 \vee \neg 4,\ 1 \vee 4 & \text{unit propagate} \\
\Longrightarrow & \neg 1 & \| & \neg 1 \vee \neg 2,\ 2 \vee 3,\ \neg 1 \vee \neg 3 \vee 4,\ 2 \vee \neg 3 \vee \neg 4,\ 1 \vee 4 & \text{backtrack} \\
\Longrightarrow & \neg 1\ 4 & \| & \neg 1 \vee \neg 2,\ 2 \vee 3,\ \neg 1 \vee \neg 3 \vee 4,\ 2 \vee \neg 3 \vee \neg 4,\ 1 \vee 4 & \text{unit propagate} \\
\Longrightarrow & \neg 1\ 4\ \overset{d}{\neg 3} & \| & \neg 1 \vee \neg 2,\ 2 \vee 3,\ \neg 1 \vee \neg 3 \vee 4,\ 2 \vee \neg 3 \vee \neg 4,\ 1 \vee 4 & \text{decide} \\
\Longrightarrow & \neg 1\ 4\ \overset{d}{\neg 3}\ 2 & \| & \neg 1 \vee \neg 2,\ 2 \vee 3,\ \neg 1 \vee \neg 3 \vee 4,\ 2 \vee \neg 3 \vee \neg 4,\ 1 \vee 4 & \text{unit propagate}
\end{array}
$$

Definition (Transition Rules)

- unit propagate $M \parallel F, C \vee l \implies M\, l \parallel F, C \vee l$

  if $M \vDash \neg C$ and l is undefined in M unit clause

- pure literal $M \parallel F \implies M\, l \parallel F$

  if l occurs in F and l^c does not occur in F and l is undefined in M

- decide $M \parallel F \implies M\, \overset{d}{l} \parallel F$

  if l or l^c occurs in F and l is undefined in M

- fail $M \parallel F, C \implies$ fail-state

  if $M \vDash \neg C$ and M contains no decision literals

- backtrack $M\, \overset{d}{l}\, N \parallel F, C \implies M\, l^c \parallel F, C$

  if $M\, \overset{d}{l}\, N \vDash \neg C$ and N contains no decision literals

Outline

Example

$\phi = (\neg 1 \vee 2) \wedge (\neg 3 \vee 4) \wedge (\neg 5 \vee \neg 6) \wedge (6 \vee \neg 5 \vee \neg 2)$

$$
\begin{array}{rrll}
 & & \| \ \neg 1 \vee 2,\ \neg 3 \vee 4,\ \neg 5 \vee \neg 6,\ 6 \vee \neg 5 \vee \neg 2 & \\
\Longrightarrow & \overset{d}{1} & \| \ \neg 1 \vee 2,\ \neg 3 \vee 4,\ \neg 5 \vee \neg 6,\ 6 \vee \neg 5 \vee \neg 2 & \text{decide} \\
\Longrightarrow & \overset{d}{1}\ 2 & \| \ \neg 1 \vee 2,\ \neg 3 \vee 4,\ \neg 5 \vee \neg 6,\ 6 \vee \neg 5 \vee \neg 2 & \text{unit propagate} \\
\Longrightarrow & \overset{d}{1}\ 2\ \overset{d}{3} & \| \ \neg 1 \vee 2,\ \neg 3 \vee 4,\ \neg 5 \vee \neg 6,\ 6 \vee \neg 5 \vee \neg 2 & \text{decide} \\
\Longrightarrow & \overset{d}{1}\ 2\ \overset{d}{3}\ 4 & \| \ \neg 1 \vee 2,\ \neg 3 \vee 4,\ \neg 5 \vee \neg 6,\ 6 \vee \neg 5 \vee \neg 2 & \text{unit propagate} \\
\Longrightarrow & \overset{d}{1}\ 2\ \overset{d}{3}\ 4\ \overset{d}{5} & \| \ \neg 1 \vee 2,\ \neg 3 \vee 4,\ \neg 5 \vee \neg 6,\ 6 \vee \neg 5 \vee \neg 2 & \text{decide} \\
\Longrightarrow & \overset{d}{1}\ 2\ \overset{d}{3}\ 4\ \overset{d}{5}\ \neg 6 & \| \ \neg 1 \vee 2,\ \neg 3 \vee 4,\ \neg 5 \vee \neg 6,\ 6 \vee \neg 5 \vee \neg 2 & \text{unit propagate} \\
\Longrightarrow & \overset{d}{1}\ 2\ \neg 5 & \| \ \neg 1 \vee 2,\ \neg 3 \vee 4,\ \neg 5 \vee \neg 6,\ 6 \vee \neg 5 \vee \neg 2 & \text{backjump}
\end{array}
$$

conflict is due to $\overset{d}{1}\ 2$ and $\overset{d}{5}\ \neg 6$ hence $\neg 1 \vee \neg 5$ can be inferred

Definitions

- backtrack $\quad M \overset{d}{l} N \parallel F, C \implies M\, l^c \parallel F, C$

  if $M \overset{d}{l} N \vDash \neg C$ and N contains no decision literals

- backjump $\quad M \overset{d}{l} N \parallel F, C \implies M\, l' \parallel F, C$

  if $M \overset{d}{l} N \vDash \neg C$ and $\exists$ clause $C' \vee l'$ such that

  backjump clause

  - $F, C \vDash C' \vee l'$
  - $M \vDash \neg C'$
  - l' is undefined in M
  - l' or l'^c occurs in F or in $M \overset{d}{l} N$

Example (cont'd)

$\neg 1 \vee \neg 5$ and $\neg 2 \vee \neg 5$ are backjump clauses with respect to $\overset{d}{1}\ 2\ \overset{d}{3}\ 4\ \overset{d}{5}\ \neg 6 \parallel \phi$

Definition

basic DPLL $\mathcal{B}$ consists of transition rules

- unit propagate $\quad M \parallel F, C \vee l \implies M\, l \parallel F, C \vee l$

  if $M \vDash \neg C$ and l is undefined in M

- decide $\quad M \parallel F \implies M\, l^{d} \parallel F$

  if l or l^c occurs in F and l is undefined in M

- fail $\quad M \parallel F, C \implies$ fail-state

  if $M \vDash \neg C$ and M contains no decision literals

- backjump $\quad M\, l^{d}\, N \parallel F, C \implies M\, l' \parallel F, C$

  if $M\, l^{d}\, N \vDash \neg C$ and $\exists$ clause $C' \vee l'$ such that

  - $F, C \vDash C' \vee l'$ and $M \vDash \neg C'$
  - l' is undefined in M and l' or l'^{c} occurs in F or in $M\, l^{d}\, N$

Theorem

there are no infinite derivations $\| F \Longrightarrow_{\mathcal{B}} S_1 \Longrightarrow_{\mathcal{B}} S_2 \Longrightarrow_{\mathcal{B}} \cdots$

Proof

- for list of distinct literals M, $|M|$ is length of M
- measure state $M_0 \overset{d}{l_1} M_1 \overset{d}{l_2} M_2 \dots \overset{d}{l_k} M_k \| F$ where $M_0, \dots, M_k$ contain no decision literals by tuple $(|M_0|, |M_1|, \dots, |M_k|)$
- compare tuples lexicographically using standard order on $\mathbb{N}$
- every transition step strictly increases measure
- measure is bounded by $(n+1)$-tuple $(n, \dots, n)$ where n is total number of atoms

Example

$\| \phi = (\neg 1 \vee 2) \wedge (\neg 3 \vee 4) \wedge (\neg 5 \vee \neg 6) \wedge (6 \vee \neg 5 \vee \neg 2)$ (0)

$\Longrightarrow \; 1^d \| \phi$ decide $(0,0)$

$\Longrightarrow \; 1^d\, 2 \| \phi$ unit propagate $(0,1)$

$\Longrightarrow \; 1^d\, 2\, 3^d \| \phi$ decide $(0,1,0)$

$\Longrightarrow \; 1^d\, 2\, 3^d\, 4 \| \phi$ unit propagate $(0,1,1)$

$\Longrightarrow \; 1^d\, 2\, 3^d\, 4\, 5^d \| \phi$ decide $(0,1,1,0)$

$\Longrightarrow \; 1^d\, 2\, 3^d\, 4\, 5^d\, \neg 6 \| \phi$ unit propagate $(0,1,1,1)$

$\Longrightarrow \; 1^d\, 2\, \neg 5 \| \phi$ backjump $(0,2)$

- decide $(m_0, \dots, m_i) <_{\text{lex}} (m_0, \dots, m_i, 0)$
- unit propagate $(m_0, \dots, m_i) <_{\text{lex}} (m_0, \dots, m_i + 1)$
- backjump $(m_0, \dots, m_i) <_{\text{lex}} (m_0, \dots, m_j + 1)$ with $j < i$

Lemma

1. if $\| F \Longrightarrow_{\mathcal{B}}^{*} M \| F'$ then
   - $F = F'$
   - M does not contain complementary literals
   - M consists of distinct literals

2. if $\| F \Longrightarrow_{\mathcal{B}}^{*} M_0 \, l_1^d \, M_1 \, l_2^d \, M_2 \cdots l_k^d \, M_k \| F$ with no decision literals in $M_0, \dots, M_k$

   then $F, l_1, \dots, l_i \vDash M_i$ for all $0 \leqslant i \leqslant k$

Theorem

if $\Vert F \Longrightarrow_{\mathcal{B}} S_1 \Longrightarrow_{\mathcal{B}} \cdots \Longrightarrow_{\mathcal{B}} S_n \not\Longrightarrow_{\mathcal{B}}$ then

1. $S_n =$ fail-state if and only if F is unsatisfiable
2. $S_n = M \Vert F'$ only if F is satisfiable and $M \vDash F$

Proof

1. (only if) $\Vert F \Longrightarrow_{\mathcal{B}}^{*} M \Vert F \Longrightarrow_{\text{fail}}$ fail-state
   - M contains no decision literals and $M \vDash \neg C$ for some C in F
   - $F \vDash C$ and $F \vDash M$ and thus $F \vDash \neg C$ and thus F is unsatisfiable
2. $\Vert F \Longrightarrow_{\mathcal{B}}^{*} M \Vert F' \not\Longrightarrow_{\mathcal{B}}$
   - $F = F'$ and all literals in F are defined in M, otherwise decide is applicable
   - F contains no clause C such that $M \vDash \neg C$, otherwise backjump or fail is applicable
   - $M \vDash F$ and thus F is satisfiable

Terminology

non-chronological backtracking or conflict-driven backtracking

Question

how to find good backjump clauses ?

Answer

use conflict graph

Observation

restarts are useful to avoid wasting too much time in parts of search space without satisfying assignments

- restart $\qquad M \parallel F \quad \Longrightarrow \quad \parallel F$

Final Remarks

- restarts do not compromise completeness if number of steps between consecutive restarts strictly increases
- modern SAT solvers additionally incorporate
  - heuristics for selecting next decision literal
  - special data structures that allow for efficient unit propagation

Outline

1. Introduction
2. Propositional Logic – Review
3. Tseitin's Transformation
4. DPLL
5. **Further Reading**

Kröning and Strichmann

- Chapter 1
- Chapter 2

Further Reading

- David A. Plaisted and Steven Greenbaum
  A Structure-Preserving Clause Form Translation
  Journal of Symbolic Computation 2(3), pp. 293–304, 1986
- Martin Davis and Hilary Putnam
  A Computing Procedure for Quantification Theory
  Journal of the ACM 7(3), pp. 201–215, 1960
- Martin Davis, George Logemann, and Donald Loveland
  A Machine Program for Theorem-Proving
  Communications of the ACM 5(7), pp. 394–397, 1962

Further Reading (cont'd)

- Robert Nieuwenhuis, Albert Oliveras, and Cesare Tinelli
  Solving SAT and SAT Modulo Theories: From an Abstract Davis–Putnam–Logemann–Loveland Procedure to DPLL(T)
  Journal of the ACM 53(6), pp. 937 – 977, 2006
- Moshe Y. Vardi
  Boolean Satisfiability: Theory and Engineering
  Communications of the ACM 57(3), editor's letter, 2014

Important Concepts

- abstract DPLL
- atom
- basic DPLL
- backjump
- backtrack
- bottom
- clause
- complementary literals
- conflict graph
- conjunction
- conjunctive normal form
- cut
- decide
- disjunction
- disjunctive normal form
- equisatisfiability
- fail-state
- implication
- literal
- negation
- polarity
- pure literal
- restart
- right-associativity
- satisfiability
- semantic entailment
- semantic equivalence
- Tseitin’s transformation
- tautology
- top
- truth table
- truth values
- unique implication point (UIP)
- unit propagation
- validity
- valuation