Efficient Algorithms for Frequently Asked Questions

8. Solving SAT

Prof. Dan Olteanu

DaST Data • (Systems+Theory)

University of Zurich UZH

May 9, 2022

https://lms.uzh.ch/url/RepositoryEntry/17185308706

Agenda for This Lecture

We look at SATisfiability of CNF formulas through FAQ glasses

- Classical SAT solver: The DPLL procedure
    - Logical Resolution
- Connection to solving FAQs over the Boolean semiring

SAT instances with acyclic hypergraphs

- Are α-acyclic SAT instances solvable efficiently?
- Solving β-acyclic SAT instances efficiently

SAT

SATisfiability: Given a CNF formula F over Boolean variables, is F satisfiable?

Example: Consider the Boolean formula F over variables x_1, x_2, x_3, x_4:

$$F = (x_1 \vee \neg x_2) \wedge (x_2 \vee x_3 \vee \neg x_4) \wedge (\neg x_2 \vee \neg x_3)$$

SAT

SATisfiability: Given a CNF formula F over Boolean variables, is F satisfiable?

Example: Consider the Boolean formula F over variables x_1, x_2, x_3, x_4:

$$F = (x_1 \vee \neg x_2) \wedge (x_2 \vee x_3 \vee \neg x_4) \wedge (\neg x_2 \vee \neg x_3)$$

- F is a conjunction ($\wedge$) of clauses, each clause is a disjunction ($\vee$) of literals
  - Example of clause: $(x_1 \vee \neg x_2)$
  - Unit-clauses only consist of a single literal, e.g., (x_3)
  - Tautological clauses are always true, regardless of variable assignment, e.g., $(x_1 \vee \neg x_2 \vee \neg x_1)$

SAT

SATisfiability: Given a CNF formula F over Boolean variables, is F satisfiable?

Example: Consider the Boolean formula F over variables x_1, x_2, x_3, x_4:

$$F = (x_1 \vee \neg x_2) \wedge (x_2 \vee x_3 \vee \neg x_4) \wedge (\neg x_2 \vee \neg x_3)$$

- F is a conjunction ($\wedge$) of clauses, each clause is a disjunction ($\vee$) of literals
  - Example of clause: $(x_1 \vee \neg x_2)$
  - Unit-clauses only consist of a single literal, e.g., (x_3)
  - Tautological clauses are always true, regardless of variable assignment, e.g., $(x_1 \vee \neg x_2 \vee \neg x_1)$
- Each literal is an occurrence of a variable either positively or negatively
  - Example of literals: x_2 or $\neg x_2$
  - Single-phase variables occur either only positively or only negatively, e.g., $\neg x_4$

SAT

SATisfiability: Given a CNF formula F over Boolean variables, is F satisfiable?

Example: Consider the Boolean formula F over variables x_1, x_2, x_3, x_4:

$$F = (x_1 \lor \neg x_2) \land (x_2 \lor x_3 \lor \neg x_4) \land (\neg x_2 \lor \neg x_3)$$

- F is a conjunction ($\land$) of clauses, each clause is a disjunction ($\lor$) of literals
  - Example of clause: $(x_1 \lor \neg x_2)$
  - Unit-clauses only consist of a single literal, e.g., (x_3)
  - Tautological clauses are always true, regardless of variable assignment, e.g., $(x_1 \lor \neg x_2 \lor \neg x_1)$
- Each literal is an occurrence of a variable either positively or negatively
  - Example of literals: x_2 or $\neg x_2$
  - Single-phase variables occur either only positively or only negatively, e.g., $\neg x_4$
- Possible satisfying assignment: $x_2 = 0$, $x_3 = 1$, anything else for x_1, x_4

SAT as FAQ

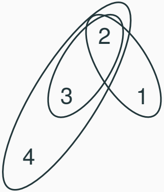
Any SAT instance can be immediately encoded in FAQ over the Boolean semiring

- Each variable in the CNF formula becomes a variable in the FAQ expression
- One factor per clause, mapping (non-)satisfying assignments to 1 (resp. 0)

$$F = \underbrace{(x_1 \vee \neg x_2)}_{\psi_{12}(x_1, x_2)} \wedge \underbrace{(x_2 \vee x_3 \vee \neg x_4)}_{\psi_{234}(x_2, x_3, x_4)} \wedge \underbrace{(\neg x_2 \vee \neg x_3)}_{\psi_{23}(x_2, x_3)}$$

$$\phi() = \bigvee_{x_1, x_2, x_3, x_4} \psi_{12}(x_1, x_2) \wedge \psi_{234}(x_2, x_3, x_4) \wedge \psi_{23}(x_2, x_3)$$

- Hypergraph: One hyperedge per clause, one node per variable (disregard $\neg$)



Representation of Factors for Clauses (1/2)

Trivial representation: Truth table of variables in the clause

- The factor corresponding to a clause has one tuple per satisfying assignment of the variables
- Example: The clause $(x_2 \vee x_3 \vee \neg x_4)$ is represented by the factor

x_2	x_3	x_4	$\psi_{234}(x_2, x_3, x_4)$
0	0	0	1
0	1	0	1
1	0	0	1
1	1	0	1
0	1	1	1
1	0	1	1
1	1	1	1

The only assignment that is not satisfying: $x_2 = 0, x_3 = 0, x_4 = 1$

Problems with this representation:

- For a clause with n variables, the factor can have up to 2^n tuples
- Yannakakis/LFTJ take time proportional to factor sizes, so exponential in n

Representation of Factors for Clauses (2/2)

Compact, natural representation: The clause itself

- + Only takes $O(n)$ size, where n is the number of variables
- − Cannot represent arbitrary relationships between the variables
  - Cannot represent the result of semi-join reduction used by Yannakakis
  - Cannot represent factors defined by marginalisation of variables over clauses
  - Can only represent a disjunction of literals

Representation of Factors for Clauses (2/2)

Compact, natural representation: The clause itself

- \+ Only takes $O(n)$ size, where n is the number of variables
- – Cannot represent arbitrary relationships between the variables
  - Cannot represent the result of semi-join reduction used by Yannakakis
  - Cannot represent factors defined by marginalisation of variables over clauses
  - Can only represent a disjunction of literals

We want a variable-marginalisation algorithm, much like LFTJ

- Marginalise out one variable at a time
  - Special case: Single-phase variables
  - General case: Resolution
- Special case for clauses: Conjunction of contradicting unit-clauses
- Special case for clauses: Tautological clauses

The DPLL Procedure

The Davis-Putnam (DP) Algorithm (1960): Building Block 1/4

1. Find every single-phase variable and eliminate its clauses

$$(\neg x_1 \vee \neg x_2 \vee \neg x_4) \wedge (x_1 \vee \neg x_3 \vee \neg x_4) \wedge (\neg x_1 \vee \neg x_2 \vee x_4) \wedge (\neg x_1 \vee \neg x_4 \vee \neg x_5) \wedge (\neg x_1 \vee x_3 \vee x_5)$$

Variable x_2 only occurs negatively: Set $\neg x_2 = 1$ and eliminate the clauses

$$(\neg x_1 \vee \neg x_2 \vee \neg x_4) \wedge (x_1 \vee \neg x_3 \vee \neg x_4) \wedge (\neg x_1 \vee \neg x_2 \vee x_4) \wedge (\neg x_1 \vee \neg x_4 \vee \neg x_5) \wedge (\neg x_1 \vee x_3 \vee x_5)$$

The Davis-Putnam (DP) Algorithm (1960): Building Block 1/4

1. Find every single-phase variable and eliminate its clauses

$$(\neg x_1 \vee \neg x_2 \vee \neg x_4) \wedge (x_1 \vee \neg x_3 \vee \neg x_4) \wedge (\neg x_1 \vee \neg x_2 \vee x_4) \wedge (\neg x_1 \vee \neg x_4 \vee \neg x_5) \wedge (\neg x_1 \vee x_3 \vee x_5)$$

Variable x_2 only occurs negatively: Set $\neg x_2 = 1$ and eliminate the clauses

$$(\neg x_1 \vee \neg x_2 \vee \neg x_4) \wedge (x_1 \vee \neg x_3 \vee \neg x_4) \wedge (\neg x_1 \vee \neg x_2 \vee x_4) \wedge (\neg x_1 \vee \neg x_4 \vee \neg x_5) \wedge (\neg x_1 \vee x_3 \vee x_5)$$

$$(\neg x_1 \vee \neg x_2 \vee \neg x_4) \wedge (x_1 \vee \neg x_3 \vee \neg x_4) \wedge (\neg x_1 \vee \neg x_2 \vee x_4) \wedge (\neg x_1 \vee \neg x_4 \vee \neg x_5) \wedge (\neg x_1 \vee x_3 \vee x_5)$$

The Davis-Putnam (DP) Algorithm (1960): Building Block 1/4

1. Find every single-phase variable and eliminate its clauses

$$(\neg x_1 \vee \neg x_2 \vee \neg x_4) \wedge (x_1 \vee \neg x_3 \vee \neg x_4) \wedge (\neg x_1 \vee \neg x_2 \vee x_4) \wedge (\neg x_1 \vee \neg x_4 \vee \neg x_5) \wedge (\neg x_1 \vee x_3 \vee x_5)$$

Variable x_2 only occurs negatively: Set $\neg x_2 = 1$ and eliminate the clauses

$$(\neg x_1 \vee \neg x_2 \vee \neg x_4) \wedge (x_1 \vee \neg x_3 \vee \neg x_4) \wedge (\neg x_1 \vee \neg x_2 \vee x_4) \wedge (\neg x_1 \vee \neg x_4 \vee \neg x_5) \wedge (\neg x_1 \vee x_3 \vee x_5)$$

$$(x_1 \vee \neg x_3 \vee \neg x_4) \wedge (\neg x_1 \vee \neg x_4 \vee \neg x_5) \wedge (\neg x_1 \vee x_3 \vee x_5)$$

This filtering done in time linear in the number of clauses and variables

- Simulated by plain variable marginalisation in FAQ:

$$\psi_{134}(x_1, x_3, x_4) \wedge \psi_{135}(x_1, x_3, x_5) \wedge \psi_{145}(x_1, x_4, x_5) \wedge \underbrace{\bigvee_{x_2} \psi_{124}(x_1, x_2, x_4) \wedge \psi'_{124}(x_1, x_2, x_4)}_{1}$$

The Davis-Putnam (DP) Algorithm (1960): Building Block 2/4

2. Eliminate tautological clauses

The clause $(\neg x_2 \vee x_2 \vee x_1 \vee x_3 \vee x_4)$ evaluates to 1 for any variable assignment

What does tautology correspond to in the general FAQ world?

- Corresponding factor does not filter out any possible values for its variables
- In DB: Factor is Cartesian product of the active domains of its variables

The Davis-Putnam (DP) Algorithm (1960): Building Block 3/4

3. Identify unit-clause contradictions

$$(\neg x_1 \vee \neg x_2 \vee \neg x_4) \wedge (\neg x_3) \wedge (\neg x_1 \vee \neg x_2 \vee x_4) \wedge (\neg x_1 \vee \neg x_4 \vee \neg x_5) \wedge (x_3)$$

There are two contradicting unit clauses: $(\neg x_3)$ and (x_3).

$$(\neg x_1 \vee \neg x_2 \vee \neg x_4) \wedge (\neg x_3) \wedge (\neg x_1 \vee \neg x_2 \vee x_4) \wedge (\neg x_1 \vee \neg x_4 \vee \neg x_5) \wedge (x_3)$$

The conjunction of $(\neg x_3)$ and (x_3), and the entire formula, always evaluates to 0.

$$\begin{aligned} &(\neg x_1 \vee \neg x_2 \vee \neg x_4) \wedge (\neg x_3) \wedge (\neg x_1 \vee \neg x_2 \vee x_4) \wedge (\neg x_1 \vee \neg x_4 \vee \neg x_5) \wedge (x_3) \\ &= (\neg x_1 \vee \neg x_2 \vee \neg x_4) \wedge (\neg x_1 \vee \neg x_2 \vee x_4) \wedge (\neg x_1 \vee \neg x_4 \vee \neg x_5) \wedge 0 = 0 \end{aligned}$$

The Davis-Putnam (DP) Algorithm (1960): Building Block 3/4

3. Identify unit-clause contradictions

$(\neg x_1 \vee \neg x_2 \vee \neg x_4) \wedge (\neg x_3) \wedge (\neg x_1 \vee \neg x_2 \vee x_4) \wedge (\neg x_1 \vee \neg x_4 \vee \neg x_5) \wedge (x_3)$

There are two contradicting unit clauses: $(\neg x_3)$ and (x_3).

$(\neg x_1 \vee \neg x_2 \vee \neg x_4) \wedge (\neg x_3) \wedge (\neg x_1 \vee \neg x_2 \vee x_4) \wedge (\neg x_1 \vee \neg x_4 \vee \neg x_5) \wedge (x_3)$

The conjunction of $(\neg x_3)$ and (x_3), and the entire formula, always evaluates to 0.

$$
\begin{aligned}
&(\neg x_1 \vee \neg x_2 \vee \neg x_4) \wedge (\neg x_3) \wedge (\neg x_1 \vee \neg x_2 \vee x_4) \wedge (\neg x_1 \vee \neg x_4 \vee \neg x_5) \wedge (x_3) \\
&= (\neg x_1 \vee \neg x_2 \vee \neg x_4) \wedge (\neg x_1 \vee \neg x_2 \vee x_4) \wedge (\neg x_1 \vee \neg x_4 \vee \neg x_5) \wedge 0 = 0
\end{aligned}
$$

This building block can be simulated using full reducers in Yannakakis

- Semi-join reduction of the factor for the clause (x_3) using the factor for the clause $(\neg x_3)$ yields the factor representing the constant 0

The Davis-Putnam (DP) Algorithm (1960): Building Block 4/4

4. Eliminate a non-single-phase variable by resolution

Evaluate $F = (x \vee \alpha) \wedge (\neg x \vee \beta)$, where α, β are disjunctions of literals without x.

A. Marginalisation: Marginalise x in F $\underbrace{\big((0 \vee \alpha) \wedge (1 \vee \beta)\big)}_{(x=0)\wedge\alpha} \vee \underbrace{\big((1 \vee \alpha) \wedge (0 \vee \beta)\big)}_{(x=1)\wedge\beta}$

- We obtain the formula $(\neg x \wedge \alpha) \vee (x \wedge \beta)$ equivalent with F
- We proceed with the evaluation of α in case $x = 0$ and of β in case $x = 1$

The Davis-Putnam (DP) Algorithm (1960): Building Block 4/4

4. Eliminate a non-single-phase variable by resolution

Evaluate $F = (x \vee \alpha) \wedge (\neg x \vee \beta)$, where α, β are disjunctions of literals without x.

A. Marginalisation: Marginalise x in F $\underbrace{((0 \vee \alpha) \wedge (1 \vee \beta))}_{(x=0) \wedge \alpha} \vee \underbrace{((1 \vee \alpha) \wedge (0 \vee \beta))}_{(x=1) \wedge \beta}$

- We obtain the formula $(\neg x \wedge \alpha) \vee (x \wedge \beta)$ equivalent with F
- We proceed with the evaluation of α in case $x = 0$ and of β in case $x = 1$

B. Resolution: Replace F by equi-satisfiable resolvent clause $(\alpha \vee \beta)$

$(x \vee \alpha) \wedge (\neg x \vee \beta)$ is satisfiable if and only if $(\alpha \vee \beta)$ is satisfiable

x	α	β	F	$(\alpha \vee \beta)$
0	0	0	0	0
0	0	1	0	1
0	1	0	1	1
0	1	1	1	1

x	α	β	F	$(\alpha \vee \beta)$
1	0	0	0	0
1	0	1	1	1
1	1	0	0	1
1	1	1	1	1

More on Resolution

Replace $(x \vee \alpha) \wedge (\neg x \vee \beta)$ by equi-satisfiable clause $(\alpha \vee \beta)$

General case: Formula has n clauses $(x \vee \alpha_i)$ and m clauses $(\neg x \vee \beta_j)$

- $\forall i \in [n], j \in [m]$: Conjunction $(x \vee \alpha_i) \wedge (\neg x \vee \beta_j)$ has resolvent $(\alpha_i \vee \beta_j)$
- We replace $\bigwedge_{i \in [n]} (x \vee \alpha_i) \wedge \bigwedge_{j \in [m]} (\neg x \vee \beta_j)$ by $\bigwedge_{i \in [n], j \in [m]} (\alpha_i \vee \beta_j)$
- The new and old formulas are equi-satisfiable
- Variable x does not occur anymore in the new formula

More on Resolution

Replace $(x \vee \alpha) \wedge (\neg x \vee \beta)$ by equi-satisfiable clause $(\alpha \vee \beta)$

General case: Formula has n clauses $(x \vee \alpha_i)$ and m clauses $(\neg x \vee \beta_j)$

- $\forall i \in [n], j \in [m]$: Conjunction $(x \vee \alpha_i) \wedge (\neg x \vee \beta_j)$ has resolvent $(\alpha_i \vee \beta_j)$
- We replace $\bigwedge_{i \in [n]}(x \vee \alpha_i) \wedge \bigwedge_{j \in [m]}(\neg x \vee \beta_j)$ by $\bigwedge_{i \in [n], j \in [m]}(\alpha_i \vee \beta_j)$
- The new and old formulas are equi-satisfiable
- Variable x does not occur anymore in the new formula

Complexity:

- For each variable x, we replace $n + m$ clauses by $n \cdot m$ resolvent clauses
- The complexity can be exponential in number of variables
- Exponential time unavoidable in worst case
- Polynomial time possible for 2SAT, β-acyclic SAT, Horn clauses, ...

The DP Algorithm: Putting the Building Blocks Together

Algorithm DP (CNF Formula F)

1. **if** F is empty (i.e., has no clause) **then return** 1 // Satisfiable
2. **if** F has a unit-clause contradiction **then return** 0 // Unsatisfiable
3. **if** F has single-phase variables **then** remove their clauses from F
   // These clauses can be made true

// Next eliminate a variable and replace its clauses by resolvents

4. Pick a remaining variable x
5. F' = empty-set
6. **for each** pair of clauses $(x \vee \alpha_i)$ and $(\neg x \vee \beta_j)$ in F **do**
7.   **if** $(\alpha_i \vee \beta_j)$ is not tautological **then** add $(\alpha_i \vee \beta_j)$ to F' // Resolution
8. Remove all clauses containing x or $\neg x$ from F and add to F all clauses in F'
9. **return** DP (F)

The DP Algorithm: Running Example

$$\underbrace{(\neg x_1 \vee \neg x_2 \vee \neg x_4)}_{c_1} \wedge \underbrace{(x_1 \vee \neg x_3 \vee \neg x_4)}_{c_2} \wedge \underbrace{(\neg x_1 \vee \neg x_2 \vee x_4)}_{c_3} \wedge \underbrace{(\neg x_1 \vee \neg x_4 \vee \neg x_5)}_{c_4} \wedge \underbrace{(\neg x_1 \vee x_2 \vee x_3 \vee x_5)}_{c_5}$$

$\underline{\text{DP}}(c_1 \wedge c_2 \wedge c_3 \wedge c_4 \wedge c_5)$

The DP Algorithm: Running Example

$$\underbrace{(\neg x_1 \vee \neg x_2 \vee \neg x_4)}_{c_1} \wedge \underbrace{(x_1 \vee \neg x_3 \vee \neg x_4)}_{c_2} \wedge \underbrace{(\neg x_1 \vee \neg x_2 \vee x_4)}_{c_3} \wedge \underbrace{(\neg x_1 \vee \neg x_4 \vee \neg x_5)}_{c_4} \wedge \underbrace{(\neg x_1 \vee x_2 \vee x_3 \vee x_5)}_{c_5}$$

DP($c_1 \wedge c_2 \wedge c_3 \wedge c_4 \wedge c_5$)

No unit-clause contradiction, no single-phase variable

The DP Algorithm: Running Example

$$\underbrace{(\neg x_1 \vee \neg x_2 \vee \neg x_4)}_{c_1} \wedge \underbrace{(x_1 \vee \neg x_3 \vee \neg x_4)}_{c_2} \wedge \underbrace{(\neg x_1 \vee \neg x_2 \vee x_4)}_{c_3} \wedge \underbrace{(\neg x_1 \vee \neg x_4 \vee \neg x_5)}_{c_4} \wedge \underbrace{(\neg x_1 \vee x_2 \vee x_3 \vee x_5)}_{c_5}$$

<u>DP</u>($c_1 \wedge c_2 \wedge c_3 \wedge c_4 \wedge c_5$)

No unit-clause contradiction, no single-phase variable

Pick x_1

The DP Algorithm: Running Example

$$\underbrace{(\neg x_1 \vee \neg x_2 \vee \neg x_4)}_{c_1} \wedge \underbrace{(x_1 \vee \neg x_3 \vee \neg x_4)}_{c_2} \wedge \underbrace{(\neg x_1 \vee \neg x_2 \vee x_4)}_{c_3} \wedge \underbrace{(\neg x_1 \vee \neg x_4 \vee \neg x_5)}_{c_4} \wedge \underbrace{(\neg x_1 \vee x_2 \vee x_3 \vee x_5)}_{c_5}$$

DP($c_1 \wedge c_2 \wedge c_3 \wedge c_4 \wedge c_5$)

No unit-clause contradiction, no single-phase variable

Pick x_1

Add resolvent: $(\neg x_3 \vee \neg x_4 \vee \neg x_2 \vee \neg x_4)$ for $c_2 \wedge c_1$

Resolvent: $(\neg x_3 \vee \neg x_4 \vee \neg x_2 \vee x_4)$ for $c_2 \wedge c_3$ is tautological, not added

Add resolvent: $(\neg x_3 \vee \neg x_4 \vee \neg x_4 \vee \neg x_5)$ for $c_2 \wedge c_4$

Resolvent: $(\neg x_3 \vee \neg x_4 \vee x_2 \vee x_3 \vee x_5)$ for $c_2 \wedge c_5$ is tautological, not added

The DP Algorithm: Running Example

$$\underbrace{(\neg x_1 \vee \neg x_2 \vee \neg x_4)}_{c_1} \wedge \underbrace{(x_1 \vee \neg x_3 \vee \neg x_4)}_{c_2} \wedge \underbrace{(\neg x_1 \vee \neg x_2 \vee x_4)}_{c_3} \wedge \underbrace{(\neg x_1 \vee \neg x_4 \vee \neg x_5)}_{c_4} \wedge \underbrace{(\neg x_1 \vee x_2 \vee x_3 \vee x_5)}_{c_5}$$

$\underline{\text{DP}}(c_1 \wedge c_2 \wedge c_3 \wedge c_4 \wedge c_5)$

No unit-clause contradiction, no single-phase variable

Pick x_1

Add resolvent: $(\neg x_3 \vee \neg x_4 \vee \neg x_2 \vee \neg x_4)$ for $c_2 \wedge c_1$

Resolvent: $(\neg x_3 \vee \neg x_4 \vee \neg x_2 \vee x_4)$ for $c_2 \wedge c_3$ is tautological, not added

Add resolvent: $(\neg x_3 \vee \neg x_4 \vee \neg x_4 \vee \neg x_5)$ for $c_2 \wedge c_4$

Resolvent: $(\neg x_3 \vee \neg x_4 \vee x_2 \vee x_3 \vee x_5)$ for $c_2 \wedge c_5$ is tautological, not added

$\underline{\text{DP}}\big((\neg x_3 \vee \neg x_4 \vee \neg x_2 \vee \neg x_4) \wedge (\neg x_3 \vee \neg x_4 \vee \neg x_4 \vee \neg x_5)\big)$

The DP Algorithm: Running Example

$$\underbrace{(\neg x_1 \vee \neg x_2 \vee \neg x_4)}_{c_1} \wedge \underbrace{(x_1 \vee \neg x_3 \vee \neg x_4)}_{c_2} \wedge \underbrace{(\neg x_1 \vee \neg x_2 \vee x_4)}_{c_3} \wedge \underbrace{(\neg x_1 \vee \neg x_4 \vee \neg x_5)}_{c_4} \wedge \underbrace{(\neg x_1 \vee x_2 \vee x_3 \vee x_5)}_{c_5}$$

DP($c_1 \wedge c_2 \wedge c_3 \wedge c_4 \wedge c_5$)

No unit-clause contradiction, no single-phase variable

Pick x_1

Add resolvent: $(\neg x_3 \vee \neg x_4 \vee \neg x_2 \vee \neg x_4)$ for $c_2 \wedge c_1$

Resolvent: $(\neg x_3 \vee \neg x_4 \vee \neg x_2 \vee x_4)$ for $c_2 \wedge c_3$ is tautological, not added

Add resolvent: $(\neg x_3 \vee \neg x_4 \vee \neg x_4 \vee \neg x_5)$ for $c_2 \wedge c_4$

Resolvent: $(\neg x_3 \vee \neg x_4 \vee x_2 \vee x_3 \vee x_5)$ for $c_2 \wedge c_5$ is tautological, not added

DP$\big((\neg x_3 \vee \neg x_4 \vee \neg x_2 \vee \neg x_4) \wedge (\neg x_3 \vee \neg x_4 \vee \neg x_4 \vee \neg x_5)\big)$

Single-phase variables: Set $\neg x_2 = \neg x_3 = \neg x_4 = \neg x_5 = 1$

The DP Algorithm: Running Example

$$\underbrace{(\neg x_1 \vee \neg x_2 \vee \neg x_4)}_{c_1} \wedge \underbrace{(x_1 \vee \neg x_3 \vee \neg x_4)}_{c_2} \wedge \underbrace{(\neg x_1 \vee \neg x_2 \vee x_4)}_{c_3} \wedge \underbrace{(\neg x_1 \vee \neg x_4 \vee \neg x_5)}_{c_4} \wedge \underbrace{(\neg x_1 \vee x_2 \vee x_3 \vee x_5)}_{c_5}$$

<u>DP</u>$(c_1 \wedge c_2 \wedge c_3 \wedge c_4 \wedge c_5)$

No unit-clause contradiction, no single-phase variable

Pick x_1

Add resolvent: $(\neg x_3 \vee \neg x_4 \vee \neg x_2 \vee \neg x_4)$ for $c_2 \wedge c_1$

Resolvent: $(\neg x_3 \vee \neg x_4 \vee \neg x_2 \vee x_4)$ for $c_2 \wedge c_3$ is tautological, not added

Add resolvent: $(\neg x_3 \vee \neg x_4 \vee \neg x_4 \vee \neg x_5)$ for $c_2 \wedge c_4$

Resolvent: $(\neg x_3 \vee \neg x_4 \vee x_2 \vee x_3 \vee x_5)$ for $c_2 \wedge c_5$ is tautological, not added

<u>DP</u>$\big((\neg x_3 \vee \neg x_4 \vee \neg x_2 \vee \neg x_4) \wedge (\neg x_3 \vee \neg x_4 \vee \neg x_4 \vee \neg x_5)\big)$

Single-phase variables: Set $\neg x_2 = \neg x_3 = \neg x_4 = \neg x_5 = 1$

Remove the clauses of single-phase variables

The DP Algorithm: Running Example

$$\underbrace{(\neg x_1 \vee \neg x_2 \vee \neg x_4)}_{c_1} \wedge \underbrace{(x_1 \vee \neg x_3 \vee \neg x_4)}_{c_2} \wedge \underbrace{(\neg x_1 \vee \neg x_2 \vee x_4)}_{c_3} \wedge \underbrace{(\neg x_1 \vee \neg x_4 \vee \neg x_5)}_{c_4} \wedge \underbrace{(\neg x_1 \vee x_2 \vee x_3 \vee x_5)}_{c_5}$$

DP($c_1 \wedge c_2 \wedge c_3 \wedge c_4 \wedge c_5$)

No unit-clause contradiction, no single-phase variable

Pick x_1

Add resolvent: $(\neg x_3 \vee \neg x_4 \vee \neg x_2 \vee \neg x_4)$ for $c_2 \wedge c_1$

Resolvent: $(\neg x_3 \vee \neg x_4 \vee \neg x_2 \vee x_4)$ for $c_2 \wedge c_3$ is tautological, not added

Add resolvent: $(\neg x_3 \vee \neg x_4 \vee \neg x_4 \vee \neg x_5)$ for $c_2 \wedge c_4$

Resolvent: $(\neg x_3 \vee \neg x_4 \vee x_2 \vee x_3 \vee x_5)$ for $c_2 \wedge c_5$ is tautological, not added

DP$\big((\neg x_3 \vee \neg x_4 \vee \neg x_2 \vee \neg x_4) \wedge (\neg x_3 \vee \neg x_4 \vee \neg x_4 \vee \neg x_5)\big)$

Single-phase variables: Set $\neg x_2 = \neg x_3 = \neg x_4 = \neg x_5 = 1$

Remove the clauses of single-phase variables

There is no clause left

The DP Algorithm: Running Example

$$\underbrace{(\neg x_1 \vee \neg x_2 \vee \neg x_4)}_{c_1} \wedge \underbrace{(x_1 \vee \neg x_3 \vee \neg x_4)}_{c_2} \wedge \underbrace{(\neg x_1 \vee \neg x_2 \vee x_4)}_{c_3} \wedge \underbrace{(\neg x_1 \vee \neg x_4 \vee \neg x_5)}_{c_4} \wedge \underbrace{(\neg x_1 \vee x_2 \vee x_3 \vee x_5)}_{c_5}$$

<u>DP</u>($c_1 \wedge c_2 \wedge c_3 \wedge c_4 \wedge c_5$)

- No unit-clause contradiction, no single-phase variable
- Pick x_1
- Add resolvent: $(\neg x_3 \vee \neg x_4 \vee \neg x_2 \vee \neg x_4)$ for $c_2 \wedge c_1$
- Resolvent: $(\neg x_3 \vee \neg x_4 \vee \neg x_2 \vee x_4)$ for $c_2 \wedge c_3$ is tautological, not added
- Add resolvent: $(\neg x_3 \vee \neg x_4 \vee \neg x_4 \vee \neg x_5)$ for $c_2 \wedge c_4$
- Resolvent: $(\neg x_3 \vee \neg x_4 \vee x_2 \vee x_3 \vee x_5)$ for $c_2 \wedge c_5$ is tautological, not added

- <u>DP</u>($(\neg x_3 \vee \neg x_4 \vee \neg x_2 \vee \neg x_4) \wedge (\neg x_3 \vee \neg x_4 \vee \neg x_4 \vee \neg x_5)$)
  - Single-phase variables: Set $\neg x_2 = \neg x_3 = \neg x_4 = \neg x_5 = 1$
  - Remove the clauses of single-phase variables
  - There is no clause left
  - <u>DP</u>($\emptyset$)
    - return 1
  - return 1
- return 1

The DPLL Algorithm (1962)

DPLL refines DP. They are both complete, i.e., decide SAT for any CNF formula

- Backtracking-based search using repeated variable marginalisation
- Single-phase variable elimination like for DP
- Unit propagation
    - Unit clause (ℓ): Literal ℓ has to be set to 1, no choice!
    - Every clause that contains ℓ is removed (becomes 1)
    - Every clause that contains $\neg\ell$ is updated by removing $\neg\ell$ (which is 0)
    - This often leads to deterministic cascades of units

The DPLL Algorithm (1962)

DPLL refines DP. They are both complete, i.e., decide SAT for any CNF formula

- Backtracking-based search using repeated variable marginalisation
- Single-phase variable elimination like for DP
- Unit propagation
  - Unit clause (ℓ): Literal ℓ has to be set to 1, no choice!
  - Every clause that contains ℓ is removed (becomes 1)
  - Every clause that contains $\neg\ell$ is updated by removing $\neg\ell$ (which is 0)
  - This often leads to deterministic cascades of units

DPLL is a special case of LFTJ in the Boolean domain

- Backbone for both: variable marginalisation
- Single-phase variable elimination is a special case of marginalisation
  - For a Boolean variable, we sum over two cases: 0 and 1
  - For a single-phase variable, only one case is useful: its literal is 1
- Unit propagation is akin to Yannakakis (semi-join) reducer

The DPLL Algorithm (1962)

Algorithm DPLL (CNF Formula F)

1. **if** F only has single-phase variables **then return** 1 // Satisfiable
2. **if** F has an empty clause **then return** 0 // Unsatisfiable

//Next replace every occurrence of literal ℓ with 1 and of $\neg\ell$ to 0

3. **for each** unit clause (ℓ) in F **do** $F :=$ unit-propagate(ℓ, F)
4. **for each** single-phase variable ℓ in F **do** $F :=$ single-phase(ℓ, F)

5. **if** F has no literal, i.e., it is constant, **then** return F

//Next choose a literal to marginalise

6. ℓ = choose-literal(F)
7. **return** DPLL $(F \wedge (\ell))$ | DPLL $(F \wedge (\neg\ell))$

The DPLL Algorithm: Running Example

$F = (\neg x_1 \vee \neg x_2 \vee \neg x_4) \wedge (x_1 \vee \neg x_3 \vee \neg x_4) \wedge (\neg x_1 \vee \neg x_2 \vee x_4) \wedge (\neg x_1 \vee \neg x_4 \vee \neg x_5) \wedge (\neg x_1 \vee x_2 \vee x_3 \vee x_5)$

DPLL (F)
  Choose literal $\ell = \neg x_1$
  DPLL ($F \wedge (\neg x_1)$)
    Propagate unit clause $(\neg x_1)$ in F to obtain $F := (\neg x_3 \vee \neg x_4)$
    Single-phase variables: Set $\neg x_3 = \neg x_4 = 1$, F becomes (1)
    There is no literal left in F, return 1
  return 1

In case we would first recurse with DPLL ($F \wedge (x_1)$):
  Propagate unit clause (x_1) in F to obtain
    $F := (\neg x_2 \vee \neg x_4) \wedge (\neg x_2 \vee x_4) \wedge (\neg x_4 \vee \neg x_5) \wedge (x_2 \vee x_3 \vee x_5)$
  Single-phase variable: Set $x_3 = 1$ to obtain
    $F := (\neg x_2 \vee \neg x_4) \wedge (\neg x_2 \vee x_4) \wedge (\neg x_4 \vee \neg x_5)$
  Single-phase variables: Set $\neg x_2 = \neg x_5 = 1$, F becomes (1)
  There is no literal left in F, return 1
return 1

Acyclic SAT

α- and β-Acyclic SAT

- Well-known: SAT is NP-hard (Cook's Theorem)

α- and β-Acyclic SAT

- Well-known: SAT is NP-hard (Cook's Theorem)
- Bad news: α-acyclic SAT is still NP-hard

α- and β-Acyclic SAT

- Well-known: SAT is NP-hard (Cook's Theorem)
- Bad news: α-acyclic SAT is still NP-hard
- Good News: β-acyclic SAT can be solved in polynomial time using the DP algorithm

α-acyclic SAT is NP-hard (1/3)

Polynomial reduction from arbitrary SAT to α-acyclic SAT

Given: Arbitrary CNF formula F

Construct: α-Acyclic CNF formula F' that is equi-satisfiable to F

α-acyclic SAT is NP-hard (1/3)

Polynomial reduction from arbitrary SAT to α-acyclic SAT

Given: Arbitrary CNF formula F

Construct: α-Acyclic CNF formula F' that is equi-satisfiable to F

- Let $F = c_1 \wedge \ldots \wedge c_m$ with variables $x_1, \ldots, x_n$
- Set $F' = c_1 \wedge \ldots \wedge c_m \wedge (x_1 \vee \ldots \vee x_n \vee x_0)$ with fresh variable x_0

α-acyclic SAT is NP-hard (1/3)

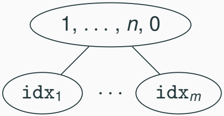
Polynomial reduction from arbitrary SAT to α-acyclic SAT

Given: Arbitrary CNF formula F

Construct: α-Acyclic CNF formula F' that is equi-satisfiable to F

- Let $F = c_1 \wedge \ldots \wedge c_m$ with variables $x_1, \ldots, x_n$
- Set $F' = c_1 \wedge \ldots \wedge c_m \wedge (x_1 \vee \ldots \vee x_n \vee x_0)$ with fresh variable x_0

F' is α-acyclic, since it has a join tree



where $\texttt{idx}_i$ is the index set of the variables in c_i

α-acyclic SAT is NP-hard (2/3)

$$F = c_1 \wedge \ldots \wedge c_m$$

equi-satisfiable to

$$F' = c_1 \wedge \ldots \wedge c_m \wedge (x_1 \vee \ldots \vee x_n \vee x_0)$$

α-acyclic SAT is NP-hard (2/3)

$$F = c_1 \wedge \ldots \wedge c_m$$

equi-satisfiable to

$$F' = c_1 \wedge \ldots \wedge c_m \wedge (x_1 \vee \ldots \vee x_n \vee x_0)$$

Example

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

$$F' = (x_1 \vee \neg x_2 \vee x_3) \wedge (\neg x_1 \vee x_2 \vee \neg x_3) \wedge (\neg x_1 \vee \neg x_2 \vee \neg x_3) \wedge (x_1 \vee x_2 \vee x_3 \vee x_0)$$

α-acyclic SAT is NP-hard (2/3)

$$F = c_1 \wedge \ldots \wedge c_m$$

equi-satisfiable to

$$F' = c_1 \wedge \ldots \wedge c_m \wedge (x_1 \vee \ldots \vee x_n \vee x_0)$$

Example

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

$F' = (x_1 \vee \neg x_2 \vee x_3) \wedge (\neg x_1 \vee x_2 \vee \neg x_3) \wedge (\neg x_1 \vee \neg x_2 \vee \neg x_3) \wedge (x_1 \vee x_2 \vee x_3 \vee x_0)$

- $\{x_1 = 1, x_2 = 1, x_3 = 0\}$ satisfies F

  $\{x_1 = 1, x_2 = 1, x_3 = 0, x_0 = 1\}$ satisfies F'

α-acyclic SAT is NP-hard (2/3)

$$F = c_1 \wedge \ldots \wedge c_m$$

equi-satisfiable to

$$F' = c_1 \wedge \ldots \wedge c_m \wedge (x_1 \vee \ldots \vee x_n \vee x_0)$$

Example

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

$F' = (x_1 \vee \neg x_2 \vee x_3) \wedge (\neg x_1 \vee x_2 \vee \neg x_3) \wedge (\neg x_1 \vee \neg x_2 \vee \neg x_3) \wedge (x_1 \vee x_2 \vee x_3 \vee x_0)$

- $\{x_1 = 1, x_2 = 1, x_3 = 0\}$ satisfies F

  $\{x_1 = 1, x_2 = 1, x_3 = 0, x_0 = 1\}$ satisfies F'

- $\{x_1 = 0, x_2 = 0, x_3 = 1, x_0 = 0\}$ satisfies F'

  $\{x_1 = 0, x_2 = 0, x_3 = 1\}$ satisfies F

α-acyclic SAT is NP-hard (3/3)

$$F = c_1 \wedge \ldots \wedge c_m$$

equi-satisfiable to

$$F' = c_1 \wedge \ldots \wedge c_m \wedge (x_1 \vee \ldots \vee x_n \vee x_0)$$

General case:

F satisfiable $\Rightarrow$ F' satisfiable

- Consider satisfying assignment τ for F
- $\tau \cup \{x_0 = 1\}$ satisfies F'

α-acyclic SAT is NP-hard (3/3)

$$F = c_1 \wedge \ldots \wedge c_m$$

equi-satisfiable to

$$F' = c_1 \wedge \ldots \wedge c_m \wedge (x_1 \vee \ldots \vee x_n \vee x_0)$$

General case:

F satisfiable $\Rightarrow$ F' satisfiable

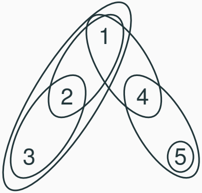
- Consider satisfying assignment τ for F
- $\tau \cup \{x_0 = 1\}$ satisfies F'

F' satisfiable $\Rightarrow$ F satisfiable

- Consider satisfying assignment τ' for F'
- $\tau - \{x_0 = 1, x_0 = 0\}$ satisfies F

Order of Variable Marginalisation Matters

$$F = \underbrace{(x_1 \vee \neg x_2 \vee \neg x_3)}_{c_1} \wedge \underbrace{(\neg x_1 \vee x_4)}_{c_2} \wedge \underbrace{(\neg x_2 \vee x_3)}_{c_3} \wedge (x_1 \vee x_2) \wedge (\neg x_4 \vee x_5) \wedge (\neg x_5)$$



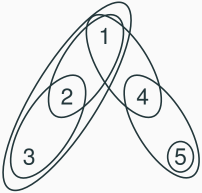
Order of Variable Marginalisation Matters

$$F = \underbrace{(x_1 \vee \neg x_2 \vee \neg x_3)}_{c_1} \wedge \underbrace{(\neg x_1 \vee x_4)}_{c_2} \wedge \underbrace{(\neg x_2 \vee x_3)}_{c_3} \wedge (x_1 \vee x_2) \wedge (\neg x_4 \vee x_5) \wedge (\neg x_5)$$

Start with the elimination of x_1

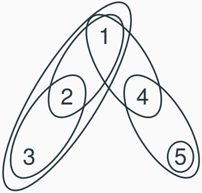
Resolvent of c_1 and c_2 is $c_{12} = (\neg x_2 \vee \neg x_3 \vee x_4)$

c_{12} is not included in any clause of F



Order of Variable Marginalisation Matters

$$F = \underbrace{(x_1 \vee \neg x_2 \vee \neg x_3)}_{c_1} \wedge \underbrace{(\neg x_1 \vee x_4)}_{c_2} \wedge \underbrace{(\neg x_2 \vee x_3)}_{c_3} \wedge (x_1 \vee x_2) \wedge (\neg x_4 \vee x_5) \wedge (\neg x_5)$$



Start with the elimination of x_1

Resolvent of c_1 and c_2 is $c_{12} = (\neg x_2 \vee \neg x_3 \vee x_4)$

c_{12} is not included in any clause of F

Start with the elimination of x_3

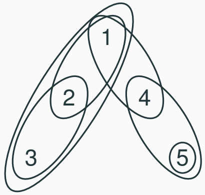
Resolvent of c_1 and c_3 is $c_{13} = (x_1 \vee \neg x_2)$

c_{13} is included in c_1

$\Longrightarrow$ No increase in the number of clauses

Order of Variable Marginalisation Matters

$$F = \underbrace{(x_1 \lor \neg x_2 \lor \neg x_3)}_{c_1} \land \underbrace{(\neg x_1 \lor x_4)}_{c_2} \land \underbrace{(\neg x_2 \lor x_3)}_{c_3} \land (x_1 \lor x_2) \land (\neg x_4 \lor x_5) \land (\neg x_5)$$



Start with the elimination of x_1

Resolvent of c_1 and c_2 is $c_{12} = (\neg x_2 \lor \neg x_3 \lor x_4)$

c_{12} is not included in any clause of F

Start with the elimination of x_3

Resolvent of c_1 and c_3 is $c_{13} = (x_1 \lor \neg x_2)$

c_{13} is included in c_1

$\Longrightarrow$ No increase in the number of clauses

β-acyclic CNF formulas admit marginalisation orders that avoid exponential increase in the number of clauses

This is thanks to a nice property of beta-acyclic hypergraphs: nested inclusion

Nested Inclusion Chain

A set $\{e_1, \ldots, e_k\}$ of sets forms a nested inclusion chain if $e_{i_1} \subseteq \ldots \subseteq e_{i_k}$ for some ordering $i_1, \ldots, i_k \in [k]$

Nested Inclusion Chain

A set $\{e_1, \ldots, e_k\}$ of sets forms a nested inclusion chain if $e_{i_1} \subseteq \ldots \subseteq e_{i_k}$ for some ordering $i_1, \ldots, i_k \in [k]$

We apply this property to the set of hyperedges of nodes in the SAT hypergraph

Nested Inclusion Chain

A set $\{e_1, \ldots, e_k\}$ of sets forms a nested inclusion chain if $e_{i_1} \subseteq \ldots \subseteq e_{i_k}$ for some ordering $i_1, \ldots, i_k \in [k]$

We apply this property to the set of hyperedges of nodes in the SAT hypergraph

For a hypergraph $\mathcal{H} = (\mathcal{V}, \mathcal{E})$, let $\partial(i) = \{e \in \mathcal{E} \mid i \in e\}$

Nested Inclusion Chain

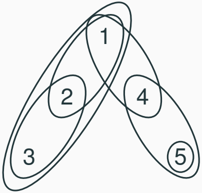
A set $\{e_1, \ldots, e_k\}$ of sets forms a nested inclusion chain if $e_{i_1} \subseteq \ldots \subseteq e_{i_k}$ for some ordering $i_1, \ldots, i_k \in [k]$

We apply this property to the set of hyperedges of nodes in the SAT hypergraph

For a hypergraph $\mathcal{H} = (\mathcal{V}, \mathcal{E})$, let $\partial(i) = \{e \in \mathcal{E} \mid i \in e\}$

Example

$\partial(4) = \{\{1, 4\}, \{4, 5\}\}$
is not a nested inclusion chain



Nested Inclusion Chain

A set $\{e_1, \ldots, e_k\}$ of sets forms a nested inclusion chain if $e_{i_1} \subseteq \ldots \subseteq e_{i_k}$ for some ordering $i_1, \ldots, i_k \in [k]$

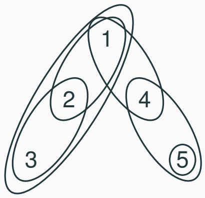
We apply this property to the set of hyperedges of nodes in the SAT hypergraph

For a hypergraph $\mathcal{H} = (\mathcal{V}, \mathcal{E})$, let $\partial(i) = \{e \in \mathcal{E} \mid i \in e\}$

Example

$\partial(4) = \{\{1, 4\}, \{4, 5\}\}$
is not a nested inclusion chain

$\partial(3) = \{\{2, 3\}, \{1, 2, 3\}\}$
is a nested inclusion chain



Property 1

Every β-acyclic hypergraph has a node i s.th. $\partial(i)$ forms a nested inclusion chain

DP Solver for β-Acyclicity SAT

Property 1

Every β-acyclic hypergraph has a node i s.th. $\partial(i)$ forms a nested inclusion chain

Property 2

β-acyclicity is closed under removal of nodes and hyperedges

DP Solver for β-Acyclicity SAT

Property 1

Every β-acyclic hypergraph has a node i s.th. $\partial(i)$ forms a nested inclusion chain

Property 2

β-acyclicity is closed under removal of nodes and hyperedges

DP algorithm for β-acyclic SAT

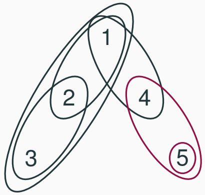
Apply the resolution rule only for a variable x_i such that $\partial(i)$ forms a nested inclusion chain

Example 1/4

$$F_0 = (x_1 \vee \neg x_2 \vee \neg x_3) \wedge (\neg x_1 \vee x_4) \wedge (\neg x_2 \vee x_3) \wedge (x_1 \vee x_2) \wedge (\neg x_4 \vee x_5) \wedge (\neg x_5)$$

Example 1/4

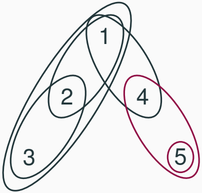
$$F_0 = (x_1 \vee \neg x_2 \vee \neg x_3) \wedge (\neg x_1 \vee x_4) \wedge (\neg x_2 \vee x_3) \wedge (x_1 \vee x_2) \wedge (\neg x_4 \vee x_5) \wedge (\neg x_5)$$



Example 1/4

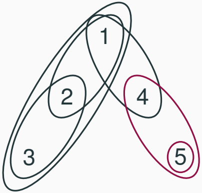
$$F_0 = (x_1 \vee \neg x_2 \vee \neg x_3) \wedge (\neg x_1 \vee x_4) \wedge (\neg x_2 \vee x_3) \wedge (x_1 \vee x_2) \wedge (\neg x_4 \vee x_5) \wedge (\neg x_5)$$

Nested inclusion chains
$\partial(5) = \{\{5\}, \{4, 5\}\}, \partial(3) = \{\{2, 3\}, \{1, 2, 3\}\}$



Example 1/4

$$F_0 = (x_1 \vee \neg x_2 \vee \neg x_3) \wedge (\neg x_1 \vee x_4) \wedge (\neg x_2 \vee x_3) \wedge (x_1 \vee x_2) \wedge (\neg x_4 \vee x_5) \wedge (\neg x_5)$$



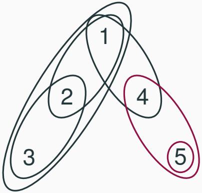
Nested inclusion chains
$\partial(5) = \{\{5\}, \{4,5\}\}, \partial(3) = \{\{2,3\}, \{1,2,3\}\}$

Do resolution on 5

Example 1/4

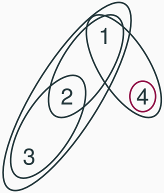
$$F_0 = (x_1 \vee \neg x_2 \vee \neg x_3) \wedge (\neg x_1 \vee x_4) \wedge (\neg x_2 \vee x_3) \wedge (x_1 \vee x_2) \wedge (\neg x_4 \vee x_5) \wedge (\neg x_5)$$

Nested inclusion chains
$\partial(5) = \{\{5\}, \{4, 5\}\}, \partial(3) = \{\{2, 3\}, \{1, 2, 3\}\}$



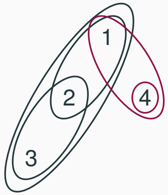
Do resolution on 5

$$F_1 = (x_1 \vee \neg x_2 \vee \neg x_3) \wedge (\neg x_1 \vee x_4) \wedge (\neg x_2 \vee x_3) \wedge (x_1 \vee x_2) \wedge (\neg x_4)$$



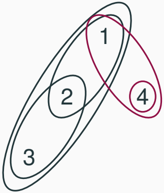
Example 2/4

$$F_1 = (x_1 \lor \neg x_2 \lor \neg x_3) \land (\neg x_1 \lor x_4) \land (\neg x_2 \lor x_3) \land (x_1 \lor x_2) \land (\neg x_4)$$



Example 2/4

$$F_1 = (x_1 \vee \neg x_2 \vee \neg x_3) \wedge (\neg x_1 \vee x_4) \wedge (\neg x_2 \vee x_3) \wedge (x_1 \vee x_2) \wedge (\neg x_4)$$

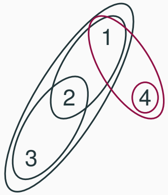


Nested inclusion chains

$\partial(4) = \{\{4\}, \{1, 4\}\}, \partial(3) = \{\{2, 3\}, \{1, 2, 3\}\}$

Example 2/4

$F_1 = (x_1 \vee \neg x_2 \vee \neg x_3) \wedge (\neg x_1 \vee x_4) \wedge (\neg x_2 \vee x_3) \wedge (x_1 \vee x_2) \wedge (\neg x_4)$



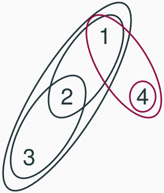
Nested inclusion chains

$\partial(4) = \{\{4\}, \{1, 4\}\}, \partial(3) = \{\{2, 3\}, \{1, 2, 3\}\}$

Do resolution on 4

Example 2/4

$$F_1 = (x_1 \vee \neg x_2 \vee \neg x_3) \wedge (\neg x_1 \vee x_4) \wedge (\neg x_2 \vee x_3) \wedge (x_1 \vee x_2) \wedge (\neg x_4)$$

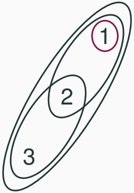


Nested inclusion chains

$\partial(4) = \{\{4\}, \{1, 4\}\}, \partial(3) = \{\{2, 3\}, \{1, 2, 3\}\}$

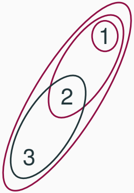
Do resolution on 4

$$F_2 = (x_1 \vee \neg x_2 \vee \neg x_3) \wedge (\neg x_1) \wedge (\neg x_2 \vee x_3) \wedge (x_1 \vee x_2)$$



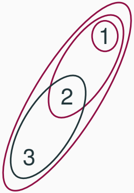
Example 3/4

$$F_2 = (x_1 \vee \neg x_2 \vee \neg x_3) \wedge (\neg x_1) \wedge (\neg x_2 \vee x_3) \wedge (x_1 \vee x_2)$$



Example 3/4

$$F_2 = (x_1 \vee \neg x_2 \vee \neg x_3) \wedge (\neg x_1) \wedge (\neg x_2 \vee x_3) \wedge (x_1 \vee x_2)$$

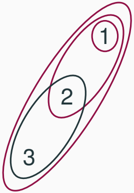


Nested inclusion chains

$\partial(1) = \{\{1\}, \{1,2\}, \{1,2,3\}\}, \partial(3) = \{\{2,3\}, \{1,2,3\}\}$

Example 3/4

$$F_2 = (x_1 \vee \neg x_2 \vee \neg x_3) \wedge (\neg x_1) \wedge (\neg x_2 \vee x_3) \wedge (x_1 \vee x_2)$$



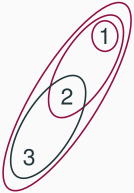
Nested inclusion chains

$\partial(1) = \{\{1\}, \{1, 2\}, \{1, 2, 3\}\}, \partial(3) = \{\{2, 3\}, \{1, 2, 3\}\}$

Do resolution on 1

Example 3/4

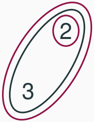
$F_2 = (x_1 \vee \neg x_2 \vee \neg x_3) \wedge (\neg x_1) \wedge (\neg x_2 \vee x_3) \wedge (x_1 \vee x_2)$



Nested inclusion chains
$\partial(1) = \{\{1\}, \{1,2\}, \{1,2,3\}\}, \partial(3) = \{\{2,3\}, \{1,2,3\}\}$

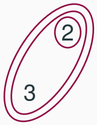
Do resolution on 1

$F_3 = (\neg x_2 \vee \neg x_3) \wedge (\neg x_2 \vee x_3) \wedge (x_2)$



Example 4/4

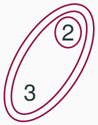
$$F_3 = (\neg x_2 \vee \neg x_3) \wedge (\neg x_2 \vee x_3) \wedge (x_2)$$



Example 4/4

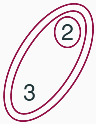
$F_3 = (\neg x_2 \vee \neg x_3) \wedge (\neg x_2 \vee x_3) \wedge (x_2)$

Nested inclusion chains
$\partial(2) = \{\{2\}, \{2,3\}, \{2,3\}\}, \partial(3) = \{\{2,3\}, \{2,3\}\}$



Example 4/4

$F_3 = (\neg x_2 \vee \neg x_3) \wedge (\neg x_2 \vee x_3) \wedge (x_2)$

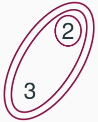


Nested inclusion chains
$\partial(2) = \{\{2\}, \{2,3\}, \{2,3\}\}, \partial(3) = \{\{2,3\}, \{2,3\}\}$

Do resolution on 2

Example 4/4

$F_3 = (\neg x_2 \vee \neg x_3) \wedge (\neg x_2 \vee x_3) \wedge (x_2)$



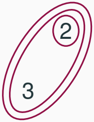
Nested inclusion chains
$\partial(2) = \{\{2\}, \{2,3\}, \{2,3\}\}, \partial(3) = \{\{2,3\}, \{2,3\}\}$

Do resolution on 2

$F_4 = (\neg x_3) \wedge (x_3)$

Example 4/4

$F_3 = (\neg x_2 \vee \neg x_3) \wedge (\neg x_2 \vee x_3) \wedge (x_2)$



Nested inclusion chains
$\partial(2) = \{\{2\}, \{2,3\}, \{2,3\}\}, \partial(3) = \{\{2,3\}, \{2,3\}\}$

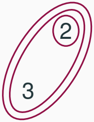
Do resolution on 2

$F_4 = (\neg x_3) \wedge (x_3)$

Unit-clause contradiction

Example 4/4

$F_3 = (\neg x_2 \vee \neg x_3) \wedge (\neg x_2 \vee x_3) \wedge (x_2)$



Nested inclusion chains
$\partial(2) = \{\{2\}, \{2,3\}, \{2,3\}\}, \partial(3) = \{\{2,3\}, \{2,3\}\}$

Do resolution on 2

$F_4 = (\neg x_3) \wedge (x_3)$

Unit-clause contradiction

$\Rightarrow$ F not satisfiable