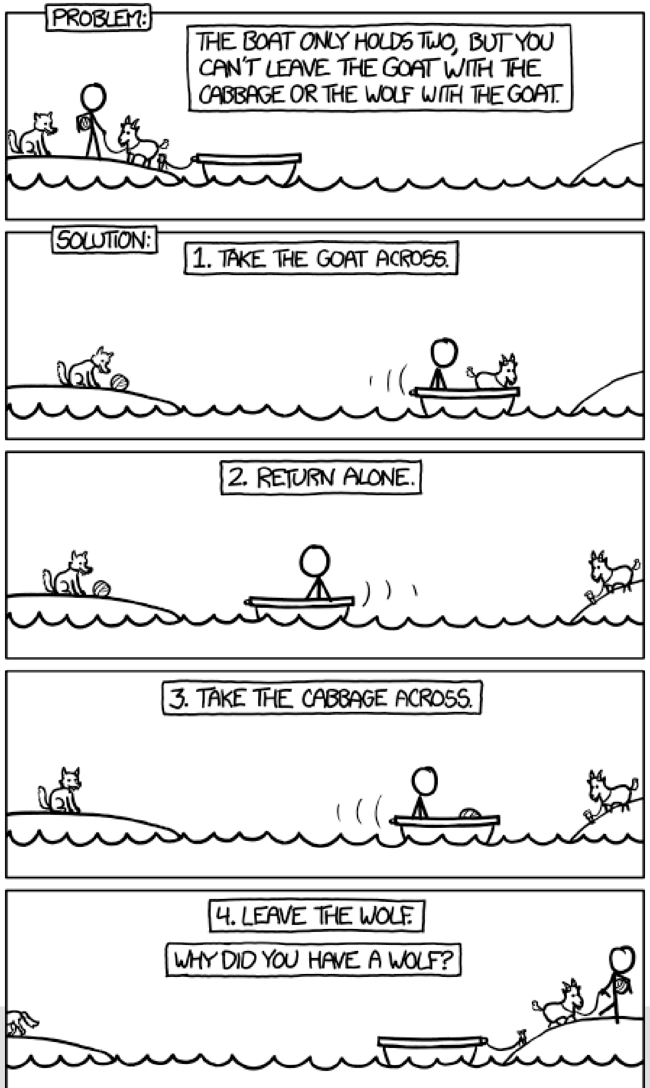
Logic and Computability

Lecture 2

SAT Solving

Bettina Könighofer
bettina.koenighofer@iaik.tugraz.at

Stefan Pranger
stefan.pranger@iaik.tugraz.at

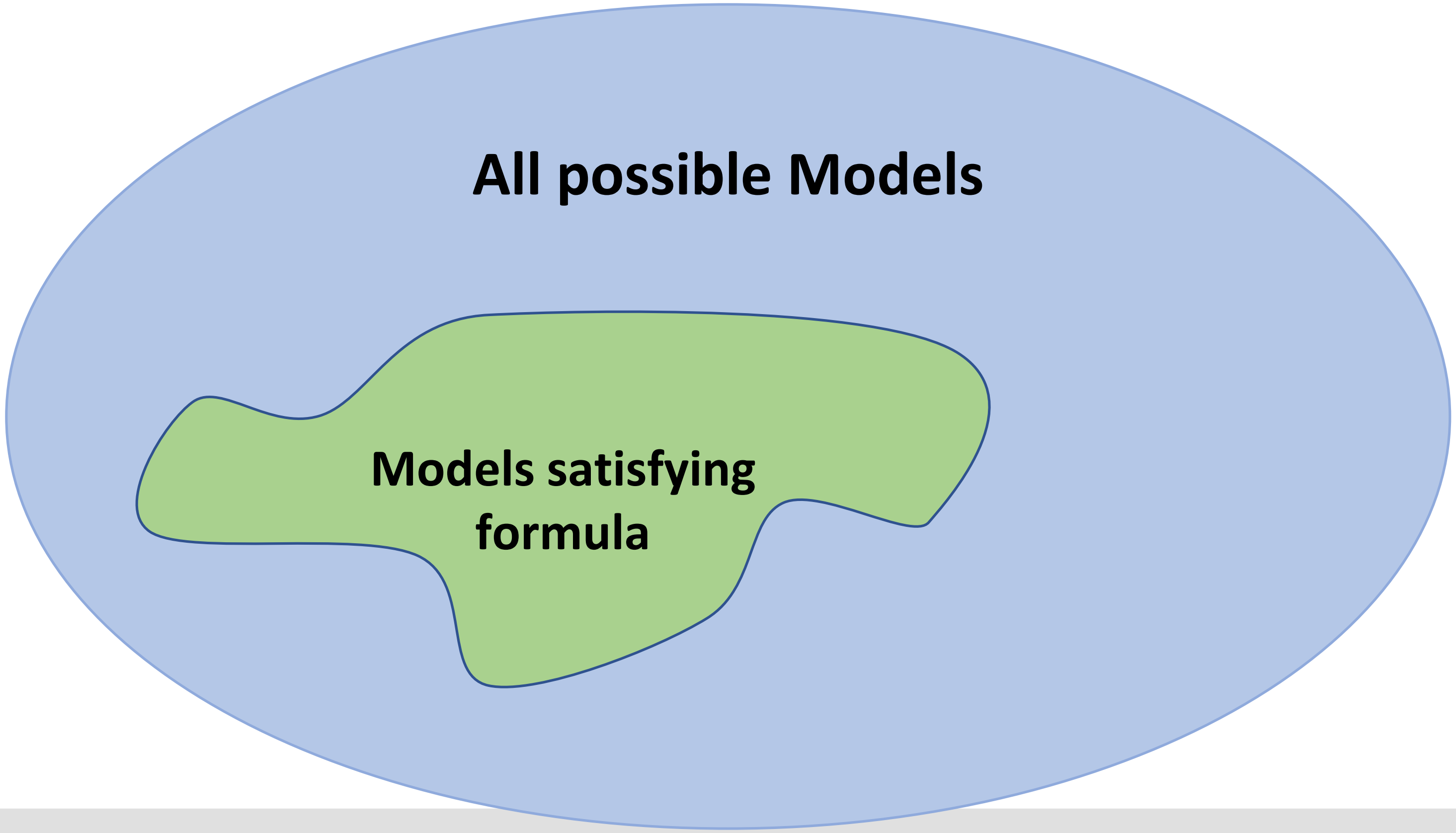


PROBLEM:
THE BOAT ONLY HOLDS TWO, BUT YOU CAN'T LEAVE THE GOAT WITH THE CABBAGE OR THE WOLF WITH THE GOAT.
SOLUTION:
1. TAKE THE GOAT ACROSS.
2. RETURN ALONE.
3. TAKE THE CABBAGE ACROSS.
4. LEAVE THE WOLF.
WHY DID YOU HAVE A WOLF?

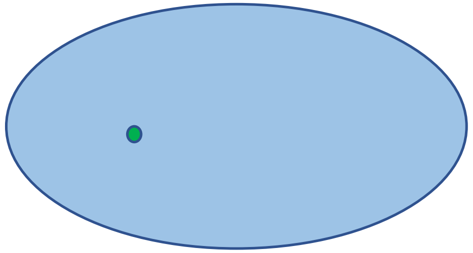
https://xkcd.com/287/

SAT Problem

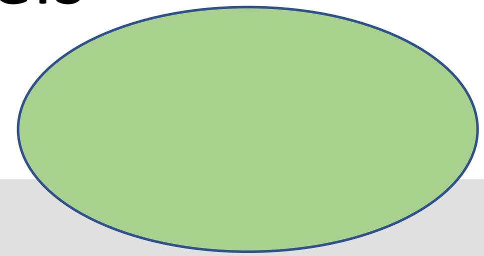
- Decide whether a formula φ is satisfiable
  - φ is SAT iff there exists a model $\mathcal{M}$ such that $\mathcal{M} \vDash \varphi$



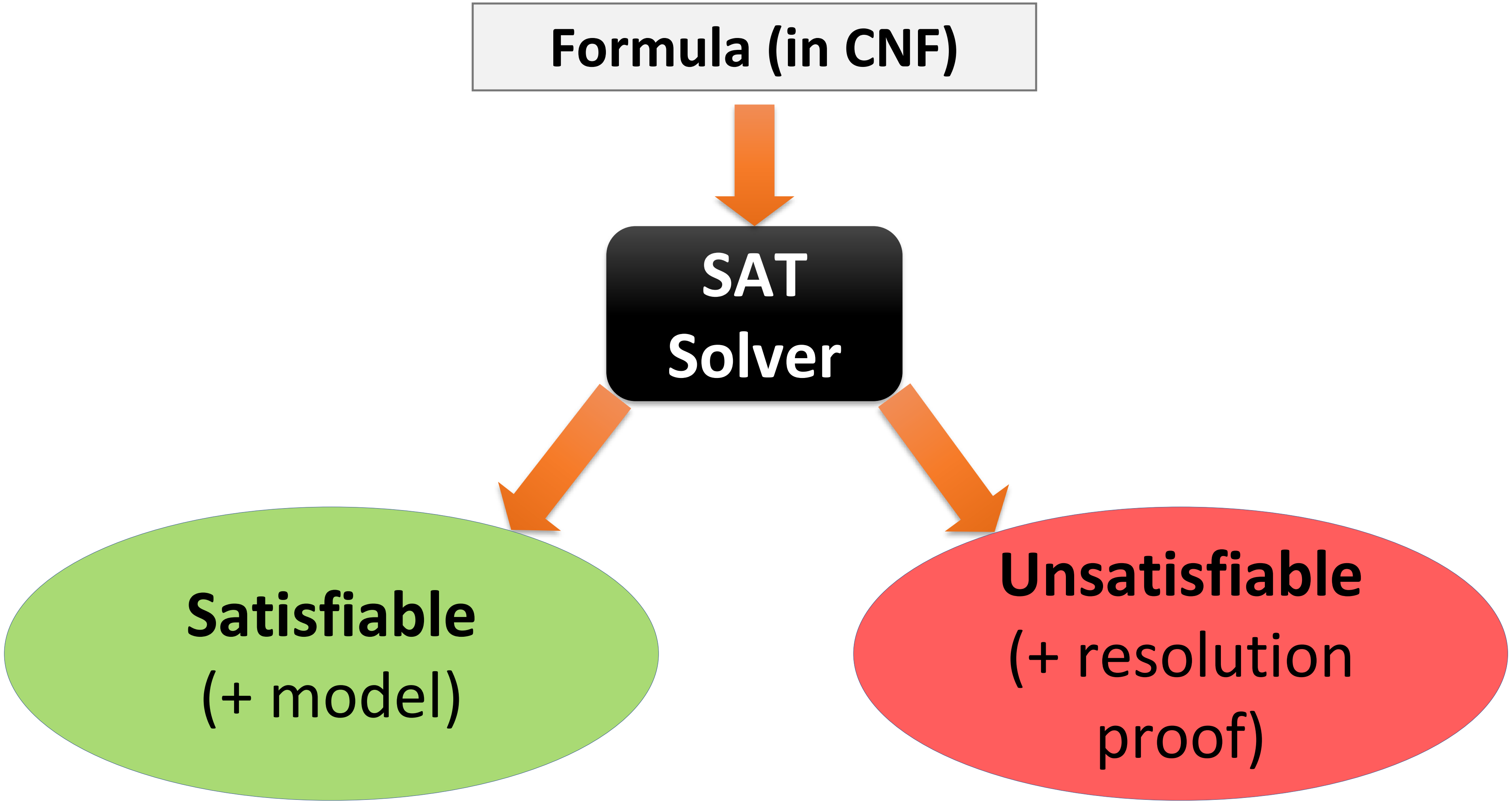
- One Model



- All Models



SAT-Solver



Formula (in CNF)

SAT Solver

Satisfiable

(+ model)

Unsatisfiable

(+ resolution proof)

Motivation – SAT Solving

- Applications
  - HW and SW Verification
  - Bounded Model Checking
  - Hardware Equivalence Checking
  - Circuit Synthesis
  - Planning (e.g., air-traffic control, telegraph routing)
  - Scheduling (sport tournaments)
  - Finite mathematics
  - Cryptanalysis
  - …

SAT Problem

- Decide whether a formula φ is satisfiable
  - φ is SAT iff there exists a model $\mathcal{M}$ such that $\mathcal{M} \vDash \varphi$
- The SAT problem is NP complete
  - $P \neq NP \Rightarrow$ worst-case exponential
- Problem: Formulas are huge!
- Automated Tools: **“SAT Solver”**
  - Highly efficient for many practical problem instances

Motivation – SAT Encoding

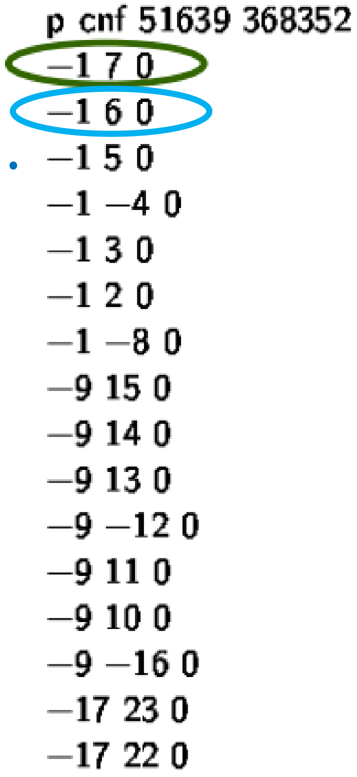
- Automatically generated from problem specifications

Nr variables

Nr clauses

$(\neg x_1 \lor x_7) \land$

$(\neg x_1 \lor x_6) \land \cdots$

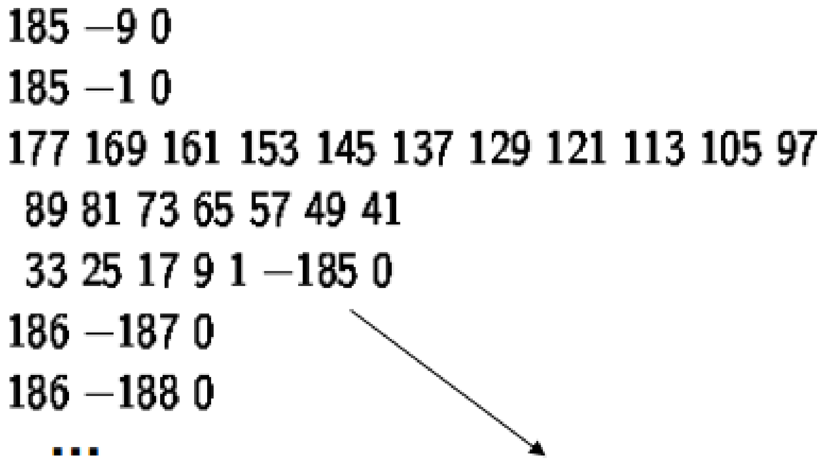


```
p cnf 51639 368352
−1 7 0
−1 6 0
−1 5 0
−1 −4 0
−1 3 0
−1 2 0
−1 −8 0
−9 15 0
−9 14 0
−9 13 0
−9 −12 0
−9 11 0
−9 10 0
−9 −16 0
−17 23 0
−17 22 0
```

Should x_1 be set to False?

10 Pages Later

- Automatically generated from problem specifications



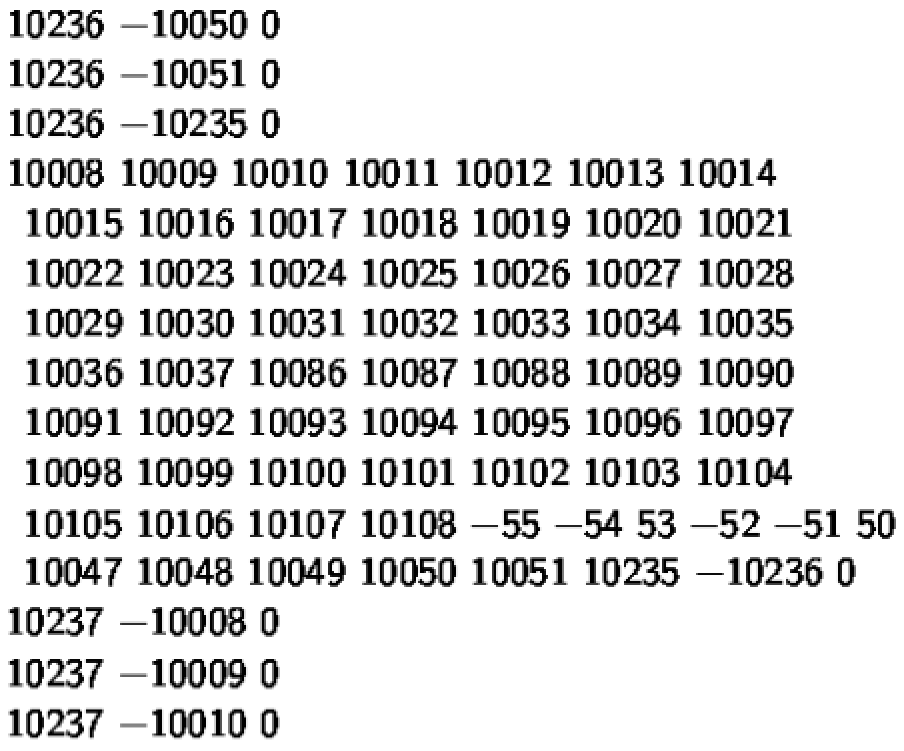
```
185 −9 0
185 −1 0
177 169 161 153 145 137 129 121 113 105 97
 89 81 73 65 57 49 41
 33 25 17 9 1 −185 0
186 −187 0
186 −188 0
   ...
```

i.e., (x_{177} or x_{169} or x_{161} or x_{153} …
x_{33} or x_{25} or x_{17} or x_9 or x_1 or (not x_{185}))

Note x_1

4.000 Pages Later

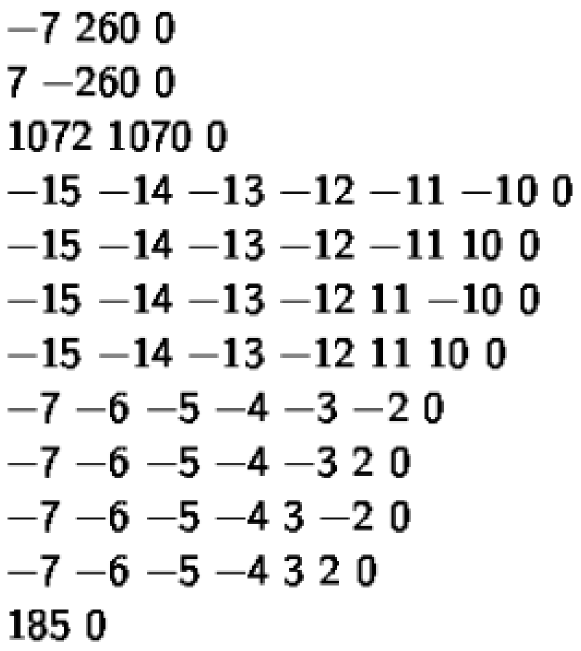
- Automatically generated from problem specifications



```
10236 −10050 0
10236 −10051 0
10236 −10235 0
10008 10009 10010 10011 10012 10013 10014
 10015 10016 10017 10018 10019 10020 10021
 10022 10023 10024 10025 10026 10027 10028
 10029 10030 10031 10032 10033 10034 10035
 10036 10037 10086 10087 10088 10089 10090
 10091 10092 10093 10094 10095 10096 10097
 10098 10099 10100 10101 10102 10103 10104
 10105 10106 10107 10108 −55 −54 53 −52 −51 50
 10047 10048 10049 10050 10051 10235 −10236 0
10237 −10008 0
10237 −10009 0
10237 −10010 0
```

…

Finally, 15.000 Pages Later

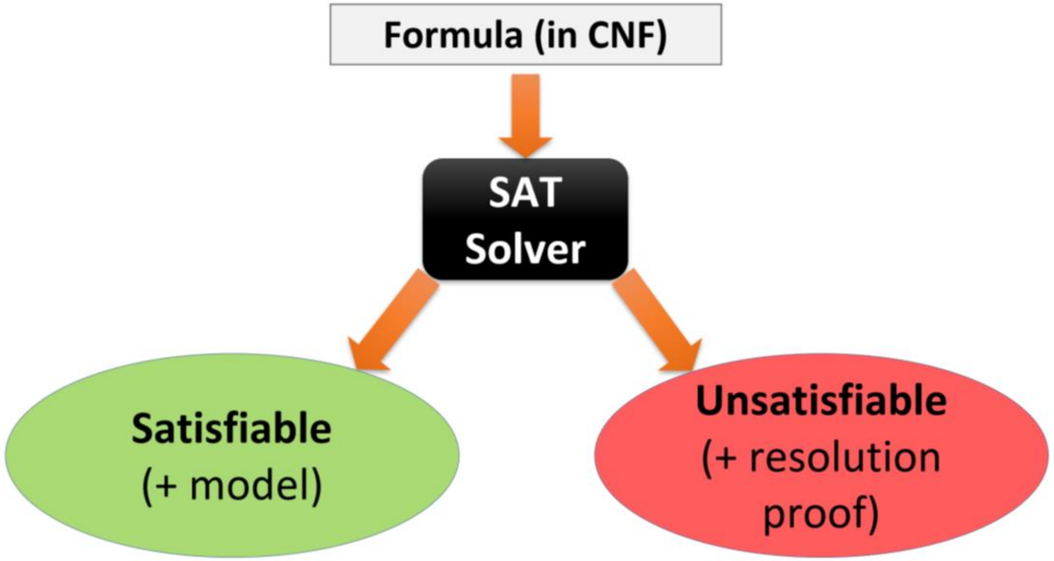


```
−7 260 0
7 −260 0
1072 1070 0
−15 −14 −13 −12 −11 −10 0
−15 −14 −13 −12 −11 10 0
−15 −14 −13 −12 11 −10 0
−15 −14 −13 −12 11 10 0
−7 −6 −5 −4 −3 −2 0
−7 −6 −5 −4 −3 2 0
−7 −6 −5 −4 3 −2 0
−7 −6 −5 −4 3 2 0
185 0
```

- Search space of truth assignments:
  - $2^{50000} \approx 3.1607 \ * 10^{15051}$
- How long to solve it?
  - Modern SAT solver needs just a few seconds!

Outline

- DPLL Algorithm
  - Boolean Constrain Propagation
  - Pure Literals
  - Conflict-Driven Clause Learning

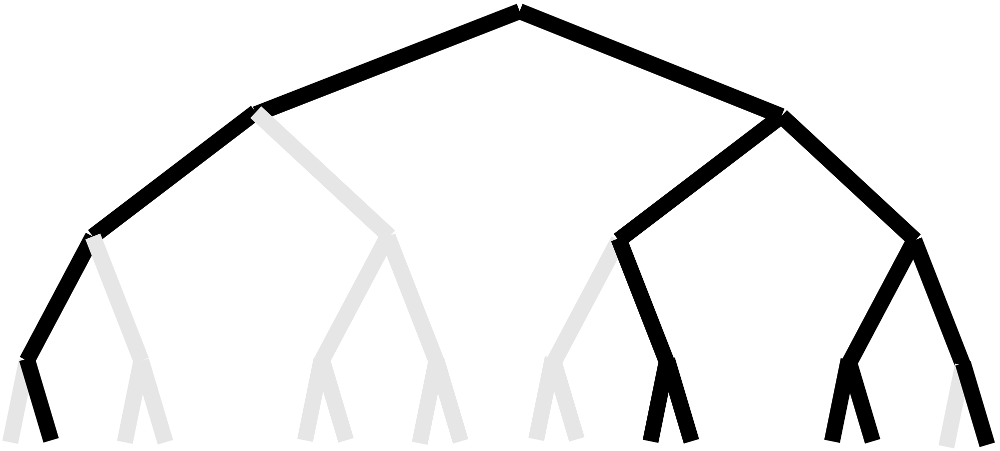


Formula (in CNF)

SAT Solver

Satisfiable (+ model)

Unsatisfiable (+ resolution proof)



DPLL Algorithm

- Introduced by Martin Davis, Hilary Putnam, Donald Loveland and George Logemann in 1962

  📖 M. Davis, G. Logemann, and D. Loveland.
  “A machine program for theorem-proving”. Communications of the ACM, 5:394-397, 1962

- Algorithm still forms the basis for most modern SAT solvers

DPLL Algorithm

- Introduced by Martin Davis, Hilary Putnam, Donald Loveland and George Logemann in 1962

  📖 M. Davis, G. Logemann, and D. Loveland.
  “A machine program for theorem-proving”. Communications of the ACM, 5:394-397, 1962

- Algorithm still forms basis for most modern SAT solvers

- Input:
  - Formula in Conjunctive Normal Form (CNF)

Conjunctive Normal Form

Conjunctive Normal Form

- **Literal:** propositional variable or its negation
  - Example: $p, \quad \neg q$

Conjunctive Normal Form

- **Literal:** propositional variable or its negation
  - Example: $p, \ \neg q$
- **Clause:** disjunction of literals
  - Example: $(p \vee \neg q \vee r)$

Conjunctive Normal Form

- **Literal:** propositional variable or its negation
  - Example: $p, \ \neg q$

- **Clause:** disjunction of literals
  - Example: $(p \vee \neg q \vee r)$

- **Conjunctive Normal Form (CNF)**
  - Conjunction of clauses:

$$(a_1 \vee a_2 \vee \cdots \vee a_n) \wedge (b_1 \vee \cdots \vee b_m) \wedge \cdots$$

where each a_i, b_j is a literal

Conjunctive Normal Form

- **Literal:** propositional variable or its negation
  - Example: $p, \;\; \neg q$

- **Clause:** disjunction of literals
  - Example: $(p \vee \neg q \vee r)$

- **Conjunctive Normal Form (CNF)**
  - Conjunction of clauses:

$$(a_1 \vee a_2 \vee \cdots \vee a_n) \wedge (b_1 \vee \cdots \vee b_m) \wedge \cdots$$

where each a_i, b_j is a literal

- **Examples**: $\varphi = \underline{a} \wedge \underline{(b \vee \neg c)} \wedge \underline{(\neg a \vee \neg b \vee c)}$
  $\varphi = \underline{\neg a}$

Notation

- Today: φ is a formula in CNF
  - $\varphi = (b \vee \neg c) \wedge (\neg a \vee \neg b \vee c)$

Notation

- Today: φ is a formula in CNF
  - $\varphi = (b \vee \neg c) \wedge (\neg a \vee \neg b \vee c)$

- **A** is an assignment of truth values to variables
  - $A = \{a \rightarrow true, b \rightarrow false, c \rightarrow false\}$
  - Total or partial assignment

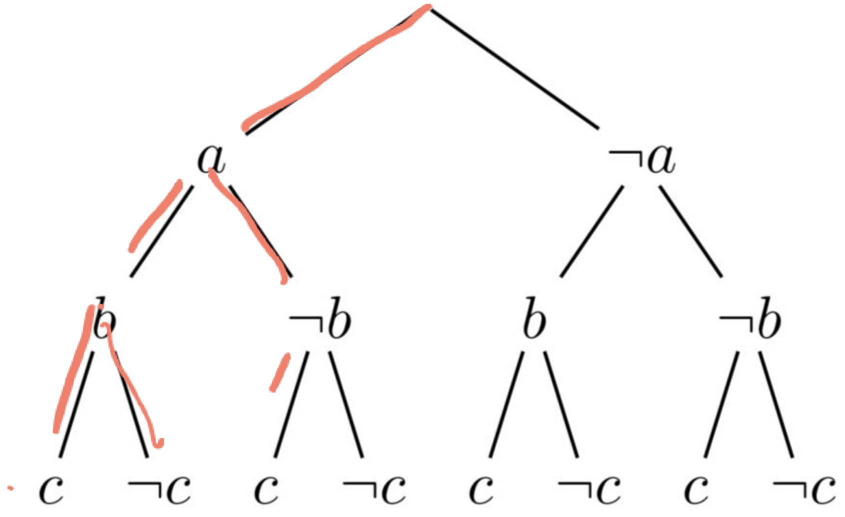
Notation

- Today: φ is a formula in CNF
  - $\underline{\varphi} = (\underset{\bot}{b} \lor \neg \underset{\bot}{c}) \land (\neg \underset{\top}{a} \lor \neg \underset{\bot}{b} \lor \underset{\bot}{c})$
- **A** is an assignment of truth values to variables
  - $\underline{A} = \{a \to true, b \to false, c \to false\}$
  - Total or partial Assignment
- $\varphi[A]$: φ with all variables set according to A
  - $\varphi[A] = \underbrace{(\bot \lor \neg\bot) \land (\neg\top \lor \neg\bot \lor \bot)} = \top \land (\bot \lor \top \lor \bot) = \underline{\underline{\top}}$

Basis Idea - Backtracking Binary Search

- Recursively search for a satisfying model/assignment
  - Search for A such that $\varphi[A] = \top$

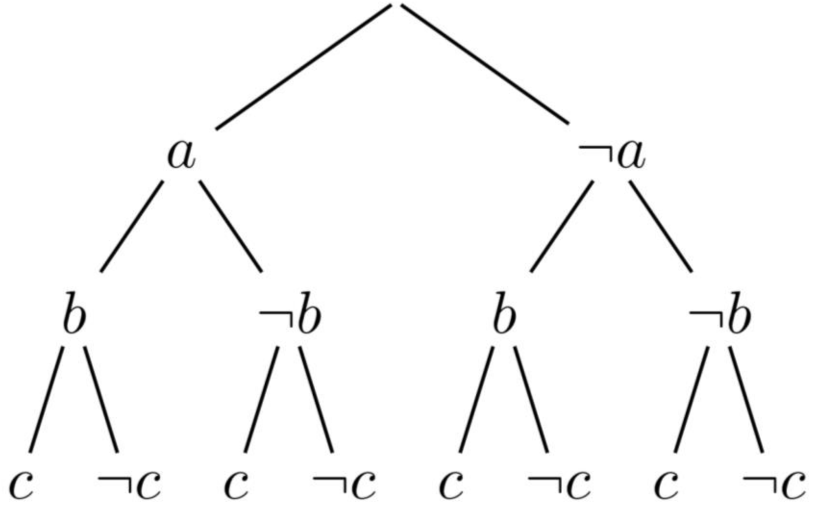
Search space for φ with variables a, b, c



Basis Idea - Backtracking Binary Search

- Recursively search for a satisfying model/assignment
  - Search for A such that $\varphi[A] = \top$
- No such A exists
  - φ is unsatisfiable

Search space for φ with variables a, b, c



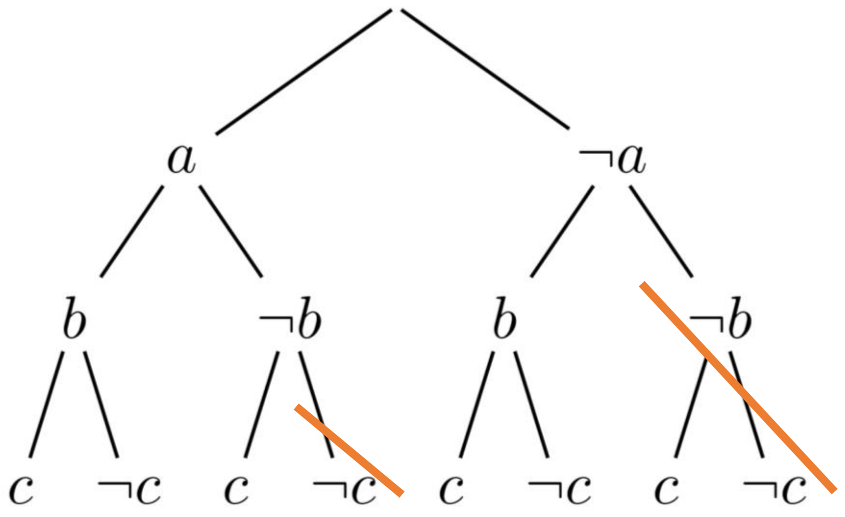
Basis Idea - Backtracking Binary Search

- Recursively search for a satisfying model/assignment
  - Search for A such that $\varphi[A] = \top$

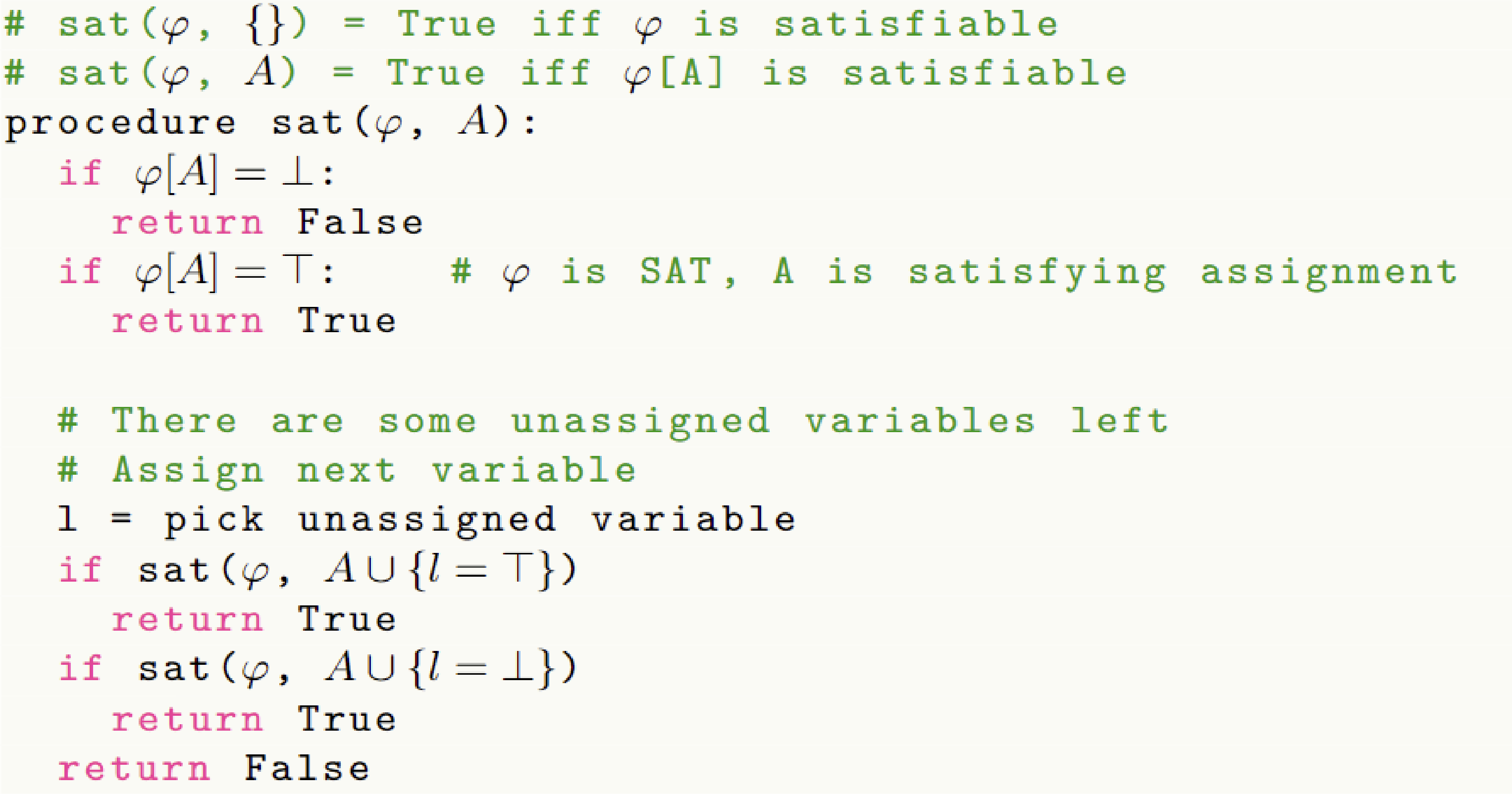
- No such A exists
  - φ is unsatisfiable

- Several optimizations to prune search tree.

Search space for φ with variables a, b, c

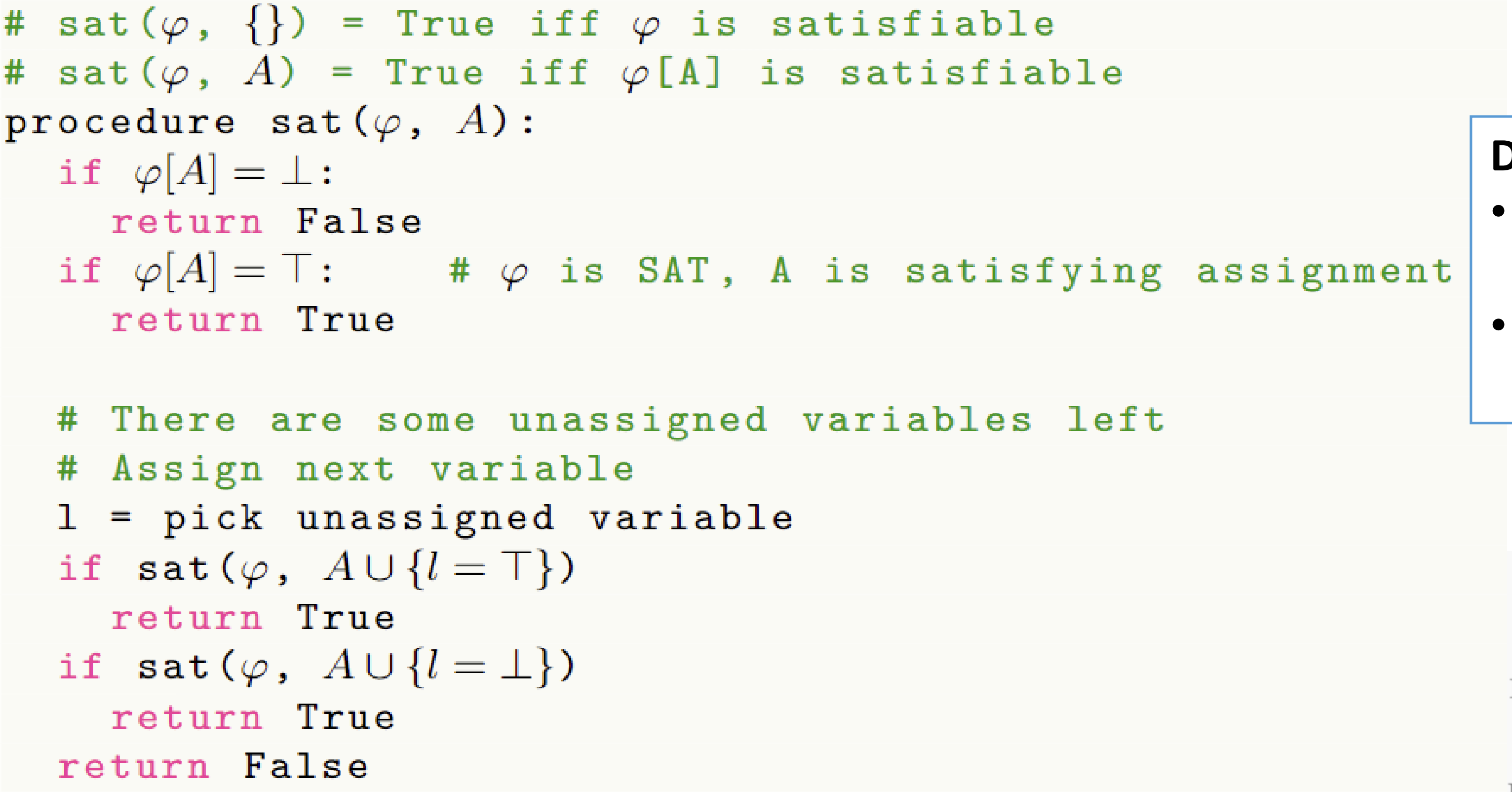


Basis Idea - Backtracking Binary Search



```
# sat($\varphi$, {}) = True iff $\varphi$ is satisfiable
# sat($\varphi$, $A$) = True iff $\varphi$[A] is satisfiable
procedure sat($\varphi$, $A$):
  if $\varphi[A] = \bot$:
    return False
  if $\varphi[A] = \top$:    # $\varphi$ is SAT, A is satisfying assignment
    return True

  # There are some unassigned variables left
  # Assign next variable
  l = pick unassigned variable
  if sat($\varphi$, $A \cup \{l = \top\}$)
    return True
  if sat($\varphi$, $A \cup \{l = \bot\}$)
    return True
  return False
```

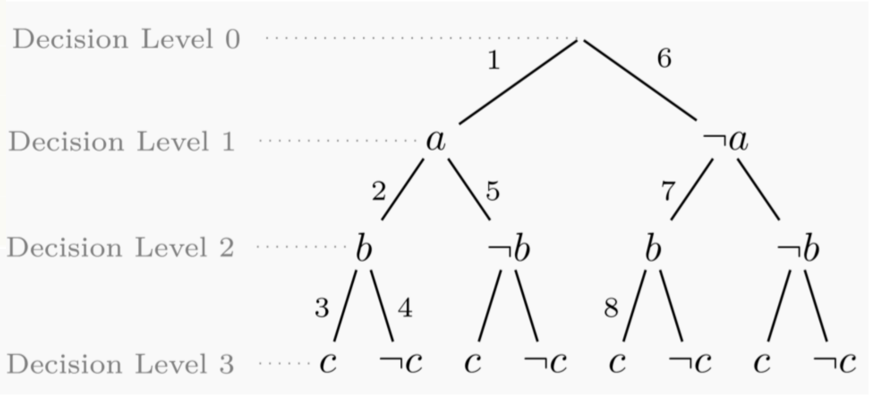
Basis Idea - Backtracking Binary Search



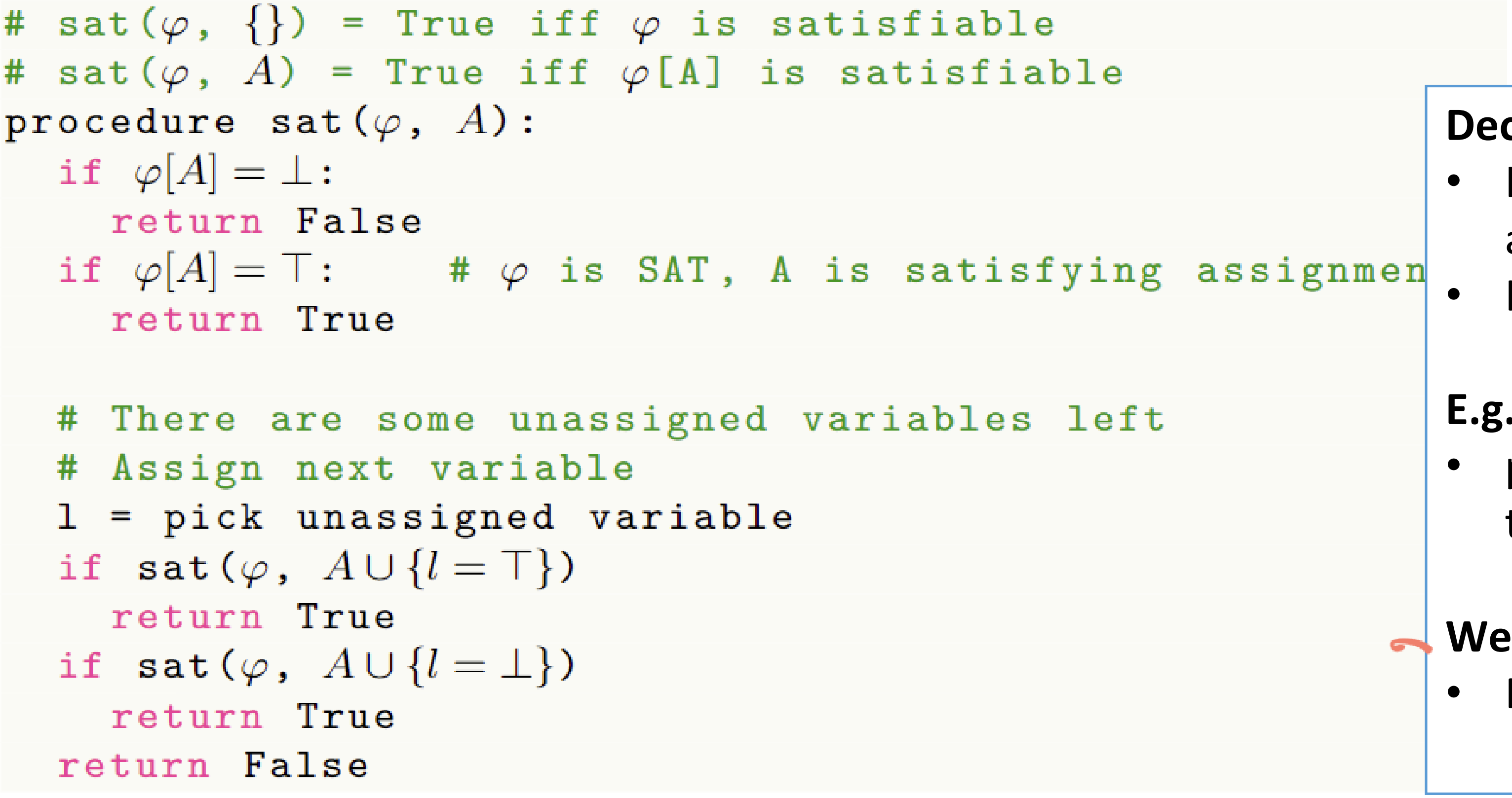
```
# sat($\varphi$, {}) = True iff $\varphi$ is satisfiable
# sat($\varphi$, $A$) = True iff $\varphi$[A] is satisfiable
procedure sat($\varphi$, $A$):
  if $\varphi[A] = \bot$:
    return False
  if $\varphi[A] = \top$:     # $\varphi$ is SAT, A is satisfying assignment
    return True

  # There are some unassigned variables left
  # Assign next variable
  l = pick unassigned variable
  if sat($\varphi$, $A \cup \{l = \top\}$)
    return True
  if sat($\varphi$, $A \cup \{l = \bot\}$)
    return True
  return False
```

Decision Level:

- Decision = algorithm assigns truth value to a variable
- Decision level = number of currently made decisions



Basis Idea - Backtracking Binary Search



```
# sat($\varphi$, {}) = True iff $\varphi$ is satisfiable
# sat($\varphi$, $A$) = True iff $\varphi$[A] is satisfiable
procedure sat($\varphi$, $A$):
  if $\varphi[A] = \bot$:
    return False
  if $\varphi[A] = \top$:    # $\varphi$ is SAT, A is satisfying assignmen
    return True

  # There are some unassigned variables left
  # Assign next variable
  l = pick unassigned variable
  if sat($\varphi$, $A \cup \{l = \top\}$)
    return True
  if sat($\varphi$, $A \cup \{l = \bot\}$)
    return True
  return False
```

Decision heuristic

- Heuristic to decide which variable should be assigned next
- Huge impact on solving time

E.g.: Dynamic Largest Individual Sum

- pick the variable and truth value, such that the most unresolved clauses become satisfied.

We will use a predefined order.

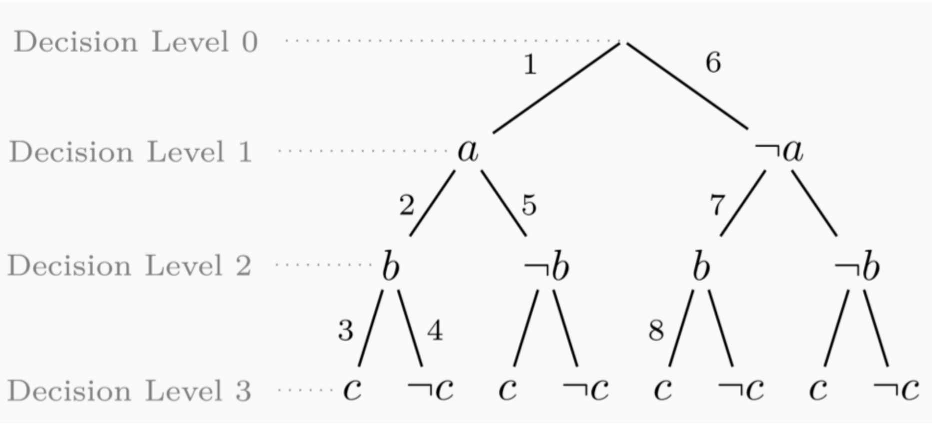
- E.g., lexicographical order, positive phase first

Tabular Execution of the Basic Search

$$\varphi := (\neg a \lor b) \land (\neg b \lor c) \land (\neg c \lor \neg a)$$

Decision heuristic: alphabetical order starting with the positive phase

Iteration	1	2	3	4	5	6	7	8	9
Dec. Level	0								
Assignment	{}								
CL1: $\{\neg a, b\}$	$\{\neg a, b\}$								
CL2: $\{\neg b, c\}$	$\{\neg b, c\}$								
CL3: $\{\neg c, \neg a\}$	$\{\neg c, \neg a\}$								
Decision	a								



Tabular Execution of the Basic Search

$$\varphi \coloneqq (\neg a \vee b) \wedge (\neg b \vee c) \wedge (\neg c \vee \neg a)$$

Decision heuristic: alphabetical order starting with the positive phase

Iteration	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**
Dec. Level	0	1	2	3					
Assignment	{}	a	a, b	a, b, c					
CL1: $\{\neg a, b\}$	$\{\neg a, b\}$	b	✓	✓					
CL2: $\{\neg b, c\}$	$\{\neg b, c\}$	$\neg b, c$	c	✓					
CL3: $\{\neg c, \neg a\}$	$\{\neg c, \neg a\}$	$\neg c$	$\neg c$	{} ✗					
Decision	a	b	c						

Evaluate clauses under current A:

- Satisfied Clause:
  - At least on of its literals is satisfied under A
  - Marked with ✓
- Conflicting clauses:
  - all of its literals are not satisfied under A
  - Marked with {}✗
- Unresolved clauses:
  - Otherwise

Tabular Execution of the Basic Search

$\varphi := (\neg a \vee b) \wedge (\neg b \vee c) \wedge (\neg c \vee \neg a)$

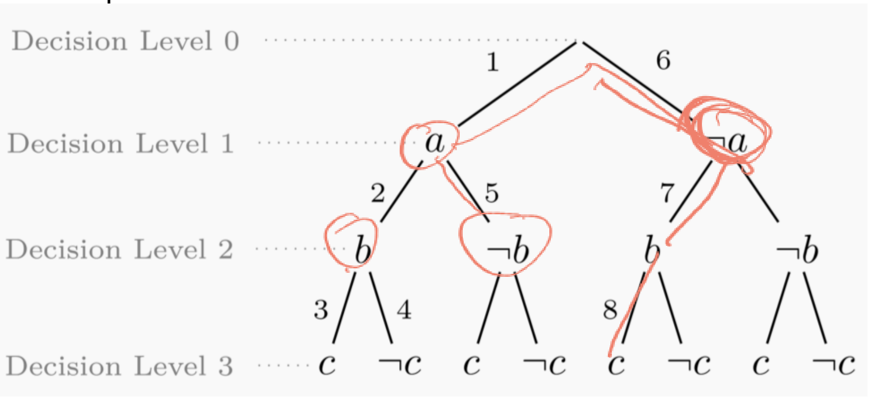
Decision heuristic: alphabetical order starting with the positive phase

$\rightarrow \mathcal{M} : \varphi^{\mathcal{M}} = T$

Step	1	2	3	4	5	6	7	8	9
Dec. Level	0	1	2	3	3	2	1	2	3
Assignment	{}	a	a,b	a,b,c	a b ¬c	a ¬b	¬a	¬a b	¬a b c
CL1: $\{\neg a, b\}$	$\neg a, b$	b	✓	✓	✓	{}	✓	✓	✓
CL2: {¬b, c}	¬b, c	¬b, c	c	✓	{}	✓	¬b, c	c	✓
CL3: {¬c, ¬a }	¬c, ¬a	¬c	¬c	{} X	✓	¬ c	✓	✓	✓
Decision	a	b	c	¬ c	¬b	¬ a	b	c	SAT

Found a conflicting clause: {} ✗

- All of its literals are not satisfied under A
- Backtrack
  - Remove last decision
  - Reduce Decision



Tabular Execution of the Basic Search

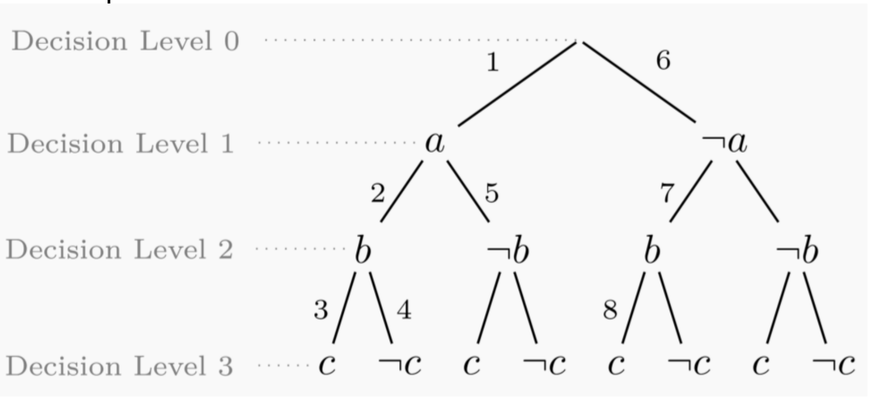
$\varphi \coloneqq (\neg a \vee b) \wedge (\neg b \vee c) \wedge (\neg c \vee \neg a)$

Decision heuristic: alphabetical order starting with the positive phase

Step	1	2	3	4	5	6	7	8	9
Dec. Level	0	1	2	3	3	2	1	2	3
Assignment	{}	a	a,b	a,b,c	a,b, ¬c	a, ¬b	¬a	¬a,b	¬a,b,c
CL1: $\{\neg a, b\}$	$\neg a, b$	b	✓	✓	✓	{} X	✓	✓	✓
CL2: {¬b, c}	¬b, c	¬b, c	c	✓	{} X	✓	¬b, c	c	✓
CL3: {¬c, ¬a }	¬c, ¬a	¬c	¬c	{} X	✓	¬c	✓	✓	✓
Decision	a	b	c	¬c	¬b	¬a	b	c	SAT

All clauses are satisfied ✓

- Report SAT and
  A as satisfying assignment



Boolean Constrain Propagation (BCP)

- Unit clause:
  - a clause with a single unassigned literal
  - Examples:
    - (a)
    - (¬b)

- Unit Clause exists ➔ set its literal
  - Very simple but very important heuristic!

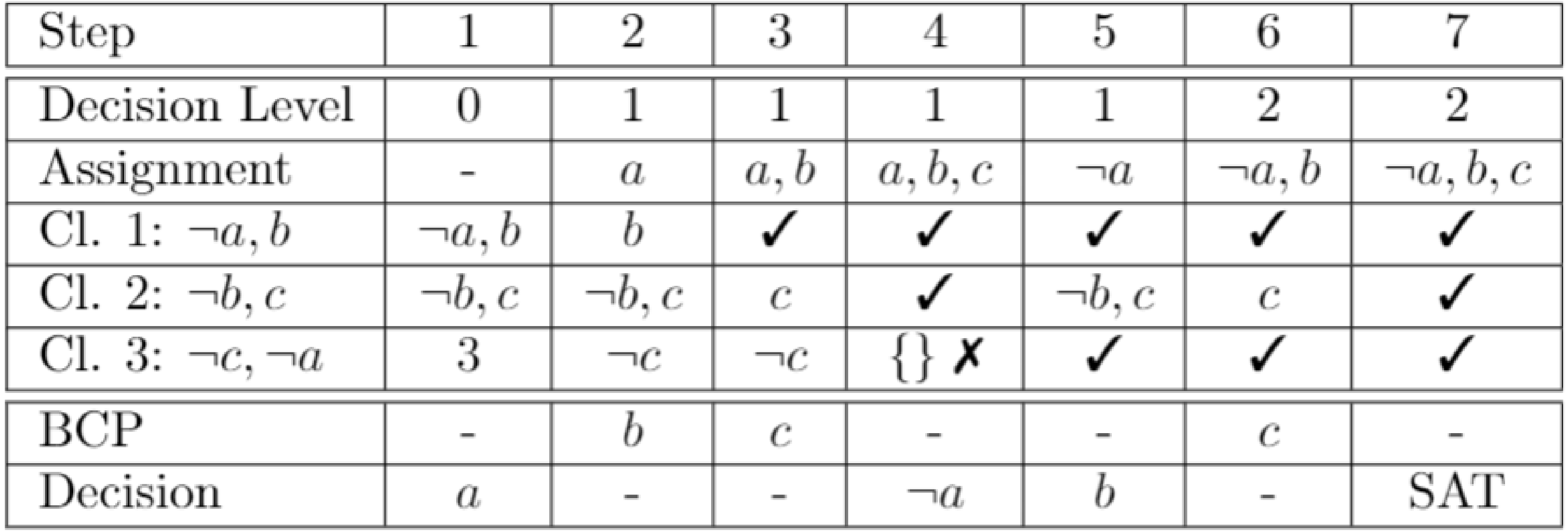
DPLL + BCP Example

- $\varphi := (\neg a \vee b) \wedge (\neg b \vee c) \wedge (\neg c \vee \neg a)$

Step	1	2	3	4	5	6	7	8	9
Dec. Level	0	1	1	1	1	2	2		
Assignment	/	a	a, b	a, b, c	¬a	¬a, b	¬a, b, c		
CL1: $\{\neg a, b\}$	¬a, b	b	✓	✓	✓	✓	✓		
CL2: {¬b, c}	¬b, c	¬b, c	c	✓	¬b, c	c	✓		
CL3: {¬c, ¬a }	¬c, ¬a	¬c	¬c	↯	✓	✓	✓		
BCP		b	c			c			
Decision	a			¬a	b		SAT		

DPLL + BCP Example

- $\varphi \coloneqq (\neg a \vee b) \wedge (\neg b \vee c) \wedge (\neg c \vee \neg a)$



Step	1	2	3	4	5	6	7
Decision Level	0	1	1	1	1	2	2
Assignment	-	a	a, b	a, b, c	$\neg a$	$\neg a, b$	$\neg a, b, c$
Cl. 1: $\neg a, b$	$\neg a, b$	b	✓	✓	✓	✓	✓
Cl. 2: $\neg b, c$	$\neg b, c$	$\neg b, c$	c	✓	$\neg b, c$	c	✓
Cl. 3: $\neg c, \neg a$	3	$\neg c$	$\neg c$	{} ✗	✓	✓	✓
BCP	-	b	c	-	-	c	-
Decision	a	-	-	$\neg a$	b	-	SAT

Pure Literals

- **Pure Literal:**
  - Unassigned literal
  - Complement does not occur in any unsatisfied clause
- Pure literals → set to **TRUE**

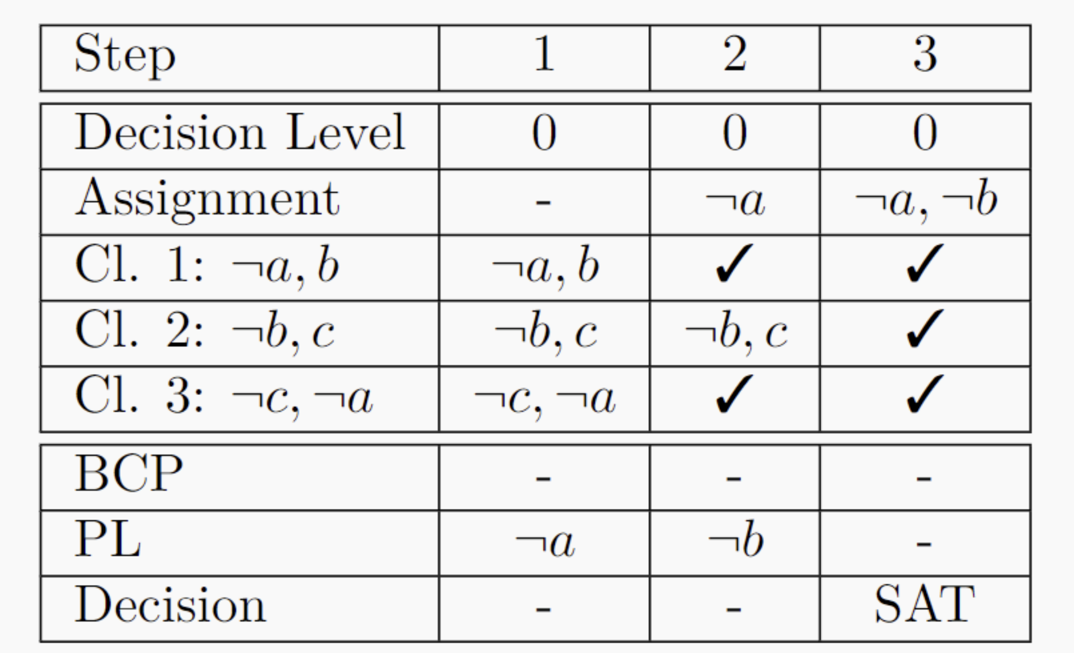
DPLL + BCP + Pure Literal Example

- $\varphi \coloneqq (\neg a \lor b) \land (\neg b \lor c) \land (\neg c \lor \neg a)$

Step	1	2	3	4	5	6	7	8	9
Dec. Level	0	0	0						
Assignment	{ }	¬a	¬a ¬b						
CL1: $\{\neg a, b\}$	¬a, b	✓	✓						
CL2: $\{\neg b, c\}$	¬b, c	¬b, c	✓						
CL3: $\{\neg c, \neg a\}$	¬c, ¬a	✓	✓						
BCP									
Pure Literal	¬a	¬b							
Decision			SAT						

DPLL+BCP+Pure Literals Example

- $\varphi \coloneqq (\neg a \vee b) \wedge (\neg b \vee c) \wedge (\neg c \vee \neg a)$



Step	1	2	3
Decision Level	0	0	0
Assignment	-	$\neg a$	$\neg a, \neg b$
Cl. 1: $\neg a, b$	$\neg a, b$	✓	✓
Cl. 2: $\neg b, c$	$\neg b, c$	$\neg b, c$	✓
Cl. 3: $\neg c, \neg a$	$\neg c, \neg a$	✓	✓
BCP	-	-	-
PL	$\neg a$	$\neg b$	-
Decision	-	-	SAT

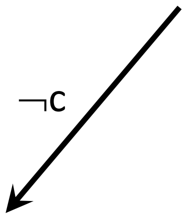
Clause Learning

Clause Learning

1. $(a \vee \neg c)$
2. $(b \vee \neg c)$
3. $(\neg a \vee \neg b \vee c)$
4. $(\neg a \vee \neg b)$
5. $(\neg a \vee b)$
6. $(a \vee \neg b)$
7. $(a \vee b)$

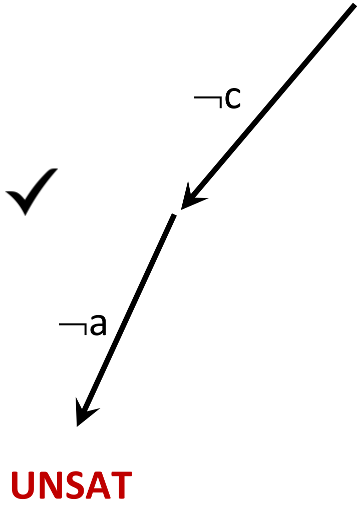
Clause Learning

1. $(a \vee \neg c)$ ✓
2. $(b \vee \neg c)$ ✓
3. $(\neg a \vee \neg b \vee c)$
4. $(\neg a \vee \neg b)$
5. $(\neg a \vee b)$
6. $(a \vee \neg b)$
7. $(a \vee b)$



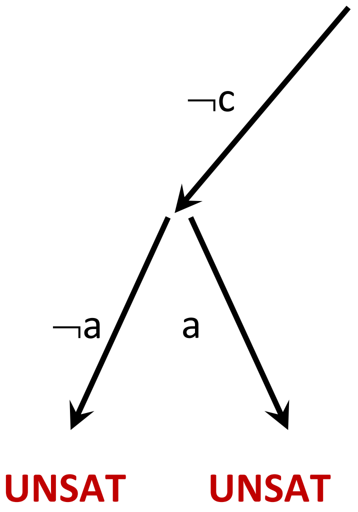
Clause Learning

1. $(a \vee \neg c)$ ✓
2. $(b \vee \neg c)$ ✓
3. $(\neg a \vee \neg b \vee c)$ ✓
4. $(\neg a \vee \neg b)$ ✓
5. $(\neg a \vee b)$ ✓
6. $(a \vee \neg b)$
7. $(a \vee b)$



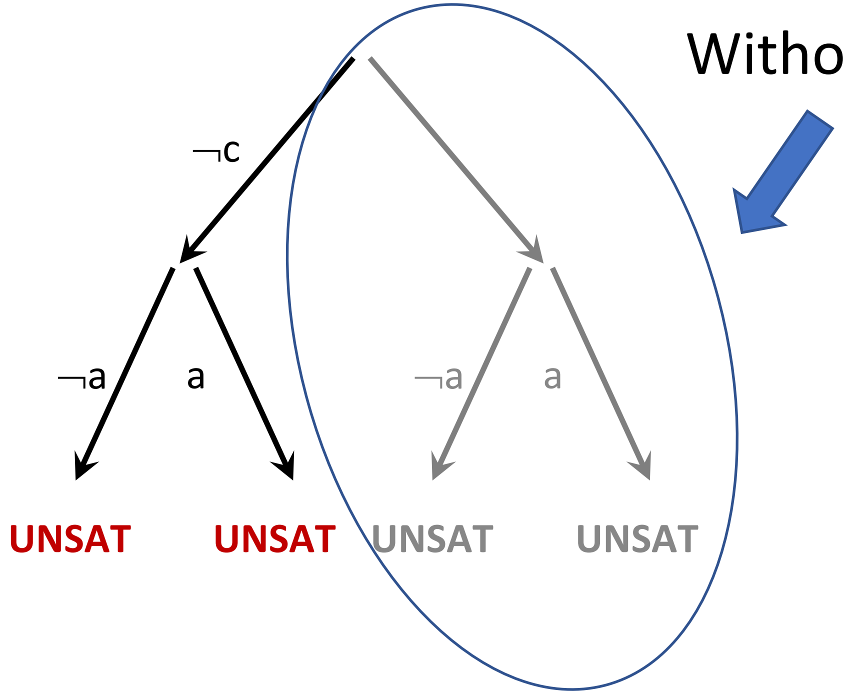
Clause Learning

1. (a ∨ ¬c) ✓
2. (b ∨ ¬c) ✓
3. (¬a ∨ ¬b ∨ c)
4. (¬a ∨ ¬b)
5. (¬a ∨ b)
6. (a ∨ ¬b) ✓
7. (a ∨ b) ✓



Clause Learning

1. $(a \vee \neg c)$
2. $(b \vee \neg c)$
3. $(\neg a \vee \neg b \vee c)$
4. $(\neg a \vee \neg b)$
5. $(\neg a \vee b)$
6. $(a \vee \neg b)$
7. $(a \vee b)$



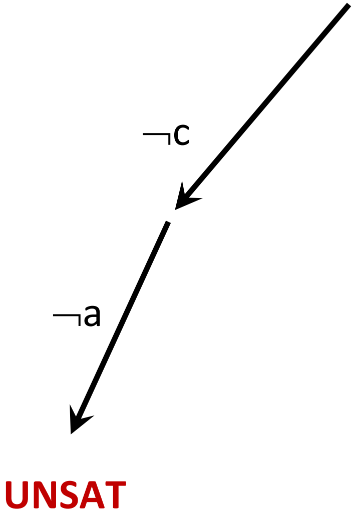
Without learning

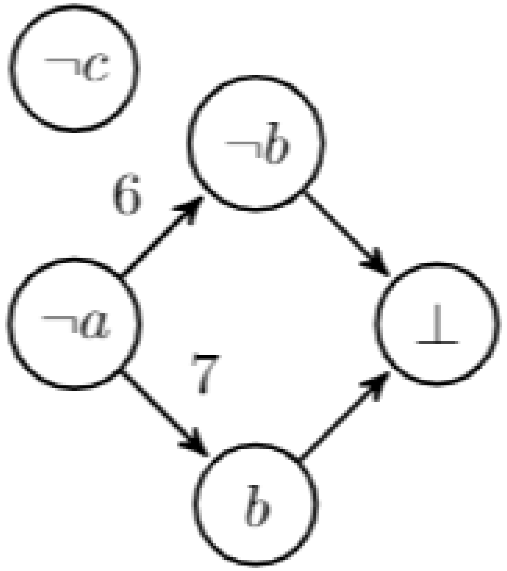
Problem is with "a":
➔ No need to try c=TRUE!

Conflict Graph

- Draw conflict graph for every conflict
- Illustrates decisions involved in conflict

1. $(a \lor \neg c)$
2. $(b \lor \neg c)$
3. $(\neg a \lor \neg b \lor c)$
4. $(\neg a \lor \neg b)$
5. $(\neg a \lor b)$
6. $(a \lor \neg b)$
7. $(a \lor b)$

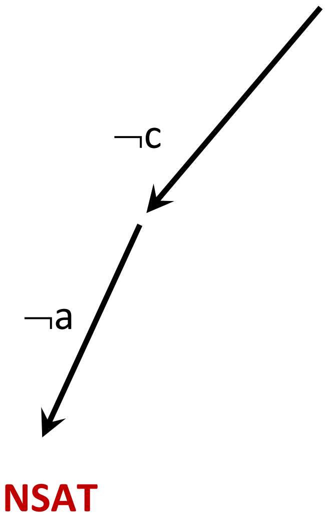




Conflict Graph

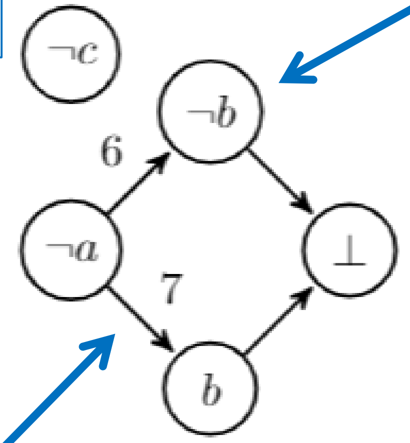
- Draw conflict graph for every conflict
- Illustrates decisions involved in conflict

1. (a ∨ ¬c)
2. (b ∨ ¬c)
3. (¬a ∨ ¬b ∨ c)
4. (¬a ∨ ¬b)
5. (¬a ∨ b)
6. (a ∨ ¬b)
7. (a ∨ b)



Root nodes: Decisions

Internal nodes: Implied (by BCP)



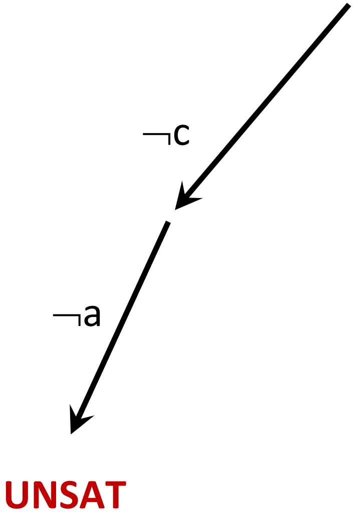
Clause numbers: b is implied by ¬a via clause 7

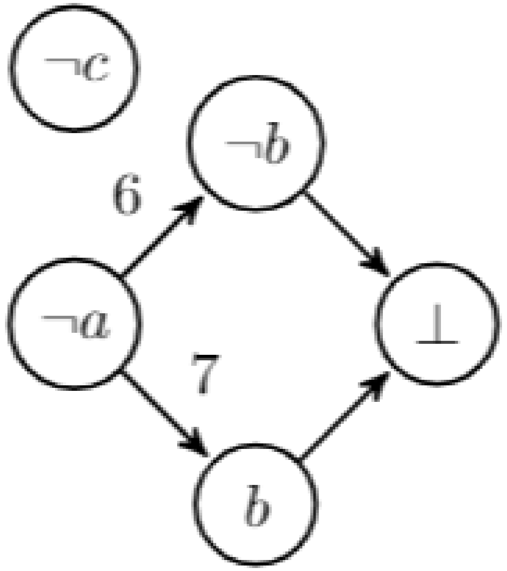
Conflict Node

Conflict Graph

- Draw conflict graph for every conflict
- Illustrates decisions involved in conflict
- To avoid conflict: change at least one decision that was involved

1. (a ∨ ¬c)
2. (b ∨ ¬c)
3. (¬a ∨ ¬b ∨ c)
4. (¬a ∨ ¬b)
5. (¬a ∨ b)
6. (a ∨ ¬b)
7. (a ∨ b)

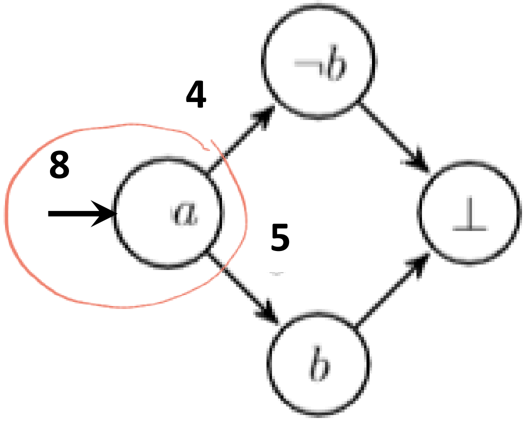




➔ Learn New Clause: (a)

Backtracking

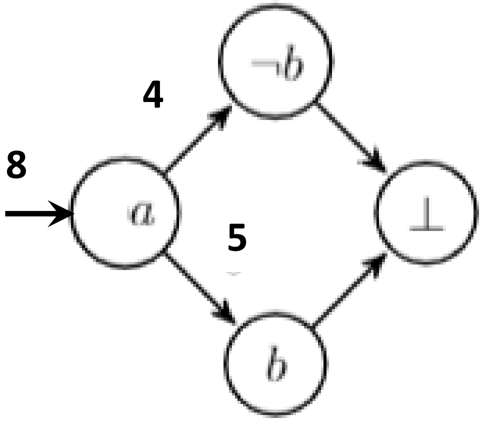
1. (a ∨ ¬c) ✓
2. (b ∨ ¬c)
3. (¬a ∨ ¬b ∨ c)
4. (¬a ∨ ¬b)
5. (¬a ∨ b)
6. (a ∨ ¬b) ✓
7. (a ∨ b) ✓
8. a ✓



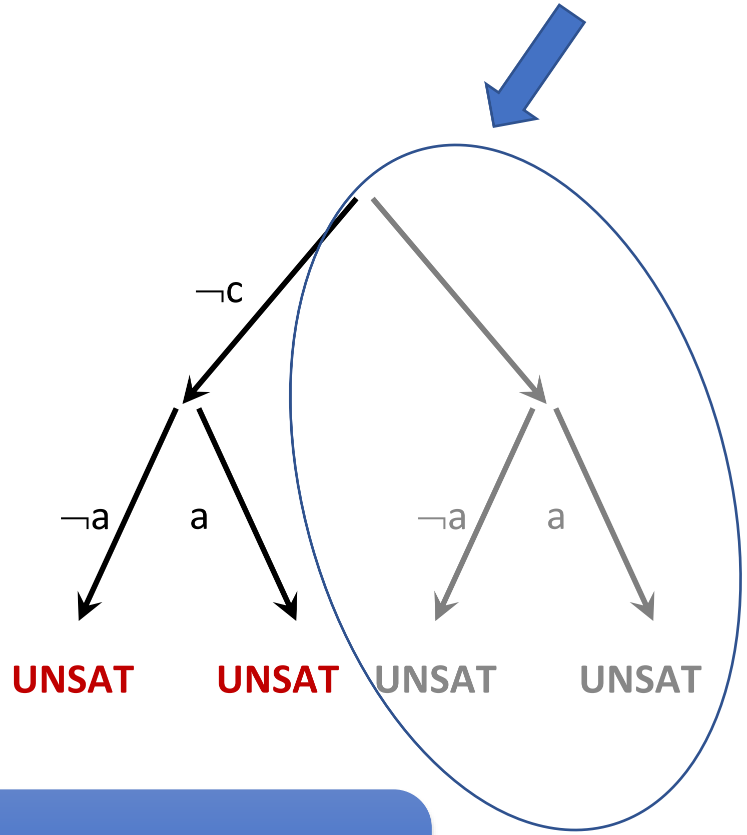
No decision was necessary
➔ We learn: UNSAT

Backtracking

1. $(a \vee \neg c)$ ✓
2. $(b \vee \neg c)$
3. $(\neg a \vee \neg b \vee c)$
4. $(\neg a \vee \neg b)$
5. $(\neg a \vee b)$
6. $(a \vee \neg b)$ ✓
7. $(a \vee b)$ ✓
8. a ✓



No need to search here



No decision was necessary
➔ We learn: UNSAT

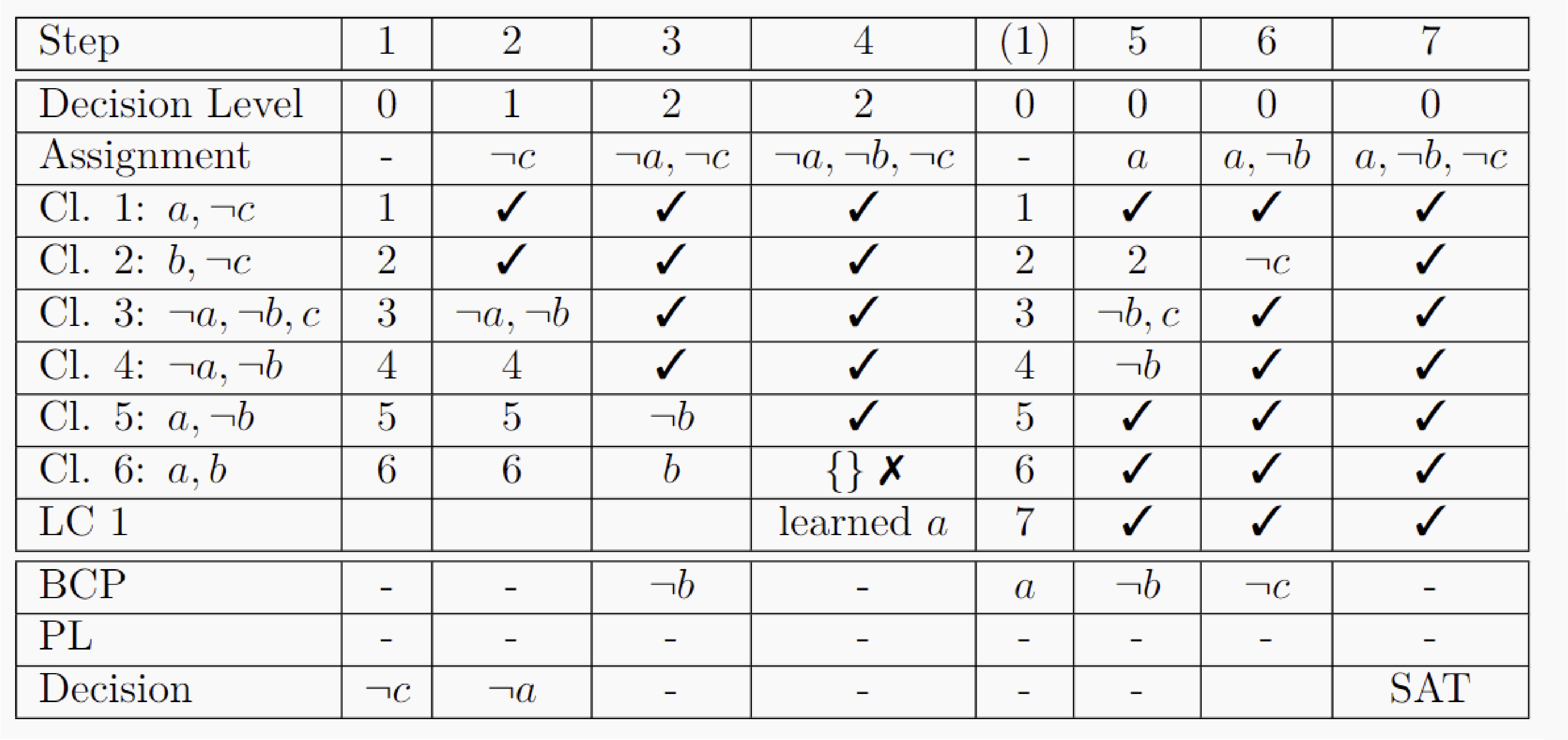
DPLL + BCP + PL + Learning

- $\varphi \coloneqq (a \vee \neg c) \wedge (b \vee \neg c) \wedge (\neg a \vee \neg b \vee c) \wedge (\neg a \wedge \neg b) \wedge (\neg a \vee b) \wedge (a \vee \neg b) \wedge (a \vee b)$

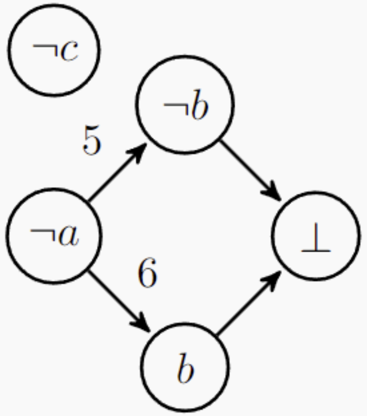
$Order: \neg c < c < \neg a < a < \neg b < b$

Step	1	2	3	4	5	6	7	8	9
Dec. Level	0	1	2	2	0	0			
Assignment	{}	¬c	¬a ¬c	¬a ¬b ¬c	a	a ¬b			
CL1: $\{a, \neg c\}$	1	✓	✓	✓	✓	✓			
CL2: $\{b, \neg c\}$	2	✓	✓	✓	b, ¬c	¬c			
CL3: $\{\neg a, \neg b, c\}$	3	¬a, ¬b	✓	✓	¬b, c	✓			
CL4: $\{\neg a, \neg b\}$	4	4	✓	✓	¬b	✓			
CL5: $\{\neg a, b\}$	5	5	✓	✓	b	{}			
CL6: $\{a, \neg b\}$	6	6	¬b	✓	✓	✓			
CL7: $\{a, b\}$	7	7	b .	{} X	✓	✓			
LC: a				learned a					
BCP			¬b	a	¬b	UNSAT			
Pure Literal									
Decision	¬c	¬a							

DPLL + BCP + PL + Learning



Step	1	2	3	4	(1)	5	6	7
Decision Level	0	1	2	2	0	0	0	0
Assignment	-	$\neg c$	$\neg a, \neg c$	$\neg a, \neg b, \neg c$	-	a	$a, \neg b$	$a, \neg b, \neg c$
Cl. 1: $a, \neg c$	1	✓	✓	✓	1	✓	✓	✓
Cl. 2: $b, \neg c$	2	✓	✓	✓	2	2	$\neg c$	✓
Cl. 3: $\neg a, \neg b, c$	3	$\neg a, \neg b$	✓	✓	3	$\neg b, c$	✓	✓
Cl. 4: $\neg a, \neg b$	4	4	✓	✓	4	$\neg b$	✓	✓
Cl. 5: $a, \neg b$	5	5	$\neg b$	✓	5	✓	✓	✓
Cl. 6: a, b	6	6	b	{} ✗	6	✓	✓	✓
LC 1				learned a	7	✓	✓	✓
BCP	-	-	$\neg b$	-	a	$\neg b$	$\neg c$	-
PL	-	-	-	-	-	-	-	-
Decision	$\neg c$	$\neg a$	-	-	-	-		SAT



Backtrack Level

- Ongoing Research Problem

- In this course:
  - → **earliest level where conflict clause is a unit clause**
  - New clause immediately be used

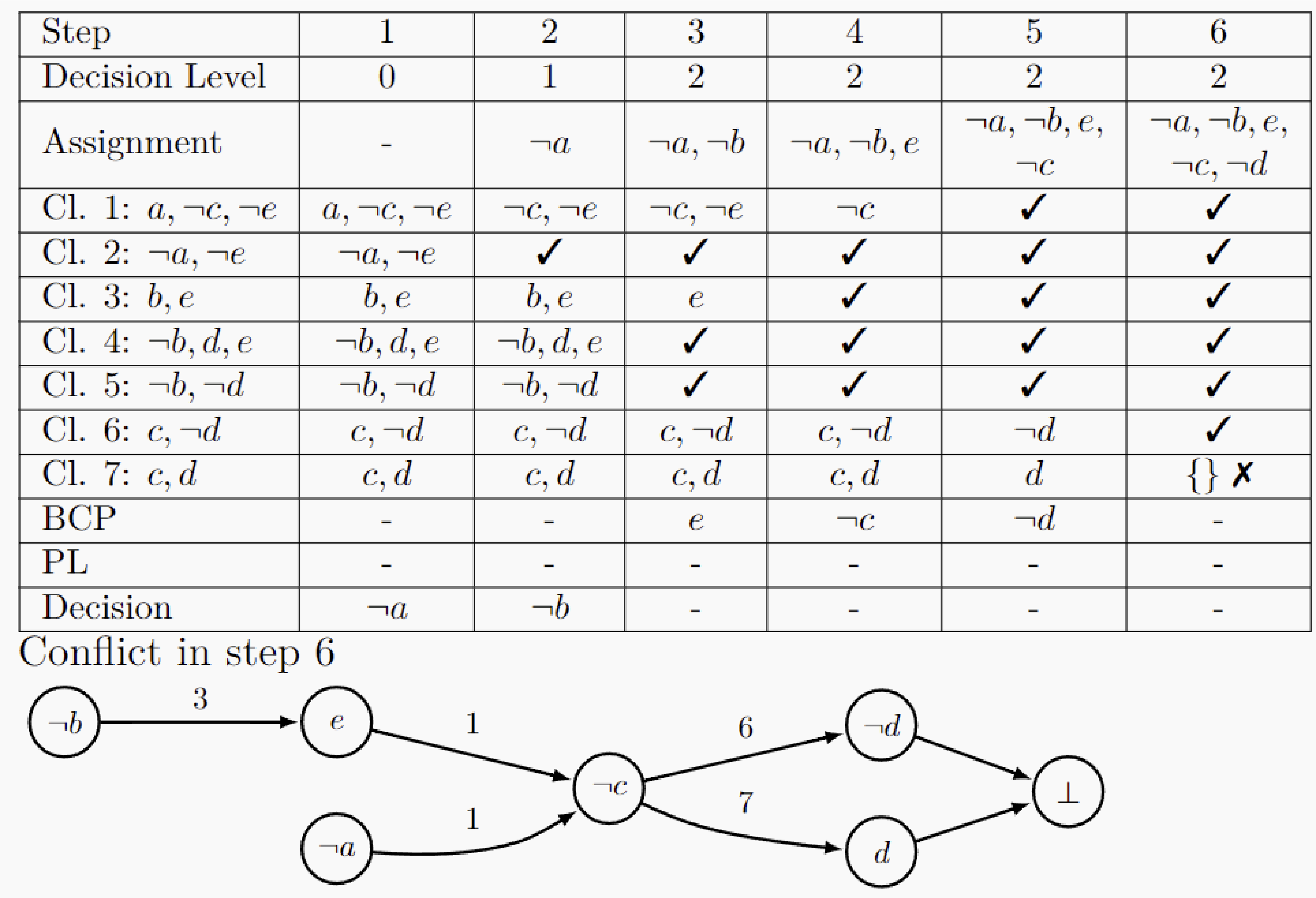
$\varphi := \; (a \vee \neg c \vee \neg e) \wedge (\neg a \vee \neg e) \wedge (b \vee e) \wedge (\neg b \vee d \vee e) \wedge (\neg b \vee \neg d) \wedge (c \vee \neg d) \wedge (c \vee d)$

*Decision heuristic: alphabetical order starting with the **negative** phase*

Step	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**	**16**
Dec. Level																
Assignment																
1: $\{a, \neg c, \neg e\}$																
2: $\{\neg a, \neg e\}$																
3: $\{b, e\}$																
4: $\{\neg b, d, e\}$																
5: $\{\neg b, \neg d\}$																
6: $\{c, \neg d\}$																
7: $\{c, d\}$																
LC 1																
LC 2																
BCP																
Pure Literal																
Decision																

$\varphi := (a \lor \neg c \lor \neg e) \land (\neg a \lor \neg e) \land (b \lor e) \land (\neg b \lor d \lor e) \land (\neg b \lor \neg d) \land (c \lor \neg d) \land (c \lor d)$

*Decision heuristic: alphabetical order starting with the **negative** phase*

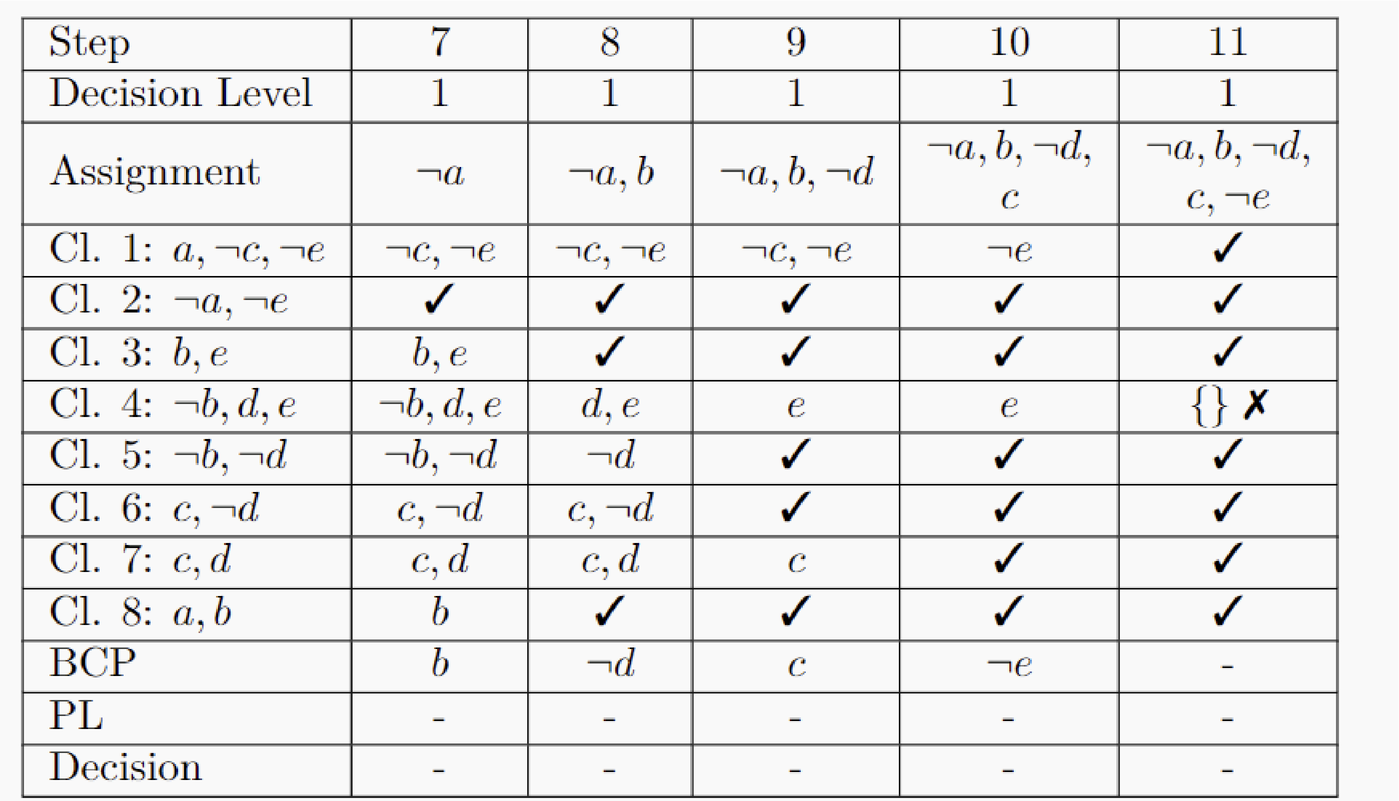


Step	1	2	3	4	5	6
Decision Level	0	1	2	2	2	2
Assignment	-	$\neg a$	$\neg a, \neg b$	$\neg a, \neg b, e$	$\neg a, \neg b, e,$ $\neg c$	$\neg a, \neg b, e,$ $\neg c, \neg d$
Cl. 1: $a, \neg c, \neg e$	$a, \neg c, \neg e$	$\neg c, \neg e$	$\neg c, \neg e$	$\neg c$	✓	✓
Cl. 2: $\neg a, \neg e$	$\neg a, \neg e$	✓	✓	✓	✓	✓
Cl. 3: b, e	b, e	b, e	e	✓	✓	✓
Cl. 4: $\neg b, d, e$	$\neg b, d, e$	$\neg b, d, e$	✓	✓	✓	✓
Cl. 5: $\neg b, \neg d$	$\neg b, \neg d$	$\neg b, \neg d$	✓	✓	✓	✓
Cl. 6: $c, \neg d$	$c, \neg d$	$c, \neg d$	$c, \neg d$	$c, \neg d$	$\neg d$	✓
Cl. 7: c, d	c, d	c, d	c, d	c, d	d	{} ✗
BCP	-	-	e	$\neg c$	$\neg d$	-
PL	-	-	-	-	-	-
Decision	$\neg a$	$\neg b$	-	-	-	-

Conflict in step 6

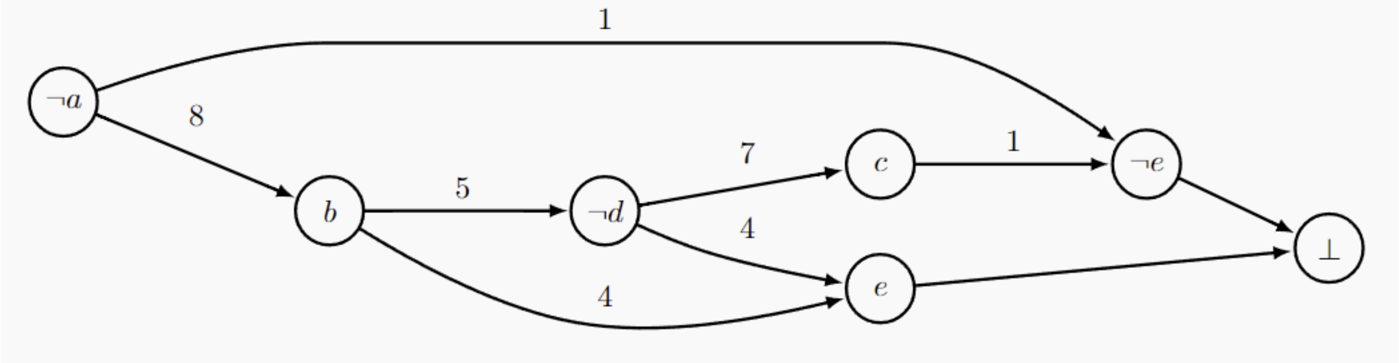
$\varphi := (a \vee \neg c \vee \neg e) \wedge (\neg a \vee \neg e) \wedge (b \vee e) \wedge (\neg b \vee d \vee e) \wedge (\neg b \vee \neg d) \wedge (c \vee \neg d) \wedge (c \vee d)$

*Decision heuristic: alphabetical order starting with the **negative** phase*



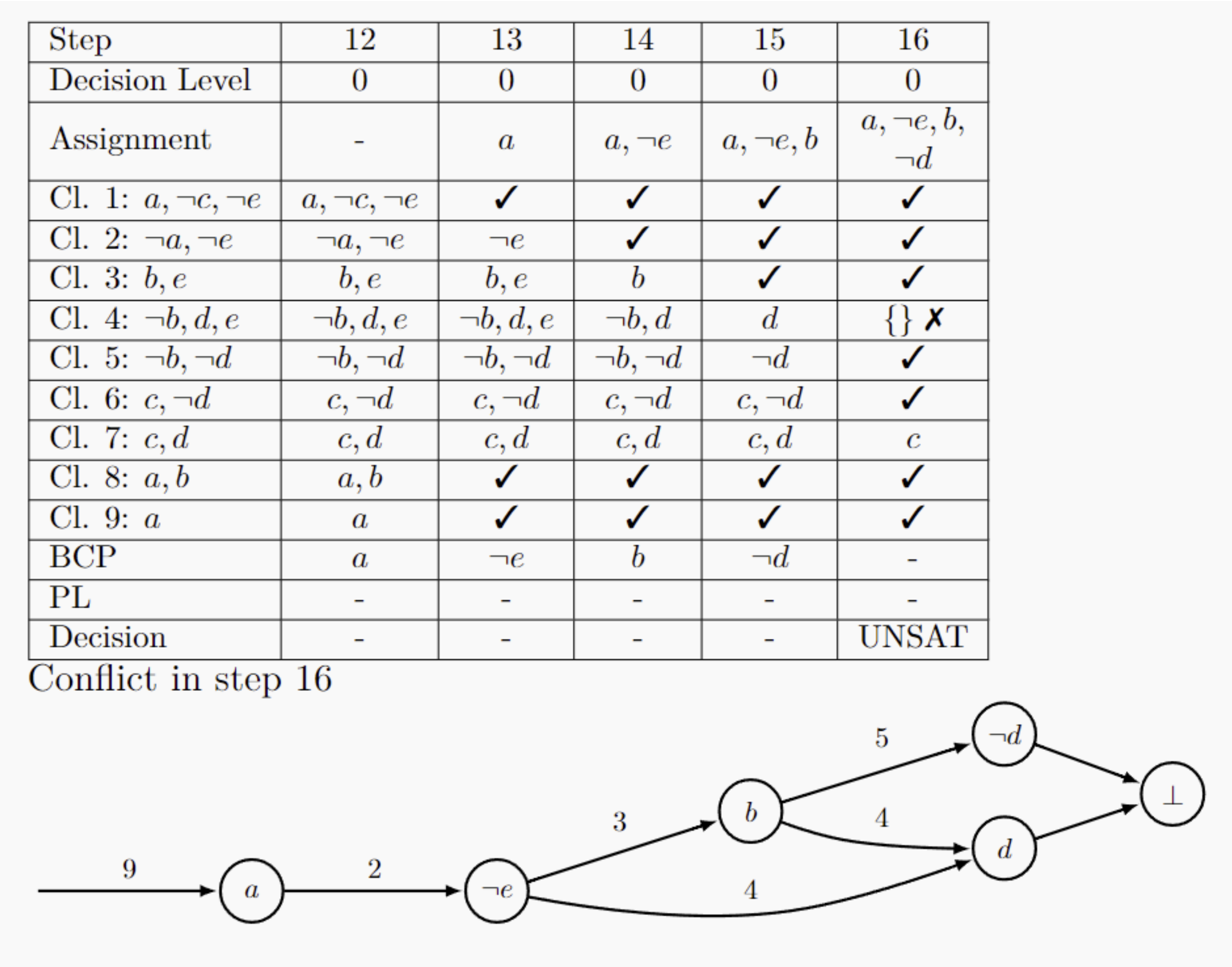
Step	7	8	9	10	11
Decision Level	1	1	1	1	1
Assignment	$\neg a$	$\neg a, b$	$\neg a, b, \neg d$	$\neg a, b, \neg d,$ c	$\neg a, b, \neg d,$ $c, \neg e$
Cl. 1: $a, \neg c, \neg e$	$\neg c, \neg e$	$\neg c, \neg e$	$\neg c, \neg e$	$\neg e$	✓
Cl. 2: $\neg a, \neg e$	✓	✓	✓	✓	✓
Cl. 3: b, e	b, e	✓	✓	✓	✓
Cl. 4: $\neg b, d, e$	$\neg b, d, e$	d, e	e	e	$\{\}$ ✗
Cl. 5: $\neg b, \neg d$	$\neg b, \neg d$	$\neg d$	✓	✓	✓
Cl. 6: $c, \neg d$	$c, \neg d$	$c, \neg d$	✓	✓	✓
Cl. 7: c, d	c, d	c, d	c	✓	✓
Cl. 8: a, b	b	✓	✓	✓	✓
BCP	b	$\neg d$	c	$\neg e$	-
PL	-	-	-	-	-
Decision	-	-	-	-	-

Conflict in step 11



$\varphi := (a \vee \neg c \vee \neg e) \wedge (\neg a \vee \neg e) \wedge (b \vee e) \wedge (\neg b \vee d \vee e) \wedge (\neg b \vee \neg d) \wedge (c \vee \neg d) \wedge (c \vee d)$

*Decision heuristic: alphabetical order starting with the **negative** phase*



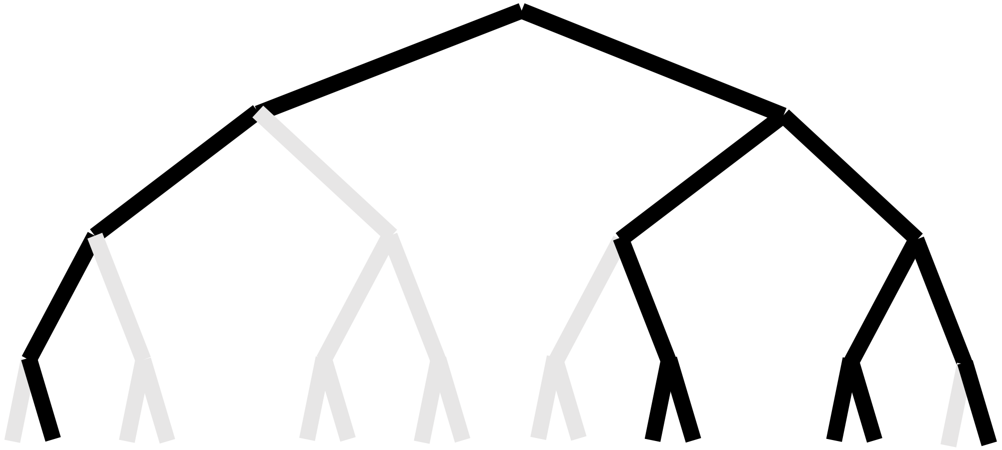
Step	12	13	14	15	16
Decision Level	0	0	0	0	0
Assignment	-	a	$a, \neg e$	$a, \neg e, b$	$a, \neg e, b, \neg d$
Cl. 1: $a, \neg c, \neg e$	$a, \neg c, \neg e$	✓	✓	✓	✓
Cl. 2: $\neg a, \neg e$	$\neg a, \neg e$	$\neg e$	✓	✓	✓
Cl. 3: b, e	b, e	b, e	b	✓	✓
Cl. 4: $\neg b, d, e$	$\neg b, d, e$	$\neg b, d, e$	$\neg b, d$	d	{} ✗
Cl. 5: $\neg b, \neg d$	$\neg b, \neg d$	$\neg b, \neg d$	$\neg b, \neg d$	$\neg d$	✓
Cl. 6: $c, \neg d$	$c, \neg d$	$c, \neg d$	$c, \neg d$	$c, \neg d$	✓
Cl. 7: c, d	c, d	c, d	c, d	c, d	c
Cl. 8: a, b	a, b	✓	✓	✓	✓
Cl. 9: a	a	✓	✓	✓	✓
BCP	a	$\neg e$	b	$\neg d$	-
PL	-	-	-	-	-
Decision	-	-	-	-	UNSAT

Conflict in step 16

DPLL + BCP + PL + Clause Learning

- Binary Search Tree
  - Worst Case: Exponential Time

- Pruning
  - Boolean Constraint Propagation (BCP)
  - Pure Literals
  - Learn Conflict Clauses



SAT Solver Output

- **Satisfiable**:
  - Satisfying Assignment

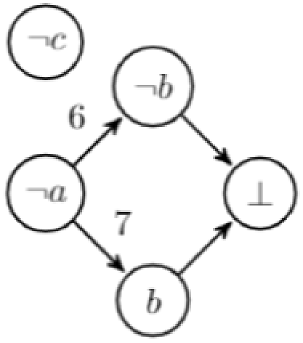
- **Unsatisfiable**
  - Proof of Unsatisfiability

Proving Unsatisfiability

- **Resolution Rule:**

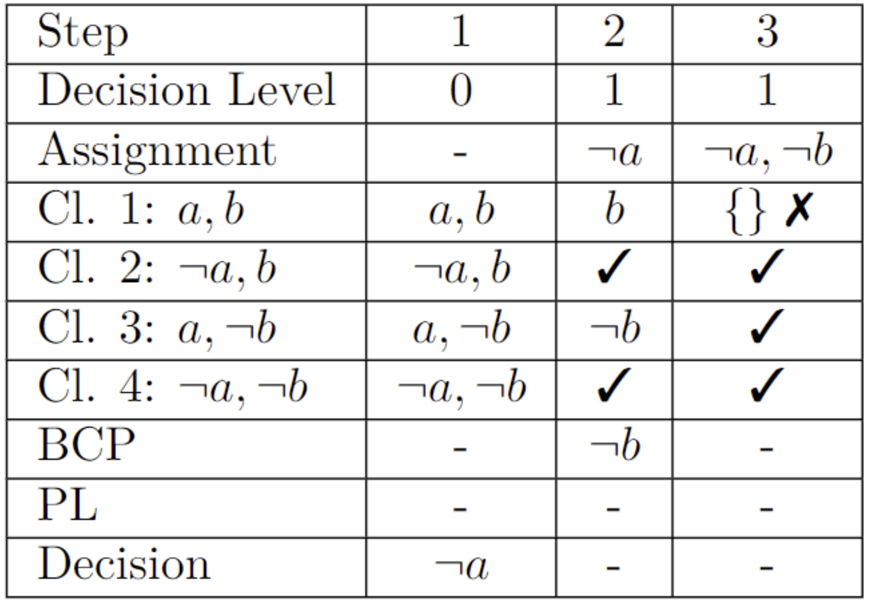
$$\frac{(a \vee b_1 \vee \ldots \vee b_n) \qquad (\neg a \vee c_1 \vee \ldots \vee c_m)}{(b_1 \vee \ldots \vee b_n \vee c_1 \vee \ldots \vee c_m)}$$

- Turn Conflict Graph Around
  - Select clause that implies conflict
  - Iteratively, resolve while back-traversing graph



$$\frac{7.\ a \vee b \qquad 6.\ a \vee \neg b}{8.\ a}$$

Resolution Proof

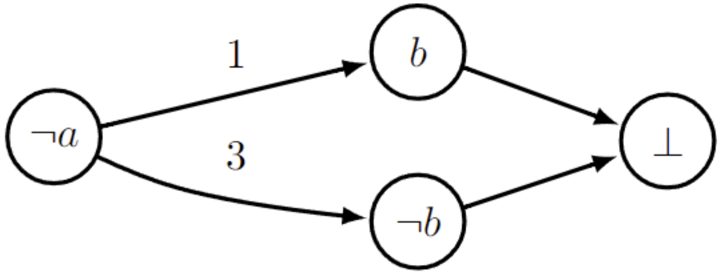


Step	1	2	3
Decision Level	0	1	1
Assignment	-	$\neg a$	$\neg a, \neg b$
Cl. 1: a, b	a, b	b	$\{\}$ ✗
Cl. 2: $\neg a, b$	$\neg a, b$	✓	✓
Cl. 3: $a, \neg b$	$a, \neg b$	$\neg b$	✓
Cl. 4: $\neg a, \neg b$	$\neg a, \neg b$	✓	✓
BCP	-	$\neg b$	-
PL	-	-	-
Decision	$\neg a$	-	-

Resolution Proof

Step	1	2	3
Decision Level	0	1	1
Assignment	-	$\neg a$	$\neg a, \neg b$
Cl. 1: a, b	a, b	b	$\{\}$ ✗
Cl. 2: $\neg a, b$	$\neg a, b$	✓	✓
Cl. 3: $a, \neg b$	$a, \neg b$	$\neg b$	✓
Cl. 4: $\neg a, \neg b$	$\neg a, \neg b$	✓	✓
BCP	-	$\neg b$	-
PL	-	-	-
Decision	$\neg a$	-	-

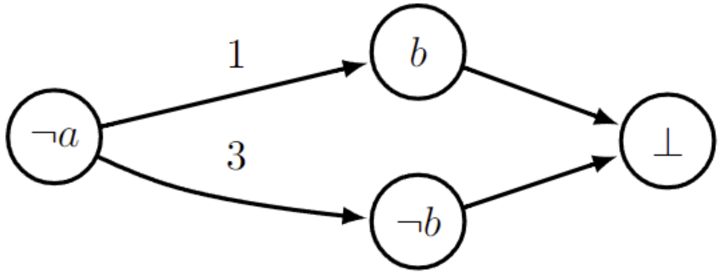
Conflict in step 3



Resolution Proof

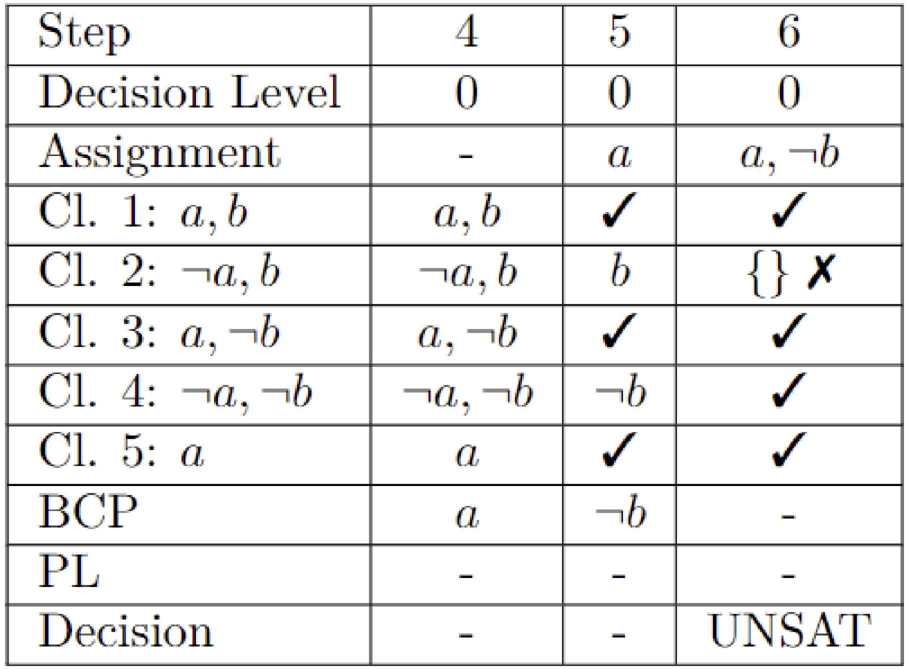
Step	1	2	3
Decision Level	0	1	1
Assignment	-	$\neg a$	$\neg a, \neg b$
Cl. 1: a, b	a, b	b	$\{\}$ ✗
Cl. 2: $\neg a, b$	$\neg a, b$	✓	✓
Cl. 3: $a, \neg b$	$a, \neg b$	$\neg b$	✓
Cl. 4: $\neg a, \neg b$	$\neg a, \neg b$	✓	✓
BCP	-	$\neg b$	-
PL	-	-	-
Decision	$\neg a$	-	-

Conflict in step 3



$$\frac{1.\ a \lor b \qquad\qquad 3.\ a \lor \neg b}{a}$$

Resolution Proof

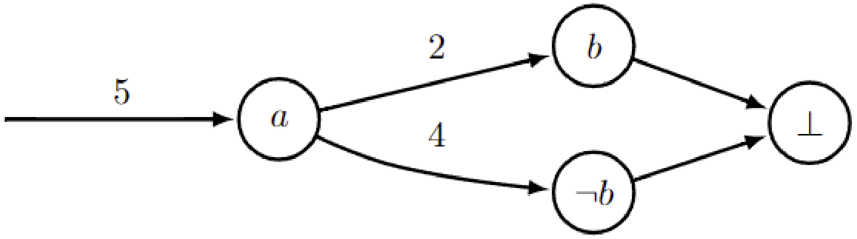


Step	4	5	6
Decision Level	0	0	0
Assignment	-	a	$a, \neg b$
Cl. 1: a, b	a, b	✓	✓
Cl. 2: $\neg a, b$	$\neg a, b$	b	$\{\}$ ✗
Cl. 3: $a, \neg b$	$a, \neg b$	✓	✓
Cl. 4: $\neg a, \neg b$	$\neg a, \neg b$	$\neg b$	✓
Cl. 5: a	a	✓	✓
BCP	a	$\neg b$	-
PL	-	-	-
Decision	-	-	UNSAT

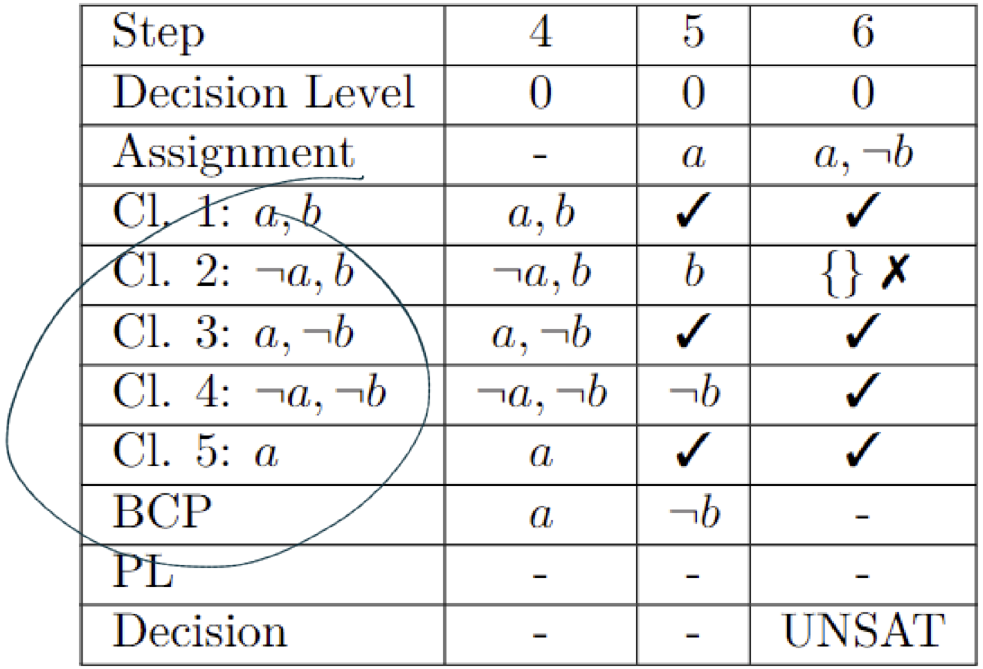
Resolution Proof

Step	4	5	6
Decision Level	0	0	0
Assignment	-	a	$a, \neg b$
Cl. 1: a, b	a, b	✓	✓
Cl. 2: $\neg a, b$	$\neg a, b$	b	{} ✗
Cl. 3: $a, \neg b$	$a, \neg b$	✓	✓
Cl. 4: $\neg a, \neg b$	$\neg a, \neg b$	$\neg b$	✓
Cl. 5: a	a	✓	✓
BCP	a	$\neg b$	-
PL	-	-	-
Decision	-	-	UNSAT

Conflict in step 6

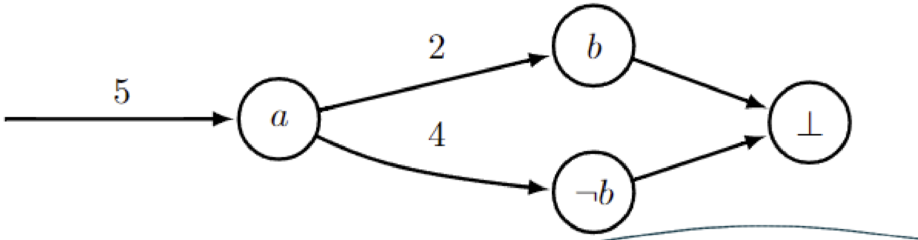


Resolution Proof



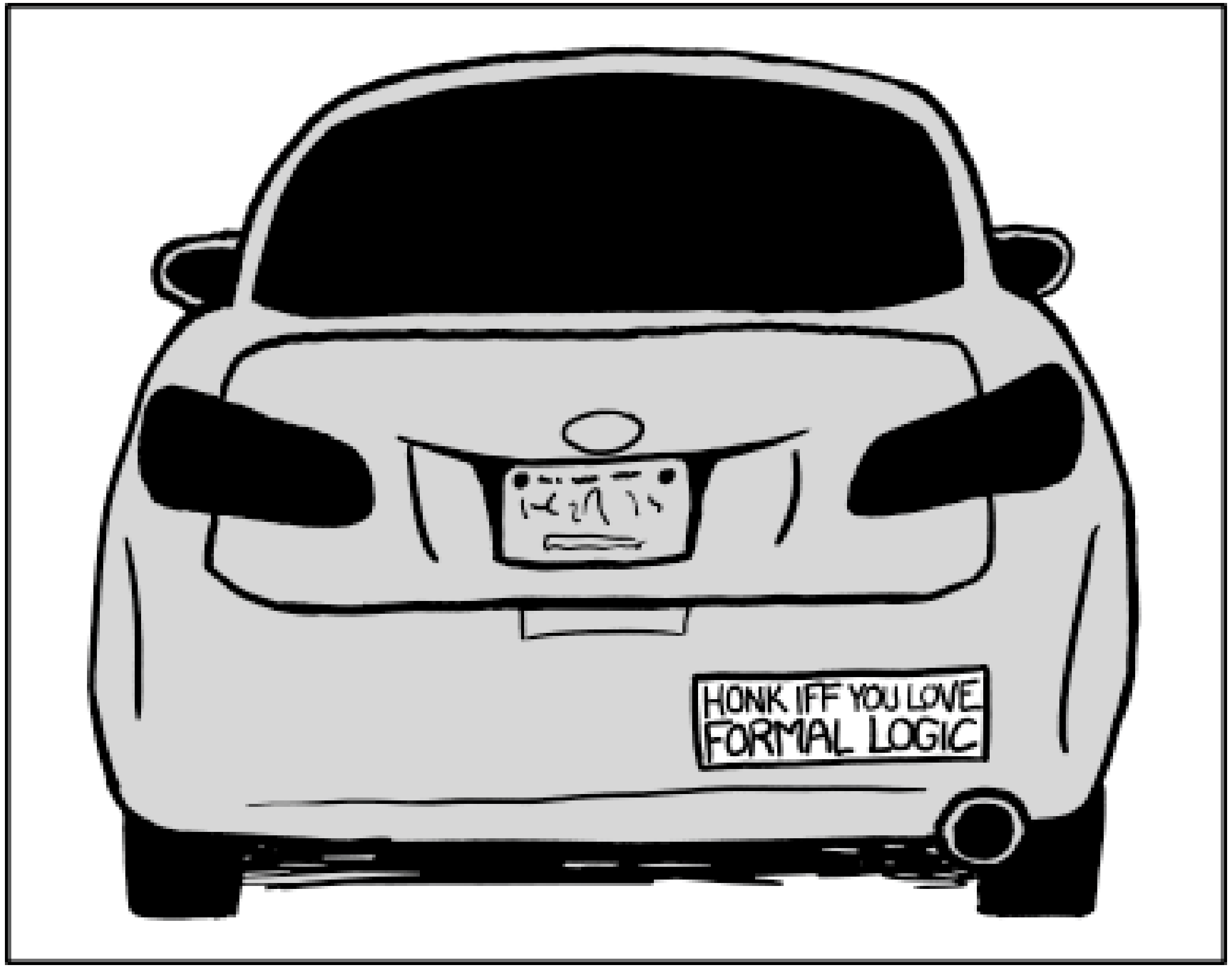
Step	4	5	6
Decision Level	0	0	0
Assignment	-	a	$a, \neg b$
Cl. 1: a, b	a, b	✓	✓
Cl. 2: $\neg a, b$	$\neg a, b$	b	{} ✗
Cl. 3: $a, \neg b$	$a, \neg b$	✓	✓
Cl. 4: $\neg a, \neg b$	$\neg a, \neg b$	$\neg b$	✓
Cl. 5: a	a	✓	✓
BCP	a	$\neg b$	-
PL	-	-	-
Decision	-	-	UNSAT

Conflict in step 6



$$\dfrac{\dfrac{2.\ \neg a \lor b \qquad 4.\ \neg a \lor \neg b}{\neg a} \qquad 5.\ a}{\bot}$$

Thank You



https://xkcd.com/1033/