

# Satisfiability Checking

## 03 Propositional logic II

Prof. Dr. Erika Ábrahám

RWTH Aachen University  
Informatik 2  
LuFG Theory of Hybrid Systems

WS 23/24

# 03 Propositional logic II

1 Normal forms

2 Enumeration and deduction

literal 为变量(atomic proposition)或变量(atomic proposition)的否定

此处atomic proposition = boolean variable

■ Definition: A **literal** is either a variable or the negation of a variable.

■ Example:  $\varphi = \neg(a \vee \neg b)$

Variables:  $AP(\varphi) = \{a, b\}$

Literals:  $lit(\varphi) = \{a, \neg b\}$  即, formula中出现的包含正负的ap 为 literals

■ Note: Equivalent formulae can have different literals.

Example:  $\varphi' = \neg a \wedge b$

Literals:  $lit(\varphi') = \{\neg a, b\}$

- Definition: a **term** is a **conjunction** of literals

- Example:  $(a \wedge \neg b \wedge c)$  使用 $\wedge$ 连接的为term

- Definition: a **clause** is a **disjunction** of literals

- Example:  $(a \vee \neg b \vee c)$  使用 $\vee$ 连接的为clause

没有“ $\vee$ ”号,  $(c)$ , 也第一个 clause

# Negation Normal Form (NNF)

A formula is in **Negation Normal Form (NNF)** iff

- 1 it contains **only  $\neg$ ,  $\wedge$  and  $\vee$**  as **connectives** and
- 2 **only variables** are **negated**.

Examples:

- $a \rightarrow b$  is **not** in NNF
- $\neg(a \vee \neg b)$  is **not** in NNF
- $\neg a \wedge b$  is **in** NNF

1. 逻辑符号只包含  $\neg$ ,  $\wedge$  和  $\vee$

2.  $\neg$  只出现在变量(ap)之前

# Converting to NNF

- Every formula can be converted to NNF in linear time:
  - eliminate all connectives other than  $\wedge$ ,  $\vee$ ,  $\neg$  and
  - use De Morgan and double-negation rules to push negations to operands.
- **Example:**  $\varphi = \neg(a \rightarrow \neg b)$ 
  - eliminate ' $\rightarrow$ ':  $\varphi = \neg(\neg a \vee \neg b)$
  - push negation using De Morgan:  $\varphi = (\neg\neg a \wedge \neg\neg b)$
  - use double-negation rule:  $\varphi = (a \wedge b)$
- Transformation to NNF can be done with an effort (time and space) that is linear in the size of the formula, if the formula does not contain nested if-and-only-if operators.  
**Idea:** Number of transformation steps  $\leq$  number of operands in the formula.

# Disjunctive Normal Form (DNF)

- Definition: A formula is said to be in **Disjunctive Normal Form (DNF)** iff it is a **disjunction of terms**.
- In other words, it is a formula of the form

$$\bigvee_i \left( \bigwedge_j l_{i,j} \right)$$

where  $l_{i,j}$  is the  $j$ -th literal in the  $i$ -th term.

- Example:

$$\varphi = (a \wedge \neg b \wedge c) \vee (\neg a \wedge d) \vee (b) \text{ is in DNF}$$

- **DNF** is a **special case** of **NNF**.

# Converting to DNF

- Every formula can be converted to DNF in **exponential** time and space:

- 1 Convert to NNF

- 2 Distribute disjunctions following the rule:

$$\models \varphi_1 \wedge (\varphi_2 \vee \varphi_3) \leftrightarrow (\varphi_1 \wedge \varphi_2) \vee (\varphi_1 \wedge \varphi_3)$$

- **Example:**

$$\begin{aligned}\varphi &= (a \vee b) \wedge (\neg c \vee d) \\ &= ((a \vee b) \wedge (\neg c)) \vee ((a \vee b) \wedge d) \\ &= (a \wedge \neg c) \vee (b \wedge \neg c) \vee (a \wedge d) \vee (b \wedge d)\end{aligned}$$

- Now consider  $\varphi_n = (a_1 \vee b_1) \wedge (a_2 \vee b_2) \wedge \dots \wedge (a_n \vee b_n)$ .
- **Q:** How many terms will the DNF have?

**A:**  $2^n$



- Q: Is the following DNF formula satisfiable?

$$(a_1 \wedge a_2 \wedge \neg a_1) \vee (a_2 \wedge a_1) \vee (a_2 \wedge \neg a_3 \wedge a_3)$$

A: Yes, because the term  $a_2 \wedge a_1$  is satisfiable.

- Q: What is the complexity of the satisfiability check of DNF formulae?

A: Linear (time and space).

- Q: Can there be any polynomial transformation into DNF?

- A: No, it would violate the NP-completeness of the problem.

# Conjunctive Normal Form (CNF)

- Definition: A formula is said to be in **Conjunctive Normal Form (CNF)** iff it is a conjunction of clauses.
- In other words, it is a formula of the form

$$\bigwedge_i \left( \bigvee_j l_{i,j} \right)$$

where  $l_{i,j}$  is the  $j$ -th literal in the  $i$ -th clause.

- Example:

$$\varphi = (a \vee \neg b \vee c) \wedge (\neg a \vee d) \wedge (b) \text{ is in CNF}$$

- Also **CNF is a special case of NNF.**

# Converting to CNF

- Every formula can be converted to CNF in **exponential** time and space:

- 1 Convert to NNF

- 2 Distribute disjunctions following the rule:

$$\models \varphi_1 \vee (\varphi_2 \wedge \varphi_3) \leftrightarrow (\varphi_1 \vee \varphi_2) \wedge (\varphi_1 \vee \varphi_3)$$

- Consider the formula  $\varphi = (a_1 \wedge b_1) \vee (a_2 \wedge b_2)$ .

Transformation:  $(a_1 \vee a_2) \wedge (a_1 \vee b_2) \wedge (b_1 \vee a_2) \wedge (b_1 \vee b_2)$

- Now consider  $\varphi_n = (a_1 \wedge b_1) \vee (a_2 \wedge b_2) \vee \dots \vee (a_n \wedge b_n)$ .

Q: How many clauses does the resulting CNF have?

A:  $2^n$

# Converting to CNF: Tseitin's encoding

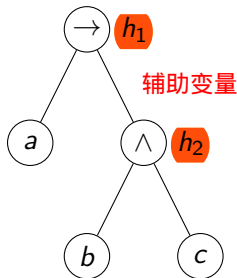
- Every formula can be converted to CNF in **linear** time and space if new variables are added.
- The original and the converted formulae are **not equivalent** but **equi-satisfiable**.

- Consider the formula

$$\varphi = (a \rightarrow (b \wedge c))$$

- Associate a new auxiliary variable with each inner (non-leaf) node. 对非叶子节点使用辅助变量
- Add constraints that define these new variables.
- Finally, enforce the truth of the root node.

Parse tree:



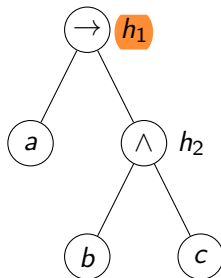
# Converting to CNF: Tseitin's encoding

## ■ Tseitin's encoding:

$$(h_1 \leftrightarrow (a \rightarrow h_2)) \wedge$$

$$(h_2 \leftrightarrow (b \wedge c)) \wedge$$

$$(h_1)$$



## ■ Each node's encoding has a CNF representation with 3 or 4 clauses.

$$h_1 \leftrightarrow (a \rightarrow h_2) \text{ in CNF: } (h_1 \vee a) \wedge (h_1 \vee \neg h_2) \wedge (\neg h_1 \vee \neg a \vee h_2)$$

$$h_2 \leftrightarrow (b \wedge c) \text{ in CNF: } (\neg h_2 \vee b) \wedge (\neg h_2 \vee c) \wedge (h_2 \vee \neg b \vee \neg c)$$

# Converting to CNF: Tseitin's encoding

- Let's go back to

$$\varphi_n = \underbrace{(x_1 \wedge y_1)}_{h_1} \vee \underbrace{(x_2 \wedge y_2)}_{h_2} \vee \cdots \vee \underbrace{(x_n \wedge y_n)}_{h_n} \stackrel{\text{SAT}}{=} h_1 \vee h_2 \cdots \vee h_n$$

- With Tseitin's encoding we need:

- n auxiliary variables  $h_1, \dots, h_n$ .
- Each adds 3 constraints.
- Top clause:  $(h_1 \vee \cdots \vee h_n)$

- Hence, we have

- $3n + 1$  clauses, instead of  $2^n$ .
- $3n$  variables rather than  $2n$ .

$$\stackrel{\text{SAT}}{=} (h_1 \leftrightarrow (x_1 \wedge y_1)) \\ \vee (h_2 \leftrightarrow (x_2 \wedge y_2)) \\ \vdots \\ \vee (h_n \leftrightarrow (x_n \wedge y_n))$$

$$(\neg h_1 \vee x_1) \wedge (\neg h_1 \vee y_1) \\ \wedge \neg$$

# Exercise

How many of the following propositional logic formulas are in NNF, DNF resp. CNF?

$\neg(a \vee \neg b)$	-	-	-
$(a \vee \neg b) \wedge (\neg a \vee b)$	NNF	-	CNF
$(x \wedge y) \vee \neg(x \wedge z)$	-	-	-
$a \wedge b \wedge c$	NNF	DNF	CNF
$a \rightarrow b$	-	-	-
$\neg p \vee \neg q$	NNF	DNF	CNF
$u$	NNF	DNF	CNF

a) NNF:4    DNF:4    CNF:5

b) NNF:4    DNF:3    CNF:4

c) NNF:5    DNF:4    CNF:4

d) NNF:5    DNF:3    CNF:5

# 03 Propositional logic II

1 Normal forms

2 Enumeration and deduction



# Two classes of algorithms for validity

- Q: Is  $\varphi$  satisfiable? (Is  $\neg\varphi$  valid?)
- Complexity: NP-Complete (Cook's theorem)
- Two classes of algorithms for finding out:
  - Enumeration of possible solutions (Truth tables etc.)
  - Deduction

# The satisfiability problem

- Given a formula  $\varphi$ , is  $\varphi$  satisfiable?

Enumeration the first:

```
Boolean SAT( $\varphi$ ) {  
    for all  $\alpha \in \text{Assign}$  对所有assignment的可能情况进行枚举  
        if Eval( $\alpha, \varphi$ ) return true;  
    return false;  
}
```

Enumeration the second:

Use substitution to eliminate all variables one by one:

$$\exists a. \varphi \quad \text{iff} \quad \varphi[0/a] \vee \varphi[1/a] \quad \text{对单一变量进行枚举(0/1)}$$

- Q: What is the difference?  
A: Branching on complete vs. partial assignments.

# Deduction

A (deductive) **proof system** consists of a set of **axioms** and **inference rules**.

- **Inference rules:**

Antecedents

Consequents

inference rules: 满足前提, 可以得到consequents

(rule name)

Meaning: If all antecedents hold then at least one of the consequents can be derived.

- **Axioms** are inference rules with no antecedents, e.g.,

$$\overline{a \rightarrow (b \rightarrow a)} \quad (H1)$$

Axioms: 无需前提, 总可以得到consequents

- **Examples:**

$$\frac{a \rightarrow b \quad b \rightarrow c}{a \rightarrow c} \quad (Trans)$$

$$\frac{a \rightarrow b \quad a}{b} \quad (M.P.)$$

- Let  $\mathcal{H}$  be a proof system.
- $\Gamma \vdash_{\mathcal{H}} \varphi$  means: There is a proof of  $\varphi$  in system  $\mathcal{H}$  whose premises are included in  $\Gamma$
- $\vdash_{\mathcal{H}}$  is called the provability (derivability) relation.

# Example

- Let  $\mathcal{H}$  be the proof system comprised of the rules **Trans** and **M.P.** that we saw earlier:

$$\frac{a \rightarrow b \quad b \rightarrow c}{a \rightarrow c} \quad (\text{Trans})$$

$$\frac{a \rightarrow b \quad a}{b} \quad (\text{M.P.})$$

- Does the following relation hold?

$$a \rightarrow b, b \rightarrow c, c \rightarrow d, d \rightarrow e, a \vdash_{\mathcal{H}} e$$

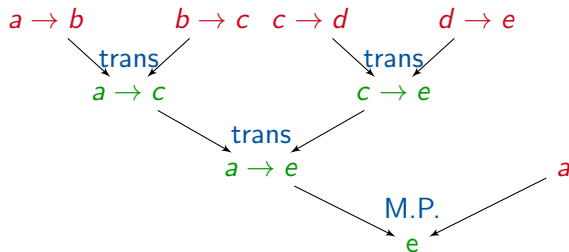
# Deductive proof: Example

$$\frac{a \rightarrow b \quad b \rightarrow c}{a \rightarrow c} \quad (Trans) \quad \frac{a \rightarrow b \quad a}{b} \quad (M.P.)$$

$a \rightarrow b, b \rightarrow c, c \rightarrow d, d \rightarrow e, a \vdash_{\mathcal{H}} e$

1. *premise* :  $a \rightarrow b$
2. *premise* :  $b \rightarrow c$
3. 1, 2, *Trans* :  $a \rightarrow c$
4. *premise* :  $c \rightarrow d$
5. *premise* :  $d \rightarrow e$
6. 4, 5, *Trans* :  $c \rightarrow e$
7. 3, 6, *Trans* :  $a \rightarrow e$
8. *premise* :  $a$
9. 7, 8, *M.P.* :  $e$

# Proof graph



# Soundness and completeness

- For a given proof system  $\mathcal{H}$ ,
  - **Soundness:** Does  $\vdash$  conclude “correct” conclusions from premises?
  - **Completeness:** Can we conclude all true statements with  $\mathcal{H}$ ?
- Correct with respect to what?

With respect to the semantic definition of the logic. In the case of propositional logic truth tables give us this.

*syntactic  $\rightarrow$  semantic*

**Soundness** of  $\mathcal{H}$ : if  $\Gamma \vdash_{\mathcal{H}} \varphi$  then  $\Gamma \models \varphi$

**Completeness** of  $\mathcal{H}$ : if  $\Gamma \models \varphi$  then  $\Gamma \vdash_{\mathcal{H}} \varphi$



## Example: Hilbert axiom system (H)

- Let H be (M.P.) together with the following axiom schemes:

$$\overline{a \rightarrow (b \rightarrow a)} \quad (H1)$$

$$\overline{((a \rightarrow (b \rightarrow c)) \rightarrow ((a \rightarrow b) \rightarrow (a \rightarrow c)))} \quad (H2)$$

$$\overline{(\neg b \rightarrow \neg a) \rightarrow (a \rightarrow b)} \quad (H3)$$

- H is **sound and complete** for propositional logic.

# Proof of soundness and completeness

- To prove **soundness** of  $H$ , prove the soundness of its axioms and inference rules (easy **with truth-tables**).

For example:

$a$	$b$	$a \rightarrow (b \rightarrow a)$
0	0	1
0	1	1
1	0	1
1	1	1

- Completeness: harder, but possible.

# The resolution proof system

- The resolution inference rule for CNF: 一次只能消去一个

$$\frac{(\cancel{\lambda} \vee \overset{\text{Positive}}{l_1} \vee l_2 \vee \dots \vee l_n) \quad (\neg \cancel{\lambda} \vee \overset{\text{negative}}{l'_1} \vee \dots \vee l'_m)}{(l_1 \vee \dots \vee l_n \vee l'_1 \vee \dots \vee l'_m)} \text{ Resolution}$$

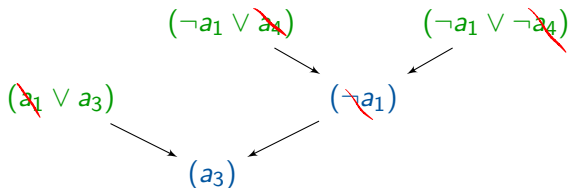
- Example:

$$\frac{(a \vee b) \wedge (\neg a \vee c)}{(b \vee c)}$$

- We first see some example proofs, before proving soundness and completeness.

# Proof by resolution

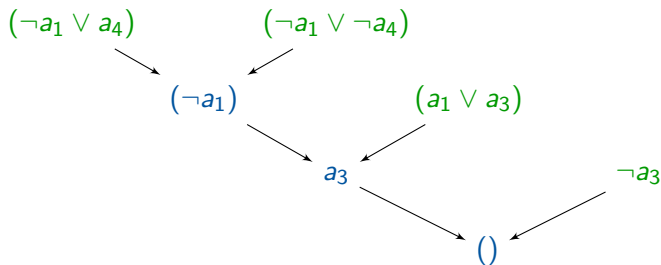
- Let  $\varphi = (a_1 \vee a_3) \wedge (\neg a_1 \vee a_2 \vee a_5) \wedge (\neg a_1 \vee a_4) \wedge (\neg a_1 \vee \neg a_4)$
- We want to prove  $\varphi \rightarrow (a_3)$



- Resolution is a sound and complete proof system for CNF.
- If the input formula is unsatisfiable, there exists a proof of the empty clause.

# Example

Let  $\varphi = (a_1 \vee a_3) \wedge (\neg a_1 \vee a_2) \wedge (\neg a_1 \vee a_4) \wedge (\neg a_1 \vee \neg a_4) \wedge (\neg a_3)$ .



# Soundness and completeness of resolution

- **Soundness** is straightforward. Just prove by truth table that

$$\models ((\varphi_1 \vee a) \wedge (\varphi_2 \vee \neg a)) \rightarrow (\varphi_1 \vee \varphi_2).$$

- **Completeness** is a bit more involved.

Basic idea: Use resolution for **variable elimination**.

$$\begin{aligned} & (a \vee \varphi_1) \wedge \dots \wedge (a \vee \varphi_n) \wedge \\ & (\neg a \vee \psi_1) \wedge \dots \wedge (\neg a \vee \psi_m) \wedge \\ & \quad R \\ & \Leftrightarrow \\ & (\varphi_1 \vee \psi_1) \wedge \dots \wedge (\varphi_1 \vee \psi_m) \wedge \\ & \quad \dots \\ & (\varphi_n \vee \psi_1) \wedge \dots \wedge (\varphi_n \vee \psi_m) \wedge \\ & \quad R \end{aligned}$$

where  $\varphi_i$  ( $i = 1, \dots, n$ ),  $\psi_j$  ( $j = 1, \dots, m$ ), and  $R$  contains neither  $a$  nor  $\neg a$ .

- How is negation normal form (NNF) defined and how can we convert logical formulas to NNF?
- How is disjunctive normal form (DNF) defined and how can we convert logical formulas to DNF?
- How is conjunctive normal form (CNF) defined and how can we convert logical formulas to CNF?
- What are the advantages and disadvantages of DNF and CNF?
  
- How can enumeration be used to solve the satisfiability problem?
- What is a deductive proof system and how can we derive proofs in it?
- When is a deductive proof system sound? When is it complete?
- What is resolution?



$A \vee B \vee \neg C \vee D$     $\neg B$     $A \vee \neg D$     $\neg A \vee \neg D$

$B \vee \neg C \vee D$     $\neg B$

$\neg C \vee D$

~~$\neg D$~~

$\neg D$

$\neg D$

true