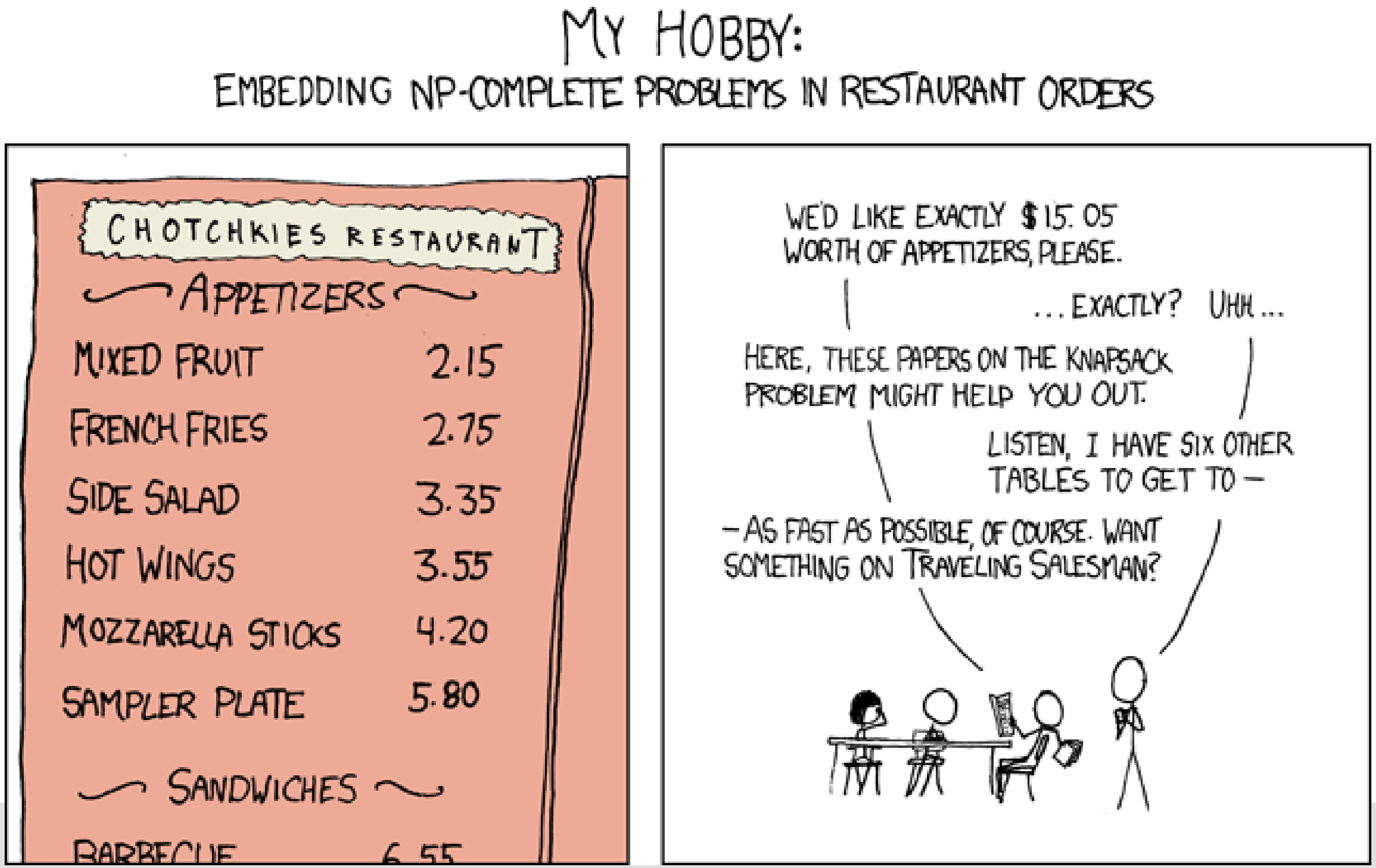
Logic and Computability

SCIENCE
PASSION
TECHNOLOGY

Combinational Equivalence Checking

Bettina Könighofer
bettina.koenighofer@iaik.tugraz.at

Stefan Pranger
stefan.pranger@iaik.tugraz.at



https://xkcd.com/287/

Recap - Topics we discussed so far

- Propositional Logic
  - Syntax and Semantics
- SAT Solving (DPLL)
  - (Efficiently) solve huge formulas
- BDDs
  - Data structure to efficiently store and manipulate formulas
- Natural Deduction
  - Prove that arguments in prop. logic are valid

Plan of Today

First Part – A few Basic Concepts of Propositional Logic

- Last lecture about propositional logic
  - Next week: predicate logic
- Several basic concepts
  - Relations between Satisfiability, Validity, and Equivalence
  - Normal Forms: CNF, DNF
  - Logical equivalences: Distributive laws, De Morgan's law…
- Tseitin Encoding
  - Computes equisatisfiable formula in CNF
- Equivalence checking via reduction to SAT

Second Part – Z3

- Introduction to SMT solver Z3
- Focus on solving formulas in propositional logic

Outline

- Algorithm - Decide equivalence of combinational circuits
  - Based on reduction to Satisfiability
- Relations between Satisfiability, Validity, and Equivalence
- Normal Forms
- Tseitin Encoding
  - Algorithm to translate formula in equisatisfiable formula in CNF

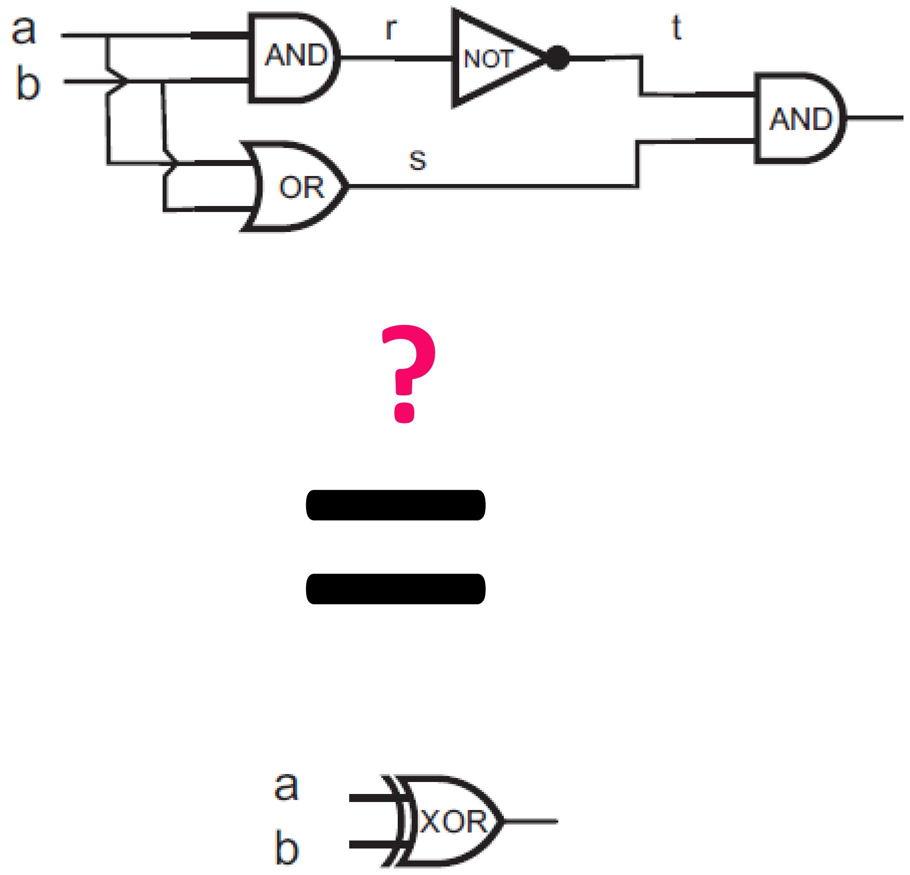
Learning Outcomes

After this lecture…

1. students can apply the algorithm to check for equivalence based on the reduction to SAT.
2. students can explain the relation between satisfiability, validity, and equivalence.
3. students can rewrite and simplify formulas by applying logical equivalences.
4. students can construct the CNF and DNF normal forms of formulas via truth tables.
5. students can apply Tseitin’s algorithm to construct formulas in CNF.
6. students can explain the concept of equisatisfiability.

Combinational Equivalence Checking

- Circuit Optimization and Synthesis Tools
  - Big Market
  - Tools can make mistakes!
  - Need to check for equivalence



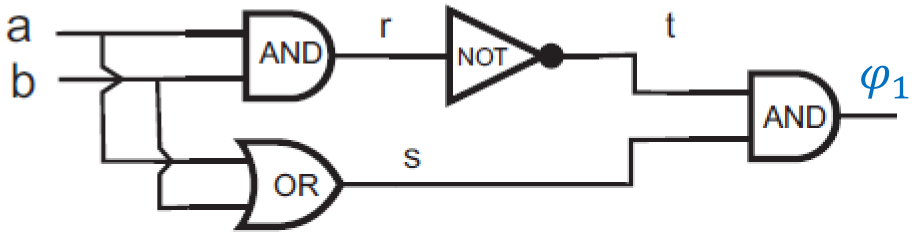
Algorithm - Circuit Equivalence via Truth Tables

- Using Truth Tables: Check for $\phi \vDash \psi$ and $\psi \vDash \phi$?
  i. e., ϕ and ψ are true for the same models
  - Exponentially large
  - → Not practicable!

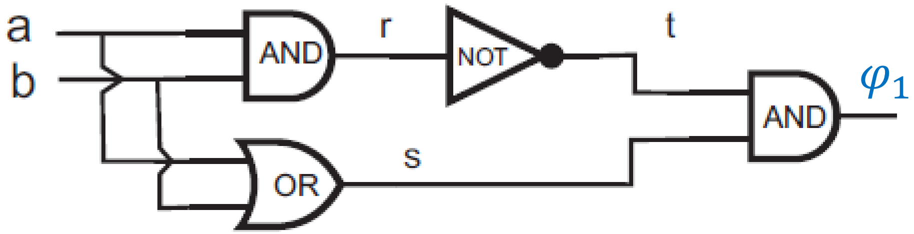
- Better way: Reduction to SAT

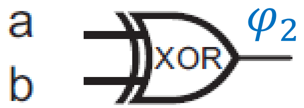
Algorithm - Circuit Equivalence based on SAT

Step 1: Encode C_1 and C_2 into formulas:



Algorithm - Circuit Equivalence based on SAT



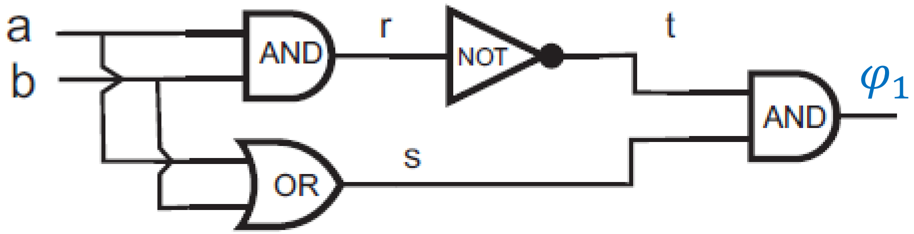


Step 1: Encode C_1 and C_2 into formulas:

$$\begin{aligned}\varphi_1 &= t \wedge s \\ &= \neg r \wedge (a \vee b) \\ &= \neg(\boldsymbol{a} \wedge \boldsymbol{b}) \wedge (\boldsymbol{a} \vee \boldsymbol{b})\end{aligned}$$

$$\varphi_2 = (\boldsymbol{a} \wedge \neg \boldsymbol{b}) \vee (\neg \boldsymbol{a} \wedge \boldsymbol{b})$$

Algorithm - Circuit Equivalence based on SAT



Step 1: Encode C_1 and C_2 into formulas:

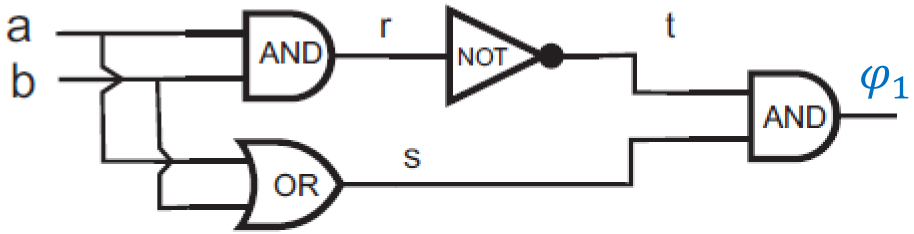
$$\varphi_1 = \neg(\boldsymbol{a} \wedge \boldsymbol{b}) \wedge (\boldsymbol{a} \vee \boldsymbol{b})$$

$$\varphi_2 = (\boldsymbol{a} \wedge \neg \boldsymbol{b}) \vee (\neg \boldsymbol{a} \wedge \boldsymbol{b})$$

Circuits are **equivalent** $\Leftrightarrow$ $\varphi_1 \oplus \varphi_2$ is **unsatisfiable**.

Algorithm - Circuit Equivalence based on SAT

Step 1: Encode C_1 and C_2 into formulas:



$$\varphi_1 = \neg(\boldsymbol{a} \wedge \boldsymbol{b}) \wedge (\boldsymbol{a} \vee \boldsymbol{b})$$

$$\varphi_2 = (\boldsymbol{a} \wedge \neg\boldsymbol{b}) \vee (\neg\boldsymbol{a} \wedge \boldsymbol{b})$$

Circuits are **equivalent** $\Leftrightarrow$ $\underbrace{\varphi_1 \oplus \varphi_2}$ is **unsatisfiable**.

SAT Solver needs CNF

Use **Tseitin** Encoding to encode $\varphi_1 \oplus \varphi_2$ as CNF

Algorithm - Circuit Equivalence based on SAT

1. Encode C_1 and C_2 into two formulas φ_1 and φ_2

2. Compute the Conjunctive Normal Form (CNF) of $\varphi_1 \oplus \varphi_2$
   - Use Tseitin Encoding

3. Give CNF($\varphi_1 \oplus \varphi_2$) to a **SAT solver**

4. C_1 and C_2 are **equivalent** if and only if $\varphi_1 \oplus \varphi_2$ is **UNSAT**

Outline

- Algorithm - Decide equivalence of combinational circuits
  - Based on reduction to Satisfiability
- Relations between Satisfiability, Validity, and Equivalence
- Normal Forms
- Tseitin Encoding

Circuits are **equivalent** $\Leftrightarrow$ $\underbrace{\varphi_1 \oplus \varphi_2}_{\text{Convert to CNF using \textbf{Tseitin} Encoding}}$ is **unsatisfiable**.

Duality: Validity and Satisfiability

- **ϕ is valid $\Leftrightarrow$ $\neg\phi$ is not satisfiable**
  ϕ is satisfiable $\Leftrightarrow$ $\neg\phi$ is not valid

- Example:
  - $\phi = (x \lor \neg x)$ is valid. Truth Table: All rows **T**.
  - $\neg\phi = \neg(x \lor \neg x) \equiv \neg x \land x$ is not satisfiable. Truth Table: All rows **F**.

- Only one decision procedure needed

Reductions

- Only one decision procedure needed

Solve / using	ϕ satisfiable?	ϕ valid?	$\phi \equiv \psi$?
Satisfiability	✓	$\neg\phi$ not satisfiable?	$\phi \oplus \psi$ not satisfiable?
Validity	$\neg\phi$ not valid?	✓	$\phi \leftrightarrow \psi$ valid?
Equivalence	$\phi \not\equiv \bot$?	$\phi \equiv \top$?	✓

Outline

- Algorithm - Decide equivalence of combinational circuits
  - Based on reduction to Satisfiability
- Relations between Satisfiability, Validity, and Equivalence
- Normal Forms
- Tseitin Encoding

Circuits are **equivalent** $\Leftrightarrow$ $\underbrace{\varphi_1 \oplus \varphi_2}_{\text{Convert to CNF using } \textbf{Tseitin} \text{ Encoding}}$ is **unsatisfiable**.

Normal Forms

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

- **Disjunctive Normal Form (DNF)**
  - Disjunction of conjunction of literals:

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

where each a_i, b_j is a literal

- **Conjunctive Normal Form (CNF)**
  - Conjunction of disjunctions of literals:

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

where each a_i, b_j is a literal

Ways to Obtain a CNF

- SAT Solvers require formula in CNF as input
- Obtain CNF via Truth Table
  - Exponential size
- Obtain CNF via logical equivalences (De Morgan’s laws, Distributive laws…)
  - Exponential size
- **Tseitin Encoding**
  - Use auxiliary variables
  - Linear blow-up
  - Produces **equisatisfiable formula** with **linear blowup**

DNF from Truth Table

Example:

p	q	r	$(r \vee q) \to (p \wedge \neg q)$
0	**0**	**0**	**1**
0	0	1	0
0	1	0	0
0	1	1	0
1	**0**	**0**	**1**
1	**0**	**1**	**1**
1	1	0	0
1	1	1	0

DNF from Truth Table

Enumerate satisfying models,
connect satisfying models with disjunctions.

Example:

p	q	r	$(r \vee q) \rightarrow (p \wedge \neg q)$
0	0	0	1
0	0	1	0
0	1	0	0
0	1	1	0
1	0	0	1
1	0	1	1
1	1	0	0
1	1	1	0

$\neg p \wedge \neg q \wedge \neg r$

$p \wedge \neg q \wedge \neg r$

$p \wedge \neg q \wedge r$

DNF: $(\neg p \wedge \neg q \wedge \neg r) \vee (p \wedge \neg q \wedge \neg r) \vee (p \wedge \neg q \wedge r)$

CNF from Truth Table

Example:

p	q	r	$(p \vee \neg q) \to r$
0	**0**	**0**	**0**
0	0	1	1
0	1	0	1
0	1	1	1
1	**0**	**0**	**0**
1	0	1	1
1	**1**	**0**	**0**
1	1	1	1

CNF from Truth Table

Exclude falsifying models by requiring that at least one literal per falsifying model must be different, connect with conjunctions.

Example:

p	q	r	$(p \lor \neg q) \rightarrow r$
0	0	0	0
0	0	1	1
0	1	0	1
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	0
1	1	1	1

$p \lor q \lor r$

$\neg p \lor q \lor r$

$\neg p \lor \neg q \lor r$

CNF: $(p \lor q \lor r) \land (\neg p \lor q \lor r) \land (\neg p \lor \neg q \lor r)$

Outline

- Algorithm - Decide equivalence of combinational circuits
  - Based on reduction to Satisfiability
- Relations between Satisfiability, Validity, and Equivalence
- Normal Forms
- Tseitin Encoding

Circuits are **equivalent** $\Leftrightarrow$ $\underbrace{\varphi_1 \oplus \varphi_2}_{\text{Convert to CNF using } \textbf{Tseitin} \text{ Encoding}}$ is **unsatisfiable**.

Tseitin Encoding

- Produces **equisatisfiable formula** in CNF with **linear blowup**
- Trick: Use auxiliary variables
- **Definition of equisatisfiability:**

ϕ and ψ are **equisatisfiable** $\Leftrightarrow$ either both are satisfiable, or both are unsatisfiable

- For equivalence checking, we only need the info SAT or UNSAT

Tseitin Encoding

- Step 1
  - Assign new variables to each sub-formula
- Step 2
  - Add explanation for each new variable
- Step 3
  - Apply Tseitin Rewrite Rules to obtain equisatisfiable CNF

$$\begin{aligned}
\chi \leftrightarrow (\varphi \vee \psi) &\quad\Leftrightarrow\quad (\neg\varphi \vee \chi) \wedge (\neg\psi \vee \chi) \wedge (\neg\chi \vee \varphi \vee \psi) \\
\chi \leftrightarrow (\varphi \wedge \psi) &\quad\Leftrightarrow\quad (\neg\chi \vee \varphi) \wedge (\neg\chi \vee \psi) \wedge (\neg\varphi \vee \neg\psi \vee \chi) \\
\chi \leftrightarrow \neg\varphi &\quad\Leftrightarrow\quad (\neg\chi \vee \neg\varphi) \wedge (\varphi \vee \chi)
\end{aligned}$$

Example – Tseitin Encoding

Use Tseitin encoding to compute the CNF of $\varphi = ((p \vee q) \wedge r) \vee \neg p$.

Rewrite Rules

$\chi \leftrightarrow (\varphi \vee \psi)$	$\Leftrightarrow$	$(\neg\varphi \vee \chi) \wedge (\neg\psi \vee \chi) \wedge (\neg\chi \vee \varphi \vee \psi)$
$\chi \leftrightarrow (\varphi \wedge \psi)$	$\Leftrightarrow$	$(\neg\chi \vee \varphi) \wedge (\neg\chi \vee \psi) \wedge (\neg\varphi \vee \neg\psi \vee \chi)$
$\chi \leftrightarrow \neg\varphi$	$\Leftrightarrow$	$(\neg\chi \vee \neg\varphi) \wedge (\varphi \vee \chi)$

Example – Tseitin Encoding

Use Tseitin encoding to compute the CNF of $\varphi = ((p \lor q) \land r) \lor \neg p$.

Rewrite Rules

$$\begin{aligned}
\chi \leftrightarrow (\varphi \lor \psi) &\quad\Leftrightarrow\quad (\neg\varphi \lor \chi) \land (\neg\psi \lor \chi) \land (\neg\chi \lor \varphi \lor \psi) \\
\chi \leftrightarrow (\varphi \land \psi) &\quad\Leftrightarrow\quad (\neg\chi \lor \varphi) \land (\neg\chi \lor \psi) \land (\neg\varphi \lor \neg\psi \lor \chi) \\
\chi \leftrightarrow \neg\varphi &\quad\Leftrightarrow\quad (\neg\chi \lor \neg\varphi) \land (\varphi \lor \chi)
\end{aligned}$$

$$\underbrace{(\underbrace{(\underbrace{p \lor q}_{x_1}) \land r}_{x_2}) \lor \underbrace{\neg p}_{x_3}}_{x_\varphi}$$

$$\begin{aligned}
CNF(\varphi) = \;& (\neg p \lor x_1) \land (\neg q \lor x_1) \land (\neg x_1 \lor p \lor q) \\
& \land (\neg x_2 \lor x_1) \land (\neg x_2 \lor r) \land (\neg x_1 \lor \neg r \lor x_2) \\
& \land (\neg x_3 \lor \neg p) \land (p \lor x_3) \\
& \land (\neg x_2 \lor x_\varphi) \land (\neg x_3 \lor x_\varphi) \land (\neg x_\varphi \lor x_2 \lor x_3) \\
& \land x_\varphi
\end{aligned}$$

Example – Tseitin Encoding

Use Tseitin encoding to compute the CNF of $\varphi = \neg(a \vee \neg b) \vee (\neg a \wedge c)$.

Rewrite Rules

$$
\begin{aligned}
\chi \leftrightarrow (\varphi \vee \psi) &\quad\Leftrightarrow\quad (\neg\varphi \vee \chi) \wedge (\neg\psi \vee \chi) \wedge (\neg\chi \vee \varphi \vee \psi) \\
\chi \leftrightarrow (\varphi \wedge \psi) &\quad\Leftrightarrow\quad (\neg\chi \vee \varphi) \wedge (\neg\chi \vee \psi) \wedge (\neg\varphi \vee \neg\psi \vee \chi) \\
\chi \leftrightarrow \neg\varphi &\quad\Leftrightarrow\quad (\neg\chi \vee \neg\varphi) \wedge (\varphi \vee \chi)
\end{aligned}
$$

Example – Tseitin Encoding

Use Tseitin encoding to compute the CNF of $\varphi = \neg(a \vee \neg b) \vee \ (\neg a \wedge c)$.

$$\underbrace{\underbrace{\neg\underbrace{(a \vee \underbrace{\neg b}_{x_4})}_{x_3}}_{x_1} \vee \underbrace{(\underbrace{\neg a}_{x_5} \wedge c)}_{x_2}}_{x_\varphi}$$

$$\begin{array}{rcl}
\chi \leftrightarrow (\varphi \vee \psi) & \Leftrightarrow & (\neg\varphi \vee \chi) \wedge (\neg\psi \vee \chi) \wedge (\neg\chi \vee \varphi \vee \psi) \\
\chi \leftrightarrow (\varphi \wedge \psi) & \Leftrightarrow & (\neg\chi \vee \varphi) \wedge (\neg\chi \vee \psi) \wedge (\neg\varphi \vee \neg\psi \vee \chi) \\
\chi \leftrightarrow \neg\varphi & \Leftrightarrow & (\neg\chi \vee \neg\varphi) \wedge (\varphi \vee \chi)
\end{array}$$

$$\begin{aligned}
CNF(\varphi) = x_\varphi \wedge & \\
& (\neg x_1 \vee x_\varphi) \wedge (\neg x_2 \vee x_\varphi) \wedge (\neg x_\varphi \vee x_1 \vee x_2) \wedge \\
& (\neg x_1 \vee \neg x_3) \wedge (x_1 \vee x_3) \wedge \\
& (\neg a \vee x_3) \wedge (\neg x_4 \vee x_3) \wedge (\neg x_3 \vee a \vee x_4) \wedge \\
& (\neg x_2 \vee x_5) \wedge (\neg x_2 \vee c) \wedge (\neg x_5 \vee \neg c \vee x_2) \wedge \\
& (\neg x_4 \vee \neg b) \wedge (x_4 \vee b) \wedge \\
& (\neg x_5 \vee \neg a) \wedge (x_5 \vee a)
\end{aligned}$$

Derive Rewrite Rules

- $r \leftrightarrow (p \wedge q)$ … rewrite it to a CNF

$$a \rightarrow b \equiv \neg a \vee b$$

De-Morgan

$$\neg(a \wedge b) \equiv \neg a \vee \neg b$$
$$\neg(a \vee b) \equiv \neg a \wedge \neg b$$

Distributive Law

$$a \vee (b \wedge c) \equiv (a \vee b) \wedge (a \vee c)$$
$$a \wedge (b \vee c) \equiv (a \wedge b) \vee (a \wedge c)$$

Derive Rewrite Rules

- $r \leftrightarrow (p \wedge q)$ … rewrite it to a CNF
- $(r \rightarrow p \wedge q) \wedge (p \wedge q \rightarrow r)$
- $(\neg r \vee (p \wedge q)) \wedge (\neg(p \wedge q) \vee r)$
- $(\neg r \vee p) \wedge (\neg r \vee q) \wedge (\neg p \vee \neg q \vee r)$

$$a \rightarrow b \equiv \neg a \vee b$$

De-Morgan

$$\neg(a \wedge b) \equiv \neg a \vee \neg b$$
$$\neg(a \vee b) \equiv \neg a \wedge \neg b$$

Distributive Law

$$a \vee (b \wedge c) \equiv (a \vee b) \wedge (a \vee c)$$
$$a \wedge (b \vee c) \equiv (a \wedge b) \vee (a \wedge c)$$

Derive Rewrite Rules

- $r \leftrightarrow (p \vee q)$ … rewrite it to a CNF

$$a \rightarrow b \equiv \neg a \vee b$$

De-Morgan

$$\neg(a \wedge b) \equiv \neg a \vee \neg b$$
$$\neg(a \vee b) \equiv \neg a \wedge \neg b$$

Distributive Law

$$a \vee (b \wedge c) \equiv (a \vee b) \wedge (a \vee c)$$
$$a \wedge (b \vee c) \equiv (a \wedge b) \vee (a \wedge c)$$

Derive Rewrite Rules

- $r \leftrightarrow (p \vee q)$ … rewrite it to a CNF
- $((p \vee q) \rightarrow r) \wedge (r \rightarrow p \vee q)$
- $(\neg(p \vee q) \vee r) \wedge (\neg r \vee p \vee q)$
- $((\neg p \wedge \neg q) \vee r) \wedge (\neg r \vee p \vee q)$
- $(\neg p \vee r) \wedge (\neg q \vee r) \wedge (\neg r \vee p \vee q)$

$$a \rightarrow b \equiv \neg a \vee b$$

De-Morgan

$$\neg(a \wedge b) \equiv \neg a \vee \neg b$$
$$\neg(a \vee b) \equiv \neg a \wedge \neg b$$

Distributive Law

$$a \vee (b \wedge c) \equiv (a \vee b) \wedge (a \vee c)$$
$$a \wedge (b \vee c) \equiv (a \wedge b) \vee (a \wedge c)$$

Example

Derive the rewrite rule for $x \leftrightarrow (p \rightarrow q)$.

$a \rightarrow b \equiv \neg a \vee b$

De-Morgan

$\neg(a \wedge b) \equiv \neg a \vee \neg b$
$\neg(a \vee b) \equiv \neg a \wedge \neg b$

Distributive Law

$a \vee (b \wedge c) \equiv (a \vee b) \wedge (a \vee c)$
$a \wedge (b \vee c) \equiv (a \wedge b) \vee (a \wedge c)$

Example

Derive the rewrite rule for $x \leftrightarrow (p \rightarrow q)$.

$$
\begin{aligned}
x \leftrightarrow (p \rightarrow q) &\Leftrightarrow x \leftrightarrow (p \rightarrow q) \\
&\Leftrightarrow (x \rightarrow (p \rightarrow q)) \wedge ((p \rightarrow q) \rightarrow x) \\
&\Leftrightarrow (x \rightarrow (\neg p \vee q)) \wedge ((\neg p \vee q) \rightarrow x) \\
&\Leftrightarrow (\neg x \vee (\neg p \vee q)) \wedge (\neg(\neg p \vee q) \vee x) \\
&\Leftrightarrow (\neg x \vee \neg p \vee q) \wedge ((\neg\neg p \wedge \neg q) \vee x) \\
&\Leftrightarrow (\neg x \vee \neg p \vee q) \wedge ((p \wedge \neg q) \vee x) \\
&\Leftrightarrow (\neg x \vee \neg p \vee q) \wedge ((p \vee x) \wedge (\neg q \vee x)) \\
&\Leftrightarrow (\neg x \vee \neg p \vee q) \wedge (p \vee x) \wedge (\neg q \vee x)
\end{aligned}
$$

$a \rightarrow b \equiv \neg a \vee b$

De-Morgan

$\neg(a \wedge b) \equiv \neg a \vee \neg b$
$\neg(a \vee b) \equiv \neg a \wedge \neg b$

Distributive Law

$a \vee (b \wedge c) \equiv (a \vee b) \wedge (a \vee c)$
$a \wedge (b \vee c) \equiv (a \wedge b) \vee (a \wedge c)$

CEC Example

Check whether $\varphi_1 = a \wedge \neg b$ and $\varphi_2 = \neg(\neg a \vee b)$ are equivalent using the reduction to SAT.

CEC Example

Check whether $\varphi_1 = a \land \neg b$ and $\varphi_2 = \neg(\neg a \lor b)$ are equivalent using the reduction to SAT.

Step 1) Build $\varphi = \varphi_1 \oplus \varphi_2$

$$\begin{aligned}\varphi &= \varphi_1 \oplus \varphi_2 \\ &= [\varphi_1 \lor \varphi_2] \land \neg[\varphi_1 \land \varphi_2] = \\ &= [(a \land \neg b) \lor (\neg(\neg a \lor b))] \land \neg[(a \land \neg b) \land (\neg(\neg a \lor b))]\end{aligned}$$

Step 2) Compute CNF of φ via Tseitin

$$\underbrace{\underbrace{\Big[\underbrace{(a \land \underbrace{\neg b}_{x_7})}_{x_3} \lor \underbrace{(\neg \underbrace{(\underbrace{\neg a}_{x_8} \lor b)}_{x_6})}_{x_4}\Big]}_{x_1} \land \underbrace{\neg \underbrace{\Big[\underbrace{(a \land \underbrace{\neg b}_{x_7})}_{x_3} \lor \underbrace{(\neg \underbrace{(\underbrace{\neg a}_{x_8} \lor b)}_{x_6})}_{x_4}\Big]}_{x_1}}_{x_2}}_{x_\varphi}$$

$$\begin{aligned}CNF(\varphi) = x_\varphi \land \; & (\neg x_\varphi \lor x_1) \land (\neg x_\varphi \lor x_2) \land (\neg x_1 \lor \neg x_2 \lor x_\varphi) \land \\ & (\neg x_1 \lor \neg x_2) \land (x_1 \lor x_2) \land \\ & (\neg x_3 \lor x_1) \land (\neg x_4 \lor x_1) \land (\neg x_1 \lor x_3 \lor x_4) \land \\ & (\neg x_3 \lor a) \land (\neg x_3 \lor x_7) \land (\neg a \lor \neg x_7 \lor x_3) \land \\ & (\neg x_4 \lor \neg x_6) \land (x_4 \lor x_6) \land \\ & (\neg x_8 \lor x_6) \land (\neg b \lor x_6) \land (\neg x_6 \lor x_8 \lor b) \land \\ & (\neg x_7 \lor \neg b) \land (x_7 \lor b) \land \\ & (\neg x_8 \lor \neg a) \land (x_8 \lor a)\end{aligned}$$

Step 3) Check via SAT Solver: Is the CNF of φ satisfiable?

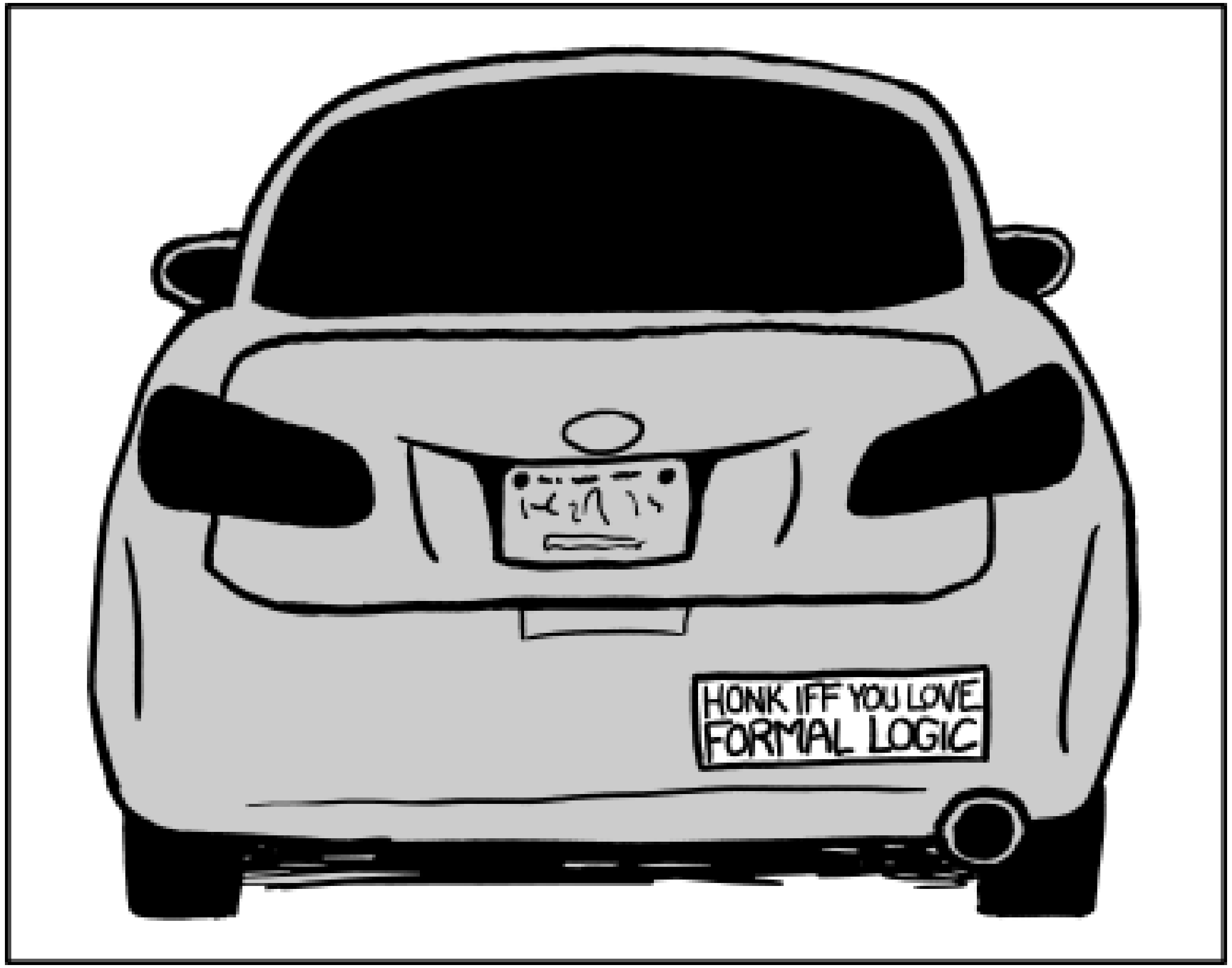
Step 4) Interpret result: φ_1 and φ_2 are **equivalent** if and only if $\varphi_1 \oplus \varphi_2$ is **UNSAT**

Learning Outcomes

After this lecture…

1. students can apply the algorithm to check for equivalence based on the reduction to SAT.
2. students can explain the relation between satisfiability, validity, and equivalence.
3. students can rewrite and simplify formulas by applying logical equivalences.
4. students can construct the CNF and DNF normal forms of formulas via truth tables.
5. students can apply Tseitin's algorithm to construct formulas in CNF.
6. students can explain the concept of equisatisfiability.

Thank You



https://xkcd.com/1033/