Questionnaire “Logic and Computability”

Summer Term 2024

Contents

5 Combinational Equivalence Checking

5.1 Normal Forms

5.1.1 Define the *Disjunctive Normal Form (DNF)* of formulas in propositional logic. Use the proper terminology and give an example.

5.1.2 Define the *Conjunctive Normal Form (CNF)* of formulas in propositional logic. Use the proper terminology and give an example.

5.1.3 Given a formula in propositional logic. Explain how to extract a *CNF* representation as well as a *DNF* representation of φ using the truth table from φ.

5.1.4 Given the formula $\varphi = (q \to p) \wedge (r \vee \neg p)$. Compute its representation in Disjunctive Normal Form (*DNF*) using a truth table.

5.1.5 Given the formula $\varphi = (q \to p) \wedge (r \vee \neg p)$. Compute its representation in Conjunctive Normal Form (*CNF*) using a truth table.

5.1.6 Given the formula $\varphi = (a \wedge \neg b \wedge \neg c) \vee ((\neg c \to a) \to b)$. Use the truth table of φ to compute its representation in (a) CNF and (b) DNF.

5.1.7 Given the formula $\varphi = (q \to \neg r) \wedge \neg(p \vee q \vee \neg r)$. Use the truth table of φ to compute its representation in (a) CNF and (b) DNF.

5.1.8 Given the formula $\varphi = \neg(a \to \neg b) \vee (\neg a \to c)$. Use the truth table of φ to compute its representation in (a) CNF and (b) DNF.

5.1.9 Consider the propositional formula $\varphi = (\neg(\neg a \wedge b) \wedge \neg c)$. Fill out the truth table for φ and its subformulas. Compute a CNF as well as a DNF for φ from the truth table.

a	b	c	$\neg a$	$\neg a \wedge b$	$\neg(\neg a \wedge b)$	$\neg c$	$\varphi = (\neg(\neg a \wedge b) \wedge \neg c)$
F	**F**	**F**					
F	**F**	**T**					
F	**T**	**F**					
F	**T**	**T**					
T	**F**	**F**					
T	**F**	**T**					
T	**T**	**F**					
T	**T**	**T**					

5.1.10 Consider the propositional formula $\varphi = (p \vee \neg q) \to (\neg p \wedge \neg r)$. Fill out the truth table for φ and its subformulas. Compute a CNF as well as a DNF for φ from the truth table.

p	q	r	$\neg q$	$p \vee \neg q$	$\neg p$	$\neg r$	$\neg p \wedge \neg r$	$\varphi = (p \vee \neg q) \to (\neg p \wedge \neg r)$
F	**F**	**F**						
F	**F**	**T**						
F	**T**	**F**						
F	**T**	**T**						
T	**F**	**F**						
T	**F**	**T**						
T	**T**	**F**						
T	**T**	**T**						

5.1.11 Look a the following statements and tick all items that conform to a *DNF*.

- ☐ $a \lor b$
- ☐ A DNF is a conjunction of clauses.
- ☐ $(a \lor b) \land (\neg b \lor \neg a \lor c) \land \neg c$
- ☐ $(a \land b) \lor (\neg b \land \neg a \land c) \lor \neg c$
- ☐ A DNF is a conjunction of disjunctions of literals.
- ☐ b
- ☐ $a \land b \land \neg c$
- ☐ $(\neg a \land b) \land (\neg a \land c)$
- ☐ A DNF is a disjunction of cubes.
- ☐ $\neg(a \land \neg b) \land c$
- ☐ A DNF is a disjunction of conjunctions of literals.
- ☐ $a \land \neg b$

5.1.12 In the following list, tick all items that conform to the Conjunctive Normal Form (CNF).

- ☐ $(a \land b \land \neg c) \lor (\neg b \land \neg c) \lor (e \land \neg f)$
- ☐ a
- ☐ $\neg b$
- ☐ $a \land \neg b$
- ☐ $a \lor \neg b$
- ☐ $a \lor (\neg b \land c)$
- ☐ $(a \lor \neg b) \land c$
- ☐ $\neg(p \lor q)$
- ☐ $x \lor \neg y \lor z$

5.1.13 In the following list, tick all items that conform to the Disjunctive Normal Form (DNF).

- ☐ $(a \land b \land \neg c) \lor (\neg b \land \neg c) \lor (e \land \neg f)$
- ☐ $(a \lor b \lor \neg c) \land (\neg b \lor \neg c) \land (e \lor \neg f)$
- ☐ $\neg b$
- ☐ $a \land \neg b$
- ☐ $a \lor \neg b$
- ☐ $a \lor (\neg b \land c)$
- ☐ $(a \lor \neg b) \land c$
- ☐ $\neg(p \lor q)$
- ☐ $x \lor \neg y \lor z$

5.2 Relations between Satisfiability, Validity, Equivalence and Entailment

5.2.1 Explain the duality of *satisfiability* and *validity*.

5.2.2 How can you check whether it is true that $\varphi \models \psi$ using a decision procedure for (a) *satisfiability* or (b) *validity*?

5.2.3 *A formula φ is valid, if and only if $\neg\varphi$ is not satisfiable.* Explain why this statement holds true.

5.2.4 Given two propositional logic formulas φ and ψ. How can we check whether $\varphi \equiv \psi$ using a decision procedure for (a) satisfiability, (b) for validity, and (c) for semantic entailment?

5.2.5 Given a propositional logic formula φ. How can we check whether φ is *valid* using a decision procedure for (a) satisfiability and (b) equivalence?

5.2.6 Given a propositional logic formula φ. Tick all statements that are true.

- ☐ A formula φ is *valid*, if and only if $\neg\varphi$ is *satisfiable*.
- ☐ A formula ψ is *satisfiable*, if and only if $\neg\varphi$ is *valid*.
- ☐ A formula φ is *satisfiable*, if and only if $\neg\varphi$ is *not valid*.
- ☐ A formula φ is *valid*, if and only if $\neg\varphi$ is *not satisfiable*.

5.2.7 Given two propositional logic formulas φ and ψ. Tick all statements that are true.

- ☐ If $\neg\varphi$ is not satisfiable, φ is not valid.
- ☐ If $\top \models \varphi$, φ is valid.
- ☐ If $\varphi \leftrightarrow \psi$ is valid, φ entails ψ.
- ☐ If $\varphi \rightarrow \psi$ is valid, both formulas are equivalent.

5.2.8 Given two propositional logic formulas φ and ψ. Tick all statements that are true.

- ☐ If $\varphi \wedge \neg\psi$ is not satisfiable, φ entails ψ.
- ☐ If $\neg\varphi$ is not valid, φ is satisfiable.
- ☐ If φ entails ψ and ψ entails φ, both formulas are equivalent.
- ☐ If φ is equivalent to $\top$, φ is valid..

5.3 Combinational Equivalence Checking

5.3.1 Explain the algorithm used to decide the equivalence of combinational circuits via the reduction to satisfiability.

5.3.2 Give the definition of equisatisfiability.

5.3.3 Given a propositional logic formula φ, the Tseitin transformation computes an equisatisfiable formula φ' in CNF. Why is this enough for equivalence checking?

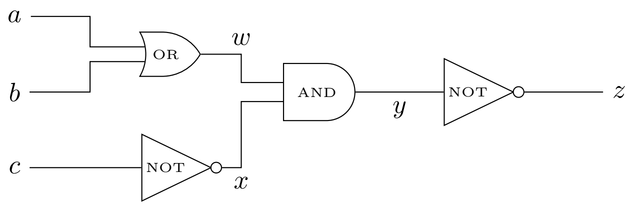
5.3.4 (a) What does it mean that two formulas φ and ψ are *equisatisfiable*? (b) Explain the difference between *satisfiability* and *equisatisfiability*.

5.3.5 Explain the algorithm of *Tseitin transformation* to obtain an equisatisfiable formula in CNF. Give step-by-step instructions of how to apply Tseitin transformation to a propositional formula.
(Note: Focus on the concept. You do *not* need to quote the rewrite rules!)

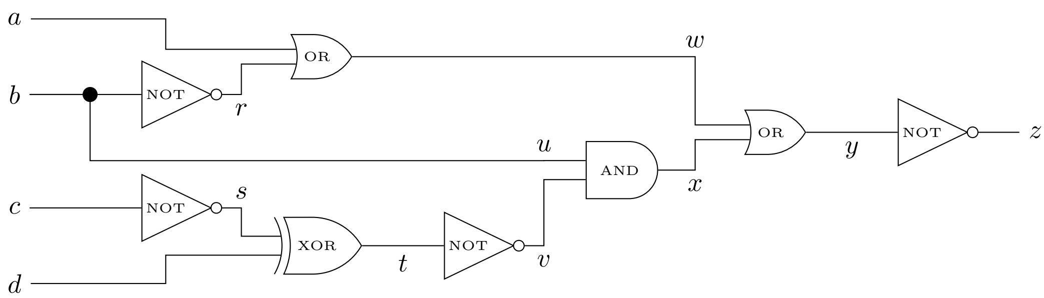
5.3.6 What is the advantage of applying *Tseitin transformation* to obtain an equisatisfiable CNF, especially compared to using truth tables?

5.3.7 Derive a Rewrite-Rule for an implication node, i.e., what clauses are introduced by the node $x \leftrightarrow (p \to q)$?

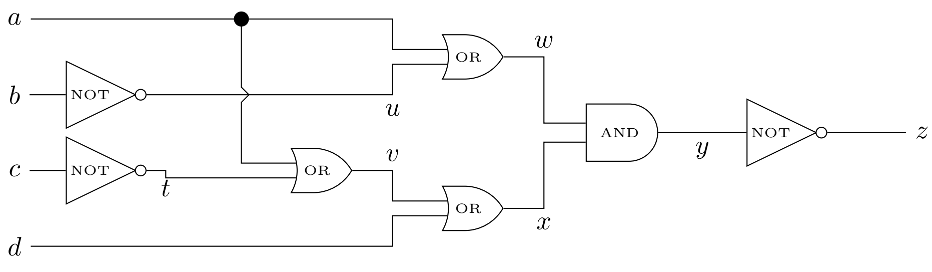
5.3.8 Compute the propositional formula φ represented by the following circuit. Furthermore, compute an equisatisfiable formula φ' using the Tseitin transformation.



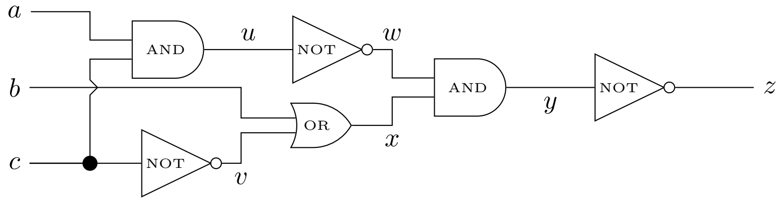
5.3.9 Compute the propositional formula φ represented by the following circuit. Furthermore, compute an equisatisfiable formula φ' using the Tseitin transformation.



5.3.10 Compute the propositional formula φ represented by the following circuit. Furthermore, compute an equisatisfiable formula φ' using the Tseitin transformation.



5.3.11 Compute the propositional formula φ represented by the following circuit. Furthermore, compute an equisatisfiable formula φ' using the Tseitin transformation.



We list the *Tseitin-rewriting rules* to be applied for the following examples.

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

5.3.12 Apply the Tseitin transformation to $\varphi = \neg(a \vee \neg b) \vee (\neg a \wedge c)$. For each variable you introduce, clearly indicate which subformula it represents.

5.3.13 Apply the Tseitin transformation to $\varphi = ((p \vee q) \wedge r) \vee \neg p$. For each variable you introduce, clearly indicate which subformula it represents.

5.3.14 Apply the Tseitin transformation to $\varphi = \neg(\neg b \wedge \neg c) \vee (\neg c \wedge a)$. For each variable you introduce, clearly indicate which subformula it represents.

5.3.15 Apply the Tseitin transformation to $\varphi = (q \wedge \neg r) \vee \neg(q \wedge \neg r)$. For each variable you introduce, clearly indicate which subformula it represents.

5.3.16 Apply the Tseitin transformation to $\varphi = (\neg(\neg a \wedge b) \wedge \neg c)$. For each variable you introduce, clearly indicate which subformula it represents.

5.3.17 Apply the Tseitin transformation to $\varphi = (p \vee \neg q) \vee (\neg p \wedge \neg r)$. For each variable you introduce, clearly indicate which subformula it represents.

5.3.18 Apply the Tseitin transformation to $\varphi = \neg(p \rightarrow q) \wedge (r \wedge p)$. For each variable you introduce, clearly indicate which subformula it represents. Derive the Tseitin transformation rule for $\rightarrow$ or transform the input such that you can use the rules above.

5.3.19 Check whether $\varphi_1 = a \wedge \neg b$ and $\varphi_2 = \neg(\neg a \vee b)$ are semantically equivalent using the reduction to satisfiability. Follow the algorithm discussed in the lecture and state the final formula that is used as input for a SAT solver.

5.3.20 Check whether $\varphi_1 = (a \wedge b) \vee \neg c$ and $\varphi_2 = (a \vee \neg c) \wedge (b \vee \neg c)$ are semantically equivalent using the reduction to satisfiability. Follow the algorithm discussed in the lecture and state the final formula that is used as input for a SAT solver.