Questionnaire "Logic and Computability"

Summer Term 2022

Contents

1 Propositional Logic

1.1 Lecture

1.1.1 Declarative Sentences

1. [Lecture] Look at the following statements and tick them if they are true.

   ☐ "*Give me the butter.*" is a declarative sentence.
   ☐ Questions are always declarative sentences.
   ☐ Declarative sentences can be true and false at the same time.
   ☐ "*My best friend is staying overnight.*" is a declarative sentence.

2. [Lecture] Model the following sentences as detailed as possible in propositional logic.

   (a) Alice will either take the bike or the tram to get to the concert, not both.
   (b) Students will have to take an exam at the end of the semester.
   (c) If he is hungry and the fridge is not empty, he cooks for himself.

3. [Lecture] Model the following sentences as detailed as possible in propositional logic.

   (a) If the air temperature is above 30°C, then the water temperature is above 20°C and I am able to go for a swim.
   (b) Your kid will be safe if and only if it learns to swim.
   (c) What time is it?

1.1.2 Syntax of Propositional Logic

4. [Lecture] Give the definition of well-formed formulas in propositional logic.

5. [Lecture] Let p, q and r be a atomic propositions. Tick all statements that are true.

   ☐ "$\neg p \wedge \vee\ q$" is a propositional formula.
   ☐ "$(p \wedge q) \vee (r \rightarrow p)$" is a propositional formula.
   ☐ "$\neg p$" is a propositional formula.
   ☐ "$\vee$" is a propositional formula.
   ☐ "p" is a propositional formula.

6. [Lecture] Determine whether the string $\neg(a \vee \neg\neg b)$ is a well-formed formula using the parse tree. Explain your answer.

7. [Lecture] Determine whether the string $\neg(a \vee \neg b\neg)$ is a well-formed formula using the parse tree. Explain your answer.

1.1.3 Semantics of Propositional Logic

8. [Lecture] What do we refer to if we talk about the *syntax* of propositional logic and what do we understand under the *semantics* of propositional logic. What is the difference between syntax and semantic?

9. [Lecture] Give the definition of a model $\mathcal{M}$ of a formula in propositional logic?

10. [Lecture] Consider the propositional formula $\varphi = \neg(\neg p \vee q) \rightarrow (p \wedge \neg r)$.

    Find a propositional formula ψ that is syntactically different from φ, but semantically equivalent to φ. Show the semantic equivalence of φ and ψ using truth tables.

11. [Lecture] Consider the propositional formula $\varphi = (p \wedge q) \rightarrow (q \vee \neg r)$. Fill out the truth table for φ and its subformulas.

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

12. [Lecture] Given is a formula $\varphi = (p \vee (\neg q \rightarrow r)) \wedge (\neg r \rightarrow p)$ and a model $\mathcal{M} = \{p = F,\ q = T,\ r = T\}$. Determine the truth value of φ for the given model $\mathcal{M}$ using its parse tree.

1.1.4 Semantic Entailment, Equivalence, Satisfiability and Validity

13. [Lecture] Give the definition of semantic entailment.

14. [Lecture] Give the definition of semantic equivalence.

15. [Lecture] Give the definition of validity.

16. [Lecture] Give the definition of satisfiability and unsatisfiability.

17. [Lecture] Consider a formula φ in propositional logic. Let the number of propositional variables in φ be n. How many lines does the truth table for φ have?

18. [Lecture] Consider a truth table for a propositional formula φ that has R rows. How many propositional variables does φ have?

19. [Lecture] Consider a formula φ in propositional logic. In the following list, tick all statements that are true.

    - ☐ If φ is not satisfiable, $\neg\varphi$ is valid.
    - ☐ If φ is valid, $\neg\varphi$ is not valid.
    - ☐ If φ is valid, $\neg\varphi$ is not satisfiable.

☐ If φ is not valid, $\neg\varphi$ is satisfiable.

20. [Lecture] Given are the truth tables for the propositional logic formulas φ and ψ. Determine whether it holds that $\varphi \models \psi$, $\psi \models \varphi$, or neither.

p	q	r	φ	ψ
F	**F**	**F**	F	F
F	**F**	**T**	T	T
F	**T**	**F**	F	F
F	**T**	**T**	T	T
T	**F**	**F**	F	F
T	**F**	**T**	F	T
T	**T**	**F**	T	T
T	**T**	**T**	T	T

21. [Lecture] Consider the propositional formulas $\varphi = (p \to q) \vee \neg r$ and $\psi = (\neg r \wedge p) \vee (\neg q \to \neg r)$.

(a) Fill out the truth table for φ and ψ and their subformulas.

p	q	r	$\neg q$	$\neg r$	$p \to q$	$\neg r \wedge p$	$\neg q \to \neg r$	φ	ψ
F	**F**	**F**							
F	**F**	**T**							
F	**T**	**F**							
F	**T**	**T**							
T	**F**	**F**							
T	**F**	**T**							
T	**T**	**F**							
T	**T**	**T**							

(b) Which of the formulas is satisfiable?

(c) Which of the formulas is valid?

(d) Which of the two formulas φ and ψ entails the other?

1.1.5 Modelling Example

22. [Lecture] Use propositional logic to solve Sudoku. Rules: A Sudoku grid consists of a 9x9 square, which is partitioned into nine 3x3 squares. The goal of the game is to write one number from 1 to 9 in each cell in such a way, that each row, each column, and each 3x3-square contains each number exactly once. Usually several numbers are already given.

	6		7			1	5	
		3	9			8		
		2	3				4	9
		7			4			
	4			9			8	
			1			4		
6	7				9	3		
		9			2	5		
	2	8			7		6	

Sudoku

In order to model SUDOKU using propositional logic, we first need to define the propositional variables that we want to use in our formula. We define variables x_{ijk} for every row i, for

every column j, and for every value k. This encoding yields to 729 variables ranging from x_{111} to x_{999}. Using this variables, define the constraints for the rows, the columns, the 3x3-squares and the predefined numbers.

1.2 Self-Assessment

1.2.1 Declarative Sentences

23. [Self-Assessment] Model the following sentences as detailed as possible in propositional logic.
    (a) If all students prepare themselves appropriately, everyone will pass the exam.
    (b) Graz is the second biggest city of Austria.
    (c) If I only had more money!
24. [Self-Assessment] Model the following sentences as detailed as possible in propositional logic.
    (a) If I pass the exam, then if I pass it with more than 90 Points I will get the best grade possible.
    (b) Do you like Pizza?
    (c) All cats hate dogs and love mice.
25. [Self-Assessment] Model the following sentences as detailed as possible in propositional logic.
    (a) Bob will win the lottery, if and only if he gets all the numbers right.
    (b) Mozart was born in Salzburg, not in Innsbruck.
    (c) If the year is a leap-year, then February will have 29 days.
26. [Self-Assessment] Model the following sentences as detailed as possible in propositional logic.
    (a) Either Bob, Alice or neither are going to the lecture today.
    (b) Today is Friday, if and only if yesterday was Thursday and tomorrow is not Sunday.
    (c) Try to be patient and please be quiet.
27. [Self-Assessment] Model the following sentences as detailed as possible in propositional logic.
    (a) If a formula is unsat, it cannot be valid.
    (b) It can be proven that there exists an infinite number of primes.
    (c) A sentence is called declarative, if and only if it can be assigned a truth value.
28. [Self-Assessment] Model the following sentences as detailed as possible in propositional logic.
    (a) Today it will be either be foggy or it will rain today, but not both.
    (b) If and only if everybody comes in a costume to the party, we have a carnival.
    (c) No pain, no gain.
29. [Self-Assessment] Model the following sentences as detailed as possible in propositional logic.
    (a) If all students pass, the professor will be happy.
    (b) Bob is taller than Alice, but shorter than Charlie.
    (c) If there is lightning there must be thunder and vice versa.
30. [Self-Assessment] Model the following sentences as detailed as possible in propositional logic.
    (a) If the past hurts, you can either run from it, or learn from it.
    (b) If life gives you lemons, make lemonade.
    (c) A good pizza has salami or tuna on it, but not both at the same time.

1.2.2 Syntax of Propositional Logic

31. [Self-Assessment] Consider a formula φ in propositional logic. How is a propositional formula constructed and of what elements does it consist of?

32. [Self-Assessment] How can you determine using a parse tree whether a string is a *well-formed formula*?

33. [Self-Assessment] Determine whether the string $(a \vee b) \rightarrow (\neg(x\neg))$ is a well-formed formula using the parse tree. Explain your answer.

34. [Self-Assessment] Given the formula $\varphi = p \vee q \wedge q \rightarrow \neg r \leftrightarrow \neg p \wedge s$, how should the formula be interpreted according to the binding priorities? Make brackets to make the correct binding priorities clear and draw the parse tree for φ.

1.2.3 Semantics of Propositional Logic

35. [Self-Assessment] Give the definition of the *semantics* of propositional logic.

36. [Self-Assessment] Give the definition of a model $\mathcal{M}$ of a formula in propositional logic?

37. [Self-Assessment] What is the difference between a *satisfying model* and a *falsifying model* of a formula in propositional logic? Give a satisfying and a falsifying model for the formula $\varphi = a \rightarrow b$.

38. [Self-Assessment] Given is a formula $\varphi = ((p \wedge \neg q) \rightarrow (p \vee \neg r)) \wedge (\neg q \rightarrow \neg r)$ and a model $\mathcal{M} = \{p = T,\ q = F,\ r = T\}$. Determine the truth value of φ for the given model $\mathcal{M}$ using its parse tree.

39. [Self-Assessment] Given is a formula $\varphi = ((q \rightarrow \neg p) \vee r) \rightarrow (q \wedge (r \rightarrow p))$. Determine a satisfying model $\mathcal{M}_1$ and a falsifying model $\mathcal{M}_2$ using its parse tree.

40. [Self-Assessment] Given is a formula $\varphi = (\neg(r \leftrightarrow q) \rightarrow \neg r) \wedge (\neg(r \rightarrow q) \vee (p \rightarrow q))$. Determine a satisfying model $\mathcal{M}_1$ and a falsifying model $\mathcal{M}_2$ using its parse tree.

1.2.4 Semantic Entailment, Equivalence, Satisfiability and Validity

41. [Self-Assessment] Consider a formula φ in propositional logic. In the following list, tick all statements that are true.

    - ☐ If φ is a tautology, a falsifying model can be found.
    - ☐ If φ is equivalent to ψ, a satisfying model for φ always satisfies ψ.
    - ☐ If φ has no satisfying model, it is called a tautology.
    - ☐ If φ semantically entails ψ, a satisfying model for ψ always satisfies φ.

42. [Self-Assessment] Why are truth tables, in general, not used to determine equivalence of large formulas?

43. [Self-Assessment] Consider a formula φ in propositional logic. You want to test whether φ is *valid*. However, you only have a procedure for checking satisfiability. Describe how to use this procedure to determine whether φ is valid.

44. [Self-Assessment] Consider the propositional formulas $\varphi = (p \vee q) \to r$, and $\psi = r \vee (\neg p \wedge \neg q)$.

    (a) Fill out the truth table for φ and ψ (and their subformulas).

p	q	r	$\neg p$	$\neg q$	$p \vee q$	$\neg p \wedge \neg q$	φ	ψ
F	**F**	**F**						
F	**F**	**T**						
F	**T**	**F**						
F	**T**	**T**						
T	**F**	**F**						
T	**F**	**T**						
T	**T**	**F**						
T	**T**	**T**						

    (b) Which of the formulas is satisfiable?

    (c) Which of the formulas is valid?

    (d) Is φ equivalent to ψ?

    (e) Does φ semantically entail ψ?

    (f) Does ψ semantically entail φ?

45. [Self-Assessment] Consider the Boolean functions φ_1 and φ_2 over variables p, q, and r. Their truth table is given below.

p	q	r	φ_1	φ_2	ψ	γ
F	**F**	**F**	**F**	**F**		
F	**F**	**T**	**F**	**F**		
F	**T**	**F**	**T**	**T**		
F	**T**	**T**	**F**	**F**		
T	**F**	**F**	**T**	**F**		
T	**F**	**T**	**F**	**T**		
T	**T**	**F**	**F**	**F**		
T	**T**	**T**	**F**	**F**		

    (a) Fill the column for ψ such that φ_1 entails ψ (i.e., $\varphi_1 \models \psi$), but φ_2 does *not* entail ψ (i.e., $\varphi_2 \not\models \psi$).

    (b) Fill the column for γ such that φ_1 implies γ (i.e., $\varphi_1 \to \gamma$) as well as φ_2 implies γ (i.e., $\varphi_2 \to \gamma$).

46. [Self-Assessment] Consider the propositional formula $\varphi = p \to (q \to r)$.

    (a) Fill out the truth table for φ and its subformulas.

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

(b) Is φ satisfiable?

(c) Give a formula ψ that is semantically equivalent to φ, but does not use the "$\rightarrow$" connective.

(d) How can you check whether ψ is semantically equivalent to φ?

47. [Self-Assessment] Consider the propositional formula $\varphi = (p \rightarrow q) \wedge (q \rightarrow r) \wedge (\neg r \vee p)$.

(a) Fill out the truth table for φ (and its subformulas).

p	q	r	$(p \rightarrow q)$	$(q \rightarrow r)$	$\neg r$	$(\neg r \vee p)$	φ
F	**F**	**F**					
F	**F**	**T**					
F	**T**	**F**					
F	**T**	**T**					
T	**F**	**F**					
T	**F**	**T**					
T	**T**	**F**					
T	**T**	**T**					

(b) Is φ satisfiable?

(c) Is φ valid?

(d) Give a formula ψ that semantically entails φ (i.e., it should be the case that $\psi \models \varphi$).

(e) How can you check, using a truth table, whether ψ semantically entails φ?

48. [Self-Assessment] Consider the propositional formula $\varphi = (\neg p \rightarrow r) \wedge (r \rightarrow \neg p) \wedge q$.

(a) Fill out the truth table for φ (and its subformulas).

p	q	r	$\neg p$	$(\neg p \rightarrow r)$	$(r \rightarrow \neg p)$	φ
F	**F**	**F**				
F	**F**	**T**				
F	**T**	**F**				
F	**T**	**T**				
T	**F**	**F**				
T	**F**	**T**				
T	**T**	**F**				
T	**T**	**T**				

(b) Is the negation of φ satisfiable?

(c) Is the negation of φ valid?

(d) Give a formula ψ that semantically entails φ (i.e., it should be the case that $\psi \models \varphi$).

(e) Give a formula ψ such that φ semantically entails ψ (i.e., it should be the case that $\varphi \models \psi$).

49. [Self-Assessment] Consider the propositional formula $\varphi = ((p \rightarrow q) \wedge (\neg p \rightarrow \neg q)) \rightarrow r$.

(a) Fill out the truth table for φ and its subformulas.

p	q	r	$\neg p$	$\neg q$	$(p \rightarrow q)$	$(\neg p \rightarrow \neg q)$	$(p \rightarrow q) \wedge (\neg p \rightarrow \neg q)$	φ
F	**F**	**F**						
F	**F**	**T**						
F	**T**	**F**						
F	**T**	**T**						
T	**F**	**F**						
T	**F**	**T**						
T	**T**	**F**						
T	**T**	**T**						

(b) Is φ unsatisfiable?

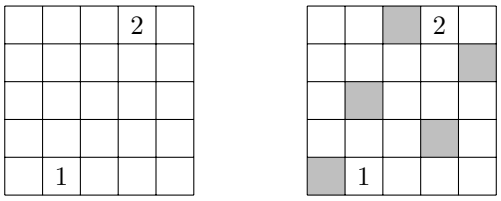
(c) Is the negation of φ valid?

(d) Give a formula ψ that is semantically equivalent to φ, but does not use the "$\rightarrow$" connective.

1.2.5 Modelling Example

50. [Self-Assessment] Describe the Latin Square Puzzle using propositional logic.

In the Latin Square Puzzle one has to color cells in an (n×n) grid such that there is exactly one colored cell in each row and each column. Furthermore, colored cells must not be adjacent to each other (also not diagonally). Numbers contained in certain cells of the grid indicate the exact number of colored cells that have to be adjacent (including diagonally) to it. Numbered cells can contain the numbers 0, 1, 2 and cannot be colored.



Example Latin Square Puzzle and its solution

Find propositional formulas which describe the puzzle and which could be used to solve it. Focus on explaining the concept of the formulas. You do not have to explicitly list all formulas and you do not have to solve the puzzle.

Hints: Use propositional atoms $c_{i,j}$, $c_{i,j,0}$, $c_{i,j,1}$,$c_{i,j,2}$ to represent each cell of the $(n \times n)$ game board. If $c_{i,j}$ has the value *True*, the cell i, j is colored, otherwise it is not colored. If $c_{i,j,x}$ has the value *True*, the cell i, j contains the number x.

Express the following constraints:

(a) There is exactly one colored cell in row i.

(b) No colored cells are adjacent to each other.

(c) No numbered cells can be colored.

(d) Numbered cells are adjacent to the indicated amount of colored cells.

2 Natural deduction for Propositional Logic

2.1 Lecture

For each of the following sequents, either provide a natural deduction proof, or a counter-example that proves the sequent invalid.

For proofs, clearly indicate which rule, and what assumptions/premises/
intermediate results you are using in each step. Also clearly indicate the scope of any boxes you use.

For counterexamples, give a complete model. Show that the model satisfies the premise(s) of the sequent in question, but does not satisfy the respective conclusion.

2.1.1 Rules for natural deduction

1. [Lecture] Give the definition of a sequent. Give an example of a sequent and name the parts the sequent consists of.
2. [Lecture] Look at the following statements and tick them if they are true.

   ☐ In a sequent, premises entail a conclusion.
   ☐ In a sequent, conclusions entail a premise.
   ☐ A sequent is valid, if no proof for it can be found.
   ☐ A sequent is valid, if a proof for it can be found.

3. [Lecture] State the *AND-introduction* rule ($\wedge$i). Explain how the rule works.
4. [Lecture] $p, q, r \vdash p \wedge (q \wedge r)$
5. [Lecture] $p \wedge (q \wedge r) \vdash q$
6. [Lecture] $p \wedge q, \neg q \wedge r \vdash \neg\neg p \wedge \neg\neg r$
7. [Lecture] $\neg\neg\neg p \wedge q, \neg\neg r \vdash r \wedge \neg p \wedge \neg\neg q$
8. [Lecture] $p \wedge q, q \rightarrow \neg\neg r \vdash p \wedge r$
9. [Lecture] Explain the *implication-elimination* rule ($\rightarrow$e). Show how the *Modus Tollens* rule derives from the $\rightarrow$e rule?
10. [Lecture] $\neg p \rightarrow q, \neg\neg\neg q \wedge r \vdash p \wedge \neg\neg\neg q$
11. [Lecture] Translate the following reasoning into a sequent. If the sequent is valid, proof it using the rules of natural deduction. If the sequent is not valid, provide a counter example.

    If I press the button, the window opens.
    I pressed the button.
    Therefore, the window is open.

12. [Lecture] Explain the concept of boxes in deduction rules and why they are needed. What does it mean if you make an *assumption* within a box? Where is this assumption valid?
13. [Lecture] $p \rightarrow (q \wedge r), q \rightarrow s \vdash p \rightarrow (s \wedge r)$
14. [Lecture] Why are there two rules for the $\vee$-*introduction* rule. Explain, why you are able to connect any formula to a certain formula φ using the connective $\vee$.
15. [Lecture] $p \wedge q, r \rightarrow s \vdash (p \vee (r \rightarrow s)) \wedge (q \vee ((t \vee r) \rightarrow u))$

16. [Lecture] Explain the *OR-elimination* ($\vee$-e) rule of the natural deduction calculus. In particular, why does it rule require two boxes?
17. [Lecture] $p \vee \neg\neg q, \neg p \wedge \neg q \vdash s \vee \neg t$
18. [Lecture] $\neg q \vee \neg p \vdash \neg(q \wedge p)$
19. [Lecture] $\vdash ((p \to q) \to p) \to p$
20. [Lecture] $\neg(q \wedge p) \vdash \neg q \vee \neg p$
21. [Lecture] Explain in your own words, how to proof a sequent in the natural deduction calculus. What steps do you need to take and which tips can be helpful when solving such proofs?

2.1.2 Soundness and completeness of natural deduction

22. [Lecture] "Natural deduction for propositional logic is *sound* and *complete*." Explain in your own words what this means.
23. [Lecture] How can you show that a sequent is not valid? Is this a consequence of soundness or completeness. Explain your answer.
24. [Lecture] $p \to q, q \to r \vdash r$.
25. [Lecture] Translate the following reasoning into a sequent. If the sequent is valid, proof it using the rules of natural deduction. If the sequent is not valid, provide a counter example.

    If I press the button, the window opens.
    The window is open.
    Therefore, I pressed the button.

2.2 Practicals

For each of the following sequents, either provide a natural deduction proof, or a counter-example that proves the sequent invalid.

For proofs, clearly indicate which rule, and what assumptions/premises/
intermediate results you are using in each step. Also clearly indicate the scope of any boxes you use.

For counterexamples, give a complete model. Show that the model satisfies the premise(s) of the sequent in question, but does not satisfy the respective conclusion. For each of the following sequents, either provide a natural deduction proof, or a counter-example that proves the sequent invalid.

1. [Practicals] **[2 Points]**

   (a) If I am ill, I go to the doctor.
   I am ill.
   Therefore, I go to the doctor.

   (b) If I am ill, I go to the doctor.
   I go to the doctor.
   Therefore, I am ill.

(c) (Solve without using the Modus Tollens)
If I am ill, I go to the doctor.
I did not go to the doctor.
Therefore, I am not ill.

2. [Practicals] **[2 Points]**
   (a) $(p \land q) \land \neg r \vdash q \lor r$
   (b) $(p \lor q) \land \neg r \vdash q \land r$
3. [Practicals] **[2 Points]**
   (a) $\vdash (p \to q) \to p$
   (b) $\vdash p \to (q \to p)$
4. [Practicals] **[2 Points]** $\neg(a \land b) \lor \neg c \vdash \neg(a \land b) \to c \lor a$
5. [Practicals] **[2 Points]** $p \land q \lor r \vdash (p \lor r) \land (q \lor r)$
6. [Practicals] **[2 Points]** $\neg\neg x \to \neg y \land z \vdash z \to \neg x \land \neg\neg y$
7. [Practicals] **[2 Points]** $\vdash \neg(p \land q) \lor p$
8. [Practicals] **[2 Points]** $\neg(a \lor b) \vdash \neg a \land \neg b$
9. [Practicals] **[2 Points]** $(s \lor \neg u) \to t \vdash (\neg s \land u) \lor t$
10. [Practicals] **[2 Points]** $\neg\neg k \to (l \lor m), \neg\neg\neg l \to m \vdash \neg k \lor (l \lor \neg\neg m)$

2.3 Self Evaluation

For each of the following sequents, either provide a natural deduction proof, or a counter-example that proves the sequent invalid.

For proofs, clearly indicate which rule, and what assumptions/premises/
intermediate results you are using in each step. Also clearly indicate the scope of any boxes you use.

For counterexamples, give a complete model. Show that the model satisfies the premise(s) of the sequent in question, but does not satisfy the respective conclusion.

2.3.1 Rules for natural deduction

25. [Self-Assessment] $p \land (q \land r) \vdash (p \land q) \land r$
26. [Self-Assessment] $\neg\neg p \land \neg\neg q, r \land s \vdash (p \land r) \land \neg\neg s$
27. [Self-Assessment] $(p \to q) \land (q \to r), p \vdash \neg\neg r \land \neg p$
28. [Self-Assessment] $(\neg p \to q) \land (q \to r), \neg r \vdash \neg\neg\neg r \land \neg p$
29. [Self-Assessment] Explain the *implication-introduction* rule ($\to$i).
30. [Self-Assessment] $(p \to q) \to r \vdash \neg r \land \neg s \to \neg(p \to q)$
31. [Self-Assessment] $p \to q \vdash (r \to p) \to (r \to q)$
32. [Self-Assessment] $p \to q, p \land (r \lor q) \vdash (q \to p) \to ((s \land t) \lor q) \land (r \lor q)$
33. [Self-Assessment] $p \lor q, \neg p \lor r \vdash q \lor r$

34. [Self-Assessment] $p \to q, p \land r \lor q \vdash (q \to p) \to ((s \land t) \lor q) \land (r \lor q)$

35. [Self-Assessment] $p \lor q, p \to r, \neg s \to \neg q \vdash r \lor s$

36. [Self-Assessment] Look at the following statements and tick them if they are true.

    ☐ Given two premises φ and ψ, we can conclude that $\varphi \land \psi$ holds using $\land$-introduction.
    ☐ Given two premises φ and ψ, we can conclude that $\varphi \lor \psi$ holds using $\lor$-introduction.
    ☐ Given a premise $\varphi \land \psi$, we can conclude φ with $\land$-elimination.
    ☐ Given a premise $\varphi \lor \psi$, we can conclude φ with $\lor$-elimination.

37. [Self-Assessment] $\vdash p \to (q \to p)$

38. [Self-Assessment] Translate the following reasoning into a sequent. If the sequent is valid, proof it using the rules of natural deduction. If the sequent is not valid, provide a counter example.

    If I press the button, the window opens.
    The window is not open.
    Therefore, I didn't press the button.

39. [Self-Assessment] Explain the $\bot$-*elimination* rule of the natural deduction calculus. Why can you deduce a formula φ from something, that is wrong?

40. [Self-Assessment] $\neg q \lor p \vdash q \to (p \lor r)$

41. [Self-Assessment] $p \to (q \lor r), \neg q \land \neg r \vdash \neg p$

42. [Self-Assessment] Derive the *Proof-By-Contradiction*-rule from the $\neg$-*introduction* rule.

43. [Self-Assessment] $\neg(q \lor p) \vdash \neg q \land p$

44. [Self-Assessment] $\vdash (p \to q) \lor (q \to r)$

45. [Self-Assessment] $(p \to q) \land (q \to p) \vdash (p \land q) \lor (\neg p \land \neg q)$

46. [Self-Assessment] Look at the following statements and tick them if they are true.

    ☐ A sequent always has to have at least one premise to be formally correct.
    ☐ When proving a sequent you start your proof with the premise(s) and end it with the conclusion.
    ☐ A natural deduction proof can theoretically have infinite assumption boxes in it.
    ☐ A natural deduction rule can only be applied on the bottom-level connective of a formula.

2.3.2 Soundness and completeness of natural deduction

47. [Self-Assessment] Explain what it means that natural deduction for propositional logic is *sound*. What is the difference to *completeness*?

48. [Self-Assessment] Explain what it means that natural deduction for propositional logic is *sound*. What is the difference to *completeness*?

49. [Self-Assessment] Look at the following statements and tick them if they are true.

    ☐ Any sequent that is a correct semantic entailment can be proven.

- ☐ Any sequent that can be proven is a correct semantic entailment.
- ☐ If a sequent is not provable, the semantic entailment relation does hold.
- ☐ If for a sequent the semantic entailment relation does not hold, it cannot be proven with natural deduction.

50. [Self-Assessment] Natural deduction for propositional logic is sound and complete. In the following list, mark each statement with either **S**, **C**, **B**, or **N**, depending on whether the corresponding statement follows from **S**oundness, **C**ompleteness, **B**oth, or **N**either. (Note: If a statement is in itself factually wrong, or has nothing to do with soundness and completeness, mark it **N**, since it follows from neither soundness nor completeness.)

- ☐ There is no correct sequent for which there is no proof.
- ☐ Every sequent has a proof.
- ☐ If all models that satisfy the premise(s) of a given sequent also satisfy the conclusion of the sequent, there exists a proof for the sequent.
- ☐ A sequent has a proof if and only if it is semantically correct.
- ☐ An incorrect sequent does not have a proof.
- ☐ Every propositional formula is either valid or not valid.
- ☐ If a model satisfies the premise(s) of a given sequent, but does not satisfy the conclusion of the sequent, it is not possible to construct a proof for the sequent.

51. [Self-Assessment] Natural deduction for propositional logic is sound and complete. In the following list, mark each statement with either **S**, **C**, **B**, or **N**, depending on whether the corresponding statement follows from **S**oundness, **C**ompleteness, **B**oth, or **N**either. (Note: If a statement is in itself factually wrong, or has nothing to do with soundness and completeness, mark it **N**, since it follows from neither soundness nor completeness.)

- ☐ A sequent has a proof if and only if it is semantically correct.
- ☐ If a model satisfies the premise(s) of a given sequent, but does not satisfy the conclusion of the sequent, it is not possible to construct a proof for the sequent.
- ☐ There is no correct sequent for which there is no proof.
- ☐ Every sequent has a proof.
- ☐ An incorrect sequent does not have a proof.
- ☐ Every propositional formula is either valid or not valid.
- ☐ If all models that satisfy the premise(s) of a given sequent also satisfy the conclusion of the sequent, there exists a proof for the sequent.

52. [Self-Assessment] Given an invalid sequent, how do you prove its invalidity?

53. [Self-Assessment] $\neg(p \vee \neg q) \vdash p$

54. [Self-Assessment] $p \to q \vdash ((p \vee q) \to p) \wedge (p \to (p \vee q))$

55. [Self-Assessment] $p \vee q, \neg q \vee r \vdash r$

56. [Self-Assessment] $(p \wedge q) \to (\neg r \wedge \neg s), \neg r \wedge \neg s \vdash p$

3 Combinational Equivalence Checking

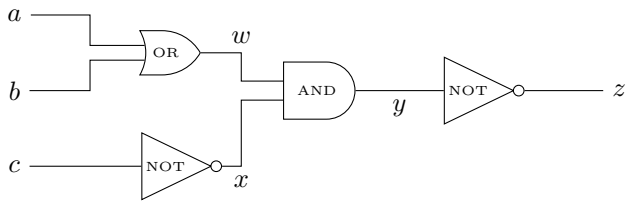
3.1 Lecture

3.1.1 Translating a Circuit into a Formula

1. [Lecture] Explain the algorithm of how to decide the equivalence of combinational circuits via the reduction to satisfiability.

2. [Lecture] Explain the process of translating a combinational circuit into a propositional formula. Draw a combinational circuit with 2 or 3 gates and give the corresponding propositional formula.

3. [Lecture] Compute the propositional formula of the following circuit.



3.1.2 Relations between Satisfiability, Validity, Equivalence and Semantic Entailment

4. [Lecture] Explain the duality of *satisfiability* and *validity* and additional provide examples that show the duality.

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

3.1.3 Normal Forms

6. [Lecture] Explain the following terms and give examples: (a) literal, (b) cube, and (c) clause.

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

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

3.1.4 Tseitin Encoding

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

$$\chi \leftrightarrow (\varphi \vee \psi) \quad \Leftrightarrow \quad (\neg\varphi \vee \chi) \wedge (\neg\psi \vee \chi) \wedge (\neg\chi \vee \varphi \vee \psi) \tag{1}$$

$$\chi \leftrightarrow (\varphi \wedge \psi) \quad \Leftrightarrow \quad (\neg\chi \vee \varphi) \wedge (\neg\chi \vee \psi) \wedge (\neg\varphi \vee \neg\psi \vee \chi) \tag{2}$$

$$\chi \leftrightarrow \neg\varphi \quad \Leftrightarrow \quad (\neg\chi \vee \neg\varphi) \wedge (\varphi \vee \chi) \tag{3}$$

9. [Lecture] What is the advantage of applying *Tseitin encoding* to obtain a CNF, especially compared to using truth tables?

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

11. [Lecture] Explain the concept of equisatisfiability. Given a propositional logic formula φ, the Tseitin algorithm computes an equisatisfiable formula $CNF(\varphi)$ in CNF. Why is this enough for equivalence checking?

12. [Lecture] Apply Tseitin's encoding to the following formula: $\varphi = \neg(a \vee \neg b) \vee (\neg a \wedge c)$. For each variable you introduce, clearly indicate which subformula of φ it represents.

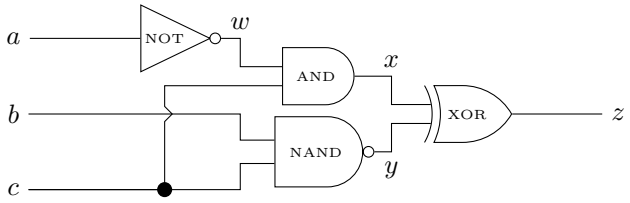
3.1.5 CEC Example

13. [Lecture] Check whether $\varphi_1 = a \wedge \neg b$ and $\varphi_2 = \neg(\neg a \vee b)$ are semantically equivalent using the reduction to satisfiability. Prepare everything until you have a formula CNF(φ), that you can give to a SAT solver.

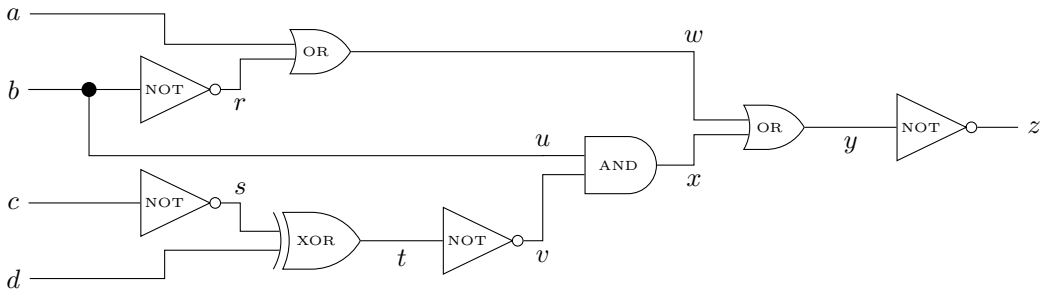
3.2 Self-Assessment

3.2.1 Translating a Circuit into a Formula

14. [Self-Assessment] Compute the propositional formula of the following circuit.



15. [Self-Assessment] Compute the propositional formula of the following circuit.



3.2.2 Relations between Satisfiability, Validity, Equivalence and Entailment

16. [Self-Assessment] *A formula φ is valid, if and only if $\neg\varphi$ is not satisfiable.* Explain why this statement holds in your own words.

17. [Self-Assessment] 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?

18. [Self-Assessment] Given a propositional logic formula φ. How can we check whether φ is *valid* using a decision procedure for (a) satisfiability and (b) equivalence?

19. [Self-Assessment] 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*.

20. [Self-Assessment] 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.

21. [Self-Assessment] 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..

3.2.3 Normal Forms

22. [Self-Assessment] Define the *Disjunctive Normal Form (DNF)* of formulas in propositional logic. Use the proper terminology and give an example.

23. [Self-Assessment] Define the *Conjunctive Normal Form (CNF)* of formulas in propositional logic. Use the proper terminology and give an example.

24. [Self-Assessment] Tick all statements that are true.

- ☐ A *clause* is a disjunction of literals.
- ☐ A *clause* is a conjunction of literals.
- ☐ A *cube* is disjunction of literals.
- ☐ A *cube* is a conjunction of literals.

25. [Self-Assessment] Given the formula φ with the variables $x_1, ..., x_n$. Tick all statements that are true.

- ☐ A *literal* is a variables x_i or its negation.
- ☐ A *literal* forms a formula in conjunctive normal form.
- ☐ A *literal* forms a formula in disjunctive normal form.
- ☐ A *literal* is called *positive*, if it is the negation of a variable.
- ☐ A *literal* is called *negative*, if it is the negation of a variable.

26. [Self-Assessment] Look a the following statements and tick all items that conform to a *DNF*.

- ☐ $a \vee b$
- ☐ A DNF is a conjunction of clauses.
- ☐ $(a \vee b) \wedge (\neg b \vee \neg a \vee c) \wedge \neg c$
- ☐ $(a \wedge b) \vee (\neg b \wedge \neg a \wedge c) \vee \neg c$
- ☐ A DNF is a conjunction of disjunctions of literals.
- ☐ b

- ☐ $a \wedge b \wedge \neg c$
- ☐ $(\neg a \wedge b) \wedge (\neg a \wedge c)$
- ☐ A DNF is a disjunction of cubes.
- ☐ $\neg(a \wedge \neg b) \wedge c$
- ☐ A DNF is a disjunction of conjunctions of literals.
- ☐ $a \wedge \neg b$

27. [Self-Assessment] Tick each correct ending of the following sentence. "A *Conjunctive Normal Form* is …

- ☐ …a conjunction of disjunctions of literals."
- ☐ …a conjunction of clauses."
- ☐ …a formula that consists only of logical AND operations on sub-formulas which only consist of OR operations on just variables and negations of variables."

28. [Self-Assessment] SAT solvers usually require input formulas to be in *Conjunctive Normal Form* (CNF). In the following list, tick all items that conform to CNF.

- ☐ A formula φ that consists of a conjunction of clauses $c_1, c_2, \ldots, c_n$.
- ☐ A formula φ that consists of a disjunction of clauses $c_1, c_2, \ldots, c_n$.
- ☐ A formula φ that consists of a conjunction of cubes $c_1, c_2, \ldots, c_n$.
- ☐ A formula φ that consists of a disjunction of cubes $c_1, c_2, \ldots, c_n$.
- ☐ A literal l.

29. [Self-Assessment] In the following list, tick all items that conform to the Conjunctive Normal Form (CNF).

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

30. [Self-Assessment] In the following list, tick all items that conform to the Disjunctive Normal Form (DNF).

- ☐ $(a \wedge b \wedge \neg c) \vee (\neg b \wedge \neg c) \vee (e \wedge \neg f)$
- ☐ $(a \vee b \vee \neg c) \wedge (\neg b \vee \neg c) \wedge (e \vee \neg f)$
- ☐ $\neg b$
- ☐ $a \wedge \neg b$
- ☐ $a \vee \neg b$
- ☐ $a \vee (\neg b \wedge c)$
- ☐ $(a \vee \neg b) \wedge c$
- ☐ $\neg(p \vee q)$

☐ $x \vee \neg y \vee z$

31. [Self-Assessment] 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.

32. [Self-Assessment] Given the formula $\varphi = (a \wedge \neg b \wedge \neg c) \vee ((\neg c \rightarrow a) \rightarrow b)$. Use the truth table of φ to compute its representation in (a) CNF and (b) DNF.

33. [Self-Assessment] Given the formula $\varphi = (q \rightarrow \neg r) \wedge \neg(p \vee q \vee \neg r)$. Use the truth table of φ to compute its representation in (a) CNF and (b) DNF.

34. [Self-Assessment] Consider the propositional formula $\varphi = (p \vee \neg q) \rightarrow (\neg p \wedge \neg r)$. 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	$\neg q$	$p \vee \neg q$	$\neg p$	$\neg r$	$\neg p \wedge \neg r$	$\varphi = (p \vee \neg q) \rightarrow (\neg p \wedge \neg r)$
F	**F**	**F**						
F	**F**	**T**						
F	**T**	**F**						
F	**T**	**T**						
T	**F**	**F**						
T	**F**	**T**						
T	**T**	**F**						
T	**T**	**T**						

35. [Self-Assessment] Given the formula $\varphi = \neg(a \rightarrow \neg b) \vee (\neg a \rightarrow c)$. Use the truth table of φ to compute its representation in (a) CNF and (b) DNF.

36. [Self-Assessment] 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**					

3.2.4 Tseitin Encoding

Consider the following logic equivalences when applying Tseitin's encoding:

$$\chi \leftrightarrow (\varphi \vee \psi) \quad \Leftrightarrow \quad (\neg\varphi \vee \chi) \wedge (\neg\psi \vee \chi) \wedge (\neg\chi \vee \varphi \vee \psi) \tag{4}$$

$$\chi \leftrightarrow (\varphi \wedge \psi) \quad \Leftrightarrow \quad (\neg\chi \vee \varphi) \wedge (\neg\chi \vee \psi) \wedge (\neg\varphi \vee \neg\psi \vee \chi) \tag{5}$$

$$\chi \leftrightarrow \neg\varphi \quad \Leftrightarrow \quad (\neg\chi \vee \neg\varphi) \wedge (\varphi \vee \chi) \tag{6}$$

37. [Self-Assessment] (a) What does it mean that two formulas φ and ψ are *equisatisfiable*? (b) Explain the difference between *satisfiability* and *equisatisfiability*.

38. [Self-Assessment] Suppose you have a propositional formula φ. Let ψ be the result of applying Tseitin's encoding to φ. Is φ *equivalent* to ψ? Provide a reason for your answer and explain the relation between φ and ψ.

39. [Self-Assessment] Explain the concept of *Tseitin's Encoding* to obtain formulas in CNF. Give step-by-step instructions of how to apply Tseitin's encoding to a propositional formula. (Note: Focus on the concept. You do *not* need to quote the rewrite rules!)

40. [Self-Assessment] Derive a Rewrite-Rule for a NAND node, i.e., what clauses are introduced by the node $x \leftrightarrow (p \text{ NAND } q)$?

41. [Self-Assessment] Derive a Rewrite-Rule for a NOR node, i.e., what clauses are introduced by the node $x \leftrightarrow (p \text{ NOR } q)$?

42. [Self-Assessment] Derive a Rewrite-Rule for a XOR node, i.e., what clauses are introduced by the node $x \leftrightarrow (p \oplus q)$?

43. [Self-Assessment] Apply Tseitin's encoding to the following formula:

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

For each variable you introduce, clearly indicate which subformula of φ it represents.

44. [Self-Assessment] Apply Tseitin's encoding to the following formula:

$$\varphi = (q \wedge \neg r) \vee \neg(q \wedge \neg r)$$

. For each variable you introduce, clearly indicate which subformula of φ it represents.

45. [Self-Assessment] Apply Tseitin's encoding to the following formula:

$$\varphi = (\neg(\neg a \wedge b) \wedge \neg c).$$

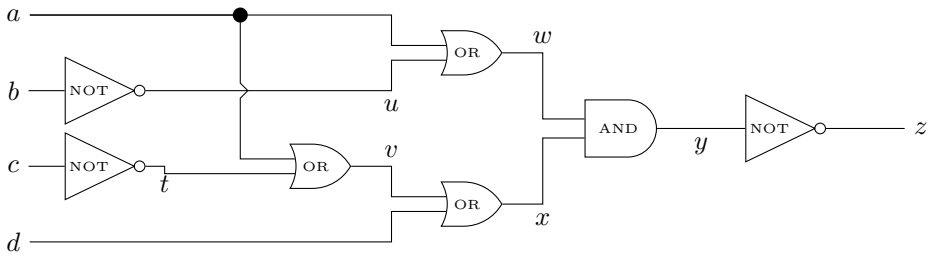
For each variable you introduce, clearly indicate which subformula of φ it represents.

46. [Self-Assessment] Apply Tseitin's encoding to the following formula:

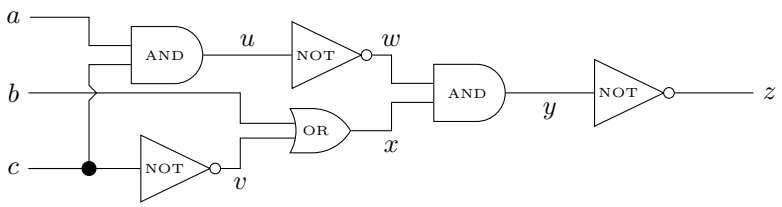
$$\varphi = (p \vee \neg q) \vee (\neg p \wedge \neg r).$$

For each variable you introduce, clearly indicate which subformula of φ it represents.

47. [Self-Assessment] Compute the propositional formula of the following circuit and transform it into an equisatisfiable formula in CNF by applying Tseitin's encoding. For each variable you introduce, clearly indicate which subformula of φ it represents.



48. [Self-Assessment] Compute the propositional formula of the following circuit and transform it into an equisatisfiable formula in CNF by applying Tseitin's encoding. For each variable you introduce, clearly indicate which subformula of φ it represents.



3.2.5 CEC Example

49. [Self-Assessment] 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. Prepare everything until you have a formula $\mathrm{CNF}(\varphi)$, that you can give to a SAT solver.

4 SAT Solvers

4.1 Lecture

4.1.1 The DPLL-Algorithm

1. [Lecture] Use the DPLL algorithm (*without* BCP, PL and clause learning) to determine whether or not the set of clauses given is satisfiable. Decide variables in alphabetical order starting with the *positive* phase. If the set of clauses resulted in `SAT`, give a satisfying model.

   Clause 1: $(\neg a \vee b)$

   Clause 2: $(\neg b \vee c)$

   Clause 3: $(\neg c \vee d)$

   Clause 4: $(\neg d \vee e)$

   Clause 5: $(\neg e \vee \neg a)$

2. [Lecture] In the context of the DPLL algorithm, explain what a *Unit Clause* is. Give an example.

3. [Lecture] Use the DPLL algorithm with *Boolean Constrain Propagation* (*without* PL and clause learning) to determine whether or not the set of clauses given is satisfiable. Decide variables in alphabetical order starting with the *positive* phase. If the set of clauses resulted in `SAT`, give a satisfying model.

   Clause 1: $(\neg a \vee b)$

   Clause 2: $(\neg b \vee c)$

   Clause 3: $(\neg c \vee d)$

   Clause 4: $(\neg d \vee e)$

   Clause 5: $(\neg e \vee \neg a)$

4. [Lecture] In the context of the DPLL algorithm, explain what a *Pure Literal* is. Give an example.

5. [Lecture] In the context of the DPLL algorithm, explain why it is advantageous to apply *Boolean Constrain Propagation (BCP)* and *Pure Literals (PL)* before making a decision.

6. [Lecture] Use the DPLL algorithm with *Boolean Constrain Propagation* and *Pure Literals* (*without* clause learning) to determine whether or not the set of clauses given is satisfiable. Decide variables in alphabetical order starting with the *positive* phase. If the set of clauses resulted in `SAT`, give a satisfying model.

   Clause 1: $(\neg a \vee b)$

   Clause 2: $(\neg b \vee c)$

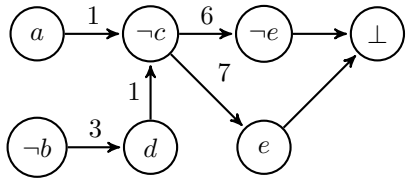
   Clause 3: $(\neg c \vee d)$

   Clause 4: $(\neg d \vee e)$

   Clause 5: $(\neg e \vee \neg a)$

7. [Lecture] In the context of the DPLL algorithm, explain what *Conflict-Driven Clause Learning* is and why most modern SAT solvers use this technique.

8. [Lecture] Consider the following conflict graph with the following set of clauses:



Clause 1: $\{\neg a, \neg c, \neg d\}$

Clause 2: $\{a, \neg d\}$

Clause 3: $\{b, d\}$

Clause 4: $\{\neg b, d, e\}$

Clause 5: $\{\neg b, \neg e\}$

Clause 6: $\{c, \neg e\}$

Clause 7: $\{c, e\}$

Give the resolution proof for the given conflict graph and clauses and state the clause to be learned from the conflict.

9. [Lecture] Use the DPLL algorithm with conflict-driven clause learning to determine whether or not the set of clauses given is satisfiable. Decide variables in alphabetical order starting with the *negative* phase. For conflicts, draw conflict graphs after the end of the table, and add the learned clause to the table.
If the set of clauses resulted in `SAT`, give a satisfying model. If the set of clauses resulted in `UNSAT`, give a resolution proof that shows that the conjunction of the clauses from the table is unsatisfiable.

Clause 1: $\{\neg a, \neg b\}$

Clause 2: $\{a, c\}$

Clause 3: $\{b, \neg c\}$

Clause 4: $\{\neg b, d\}$

Clause 5: $\{\neg c, \neg d\}$

Clause 6: $\{c, e\}$

Clause 7: $\{c, \neg e\}$

10. [Lecture] Use the DPLL algorithm with conflict-driven clause learning to determine whether or not the set of clauses given is satisfiable. Decide variables in alphabetical order starting with the *positive* phase. For conflicts, draw conflict graphs after the end of the table, and add the learned clause to the table.
If the set of clauses resulted in `SAT`, give a satisfying model. If the set of clauses resulted in `UNSAT`, give a resolution proof that shows that the conjunction of the clauses from the table is unsatisfiable.

Clause 1: $(\neg a \vee d)$

Clause 2: $(\neg d \vee c)$

Clause 3: $(\neg b \vee e)$

Clause 4: $(\neg b \vee \neg e)$

Clause 5: $(b \vee f)$

Clause 6: $(b \vee \neg f)$

11. [Lecture] Use the DPLL algorithm with conflict-driven clause learning to determine whether or not the set of clauses given is satisfiable. Decide variables in alphabetical order starting with the *negative* phase. For conflicts, draw conflict graphs after the end of the table, and add the learned clause to the table.
    If the set of clauses resulted in `SAT`, give a satisfying model. If the set of clauses resulted in `UNSAT`, give a resolution proof that shows that the conjunction of the clauses from the table is unsatisfiable.

    Clause 1: $(\neg a \vee \neg c)$

    Clause 2: $(b \vee c)$

    Clause 3: $(\neg b \vee \neg d)$

    Clause 4: $(\neg d \vee e)$

    Clause 5: $(d \vee e)$

    Clause 6: $(a \vee \neg c \vee \neg e)$

    Clause 7: $(\neg b \vee c \vee d)$

4.2 Practicals

For the following exercises use the DPLL algorithm (including Boolean Constraint Propagation (BCP), pure literals, and conflict-driven clause learning) to check on paper, if the following CNF formulas are satisfiable.

If the formula is satisfiable, give a satisfying model, else show a complete resolution proof for the formula's unsatisfiability.

- Write down all the steps of the DPLL algorithm,
- draw the conflict graphs,
- and state the resolution proofs for all learned clauses

Rules:

- When resolving a conflict, only undo the last decision.
- Choose variables for decisions, BCP and pure literals in alphabetical order, starting with the *negative* phase ($\neg a > a > \neg b > b ...$).
- Always try to perform BCP first, before checking for pure literals, before making a decision.

1. [Practicals] **[2 Points]**

   Clause 1: $\{a, b, c\}$
   Clause 2: $\{\neg a, \neg b, \neg c\}$
   Clause 3: $\{a, c, \neg e\}$
   Clause 4: $\{\neg b, \neg c, e\}$
   Clause 5: $\{b, e\}$
   Clause 6: $\{b, \neg d\}$
   Clause 7: $\{\neg c, d\}$
   Clause 8: $\{\neg c, e\}$

2. [Practicals] **[2.5 Points]**

   Clause 1: $\{\neg a, c\}$
   Clause 2: $\{\neg a, b, \neg c\}$
   Clause 3: $\{\neg b, e\}$
   Clause 4: $\{a, d\}$
   Clause 5: $\{a, \neg c\}$
   Clause 6: $\{\neg a, \neg e\}$
   Clause 7: $\{a, \neg b\}$
   Clause 8: $\{b, \neg d\}$

3. [Practicals] **[2.5 Points]**

   Clause 1: $\{a, \neg b, c\}$
   Clause 2: $\{b, \neg c, d\}$
   Clause 3: $\{a, \neg b\}$
   Clause 4: $\{a, c\}$
   Clause 5: $\{\neg c, \neg d\}$

4. [Practicals] **[3 Points]**

   Clause 1: $\{a, \neg b\}$
   Clause 2: $\{a, c\}$
   Clause 3: $\{\neg a, e\}$
   Clause 4: $\{b, c\}$
   Clause 5: $\{b, d\}$
   Clause 6: $\{b, \neg e\}$
   Clause 7: $\{\neg d, e\}$

5. [Practicals] **[3 Points]**

   Clause 1: $\{a, b, c\}$
   Clause 2: $\{\neg a, b\}$
   Clause 3: $\{\neg b, c\}$
   Clause 4: $\{\neg c, d\}$
   Clause 5: $\{\neg c, e\}$
   Clause 6: $\{\neg d, \neg e\}$

6. [Practicals] **[4 Points]**

   Clause 1: $\{a, \neg c, \neg e\}$
   Clause 2: $\{\neg a, \neg e\}$
   Clause 3: $\{b, e\}$
   Clause 4: $\{\neg b, d, e\}$
   Clause 5: $\{\neg b, \neg d\}$
   Clause 6: $\{c, \neg d\}$
   Clause 7: $\{c, d\}$

7. [Practicals] **[3 Points]**

   You are about to plan a train journey in Europe, but you are not yet sure, where to go. You have a few cities in mind, but there are a few restrictions due to a pandemic:

   Your biggest wish is to go to Paris, you are definitely going there. After visiting Paris you are either going to London, or to Madrid, but not both. There is no direct train from your home to Paris, therefore you can take a train either via Berlin or via Zurich. On your way back you can choose between Amsterdam or Zurich. As you want to visit as many cities as possible, you do not want to go trough Zurich twice, therefore you have to go at least through once trough Amsterdam or Berlin. As traveling is currently restricted due to a pandemic, you may not visit Madrid after you visited Berlin and vice versa. You may also not visit London after you went to Amsterdam and vice versa.

   Create a CNF from this description. You can use the following rule to make the formula shorter:

   $$(\neg s \wedge t) \vee (s \wedge \neg t) \vdash \neg s \vee \neg t$$

   Then use the DPLL algorithm to figure out which which cities would be theoretically possible to visit during the vacation. Formulate your answer as a sentence in English.

4.3 Self-Assessment

4.3.1 The SAT-Problem

12. [Self-Assessment] Define the *Boolean Satisfiability Problem*?

13. [Self-Assessment] What is the complexity of the SAT-Problem? What does its complexity imply?

4.3.2 The DPLL-Algorithm

14. [Self-Assessment] Explain the basic *DPLL algorithm* for checking satisfiability of propositional formulas in *Conjunctive Normal Form (CNF)*. Give a pseudo-code implementation to illustrate your explanations. For simplicity, you can skip all advanced concepts such as Boolean Constraint Propagation, Pure Literals, and Clause Learning.

15. [Self-Assessment] *SAT solvers* make choices based on *heuristics* on which variable and value to pick for the next decision. (a) Why is the variable order for decisions important for the performance of SAT solvers? (b) Explain a commonly used decision heuristics.

16. [Self-Assessment] Given a formula φ in CNF representation. (a) What is a *partial assignment* of variables? (b) What is a *total assignment* of variables? (c) What does it mean that a clause in *conflicting* with an assignment? (d) What does it mean that a clause in *satisfied* by an assignment?

17. [Self-Assessment] Given an formula φ in CNF representation and an assignment A. Tick the following statements if they are true.

    - ☐ A clause is *satisfied* by A, if A makes a clause true.
    - ☐ If a clause is *conflicting* with an assignment A, if the assignment makes the clause false.
    - ☐ If a clause is *conflicting* with an assignment A, all variables in the clause are given the opposite value in A.
    - ☐ A expression $\varphi[A]$ means that all variables within φ are assigned according to its truth values in A.

18. [Self-Assessment] Within the context of DPLL, explain the terms *decision* and *decision level*.

19. [Self-Assessment] Given the set of clauses $C_\varphi = \{\{a, \neg b\}, \{\neg a, c\}, \{b, \neg c\}, \{\neg a, \neg c\}\}$ and the assignment $A = \{\neg a\}$. Tick the correct statements.

    - ☐ $\varphi[A] = \{\{a, \neg b\}, \{\neg a, c\}, \{\neg a, \neg c\}\}$
    - ☐ $\varphi[A] = \{\{c\}, \{b, \neg c\}, \{\neg c\}\}$
    - ☐ $\varphi[A] = \{\{\neg b\}, \{b, \neg c\}\}$
    - ☐ $\varphi[A] = \{\{\neg b\}, \{c\}, \{b, \neg c\}, \{\neg c\}\}$

20. [Self-Assessment] In the context of the DPLL algorithm, what does a conflict that arises at decision level 0 imply about the satisfiability or unsatisfiability of a formula? Explain your answer.

21. [Self-Assessment] Use the DPLL algorithm (*without* BCP, PL and clause learning) to determine whether or not the set of clauses given is satisfiable. Decide variables in alphabetical order starting with the *positive* phase. If the set of clauses resulted in `SAT`, give a satisfying model.

    Clause 1: $(\neg a \vee b \vee \neg c)$

Clause 2: $(a \vee \neg b \vee c)$

Clause 3: $(\neg a \vee \neg b \vee c)$

Clause 4: $(a \vee b \vee \neg c)$

22. [Self-Assessment] Consider the formula φ that consists of the conjunction of the following clauses:

    Clause 1: $(\neg a \vee b)$

    Clause 2: $(\neg a \vee \neg d)$

    Clause 3: $(c \vee \neg b)$

    Clause 4: $(\neg c \vee d)$

    Use the DPLL algorithm (*without* BCP, PL and clause learning) to determine whether or not the set of clauses given is satisfiable. If the set of clauses resulted in `SAT`, give a satisfying model.

    (a) Decide variables in alphabetical order starting with the *positive* phase.

    (b) Decide variables in alphabetical order starting with the *negative* phase.

    (c) What differences can you see between 22a and 22b? Explain in your own words, why for the DPLL algorithm making good decisions is very important.

23. [Self-Assessment] In the context of the DPLL algorithm, explain what *Boolean Constraint Propagation* is. Give an example.

24. [Self-Assessment] Use the DPLL algorithm with *Boolean Constrain Propagation* (*without* PL and clause learning) to determine whether or not the set of clauses given is satisfiable. Decide variables in alphabetical order starting with the *positive* phase. If the set of clauses resulted in `SAT`, give a satisfying model.

    Clause 1: $(\neg d \vee \neg b \vee \neg a)$

    Clause 2: $(\neg e \vee a \vee \neg f)$

    Clause 3: $(\neg a \vee c \vee b)$

    Clause 4: $(f \vee a \vee e)$

    Clause 5: $(d \vee \neg a \vee \neg b)$

    Clause 6: $(\neg a \vee \neg c \vee b)$

25. [Self-Assessment] Why does the DPLL algorithm check for *Boolean Constraint Propagations (BCP)* and *Pure Literals (PL)* before making a decision?

26. [Self-Assessment] Why is the decision level in the DPLL algorithm only incremented after a decision was made but not when the *Pure Literal Rule* or the *Boolean Constrain Propagation Rule* was applied?

27. [Self-Assessment] Use the DPLL algorithm with *Boolean Constrain Propagation* and *Pure Literals* (*without* clause learning) to determine whether or not the set of clauses given is satisfiable. Decide variables in alphabetical order starting with the *positive* phase. If the set of clauses resulted in `SAT`, give a satisfying model.

    Clause 1: $(\neg c \vee d)$

    Clause 2: $(a \vee \neg d \vee \neg e)$

    Clause 3: $(b \vee \neg c)$

    Clause 4: $(c \vee e)$

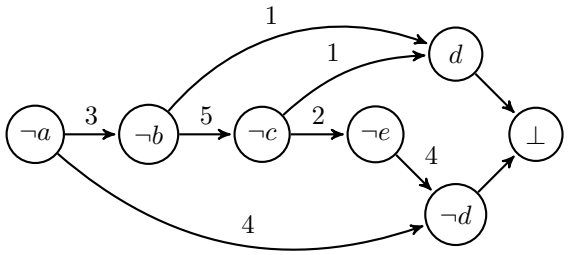
Clause 5: $(\neg b \vee \neg c)$

Clause 6: $(a \vee b)$

28. [Self-Assessment] Explain conflict driven clause learning (CDCL). How do learned clauses prevent the DPLL algorithm of running into already observed conflicts multiple times?

29. [Self-Assessment] In the context of DPLL, give the definition of the *resolution rule* used to construct a resolution proof. Show how the resolution rule derives from the basic natural deduction rules by providing a natural deduction proof.

30. [Self-Assessment] Consider the following conflict graph with the following set of clauses:



Clause 1: $\{b, c, d\}$

Clause 2: $\{c, \neg e\}$

Clause 3: $\{a, \neg b\}$

Clause 4: $\{a, \neg d, e\}$

Clause 5: $\{b, \neg c\}$

State the learned clause by making a resolution proof according to the given conflict graph and given clauses.

31. [Self-Assessment] Consider the formula φ that consists of the conjunction of the following clauses:

Clause 1: $(a \vee b)$

Clause 2: $(\neg b \vee c)$

Clause 3: $(\neg a \vee \neg c)$

Clause 4: $(b \vee c)$

Clause 5: $(a \vee \neg b)$

(a) Use DPLL with learning to show that φ is unsatisfiable. Decide variables in *alphabetic order* and starting with the *positive* phase.

(b) State and briefly explain the *resolution rule*.

(c) Using your results from 31a, give a resolution proof of the unsatisfiability of φ.

32. [Self-Assessment] Use the DPLL algorithm with conflict-driven clause learning to determine whether or not the set of clauses given is satisfiable. Decide variables in alphabetical order starting with the *negative* phase. For conflicts, draw conflict graphs after the end of the table, and add the learned clause to the table.
If the set of clauses resulted in `SAT`, give a satisfying model. If the set of clauses resulted in `UNSAT`, give a resolution proof that shows that the conjunction of the clauses from the table is unsatisfiable.

Clause 1: $\{a, b, \neg c\}$

Clause 2: $\{\neg b, c, d\}$

Clause 3: $\{c, d, \neg e\}$

Clause 4: $\{\neg a, d, \neg e\}$

Clause 5: $\{a, b, \neg d\}$

Clause 6: $\{c, \neg d, e\}$

33. [Self-Assessment] Use the DPLL algorithm with conflict-driven clause learning to determine whether or not the set of clauses given is satisfiable. Decide variables in alphabetical order starting with the *negative* phase. For conflicts, draw conflict graphs after the end of the table, and add the learned clause to the table.
If the set of clauses resulted in `SAT`, give a satisfying model. If the set of clauses resulted in `UNSAT`, give a resolution proof that shows that the conjunction of the clauses from the table is unsatisfiable.

Clause 1: $\{\neg a, \neg b\}$

Clause 2: $\{a, c, e\}$

Clause 3: $\{b, \neg d\}$

Clause 4: $\{\neg c, d, e\}$

Clause 5: $\{\neg d, e\}$

Clause 6: $\{\neg a, b\}$

Clause 7: $\{a, d, \neg e\}$

34. [Self-Assessment] Use the DPLL algorithm with conflict-driven clause learning to determine whether or not the set of clauses given is satisfiable. Decide variables in alphabetical order starting with the *negative* phase. For conflicts, draw conflict graphs after the end of the table, and add the learned clause to the table.
If the set of clauses resulted in `SAT`, give a satisfying model. If the set of clauses resulted in `UNSAT`, give a resolution proof that shows that the conjunction of the clauses from the table is unsatisfiable.

Clause 1: $\{a, \neg c\}$

Clause 2: $\{b, c, e\}$

Clause 3: $\{b, \neg e\}$

Clause 4: $\{\neg a, c\}$

Clause 5: $\{d, e\}$

Clause 6: $\{b, \neg d\}$

Clause 7: $\{\neg d, \neg e\}$

Clause 8: $\{a, c\}$

35. [Self-Assessment] Use the DPLL algorithm with conflict-driven clause learning to determine whether or not the set of clauses given is satisfiable. Decide variables in alphabetical order starting with the *negative* phase. For conflicts, draw conflict graphs after the end of the table, and add the learned clause to the table.
If the set of clauses resulted in `SAT`, give a satisfying model. If the set of clauses resulted in `UNSAT`, give a resolution proof that shows that the conjunction of the clauses from the table is unsatisfiable.

Clause 1: $\{a, b\}$

Clause 2: $\{\neg a, c\}$

Clause 3: $\{a, \neg d\}$

Clause 4: $\{\neg b, c\}$

Clause 5: $\{\neg c, d\}$

Clause 6: $\{\neg c, e\}$

Clause 7: $\{d, \neg e\}$

36. [Self-Assessment] Use the DPLL algorithm with conflict-driven clause learning to determine whether or not the set of clauses given is satisfiable. Decide variables in alphabetical order starting with the *negative* phase. For conflicts, draw conflict graphs after the end of the table, and add the learned clause to the table.
If the set of clauses resulted in `SAT`, give a satisfying model. If the set of clauses resulted in `UNSAT`, give a resolution proof that shows that the conjunction of the clauses from the table is unsatisfiable.

Clause 1: $\{\neg a, \neg b\}$

Clause 2: $\{a, d, e\}$

Clause 3: $\{b, \neg c\}$

Clause 4: $\{c, \neg d, e\}$

Clause 5: $\{\neg c, e\}$

Clause 6: $\{\neg a, b\}$

Clause 7: $\{a, c, \neg e\}$

37. [Self-Assessment] Use the DPLL algorithm with conflict-driven clause learning to determine whether or not the set of clauses given is satisfiable. Decide variables in alphabetical order starting with the *negative* phase. For conflicts, draw conflict graphs after the end of the table, and add the learned clause to the table.
If the set of clauses resulted in `SAT`, give a satisfying model. If the set of clauses resulted in `UNSAT`, give a resolution proof that shows that the conjunction of the clauses from the table is unsatisfiable.

Clause 1: $\{\neg b, c, d\}$

Clause 2: $\{\neg b, \neg d\}$

Clause 3: $\{a, \neg c\}$

Clause 4: $\{\neg c, e\}$

Clause 5: $\{b, c\}$

Clause 6: $\{\neg a, \neg e\}$

38. [Self-Assessment] Use the DPLL algorithm with conflict-driven clause learning to determine whether or not the set of clauses given is satisfiable. Decide variables in alphabetical order starting with the *negative* phase. For conflicts, draw conflict graphs after the end of the table, and add the learned clause to the table.
If the set of clauses resulted in `SAT`, give a satisfying model. If the set of clauses resulted in `UNSAT`, give a resolution proof that shows that the conjunction of the clauses from the table is unsatisfiable.

Clause 1: $\{b, d\}$

Clause 2: $\{b, c\}$

Clause 3: $\{\neg b, \neg e\}$

Clause 4: $\{\neg a, \neg c\}$

Clause 5: $\{\neg c, \neg d\}$
Clause 6: $\{\neg b, c\}$
Clause 7: $\{a, b\}$
Clause 8: $\{\neg b, d, e\}$

39. [Self-Assessment] Use the DPLL algorithm with conflict-driven clause learning to determine whether or not the set of clauses given is satisfiable. Decide variables in alphabetical order starting with the *negative* phase. For conflicts, draw conflict graphs after the end of the table, and add the learned clause to the table.
If the set of clauses resulted in `SAT`, give a satisfying model. If the set of clauses resulted in `UNSAT`, give a resolution proof that shows that the conjunction of the clauses from the table is unsatisfiable.

Clause 1: $\{\neg b, d, e\}$
Clause 2: $\{b, e\}$
Clause 3: $\{c, d\}$
Clause 4: $\{\neg a, \neg e\}$
Clause 5: $\{a, \neg c, \neg e\}$
Clause 6: $\{c, \neg d\}$
Clause 7: $\{\neg b, \neg d\}$

40. [Self-Assessment] Use the DPLL algorithm with conflict-driven clause learning to determine whether or not the set of clauses given is satisfiable. Decide variables in alphabetical order starting with the *negative* phase. For conflicts, draw conflict graphs after the end of the table, and add the learned clause to the table.
If the set of clauses resulted in `SAT`, give a satisfying model. If the set of clauses resulted in `UNSAT`, give a resolution proof that shows that the conjunction of the clauses from the table is unsatisfiable.

Clause 1: $(a \vee b \vee c)$
Clause 2: $(\neg a \vee b)$
Clause 3: $(\neg b \vee c)$
Clause 4: $(\neg c \vee d)$
Clause 5: $(\neg c \vee e)$
Clause 6: $(\neg d \vee \neg e)$

41. [Self-Assessment] It is Sunday and your fridge is almost empty. You think that you can probably prepare a decent pizza with the little ingredients you have.

    You do have dough. The dough is absolutely necessary for your pizza. You also have arugula, bell pepper and eggplant. You want to put at least one of those three ingredients as toppings on your pizza. Cheese is necessary for the pizza too. You have cheddar and feta. You can use one or both kinds of cheese. You don't like the combination of feta and bell pepper, so you can put at most one of those two ingredients on your pizza. Furthermore you need to save some veggies for dinner, so you can only use either the bell pepper or the eggplant for your pizza.

    Create a CNF from this description. You can use the following rule to make the formula shorter:

    $$(\neg s \wedge t) \vee (s \wedge \neg t) \quad \vdash \quad \neg s \vee \neg t$$

    Then use a DPLL to figure out which ingredients you should use for your pizza and which ingredients you shouldn't use. Formulate your answer as a sentence.

42. [Self-Assessment] Your little cousin needs help to plan her birthday party. There are five kids she thinks about inviting, but not all of them get along. Here is what she tells you:

    My very best friend is Anthony, I have to invite him! I'm also good friends with Daisy and Connie, I want at least one of them to come. But Daisy does not like Benjamin, I can't invite them both! But I do like Benjamin, and I also like Emily. I'd want one of them to be there, or both of them. But Emily is always fighting with Daisy, so only one of them can come.

    Create a CNF from this description. You can use the following rule to make the formula shorter:

    $$(\neg s \wedge t) \vee (s \wedge \neg t) \quad \vdash \quad \neg s \vee \neg t$$

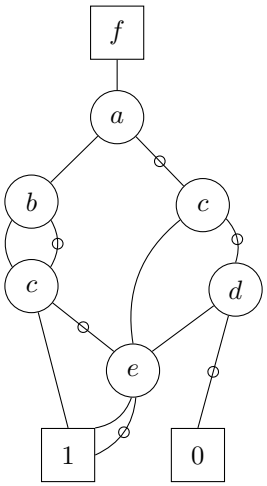
    Then use the DPLL algorithm to figure out which kids your cousin should invite to her birthday party, which kids she should not invite and which kids she can invite without upsetting any other invited guests. Formulate your answer as a sentence.

5 Binary Decision Diagrams

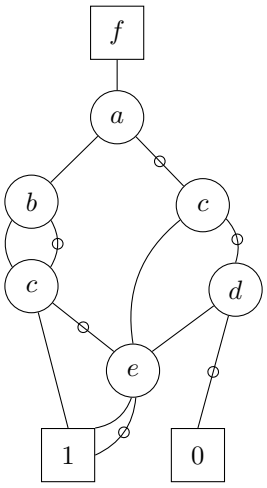
5.1 Lecture

5.1.1 Binary Decision Diagram

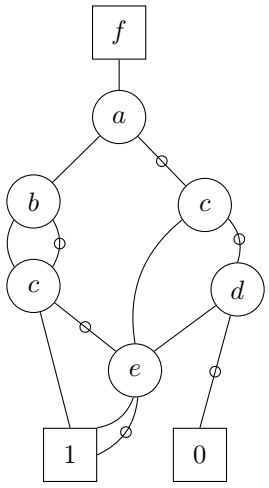
1. [Lecture] Given the *Binary Decision Diagram (BDD)* below, label and explain the different elements of the diagram.



2. [Lecture] Given the *Binary Decision Diagram (BDD)* below. (a) Find a satisfying model $\mathcal{M}_1$, i.e., $\mathcal{M}_1 \models f$. (b) Find a falsifying model $\mathcal{M}_2$, i.e., $\mathcal{M}_2 \not\models f$.



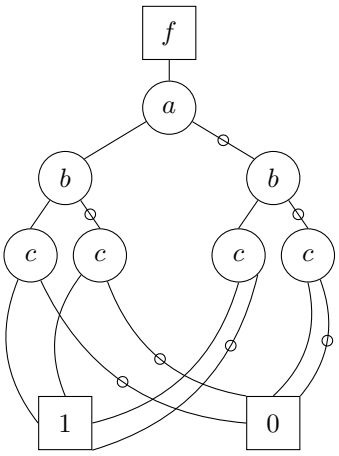
3. [Lecture] Given the *Binary Decision Diagram (BDD)* below. Construct the formula f in disjunctive normal form (DNF) that is represented by the BDD.



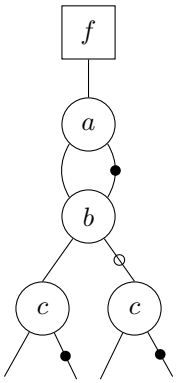
5.1.2 Reduced Ordered BDDs

In the following examples, we have the following convention: Else-edges are marked with circles. Filled circles represent the *complemented* attribute. Dangling edges are assumed to point to the constant node `true`.

4. [Lecture] Transform the given Binary Decision Diagram (BDD) into a reduced ordered BDD (ROBDD) using the variable order $a < b < c$.

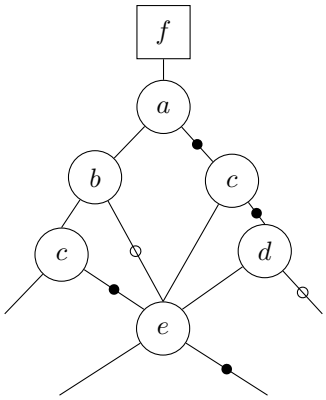


5. [Lecture] Transform the given Binary Decision Diagram (BDD) into a reduced ordered BDD (ROBDD) using the variable order $a < b < c$.



6. [Lecture] A *Reduced and Ordered Binary Decision Diagram (ROBDD)* is a canonical representation of a Boolean formula. Explain what this means and why this is the case.

7. [Lecture] Given the *Binary Decision Diagram (BDD)* below. Construct the formula f in disjunctive normal form (DNF) that is represented by the BDD.



5.1.3 Construction of Reduced Ordered BDDs

8. [Lecture] Construct a ROBDD for the formula

$$f = (a \wedge b) \vee \neg a \vee (c \leftrightarrow d)$$

using *alphabetic variable order*. Use complemented edges and a node for `true` as the only constant node. To simplify drawing, you may assume that *dangling edges* point to the constant node. Write down all cofactors that you compute to obtain the final result and mark them in the graph.

9. [Lecture] Construct a ROBDD for the formula

$$f = (r \wedge p) \vee (\neg r \wedge \neg p) \vee (s \wedge \neg r) \vee (\neg s \wedge r) \vee (\neg r \wedge q),$$

using *variable order* $p < q < r < s$. Use complemented edges and a node for `true` as the only constant node. To simplify drawing, you may assume that *dangling edges* point to the constant node. Write down all cofactors that you compute to obtain the final result and mark them in the graph.

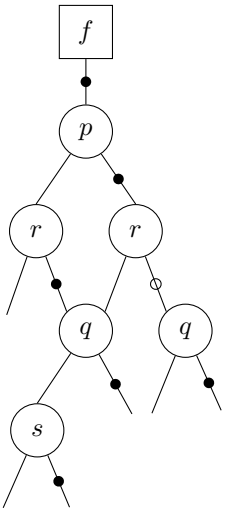
10. [Lecture] Construct a ROBDD for the formula

$$f = (r \wedge \neg p) \vee (\neg r \wedge p) \vee (s \wedge \neg r) \vee (\neg s \wedge r) \vee (r \wedge q),$$

using *variable order* $p < q < r < s$. Use complemented edges and a node for `true` as the only constant node. To simplify drawing, you may assume that *dangling edges* point to the constant node. Write down all cofactors that you compute to obtain the final result and mark them in the graph.

5.2 Practicals

1. [Practicals] **[2 Points]**



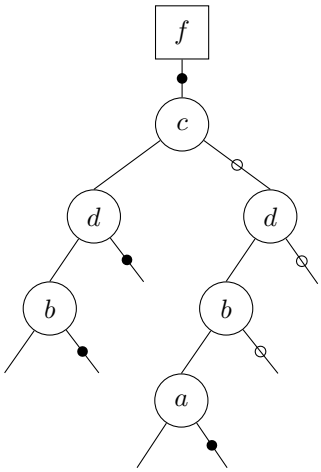
   (a) Use the BDD shown in the figure on the right to check if the formula it represents evaluates to `true` or `false` with the following variable assignments.

      i. $\mathcal{M}_1 : \; p = \top, r = \bot, q = \top, s = \bot$

      ii. $\mathcal{M}_2 : \; p = \bot, r = \bot, q = \bot, s = \top$

   (b) Find the formula f that is represented by the BDD.

2. [Practicals] **[2 Points]**



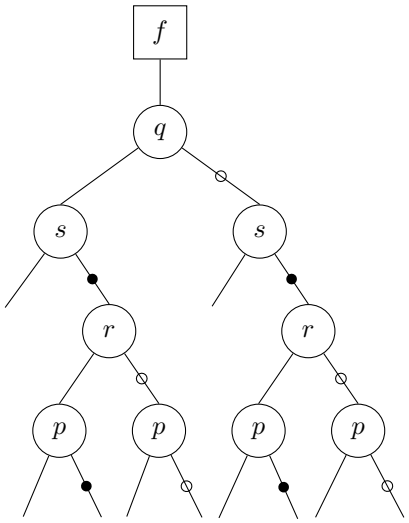
   (a) Use the BDD shown in the figure on the right to check if the formula it represents evaluates to `true` or `false` with the following variable assignments.

      i. $\mathcal{M}_1 : \; a = \bot, b = \top, c = \bot, d = \top$

      ii. $\mathcal{M}_2 : \; a = \top, b = \top, c = \top, d = \top$

   (b) Find the formula f that is represented by the BDD.

3. [Practicals] **[2 Points]** Convert the following BDD into a *reduced ordered* BDD.



4. [Practicals] **[3 Points]** Construct a ROBDD for the formula

$$f = (a \wedge d \wedge c) \vee (b \wedge \neg d \wedge \neg a) \vee (c \rightarrow \neg d) \vee (a \rightarrow \neg b)$$

using *variable order* $b < a < d < c$. Use complemented edges and a node for `true` as the only constant node. To simplify drawing, you may assume that *dangling edges* point to the constant node. Write down all cofactors that you compute to obtain the final result and mark them in the graph.

5. [Practicals] **[3.5 Points]** Construct a reduced ordered binary decision diagram (ROBDD) for the formula

$$f = (p \oplus q) \wedge \neg r$$

using *variable order* $p < q < r$. Use complemented edges and a node for `true` as the only constant node. To simplify drawing, you may assume that *dangling edges* point to the constant node. Write down all cofactors that you compute to obtain the final result and mark them in the graph.

6. [Practicals] **[3.5 Points]** Construct a ROBDD for the formula

$$f = (p \leftrightarrow q) \wedge (r \leftrightarrow s)$$

using *variable order* $r < s < p < q$. Use complemented edges and a node for `true` as the only constant node. To simplify drawing, you may assume that *dangling edges* point to the constant node. Write down all cofactors that you compute to obtain the final result and mark them in the graph.

7. [Practicals] **[4 Points]**

   (a) Construct a Reduced Ordered Binary Decision Diagram (ROBDD) for the formula

$$f = (a \vee b \vee c) \wedge \neg d$$

   using *variable order* $c < a < d < b$. Use complemented edges and a node for `true` as the only constant node. To simplify drawing, you may assume that *dangling edges*

point to the constant node. Write down all cofactors that you compute to obtain the final result and mark them in the graph.

(b) Construct a Reduced Ordered Binary Decision Diagram (ROBDD) for f with a different variable order. The ROBDD should result in a *smaller* ROBDD, w.r.t. the number of nodes.

5.3 Self-Assessment

5.3.1 Binary Decision Diagram

11. [Self-Assessment]

    Give the definition of a *Directed Acrylic Graph (DAG)*. Is the relation between a Binary Decision Diagrams (BDDs) and DAGs?

12. [Self-Assessment]

    - Give the definition of a Binary Decision Diagram (BDD).
    - Draw an example and label and explain the different elements of the diagram.
    - Explain the underlying structure of a BDD and explain, why BDDs are not trees.

5.3.2 Reduced Ordered BDDs

13. [Self-Assessment] In the following list tick all items which can be part of a *Reduced Ordered Binary Decision Diagram (ROBDD)*.

    ☐ Function nodes
    ☐ (Complemented) edges
    ☐ Self-Loops
    ☐ Constant ”true”-node
    ☐ Variable pointers

14. [Self-Assessment] Assume that you have already constructed a *Reduced Ordered Binary Decision Diagram (ROBDD)* for a given formula and variable order. What can happen, if you change the variable order and you draw the ROBDD for the same formula with the new order again?

15. [Self-Assessment] What is the worst-case size of a *Reduced Ordered Binary Decision Diagrams (ROBDDs)* with respect to the formula that it represents. What is the advantage of using a ROBDD to represent a formula compared to using a truth table?

16. [Self-Assessment] How many nodes does a *Reduced Ordered Binary Decision Diagrams (ROBDDs)* for a Boolean formula with n variables have, in worst-case?

    ☐ $2n$
    ☐ $\mathcal{O}(n^2)$
    ☐ $\mathcal{O}(2^n)$
    ☐ $2^{n+1} - 1$
    ☐ n^2
    ☐ infinitely many

17. [Self-Assessment] What is the worst-case size of a *Reduced Ordered Binary Decision Diagrams (ROBDDs)* with respect to the formula that it represents. What is the advantage of using a ROBDD to represent a formula compared to using a truth table?

18. [Self-Assessment] Tick all properties that apply to a *Reduced Ordered Binary Decision Diagram (ROBDD)*.

    ☐ A ROBDD is a canonical representation of its respective formula, for any fixed variable order.

    ☐ Since it is reduced, the number of nodes in the ROBDD does not exceed $2n^2$, where n is the number of variables.

    ☐ The graph of an ROBDD may contain cycles.

    ☐ A ROBDD represents a Boolean formula as directed acyclic graph (DAG).

    ☐ Every node with two regular outgoing edges has two distinct child nodes.

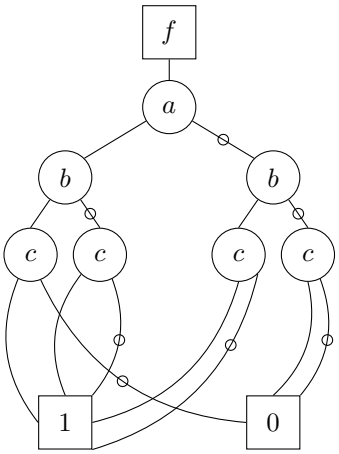
    ☐ No two nodes in an ROBDD represent the same formula.

19. [Self-Assessment] Using BDDs, how can you perform a negation of a formula in constant time?

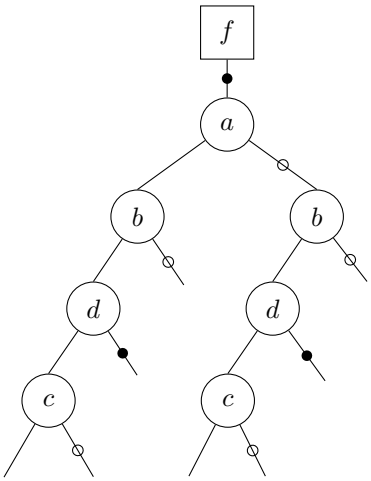
20. [Self-Assessment] Given a *Reduced and Ordered Binary Decision Diagram (ROBDD)*. Explain how you can find the propositional logic formula f that is represented by a given ROBDD?

21. [Self-Assessment] In the context of *Binary Decision Diagrams (BDDs)*, what are redundant nodes? Explain them in a few words and give an example of such a redundancy.

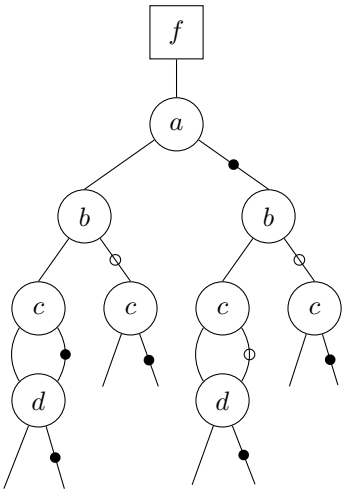
22. [Self-Assessment] Given the *Binary Decision Diagram (BDD)* below. Transform the BDD into a *Reduced Ordered Binary Decision Diagram (ROBDD)*.



23. [Self-Assessment] Given the *Binary Decision Diagram (BDD)* below. Transform the BDD into a *Reduced Ordered Binary Decision Diagram (ROBDD)*.



24. [Self-Assessment] Given the *Binary Decision Diagram (BDD)* below. Transform the BDD into a *Reduced Ordered Binary Decision Diagram (ROBDD)*.



25. [Self-Assessment] In the context of *Binary Decision Diagrams (BDDs)*, how does the variable order impact the BDD?

26. [Self-Assessment] Tick all properties that apply to a *reduced* and *ordered* BDD (ROBDD).

    ☐ If the *else*-edge of a node is complemented, it may point to the same child node as the *then*-edge.

    ☐ The size of a BDD is independent on the variable order.

    ☐ Logic operations, such as conjunction or disjunction, can be performed in polynomial time.

    ☐ The function of the *then*-edge and the *else*-edge of a terminal node is always true.

27. [Self-Assessment] Tick all properties that apply to a *reduced* and *ordered* BDD (ROBDD).

    ☐ Checks for entailment can be done in constant time.

    ☐ Using complemented edges, negation can be performed in constant time.

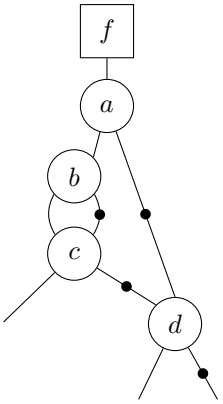
    ☐ Some formulas that can be expressed by truth tables cannot be expressed by BDDs.

- ☐ Equivalence checks can be performed in constant time (assuming that the BDDs for the formula to check are already available).
- ☐ The size of a BDD may depend significantly on the variable order, which is hard to optimize.

28. [Self-Assessment] Consider a *Reduced and Ordered Binary Decision Diagram*. Explain the meaning of the terms *reduced* and *ordered* in this context. Moreover, for each of these terms, draw an example of a Binary Decision Diagram that **does <u>not</u> have** the respective property.

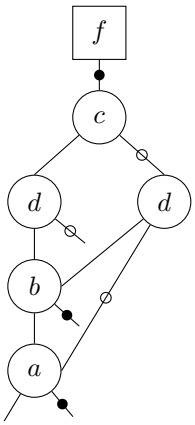
29. [Self-Assessment] Given the *Binary Decision Diagram (BDD)* below. State the formula f that is represented by the BDD.

    Note: Else-edges are marked with circles. Filled circles represent the *complemented* attribute. Dangling edges are assumed to point to the constant node `true`.



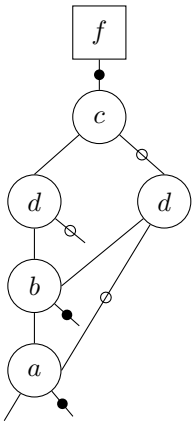
30. [Self-Assessment] Given the *Binary Decision Diagram (BDD)* below. State the formula f that is represented by the BDD.

    Note: Else-edges are marked with circles. Filled circles represent the *complemented* attribute. Dangling edges are assumed to point to the constant node `true`.



31. [Self-Assessment] Given the *Binary Decision Diagram (BDD)* below. State the formula f that is represented by the BDD.

    Note: Else-edges are marked with circles. Filled circles represent the *complemented* attribute. Dangling edges are assumed to point to the constant node `true`.



32. [Self-Assessment] Given the *Binary Decision Diagram (BDD)* below, ...

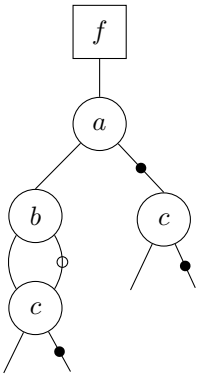
    (a) ... check if the following variable assignments evaluate to `true` or to `false`.

        i. $a = \top, b = \top, c = \bot$

        ii. $a = \bot, b = \bot, c = \bot$

    (b) ... find a propositional formula f, that is represented by the BDD.

    Note: Else-edges are marked with circles. Filled circles represent the *complemented* attribute. Dangling edges are assumed to point to the constant node `true`.



33. [Self-Assessment] Given the *Binary Decision Diagram (BDD)* below, ...

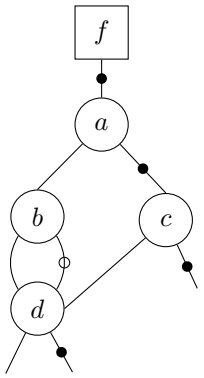
    (a) ... check if the following variable assignments evaluate to `true` or to `false`.

        i. $a = \top, b = \top, c = \bot, d = \bot$

        ii. $a = \bot, b = \bot, c = \top, d = \top$

    (b) ... find a propositional formula f, that is represented by the BDD.

    Note: Else-edges are marked with circles. Filled circles represent the *complemented* attribute. Dangling edges are assumed to point to the constant node `true`.



5.3.3 Construction of Reduced Ordered BDDs

34. [Self-Assessment] What is a cofactor of a formula? Given an example of a propositional logic formula and compute the positive and the negative cofactor for one variable of this formula.

35. [Self-Assessment] Construct a Reduced Ordered Binary Decision Diagram (ROBDD) for the formula

$$f = (\neg a \vee b) \wedge (a \vee b),$$

using *alphabetic variable order*. Use complemented edges and a node for `true` as the only constant node. To simplify drawing, you may assume that *dangling edges* point to the constant node. Write down all cofactors that you compute to obtain the final result and mark them in the graph.

36. [Self-Assessment] Construct a Reduced Ordered Binary Decision Diagram (ROBDD) for the formula

$$f = (\neg x \vee \neg y) \wedge (x \wedge (y \vee z)),$$

using *variable order* $y < z < x$. Use complemented edges and a node for `true` as the only constant node. To simplify drawing, you may assume that *dangling edges* point to the constant node. Write down all cofactors that you compute to obtain the final result and mark them in the graph.

37. [Self-Assessment] Construct a Reduced Ordered Binary Decision Diagram (ROBDD) for the formula

$$f = (\neg x \wedge \neg y) \vee (x \wedge y),$$

using *variable order* $z < x < y$. Use complemented edges and a node for `true` as the only constant node. To simplify drawing, you may assume that *dangling edges* point to the constant node. Write down all cofactors that you compute to obtain the final result and mark them in the graph.

38. [Self-Assessment] Construct a Reduced Ordered Binary Decision Diagram (ROBDD) for the formula

$$f = (\neg p \vee r) \wedge (q \vee \neg p) \wedge (\neg q \vee p)$$

using *variable order* $r < q < p$. Use complemented edges and a node for `true` as the only constant node. To simplify drawing, you may assume that *dangling edges* point to the constant node. Write down all cofactors that you compute to obtain the final result and mark them in the graph.

39. [Self-Assessment] Construct a Reduced Ordered Binary Decision Diagram (ROBDD) for the formula

$$f = (q \wedge \neg s) \vee (s \wedge (\neg r \vee p)) \vee (p \wedge q \wedge r)$$

using *variable order* $p < q < r < s$. Use complemented edges and a node for `true` as the only constant node. To simplify drawing, you may assume that *dangling edges* point to the constant node. Write down all cofactors that you compute to obtain the final result and mark them in the graph.

6 Predicate Logic

6.1 Lecture

6.1.1 Predicates and Quantifiers

1. [Lecture] Model the following declarative sentences with predicate logic, as detailed as possible. Clearly indicate the intended meaning of all function, predicate, and constant symbols that you use.

   (a) Some students like Alice.

   (b) Every teacher likes Bob.

   (c) Some students like every teacher.

   (d) Some students and Bob play a game.

   (e) Not every student plays games.

   (f) Some teachers play no games.

2. [Lecture] Model the following declarative sentences with predicate logic, as detailed as possible. Clearly indicate the intended meaning of all function, predicate, and constant symbols that you use.

   (a) Alice has no sister.

   (b) A person who wears a crown is either a king or a queen.

   (c) Not everybody likes everybody.

   (d) Everybody loves somebody.

3. [Lecture] Model the following declarative sentences with predicate logic, as detailed as possible. Clearly indicate the intended meaning of all function, predicate, and constant symbols that you use.

   (a) The construction side takes a long time, is noisy, and not blocks the sun.

   (b) If there is no school, at least one parent of each kid has to take vacation and cannot got to work.

   (c) All students have to take the exam eventually.

4. [Lecture] Model the following declarative sentences with predicate logic, as detailed as possible. Clearly indicate the intended meaning of all function, predicate, and constant symbols that you use.

   (a) If all kids wear gloves, then all parents will be happy.

   (b) All kids love pizza and spaghetti.

   (c) All kids are fun, energetic, and cannot sit still.

5. [Lecture] Consider the following declarative sentence (known as *Goldbach's Conjecture*):

   "Every even integer greater than 2 is equal to the sum of two prime numbers."

   Model this sentence with predicate logic, as detailed as possible. Clearly indicate the intended meaning of all function, predicate, and constant symbols that you use.

6.1.2 Syntax of Predicate Logic

6. [Lecture] The syntax of predicate logic is defined via 2 types of sorts: *terms* and *formulas*. What are terms and what are formulas? Give examples for both.

7. [Lecture] Give the definition of the syntax of predicate logic. Therefore, give the definition of *terms* and *formulas*.

8. [Lecture] Draw a syntax tree for the following formula:

$$\forall x \Big(\big(P(x,y) \to P(x,x) \big) \vee \big(Q(y,z) \wedge \exists y \; R(x,y,z) \big) \Big)$$

9. [Lecture] Draw the syntax tree for the following formula:

$$\forall x \exists y \; (P(x, f(y)) \wedge Q(y,z) \to R(f(z))).$$

6.1.3 Free and Bound Variables

10. [Lecture] Given the formula

$$P(x,y) \vee \exists y \forall x \; \big(Q(x,y) \wedge R(y,z) \big),$$

construct a syntax tree for φ and determine the *scope* of its quantifiers and which occurrences of the variables are *free* and which are *bound*.

11. [Lecture] Given the formula

$$\varphi = \forall x \exists z \; \big(\neg P(x) \vee Q(y, f(z)) \big) \to \big(\exists x \; P(y) \wedge Q(f(x), z) \big),$$

construct a syntax tree for φ and determine the *scope* of its quantifiers and which occurrences of the variables are *free* and which are *bound*.

6.1.4 Semantics of Predicate Logic

12. [Lecture] Give a model $\mathcal{M}$ for the following formula:

$$\varphi := \exists x \forall y P(x,y).$$

13. [Lecture] Consider the formula

$$\varphi := \forall x \exists y (P(x,y) \wedge Q(x)).$$

Give a model the satisfies the formula and a second one that falsifies the formula.

Show using the parse tree why your models satisfy are falsify the formula.

14. [Lecture] Consider the formula

$$\varphi = \exists x \forall y \; \big(P(x,y) \to (Q(x,y) \vee R(x,y)) \big).$$

Does the following model $\mathcal{M}$ satisfy the formula?
$\mathcal{A} = \{a, b\}$
$P^{\mathcal{M}} = \{(a,a), (a,b)\}$
$Q^{\mathcal{M}} = \{(a,a), (b,a)\}$
$R^{\mathcal{M}} = \{(a,a), (b,b)\}$

15. [Lecture] Give the definition of a model in predicate logic. Discuss what needs to be defined in a *model* of a predicate logic formula. Give an example for each data that could be contained in a model.

16. [Lecture] For the following formula in *Predicate Logic*, find a *model* that satisfies the formula and one that does not. Draw a syntax tree and state all free variables while solving this task.

$$\forall x \exists y \left(P(f(y)) \land P(x)\right) \rightarrow Q(f(f(y)))$$

6.2 Self-Assessment

6.2.1 Predicates and Quantifiers

17. [Self-Assessment] Consider the following declarative sentences:

"Every person who has the same parents as John Doe and is different from John Doe himself is a sibling of John Doe."

Model this sentence with predicate logic, as detailed as possible. Clearly indicate the intended meaning of all function, predicate, and constant symbols that you use.
Also, model the same sentence in propositional logic, as detailed as possible. Clearly indicate the intended meaning of each propositional variable you use.

18. [Self-Assessment] Translate the following sentences into predicate logic. Be as precise as possible. Give the meaning of any function and predicate symbols you use.

    (a) Nobody knows everybody.

    (b) All birds can fly, except for penguins and ostrichs.

    (c) Not all birds can fly, but some birds can fly.

    (d) All kids are cute and quite if and only if they are sleeping

19. [Self-Assessment] Translate the following sentences into predicate logic. Be as precise as possible. Give the meaning of any function and predicate symbols you use.

    (a) Every even integer greater than 2 is equal to the sum of two prime numbers.

    (b) Every person who has the same parents as John Doe and is different from John Doe himself is a sibling of John Doe.

20. [Self-Assessment] Consider the following declarative sentence:

"For every natural number it holds that it is prime if and only if there is no smaller natural number, except for 1, that divides it."

Model this sentence with predicate logic, as detailed as possible. Clearly indicate the intended meaning of all function, predicate, and constant symbols that you use.

Also, model the same sentence in propositional logic, as detailed as possible. Clearly indicate the intended meaning of each propositional variable you use.

21. [Self-Assessment]

"For all triangles it holds it is a scalene triangle iff all its sides have different lengths and all its angles have different measure."

Model this sentence with predicate logic, as detailed as possible. Clearly indicate the intended meaning of all function, predicate, and constant symbols that you use.

22. [Self-Assessment]

    "Everyone gets a break once in a while, but the break cannot last forever"

    Model this sentence with predicate logic, as detailed as possible. Clearly indicate the intended meaning of all function, predicate, and constant symbols that you use.

    Also, model the same sentence in propositional logic, as detailed as possible. Clearly indicate the intended meaning of each propositional variable you use.

23. [Self-Assessment] Model the following sentences with predicate logic, as detailed as possible. Clearly indicate the intended meaning of all function, predicate, and constant symbols that you use.

    (a) Every integer is greater or equal to one.

    (b) For any two integers, their sum is smaller than their product

6.2.2 Syntax of Predicate Logic

24. [Self-Assessment] Given is the following formula in predicate logic

$$\varphi = \forall x \exists y \Big(\big(Q(x,y) \land P(x,y)\big) \to \big(R(y,x) \land P(x,y)\big) \Big).$$

    Draw the syntax tree for φ.

25. [Self-Assessment] Given is the following formula in predicate logic

$$\varphi = \exists x \forall y \Big(\big(P(x,y) \to Q(x,y)\big) \lor \big(P(y,x) \to R(x,y)\big) \Big).$$

    Draw the syntax tree for φ.

6.2.3 Free and Bound Variables

26. [Self-Assessment] in the context of defining the syntax of predicate logic:

    (a) What is the scope of a quantifier?

    (b) What is the difference between *free* and *bound* variables?

    Given an example.

27. [Self-Assessment] In the context of *Predicate Logic*, give a definition of *substitution* of variables.

28. [Self-Assessment] What does it mean to *substitute a term t for a variable x in a predicate logic formula?* Which rules to you have to consider when performing substitution? Give an example.

29. [Self-Assessment] Consider the following formula.

$$\varphi := \forall y \big(P(x) \land Q(y)\big) \lor \big(R(y) \land Q(x)\big)$$

    (a) Compute $\varphi[f(x)/x]$.

    (b) Compute $\varphi[f(y)/x]$.

    (c) Compute $\varphi[f(z)/x]$.

30. [Self-Assessment] Consider the following formula.

$$\varphi := \forall y \big(P(x) \land Q(y)\big) \to \exists x \big(R(y) \land Q(x)\big)$$

(a) Compute $\varphi[f(y)/x]$.
(b) Compute $\varphi[f(x)/y]$.
(c) Compute $\varphi[k/z]$.
(d) Compute $\varphi[x/z]$.

31. [Self-Assessment] Given the formula

$$\varphi = \forall x \exists z \left(\neg P(x) \vee Q(y, f(z))\right) \rightarrow \left(\neg \exists x \; P(y) \wedge Q(f(x), z)\right).$$

(a) Compute $\varphi[f(y)/x]$.
(b) Compute $\varphi[f(x)/y]$.
(c) Compute $\varphi[k/z]$.
(d) Compute $\varphi[x/z]$.

6.2.4 Semantics of Predicate Logic

32. [Self-Assessment] In the following list, tick all items that are required for a complete model of a formula φ in predicate logic.

- ☐ A non-empty, possibly infinite set of values for variables and functions.
- ☐ A concrete value for every bound variable in φ.
- ☐ A concrete value for free bound variable in φ.
- ☐ A definition for each predicate in φ, detailing for which values/tuples the predicate returns *true*.
- ☐ A definition for each function in φ, detailing for which values/tuples the predicate returns *true*.

33. [Self-Assessment] (a) Define a model for a formula in propositional logic?
(b) Define a model for a formula in predicate logic?
For both, state all components that the model needs to define.

34. [Self-Assessment] Given is the following formula in predicate logic

$$\varphi = \forall x \exists y \Big(\big(Q(x, y) \wedge P(x, y)\big) \rightarrow \big(R(y, x) \wedge P(x, y)\big) \Big)$$

and the model $\mathcal{M}$:

- $\mathcal{A} = \{a, b\}$
- $P^{\mathcal{M}} = \{(m, a) | m \in \mathcal{A}\}$
- $Q^{\mathcal{M}} = \{(b, m) | m \in \mathcal{A}\}$
- $R^{\mathcal{M}} = \{(a, b), (b, a), (b, b)\}$

Does the model $\mathcal{M}$ satisfy the formula φ? Explain your answer by drawing a **syntax tree** and evaluate the model $\mathcal{M}$ with the help of this syntax tree.

35. [Self-Assessment] Given is the following formula in predicate logic

$$\varphi = \exists x \forall y \Big(\big(P(x, y) \rightarrow Q(x, y)\big) \vee \big(P(y, x) \rightarrow R(x, y)\big) \Big)$$

and the model $\mathcal{M}$:

- $\mathcal{A} = \{a, b\}$

- $P^M = \{(a, b), (b, b), (b, a)\}$
- $Q^M = \{(a, b), (a, b)\}$
- $R^M = \{(a, b), (a, b)\}$

Does the model $\mathcal{M}$ satisfy the formula φ? Explain your answer by drawing a **syntax tree** and evaluate the model $\mathcal{M}$ with the help of this syntax tree.

36. [Self-Assessment] For the formula below, find one model that satisfies the formula, and one model that does not satisfy the formula. Explain your answer by drawing a **syntax tree** and evaluate the model $\mathcal{M}$ with the help of this syntax tree.

$$(P(x) \land Q(f(x))) \lor (\neg P(x) \land \neg Q(f(x))$$

37. [Self-Assessment] For the formula below, find one model that satisfies the formula, and one model that does not satisfy the formula. Explain your answer by drawing a **syntax tree** and evaluate the model $\mathcal{M}$ with the help of this syntax tree.

$$\neg \forall x((P(x) \to P(y)) \land P(x))$$

38. [Self-Assessment] For the formula below, find one model that satisfies the formula, and one model that does not satisfy the formula. Explain your answer by drawing a **syntax tree** and evaluate the model $\mathcal{M}$ with the help of this syntax tree.

$$\forall x \exists y(P(f(x), y) \land \neg P(x, f(y)))$$

39. [Self-Assessment] For each of the formulas in *Predicate Logic* below, find a *model* that satisfies the formula and one that does not. Draw a syntax tree and state all free variables while solving this task.

(a) $\neg \forall x((P(x) \to P(y)) \land P(x))$

(b) $\forall x \exists y(P(x, y) \land \neg P(f(x), f(y)))$

7 Natural Deduction for Predicate Logic

7.1 Lecture

For each of the following sequents, either provide a natural deduction proof, or a counter-example that proves the sequent invalid.

For proofs, clearly indicate which rule, and what assumptions/premises/
intermediate results you are using in each step. Also clearly indicate the scope of any boxes you use.

For counterexamples, give a complete model. Show that the model satisfies the premise(s) of the sequent in question, but does not satisfy the respective conclusion.

7.1.1 Proof Rules for Universal Quantification

1. [Lecture] $\forall x\ \big(P(x) \to Q(x)\big), \forall x\ P(x) \ \vdash\ \forall x\ Q(x)$.
2. [Lecture] $\forall x\ P(x) \land \forall x\ (P(y) \to Q(x)) \quad \vdash \quad Q(z)$
3. [Lecture] $\forall x\ P(x) \lor \forall x\ Q(x) \quad \vdash \quad \forall y\ (P(y) \lor Q(y))$

7.1.2 Proof Rules for Existential Quantification

4. [Lecture] $\forall x\ (P(x) \to Q(y)), \forall y\ (P(y) \land R(x)) \quad \vdash \quad \exists x\ Q(x))$
5. [Lecture] $\forall a \forall b\ (P(a) \land Q(b)) \quad \vdash \quad \forall a \exists b\ (P(a) \lor Q(b))$
6. [Lecture] Explain the $\exists$-elimination rule ($\exists$e). Why does this rule require a box and what does it mean that x_0 is *fresh*?
7. [Lecture] $\exists x\ \neg P(x), \forall x\ \neg Q(x) \quad \vdash \quad \exists x\ (\neg P(x) \land \neg Q(x))$
8. [Lecture] Consider the following natural deduction proof for the sequent

   $$\forall x\ (P(x) \to Q(x)), \quad \exists x\ P(x) \quad \vdash \quad \forall x Q(x).$$

   Is the proof correct? If not, explain the error in the proof and either show how to correctly prove the sequent, or give a counterexample that proves the sequent invalid.

1.	$\forall x\ (P(x) \to Q(x))$	prem.
2.	$\exists x\ P(x)$	prem.
3.	x_0	
4.	$P(x_0)$	ass.
5.	$P(x_0) \to Q(x_0)$	$\forall$e 1
6.	$Q(x_0)$	$\to$ e, 4,5
7.	$\forall x\ Q(x)$	$\forall$i 4-6
8.	$\forall x\ Q(x)$	$\exists$e 2,3-7

9. [Lecture] $\exists x\ (P(x) \to Q(y)), \quad \forall x\ P(x) \quad \vdash \quad Q(y)$

7.1.3 Quantifier Equivalences

10. [Lecture] $\forall x\ \neg(P(x) \land Q(x)) \quad \vdash \quad \neg\exists x\ (P(x) \land Q(x))$
11. [Lecture] $\neg\exists x\ (P(x) \land Q(x)) \quad \vdash \quad \forall x\ \neg(P(x) \land Q(x))$

7.1.4 Counterexamples

12. [Lecture] $\exists x\ \neg P(x), \exists x\ \neg Q(x) \quad \vdash \quad \exists x\ (\neg P(x) \land \neg Q(x))$
13. [Lecture] $\exists x\ (P(x) \to Q(y)), \quad \exists x\ P(x) \quad \vdash \quad Q(y)$

7.2 Practicals

For each of the following sequents, either provide a natural deduction proof, or a counter-example that proves the sequent invalid.

For proofs, clearly indicate which rule, and what assumptions/premises/
intermediate results you are using in each step. Also clearly indicate the scope of any boxes you use.

For counterexamples, give a complete model. Show that the model satisfies the premise(s) of the sequent in question, but does not satisfy the respective conclusion.

1. [Practicals] **[1 Point]** $(\forall x\ (\neg A(x)) \lor (\exists x\ (B(x)) \quad \vdash \quad \forall x\ (\neg A(x) \lor B(x))$
2. [Practicals] **[1 Point]** $(\forall x\ (\neg A(x)) \lor (\exists x\ (B(x)) \quad \vdash \quad \exists x\ (\neg A(x) \lor B(x))$
3. [Practicals] **[1 Point]** $\exists b\ (a \to B(b)) \quad \vdash \quad a \to \exists b\ B(b)$
4. [Practicals] **[1 Point]** $\exists x\ (S(x) \to T(x)), \neg T(z) \land \neg T(y) \quad \vdash \quad \neg S(y)$
5. [Practicals] **[1 Point]** $\forall r\ U(r) \land \forall r\ (S(r) \to T(r)) \quad \vdash \quad \exists r\ \neg T(x) \to \exists r(\neg S(r) \land U(r))$
6. [Practicals] **[1 Point]** $\exists a\ (P(a) \lor Q(a)), \quad \exists a\ P(a) \to R(c), \quad \exists b\ Q(b) \to R(c) \quad \vdash \quad R(c)$
7. [Practicals] **[1 Point]** $\exists x\ P(x) \to \exists x\ Q(x) \quad \vdash \quad \exists x\ (P(x) \to Q(x))$

7.3 Self-Assessment

For each of the following sequents, either provide a natural deduction proof, or a counter-example that proves the sequent invalid.

For proofs, clearly indicate which rule, and what assumptions/premises/
intermediate results you are using in each step. Also clearly indicate the scope of any boxes you use.

For counterexamples, give a complete model. Show that the model satisfies the premise(s) of the sequent in question, but does not satisfy the respective conclusion.

7.3.1 Proof Rules for Universal Quantification

14. [Self-Assessment] Explain the $\forall$-introduction rule and the $\forall$-elimination rule. Explain why one rule needs a box while the other one does not. What does it mean that x_0 needs to be fresh?
15. [Self-Assessment] $\forall x\ (P(x) \land Q(x)) \quad \vdash \quad \forall x\ ((Q(x) \lor R(x)) \land (R(x) \lor P(x)))$
16. [Self-Assessment] $\forall x\ (P(x) \lor Q(x)), \quad \forall x\ (\neg P(x)) \quad \vdash \quad \forall x\ (Q(x))$

7.3.2 Proof Rules for Existential Quantification

17. [Self-Assessment] $\exists x\ (Q(x) \to R(x)), \quad \exists x\ (P(x) \land Q(x)) \quad \vdash \quad \exists x\ (P(x) \land R(x))$
18. [Self-Assessment] $\forall x\ (Q(x) \to R(x)), \quad \exists x\ (P(x) \land Q(x)) \quad \vdash \quad \exists x\ (P(x) \land R(x))$

7.3.3 Quantifier Equivalences

19. [Self-Assessment] $\neg\exists x\forall y\ (P(x) \land Q(y)) \quad \vdash \quad \forall x\exists y\ \neg(P(x) \land Q(y))$

20. [Self-Assessment] $\forall x\exists y\ \neg(P(x) \land Q(y)) \quad \vdash \quad \neg\exists x\forall y\ (P(x) \land Q(y))$

7.3.4 Counterexamples

21. [Self-Assessment] $\neg\exists x\ \neg P(x) \quad \vdash \quad \forall x\ \neg P(x)$

7.3.5 Mixed Examples

22. [Self-Assessment] $\forall x(P(x) \lor Q(y)), \forall x(P(x) \to R(z)), \forall y(Q(y) \to R(z)) \quad \vdash \quad R(z)$

23. [Self-Assessment] $\exists y\forall x\ (P(x, y)) \quad \vdash \quad \forall x\exists y\ (P(x, y))$

24. [Self-Assessment] $\exists a\forall b\ (S(b, a) \land T(b, a)) \quad \vdash \quad \forall b\forall a\ (S(b, a) \land T(b, a))$

25. [Self-Assessment] $P(y) \to \forall x Q(x),\ \exists x\neg Q(x) \ \vdash \ \exists x\neg P(x)$

26. [Self-Assessment] Consider the following natural deduction proof for the sequent

$$\exists x\ \neg P(x) \quad \vdash \quad \neg\forall x\ P(x).$$

Is the proof correct? If not, explain the error in the proof and either show how to correctly prove the sequent, or give a counterexample that proves the sequent invalid.

1.		$\exists x\ \neg P(x)$	prem.
2.	┌	$\forall x\ P(x)$	ass.
3.	│	$P(x_0)$	$\forall$e 2
4.	│	$\exists x\ P(x)$	$\exists$i 3
5.	└	$\bot$	$\neg$e 1,4
6.		$\neg\forall x\ P(x)$	$\neg$e 2-5

27. [Self-Assessment] Consider the following natural deduction proof for the sequent

$$\exists x\ P(x) \lor \exists x\ Q(x) \quad \vdash \quad \exists x\ (P(x) \lor Q(x)).$$

Is the proof correct? If not, explain the error in the proof and either show how to correctly prove the sequent, or give a counterexample that proves the sequent invalid.

1.			$\exists x\ P(x) \lor \exists x\ Q(x)$	prem.
2.	┌		$\exists x\ P(x)$	ass.
3.	│	x_0 ┌	$P(x_0)$	ass.
4.	│	└	$P(x_0) \lor Q(x_0)$	$\lor$i$_1$ 3
5.	└		$\exists x\ (P(x) \lor Q(x))$	$\exists$e 2,3-4
6.	┌		$\exists x\ Q(x)$	ass.
7.	│	x_0 ┌	$Q(x_0)$	ass.
8.	│	└	$P(x_0) \lor Q(x_0)$	$\lor$i$_2$ 7
9.	└		$\exists x\ (P(x) \lor Q(x))$	$\exists$e 6,7-8
10.			$\exists x\ (P(x) \lor Q(x))$	$\lor$e 1,2-5,6-9

28. [Self-Assessment] $\forall x\exists y\ (P(x) \to Q(y)), P(s) \quad \vdash \quad \exists x\forall y\ (\neg P(x) \lor Q(y))$

8 Transition Systems

8.1 Lecture

8.1.1 Transition Systems

1. [Lecture] Draw the graph for a *transition system* $\mathcal{T}$ with: $S = \{s_1, s_2, s_3, s_4\}$,
   $S_0 = \{s_2\}$,
   $R = \{\{s_1, s_2\}, \{s_1, s_1\}, \{s_2, s_4\}, \{s_2, s_3\}, \{s_3, s_1\}, \{s_4, s_2\}, \{s_4, s_3\}\}$,

2. [Lecture] Consider the example of an elevator. Initially, the elevator is in the ground floor. From the ground floor, it can either go basement, stay there for a while, and then go back to the ground floor, or it can go from the ground floor to the second floor, stay there for a while, and go back to the ground floor. While traveling between ground floor to second floor, the elevator passes the first floor, but it cannot stop there.

   Model this elevator as *transition system*.

8.1.2 Symbolic Encoding

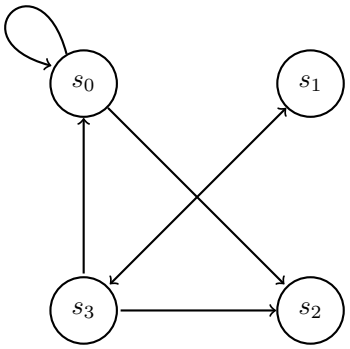
Symbolic Representation of States

3. [Lecture] Given a state space of size $|S| = 2^4 = 16$, give the symbolic encoding for the following states: (a) s_7, (b) s_{15}, and (c) s_{10}.

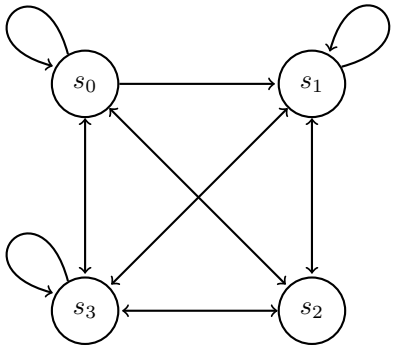
4. [Lecture] Given is the set of states $S = \{s_0, \ldots, s_7\}$. Find formulas in propositional logic that symbolically represent the sets $A = \{s_7, s_6, s_3, s_2\}$, $B = \{s_1, s_3, s_5, s_7\}$, and $C = \{s_7, s_6, s_0, s_1\}$.

Symbolic Representation of the Transition Relation

5. [Lecture] Find a *symbolic encoding* for the *transition relation* of the following *transition system* and simplify your formulas. Use a binary encoding to encode the states, e.g., encode the state s_2 with the formula $v1 \wedge \neg v_0$.



6. [Lecture] Find a *symbolic encoding* for the *transition relation* of the following *transition system* and simplify your formulas. Use a binary encoding to encode the states, e.g., encode the state s_2 with the formula $v1 \wedge \neg v_0$.



Symbolic Encoding and Set Operations of Arbitrary Sets

7. [Lecture] Consider the domain $A = \{Spain, France, Italy, Germany\}$ and the two different symbolic encodings for A given below. Which one gives a shorter symbolic representation for the set $B = \{France, Germany\}$? Illustrate your answer by giving the representing formulas for B in both encodings.

Encoding 1		
Element	v_1	v_0
Spain	0	0
France	1	0
Italy	0	1
Germany	1	1

Encoding 2		
Element	v_1	v_0
Spain	0	0
France	1	0
Italy	1	1
Germany	0	1

8. [Lecture] Find a symbolic binary encoding for $X = \{0, 1, \ldots, 31\}$. Use it to find formulas that symbolically represent the sets A and B and simplify the formulas:

   - $A = \{12, 13, 14, 15, 28, 29, 30, 31\}$
   - $B = \{x \in X \mid 0 \leq x \leq 15\}$

   Furthermore, give the formulas representing the sets $C = A \cap B$ and $D = A \cup B$.

8.2 Self-Assessment

8.2.1 Transition Systems

9. [Self-Assessment] Draw the graph for a *transition system* $\mathcal{T}$ with:

   $S = \{s_0, s_1, s_2\}$,
   $S_0 = \{s_0, s_1\}$,
   $R = \{\{s_0, s_0\}, \{s_0, s_1\}, \{s_0, s_2\}, \{s_1, s_0\}, \{s_1, s_1\}, \{s_1, s_2\}, \{s_2, s_0\}, \{s_2, s_1\}, \{s_2, s_2\}\}$.

10. [Self-Assessment]

    Consider the example of a controller for a lamp.

    Initially the light is off. Pressing the button once turns on the light and the light glows white. From this state, any short-lasting pressure of the button causes the light to switch its color randomly between white, red, green, blue, and yellow. At any state, pressing the button for a longer time turns the light off.

    Model the lamp controller as *transition system*.

8.2.2 Symbolic Encoding

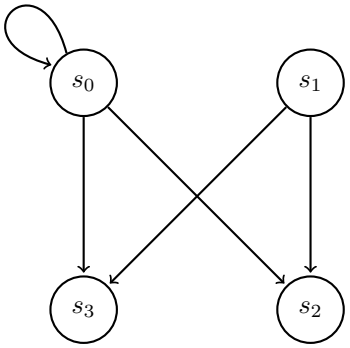
Symbolic Representation of States

11. [Self-Assessment] Given a state space of size $|S| = 2^4 = 16$. Give the symbolic encoding for the following states: (a) s_4, (b) s_9, and (c) s_{13}.

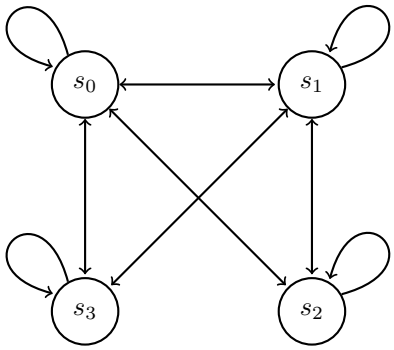
12. [Self-Assessment] Given is the set of states $S = \{s_0, \ldots, s_7\}$. Find formulas in propositional logic that symbolically represent the sets $A = \{s_0, s_2, s_4, s_6\}$, $B = \{s_0, s_1, s_2, s_3\}$, and $C = \{s_7, s_1\}$.

Symbolic Representation of the Transition Relation

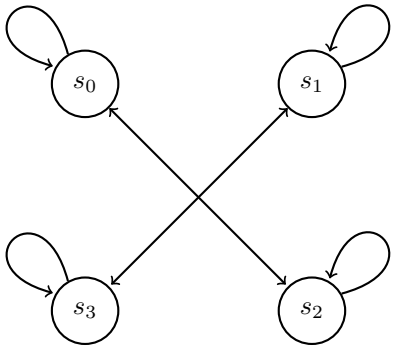
13. [Self-Assessment] Find a *symbolic encoding* for the set of initial states and the *transition relation* of the following *transition system* and simplify your formulas. Use a binary encoding to encode the states, e.g., encode the state s_2 with the formula $v1 \wedge \neg v_0$.



14. [Self-Assessment] Find a *symbolic encoding* for the set of initial states and the *transition relation* of the following *transition system* and simplify your formulas. Use a binary encoding to encode the states, e.g., encode the state s_2 with the formula $v1 \wedge \neg v_0$.



15. [Self-Assessment] Find a *symbolic encoding* for the set of initial states and the *transition relation* of the following *transition system* and simplify your formulas. Use a binary encoding to encode the states, e.g., encode the state s_2 with the formula $v1 \wedge \neg v_0$.



16. [Self-Assessment] Define the *transition system* from the following symbolically encoded transition relations and draw the corresponding graph:

$$\begin{aligned}(v_1 \wedge v_0 \wedge \neg v_1' \wedge \neg v_0') \ \vee \\ (\neg v_1 \wedge v_0 \wedge \neg v_1' \wedge v_0') \ \vee \\ (v_1 \wedge v_0 \wedge v_1' \wedge v_0')\end{aligned}$$

17. [Self-Assessment] Define the *transition system* from the following symbolically encoded transition relations and draw the corresponding graph:

$$\begin{aligned}(\neg v_1 \wedge \neg v_0 \wedge v_1' \wedge v_0') \ \vee \\ (\neg v_1 \wedge v_0 \wedge \neg v_1' \wedge \neg v_0') \ \vee \\ (\neg v_1 \wedge \neg v_0 \wedge \neg v_1' \wedge \neg v_0')\end{aligned}$$

Symbolic Encoding and Set Operations of Arbitrary Sets

18. [Self-Assessment] What is the main advantage of symbolic set operations over non-symbolic operations that enumerate all set elements explicitly?

19. [Self-Assessment] Listed are the participants of a seminar as well as their choice of snacks. Find a symbolic encodings for the participants. For for this encoding, give the symbolic representation of the set B of all participants that ordered *bananas*, and the set C of all participants that ordered cake.

Name	Snack
Alice	banana
Bob	cake
Carl	banana
David	banana
Eve	cake
Frank	cake
Greg	orange
Hank	cake

20. [Self-Assessment] Given a state space of size $|S| = 2048$, find a symbolic binary encoding for this state space and compute the characteristic function for the sets of states

$$B = \{s_0, s_1, s_2, ..., s_{1023}\} \text{ and } C = \{s_{512}, s_{513}, , s_{514}, ..., s_{1535}.\}$$

Then compute the characteristic function for the sets $D = B \cup C$ and $E = B \setminus C$. If possible, simplify the formulas.

21. [Self-Assessment] The following table shows eight students and their means of transportation. Find a symbolic encodings representing the list of students. For this encoding, give the symbolic representation of the set B of all students that go by *bike*, and the set C of all students that go by *car*.

Name	Transportation
Alice	Car
Bob	Bike
Carl	Tram
David	Bike
Eve	Tram
Frank	Bike
Greg	Tram
Hank	Bike

22. [Self-Assessment] Consider the domain $A = \{Spain, France, Italy, Germany\}$ and the two different symbolic encodings for A given below. Which one gives a shorter symbolic representation for the set $B = \{France, Italy\}$? Illustrate your answer by giving the representing formulas for B in both encodings.

Encoding 1		
Element	v_1	v_0
Spain	0	0
France	1	0
Italy	0	1
Germany	1	1

Encoding 2		
Element	v_1	v_0
Spain	0	0
France	1	0
Italy	1	1
Germany	0	1

23. [Self-Assessment] Consider the following set operations and relations between two sets X and Y, and an element a:

    (a) Union: $X \cup Y$
    (b) Intersection: $X \cap Y$
    (c) Set Difference: $X \setminus Y$
    (d) Containment: $a \in X$?
    (e) Subset: $X \subseteq Y$?
    (f) Strict Subset: $X \subset Y$?
    (g) Emptiness: $X = \emptyset$?
    (h) Equality: $X = Y$?

    Let x and y be the symbolic representations of X and Y respectively, and let α be the symbolic encoding of element a. For each of the following items, state which of the above operations is performed, or which of the above questions is answered. Write the letters of the corresponding operation/question into the boxes of the items below. Note that some of the items below do not perform any of the above operations or answer any of the above questions. Put a "–" in the box of these items. Also note that some of the items below might do the same computation or answer the same question.

    ☐ $\neg x \vee y$

    ☐ $x \wedge y$

    ☐ $x \equiv \top$?

- ☐ $x \equiv y$?
- ☐ $(x \to y) \wedge (y \to x)$?
- ☐ $x \equiv \bot$?
- ☐ $y \wedge \neg x$
- ☐ $x \to \bot$?
- ☐ $\alpha \models x$?
- ☐ $\alpha \models \neg x$?
- ☐ $\neg\alpha \models x$?
- ☐ $x \to \alpha$?
- ☐ $y \to x$?
- ☐ $x \to y$?
- ☐ $(x \to y) \wedge (x \not\equiv y)$?

24. [Self-Assessment] Find a symbolic binary encoding for $X = \{0, 1, \ldots, 31\}$. Use it to compute formulas in propositional logic that symbolically represent the following sets.

    - $B = \{4, 5, 12, 13, 20, 21, 28, 29\}$
    - $C = \{1, 2, 13, 14\}$

    Compute the characteristic functions of the following sets by symbolic operations, using your results from before.

    (a) $D = B \cup C$
    (b) $E = X \setminus D$

25. [Self-Assessment] Find a symbolic binary encoding for $X = \{0, 1, \ldots, 31\}$. Use it to compute formulas in propositional logic that symbolically represent the following sets.

    - $B = \{x \in X \mid \text{x is even}\}$
    - $C = \{x \in X \mid \text{x is odd}\}$
    - $D = \{0, 1, 2, 3, 4, 5, 6, 7\}$

    Compute the characteristic functions of the following sets by symbolic operations, using your results from before.

    (a) $E = B \cup D$
    (b) $F = C \cap E$
    (c) $G = E \setminus F$

26. [Self-Assessment] Find a symbolic binary encoding for $X = \{0, 1, \ldots, 31\}$. Use it to compute formulas in propositional logic that symbolically represent the following sets.

    - $B = \{8, 9, 10, 11, 12, 13, 14, 15\}$
    - $C = \{x \in X \mid 0 \leq x \leq 15\}$

Compute the characteristic functions of the following sets by symbolic operations, using your results from before.

(a) $D = B \cup C$

(b) $E = B \cap C$

(c) $F = C \setminus B$

27. [Self-Assessment] Assume you are given the formulas a and b, which symbolically represent the sets A and B, respectively. Give the formula c, which symbolically represents the set $C = A \setminus B$.

28. [Self-Assessment] Assume you are given the formulas a and b, which symbolically represent the sets A and B, respectively. What would you have to check on a, b to test whether or not A is a strict subset of B, i.e., $A \subset B$?

29. [Self-Assessment] Given a state space of size $|S| = 64$. Find a symbolic binary encoding for this state space and compute the formulas that symbolically represent the sets

$$B = \{s_{32}, s_{33}, s_{34}, ..., s_{63}\} \text{ and } C = \{s_{16}, s_{17}, , s_{18}, ..., s_{40}\}.$$

Following, compute the formulas that represent the sets $D = B \cap C$, $E = B \cup C$, $F = B \setminus C$ and $G = C \setminus B$.

30. [Self-Assessment] Given a state space of size $|S| = 64$, find a symbolic binary encoding for this state space and compute the formulas that symbolically represent the sets of states

$$B = \{s_{16}, s_{17}, s_{18}, ..., s_{32}\} \text{ and } C = \{s_{24}, s_{25}, , s_{26}, ..., s_{64}.\}$$

Then compute the formulas that symbolically represent the sets $D = B \cap C$ and $E = B \cup C$.

9 Satisfiability Modulo Theories

9.1 Lecture

9.1.1 Definitions and Notations

1. [Lecture] Give the definition of a theory of formulas in first-order logic.
2. [Lecture] Explain the concept of a theory in first-order logic using the *theory of Linear Integer Arithmetic* $\mathcal{T}_{LIA}$ as example.
3. [Lecture] Explain the problem of *satisfiability modulo theories*. As part of your explanation, explain what a *theory* is and explain the meaning of *theory-satisfiability*.
4. [Lecture] Explain the terms $\mathcal{T}$*-terms*, $\mathcal{T}$*-atoms* and $\mathcal{T}$*-literals* for SMT formulas.
5. [Lecture] What is the difference between a model of an SMT formula and a model of a predicate logic formula without a theory?
6. [Lecture] Given the signature $\Sigma_{EUF} := \{=, a, b, c, d, ..., f, g, h, ..., P, Q, R, ...\}$ of the *Theory of Equality and Uninterpreted Functions* $\mathcal{T}_{EUF}$. State the axioms $\mathcal{A}_{EUF}$ of $\mathcal{T}_{EUF}$.
7. [Lecture] Explain the concepts of *eager encoding* and *lazy encoding* in the context of solving formulas in SMT.

9.1.2 Eager Encoding

8. [Lecture] Explain the concept of *eager encoding* to solve formulas in in SMT. Give the 3 main steps that are performed in algorithms based on eager encoding.
9. [Lecture] Explain the specific translations used in *eager encoding* to decide formulas in the theory of equality and uninterpreted functions.
10. [Lecture] Given the formula

$$\varphi_{EUF} \quad := \quad f(x) = f(y) \ \vee \ (z = y \wedge z \neq f(z))$$

Apply the *Ackermann* reduction algorithm to compute an equisatisfiable formula in $\mathcal{T}_E$.

11. [Lecture] Given the formula

$$\varphi_{EUF} \quad := \quad f(g(x)) = f(y) \ \vee \ (z = g(y) \wedge z \neq f(z))$$

Apply the *Ackermann* reduction algorithm to compute an equisatisfiable formula in $\mathcal{T}_E$.

12. [Lecture] Perform the graph-based reduction on the following formula to compute an equsatisfiable formula in propositional logic.

Given the formula

$$\varphi_{EUF} \quad := \quad f(x, y) = f(y, z) \ \vee \ (z = f(y, z) \wedge f(x, x) \neq f(x, y))$$

Apply the *Ackermann* reduction algorithm to compute an equisatisfiable formula in $\mathcal{T}_E$.

13. [Lecture] Perform graph-based reduction to translate the following formula in $\mathcal{T}_E$ into an equisatisfiable formula in propositional logic.

$$\varphi_E \quad := \quad (a = b \ \vee \ a = d) \rightarrow (b = c \ \wedge \ c \neq d)$$

14. [Lecture] Perform graph-based reduction to translate the following formula in $\mathcal{T}_E$ into an equisatisfiable formula in propositional logic.

$$\varphi_E \quad := \quad (a = b \ \vee \ a = d) \rightarrow (b = c \ \wedge \ c \neq e \wedge \ e \neq d)$$

9.1.3 Lazy Encoding

15. [Lecture] Explain the concept of *Lazy Encoding* to decide satisfiability of formulas in a first-order theory.

16. [Lecture] Consider the following formula in the conjunctive fragment of $\mathcal{T}_{EUF}$.

$$\begin{aligned}\varphi_{EUF} \quad := \quad & x = f(y) \wedge x \neq y \wedge y \neq u \wedge y = f(u) \wedge z \neq f(u) \wedge \\ & u = v \wedge v = z \wedge v = f(y) \wedge v \neq f(z) \wedge f(x) \neq f(z)\end{aligned}$$

Use the *Congruence Closure* algorithm to determine whether this formula is satisfiable.

9.2 Practicals

17. [Practicals] **[3 Points]** Given the formula:

$\varphi_{EUF} := f(x) = y \wedge x = g(x) \vee x \neq f(x) \wedge g(x) = f(g(x)) \vee y \neq g(x) \wedge x = f(y) \wedge g(y) = f(g(x))$

Apply the *Ackermann* reduction algorithm to compute an equisatisfiable formula in $\mathcal{T}_E$.

18. [Practicals] **[3 Points]** Given the formula:

$$\varphi_{EUF} \quad := \quad x = f(x,y) \wedge x \neq y \leftrightarrow z = f(x,y) \vee f(y,z) \neq z \wedge y \neq f(x,y) \vee y = f(x,z)$$

Apply the *Ackermann* reduction algorithm to compute an equisatisfiable formula in $\mathcal{T}_E$.

19. [Practicals] **[3 Points]** Perform graph-based reduction to translate the following formula in $\mathcal{T}_E$ into an equisatisfiable formula in propositional logic.

$$\varphi_E := x \neq y \wedge y = c \vee c = d \rightarrow \neg(d \neq z \vee z = a) \wedge \neg(a = b \wedge x \neq z).$$

20. [Practicals] **[5 Points]** Consider the following formula in $\mathcal{T}_{EUF}$:

$$\varphi_{EUF} \quad := (y = z \vee f(x) = f(y)) \rightarrow (x = z \vee f(x) = x \wedge f(x) = y)$$

Use Ackermann's reduction to compute an equisatisfiable formula in $\mathcal{T}_E$.

Then perform the graph-based reduction on the outcome of Ackermann's reduction to construct an equisatisfiable propositional formula.

21. [Practicals] **[3 Points]** Use the Congruence-Closure algorithm to check if the following assignment for the equalities is satisfiable.

$$\varphi_{EUF} \quad := \quad f(b) = a \wedge e = b \wedge c = f(c) \wedge d \neq f(e) \wedge f(a) = f(d) \wedge a \neq f(c) \wedge d = f(a)$$

22. [Practicals] **[3 Points]** Use the Congruence-Closure algorithm to check if the following assignment for the equalities is satisfiable.

$$\begin{aligned}\varphi_{EUF} \quad := \quad & f(o) = k \wedge l \neq f(m) \wedge n \neq l \wedge f(k) = m \wedge f(o) = f(k) \wedge o \neq k \wedge \\ & l \neq f(n) \wedge f(m) \neq k \wedge m \neq f(m) \wedge o = n \wedge f(m) = o\end{aligned}$$

9.3 Self-Assessment

9.3.1 Definitions and Notations

23. [Self-Assessment] Explain the concept of a theory in first-order logic using the *theory of Linear Real Arithmetic* $\mathcal{T}_{LRA}$ as example.

24. [Self-Assessment] In the following list tick all formulas that are axioms of the theory of equalities and uninterpreted functions $\mathcal{T}_{EUF}$.

    ☐ $\forall x\,(x = x)$
    ☐ $\forall x\,\forall y\,(x = y \vee y = x)$
    ☐ $\forall x\,\forall y\,\forall z\,(x = y \wedge y = z \rightarrow x = z)$
    ☐ $\forall x\,\forall y\,(f(x) = f(y) \rightarrow x = y)$

25. [Self-Assessment] A first-order theory $\mathcal{T}$ is defined by a signature Σ and a set of axioms $\mathcal{A}$. Consider the *Theory of Equality* $\mathcal{T}_E$. Give its signature Σ_E and its axioms $\mathcal{A}_E$.

26. [Self-Assessment] What is an *uninterpreted function*? What is the difference between an uninterpreted and an interpreted function? What are the properties of an uninterpreted function?

27. [Self-Assessment] Provide the answers to the following questions:

    - When is a formula $\mathcal{T}$-valid?
    - When is a formula $\mathcal{T}$-satisfiable?
    - When do we have $\mathcal{T}$-entailment of two formulas?

9.3.2 Eager Encoding

28. [Self-Assessment] In the following list tick all statements that conform to the eager encoding approach for the implementation of SMT solver.

    ☐ Eager encoding is based on the interaction between a SAT solver and a so-called theory solver.
    ☐ Eager encoding involves translating the original formula to an equisatisfiable boolean formula in a single step.
    ☐ Eager encoding is based on the direct encoding of axioms.
    ☐ Eager encoding starts with no constraints at all and adds constraints only when needed.

29. [Self-Assessment]

    - Explain the concept of *Eager Encoding* to decide satisfiability of formulas in a first-order theory.
    - Explain how eager encoding works on the *Theory of Equality* $\mathcal{T}_E$.

30. [Self-Assessment] In the following text fill the blanks with the missing word(s).

    The Ackermann's reduction is used to reduce a formula φ_{in} in _____ _____ to a formula in _____ _____ that is equisatisfiable. Two formulas are equisatisfiable if _____. The algorithm adds explicit constraints to the formula φ_{in} to enforce _____. These constraints say, that $\forall\bar{x}\forall\bar{y}\ (\bigwedge_i x_i = y_i) \rightarrow$ _____). The resulting equisatisfiable formula consists of two parts and is of the form: $\varphi_{out} := \varphi_C \wedge \hat{\varphi_{in}}$. The right part of the formula $\hat{\varphi_{in}}$ describes the flattening original formula in which we replace _____ with _____.

31. [Self-Assessment] For the following $\mathcal{T}_{EUF}$-formula, compute an equivalid formula φ_E in the *Theory of Equality* $\mathcal{T}_E$, by applying *Ackermann's Reduction*.

$$\varphi_{EUF} \quad := \quad f(x,y) = g(x) \rightarrow [f(g(y),z) = x \vee \neg(g(z) = y)]\,.$$

32. [Self-Assessment] Consider the following formula in $\mathcal{T}_{EUF}$.

$$\varphi_{EUF} \quad := \quad f(g(x),h(y)) = a \ \vee \ b = f(u,v) \ \rightarrow \ k(a,b) = u \wedge v = k(x,y)$$

Use Ackermann's reduction to compute an equisatisfiable formula in $\mathcal{T}_E$.

33. [Self-Assessment] In the context of *Eager Encoding* within the *Satisfiability Modulo Theories (SMT)*, explain how instances of *reflexivity, symmetry* and *transitivity* are handled within the *Graph-based Reduction*.

34. [Self-Assessment] In the following text fill the blanks with the missing word(s).

The Graph Based Reduction is used to reduce a formula φ_{in} in _____ _____ to a formula in _____ _____ that is equisatisfiable. Two formulas are equisatisfiable if _____ _____. In the first step of the algorithm, we create a Non-Polar Equality Graph and in the next step we make it chordal. The graph is chordal, if _____ _____. We introduce fresh propositional variables for each equation to ensure _____. In order to ensure transitivity, the algorithm adds constraints of the form _____ for all _____ in the graph. The resulting equisatisfiable formula consists of two parts and is of the form: $\varphi_{out} := \varphi_{TC}$ ___ $\hat{\varphi_{in}}$. The right part of the formula $\hat{\varphi_{in}}$ describes the flattening original formula in which we replace _____ with _____.

35. [Self-Assessment] Consider the following formula from $\mathcal{T}_E$.

$$\varphi_{EUF} := \big[f_y = g_x \wedge f_y = y\big] \vee \big[f_y = f_x \wedge y \neq f_{gy}\big] \vee \big[f_x = f_y \wedge f_y = y\big] \vee \big[f_x = f_{gy} \wedge f_y \neq y\big]$$

Use the graph-based algorithm to construct an equisatisfiable propositional formula φ_{prop}.

What would you have to change if you would want to check φ_E for *validity* instead of satisfiability?

36. [Self-Assessment] Consider the following formula from $\mathcal{T}_E$.

$$\varphi_{EUF} \quad := \quad x \neq y \wedge y = g_x \vee g_x = g_y \rightarrow \neg(g_y \neq z \vee z = f_x) \wedge \neg(f_x = f_y \wedge x \neq z)$$

Use the graph-based algorithm to construct an equivalid propositional formula φ_{prop}.

37. [Self-Assessment] Consider the following formula in $\mathcal{T}_{EUF}$.

$$\varphi_{EUF} \quad := \quad f(x) = f(y) \wedge f(y) = y \vee f(g(x)) = f(f(y)) \wedge g(x) = x \tag{7}$$
$$\vee f(x) \neq f(y) \wedge y \neq g(f(y)) \wedge x \neq g(x) \tag{8}$$

Use Ackermann's reduction to compute an equisatisfiable formula in $\mathcal{T}_E$.

Then perform the graph-based reduction on the outcome of Ackermann's reduction to construct an equisatisfiable propositional formula φ_{prop}.

9.3.3 Lazy Encoding

38. [Self-Assessment] In the following list tick all statements that conform to the lazy encoding approach for the implementation of SMT solver.

    ☐ Lazy encoding is based on the interaction between a SAT solver and a so-called theory solver.

    ☐ Lazy encoding involves translating the original formula to an equisatisfiable Boolean formula in a single step.

    ☐ Lazy encoding is based on the direct encoding of axioms.

    ☐ Lazy encoding starts with no constraints at all and adds constraints only when needed.

39. [Self-Assessment] *Satisfiability Modulo Theories (SMT)* solvers can be implemented via lazy encoding or via eager encoding. Give a short explanation about both approaches and point out their main differences.

40. [Self-Assessment] To decide SMT formulas, the lazy approach uses a theory solver in combination with a SAT solver. Explain what a theory solver is. Explain what the inputs and outputs of a theory solver are and how it is used within the lazy encoding approach.

41. [Self-Assessment] In the following text fill the blanks with the missing word(s).

    One way to solve *Satisfiability* _____ *Theories* problems works as follows. First, the propositional skeleton of the formula in question is given to a _____ solver. If this solver returns _____, we terminate with answer _____. In the other case, the solver returns a _____, which is a _____ of theory literals. This can be given to a _____ solver that can decide the _____ fragment of the theory in question. If this solver returns _____, we terminate with answer _____. Otherwise, we add a _____ _____ to the propositional skeleton, to prevent the same _____ from occurring again, and run the _____ solver on the augmented propositional skeleton. This loop is repeated until either the _____ solver returns _____ (in which case the answer is _____), or the _____ solver returns _____ (in which case the answer is _____). This entire procedure is called _____ encoding.

42. [Self-Assessment] In the following list, mark all items that are true for an *eager encoding* procedure for $\mathcal{T}_{UE}$ with **E**, mark all items that are true for a *lazy encoding* procedure with **L**, and mark all items which neither belong to an eager nor a lazy encoding procedure with **N**.

    ☐ Only one call to a propositional SAT solver is required.

    ☐ A propositional formula that is equisatisfiable to the original theory formula is constructed before calling any solver.

    ☐ A propositional SAT solver and a theory solver for the conjunctive fragment of the theory interact with each other.

    ☐ For a theory-inconsistent assignment of literals, a blocking clause is created.

43. [Self-Assessment] Use the Congruence-Closure algorithm to check if the following assignment for the equalities is satisfiable.

    $\varphi_{EUF} \quad := \quad x = y \wedge y = f(y) \wedge y \neq f(x) \wedge z = f(z) \wedge f(z) = f(x) \wedge z = f(y)$

44. [Self-Assessment] Explain for what the *Congruence Closure* algorithm is used. What are the inputs and outputs of the algorithm? What does the algorithm compute? Explain the individual steps of the algorithm.

45. [Self-Assessment] Consider the following formula in the conjunctive fragment of $\mathcal{T}_{EUF}$.

$$\varphi_{EUF} \quad := \quad f(a) = c \wedge f(c) \neq f(d) \wedge b = f(c) \wedge a \neq f(c) \wedge c = d \wedge b \neq d \wedge a = c$$

Use the *Congruence Closure* algorithm to determine whether this formula is satisfiable.

46. [Self-Assessment] Consider the following formula in the conjunctive fragment of $\mathcal{T}_{EUF}$.

$$\varphi_{EUF} \quad := \quad a = b \wedge c \neq d \wedge f(a) = c \wedge f(b) \neq f(c) \wedge f(a) = f(d) \wedge f(b) = c \wedge f(d) = f(c)$$

Use the *Congruence Closure* algorithm to determine whether this formula is satisfiable.

47. [Self-Assessment] Consider the following formula in the conjunctive fragment of $\mathcal{T}_{EUF}$. Let $a \neq b$ be a shorthand notation for $\neg(a = b)$.

$$f(b) = a \wedge c \neq d \wedge f(e) = b \wedge d \neq f(b) \wedge f(a) = f(e) \wedge$$
$$b \neq f(b) \wedge a \neq e \wedge f(a) = e \wedge a = c \wedge f(b) \neq e \wedge d = f(c)$$

Use the *Congruence Closure* algorithm to determine whether this formula is satisfiable.

10 Temporal Logic

10.1 Lecture

1. [Lecture] Translate the following sentences in computation tree logic $CTL^\star$.
   - In every execution the system gives a grant infinitely often.
   - There exists an execution in which the system sends a request finitely often.
2. [Lecture] Given the following execution word w of a Kripke structure. Evaluate the formula φ on w. Evaluate each sub-formula for any execution step using the provided table.
   - $w = \{\}, \{a\}, \{a\}, \{b\}, \{\}, \{a\}, \{a, b\}^\omega$
   - $\varphi = Xa \vee aUb$

Step	0	1	2	3	4	5	ω
a	0	1	1	0	0	1	1
b	0	0	0	1	0	0	1
Xa							
aUb							
$Xa \vee aUb$							

3. [Lecture] Given the following execution word w of a Kripke structure. Evaluate the formula φ on w. Evaluate each sub-formula for any execution step using the provided table.
   - $w = \{\}, \{a\}, \{\}, \{a, b, c\}, \{a\}, \{a, b\}, (\{a\}, \{a, c\}, \{a, c\})^\omega$
   - $\varphi = Ga \rightarrow (Fb \vee c)$

Step	0	1	2	3	4	5	ω		
a	0	1	0	1	1	1	1	1	1
b	0	0	0	1	0	1	0	0	0
c	0	0	0	1	0	0	0	1	1
Ga									
Fb									
$Fb \vee c$									
$Ga \rightarrow (Fb \vee c)$									

4. [Lecture] Given the following execution word w of a Kripke structure. Evaluate the formula φ on w. Evaluate each sub-formula for any execution step using the provided table.
   - $w = \{\}, \{a\}, \{\}, \{a, b, c\}, \{a\}, \{a, b\}, (\{a\}, \{a, c\}, \{a, c\})^\omega$
   - $\varphi = GFa \rightarrow (FG\neg b \wedge c)$

Step	0	1	2	3	4	5	ω		
a	0	1	0	1	1	1	1	1	1
b	0	0	0	1	0	1	0	0	0
c	0	0	0	1	0	0	0	1	1
GFa									
$FG\neg b$									
$FG\neg b \wedge c$									
$GFa \rightarrow (FG\neg b \wedge c)$									

5. [Lecture] Translate the following sentences in computation tree logic $CTL^\star$.

- For any execution, it always holds that whenever the robot visits region A, it visits region C within the next two steps.
- There exists an execution such that the robot visits region C within the next two steps after visiting region A.

6. [Lecture] Translate the following sentences in computation tree logic $CTL^\star$.
   - The robot can visit region A infinitely often and region C infinitely often
   - Always, the robot visits region A infinitely often and region C infinitely often.
   - If the robot visits region A infinitely often, it should also visit region C finitely often.
7. [Lecture] Given the following Kripke structure $\mathcal{K}$. Does the initial state s_0 of $\mathcal{K}$ satisfy the following formulas?
   - $\varphi_1 \coloneqq EXX(a \wedge b)$
   - $\varphi_2 \coloneqq EXAX(a \wedge b)$

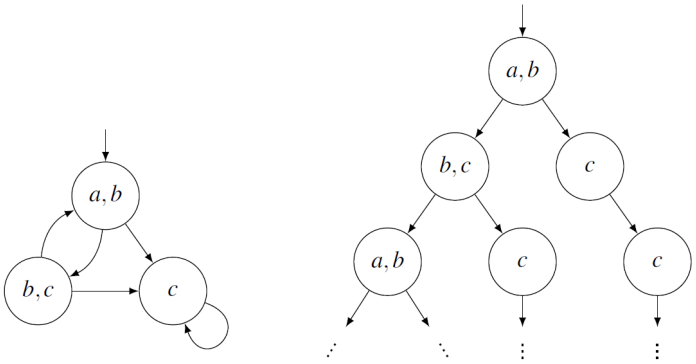


Figure 1: Left: Kripke structure of Example 7, Right: Corresponding computation tree

8. [Lecture] Given the following Kripke structure $\mathcal{K}$. Does the initial state s_0 of $\mathcal{K}$ satisfy the following formulas?
   - $\varphi_1 \coloneqq EXp$
   - $\varphi_2 \coloneqq EG\neg p$

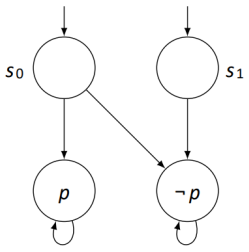


Figure 2: Kripke structure of Example 8

10.2 Self-Assessment

9. [Self-Assessment] Give the definition of a *Kripke structure*. Explain the components of the tuple a Kripke structure consists of. Give an example of a Kripke structure in the representation of a graph.

10. [Self-Assessment] Give the definition of *paths* and *words* of Kripke structures. Give an example in which you draw a graph representing a Kripke structure, and give one possible infinite path and corresponding word.

11. [Self-Assessment] What does a *computation tree* of a Kripke structure represent? Give an example in which you draw a graph representing a Kripke structure, and draw the first 3 levels of the computation tree of this Kripke structure.

12. [Self-Assessment]

    The temporal operators describe properties that hold along a given infinite path ρ through the computation tree of a Kripke structure. Given two formulas φ and ψ describing state properties.

    - Which are the properties that ρ needs to satisfy such that $\rho \vDash G\varphi$?
    - Which are the properties that ρ needs to satisfy such that $\rho \vDash F\varphi$?
    - Which are the properties that ρ needs to satisfy such that $\rho \vDash X\varphi$?
    - Which are the properties that ρ needs to satisfy such that $\rho \vDash \varphi U \psi$?

13. [Self-Assessment] Consider an ordinary traffic junction with incoming lanes from the north, south, east and west. We want to formulate relevant constraints that a traffic light system has to fulfill.

    Give a set of propositional variables that model whether the north and south *or* the east and the west get the

    - green,
    - yellow or
    - red

    light, respectively.

    Formulate the following sentences using $CTL^\star$:

    (a) The north/south lanes will never get the green light at same time as the east/west lanes.
    (b) Whenever the north/south lane receive the green light it will stay green until it changes to yellow.
    (c) When the east/west lane has the red light, it will eventually get the yellow and red light until the light switches to green.

14. [Self-Assessment] Given the following execution word w of a Kripke structure. Evaluate the formula φ on w. Evaluate each sub-formula for any execution step using the provided table.

    - $w = \{\}, \{a\}, \{\}, \{a, b\}, \{a\}, \{a, b\}, (\{a\}, \{a, b\}, \{a\})^\omega$
    - $\varphi = FGa \rightarrow FGb$

Step	0	1	2	3	4	5	ω		
a	0	1	0	1	1	1	1	1	1
b	0	0	0	1	0	1	0	1	0
FGa									
FGb									
$FGa \rightarrow FGb$									

15. [Self-Assessment] Given the following execution word w of a Kripke structure. Evaluate the formula φ on w. Evaluate each sub-formula for any execution step using the provided table.

    - $w = \{a\}, \{a\}, \{a\}, \{b, c\}, \{a\}, \{a, b\}(\{a\}, \{c\})^\omega$
    - $\varphi = aUc \vee Fb$

Step	0	1	2	3	4	5	ω	
a	1	1	1	0	1	1	1	0
b	0	0	0	1	0	1	0	0
c	0	0	0	1	0	0	0	1
aUc								
Fb								
$aUc \vee Fb$								

16. [Self-Assessment] Give the definition of the syntax of the computation tree logic $CTL^\star$. In particular, give the definition of state formulas and path formulas.

17. [Self-Assessment] Give an intuitive explanation of the semantics of computation tree logic $CTL^\star$. Therefore, explain the semantics of the introduced path quantifiers and temporal operators with respect to the computation tree of a Kripke structure.