Questionnaire “Logic and Computability”

Summer Term 2024

Contents

2 SAT Solvers

2.1 The SAT-Problem

2.1.1 Define the *Boolean Satisfiability Problem*?

2.1.2 What is the complexity of the SAT-Problem? What does its complexity imply?

2.2 The DPLL Algorithm

2.2.1 Given a formula φ in CNF. (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?

2.2.2 Let φ be a formula in CNF and A an assignment of variables in φ. Mark the following statements that are true.

- ☐ A clause is *satisfied* by A, if all variables are satisfied under A.
- ☐ If a clause is *conflicting* under an assignment A, if all its variables are not satisfied under A.
- ☐ If a clause is *conflicting* with an assignment A, all variables in the clause are given the opposite value in A.
- ☐ By the expression '$\varphi[A]$' we denote that a variable within φ is satisfied under A.

2.2.3 Given the set of clauses $C_\varphi = \{\{a, \neg b\}, \{\neg a, c\}, \{b, \neg c\}, \{\neg a, \neg c\}\}$ and the assignment $A = \{\neg a\}$. Compute $\varphi[A]$.

2.2.4 Given the set of clauses $C_\varphi = \{\{\neg a, b, c\}, \{a, c\}, \{\neg a, \neg c\}, \{\neg a, b, \neg c\}\}$ and the assignment $A = \{a\}$. Compute $\varphi[A]$.

2.2.5 Given the set of clauses $C_\varphi = \{\{\neg a, b\}, \{a, c\}, \{a, b, \neg c\}, \{a, \neg c\}\}$ and the assignment $A = \{\neg c\}$. Compute $\varphi[A]$.

2.2.6 Given the set of clauses $C_\varphi = \{\{a, \neg b\}, \{\neg a, b, c\}, \{a, b, \neg c\}, \{\neg a, \neg b, \neg c\}\}$ and the assignment $A = \{b\}$. Compute $\varphi[A]$.

2.2.7 In the context of the DPLL algorithm, explain the term *decision heuristic*. Why is it important that a SAT solver implements a good decision heuristic?

2.2.8 In the context of the DPLL algorithm, explain what a *unit clause* is. Give an example.

2.2.9 In the context of the DPLL algorithm, explain what a *pure literal* is. Give an example.

2.2.10 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.

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

2.2.12 *SAT solvers* make choices based on *heuristics* on which variable and value to pick for the next decision. (a) Why is the order in which variables are chosen for decisions important for the performance of SAT solvers? (b) How does a SAT solver implementing the *dynamic largest individual sum* heuristic choose its next assignment?

2.2.13 Within the context of DPLL, explain the terms *decision* and *decision level*.

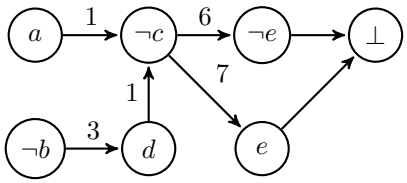
2.2.14 In the context of the DPLL algorithm, what does a conflict arising at decision level 0 imply about the satisfiability or unsatisfiability of a formula? Explain your answer.

2.2.15 In the context of the DPLL algorithm, explain what *Boolean Constraint Propagation* is.

2.2.16 Explain *conflict driven clause learning*. How do learned clauses prevent the DPLL algorithm of running into already observed conflicts multiple times?

2.2.17 In the context of DPLL, give the definition of the *resolution rule* used to perform a resolution proof. How can we use a resolution proof to show that an input formula is unsatisfiable?

2.2.18 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.

2.2.19 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: $(\neg e \vee \neg a)$

2.2.20 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: $(\neg e \vee \neg a)$

2.2.21 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: $(\neg e \vee \neg a)$

2.2.22 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: $\{c, \neg e\}$

2.2.23 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: $(b \vee \neg f)$

2.2.24 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: $(\neg b \vee c \vee d)$

2.2.25 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: $(a \vee b \vee \neg c)$

2.2.26 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: $(\neg a \vee \neg c \vee b)$

2.2.27 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: $(a \vee \neg d \vee \neg e)$

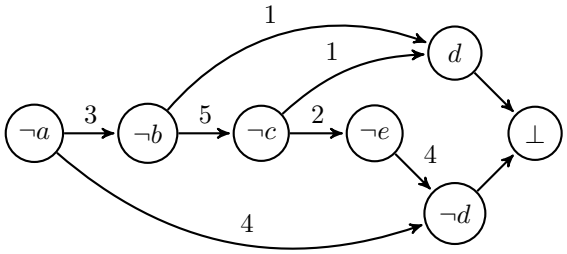
Clause 3: $(b \vee \neg c)$

Clause 4: $(c \vee e)$

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

Clause 6: $(a \vee b)$

2.2.28 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.

2.2.29 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 2.2.29a, give a resolution proof of the unsatisfiability of φ.

2.2.30 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: $\{c, \neg d, e\}$

2.2.31 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: $\{a, d, \neg e\}$

2.2.32 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: $\{a, c\}$

2.2.33 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: $\{d, \neg e\}$

2.2.34 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: $\{a, c, \neg e\}$

2.2.35 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: $\{\neg c, d\}$

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

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

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

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

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

2.2.36 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: $\{\neg b, d, e\}$

2.2.37 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: $\{\neg b, \neg d\}$

2.2.38 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: $(\neg d \vee \neg e)$

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

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

2.2.41 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: $\{\neg a, c\}$

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

2.2.42 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: $\{a, b, c\}$

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

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

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

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

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

2.2.43 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: $\{a, \neg c, \neg e\}$

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: $\{c, d\}$

2.2.44 Use the DPLL algorithm with BCP to determine whether or not the set of clauses given is satisfiable. Decide variables in alphabetical order starting with the *negative* phase.
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: $(\neg x \vee y)$

Clause 2: $(\neg y \vee \neg z)$

Clause 3: $(x \vee \neg w)$

Clause 4: $(w \vee z)$

Clause 5: $(\neg z \vee u)$

Clause 6: $(\neg u \vee \neg x)$

2.2.45 Use the DPLL algorithm with BCP to determine whether or not the set of clauses given is satisfiable. Decide variables in alphabetical order starting with the *negative* phase.
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: (¬b ∨c ∨d)

Clause 3: (a ∨¬c)

Clause 4: (¬d ∨e)

Clause 5: (¬e ∨f)

Clause 6: (¬f ∨g)

Clause 7: (¬g)

2.2.46 Use the DPLL algorithm with no explicit heuristics to determine whether or not the set of clauses given is satisfiable. Decide variables in alphabetical order starting with the *negative* phase.
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 ∨c)

Clause 2: (¬a ∨¬b)

Clause 3: (¬a ∨¬c)

Clause 4: (b ∨¬c)

Clause 5: (¬b ∨c)

2.2.47 Use the DPLL algorithm with no explicit heuristics to determine whether or not the set of clauses given is satisfiable. Decide variables in alphabetical order starting with the *negative* phase.
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: (¬a ∨c)

Clause 3: (¬b ∨¬c)

Clause 4: (c ∨d)

Clause 5: (¬d ∨¬a)

Clause 6: (b ∨¬d)

Clause 7: (¬c ∨¬d)

2.2.48 Use the DPLL algorithm with BCP and PL to determine whether or not the set of clauses given is satisfiable. Decide variables in alphabetical order starting with the *negative* phase.
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: (¬a ∨c)

Clause 3: (d ∨e)

Clause 4: (¬b ∨¬c)

Clause 5: (e ∨¬f)

Clause 6: (¬d ∨¬f)

Clause 7: (d ∨¬f)

Clause 8: (b ∨¬e)

Clause 9: (a ∨¬f)

2.2.49 Use the DPLL algorithm with BCP, PL and CDCL 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: (x ∨y ∨w)

Clause 2: (¬x ∨z)

Clause 3: (¬y ∨w)

Clause 4: (¬z ∨¬w)

Clause 5: (¬y)

Clause 6: (¬x ∨w)

2.2.50 Use the DPLL algorithm with BCP, PL and CDCL 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: (¬a ∨c)

Clause 3: (c ∨d)

Clause 4: (¬b ∨¬c)

Clause 5: (d ∨e)

Clause 6: (¬e ∨f)

Clause 7: (¬d ∨¬f)

Clause 8: (b ∨¬f)

Clause 9: (¬b ∨e)

2.2.51 Use the DPLL algorithm with BCP, PL and CDCL 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: (u ∨v)

Clause 2: (¬u ∨w)

Clause 3: (v ∨¬x)

Clause 4: (w ∨y)

Clause 5: (¬v ∨¬y)

Clause 6: (¬w ∨z)

Clause 7: (¬z ∨x)

Clause 8: (y ∨¬z)

Clause 9: (x ∨¬u)

2.2.52 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: $(a \vee b \vee c)$

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

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

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

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

2.2.53 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 *negative* phase. If the set of clauses resulted in `SAT`, give a satisfying model.

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

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

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

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

2.2.54 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: $(\neg f \vee \neg b \vee \neg a)$

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

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

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

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

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

2.2.55 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: $(\neg c \vee \neg d)$

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

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

Clause 4: $(c \vee b)$

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

Clause 6: $(a \vee b)$

2.2.56 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: $(\neg a \vee \neg b)$

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

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

Clause 4: $(b \vee c)$

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

2.2.57 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: $\{\neg a, b, c\}$

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

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

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

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

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

2.2.58 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: $\{\neg a, \neg c, \neg e\}$

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

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

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

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

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

2.2.59 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: $\{b, e\}$

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

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

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

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

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

2.2.60 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: $\{\neg a, c\}$

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

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

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

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

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

2.2.61 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: $\{\neg b, \neg e\}$

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

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

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

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

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

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

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

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

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

2.2.65 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: $\{a, b, c\}$

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

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

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

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

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

2.2.66 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: $\{\neg c, \neg d\}$

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

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

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

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

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

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

2.2.67 Use the DPLL algorithm with BCP, PL and CDCL 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: (x $\vee$y $\vee$w)

Clause 2: ($\neg$x $\vee$z $\vee\neg$w)

Clause 3: ($\neg$y $\vee$w)

Clause 4: ($\neg$z $\vee\neg$w)

Clause 5: ($\neg$y $\vee$z)

Clause 6: ($\neg$x $\vee$w)

2.2.68 Use the DPLL algorithm with BCP, PL and CDCL 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: (u $\vee\neg$v)

Clause 2: ($\neg$u $\vee$w)

Clause 3: (v $\vee\neg$x)

Clause 4: (w ∨¬y)

Clause 5: (v ∨¬y)

Clause 6: (¬w ∨z)

Clause 7: (z ∨x)

Clause 8: (y ∨¬z)

2.2.69 Use the DPLL algorithm with BCP and PL to determine whether or not the set of clauses given is satisfiable. Decide variables in alphabetical order starting with the *negative* phase.
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: (a ∨c)

Clause 3: (d ∨e)

Clause 4: (¬b ∨¬c)

Clause 5: (e ∨¬f)

Clause 6: (¬d ∨¬f)

Clause 7: (d ∨¬f)

Clause 8: (b ∨e)

Clause 9: (a ∨¬f)

2.2.70 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 *negative* phase. If the set of clauses resulted in `SAT`, give a satisfying model.

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

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

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

Clause 4: $(c \vee d)$

2.2.71 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: $(a \vee \neg b \vee c)$

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

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

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

Clause 5: $(c \vee e)$

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

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

Clause 8: $(\neg c \vee e)$

2.2.72 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: $\{a, d\}$

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

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

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

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

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

Clause 7: $\{a, \neg b\}$

Clause 8: $\{b, \neg d\}$

2.2.73 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: $\{\neg a, \neg d\}$

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

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

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

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

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

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

2.2.74 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: $\{a, \neg c, \neg e\}$

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

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

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

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

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

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

2.2.75 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: $\{b, c, d\}$

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

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

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

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

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

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

Clause 8: $\{\neg b, c, \neg d\}$

2.2.76 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: $\{\neg a, d\}$

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

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

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

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

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

2.2.77 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: $\{\neg a, b, c\}$

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

2.2.78 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: $\{b, \neg d, a\}$

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

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

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

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

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

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

Clause 8: $\{a, \neg b\}$

2.2.79 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: $\{a, b, c\}$

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

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

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

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

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

2.2.80 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: $\{\neg a, d\}$

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

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

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

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

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

2.2.81 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: $\{a, \neg b, c\}$

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

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

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

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

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

2.2.82 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: $\{a, b, c\}$

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

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

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

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

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

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

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

2.2.83 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: $\{\neg b, c\}$

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

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

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

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