Informatics 1 - Computation & Logic: Tutorial 4

Propositional Logic: Resolution [1]

Week 7: 26-30 October 2015

Please attempt the entire worksheet in advance of the tutorial, and bring with you all work, including (if a computer is involved) printouts of code and test results. Tutorials cannot function properly unless you do the work in advance.

You may work with others, but you must understand the work; you can't phone a friend during the exam.

Assessment is formative, meaning that marks from coursework do not contribute to the final mark. But coursework is not optional. If you do not do the coursework you are unlikely to pass the exams.

Attendance at tutorials is **obligatory**; please let your tutor know if you cannot attend.

Resolution

The following propositional logic entailment is valid:

$$(P \wedge Q) \rightarrow R,\ P,\ Q \models R \tag{1}$$

In Tutorials 2 you learned how to prove that an entailment is valid using truth tables. In this assignment we are concerned with another method: *resolution*.

The resolution method proceeds in FOUR steps:

(1) Produce an expression that characterises a counterexample to the entailment

A counterexample to the entailment $X_1, X_2, \ldots, X_n \models Y$ would satify the expression $X_1 \wedge X_2 \wedge \ldots \wedge X_n \wedge \neg Y$. To show that the entailment is valid we must show that there is no counterexample: that is, we must show that this

[1] *This tutorial exercise sheet was originally written by Paolo Besana, and extended by Thomas French, Areti Manataki, Michael Fourman, and Dave Cochran. Send comments to* `Michael.Fourman@ed.ac.uk`

expression is inconsistent (i.e. the negation of the conclusion is inconsistent with the premises). Thus we start by converting the entailment in (1) into the following expression:

$$((P \wedge Q) \rightarrow R) \wedge P \wedge Q \wedge \neg R \tag{2}$$

If we can prove that this expression is inconsistent, then we have proved that the entailment in (1) is valid.

(2) **Convert the expression into conjunctive normal form**

An expression is in conjunctive normal form (CNF) if it is a *conjunction of disjunctions of literals*, where a literal is either an atomic propositional symbol or a negated atomic propositional symbol.
We treat $\top$ as the empty conjunction (of no clauses), and $\bot$ as the disjunction (of no literals). So, a clause is $\bot$, or a literal, or a disjunction of clauses; and a conjunctive normal form is a conjunction of zero or more clauses. We will often call the individual clauses *constraints*.
For example, the following expressions are all in CNF:

$$\neg P$$
$$P \vee \neg Q \vee \neg R$$
$$P \wedge (\neg Q \vee \neg R)$$
$$P \wedge \neg Q \wedge \neg R$$
$$(P \vee \neg Q) \wedge (\neg R \vee P)$$
$$(P \vee Q \vee \neg R) \wedge (\neg Q \vee \neg R) \wedge P \wedge (\neg S \vee \neg P)$$

However the following expressions are *not* in CNF:

$$(P \vee \neg Q) \wedge (\neg R \rightarrow P)$$
$$(P \wedge \neg Q) \vee (\neg R \wedge P)$$
$$\neg P \wedge (\neg\neg Q \vee R)$$
$$(P \vee \neg Q) \wedge \neg(R \vee P)$$

To convert an arbitrary expression of propositional logic into CNF, we apply the following equivalences:

$$\begin{aligned}
\neg(X \wedge Y) &\equiv \neg X \vee \neg Y \\
X \rightarrow Y &\equiv \neg X \vee Y \\
\neg(X \vee Y) &\equiv \neg X \wedge \neg Y \\
X \leftrightarrow Y &\equiv (X \rightarrow Y) \wedge (Y \rightarrow X) \\
X \vee (Y \wedge Z) &\equiv (X \vee Y) \wedge (X \vee Z) \\
X \wedge (Y \vee Z) &\equiv (X \wedge Y) \vee (X \wedge Z) \\
\neg\neg X &\equiv X
\end{aligned}$$

We make liberal use of the associativity of conjunction and disjunction,

$$\begin{aligned} X \wedge (Y \wedge Z) &\equiv (X \wedge Y) \wedge Z \qquad \text{(for which we write } X \wedge Y \wedge Z\text{)} \\ X \vee (Y \vee Z) &\equiv (X \vee Y) \vee Z \qquad \text{(for which we write } X \vee Y \vee Z\text{)} \end{aligned}$$

Thus we can convert the expression in (2) into CNF as follows:

$$\begin{aligned} & \underline{((P \wedge Q) \rightarrow R)} \wedge P \wedge Q \wedge \neg R \\ \Rightarrow \quad & (\underline{\neg(P \wedge Q)} \vee R) \wedge P \wedge Q \wedge \neg R \\ \Rightarrow \quad & (\neg P \vee \neg Q \vee R) \wedge P \wedge Q \wedge \neg R \end{aligned}$$

In other words, the expression in (2) is logically equivalent to the following CNF expression:

$$(\neg P \vee \neg Q \vee R) \wedge P \wedge Q \wedge \neg R \tag{3}$$

You can verify this using a truth table.

(3) **Express the CNF expression in clausal form**

To turn a CNF expression into clausal form, simply turn each conjunct into a *set* of literals, and then convert the whole conjunction into a *set of sets* of literals.

The CNF expression in (3) can be converted into clausal form as follows:

$$\begin{aligned} & (\neg P \vee \neg Q \vee R) \wedge P \wedge Q \wedge \neg R \\ \Leftrightarrow \quad & \{\neg P, \neg Q, R\} \wedge \{P\} \wedge \{Q\} \wedge \{\neg R\} \\ \Leftrightarrow \quad & \big\{\{\neg P, \neg Q, R\}\,,\ \{P\}\,,\ \{Q\}\,,\ \{\neg R\}\big\} \end{aligned}$$

Thus the clausal form of the CNF expression in (3) is the following:

$$\big\{\{\neg P, \neg Q, R\}\,,\ \{P\}\,,\ \{Q\}\,,\ \{\neg R\}\big\} \tag{4}$$

(4) **Apply the resolution rule to the expression in clausal form until no literals are left**

A simple application of the resolution rule is to derive $w \vee x \vee y \vee z$ from $(A \vee w \vee x)$ and $(\neg A \vee y \vee z)$, where w, x, y and z are literals, and A is a logical atom. Recall that for any valuation satisfying $(N \vee w \vee x) \wedge (\neg N \vee y \vee z)$, $w \vee x \vee y \vee z$ is also guaranteed to be satisfied. We express this in clausal form as follows;

$$\frac{\big\{\{N, w, x\}\,, \{\neg N, y, z\}\big\}}{\big\{\{w, x, y, z\}\big\}}$$

In words, two complementary literals, here A and $\neg A$, from the different clauses are removed, and the remaining literals from the two clauses, here w, x, y and z, are merged into a new clause.

More generally, the resolution rule is as follows, where $\mathcal{A}$ and $\mathcal{B}$ are sets of literals:

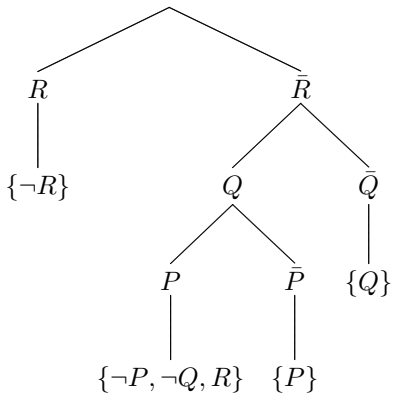
$$\frac{\{X\} \cup \mathcal{A}, \{\neg X\} \cup \mathcal{B}}{\mathcal{A} \cup \mathcal{B}}$$

If we apply the resolution rule to the clausal form in (4), one derivation is:

$$\dfrac{\dfrac{\{\neg R\} \qquad \dfrac{\dfrac{\{\neg P, \neg Q, R\} \quad \{P\}}{\{\neg Q, R\}}\,(P) \qquad \{Q\}}{\{R\}}\,(Q)}{\{\}}\,(R)$$

Note that to show an entailment is valid it is only required to give a *single derivation* of the empty clause. Once we have a derivation of the empty clause, we can construct a refutation tree—since no valuation makes the empty clause true, and if a valuation makes the conclusion of a resolution false then it must make at least one of the assumptions false.

The refutation tree has the same structure as the resolution tree (but inverted):



This shows that for every assignment of truth values to the variables, P, Q, R, at least one of the clauses in our CNF is falsified, so there is no assignment that makes them all true. Each internal node of the tree, except the root, which represents the state where no truth values have been assigned, is labelled with a literal made true (where $\bar{X} = \neg X$), so each path to a leaf represents a truth value assignment to one or more atoms. Each leaf is labelled by a clause falsified by the assignment that leads to it.
Note that, for example, to falsify $\neg R$ we must make the atom R true, so the truth assignments in the refutation tree are dual to the corresponding literals in the resolution tree.

In this example, it does not really matter in which order we choose the different opportunities for resolution. However, for realistic examples different choices may have very different running times.

A simple algorithm for applying the resolution rule is the **Davis-Putnam algorithm**. In this, at each stage some variable is picked, and each clause containing a positive occurrence of that variable is paired with each clause containing a negative occurrence of that variable. So it there are n clauses including the literal A and m clauses including the literal $\neg A$ then we produce $m \times n$ new clauses. Once we have resolved *all* of these pairs we can forget any clauses containing the atom A (or its negation). Any clauses containing both A and $\neg A$ (or any resulting clause containg some complementary pair of literals) can be dropped from a clausal form: since every valuation makes such a clause true, they place no constraints on a satisfying valuation for the clausal form. One heuristic for the Davis-Putnam algorithm, is to first pick variables that occur in few clauses.

Lastly, let us consider a case in which the original inference is invalid, and so a counter-example can be generated:

$$(P \wedge Q) \rightarrow R,\ P,\ Q \models \neg R \tag{5}$$

This, of course, is simply (1) with the conclusion negated. Hence, we try to refute:

$$((P \wedge Q) \rightarrow R) \wedge P \wedge Q \wedge R \tag{6}$$

In CNF:

$$(\neg P \vee \neg Q \vee R) \wedge P \wedge Q \wedge R \tag{7}$$

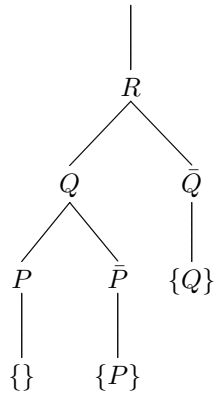
In clausal form:

$$\big\{\{\neg P, \neg Q, R\},\ \{P\},\ \{Q\},\ \{R\}\big\} \tag{8}$$

We may apply resolution as follows

$$\dfrac{\dfrac{\{\neg P, \neg Q, R\} \quad \{P\}}{\{\neg Q, R\}}\ (P) \qquad \{Q\}}{\{R\}}\ (Q)$$

Here we are left with one unresolved literal; hence the refutation fails. Since a partial valuation making the conclusion of a resolution valid can be extended to make both assumptions valid, we can invert the tree to produce a counter-example:



Here, the leftmost branch refutes no conjuncts, hence the whole conjunction can be satisfied with P, Q, and R evaluated to $\top$.

1. Given the following entailments:

   (a) $C \models \neg\neg A \to B$

      i. Convert to an expression:

      $C \wedge \neg(A \to B)$

      ii. Convert to CNF:

      $C \wedge \neg(\neg A \vee B)$
      $C \wedge \neg\neg A \wedge \neg B$
      $C \wedge A \wedge \neg B$

   (b) $A \to B \models A \wedge B$

      i. Convert to an expression:

      $(A \to B) \wedge \neg(A \wedge B)$

      ii. Convert to CNF:

      $(\neg A \vee B) \wedge \neg(A \wedge B)$
      $(\neg A \vee B) \wedge (\neg A \vee \neg B)$

   (c) $\neg(B \vee C) \models A \vee \neg B$

      i. Convert to an expression:

      $\neg(B \vee C) \wedge \neg(A \vee \neg B)$

      ii. Convert to CNF:

      $\neg B \wedge \neg C \wedge \neg(A \vee \neg B)$
      $\neg B \wedge \neg C \wedge \neg A \wedge B$

(d) $\neg(\neg A \vee C), B \to (D \wedge C) \models A \wedge B$

i. Convert to an expression:

$\neg(\neg A \vee C) \wedge (B \to (D \wedge C)) \wedge \neg(A \wedge B)$

ii. Convert to CNF:

$A \wedge \neg C \wedge (B \to (D \wedge C)) \wedge \neg(A \wedge B)$
$A \wedge \neg C \wedge (\neg B \vee (D \wedge C)) \wedge \neg(A \wedge B)$
$A \wedge \neg C \wedge (\neg B \vee D) \wedge (\neg B \vee C) \wedge \neg(A \wedge B)$
$A \wedge \neg C \wedge (\neg B \vee D) \wedge (\neg B \vee C) \wedge (\neg A \vee \neg B)$

(e) $B \vee \neg E, C \leftrightarrow D \models A \to (B \to \neg C)$

i. Convert to an expression:

$(B \vee \neg E) \wedge (C \leftrightarrow D) \wedge \neg(A \to (B \to \neg C))$

ii. Convert to CNF:

$(B \vee \neg E) \wedge (C \leftrightarrow D) \wedge \neg(A \to (B \to \neg C))$
$(B \vee \neg E) \wedge (C \to D) \wedge (D \to C) \wedge \neg(A \to (B \to \neg C))$
$(B \vee \neg E) \wedge (\neg C \vee D) \wedge (\neg D \vee C) \wedge \neg(A \to (B \to \neg C))$
$(B \vee \neg E) \wedge (\neg C \vee D) \wedge (\neg D \vee C) \wedge \neg(\neg A \vee \neg B \vee \neg C)$
$(B \vee \neg E) \wedge (\neg C \vee D) \wedge (\neg D \vee C) \wedge A \wedge B \wedge C$

2. Consder the equivalences listed on pages 2 and 3 for converting formulae into CNF. So far you have encountered them as a set of strategies which you can employ in order to produce CNF from Well-Formed Formula (WFFs); now, we are going to consider them in a computational light.

   (a) Show that for any WFF, it is possible to convert it to CNF using the formulae on page 2.

   See attached sheet at the end of the document.

   (b) Outline an algorithm by which arbitrary WFFs may be converted to CNF. It is not necessary to go into great detail with this; merely specify in what order and under which conditions the different rules should be applied.

   Eliminate arrows;
   Push negations in;
   Push $\vee$ inside $\wedge$.

3. Use resolution to prove whether the following argument is valid:

$$\neg A \to \neg B,\ (\neg B \wedge A) \to D \models D$$

(a) Convert into an expression:

$(\neg A \to \neg B) \wedge ((\neg B \wedge A) \to D) \wedge \neg D$

(b) Convert to CNF:

$(A \vee \neg B) \wedge (\neg(\neg B \wedge A) \vee D) \wedge \neg D$
$(A \vee \neg B) \wedge (B \vee \neg A \vee D) \wedge \neg D$

(c) Convert to clausal form:

$\{\{A,\ \neg B\}, \{B,\ \neg A,\ D\}, \{\neg D\}\}$

(d) Apply the Davis-Putnam algorithm for resolution and state whether the original argument is valid:

$$\frac{\{A,\ \neg B\}, \{B,\ \neg A,\ D\}}{\{\neg B, B, D\} = \top}\ (A)$$

This application of the resolution rule (with $\top$ as conclusion) tells us that we can extend any partial valuation not giving a value for A to one that makes its two assumptions true. If our partial valuation makes B true, then the extension must make A true; if the partial valuation makes B false then our extension must make A false.
This leaves $\neg D$ as the only remaining literal. A satisfying valuation for the original cluases must therefore make D false, and give A and B the same value.
Resolving on B gives a similar result. Resolving first on D eliminates all literals

$$\dfrac{\dfrac{\{A, \neg B\} \quad \dfrac{\{B,\ \neg A,\ D\} \quad \{\neg D\}}{\{B, \neg A\}}\ D}{\top}\ A}$$

This tells us again that we can extend any partial valuation not assigning values to A and D to a valuation satisfying all of the original clauses.

(e) If the entailment is valid, draw the refutation tree derived from your resolution proof. If it is invalid, give a counterexample (a satisfying assignment of truth values).

As per the discussion on the previous page,
$A = \bot$, $B = \bot$, $D = \bot$, *satisfies the clauses,*
as does $A = \top$, $B = \top$, $D = \bot$.

4. Use resolution to prove whether the following argument is valid:

$$\neg F \to \neg P,\ (\neg P \wedge Q) \to R \models \neg F \to (R \wedge \neg Q)$$

(a) Convert to an expression:

$(\neg F \to \neg P) \wedge ((\neg P \wedge Q) \to R) \wedge \neg(\neg F \to (R \wedge \neg Q))$

(b) Convert to CNF:

$(F \vee \neg P) \wedge (\neg(\neg P \wedge Q) \vee R) \wedge \neg(F \vee (R \wedge \neg Q))$
$(F \vee \neg P) \wedge (P \vee \neg Q \vee R) \wedge \neg F \wedge \neg(R \wedge \neg Q)$
$(F \vee \neg P) \wedge (P \vee \neg Q \vee R) \wedge \neg F \wedge (\neg R \vee Q)$

(c) Convert to clausal form:

$\{\{F, \neg P\}, \{P, \neg Q, R\}, \{\neg F\}, \{\neg R, Q\}\}$

(d) Apply the Davis-Putnam algorithm for resolution and state whether the original argument is valid:

$$\dfrac{\dfrac{\dfrac{\{F, \neg P\} \quad \{\neg F\}}{\{\neg P\}}\ (F) \qquad \{P, \neg Q, R\}}{\{\neg Q, R\}}\ (P) \qquad \{\neg R, Q\}}{\{R, \neg R\} = \top}\ (Q)$$

At each step, we have exhaustively resolved all pairs of (hitherto unresolved) clauses containing complementary literals, and have not found the empty clause. This proves that the clauses are satisfiable and the argument is not valid.
We can work up the proof tree to extend any partial valuation not assigning values to F, P, Q to a satisfying valuation. If our initial caluation makes R true, then at successive steps we must make Q true, P false, and F false. If the initial valuation makes R false then we must make Q false, but the other steps remain the same.

(e) If the entailment is valid, draw overleaf the refutation tree derived from your resolution proof. If it is invalid, give a counterexample (a satisfying assignment of truth values).

$F = \bot$, $P = \bot$, $Q = \bot$, $R = \bot$, satisfies the clauses, as does $F = \bot$, $P = \bot$, $Q = \top$, $R = \top$.

5. Use resolution to prove whether the following argument is valid:

$$A \to \neg C,\ (\neg B \vee D) \to A \models (D \wedge \neg B) \to (A \wedge \neg C)$$

(a) Convert to an expression:

$(A \to \neg C) \wedge ((\neg B \vee D) \to A) \wedge \neg ((D \wedge \neg B) \to (A \wedge \neg C))$

(b) Convert to CNF:

$(\neg A \vee \neg C) \wedge (\neg (\neg B \vee D) \vee A) \wedge \neg (\neg (D \wedge \neg B) \vee (A \wedge \neg C))$
$(\neg A \vee \neg C) \wedge ((B \wedge \neg D) \vee A) \wedge \neg (\neg D \vee B \vee (A \wedge \neg C))$
$(\neg A \vee \neg C) \wedge (B \vee A) \wedge (\neg D \vee A) \wedge (D \wedge \neg B \wedge \neg (A \wedge \neg C))$
$(\neg A \vee \neg C) \wedge (B \vee A) \wedge (\neg D \vee A) \wedge D \wedge \neg B \wedge (\neg A \vee C)$

(c) Convert to clausal form:

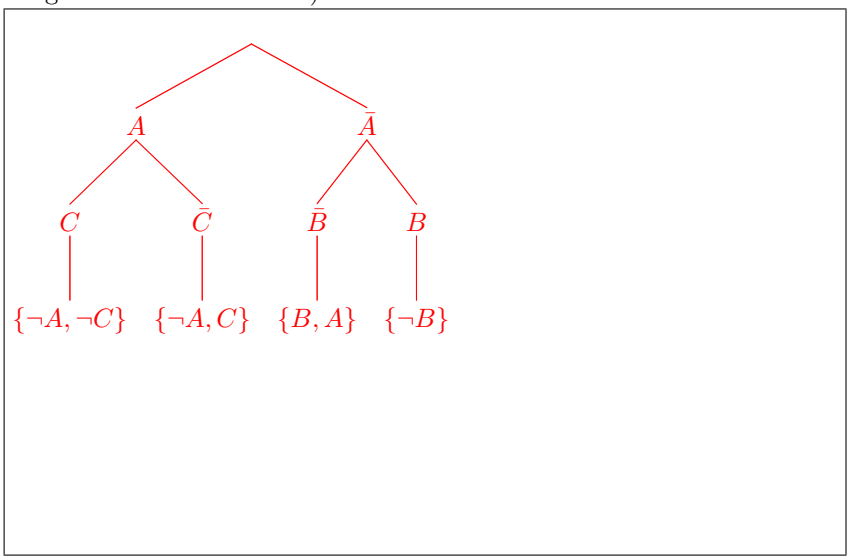
$\{\{\neg A, \neg C\}, \{B, A\}, \{\neg D, A\}, \{D\}, \{\neg B\}, \{\neg A, C\}\}$

(d) Apply the Davis-Putnam algorithm for resolution and state whether the original argument is valid:

$$\dfrac{\dfrac{\{\neg A, \neg C\} \quad \{\neg A, C\}}{\{\neg A\}}\ (C) \qquad \dfrac{\{B, A\} \quad \{\neg B\}}{\{A\}}\ (B)}{\{\}}\ (A)$$

Our clauses are inconsistent.
This proves that the argument is valid.

(e) If the entailment is valid, draw the refutation tree derived from your resolution proof. If it is invalid, give a counterexample (a satisfying assignment of truth values).



6. Use resolution to prove whether the following argument is valid:

$$C \to A, C \to B, \neg D \to C, \neg E \to C \models \neg(A \wedge B \wedge D \wedge E)$$

(a) Convert to an expression:

$(C \to A) \wedge (C \to B) \wedge (\neg D \to C) \wedge (\neg E \to C) \wedge \neg\neg (A \wedge B \wedge D \wedge E)$

(b) Convert to CNF:

$(\neg C \vee A) \wedge (C \to B) \wedge (\neg D \to C) \wedge (\neg E \to C) \wedge \neg\neg (A \wedge B \wedge D \wedge E)$
$(\neg C \vee A) \wedge (\neg C \vee B) \wedge (\neg D \to C) \wedge (\neg E \to C) \wedge \neg\neg (A \wedge B \wedge D \wedge E)$
$(\neg C \vee A) \wedge (\neg C \vee B) \wedge (\neg\neg D \vee C) \wedge (\neg E \to C) \wedge \neg\neg (A \wedge B \wedge D \wedge E)$
$(\neg C \vee A) \wedge (\neg C \vee B) \wedge (D \vee C) \wedge (\neg E \to C) \wedge \neg\neg (A \wedge B \wedge D \wedge E)$
$(\neg C \vee A) \wedge (\neg C \vee B) \wedge (D \vee C) \wedge (\neg\neg E \vee C) \wedge \neg\neg (A \wedge B \wedge D \wedge E)$
$(\neg C \vee A) \wedge (\neg C \vee B) \wedge (D \vee C) \wedge (E \vee C) \wedge \neg\neg (A \wedge B \wedge D \wedge E)$
$(\neg C \vee A) \wedge (\neg C \vee B) \wedge (D \vee C) \wedge (E \vee C) \wedge (A \wedge B \wedge D \wedge E)$
$(\neg C \vee A) \wedge (\neg C \vee B) \wedge (D \vee C) \wedge (E \vee C) \wedge A \wedge B \wedge D \wedge E$

(c) Convert to clausal form:

$\{\{\neg C, A\}, \{\neg C, B\}, \{D, C\}, \{E, C\}, \{A\}, \{B\}, \{D\}, \{E\}\}$

(d) Apply the Davis-Putnam algorithm for resolution and state whether the original argument is valid:

$$\frac{\{\neg C, A\} \quad \{\neg C, B\} \quad \{D, C\} \quad \{E, C\}}{\{A, D\}\,\{A, E\}\,\{B, D\}\,\{B, E\}}\ (C)$$

(Note that we have 2 clauses containing C, 2 containing $\neg C$, thus $2 \times 2 = 4$ resolvents on C).
We have now exhaustively resolved all pairs of clauses containing complementary literals, and have not found the empty clause. This proves that the clauses are satisfiable and the argument is not valid.

(e) If the entailment is valid, draw the refutation tree derived from your resolution proof. If it is invalid, give a counterexample (a satisfying assignment of truth values).

Any partial valuation (omitting C) that will satisfy the clauses $\{A, D\}$. $\{A, E\}$. $\{B, D\}$. $\{B, E\}$. $\{A\}$. $\{B\}$. $\{D\}$. $\{E\}$ can be extended with a suitable value for C to satisfy the original clauses. Since the remaining clauses only include pure literals (no complementary pairs), they can be satisfied by a valuation making every one of these literals true. In this example, we can choose either value for C: thus, $A = \top$, $B = \top$, $C = \top$, $D = \top$, $E = \top$ will satisfy, as will $A = \top$, $B = \top$, $C = \bot$, $D = \top$, $E = \top$.

Answers to question 2.

There is a long and a short version of the answer to 2a: The short version is, any WFF ϕ can be converted into a DNF ψ by creating a disjunction of d many conjunctions of literals, where each disjunct is equivalent to one of the possible valid complete truth-assignments of ϕ, and each the possible valid complete truth-assignments of ϕ are uniquely represented by a single disjunct of ψ. Furthermore, any DNF ψ can be converted into a CNF χ consisting of c many unique clauses of d literals, where, if D_i is the i^{th} disjunct of ψ, $c = \prod_{i-1}^{d} D_i$, such that if literal $X_{i,j}$ is the j^{th} literal of the i^{th} conjunction of ψ, each disjunction in χ is of the form $(X_{1,y_1} \vee X_{2,y_2} \vee ... \vee X_{n,y_n})$, where $0 < y_i \leq |D_i|$. (To see that this is true, consider what must be the case for the CNF to be true, if all but one of the disjuncts of the DNF are true). This shows that any WFF can be converted to CNF, but since the first step in the process given is to exhaustively compute the truth-table of ϕ, making ψ empty if ϕ is invalid, it rather negates the point of doing DPLL. Therefore the long version of the answer, which is also a partial answer to 2b., is to show that arbitrary WFFs can be converted to CNF *using the equivalencies provided*:

- Any WFF including boolean functions other than $\neg$, $\wedge$, and $\vee$ can be converted to include only $\neg$, $\wedge$, and $\vee$ using equivalences such as $X \rightarrow Y \equiv \neg X \vee Y$ and $X \leftrightarrow Y \equiv (X \rightarrow Y) \wedge (Y \rightarrow X)$.
- For any WFF consisting of arbitrary nestings of negation, conjunction and disjunction, negations can be moved towards the leaves with repeated applications of De Morgan's Law; negations will then either be eliminated by double negation or passed down to the leaves. Therefore, any arbitrary WFF can be transformed into a WFF consisting only of conjunctions and disjunctions of literals, conjunctions and disjunctions.
- For any WFF consisting only of conjunctions and disjunctions of disjunctions, conjunctions and literals, if the WFF includes conjunctions nested within conjunctions or disjunctions nested within disjunctions the structure can be flattened using associativity. Therefore, any arbitrary WFF can be converted into a WFF consisting only of conjunctions and disjunctions of literals, conjunctions and disjunctions, in which conjunctions only nest within disjunctions and vice versa.
- For any WFF consisting only of conjunctions and disjunctions of literals, conjunctions and disjunctions, in which conjunctions only nest within disjunctions and vice versa:
  - If it consists of a single literal, a conjunction of literals, a disjunction of literals, or a conjunction of disjunctions of literals, it is already in CNF.
  - If it consists of a disjunction of n conjunctions of literals, it is in DNF, and can be converted into a CNF as indicated in the 'short answer'. Using equivalences, this transformation can be performed

using either double negations combined with De Morgan's Law or distributivity combined with associativity (see flowchart).

- If the WFF is neither in CNF or in DNF, note that the same procedure to change DNF to CNF can also be used to convert a conjunction or disjunctions to a disjunction of conjunctions, or vice versa, within a more complex nesting of disjunction and conjunction; the result will be a WFF with the same number of levels of nesting, but which includes conjunctions nested within conjunctions, or disjunctions nested within disjunctions, which can then be flattened using distributivity. Thus, a WFF consisting only of conjunctions and disjunctions of literals, conjunctions and disjunctions, and containing both conjunction and disjunction, with m levels of nesting where $m < 2$, can always be flattened to a WFF consisting only of conjunctions and disjunctions of literals, with $m - 1$ levels of nesting, and as such, all such WFFs can be flattened to 2 levels of nesting. A WFF consisting only of conjunctions and disjunctions of literals, with exactly 2 levels of nesting must either be in CNF, or in DNF, which we have shown can always be converted into CNF.

- Therefore, any arbitrary WFF can be converted into CNF.