

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) print-outs 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:

$$(P \wedge Q) \rightarrow R, P, Q \models R \quad (1)$$

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

¹This 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:

$$((P \wedge Q) \rightarrow R) \wedge P \wedge Q \wedge \neg R \quad (2)$$

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 \perp as the disjunction (of no literals). So, a clause is \perp , or a literal, or a disjunction of literals; 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:

$$\begin{aligned} &\neg P \\ &P \vee \neg Q \vee \neg R \\ &P \wedge (\neg Q \vee \neg R) \\ &P \wedge \neg Q \wedge \neg R \\ &(P \vee \neg Q) \wedge (\neg R \vee P) \\ &(P \vee Q \vee \neg R) \wedge (\neg Q \vee \neg R) \wedge P \wedge (\neg S \vee \neg P) \end{aligned}$$

However the following expressions are *not* in CNF:

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

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,

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

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

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

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

$$(\neg P \vee \neg Q \vee R) \wedge P \wedge Q \wedge \neg R \quad (3)$$

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:

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

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

$$\{\{-P, \neg Q, R\}, \{P\}, \{Q\}, \{\neg R\}\} \quad (4)$$

(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;

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

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:

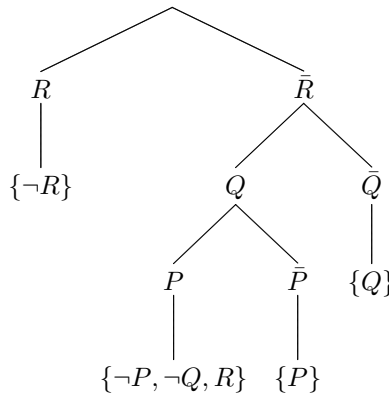
$$\frac{\{X\} \cup \mathcal{A}, \{\neg X\} \cup \mathcal{B}}{\mathcal{A} \cup \mathcal{B}}$$

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

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

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 if 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 containing 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:

$$(P \wedge Q) \rightarrow R, P, Q \models \neg R \quad (5)$$

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

$$((P \wedge Q) \rightarrow R) \wedge P \wedge Q \wedge R \quad (6)$$

In CNF:

$$(\neg P \vee \neg Q \vee R) \wedge P \wedge Q \wedge R \quad (7)$$

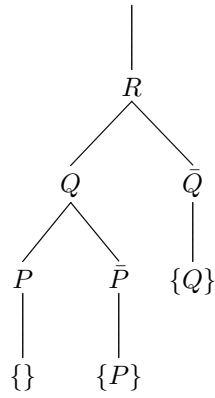
In clausal form:

$$\{\{\neg P, \neg Q, R\}, \{P\}, \{Q\}, \{R\}\} \quad (8)$$

We may apply resolution as follows

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

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) $C \models \neg\neg A \rightarrow B$

i. Convert to an expression:

ii. Convert to CNF:

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

i. Convert to an expression:

ii. Convert to CNF:

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

i. Convert to an expression:

ii. Convert to CNF:

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

i. Convert to an expression:

ii. Convert to CNF:

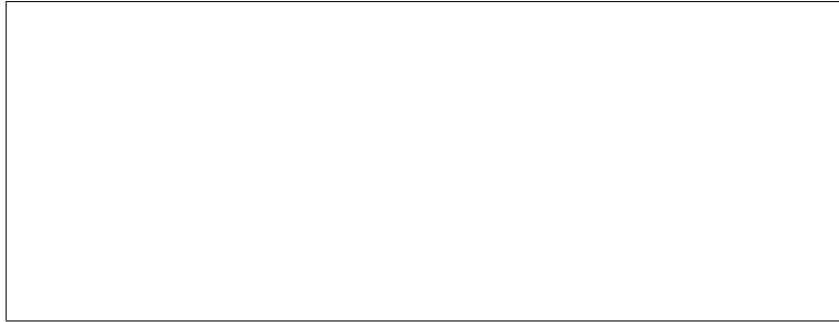
(e) $B \vee \neg E, C \leftrightarrow D \models A \rightarrow (B \rightarrow \neg C)$

i. Convert to an expression:

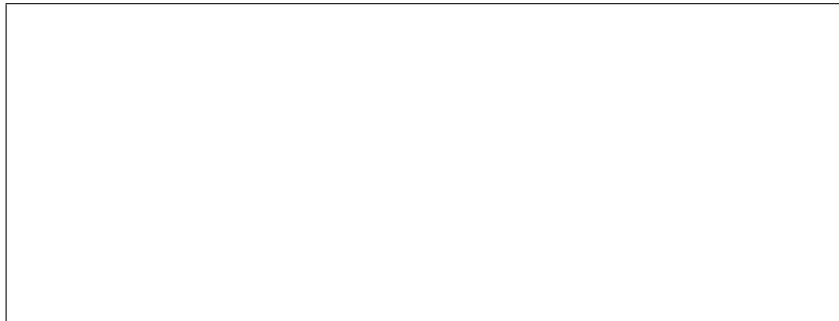
ii. Convert to CNF:

2. Consider 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.



(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.



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

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

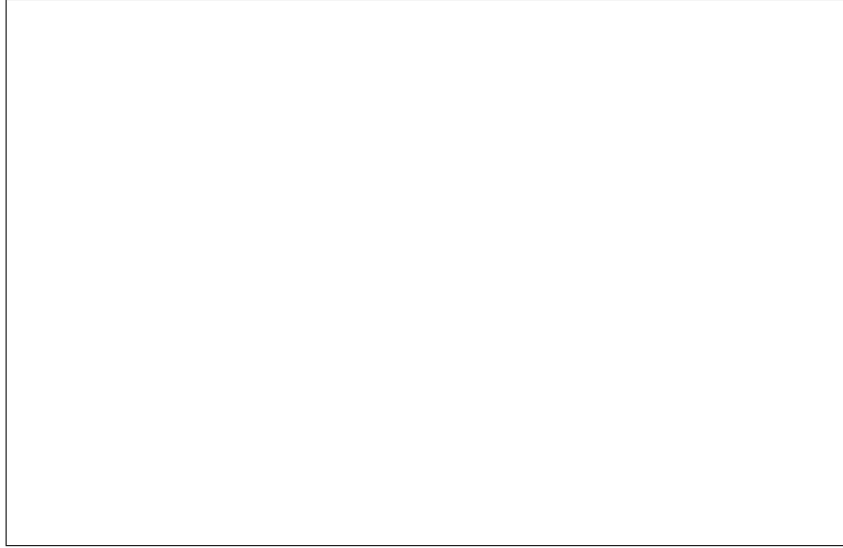
(a) Convert into an expression:

(b) Convert to CNF:

(c) Convert to clausal form:

(d) Apply the Davis-Putnam algorithm for resolution and state whether the original 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).



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

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

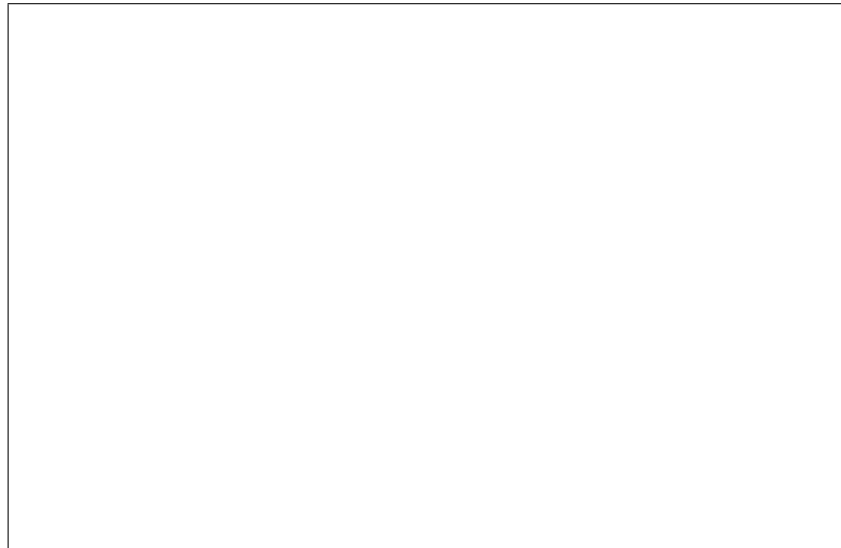
(a) Convert to an expression:

(b) Convert to CNF:

(c) Convert to clausal form:

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

- (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).



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

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

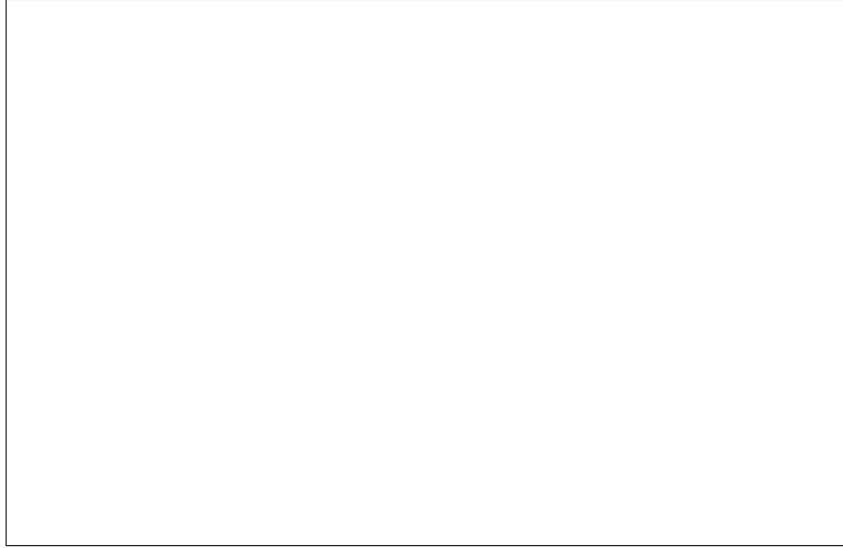
(a) Convert to an expression:

(b) Convert to CNF:

(c) Convert to clausal form:

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

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

(a) Convert to an expression:

(b) Convert to CNF:

(c) Convert to clausal form:

(d) Apply the Davis-Putnam algorithm for resolution and state whether the original 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).

