Exercise 20 Apply the Davis-Putnam procedure to the clause set

$$\{P, Q\} \qquad \{\neg P, Q\} \qquad \{P, \neg Q\} \qquad \{\neg P, \neg Q\}.$$

7.3 Introduction to resolution

Resolution is essentially the following rule of inference:

$$\frac{B \vee A \quad \neg B \vee C}{A \vee C}$$

To convince yourself that this rule is sound, note that B must be either false or true.

- if B is false, then $B \vee A$ is equivalent to A, so we get $A \vee C$
- if B is true, then $\neg B \vee C$ is equivalent to C, so we get $A \vee C$

You might also understand this rule via transitivity of $\to$ (with $D = \neg A$):

$$\frac{D \to B \quad B \to C}{D \to C}$$

A special case of resolution is when A and C are empty:

$$\frac{B \quad \neg B}{\mathbf{f}}$$

This detects contradictions.

Resolution works with disjunctions. The aim is to prove a contradiction, refuting a formula. Here is the method for proving a formula A:

1. Translate $\neg A$ into CNF as $A_1 \wedge \cdots \wedge A_m$.
2. Break this into a set of clauses: $A_1, \ldots, A_m$.
3. Repeatedly apply the resolution rule to the clauses, producing new clauses. These are all consequences of $\neg A$.
4. If a contradiction is reached, we have refuted $\neg A$.

In set notation the resolution rule is

$$\frac{\{B, A_1, \ldots, A_m\} \quad \{\neg B, C_1, \ldots, C_n\}}{\{A_1, \ldots, A_m, C_1, \ldots, C_n\}}$$

Resolution takes two clauses and creates a new one. A collection of clauses is maintained; the two clauses are chosen from the collection according to some strategy, and the new clause is added to it. If $m = 0$ or $n = 0$ then the new clause will be smaller than one of the parent clauses; if $m = n = 0$ then the new clause will be empty. A clause is true (in some interpretation) just when one of the literals is true; thus the empty clause indicates contradiction. It is written $\square$. If the empty clause is generated, resolution terminates successfully.

7.4 Examples of ground resolution

Let us try to prove

$$P \land Q \to Q \land P$$

Convert its negation to CNF:

$$\neg(P \land Q \to Q \land P)$$

We can combine steps 1 (eliminate $\to$) and 2 (push negations in) using the law $\neg(A \to B) \simeq A \land \neg B$:

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

Step 3, push disjunctions in, has nothing to do. The clauses are

$$\{P\} \qquad \{Q\} \qquad \{\neg Q, \neg P\}$$

We resolve $\{P\}$ and $\{\neg Q, \neg P\}$ as follows:

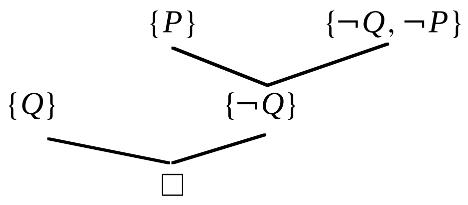
$$\frac{\{P\} \quad \{\neg P, \neg Q\}}{\{\neg Q\}}$$

The resolvent is $\{\neg Q\}$. Resolving $\{Q\}$ and with this new clause gives

$$\frac{\{Q\} \quad \{\neg Q\}}{\{\}}$$

The resolvent is the empty clause, properly written as $\square$. We have proved $P \land Q \to Q \land P$ by assuming its negation and deriving a contradiction.

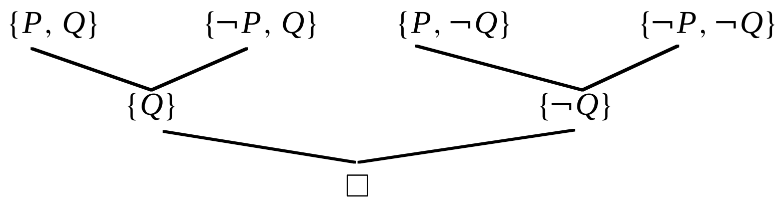
It is nicer to draw a tree like this:



Another example is $(P \leftrightarrow Q) \leftrightarrow (Q \leftrightarrow P)$. The steps of the conversion to clauses is left as an exercise; remember to negate the formula first! The final clauses are

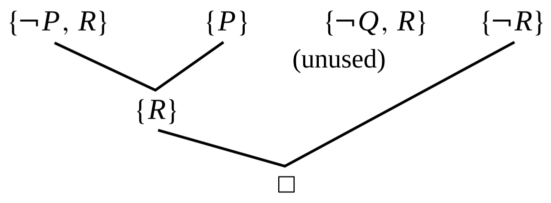
$$\{P, Q\} \qquad \{\neg P, Q\} \qquad \{P, \neg Q\} \qquad \{\neg P, \neg Q\}$$

A tree for the resolution proof is



Note that the tree contains $\{Q\}$ and $\{\neg Q\}$ rather than $\{Q, Q\}$ and $\{\neg Q, \neg Q\}$. If we forget to suppress repeated literals, we can get stuck. Resolving $\{Q, Q\}$ and $\{\neg Q, \neg Q\}$ (keeping repetitions) gives $\{Q, \neg Q\}$, a tautology. Tautologies are useless. Resolving this one with the other clauses leads nowhere. Try it.

These examples could mislead. Must a proof use each clause exactly once? No! A clause may be used repeatedly, and many problems contain redundant clauses. Here is an example:



Redundant clauses can make the theorem-prover flounder; this is a challenge facing the field.

Exercise 21 Prove $(A \to B \vee C) \to [(A \to B) \vee (A \to C)]$ using resolution.

7.5 A proof using a set of assumptions

In this example we assume

$$H \to M \vee N \qquad M \to K \wedge P \qquad N \to L \wedge P$$

and prove $H \to P$. It turns out that we can generate clauses separately from the assumptions (taken *positively*) and the conclusion (*negated*).

If we call the assumptions $A_1, \ldots, A_k$ and the conclusion B, then the desired theorem is

$$(A_1 \land \cdots \land A_k) \to B$$

Try negating this and converting to CNF. Using the law $\neg(A \to B) \simeq A \land \neg B$, the negation converts in one step to

$$A_1 \land \cdots \land A_k \land \neg B$$

Since the entire formula is a conjunction, we can separately convert $A_1, \ldots, A_k$, and $\neg B$ to clause form and pool the clauses together.

Assumption $H \to M \lor N$ is essentially in clause form already:

$$\{\neg H, M, N\}$$

Assumption $M \to K \land P$ becomes two clauses:

$$\{\neg M, K\} \qquad \{\neg M, P\}$$

Assumption $N \to L \land P$ also becomes two clauses:

$$\{\neg N, L\} \qquad \{\neg N, P\}$$

The negated conclusion, $\neg(H \to P)$, becomes two clauses:

$$\{H\} \qquad \{\neg P\}$$

A tree for the resolution proof is

$$\begin{tikzcd}
\{H\} \arrow[dr, dash] & & \{\neg H, M, N\} \arrow[dl, dash] & & & \\
& \{M, N\} \arrow[dr, dash] & & \{\neg M, P\} \arrow[dl, dash] & & \\
& & \{N, P\} \arrow[dr, dash] & & \{\neg N, P\} \arrow[dl, dash] & \\
& & & \{P\} \arrow[dr, dash] & & \{\neg P\} \arrow[dl, dash] \\
& & & & \square &
\end{tikzcd}$$

The clauses were not tried at random. Here are some points of proof strategy.

Ignoring irrelevance. Clauses $\{\neg M, K\}$ and $\{\neg N, L\}$ lead nowhere, so they were not tried. Resolving with one of these would make a clause containing K or L. There is no way of getting rid of either literal, for no clause contains $\neg K$ or $\neg L$. So this is not a way to obtain the empty clause.

Working from the goal. In each resolution step, at least one clause involves the negated conclusion (possibly via earlier resolution steps). We do not blindly derive facts from the assumptions — for, provided the assumptions are consistent, any contradiction will have to involve the negated conclusion. This strategy is called *set of support*.

Linear resolution. The proof has a linear structure: each resolvent becomes the parent clause for the next resolution step. Furthermore, the other parent clause is always one of the original set of clauses. This simple structure is very efficient because only the last resolvent needs to be saved. It is similar to the execution strategy of Prolog.

Exercise 22 Explain in more detail the conversion of this example into clauses.

Exercise 23 Prove Peirce's law, $((P \to Q) \to P) \to P$, using resolution.

Exercise 24 Prove $(Q \to R) \land (R \to P \land Q) \land (P \to Q \lor R) \to (P \leftrightarrow Q)$ using resolution.

7.6 Deletion of redundant clauses

During resolution, the number of clauses builds up dramatically; it is important to delete all redundant clauses.

Each new clause is a consequence of the existing clauses. A contradiction can only be derived if the original set of clauses is inconsistent. A clause can be deleted if it does not affect the consistency of the set. Any tautology should be deleted, since it is true in all interpretations.

Here is a subtler case. Consider the clauses

$$\{S, R\} \qquad \{P, \neg S\} \qquad \{P, Q, R\}$$

Resolving the first two yields $\{P, R\}$. Since each clause is a disjunction, any interpretation that satisfies $\{P, R\}$ also satisfies $\{P, Q, R\}$. Thus $\{P, Q, R\}$ cannot cause inconsistency, and should be deleted.

Put another way, $P \lor R$ implies $P \lor Q \lor R$. Anything that could be derived from $P \lor Q \lor R$ could also be derived from $P \lor R$. This sort of deletion is called *subsumption*; clause $\{P, R\}$ *subsumes* $\{P, Q, R\}$.

Exercise 25 Prove $(P \land Q \to R) \land (P \lor Q \lor R) \to ((P \leftrightarrow Q) \to R)$ by resolution. Show the steps of converting the formula into clauses.