Completeness of Propositional Resolution

In the slide set this is

Theorem 3. Propositional Resolution is refutationally complete.

That is, if a propositional clause set is unsatisfiable, then resolution will derive the empty clause $\Box$ eventually.

More precisely: if a clause set N is unsatisfiable then there is a resolution refutation of N.

We have looked at the DPLL procedure as a binary tree construction method. The siblings of these trees are labelled with complementary literals, which allows us to enumerate interpretations in a guided way by identifying clauses that are falsified in the interpretation induced by a current branch in the tree.

For proving theorem 3 we will employ the same kind of trees. Because we are talking about a formal proof, we need to be a bit careful and spell out explicitly all the required properties. These details can be found in the slide set at the end, under the heading "Semantic Trees".

It would be instructive to spell out the general proof in full detail. However, below I will not do that and just repeat the example presented in class on whiteboard.

Take the unsatisfiable clause set

$$N = \{\underbrace{\neg A \lor \neg B \lor \neg D}_{①}, \underbrace{\neg B \lor D}_{②}, \underbrace{\neg A \lor B}_{③}, \underbrace{A}_{④}\}$$

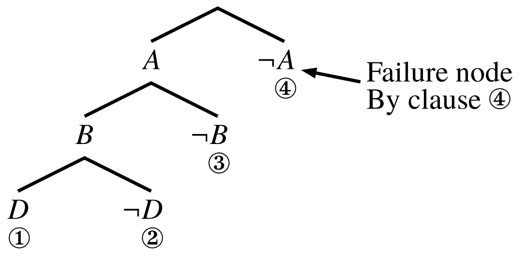
The completeness proof can (also) be thought of as an algorithm consisting of two steps. It modifies the clause set N as it goes and thereby constructs a derivation D that ends in the empty clause $\Box$, as desired. Of course we need to have that N is unsatisfiable.

Step one of the algorithm is initialization, and step 2 is executed while N does not contain the empty clause, thereby extending N and D with one or more clauses. Step 2 will work in a strictly decreasing way, similar as with the "weight" function for prove termination of the clause normalform transformation.

Step 1: The derivation D consists of the clauses in the initial clause set N, in any order. That is, e.g., D = ①, ②, ③, ④.

Step 1 also constructs a complete semantic tree $\mathcal{B}$ for $\{A, B, D\}$, the atoms occurring in N. It may look like this:

Initial semantic tree:



Because the clause set N is unsatisfiable, $\mathcal{B}$ must be closed, that is, every path ends in a failure node.[1]

It is important to note that the semantic tree construction is *not* part of the resolution calculus; it is a meta-level construction, a tool that we use to prove a property of the resolution calculus.

Step 2: Step 2 maintains a closed semantic tree $\mathcal{B}$ for the current clause set N. If $\mathcal{B}$ consists of the root node only then this entails that $\square \in N$ and we are done. (This is not the case on first execution of step 2 in the example.)

Otherwise $\mathcal{B}$ is a proper tree, not just a root-node only tree, like the one above. Now, locate any two sibling leaves. In the example these are only the leaves labelled with D and $\neg D$. Because these are failure nodes there must be clauses containing complementary literals. These are the clauses ① and ②, respectively, as annotated in the tree above.

Because the clauses ① and ② contain complementary literals, we can carry out a resolution inference, as follows (the literals resolved on are underlined):

$$\frac{\overbrace{\neg A \vee \neg B \vee \underline{\neg D}}^{①} \qquad \overbrace{\neg B \vee \underline{D}}^{②}}{\underbrace{\neg A \vee \neg B \vee \neg B}_{=:⑤}}$$

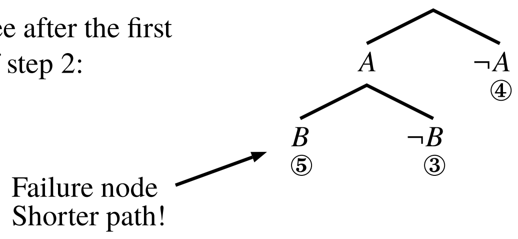
Because the leaves D and $\neg D$ are failure nodes, all other literals of the clauses ① and ② must also be falsified in the interpretations of the paths leading to the failure nodes.

Let us therefore add the clause ⑤ to N and extend the derivation by ⑤. In other words we now have $D =$ ①, ②, ③, ④, ⑤.

This enables us to loop around to step 2 again. The important thing is that there is a semantic tree $\mathcal{B}$ for our new clause set N that is *smaller* than the one at the beginning of step 2. More precisely, the resolvent ⑤ contains neither D nor $\neg D$, and contains only literals from the clauses ① and ②. This is why the resolvent ⑤ is falsified already in the *shorter* path $A - B$. The new semantic tree $\mathcal{B}$ hence looks like this, it is a proper sub-tree of the one above:

[1] A path to a failure node stands for a set of interpretations each of which falsifies some clause. For example, the path $A - B - \neg D$ stands for all interpretations $\{A \mapsto \text{true}, B \mapsto \text{true}, D \mapsto \text{false}, \ldots\}$, each of which falsifies clause ②.

Semantic tree after the first execution of step 2:



What we have just seen by means of the example should indicate the general principle: resolution inferences allow us to properly shrink the current semantic tree. We repeat this until the root-node only tree results, which, as said, indicates that we have derived the empty clause $\square$, as desired.

This is not the whole story though. There are still two unexplained cases: multiple occurrences of the same negative literal resolved on, and multiple occurrences of the same positive literal resolved on.

Our current example situation serves to illustrate what to do in the former case. Recall the clause ⑤ is $\neg A \lor \neg B \lor \neg B$. It has two occurrences of $\neg B$.

Let us try an inference rule with clause ③, as prescribed by the current semantic tree:

$$\frac{\overbrace{\neg A \lor \neg B \lor \underline{\neg B}}^{\text{③}} \qquad \overbrace{\neg A \lor \underline{B}}^{\text{③}}}{\underbrace{\neg A \lor \neg A \lor \neg B}_{=:\text{⑥}}}$$

Unlike to the situation with the first resolution inference above, the resolvent ⑥ does *not* admit the proper path prefix A for falsifying it. This is because of the extraneous literal $\neg B$ in ⑥. However, we can resolve away this occurrence of $\neg B$ by using the clause ③ again:

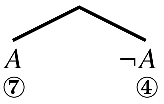
$$\frac{\overbrace{\neg A \lor \neg A \lor \underline{\neg B}}^{\text{⑥}} \qquad \overbrace{\neg A \lor \underline{B}}^{\text{③}}}{\underbrace{\neg A \lor \neg A \lor \neg A}_{=:\text{⑦}}}$$

Notice there is a general principle behind, which allows us to resolve away all occurrences of the negative literal in question. *However*, we need to make sure that the other parent clause, clause ③ in this case, has only *one* occurrence of the positive literal, B in this case. Should that not be the case, then the factoring rule can first be used to collapse all the multiple occurrences of the positive literal into one occurrence.

In our example this is not necessary, though, and we extend N and D as prescribed by the inferences. The derivation now is D = ①, ②, ③, ④, ⑤, ⑥, ⑦.

Once more we get a smaller semantic tree:

Semantic tree after the second execution of step 2:



Finally, a couple of resolution inferences:

$$\frac{\overbrace{\neg A \vee \neg A \vee \underline{\neg A}}^{\textcircled{7}} \qquad \overbrace{\underline{A}}^{\textcircled{4}}}{\underbrace{\neg A \vee \neg A}_{=:\textcircled{8}}}$$

$$\frac{\overbrace{\neg A \vee \underline{\neg A}}^{\textcircled{8}} \qquad \overbrace{\underline{A}}^{\textcircled{4}}}{\underbrace{\neg A}_{=:\textcircled{9}}}$$

$$\frac{\overbrace{\underline{\neg A}}^{\textcircled{9}} \qquad \overbrace{\underline{A}}^{\textcircled{4}}}{\Box}$$

The clause set now is $N = \{①, \ldots, ⑨, \Box\}$ and the semantic tree consists of the root-node only. The procedure stops and the final derivation, a refutation, is $D = ①, \ldots, ⑨, \Box$, as claimed.

A concluding remark. practical implementations of the resolution calculus do not search directly for a refutation. Rather, starting from the given clause set they apply resolution and factoring inference rules, in all possible ways, as long as possible and add the resolvent or factor to the clause set, until the empty clause $\Box$ is derived. The completeness proof sketched above can easily be adapted to this setting.