

Introduction to Logic

Refutation Proofs

Michael Genesereth
Computer Science Department
Stanford University

Proof Systems

Popular Types of Proof Systems:

Direct Proofs (Hilbert)

Natural Deduction (Fitch)

→ Refutation proofs (Resolution / Robinson)

Others:

Gentzen Systems

Sequent Calculi

and so forth

Both direct proof systems and natural deduction proof systems are "generative" in that they start with premises and end with conclusions. In refutation systems, we start with premises and the negations of our conclusions and derive contradictions. Resolution theorem proving is the most common system of this sort.

Direct Proof

A *direct proof* is a sequence of sentences terminating in a conclusion in which each item is either a premise, an instance of a valid schema, or the result of applying a rule of inference to earlier items in sequence.

- | | |
|----------------------|------------------------|
| 1. <i>premise</i> | Premise |
| 2. <i>premise</i> | Premise |
| ... | ... |
| <i>n. conclusion</i> | Some rule of inference |

The conclusion is proved *directly*.

One of the problems of direct and natural deduction systems is that there are schemas or rules that require that we choose from infinitely many possible instances. For example, in Or Introduction, we can disjoin results with any sentence whatsoever. Which should we choose? Similarly, when making assumptions, there are infinitely many possibilities. What should we assume?

Refutation Proof

A *refutation proof* is a sequence of sentences in which each sentence is a premise, the *negation* of a desired conclusion, or the result of applying a rule of inference to earlier items in sequence that terminates in some form of contradiction.

- | | |
|-----------------------------|------------------------|
| 1. <i>premise</i> | Premise |
| 2. <i>premise</i> | Premise |
| 3. $\neg$ <i>conclusion</i> | Negated conclusion |
| ... | ... |
| <i>n. contradiction</i> | Some rule of inference |

A refutation proof is a proof *by contradiction*.

One of the problems of direct and natural deduction systems is that there are schemas or rules that require that we choose from infinitely many possible instances. For example, in Or Introduction, we can disjoin results with any sentence whatsoever. Which should we choose? Similarly, when making assumptions, there are infinitely many possibilities. What should we assume?

Propositional Resolution

Propositional Resolution is a refutation proof system.

Just one rule of inference - the *Resolution Principle* - and no axiom schemas.

Propositional Resolution is sound and complete.

The search space in propositional resolution is smaller than that of direct proof systems or natural deduction systems.

Hitch: To order to use resolution, we need to transform sentences into a representation called clausal form.

Propositional Resolution is a proof system for Propositional Logic that does not have this problem. As we shall see, Propositional Resolution allows us to derive conclusions from premises without making arbitrary substitutions. Unlike Fitch, Propositional Resolution relies on a single rule of inference called the Resolution Principle; and unlike Hilbert, there are no axiom schemata. Using the resolution principle alone (without axiom schemata or other rules of inference), it is possible to build a reasoning system that is able to prove everything that can be proved in Hilbert or Fitch. Moreover, the search space using the resolution principle is smaller than the search space for generating Fitch proofs.

Programme

Clausal Form

Resolution Rule of Inference
Resolution Reasoning

Soundness and Completeness
Practical Matters

Box Logic

Propositional Resolution is a proof system for Propositional Logic that does not have this problem. As we shall see, Propositional Resolution allows us to derive conclusions from premises without making arbitrary substitutions. Unlike Fitch, Propositional Resolution relies on a single rule of inference called the Resolution Principle; and unlike Hilbert, there are no axiom schemata. Using the resolution principle alone (without axiom schemata or other rules of inference), it is possible to build a reasoning system that is able to prove everything that can be proved in Hilbert or Fitch. Moreover, the search space using the resolution principle is smaller than the search space for generating Fitch proofs.

Clausal Form

Clausal Form

Propositional resolution works only on expressions in *clausal form*.

Before we can apply resolution, we must first transform our sentences into clausal form.

$$(p \Rightarrow q) \longrightarrow \{\neg p, q\}$$

Fortunately, there is a simple algorithm for converting any set of Propositional Logic sentences into a logically equivalent set of sentences in clausal form.

Propositional Resolution works only on expressions in clausal form. Before the rule can be applied, the premises and conclusions must be converted to this form. Fortunately, as we shall see, there is a simple procedure for making this conversion.

Clausal Form

A *literal* is either an atomic sentence or a negation of an atomic sentence.

$$p, \neg p$$

A *clausal sentence* is either a literal or a disjunction of literals.

$$p, \neg p, \quad p \vee \neg q$$

A *clause* is a set of literals.

$$\{p\}, \{\neg p\}, \{p, \neg q\}$$

A literal is either an atomic sentence or a negation of an atomic sentence. For example, if p is a logical constant, the following sentences are both literals. A clausal sentence is either a literal or a disjunction of literals. If p and q are logical constants, then the following are clausal sentences. A clause is the set of literals in a clausal sentence. For example, the following sets are the clauses corresponding to the clausal sentences above.

Each clause is in

Clauses are Disjunctions

$$p \vee \neg q \longrightarrow \{p, \neg q\}$$

What about the empty clause?

The empty clause $\{\}$ is unsatisfiable.

Why? It is equivalent to an empty disjunction.

Note that the empty set $\{\}$ is also a clause. It is equivalent to an empty disjunction and, therefore, is unsatisfiable. As we shall see, it is a particularly important special case.

Conversion to Clausal Form

Implications Out:

$$\varphi_1 \Rightarrow \varphi_2 \longrightarrow \neg \varphi_1 \vee \varphi_2$$

$$\varphi_1 \Leftrightarrow \varphi_2 \longrightarrow (\neg \varphi_1 \vee \varphi_2) \wedge (\varphi_1 \vee \neg \varphi_2)$$

As mentioned earlier, there is a simple procedure for converting an arbitrary set of Propositional Logic sentences to an equivalent set of clauses. The conversion rules are summarized here and should be applied in order.

In the first step (Implications out), we eliminate all occurrences of the $\Rightarrow$ and $\Leftrightarrow$ operators by substituting equivalent sentences involving only the $\&$, $\vee$, and $\neg$ operators.

Conversion to Clausal Form

Implications Out:

$$\varphi_1 \Rightarrow \varphi_2 \longrightarrow \neg \varphi_1 \vee \varphi_2$$

$$\varphi_1 \Leftrightarrow \varphi_2 \longrightarrow (\neg \varphi_1 \vee \varphi_2) \wedge (\varphi_1 \vee \neg \varphi_2)$$

Negations In:

$$\neg \neg \varphi \longrightarrow \varphi$$

$$\neg(\varphi_1 \wedge \varphi_2) \longrightarrow \neg \varphi_1 \vee \neg \varphi_2$$

$$\neg(\varphi_1 \vee \varphi_2) \longrightarrow \neg \varphi_1 \wedge \neg \varphi_2$$

In the second step (Negations in), negations are distributed over other logical operators and quantifiers until each such operator applies to a single atomic sentence. The replacement rules shown here do the job. We eliminate double negations. We use deMorgan's laws to distribute negations over conjunctions and disjunctions; and we distribute negations over quantifiers. When this step is done, any remaining negations are applied directly to atomic sentences.

Conversion to Clausal Form

Distribution

$$\begin{aligned}\varphi_1 \vee (\varphi_2 \wedge \varphi_3) &\rightarrow (\varphi_1 \vee \varphi_2) \wedge (\varphi_1 \vee \varphi_3) \\ (\varphi_1 \wedge \varphi_2) \vee \varphi_3 &\rightarrow (\varphi_1 \vee \varphi_3) \wedge (\varphi_2 \vee \varphi_3) \\ \varphi \vee (\varphi_1 \vee \dots \vee \varphi_n) &\rightarrow (\varphi \vee \varphi_1 \vee \dots \vee \varphi_n) \\ (\varphi_1 \vee \dots \vee \varphi_n) \vee \varphi &\rightarrow (\varphi_1 \vee \dots \vee \varphi_n \vee \varphi) \\ \varphi \wedge (\varphi_1 \wedge \dots \wedge \varphi_n) &\rightarrow (\varphi \wedge \varphi_1 \wedge \dots \wedge \varphi_n) \\ (\varphi_1 \wedge \dots \wedge \varphi_n) \wedge \varphi &\rightarrow (\varphi_1 \wedge \dots \wedge \varphi_n \wedge \varphi)\end{aligned}$$

$$\begin{aligned}a^*(b+c) &\rightarrow (a^*b + a^*c) \\ (a+b)^*c &\rightarrow (a^*c + b^*c) \\ a+(b+c) &\rightarrow a+b+c \\ a*(b*c) &\rightarrow a*b*c\end{aligned}$$

In the third step (Distribution), we put the expression into conjunctive normal form, i.e. a conjunction of disjunctions of literals. This is accomplished by repeated use of the distribution rules listed here.

Conversion to Clausal Form

Operators Out

$$\begin{aligned}\varphi_1 \wedge \dots \wedge \varphi_n &\rightarrow \varphi_1 \\ &\dots \\ &\varphi_n \\ \varphi_1 \vee \dots \vee \varphi_n &\rightarrow \{\varphi_1, \dots, \varphi_n\}\end{aligned}$$

Example

$$(p \vee q) \wedge (r \vee r) \wedge \neg s \rightarrow \begin{array}{l} p \vee q \\ r \vee r \\ \neg s \end{array} \rightarrow \begin{array}{l} \{p, q\} \\ \{r\} \\ \{\neg s\} \end{array}$$

Finally, we eliminate operators by separating any conjunctions into its conjuncts and writing each conjunct as a separate clause.

Example

$$\begin{array}{l} g \wedge (r \Rightarrow f) \\ \text{I } g \wedge (\neg r \vee f) \\ \text{N } g \wedge (\neg r \vee f) \\ \text{D } g \wedge (\neg r \vee f) \\ \text{O } \{g\} \\ \quad \{\neg r, f\} \end{array}$$

As an example of this conversion process, consider the problem of transforming the expression shown here to clausal form. The initial expression appears on the top line, and the expressions on the labeled lines are the results of the corresponding steps of the conversion procedure. On the I step, we get rid of the implication operator. The N step has no effect since the only negation in the sentence is already applied to an atomic sentence. The D step similarly has no effect. Finally, on the O step, we convert our conjunction into two separate clauses.

Example

$\neg(g \wedge (r \Rightarrow f))$
I $\neg(g \wedge (\neg r \vee f))$
N $\neg g \vee \neg(\neg r \vee f)$
N $\neg g \vee (\neg\neg r \wedge \neg f)$
N $\neg g \vee (r \wedge \neg f)$
D $(\neg g \vee r) \wedge (\neg g \vee \neg f)$
O $\{\neg g, r\}$
 $\{\neg g, \neg f\}$

As a slightly more complicated case, consider the following conversion. We start with the same sentence except that, in this case, it is negated. As before, we start by eliminating the implication operator. We then apply the Negation rule three times. We distribute the Or over the And. And finally we convert the disjunctions to clauses. Note that, even though the sentences in these two examples are similar to start with (disagreeing on just one $\neg$ operator), the results are quite different.

Equivalence

Good News: The result of converting a set of sentences is logically equivalent to that set of sentences.

Upshot: Whatever we prove from clausal form is logically entailed by the original sentences.

In Propositional Logic, the set of clauses corresponding to any set of sentences is logically equivalent to that sentence. Hence, anything we determine about the set of clauses holds of the original sentences.

Equivalence

Good News: The result of converting a set of sentences is logically equivalent to that set of sentences.

Upshot: Whatever we prove from clausal form is logically entailed by the original sentences.

In Propositional Logic, the set of clauses corresponding to any set of sentences is logically equivalent to that sentence. Hence, anything we determine about the set of clauses holds of the original sentences.

Resolution Principle

The idea of Propositional Resolution is simple. Suppose we have the clause $\{p, q\}$. In other words, we know that p is true or q is true. Suppose we also have the clause $\{\neg q, r\}$. In other words, we know that q is false or r is true. One clause contains q , and the other contains $\neg q$. If q is false, then by the first clause p must be true. If q is true, then, by the second clause, r must be true. Since q must be either true or false, then it must be the case that either p is true or r is true. So we should be able to derive the clause $\{p, r\}$.

Intuition

Premises: $\{p, q\}$ and $\{\neg q, r\}$

If q is false, then the first clause tells us p must be true.
If q is true, then the second clause tells us r must be true.

Conclusion: $\{p, r\}$

The idea of Propositional Resolution is simple. Suppose we have the clause $\{p, q\}$. In other words, we know that p is true or q is true. Suppose we also have the clause $\{\neg q, r\}$. In other words, we know that q is false or r is true. One clause contains q , and the other contains $\neg q$. If q is false, then by the first clause p must be true. If q is true, then, by the second clause, r must be true. Since q must be either true or false, then it must be the case that either p is true or r is true. So we should be able to derive the clause $\{p, r\}$.

Resolution Principle

$$\frac{\begin{array}{l} \{\phi_1, \dots, \cancel{x}, \dots, \phi_m\} \\ \{\psi_1, \dots, \neg\cancel{x}, \dots, \psi_n\} \end{array}}{\{\phi_1, \dots, \phi_m, \psi_1, \dots, \psi_n\}}$$

This intuition is the basis for the rule of inference shown here. Given a clause containing a literal x and another clause containing the literal $\neg x$, we can infer the clause consisting of all the literals of both clauses without the complementary pair. This rule of inference is called Propositional Resolution or the Resolution Principle.

Example

$$\begin{array}{c} \{p, q\} \\ \{ \neg q, r \} \\ \hline \{p, r\} \end{array}$$

The case we just discussed is an example. If we have the clause $\{p, q\}$ and we also have the clause $\{\neg q, r\}$, then we can derive the clause $\{p, r\}$ in a single step.

Example

$$\begin{array}{c} \{p, q, r\} \\ \{\neg p\} \\ \hline \{q, r\} \end{array}$$

If either of the clauses is a singleton set, we see that the number of literals in the result is less than the number of literals in the other clause. For example, from the clause $\{p, q, r\}$ and the singleton clause $\{\neg p\}$, we can derive the shorter clause $\{q, r\}$.

Example

$$\begin{array}{c} \{p\} \\ \{\neg p\} \\ \hline \{\} \end{array}$$

Resolving two singleton clauses leads to the empty clause; i.e. the clause consisting of no literals at all, as shown below. The derivation of the empty clause means that the database contains a contradiction.

Example

$$\begin{array}{l} \{p, q\} \\ \{\neg p, \neg q\} \\ \hline \{q, \neg q\} \\ \{p, \neg p\} \end{array}$$

If two clauses resolve, they may have more than one resolvent because there can be more than one way in which to choose the resolvents.

Example

$$\begin{array}{l} \{p, q\} \\ \{\neg p, \neg q\} \\ \hline \{\} \end{array} \quad \textbf{Wrong!!!}$$

Note, however, when two clauses have multiple pairs of complementary literals, only one pair of literals may be resolved at a time. For example, the following is not a legal application of Propositional Resolution. If we were to allow this to go through, we would be saying that these two clauses are inconsistent. However, it is perfectly possible for $(p \vee q)$ to be true and $(\neg p \vee \neg q)$ to be true at the same time. For example, we just let p be true and q be false, and we have satisfied both clauses.

Implication Elimination

$$\begin{array}{ll} p \Rightarrow q & \{\neg p, q\} \\ p & \{p\} \\ \hline q & \{q\} \end{array}$$

It is easy to see that resolution subsumes many of our other rules of inference. Consider, for example, Implication Elimination, shown below on the left. If we have $(p \Rightarrow q)$ and we have p , then we can deduce q . The clausal form of the premises and conclusion are shown below on the right. The implication $(p \Rightarrow q)$ corresponds to the clause $\{\neg p, q\}$, and p corresponds to the singleton clause $\{p\}$. We have two clauses with a complementary literal, and so we cancel the complementary literals and derive the clause $\{q\}$, which is the clausal form of q .

Negation Introduction

$p \Rightarrow q$	$\{\neg p, q\}$
$p \Rightarrow \neg q$	$\{\neg p, \neg q\}$
$\neg p$	$\{\neg p\}$

It is easy to see that resolution subsumes many of our other rules of inference. Consider, for example, Implication Elimination, shown below on the left. If we have $(p \Rightarrow q)$ and we have p , then we can deduce q . The clausal form of the premises and conclusion are shown below on the right. The implication $(p \Rightarrow q)$ corresponds to the clause $\{\neg p, q\}$, and p corresponds to the singleton clause $\{p\}$. We have two clauses with a complementary literal, and so we cancel the complementary literals and derive the clause $\{q\}$, which is the clausal form of q .

Transitivity

- | | |
|----------------------|--------------------------------|
| 1. $p \Rightarrow q$ | Premise |
| 2. $q \Rightarrow r$ | Premise |
| 3. p | Assumption |
| 4. q | Implication Elimination: 1, 3 |
| 5. r | Implication Elimination: 2, 4 |
| 6. $p \Rightarrow r$ | Implication Introduction: 3, 5 |

For example, in the proof we just saw, we used this assumption operation in the nested subproof even though p was not among the given premises.

Transitivity

$p \Rightarrow q$	$\{\neg p, q\}$
$q \Rightarrow r$	$\{\neg q, r\}$
$p \Rightarrow r$	$\{\neg p, r\}$

It is easy to see that resolution subsumes many of our other rules of inference. Consider, for example, Implication Elimination, shown below on the left. If we have $(p \Rightarrow q)$ and we have p , then we can deduce q . The clausal form of the premises and conclusion are shown below on the right. The implication $(p \Rightarrow q)$ corresponds to the clause $\{\neg p, q\}$, and p corresponds to the singleton clause $\{p\}$. We have two clauses with a complementary literal, and so we cancel the complementary literals and derive the clause $\{q\}$, which is the clausal form of q .

Mary, Pat, and Quincy

If Mary loves Pat, then Mary loves Quincy. If it is Monday, then Mary loves Pat or Quincy. If it is Monday, does Mary love Quincy?

- | | | |
|----|--------------------------|--------------------------------|
| 1. | $p \Rightarrow q$ | Premise |
| 2. | $m \Rightarrow p \vee q$ | Premise |
| 3. | m | Assumption |
| 4. | $p \vee q$ | Implication Elimination: 2, 3 |
| 5. | q | Assumption |
| 6. | $q \Rightarrow q$ | Implication Introduction: 5, 5 |
| 7. | q | Or Elimination: 4, 1, 6 |
| 8. | $m \Rightarrow q$ | Implication Introduction: 3, 7 |

As another example, recall the example of formal reasoning introduced in Chapter 1. We said that, whenever we have two rules in which the left hand side of one contains a proposition constant that occurs on the right hand side of the other, then we can cancel those constants and deduce a new rule by combining the remaining constants on the left hand sides of both rules and the remaining constants on the right hand sides of both rules. As it turns out, this is just Propositional Resolution.

Example

$p \Rightarrow q$	$\{\neg p, q\}$
$m \Rightarrow p \vee q$	$\{\neg m, p, q\}$
$m \Rightarrow q$	$\{\neg m, q\}$

Given $(p \Rightarrow q)$ and $(m \& r \Rightarrow p \vee q)$, we deduce $(m \& r \Rightarrow q)$. On the right, we have the clausal form of the sentences on the left. In place of the first sentence, we have the clause $\{\neg p, q\}$; and, in place of the second sentence, we have $\{\neg m, \neg r, p, q\}$. Using resolution, we can deduce $\{\neg m, \neg r, q\}$, which is the clausal form of the sentence we derived on the left.

Resolution Reasoning

Reasoning with the Resolution Principle is analogous to reasoning with other rules of inference. We start with premises; we apply the Resolution Principle to those premises; we apply it again to the results of those applications; and so forth until we get to our desired conclusion or we run out of things to do.

Resolution Derivation

A *resolution derivation* of a conclusion from a set of premises is a finite sequence of clauses terminating in the conclusion in which each clause is either a premise or the result of applying the resolution principle to earlier elements of the sequence.

Reasoning with the Resolution Principle is analogous to reasoning with other rules of inference. We start with premises; we apply the Resolution Principle to those premises; we apply the rule to the results of those applications; and so forth until we get to our desired conclusion or we run out of things to do.

Formally, we define a resolution derivation of a conclusion from a set of premises to be a finite sequence of clauses terminating in the conclusion in which each clause is either a premise or the result of applying the Resolution Principle to earlier members of the sequence.

Note that our definition of resolution derivation is analogous to our definition of linear proof. However, in this case, we do not use the word proof, because we reserve that word for a slightly different concept, which is discussed later.

Or Elimination

- | | | |
|----|-----------------|-------------------|
| 1. | $\{\neg p, r\}$ | $p \Rightarrow r$ |
| 2. | $\{\neg q, r\}$ | $q \Rightarrow r$ |
| 3. | $\{p, q\}$ | $p \vee q$ |
| 4. | $\{q, r\}$ | 1, 3 |
| 5. | $\{r\}$ | 2, 4 |

In many cases, it is possible to find resolution derivations of conclusions from premises. Suppose, for example, we are given the clauses $\{\neg p, r\}$ and $\{\neg q, r\}$ and $\{p, q\}$. Then we can derive the conclusion $\{r\}$ as shown here.

Resolution Not Generatively Complete

Seemingly Bad News: Using the Resolution Principle alone, it is not possible to generate every clause that is logically entailed by a set of premises.

Premises: $\{p\}$ and $\{q\}$
Conclusion: $\{p, q\}$

Premises: none
Conclusion: $\{p, \neg p\}$

But resolution cannot generate these results!

It is noteworthy that the resolution is not generatively complete, i.e. it is not possible to find resolution derivations for all clauses that are logically entailed by a set of premise clauses. For example, given the clause $\{p\}$ and the clause $\{q\}$, there is no resolution derivation of $\{p, q\}$, despite the fact that it is logically entailed by the premises in this case.

As another example, consider that valid clauses (such as $\{p, \neg p\}$) are always true, and so they are logically entailed by any set of premises, including the empty set. However, Propositional Resolution requires some premises to have any effect. Given an empty set of premises, we would not be able to derive any conclusions, including these valid clauses.

Resolution Determines Unsatisfiability

Good News: If a set of clauses is unsatisfiable, it is possible to derive the empty clause using the Resolution Principle.

1.	$\{p, q\}$	Premise
2.	$\{p, \neg q\}$	Premise
3.	$\{\neg p, q\}$	Premise
4.	$\{\neg p, \neg q\}$	Premise
5.	$\{p\}$	1, 2
6.	$\{\neg p\}$	3, 4
7.	$\{\}$	5, 6

On the positive side, if a set Δ of Propositional Logic sentences is unsatisfiable, then there is guaranteed to be a resolution derivation of the empty clause from the clausal form of Δ .

As an example, consider the clauses $\{p, q\}$, $\{p, \neg q\}$, $\{\neg p, q\}$, and $\{\neg p, \neg q\}$. There is no truth assignment that satisfies all four of these clauses. Consequently, starting with these clauses, we should be able to derive the empty clause; and we can. A resolution derivation is shown below.

Resolution Method

Unsatisfiability Determination: If a set of clauses is unsatisfiable, it is possible to derive the empty clause using the Resolution Principle.

Unsatisfiability Theorem: $\Delta \models \phi$ if and only if $\Delta \cup \{\neg \phi\}$ is unsatisfiable.

Resolution Method: To prove that a set Δ of sentences logically entails a conclusion ϕ , write $\Delta \cup \{\neg \phi\}$ in clausal form and derive the empty clause.

38

The reason this is good news is that we can use the relationship between unsatisfiability and logical entailment to produce a method for determining logical entailment as well. Recall that the Unsatisfiability Theorem introduced in Chapter 3 tells that a set Δ of sentences logically entails a sentence ϕ if and only if the set of sentences $\Delta \cup \{\neg \phi\}$ is inconsistent.

So, to determine logical entailment, all we need to do is to negate our goal, add it to our premises, and use Resolution to determine whether the resulting set is unsatisfiable. A resolution proof of a sentence ϕ from a set Δ of sentences is a resolution derivation of the empty clause from the clausal form of $\Delta \cup \{\neg \phi\}$. A sentence ϕ is provable from a set of sentences Δ by Propositional Resolution (written $\Delta \vdash \phi$) if and only if there is a resolution proof of ϕ from Δ .

Example

Given p , $(p \Rightarrow q)$, and $(p \Rightarrow q) \Rightarrow (q \Rightarrow r)$, prove r .

$(p \Rightarrow q) \Rightarrow (q \Rightarrow r)$
 I $\neg(\neg p \vee q) \vee (\neg q \vee r)$
 N $(\neg \neg p \wedge \neg q) \vee (\neg q \vee r)$
 N $(p \wedge \neg q) \vee (\neg q \vee r)$
 D $(p \vee (\neg q \vee r)) \wedge (\neg q \vee (\neg q \vee r))$
 D $(p \vee \neg q \vee r) \wedge (\neg q \vee \neg q \vee r)$
 O $\{p, \neg q, r\}$
 $\{\neg q, r\}$

39

As an example of a resolution proof, consider one of the problems we saw earlier. We have three premises - p , $(p \Rightarrow q)$, and $(p \Rightarrow q) \Rightarrow (q \Rightarrow r)$. Our job is to prove r . The most interesting part of this proof is converting the third premise to clausal form.

Proof

1. $\{p\}$ p
2. $\{\neg p, q\}$ $p \Rightarrow q$
3. $\{p, \neg q, r\}$ $(p \Rightarrow q) \Rightarrow (q \Rightarrow r)$
4. $\{\neg q, r\}$ $(p \Rightarrow q) \Rightarrow (q \Rightarrow r)$
5. $\{\neg r\}$ Negated Goal
6. $\{q\}$ 1, 2
7. $\{r\}$ 6, 4
8. $\{\}$ 7, 5

40

Once we have converted our sentences and the negated goal to clausal form, we can do the proof. The first four clauses in the proof correspond to the first two premises of the problem, and the final clause comes from the negation of the goal. Resolving the first clause with the second, we get the clause q , shown on line 6. Resolving this with the fourth clause gives us r . And resolving this with the clause on line 5 gives us the empty clause.

Example

Show $(p \Rightarrow (q \Rightarrow p))$ is valid, i.e. $\{\} \models (p \Rightarrow (q \Rightarrow p))$.

- $\neg(p \Rightarrow (q \Rightarrow p))$
- I $\neg(\neg p \vee (\neg q \vee p))$
- N $\neg \neg p \wedge \neg(\neg q \vee p)$
- N $p \wedge \neg(\neg q \vee p)$
- N $p \wedge (\neg \neg q \wedge \neg p)$
- N $p \wedge (q \wedge \neg p)$
- D $p \wedge q \wedge \neg p$
- O $\{p\}$
- $\{q\}$
- $\{\neg p\}$

41

Here is another example, this time illustrating the way in which we can use Resolution to prove valid sentences. Let's say that we have no premises at all and we want to prove $(p \Rightarrow (q \Rightarrow p))$, an instance of the Implication Creation axiom schema.

The first step is to negate this sentence and convert to clausal form. A trace of the conversion process is shown below. Note that we end up with three clauses.

Proof

1. $\{p\}$ $(p \Rightarrow (q \Rightarrow p))$
2. $\{q\}$ $(p \Rightarrow (q \Rightarrow p))$
3. $\{\neg p\}$ $(p \Rightarrow (q \Rightarrow p))$
4. $\{\}$ 1, 3

42

Finally, we take these clauses and produce a resolution derivation of the empty clause in one step.

One of the best features of Propositional Resolution is that it is much more focussed than the other proof methods we have seen. There is no need to choose instantiations carefully or to search through infinitely many possible instantiations for rules of inference.

Moreover, unlike the other methods we have seen, Propositional Resolution can be used in a proof procedure that always terminates without losing completeness. Since there are only finitely many clauses that can be constructed from a finite set of proposition constants, the procedure eventually runs out of new conclusions to draw, and when this happens we can terminate our search for a proof.

Soundness and Completeness

Logical Entailment and Provability

A set of premises Δ *logically entails* a conclusion ϕ (written $\Delta \models \phi$) if and only if every interpretation that satisfies Δ also satisfies ϕ .

If there exists a proof of a sentence ϕ from a set Δ of premises using the rules of inference in R , we say that ϕ is *provable* from Δ using R (written $\Delta \vdash_R \phi$).

In talking about Logic, we now have two notions - logical entailment and provability. A set of premises logically entails a conclusion if and only if every truth assignment that satisfies the premises also satisfies the conclusion. A sentence is provable from a set of premises if and only if there is a finite proof of the conclusion from the premises. The concepts are quite different. One is based on truth assignments; the other is based on symbolic manipulation of expressions. Yet, for the proof systems we have been examining, they are closely related.

Soundness and Completeness

A proof system is *sound* if and only if every provable conclusion is logically entailed.

If $\Delta \vdash \phi$, then $\Delta \models \phi$.

A proof system is *complete* if and only if every logical conclusion is provable.

If $\Delta \models \phi$, then $\Delta \vdash \phi$.

We say that a proof system is sound if and only if every provable conclusion is logically entailed. In other words, if $\Delta \vdash \phi$, then $\Delta \models \phi$. We say that a proof system is complete if and only if every logical conclusion is provable. In other words, if $\Delta \models \phi$, then $\Delta \vdash \phi$.

Resolution

Theorem: Resolution is sound and complete for Propositional Logic.

$\Delta \models \varphi$ if and only if $\Delta \vdash_{\text{Resolution}} \varphi$.

Upshot: The truth table method and the resolution method succeed in exactly the same cases!

Resolution is sound and complete for Propositional Logic. In other words, for this system, logical entailment and provability are identical. An arbitrary set of sentences Δ logically entails an arbitrary sentence ϕ if and only if ϕ is provable from Δ using Resolution.

Practical Matters

Two Finger Method

1. $\{p, q\}$ Premise
2. $\{p, \neg q\}$ Premise
3. $\{\neg p, q\}$ Premise
4. $\{\neg p, \neg q\}$ Premise

Two Finger Method

-→ 1. $\{p, q\}$ Premise
- 2. $\{p, \neg q\}$ Premise
- 3. $\{\neg p, q\}$ Premise
- 4. $\{\neg p, \neg q\}$ Premise

Two Finger Method

-→ 1. $\{p, q\}$ Premise
- 2. $\{p, \neg q\}$ Premise
- 3. $\{\neg p, q\}$ Premise
- 4. $\{\neg p, \neg q\}$ Premise
- 5. $\{p\}$ 1, 2

Two Finger Method

- 1. $\{p, q\}$ Premise
-→ 2. $\{p, \neg q\}$ Premise
- 3. $\{\neg p, q\}$ Premise
- 4. $\{\neg p, \neg q\}$ Premise
- 5. $\{p\}$ 1, 2

Two Finger Method

-→ 1. $\{p, q\}$ Premise
- 2. $\{p, \neg q\}$ Premise
- 3. $\{\neg p, q\}$ Premise
- 4. $\{\neg p, \neg q\}$ Premise
- 5. $\{p\}$ 1, 2
- 6. $\{q\}$ 1, 3

Two Finger Method

- 1. $\{p, q\}$ Premise
-→ 2. $\{p, \neg q\}$ Premise
- 3. $\{\neg p, q\}$ Premise
- 4. $\{\neg p, \neg q\}$ Premise
- 5. $\{p\}$ 1, 2
- 6. $\{q\}$ 1, 3
- 7. $\{\neg q, q\}$ 2, 3
- ...

Two Finger Method

- 1. $\{p, q\}$ Premise
- 2. $\{p, \neg q\}$ Premise
-→ 3. $\{\neg p, q\}$ Premise
- 4. $\{\neg p, \neg q\}$ Premise
- 5. $\{p\}$ 1, 2
- 6. $\{q\}$ 1, 3
- 7. $\{\neg q, q\}$ 2, 3
- ...

Proof as Produced by Two-Finger Method

1. $\{p, q\}$	$p \vee q$	11. $\{\neg p\}$	3,4
2. $\{p, \neg q\}$	$p \vee \neg q$	12. $\{q\}$	3,5
3. $\{\neg p, q\}$	$\neg p \vee q$	13. $\{\neg q\}$	4,5
4. $\{\neg p, \neg q\}$	$\neg p \vee \neg q$	14. $\{p\}$	2,6
5. $\{p\}$	1,2	15. $\{\neg p\}$	4,6
6. $\{q\}$	1,3	16. $\{p, q\}$	1,7
7. $\{\neg q, q\}$	2,3	17. $\{\neg q, p\}$	2,7
8. $\{p, \neg p\}$	2,3	18. $\{\neg p, q\}$	3,7
9. $\{q, \neg q\}$	1,4	19. $\{\neg q, \neg p\}$	4,7
9.5 $\{p, \neg p\}$	1,4	20. $\{q\}$	6,7
10. $\{\neg q\}$	2,4		

Proof as Produced by Two-Finger Method

21. $\{\neg q, q\}$	7,7	31. $\{\neg q, p\}$	2,9
22. $\{\neg q, q\}$	7,7	32. $\{\neg p, q\}$	3,9
23. $\{q, p\}$	1,8	33. $\{\neg q, \neg p\}$	4,9
24. $\{\neg q, p\}$	2,8	34. $\{q\}$	6,9
25. $\{\neg p, q\}$	3,8	35. $\{\neg q, q\}$	7,9
26. $\{\neg p, \neg q\}$	4,8	36. $\{q, \neg q\}$	9,9
27. $\{p\}$	5,8	37. $\{q, \neg q\}$	9,9
28. $\{\neg p, p\}$	8,8	38. $\{p\}$	1,10
29. $\{\neg p, p\}$	8,8	39. $\{\neg p\}$	3,10
30. $\{p, q\}$	1,9	40. $\{\}$	6,10

Proof with Identical Clause Elimination

1. $\{p, q\}$	$p \vee q$
2. $\{p, \neg q\}$	$p \vee \neg q$
3. $\{\neg p, q\}$	$\neg p \vee q$
4. $\{\neg p, \neg q\}$	$\neg p \vee \neg q$
5. $\{p\}$	1,2
6. $\{q\}$	1,3
7. $\{\neg q, q\}$	2,3
8. $\{p, \neg p\}$	2,3
9. $\{q, \neg q\}$	1,4
10. $\{\neg q\}$	2,4
11. $\{\neg p\}$	3,4
12. $\{\}$	6,10

Identical Clause Elimination

Metatheorem: There is a resolution refutation of Δ if and only if there is a resolution refutation from Δ in which no clause occurs twice. (Obviously.)

Upshot: If you generate a clause that is already in the proof, do not include it again.

Metatheorem: There are only finitely many clauses that can be formed from a finite set of proposition constants.

Upshot: You will eventually run out of things to do. So possible to terminate search in finite time!!!

Tautology Elimination

A *tautology* is a clause with a complementary pair of literals.

$$\{q, \neg q\}$$

$$\{p, q, r, \neg q\}$$

Metatheorem: There is a resolution refutation of Δ if and only if there is a resolution refutation from Δ with tautology elimination.

Proof with ICE and TE

- | | | |
|----|----------------------|----------------------|
| 1. | $\{p, q\}$ | $p \vee q$ |
| 2. | $\{p, \neg q\}$ | $p \vee \neg q$ |
| 3. | $\{\neg p, q\}$ | $\neg p \vee q$ |
| 4. | $\{\neg p, \neg q\}$ | $\neg p \vee \neg q$ |
| 5. | $\{p\}$ | 1,2 |
| 6. | $\{q\}$ | 1,3 |
| 7. | $\{\neg q\}$ | 2,4 |
| 8. | $\{\neg p\}$ | 3,4 |
| 9. | $\{\}$ | 6,7 |

Motivation for Subsumption

1. $\{p, q\}$ Premise
2. $\{p, q, r\}$ Premise
3. $\{q, r\}$ Premise
4. $\{\neg p\}$ Premise
5. $\{\neg q\}$ Premise
6. $\{\neg r\}$ Premise

Propositional Subsumption

A clause Φ *subsumes* Ψ if and only if Φ is a subset of Ψ .

Example: $\{p, q\}$ subsumes $\{p, q, r\}$

Metatheorem: There is a resolution refutation of Δ if and only if there is a resolution refutation from Δ with Propositional Subsumption.

Note

The resolution of two clauses sometimes produces a clause that subsumes one of its parents.

1. $\{p\}$ Premise
2. $\{\neg r, q\}$ Premise
3. $\{r\}$ Premise
4. $\{\neg p, \neg q, \neg r\}$ Premise
5. $\{\neg q, \neg r\}$ 1,4
6. $\{\neg r\}$ 2,5
7. $\{\}$ 3,6

Example of Pure Literal Elimination

1. $\{p, q\}$ Premise
2. $\{\neg p, r\}$ Premise
3. $\{\neg q, r\}$ Premise
4. $\{\neg q, s\}$ Premise
5. $\{\neg r\}$ Goal

Pure Literal Elimination

A *literal* in a database is *pure* if and only if there is no complementary occurrence of the literal in the database.

A *clause* is *superfluous* if and only if it contains a pure literal.

Metatheorem: There is a resolution refutation of Δ if and only if there is a resolution refutation from Δ in which all superfluous clauses are removed.

Example

1. $\{p, q\}$ Premise
2. $\{\neg p, r\}$ Premise
3. $\{\neg q, r\}$ Premise
4. $\{\neg q, s\}$ Premise
5. $\{\neg r\}$ Goal

Note

The removal of a superfluous clause may create new pure literals and new superfluous clauses.

1. $\{p, q\}$ $p \vee q$
2. $\{\neg p, r\}$ $p \Rightarrow r$
3. $\{\neg q, r\}$ $q \Rightarrow r$
4. $\{\neg q, s, t\}$ $q \Rightarrow s \vee t$
5. $\{\neg r\}$ $\neg r$
6. $\{\neg t\}$ $\neg t$

Strategies

Elimination Strategies (Constraints on clauses):

Identical Clause Elimination

Pure Literal Elimination

Tautology Elimination

Subsumption Elimination

Restriction Strategies (Constraints on inferences):

Unit Restriction

Input Restriction

Linear Restriction

Set of Support Restriction

Resolution Tools

Course Website

<http://logica.stanford.edu>

Show Quine on $g \& (r \Rightarrow f)$ and $\sim(g \& (r \Rightarrow f))$

Show Robinson

$p \Rightarrow q$

$m \Rightarrow plq$

$m \Rightarrow q$

Demonstrate benefits and problems of Select All

