6.825 Techniques in Artificial Intelligence

Resolution Theorem Proving: First Order Logic

Resolution with variables

Clausal form

We've been doing first-order logic and thinking about how to do proofs. Last time we looked at how to do resolution in the propositional case, and we looked at how to do unification -- that is, essentially matching of terms, figuring out which variables you have to match up with which other variables or functions or constants in order to get two terms to match up and look the same. And so why do unification? Because it gives us a tool for doing resolution in the first-order case.

Resolution with Variables

$\alpha \vee \phi$ [rename]

$\neg\psi \vee \beta$ [rename] MGU(ϕ,ψ) = θ

$$\frac{\alpha \vee \phi \qquad \neg\psi \vee \beta}{(\alpha \vee \beta)\theta}$$

Here's the rule for first-order resolution.

It says if you have a formula alpha or phi and another formula not psi or beta, and you can unify phi and psi with unifier theta, then you're allowed to conclude alpha or beta with the substitution theta applied to it.

Resolution with Variables

$\alpha \vee \phi$ [rename]
$\neg\psi \vee \beta$ [rename] $MGU(\phi,\psi) = \theta$

$(\alpha \vee \beta)\theta$

$P(x) \vee Q(x,y)$
$\neg P(A) \vee R(B,z)$

$\theta = \{x/A\}$

Let's look at an example. Let's say we have P(x) or Q(x,y) and we also have not P(A) or R(B,x). What are we going to be able to resolve here? P(x) will be phi, Q(x,y) will be alpha, P(A) will be psi and R(B,x) will be beta. The unifier will be {x/A}.

Resolution with Variables

$\alpha \vee \phi$ [rename]

$\neg\psi \vee \beta$ [rename] MGU(ϕ,ψ) = θ

$$\frac{\begin{array}{l}\alpha \vee \phi \\ \neg\psi \vee \beta\end{array}}{(\alpha \vee \beta)\theta}$$

$$\frac{\begin{array}{r}P(x) \vee Q(x,y) \\ \neg P(A) \vee R(B,z)\end{array}}{(Q(x,y) \vee R(B,z))\theta}$$

$\theta = \{x/A\}$

So, we get rid of the P literals, and end up with Q(x,y) v R(B,z), but then we have to apply our substitution to the result.

Resolution with Variables

$\alpha \vee \phi$ [rename]

$\neg\psi \vee \beta$ [rename] MGU(ϕ,ψ) = θ

$$\frac{\alpha \vee \phi \qquad \neg\psi \vee \beta}{(\alpha \vee \beta)\theta}$$

$$\frac{\begin{array}{l} P(x) \vee Q(x,y) \\ \neg P(A) \vee R(B,z) \end{array}}{(Q(x,y) \vee R(B,z))\theta}$$

$Q(A,y) \vee R(B,z)$

$\theta = \{x/A\}$

Finally, we end up with Q(A,y) or R(B,z).

Resolution with Variables

$\alpha \vee \phi$ [rename]
$\neg\psi \vee \beta$ [rename]

$$\frac{\begin{array}{l}\alpha \vee \phi \\ \neg\psi \vee \beta\end{array}}{(\alpha \vee \beta)\theta} \qquad \mathrm{MGU}(\phi,\psi) = \theta$$

$$\frac{\begin{array}{l}P(x) \vee Q(x,y) \\ \neg P(A) \vee R(B,x)\end{array}}{\quad}$$

$$\frac{\begin{array}{l}P(x) \vee Q(x,y) \\ \neg P(A) \vee R(B,z)\end{array}}{(Q(x,y) \vee R(B,z))\theta}$$

$$Q(A,y) \vee R(B,z)$$

$\theta = \{x/A\}$

Now let's explore what happens if we have X's in the other formula. So what if we replaced the z in the second sentence by an x.

Resolution with Variables

$\alpha \lor \phi$ [rename]

$\neg\psi \lor \beta$ [rename]

$$\frac{\alpha \lor \phi \qquad \neg\psi \lor \beta}{(\alpha \lor \beta)\theta} \qquad \mathrm{MGU}(\phi,\psi) = \theta$$

All vars implicitly univ. quantified

$\forall xy.\ P(x) \lor Q(x,y)$

$\forall z.\ \neg P(A) \lor R(B,z)$

$(Q(x,y) \lor R(B,z))\theta$

$Q(A,y) \lor R(B,z)$

$\theta = \{x/A\}$

$\forall xy.\ \ P(x) \lor Q(x,y)$

$\forall x.\ \ \neg P(A) \lor R(B,x)$

Scope of var is local to a clause. Use renaming to keep vars distinct

The x's in the two sentences are actually different. There is an implicit universal quantifier on the outside of each of these sentences (we'll see exactly how we get sentences ready for resolution in the next few slides). So, in order to avoid being confused by the fact that these two variables named x need not refer to the same thing, we will "rename them apart".

Resolution with Variables

α v ϕ [rename]
$\neg\psi$ v β [rename]   MGU(ϕ,ψ) = θ

$$\frac{\alpha \vee \phi \qquad \neg\psi \vee \beta}{(\alpha \vee \beta)\theta}$$

All vars implicitly
univ. quantified

$\forall$ xy. P(x) v Q(x,y)
$\forall$z. ¬P(A) v R(B,z)

(Q(x,y) v R(B,z))θ
Q(A,y) v R(B,z)

θ = {x/A}

$\forall$ xy. P(x) v Q(x,y)
$\forall$ x. ¬ P(A) v R(B,x)

Scope of var is local to a clause.
Use renaming to keep vars distinct

$\forall$ x_1y. P(x_1) v Q(x_1,y)
$\forall$ x_2. ¬ P(A) v R(B,x_2)

So that means that before you try to do a resolution step, you're really supposed to rename the variables in the two sentences so that they don't share any variables in common.

You won't usually do this that explicitly on your paper, but if you were going to implement this, or if you find yourself with the same variable in both sentences and it's getting confusing, then you should rename the sentences apart.

Resolution with Variables

$$\frac{\begin{array}{l}\alpha \vee \phi \text{ [rename]} \\ \neg\psi \vee \beta \text{ [rename]}\end{array}}{(\alpha \vee \beta)\theta} \qquad MGU(\phi,\psi) = \theta$$

All vars implicitly univ. quantified

$$\frac{\begin{array}{l}\forall xy.\ P(x) \vee Q(x,y) \\ \forall z. \neg P(A) \vee R(B,z)\end{array}}{(Q(x,y) \vee R(B,z))\theta}$$

$$Q(A,y) \vee R(B,z)$$

$\theta = \{x/A\}$

$$\frac{\begin{array}{l}\forall xy.\ \ P(x) \vee Q(x,y) \\ \forall x.\ \ \neg P(A) \vee R(B,x)\end{array}}{}$$

Scope of var is local to a clause.
Use renaming to keep vars distinct

$$\frac{\begin{array}{l}\forall x_1y.\ \ P(x_1) \vee Q(x_1,y) \\ \forall x_2.\ \ \neg P(A) \vee R(B,x_2)\end{array}}{(Q(x_1,y) \vee R(B,x_2))\theta}$$

$$Q(A,y) \vee R(B,x_2)$$

$\theta = \{x_1/A\}$

The easiest thing to do is to just go through and give every variable a new name. It's OK to do that. You just have to do it consistently for each clause. So you could rename to P of X1 or Q of X1Y1, and you can name this one not P of A or R of BX2. And then you could apply the resolution rule and you don't get into any trouble.

Resolution

Input are sentences in conjunctive normal form with no apparent quantifiers (implicit universal quantifiers).

I introduced the resolution rule in detail so that we can see what we're trying to do, The resolution rule takes sentences in conjunctive normal form with apparently no quantifiers, right? The rule doesn't say anything about quantifiers. I told you that clauses have kind of an implicit quantifier in them. But now we've been looking at languages that have quantifiers.

Resolution

Input are sentences in conjunctive normal form with no apparent quantifiers (implicit universal quantifiers).

How do we go from the full range of sentences in FOL, with the full range of quantifiers, to sentences that enable us to use resolution as our single inference rule?

So the question is: how do we go from sentences with the whole rich set of quantifiers into a form that lets us use resolution? Because it's going to turn out that even in first-order logic, resolution is a complete proof procedure all by itself. We're not going to need any more inference rules.

Resolution

Input are sentences in conjunctive normal form with no apparent quantifiers (implicit universal quantifiers).

How do we go from the full range of sentences in FOL, with the full range of quantifiers, to sentences that enable us to use resolution as our single inference rule?

We will convert the input sentences into a new normal form called clausal form.

So what we're going to do is introduce a normal form that's kind of like conjunctive normal form, only it deals with quantifiers, too. It's called *clausal* form. Or, sometimes, *prenex normal form*.

Converting to Clausal Form

Rather than give you a definition, I'm going to teach you a procedure to convert any sentence in first-order logic into clausal form. And we'll do a bunch of examples as we go through the procedure, just so that you know how it goes.

Converting to Clausal Form

1. Eliminate $\rightarrow$, $\leftrightarrow$

$$\alpha \rightarrow \beta \qquad\qquad \neg\alpha \vee \beta$$

The first step you guys know very well is to eliminate implications. So you know how to do that. Anywhere you see a A right arrow B, you just change it into not A or B.

Converting to Clausal Form

1. Eliminate $\rightarrow$, $\leftrightarrow$

$\alpha \rightarrow \beta$	$\neg\alpha \vee \beta$

2. Drive in $\neg$

$\neg(\alpha \vee \beta)$	$\neg\alpha$ Æ $\neg\beta$
$\neg(\alpha$ Æ $\beta)$	$\neg\alpha \vee \neg\beta$
$\neg\neg\alpha$	α
$\neg\exists x.\ P(x)$	$\forall x.\ \neg\ P(x)$
$\neg\forall x.\ P(x)$	$\exists x.\ \neg\ P(x)$

The next thing you do is to drive in negation. And you already basically know how to do that. We have deMorgan's laws to deal with conjunction and disjunction, and we can eliminate double negations.

As a kind of extension of deMorgan's laws, we also have that not exists x P(x) turns into forall x not P(x). And that not forall x P(x) turns into exists x such that not P(x).

Converting to Clausal Form

1. Eliminate $\rightarrow$, $\leftrightarrow$

$\alpha \rightarrow \beta$	$\neg\alpha \vee \beta$

2. Drive in $\neg$

$\neg(\alpha \vee \beta)$	$\neg\alpha$ Æ $\neg\beta$
$\neg(\alpha$ Æ $\beta)$	$\neg\alpha \vee \neg\beta$
$\neg\neg\alpha$	α
$\neg\exists x.\ P(x)$	$\forall x.\ \neg\ P(x)$
$\neg\forall x.\ P(x)$	$\exists x.\ \neg\ P(x)$

3. Rename variables apart

$$\forall x.\ \exists y.\ (P(x) \rightarrow \forall x.\ Q(x,y))$$
$$\forall x_1.\ \exists y_2.\ (P(x_1) \rightarrow \forall x_3.Q(x_3,y_2))$$

The next step is to rename variables apart. The idea here is that if every quantifier in your sentence should be over a different variable. So, if you had two different quantifications over x, you should rename one of them to use a different variable (which doesn't change the semantics at all).

In this example, we have two quantifications involving the variable x. It's especially confusing in this case, because they're nested. The rules are like those for a programming language: a variable is captured by the enclosing quantifier. So the x in Q(x,y) is really a different variable from the x in P(x). To make this distinction clear, and to automate the downstream processing into clausal form, we'll just rename each of the variables.

Converting to Clausal Form, II

4. Skolemize

Now, here's the step that some people find confusing. The name is already a good one. Step four is to skolemize, named after some logician named Skolem. Imagine that you have a sentence that looks like: there exists an X such that P of X. The goal here is to somehow arrive at a representation that doesn't have any quantifiers in it. Now, if we only had one kind of quantifier, it would be easy because we could just mention variables and all the variables would be implicitly quantified by the kind of quantifier that we have. But because we have two quantifiers, if we dropped all the quantifiers off, there's a mess, because you don't know which kind of quantification is supposed to apply to which variable.

Converting to Clausal Form, II

4. Skolemize
   - Substitute brand new name for each existentially quantified variable

     - $\exists x. P(x) \Rightarrow P(Fred)$

So, the Skolem insight is that when you have an existential quantification like this, you're saying there is such a thing as a unicorn, let's say that P is a unicorn. There exists a thing such that it's a unicorn. You can just say, all right, well, if there is one, let's call it Fred. That's it. That's what Skolemization is. So instead of writing exists an X such that P of X, you say P of Fred. The trick is that it absolutely must be a new name. It can't be any other name of any other thing that you know about. If you're in the process of inferring things about John and Mary, then it's not good to say, oh, there's a unicorn and it's John -- because that's kind of adding some information to the picture.

So to skolemize, in the simple case, means substitute brand-new name for each existentially quantified variable.

Converting to Clausal Form, II

4. Skolemize
   - Substitute brand new name for each existentially quantified variable

     - $\exists$ x. P(x) $\Rightarrow$ P(Fred)
     - $\exists$ x. P(x,y) $\Rightarrow$ P(X_{11}, Y_{13})

For example, if I have exists XY such that P of XY, then it's going to have to turn into P of X11, Y13. So if you have two different variables here, they have to be given different names.

Converting to Clausal Form, II

4. Skolemize
   - Substitute brand new name for each existentially quantified variable

     - $\exists$ x. P(x) $\Rightarrow$ P(Fred)
     - $\exists$ x. P(x,y) $\Rightarrow$ P(X_{11}, Y_{13})
     - $\exists$ x. P(x) Æ Q(x) $\Rightarrow$ P(Blue) Æ Q(Blue)

But the names also have to persist so that if you have exists an X such that P of X and Q of X, then if you skolemize that expression you should get P of Blue and Q of Blue. You make up a name and you put it in there, but every occurrence of this variable has to get mapped into that same unique name.

Converting to Clausal Form, II

4. Skolemize
   - Substitute brand new name for each existentially quantified variable

     - ∃ x. P(x) ⇒ P(Fred)
     - ∃ x. P(x,y) ⇒ P(X_{11}, Y_{13})
     - ∃ x. P(x) Æ Q(x) ⇒ P(Blue) Æ Q(Blue)
     - ∃ y. ∀ x. Loves(x,y)
     - ∀ x. ∃ y. Loves(x,y)

All right. If that's all we had to do it wouldn't be too bad. But there's one more case.

We can illustrate it by looking at these two interpretations of “Everyone loves someone”.

Converting to Clausal Form, II

4. Skolemize
   - Substitute brand new name for each existentially quantified variable
   - Substitute a new function of all universally quantified variables in enclosing scopes for each existentially quantified variable.
     - $\exists$ x. P(x) $\Rightarrow$ P(Fred)
     - $\exists$ x. P(x,y) $\Rightarrow$ P(X_{11}, Y_{13})
     - $\exists$ x. P(x) Æ Q(x) $\Rightarrow$ P(Blue) Æ Q(Blue)
     - $\exists$ y. $\forall$ x. Loves(x,y) $\Rightarrow$ $\forall$ x. Loves(x, Englebert)
     - $\forall$ x. $\exists$ y. Loves(x,y)

In the first case, there is a single y that everyone loves. So we do ordinary skolemization and decide to call that person Englebert.

In the second case, there is a different y, potentially, for each x. So, if we were just to substitute in a single constant name for y, we'd lose that information. We'd get the same result as above, which would be wrong.

So, when you are skolemizing an existential variable, you have to look at the other quantifiers that contain the one you're skolemizing, and instead of substituting in a new constant, you substitute in a brand new function symbol, applied to any variables that are universally quantified in an outer scope.

Converting to Clausal Form, II

4. Skolemize
   - Substitute brand new name for each existentially quantified variable
   - Substitute a new function of all universally quantified variables in enclosing scopes for each existentially quantified variable.
     - ∃ x. P(x) ⇒ P(Fred)
     - ∃ x. P(x,y) ⇒ P(X_{11}, Y_{13})
     - ∃ x. P(x) Æ Q(x) ⇒ P(Blue) Æ Q(Blue)
     - ∃ y. ∀ x. Loves(x,y) ⇒ ∀ x. Loves(x, Englebert)
     - ∀ x. ∃ y. Loves(x,y) ⇒ ∀ x. Loves(x, Beloved(x))

In this case, what that means is that you substitute in some function of x, for y. Let’s call it Beloved of x. Now it’s clear that the person who is loved by x depends on the particular x you’re talking about.

Converting to Clausal Form, II

4. Skolemize
   - Substitute brand new name for each existentially quantified variable
   - Substitute a new function of all universally quantified variables in enclosing scopes for each existentially quantified variable.
     - $\exists$ x. P(x) $\Rightarrow$ P(Fred)
     - $\exists$ x. P(x,y) $\Rightarrow$ P(X_{11}, Y_{13})
     - $\exists$ x. P(x) Æ Q(x) $\Rightarrow$ P(Blue) Æ Q(Blue)
     - $\exists$ y. $\forall$ x. Loves(x,y) $\Rightarrow$ $\forall$ x. Loves(x, Englebert)
     - $\forall$ x. $\exists$ y. Loves(x,y) $\Rightarrow$ $\forall$ x. Loves(x, Beloved(x))

5. Drop universal quantifiers

Now we can drop the universal quantifiers because we just replaced all the existential quantifiers with these skolem constants or functions and so now there's only one kind of quantifier left, so we can just drop them.

Converting to Clausal Form, II

4. Skolemize
   - Substitute brand new name for each existentially quantified variable
   - Substitute a new function of all universally quantified variables in enclosing scopes for each existentially quantified variable.
     - ∃ x. P(x) ⇒ P(Fred)
     - ∃ x. P(x,y) ⇒ P(X_{11}, Y_{13})
     - ∃ x. P(x) Æ Q(x) ⇒ P(Blue) Æ Q(Blue)
     - ∃ y. ∀ x. Loves(x,y) ⇒ ∀ x. Loves(x, Englebert)
     - ∀ x. ∃ y. Loves(x,y) ⇒ ∀ x. Loves(x, Beloved(x))

5. Drop universal quantifiers
6. Convert to CNF

And then we convert to conjunctive normal form. At this point, converting to conjunctive normal form just means multiplying out the and's and the or's, because we already eliminated the arrows and pushed in the negations.

Converting to Clausal Form, II

4. Skolemize
   - Substitute brand new name for each existentially quantified variable
   - Substitute a new function of all universally quantified variables in enclosing scopes for each existentially quantified variable.
     - $\exists$ x. P(x) $\Rightarrow$ P(Fred)
     - $\exists$ x. P(x,y) $\Rightarrow$ P(X_{11}, Y_{13})
     - $\exists$ x. P(x) Æ Q(x) $\Rightarrow$ P(Blue) Æ Q(Blue)
     - $\exists$ y. $\forall$ x. Loves(x,y) $\Rightarrow$ $\forall$ x. Loves(x, Englebert)
     - $\forall$ x. $\exists$ y. Loves(x,y) $\Rightarrow$ $\forall$ x. Loves(x, Beloved(x))

5. Drop universal quantifiers
6. Convert to CNF
7. Rename the variables in each clause
   - $\forall$ x. P(x) ÆQ(x) $\Rightarrow\forall$ y. P(y) Æ$\forall$ z. Q(z)

Finally, we can rename the variables in each clause. It's okay to do that because forall x P(x) and Q(x) is equivalent to forall y P(y) and forall z P(z). In fact, you don't really need to do this step, because we're assuming that you're always going to rename the variables before you do a resolution step.

Example: Converting to clausal form

So, let's do an example from the book, starting with English sentences, writing them down in first-order logic, converting to clausal form, and then finally doing a resolution proof.

Example: Converting to clausal form

a. John owns a dog
$\exists x. D(x) \wedge O(J,x)$

John owns a dog. We can write that in first-order logic as “there exists an x such that D(x) and O(J, x)”. So, we’re letting D stand for is-a-dog and O stand for owns and J stand for John.

Example: Converting to clausal form

a. John owns a dog
∃ x. D(x) Æ O(J,x)
D(Fido) Æ O(J, Fido)

Okay. To convert this to clausal form, we can start at step 4, skolemization, because the previous three steps are unnecessary for this sentence. Since we just have an existential quantifier over x, without any enclosing universal quantifiers, we can simply pick a new name and substitute it in for x. Let's call x "fido". This will give us two clauses with no variables, and we're done.

Example: Converting to clausal form

a. John owns a dog
∃ x. D(x) Æ O(J,x)
D(Fido) Æ O(J, Fido)

b. Anyone who owns a dog is a lover-of-animals
∀ x. (∃ y. D(y) Æ O(x,y)) → L(x)

Anyone who owns a dog is a lover of animals. We can write that in FOL as "For all x, if there exists a y such that D(y) and O(x,y), then L(x)." We've added a new predicate symbol L to stand for "is a lover of animals".

Example: Converting to clausal form

a. John owns a dog
∃ x. D(x) Æ O(J,x)
D(Fido) Æ O(J, Fido)

b. Anyone who owns a dog is a lover-of-animals
∀ x. (∃ y. D(y) Æ O(x,y)) → L(x)
∀ x. (¬∃ y. (D(y) Æ O(x,y)) v L(x)

First, we get rid of the arrow. Note that the parentheses are such that the existential quantifier is part of the antecedent, but the universal quantifier is not.

Example: Converting to clausal form

a. John owns a dog
$\exists x.\ D(x) \wedge O(J,x)$
$D(Fido) \wedge O(J, Fido)$

b. Anyone who owns a dog is a lover-of-animals
$\forall x.\ (\exists y.\ D(y) \wedge O(x,y)) \rightarrow L(x)$
$\forall x.\ (\neg\exists y.\ (D(y) \wedge O(x,y)) \vee L(x)$
$\forall x.\ \forall y.\ \neg(D(y) \wedge O(x,y)) \vee L(x)$
$\forall x.\ \forall y.\ \neg D(y) \vee \neg O(x,y) \vee L(x)$

Next, we drive in the negations. We'll do it in two steps.

Example: Converting to clausal form

a. John owns a dog
∃ x. D(x) Æ O(J,x)
D(Fido) Æ O(J, Fido)

b. Anyone who owns a dog is a lover-of-animals
∀ x. (∃ y. D(y) Æ O(x,y)) → L(x)
∀ x. (¬∃ y. (D(y) Æ O(x,y)) v L(x)
∀ x. ∀ y. ¬(D(y) Æ O(x,y)) v L(x)
∀ x. ∀ y. ¬ D(y) v ¬ O(x,y) v L(x)
¬ D(y) v ¬ O(x,y) v L(x)

There's no skolemization to do, since there aren't any existential quantifiers. So, we can just drop the universal quantifiers, and we're left with a single clause.

Example: Converting to clausal form

a. John owns a dog
$\exists x.\ D(x) \wedge O(J,x)$
$D(Fido) \wedge O(J, Fido)$

b. Anyone who owns a dog is a lover-of-animals
$\forall x.\ (\exists y.\ D(y) \wedge O(x,y)) \rightarrow L(x)$
$\forall x.\ (\neg\exists y.\ (D(y) \wedge O(x,y))\ v\ L(x)$
$\forall x.\ \forall y.\ \neg(D(y) \wedge O(x,y))\ v\ L(x)$
$\forall x.\ \forall y.\ \neg D(y)\ v\ \neg O(x,y)\ v\ L(x)$
$\neg D(y)\ v\ \neg O(x,y)\ v\ L(x)$

c. Lovers-of-animals do not kill animals
$\forall x.\ L(x) \rightarrow (\forall y.\ A(y) \rightarrow \neg K(x,y))$

Lovers of animals do not kill animals. We can write that in FOL as “For all x, if L(x) then for all y, A(y) implies not K(x,y)”. We’ve added the predicate symbol A to stand for “is an animal” and the predicate symbol K to stand for x kills y.

Example: Converting to clausal form

a. John owns a dog
∃ x. D(x) Æ O(J,x)
D(Fido) Æ O(J, Fido)

b. Anyone who owns a dog is a lover-of-animals
∀ x. (∃ y. D(y) Æ O(x,y)) → L(x)
∀ x. (¬∃ y. (D(y) Æ O(x,y)) v L(x)
∀ x. ∀ y. ¬(D(y) Æ O(x,y)) v L(x)
∀ x. ∀ y. ¬ D(y) v ¬ O(x,y) v L(x)
¬ D(y) v ¬ O(x,y) v L(x)

c. Lovers-of-animals do not kill animals
∀ x. L(x) → (∀ y. A(y) → ¬ K(x,y))
∀ x. ¬ L(x) v (∀ y. A(y) → ¬ K(x,y))
∀ x. ¬ L(x) v (∀ y. ¬ A(y) v ¬ K(x,y))

First, we get rid of the arrows, in two steps.

Example: Converting to clausal form

a. John owns a dog

∃ x. D(x) Æ O(J,x)

D(Fido) Æ O(J, Fido)

b. Anyone who owns a dog is a lover-of-animals

∀ x. (∃ y. D(y) Æ O(x,y)) → L(x)

∀ x. (¬∃ y. (D(y) Æ O(x,y)) v L(x)

∀ x. ∀ y. ¬(D(y) Æ O(x,y)) v L(x)

∀ x. ∀ y. ¬ D(y) v ¬ O(x,y) v L(x)

¬ D(y) v ¬ O(x,y) v L(x)

c. Lovers-of-animals do not kill animals

∀ x. L(x) → (∀ y. A(y) → ¬ K(x,y))

∀ x. ¬ L(x) v (∀ y. A(y) → ¬ K(x,y))

∀ x. ¬ L(x) v (∀ y. ¬ A(y) v ¬ K(x,y))

¬ L(x) v ¬ A(y) v ¬ K(x,y)

Then we're left with only universal quantifiers, which we drop, yielding one clause.

More converting to clausal form

d. Either Jack killed Tuna or curiosity killed Tuna
K(J,T) v K(C,T)

We just have three more easy ones. "Either Jack killed Tuna or curiosity killed Tuna." Everything here is a constant, so we get K(J,T) or K(C,T).

More converting to clausal form

d. Either Jack killed Tuna or curiosity killed Tuna
K(J,T) v K(C,T)

e. Tuna is a cat
C(T)

"Tuna is a cat" just turns into C(T).

More converting to clausal form

d. Either Jack killed Tuna or curiosity killed Tuna
K(J,T) v K(C,T)

e. Tuna is a cat
C(T)

f. All cats are animals
¬ C(x) v A(x)

And "All cats are animals" is not C(x) or A(x). I left out the steps here, but I'm sure you can fill them in.

Curiosity Killed the Cat

	D(Fido)	a
	O(J,Fido)	a
	¬ D(y) v ¬ O(x,y) v L(x)	b
	¬ L(x) v ¬ A(y) v ¬ K(x,y)	c
	K(J,T) v K(C,T)	d
	C(T)	e
	¬ C(x) v A(x)	f

Given all these premises, we're interested in proving that curiosity killed the cat. So, first, we start our proof by entering all of the clauses from the premises on lines 1 – 7.

Curiosity Killed the Cat

	D(Fido)	a
	O(J,Fido)	a
	¬ D(y) v ¬ O(x,y) v L(x)	b
	¬ L(x) v ¬ A(y) v ¬ K(x,y)	c
	K(J,T) v K(C,T)	d
	C(T)	e
	¬ C(x) v A(x)	f
	¬ K(C,T)	Neg

Now, we add the negation of the conclusion, which is not K(C,T), and start doing the proof.

Curiosity Killed the Cat

	D(Fido)	a
	O(J,Fido)	a
	¬ D(y) v ¬ O(x,y) v L(x)	b
	¬ L(x) v ¬ A(y) v ¬ K(x,y)	c
	K(J,T) v K(C,T)	d
	C(T)	e
	¬ C(x) v A(x)	f
	¬ K(C,T)	Neg
	K(J,T)	5,8

We can apply the resolution rule to any pair of lines that contain unifiable literals. Here's one way to do the proof. We'll use the "set-of-support" heuristic (which says we should involve the negation of the conclusion in the proof), and resolve away K(C,T) from lines 5 and 8, yielding K(J,T).

Curiosity Killed the Cat

	D(Fido)	a
	O(J,Fido)	a
	¬ D(y) v ¬ O(x,y) v L(x)	b
	¬ L(x) v ¬ A(y) v ¬ K(x,y)	c
	K(J,T) v K(C,T)	d
	C(T)	e
	¬ C(x) v A(x)	f
	¬ K(C,T)	Neg
	K(J,T)	5,8
	A(T)	6,7 {x/T}

Then, we can resolve C(T) and C(x) in lines 6 and 7 by substituting T for x, and getting A(T).

Curiosity Killed the Cat

	D(Fido)	a
	O(J,Fido)	a
	¬ D(y) v ¬ O(x,y) v L(x)	b
	¬ L(x) v ¬ A(y) v ¬ K(x,y)	c
	K(J,T) v K(C,T)	d
	C(T)	e
	¬ C(x) v A(x)	f
	¬ K(C,T)	Neg
	K(J,T)	5,8
	A(T)	6,7 {x/T}
	¬ L(J) v ¬ A(T)	4,9 {x/J, y/T}

Using lines 4 and 9, and substituting J for x and T for Y, we get not L(J) or not A(T).

Curiosity Killed the Cat

	D(Fido)	a
	O(J,Fido)	a
	¬ D(y) v ¬ O(x,y) v L(x)	b
	¬ L(x) v ¬ A(y) v ¬ K(x,y)	c
	K(J,T) v K(C,T)	d
	C(T)	e
	¬ C(x) v A(x)	f
	¬ K(C,T)	Neg
	K(J,T)	5,8
	A(T)	6,7 {x/T}
	¬ L(J) v ¬ A(T)	4,9 {x/J, y/T}
	¬ L(J)	10,11

From lines 10 and 11, we get not L(J).

Curiosity Killed the Cat

	D(Fido)	a
	O(J,Fido)	a
	¬ D(y) v ¬ O(x,y) v L(x)	b
	¬ L(x) v ¬ A(y) v ¬ K(x,y)	c
	K(J,T) v K(C,T)	d
	C(T)	e
	¬ C(x) v A(x)	f
	¬ K(C,T)	Neg
	K(J,T)	5,8
	A(T)	6,7 {x/T}
	¬ L(J) v ¬ A(T)	4,9 {x/J, y/T}
	¬ L(J)	10,11
	¬ D(y) v ¬ O(J,y)	3,12 {x/J}

From 3 and 12, substituting J for X, we get not D(y) or not O(J,y).

Curiosity Killed the Cat

	¬ D(Fido)	a
	O(J,Fido)	a
	¬ D(y) v ¬ O(x,y) v L(x)	b
	¬ L(x) v ¬ A(y) v ¬ K(x,y)	c
	K(J,T) v K(C,T)	d
	C(T)	e
	¬ C(x) v A(x)	f
	¬ K(C,T)	Neg
	K(J,T)	5,8
	A(T)	6,7 {x/T}
	¬ L(J) v ¬ A(T)	4,9 {x/J, y/T}
	¬ L(J)	10,11
	¬ D(y) v ¬ O(J,y)	3,12 {x/J}
	¬ D(Fido)	13,2 {x/Fido}

From 13 and 2, substituting Fido for x, we get not D(Fido).

Curiosity Killed the Cat

	D(Fido)	a
	O(J,Fido)	a
	¬ D(y) v ¬ O(x,y) v L(x)	b
	¬ L(x) v ¬ A(y) v ¬ K(x,y)	c
	K(J,T) v K(C,T)	d
	C(T)	e
	¬ C(x) v A(x)	f
	¬ K(C,T)	Neg
	K(J,T)	5,8
	A(T)	6,7 {x/T}
	¬ L(J) v ¬ A(T)	4,9 {x/J, y/T}
	¬ L(J)	10,11
	¬ D(y) v ¬ O(J,y)	3,12 {x/J}
	¬ D(Fido)	13,2 {x/Fido}
	•	14,1

And finally, from lines 13 and 2, we derive a contradiction. Yay! Curiosity **did** kill the cat.

Proving validity

How do we use resolution refutation to prove something is valid?

So, if we want to use resolution refutation to prove that something is valid, what would we do? What do we normally do when we do a proof using resolution refutation?

Proving validity

How do we use resolution refutation to prove something is valid?

Normally, we prove a sentence is entailed by the set of axioms

We say, well, if I know all these things, I can prove this other thing I want to prove. We prove that the premises entail the conclusion.

Proving validity

How do we use resolution refutation to prove something is valid?

Normally, we prove a sentence is entailed by the set of axioms

Valid sentences are entailed by the empty set of sentences

- ϕ is valid
- $\{ \} \models \phi$ [empty set of sentences entails ϕ]
- $\{ \} \vdash \phi$ [empty set of sentences proves ϕ]

What does it mean for a sentence to be valid, in the language of entailment? That it's true in all interpretations. What that means really is that it should be derivable from nothing. A valid sentence is entailed by the empty set of sentences. The valid sentence is true no matter what. So we're going to prove something with no assumptions.

Proving validity

How do we use resolution refutation to prove something is valid?

Normally, we prove a sentence is entailed by the set of axioms

Valid sentences are entailed by the empty set of sentences

- ϕ is valid
- $\{\ \} \models \phi$ [empty set of sentences entails ϕ]
- $\{\ \} \vdash \phi$ [empty set of sentences proves ϕ]

To prove validity by refutation, negate the sentence and try to derive contradiction.

We can prove it by resolution refutation by negating the sentence and trying to derive a contradiction.

Proving validity: example

Prove validity of:
$\exists x. (P(x) \rightarrow P(A)) \text{ Æ } (P(x) \rightarrow P(B))$

So, let’s do an example. Imagine that we would like to show the validity of this sentence.

Proving validity: example

Prove validity of:
$\exists x. (P(x) \rightarrow P(A))\ \text{Æ}\ (P(x) \rightarrow P(B))$

$\neg \exists x. (P(x) \rightarrow P(A))\ \text{Æ}\ (P(x) \rightarrow P(B))$
$\neg \exists x. ((\neg P(x) \vee P(A))\ \text{Æ}\ (\neg P(x) \vee P(B))$
$\forall x. \neg((\neg P(x) \vee P(A))\ \text{Æ}\ (\neg P(x) \vee P(B))$
$\forall x. \neg(\neg P(x) \vee P(A)) \vee \neg(\neg P(x) \vee P(B))$
$\forall x. (P(x)\ \text{Æ}\ \neg P(A)) \vee (P(x)\ \text{Æ}\ \neg P(B))$

We start by negating it and converting to clausal form. It takes quite a few steps to drive in all the negations, but eventually we end up with this universally quantified statement.

Proving validity: example

Prove validity of:
∃ x. (P(x) → P(A)) Æ (P(x) → P(B))

¬ ∃ x. (P(x) → P(A)) Æ (P(x) → P(B))
¬ ∃ x. ((¬ P(x) v P(A)) Æ (¬ P(x) v P(B))
∀ x. ¬((¬ P(x) v P(A)) Æ (¬ P(x) v P(B))
∀ x. ¬(¬ P(x) v P(A)) v ¬(¬ P(x) v P(B))
∀ x. (P(x) Æ ¬ P(A)) v (P(x) Æ ¬ P(B))
(P(x) Æ ¬ P(A)) v (P(x) Æ ¬ P(B))

Since there are no other quantifiers, we can just drop the universals.

Proving validity: example

Prove validity of:
∃ x. (P(x) → P(A)) Æ (P(x) → P(B))

¬ ∃ x. (P(x) → P(A)) Æ (P(x) → P(B))
¬ ∃ x. ((¬ P(x) v P(A)) Æ (¬ P(x) v P(B))
∀ x. ¬((¬ P(x) v P(A)) Æ (¬ P(x) v P(B))
∀ x. ¬(¬ P(x) v P(A)) v ¬(¬ P(x) v P(B))
∀ x. (P(x) Æ ¬ P(A)) v (P(x) Æ ¬ P(B))
(P(x) Æ ¬ P(A)) v (P(x) Æ ¬ P(B))
(P(x) v P(x)) Æ (P(x) v ¬ P(B)) Æ (¬ P(A) v P(x)) Æ (¬ P(A) v ¬ P(B))

And now all we have to do is distribute, to get these four clauses.

Proving validity: example

Prove validity of:
∃ x. (P(x) → P(A)) Æ (P(x) → P(B))

¬ ∃ x. (P(x) → P(A)) Æ (P(x) → P(B))
¬ ∃ x. ((¬ P(x) v P(A)) Æ (¬ P(x) v P(B))
∀ x. ¬((¬ P(x) v P(A)) Æ (¬ P(x) v P(B))
∀ x. ¬(¬ P(x) v P(A)) v ¬(¬ P(x) v P(B))
∀ x. (P(x) Æ ¬ P(A)) v (P(x) Æ ¬ P(B))
(P(x) Æ ¬ P(A)) v (P(x) Æ ¬ P(B))
(P(x) v P(x)) Æ (P(x) v ¬ P(B)) Æ (¬ P(A) v P(x)) Æ (¬ P(A) v ¬ P(B))

	P(x)	
	P(x) v ¬ P(B)	
	¬ P(A) v P(x)	
	¬ P(A) v ¬ P(B)	

We enter the clauses into our proof.

Proving validity: example

Prove validity of:
∃ x. (P(x) → P(A)) Æ (P(x) → P(B))

¬ ∃ x. (P(x) → P(A)) Æ (P(x) → P(B))
¬ ∃ x. ((¬ P(x) v P(A)) Æ (¬ P(x) v P(B))
∀ x. ¬((¬ P(x) v P(A)) Æ (¬ P(x) v P(B))
∀ x. ¬(¬ P(x) v P(A)) v ¬(¬ P(x) v P(B))
∀ x. (P(x) Æ ¬ P(A)) v (P(x) Æ ¬ P(B))
(P(x) Æ ¬ P(A)) v (P(x) Æ ¬ P(B))
(P(x) v P(x)) Æ (P(x) v ¬ P(B)) Æ (¬ P(A) v P(x)) Æ (¬ P(A) v ¬ P(B))

1	P(x)	
2	P(x) v ¬ P(B)	
3	¬ P(A) v P(x)	
4	¬ P(A) v ¬ P(B)	
5	¬ P(B)	1,4 {x/A}

Now, we can resolve lines 1 and 4, substituting A for x, to get not P(B).

Proving validity: example

Prove validity of:
∃ x. (P(x) → P(A)) Æ (P(x) → P(B))

¬ ∃ x. (P(x) → P(A)) Æ (P(x) → P(B))
¬ ∃ x. ((¬ P(x) v P(A)) Æ (¬ P(x) v P(B))
∀ x. ¬((¬ P(x) v P(A)) Æ (¬ P(x) v P(B))
∀ x. ¬(¬ P(x) v P(A)) v ¬(¬ P(x) v P(B))
∀ x. (P(x) Æ ¬ P(A)) v (P(x) Æ ¬ P(B))
(P(x) Æ ¬ P(A)) v (P(x) Æ ¬ P(B))
(P(x) v P(x)) Æ (P(x) v ¬ P(B)) Æ (¬ P(A) v P(x)) Æ (¬ P(A) v ¬ P(B))

	P(x)	
	P(x) v ¬ P(B)	
	¬ P(A) v P(x)	
	¬ P(A) v ¬ P(B)	
	¬ P(B)	1,4 {x/A}
	•	1,5 {x/B}

And we can resolve 1 and 5, substituting B for x, to get a contradiction.

Recitation Problems

In each group, derive the last sentence from the others using resolution refutation.

∀ xy.F(x,y)
∀ xy.F(y,x)

∀ x.F(x)→ (G(x) Ç H(x))
G(A)↔ (H(A)Æ ¬ G(A))
¬ F(A)

∀ x.F(x)Ç G(x)
∃ x.¬ G(x)
∀ x.H(x)→ ¬ F(x)
∃ x.¬ H(x)

∃ x.F(x)
∃ y.F(y)

∀ xyz.F(x,y)Æ F(y,z)→ F(x,z)
¬ F(x,x)
∀ xy.F(x,y)→ ¬ F(y,x)

∀ x.∃ y.L(x,y)
∀ xy.L(x,y)→ H(x)
∀ x.H(x)

A Silly Recitation Problem

Symbolize the following argument, and then derive the conclusion from the premises using resolution refutation.

- Nobody who really appreciates Beethoven fails to keep silence while the Moonlight sonata is being played.
- Guinea pigs are hopelessly ignorant of music.
- No one who is hopelessly ignorant of music ever keeps silence while the moonlight sonata is being played.
- Therefore, guinea pigs never really appreciate Beethoven.

(Taken from a book by Lewis Carroll, logician and author of *Alice in Wonderland*.)

Another, Sillier Problem

You don’t have to do this one. It’s just for fun. Same type as the previous one. Also from Lewis Carroll.

- The only animals in this house are cates
- Every animal that loves to gaze at the moon is suitable for a pet
- When I detest an animal, I avoid it
- No animals are carnivorous unless they prowl at night
- No cat fails to kill mice
- No animals ever like me, except those that are in this house
- Kangaroos are not suitable for pets
- None but carnivorous animals kill mice
- I detest animals that do not like me
- Animals that prowl at night always love to gaze at the moon
- Therefore, I always avoid a kangaroo