Using first order logic (Ch. 9)

If you assume contradictory axioms, you can derive anything. It's called the principle of explosion.

Hey, you're right! I started with $P \land \neg P$ and derived your mom's phone number! That's not how that works.

Mrs. Lenhart?

Wait, this is her number! How-

Hi, I'm a friend of- why, yes, I am free tonight!

Mom!

No, box wine sounds lovely!
Announcements

Writing 2 graded (was last Thurs but I forgot to announce)
- Regrade deadline: Dec. 5

Writing 4 due on Sunday (need to decide project)
Resolution in FO logic

You try it!
1. Use logical equivalence to remove implies
2. Move logical negation next to relations
3. Standardize variables
4. Generalize existential quantifiers
5. Drop universal quantifiers
6. Distribute ORs over ANDs

Convert this to CNF:
\[ \forall x \ A(x) \iff \forall y \ B(x, y) \]
Resolution in FO logic

\[ \forall x \ A(x) \iff \forall y \ B(x, y) \]

1. \((\forall x \ A(x) \implies \forall y \ B(x, y)) \land (\forall x \ \forall y \ B(x, y) \implies A(x))\)

1. \((\forall x \neg A(x) \lor \forall y \ B(x, y)) \land (\forall x \neg \forall y \ B(x, y) \lor A(x))\)

2. \((\forall x \neg A(x) \lor \forall y \ B(x, y)) \land (\forall x \ \exists y \neg B(x, y) \lor A(x))\)

3. (nothing to do)

4. \((\forall x \neg A(x) \lor \forall y \ B(x, y)) \land (\forall x \neg B(x, Y(x)) \lor A(x))\)

5. \((\neg A(x) \lor B(x, y)) \land (\neg B(x, Y(x)) \lor A(x))\)

6. (nothing to do)

The negation goes where show in the blue box, because y is localized to one side, while not x
Resolution in FO logic

Resolution is **refutation-complete** in first-order logic (due to it being semi-decidable)

So using resolution we can tell if: “a entails b”

But we cannot tell if: “a does not entail b”

Resolution recap:
PL: complete, can do “entails” and “not entail”
FOL: refutation-complete, only does “entails”
Resolution in FO logic

Consider this KB:

\[ A(Dog) \lor A(Cat) \]
\[ \neg A(Dog) \]
\[ \forall x \ A(x) \Rightarrow B(x) \]

If we ask: B(Cat)?

\[ A(Dog) \lor A(Cat) \]
\[ \neg A(Dog) \]
\[ \forall x \ \neg A(x) \lor B(x) \]
\[ \neg B(Cat) \]

unify \{x/Cat\}

Contradiction!

KB entails B(Cat)
Resolution in FO logic

The last example worked correctly as it identified entailment. However, it has trouble giving us answers to existentials: Ask “exists x, A(x)”?

\[\begin{align*}
A(Dog) \lor A(Cat) \\
\neg A(Dog) \\
\forall x \quad \neg A(x) \lor B(x) \\
\forall x \quad \neg A(x)
\end{align*}\]

This only tells us (2 unify): A(Cat) OR A(Dog)
Thus, resolution in first-order logic will always tell you if a sentence is entailed. However, it might not be able to tell you for what values it is satisfiable.

Similar to the semi-decidable nature of FO logic, resolution is complete if entailment can be found in a finite number of inferences (or “resolves”).
Once again, I have avoided equality as it is not much fun to deal with.

Two ways to deal with this are:
1. Add rules of equality to KB
2. De/Para-modulation (i.e. more substituting)

Both can increase the complexity of the KB or inference by a large amount, so it is better to just avoid equality if possible.
Resolution and equality

There are three basic rules of equality:

1. reflexive: \( \forall x \ x = x \)
2. symmetric: \( \forall x, y \ x = y \Rightarrow y = x \)
3. transitive: \( \forall x, y, z \ x = y \land y = z \Rightarrow x = z \)

Then **for each relation/function** we have to add an explicit statement:

Relations (1 var): \( \forall x, y \ x = y \Rightarrow A(x) \iff A(y) \)

Functions (2 vars): (\( \Rightarrow \) instead of *iff*)
\( \forall a, b, x, y \ a = x \land b = y \Rightarrow F(a, b) = F(x, y) \)
Resolution and equality

Consider this KB: \( A(x) \lor B(x, F(x)) \)
\[ \forall x, y \ x = y \Rightarrow B(x, y) \]

Would need to be converted into:

\[ \forall x \ x = x \]
\[ \forall x, y \ x = y \Rightarrow y = x \]
\[ \forall x, y, z \ x = y \land y = z \Rightarrow x = z \]
\[ \forall a, x \ a = x \Rightarrow [A(a) \iff A(x)] \]
\[ \forall a, b, x, y \ a = x \land b = y \Rightarrow [B(a, b) \iff B(x, y)] \]
\[ \forall a, x \ a = x \Rightarrow F(a) = F(x) \]
\[ A(x) \lor B(x, F(x)) \]
\[ \forall x, y \ x = y \Rightarrow B(x, y) \]
Resolution and equality

Consider this KB:

\[ A(x) \lor B(x, F(x)) \]
\[ \forall x, y \ x = y \implies B(x, y) \]

Basically, you convert = into a relationship

\[ \forall x \ Eq(x, x) \]
\[ \forall x, y \ Eq(x, y) \implies Eq(y, x) \]
\[ \forall x, y, z \ Eq(x, y) \land Eq(y, z) \implies Eq(x, z) \]
\[ \forall a, x \ Eq(a, x) \implies [A(a) \iff A(x)] \]
\[ \forall a, b, x, y \ Eq(a, x) \land Eq(b, y) \implies [B(a, b) \iff B(x, y)] \]
\[ \forall a, x \ Eq(a, x) \implies Eq(F(a), F(x)) \]
\[ A(x) \lor B(x, F(x)) \]
\[ \forall x, y \ Eq(x, y) \implies B(x, y) \]
Resolution and equality

The second option doubles the available inferences instead of doubling the KB.

We allow paramodulation, in addition to the normal resolution rule.

Paramodulation is essentially substituting with a sentence that contains an equals, while also applying resolution to combine (and ensures there is no conflict in the KB).
Resolution and equality

Consider this KB:

\[ A(x) \lor B(F(x, Cat)) \lor C(x, Cat) \]
\[ [F(Dog, y) = G(y)] \lor D(y) \]

We can then unify \{x/Dog, y/Cat\} and get:

\[ A(Dog) \lor B(F(Dog, Cat)) \lor C(Dog, Cat) \]
\[ [F(Dog, Cat) = G(Cat)] \lor D(Cat) \]

Which we can infer:

\[ A(Dog) \lor B(G(Cat)) \lor C(Dog, Cat) \lor D(Cat) \]

1. Like resolution you combine sentences
2. Valid substitutions if necessary
Consider this KB:

\[ A(x) \lor B(F(x, Cat)) \lor C(x, Cat) \]

\[ [F(Dog, y) = G(y)] \lor D(y) \]

We can then unify \{x/Dog, y/Cat\} and get:

\[ A(Dog) \lor B(F(Dog, Cat)) \lor C(Dog, Cat) \]

\[ F(Dog, Cat) = G(Cat) \lor D(Cat) \]

Which we can infer:

\[ A(Dog) \lor B(G(Cat)) \lor C(Dog, Cat) \lor D(Cat) \]

1. Like resolution you combine sentences
2. Valid substitutions if necessary
Resolution efficiency

Four (brief) ways to speed up resolution:
1. Subsumption
2. Unit preference
3. Support set
4. Input resolution

1. and 2. are general and do not effect the completeness of resolution

3. and 4. can limit resolvability
Subsumption is to remove any sentences that are fully expressed by another sentence.

Consider this KB: \( \forall x \ A(x) \quad A(Cat) \)

The first sentence is more general and the second is not adding anything.

We could simply reduce the KB to: \( \forall x \ A(x) \) (and keep the same meaning).
Resolution efficiency

**Unit preference** is to always apply a clause containing one literal before any others.

Since we want to end up with an empty clause for a contradiction, this will shrink the size of the original clause.

For example: \(((A(x) \lor B(x) \lor C(x)) \land (\neg A(x)))\) ...

... will resolve to: \((B(x) \lor C(x))\)
A Support set is artificially restricting the KB and removing (what you think are) irrelevant clauses

The set of clauses you use can be based on the query, so if we have this KB:

\[ A(x) \implies B(x) \]
\[ B(x) \implies C(x) \]
\[ \exists x \ A(x) \]

Then we ask: \( \exists x \ B(x) \)?

We can see the middle sentence is worthless, so we can solve it just with the first and third
If the support set contains no equalities, there will be a large efficiency increase.

However, if the support set does not contain an important sentence you can reach an incorrect conclusion (about entailment).

Even without equality, eliminating a portion of the KB can give large speed ups (as inference is NP-hard, i.e. exponential).
Resolution efficiency

Input resolution starts with a single sentence, and only tries to apply resolution to that sentence (and the resulting sentences).

The resolution of this earlier example is one:

\[ A(Dog) \lor A(Cat) \]
\[ \neg A(Dog) \]
\[ \forall x \neg A(x) \lor B(x) \]
\[ \neg B(Cat) \]

The blue line is involved in all resolutions.

Contradiction!