Foundations of Artificial Intelligence

CS472/3
Lecture #22

Bart Selman

Slide CS472–1

Today's Lecture

**Inference**
**Resolution / Unification**
Chapter 8 & 9, R&N.

Slide CS472–2

## Inference

We've considered first-order logic.

But, how do we reason with it? Derive new info?

**A.** Use **resolution** as in propositional case

From $(\alpha \vee p) \wedge (\neg p \vee \beta)$
conclude $\alpha \vee \beta$ until you reach contradiction.

Need some extra "tricks" to deal with
**quantifiers** and **variables.**



## Resolution

I **put in clausal form**
all variables universally quantified
main trick: "Skolemization" to remove existantials.
idea: invent names for unkown objects known to exist

II **use unification** to match atomic sentences

III **apply resolution rule** to the clausal set combined
with negated goal. Attempt to generate empty clause.

**Tricks**

- **unification**: needed to match variables and terms
  between clauses that look similar
  See R&N pp. 270-271.
- **normalization**: put in **clausal form**
  move quantifiers / $\wedge$ / $\vee$ etc.
  and **Skolemization** — remove $\exists$ by giving
  an arbitrary, but unique name to the object in question.
  E.g. $D$ for the dog owned by Jack.
  See R&N pp. 281-282.

**Slide CS472–5**

**Unification**

UNIFY (P,Q) takes two atomic sentences P and Q and returns a substitution that makes P and Q **look the same.**

Rules for substitutions:

- Can replace a variable by a constant.
- Can replace a variable by a variable.
- Can replace a variable by a function expression, as long as the function expression does not contain the variable.

**Unifier**: a substitution that makes two clauses resolvable.
$v_1 \rightarrow C; v_2 \rightarrow v_3; v_4 \rightarrow f(...)$

**Slide CS472–6**

**Unification — Purpose**

Given:

$Knows(John, x) \rightarrow Hates(John, x)$
$Knows(John, Jim)$

Derive

$Hates(John, Jim)$

Need **unifier** $\{x/Jim\}$ before resolution.
(simplest case)



$\neg Knows(John, x) \vee Hates(John, x)$ and $Knows(John, Jim)$

How do we resolve? First, match them.
Solution:

UNIFY($Knows(John, x)$,$Knows(John, Jim)$) $= \{x/Jim\}$

Gives

$\neg Knows(John, Jim) \vee Hates(John, Jim)$ and
$Knows(John, Jim)$

Conclude by resolution

$Hates(John, Jim)$

## Unification (example)

general rule:

$Knows(John, x) \rightarrow Hates(John, x)$

facts:

$Knows(John, Jim)$

$Knows(y, Leo)$

$Knows(y, Mother(y))$

$Knows(x, Jane)$

"matching facts to general rules"



UNIFY($Knows(John, x)$,$Knows(John, Jim)$) $= \{x/Jim\}$

UNIFY($Knows(John, x)$,$Knows(y, Leo)$) $= \{x/Leo, y/John\}$

UNIFY($Knows(John, x)$,$Knows(y, Mother(y))$) $=$
$\{y/John, x/Mother(John)\}$;

UNIFY($Knows(John, x)$,$Knows(x, Jane)$) $= fail$

- Last one fails because x can't take on both the value John **and** the value Jane. But intuitively we know that everyone John knows he hates and everyone knows Jane so we should be able to infer that John hates Jane.
- This is why we require, if possible, that every variable has a separate name. Knows(John,x) and Knows(y,Jane) works.



### Most General Unifier

In cases where there is more than one substitution choose the one that makes the least commitment (most general) about the bindings.

UNIFY($Knows(John, x), Knows(y, z)$)

$= \{y/John, x/z\}$

or $\{y/John, x/z, z/Freda\}$

or $\{y/John, x/John, z/John\}$

or ....

See page 303 for general unification algorithm. $O(n^2)$

## Normal form: Clausal



See pages 281 & 282, R&N.

- **Eliminate implication**

  $p \Rightarrow q$ becomes $\neg p \vee q$
- **Move $\neg$ inwards**

  e.g., $\neg(p \vee q)$ becomes $(\neg p \wedge \neg q)$

  $\neg \exists x \,.\, p$ becomes $\forall x \; \neg p$

  $\neg \forall x \,.\, p$ becomes $\ldots$
- **Standarize variables**

  rename variables to avoid conflicts.
- **Move quantifiers left**

  e.g., $p \vee \forall x \; q$ becomes $\forall x \; (p \vee q)$

- **Skolemize** (remove existentials)
  e.g. $\forall x\ Person(x) \Rightarrow \exists y\ Heart(y) \wedge Has(x, y)$
  consider:
  $\forall x\ Person(x) \Rightarrow Heart(H) \wedge Has(x, H)$
  problem??
  $\forall x\ Person(x) \Rightarrow Heart(F(x)) \wedge Has(x, F(x))$
- **Distribute $\wedge$ over $\vee$**
  $(a \wedge b) \vee c$ becomes $(a \vee c) \wedge (b \vee c)$
- **Flatten nested conjunctions and disjunctions**
  e.g. $(a \vee b) \vee c$ becomes $(a \vee b \vee c)$



**Example**

Jack owns a dog.

Every dog owner is an animal lover.

No animal lover kills an animal.

Either Jack or Curiosity killed the cat, who is named Tuna.

Did Curiosity kill the cat?

**Original Sentences (Plus Background Knowledge)**

1. $\exists x : Dog(x) \wedge Owns(Jack, x)$
2. $\forall x \ \ (\exists y \ \ Dog(y) \wedge Owns(x, y)) \rightarrow AnimalLover(x)$
3. $\forall x \ \ AnimalLover(x) \rightarrow \forall y \ \ Animal(y) \rightarrow \neg Kills(x, y)$
4. $Kills(Jack, Tuna) \vee Kills(Curiosity, Tuna)$
5. $Cat(Tuna)$
6. $\forall x \ \ Cat(x) \rightarrow Animal(x)$

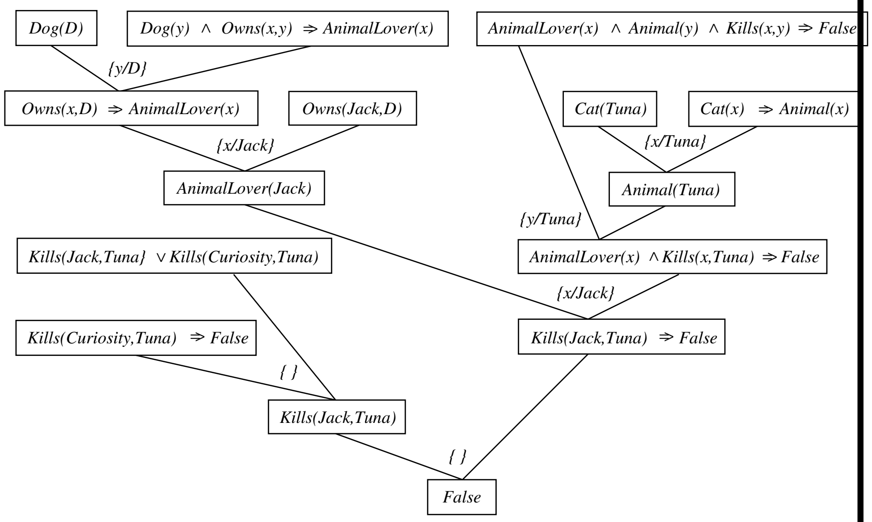
**Slide CS472–17**

**Clausal Form**

1. $Dog(D)$ (D is the function that finds Jack's dog)
2. $Owns(Jack, D)$
3. $\neg Dog(S(x)) \vee \neg Owns(x, S(x)) \vee AnimalLover(x)$
4. $\neg AnimalLover(w) \vee \neg Animal(y) \vee \neg Kills(w, y)$
5. $Kills(Jack, Tuna) \vee Kills(Curiosity, Tuna)$
6. $Cat(Tuna)$
7. $\neg Cat(z) \vee Animal(z)$

**Slide CS472–18**

## Proof by Resolution with Refutation




Slide CS472–19

## Completeness

**Resolution** with **unification** applied to **clausal form**, is **refutation** complete.

Interesting proof! Based on building an "artificial" domain of interpretation, called the **Herbrand universe**. See Section 9.7 R&N.

Slide CS472–20

## Practice

Complete in principle, in practice still significant
search problem!

Many different search strategies: **resolution strategies**