

# First-order Logic

[RN2] Sec 7.1-7.6 Chap 8-9  
[RN3] Sec 7.1-7.6 Chap 8-9

CS 486/686  
University of Waterloo  
Lecture 20: November 15, 2012

1

CS486/686 Lecture Slides (c) 2012 P. Poupart

## Outline

- First-order logic
  - Syntax and semantics
- Inference
  - Propositionalization with ground inference
  - Lifted resolution

2

CS486/686 Lecture Slides (c) 2012 P. Poupart

# Logic

- General language
  - Can encode any kind of deterministic and discrete knowledge
  - Well defined syntax and semantics
  - Knowledge base: store domain knowledge
- General algorithms
  - Search by inference
  - Can infirm or confirm conclusions

3

CS486/686 Lecture Slides (c) 2012 P. Poupart

# Syntax of a very simple logic

- Sentence  $\rightarrow$  AtomicSentence | ComplexSentence
- AtomicSentence  $\rightarrow$  True | False | Symbol
- Symbol  $\rightarrow$  P | Q | R | ...
- ComplexSentence  $\rightarrow$   $\neg$ Sentence
  - | (Sentence  $\wedge$  Sentence )
  - | (Sentence  $\vee$  Sentence)
  - | (Sentence  $\Rightarrow$  Sentence )
  - | (Sentence  $\Leftrightarrow$  Sentence)

Symbols are binary variables

4

CS486/686 Lecture Slides (c) 2012 P. Poupart

## Logical connector semantics

P	Q	$\neg P$	$P \wedge Q$	$P \vee Q$	$P \Rightarrow Q$	$P \Leftrightarrow Q$
false	false	true	false	false	true	true
false	true	true	false	true	true	false
true	false	false	false	true	false	false
true	true	false	true	true	true	true

5

CS486/686 Lecture Slides (c) 2012 P. Poupart

## Recall the Street Puzzle

1 2 3 4 5

$N_i = \{\text{English, Spaniard, Japanese, Italian, Norwegian}\}$

$C_i = \{\text{Red, Green, White, Yellow, Blue}\}$

$D_i = \{\text{Tea, Coffee, Milk, Fruit-juice, Water}\}$

$J_i = \{\text{Painter, Sculptor, Diplomat, Violinist, Doctor}\}$

$A_i = \{\text{Dog, Snails, Fox, Horse, Zebra}\}$

The Englishman lives in the Red house

The Spaniard has a Dog

The Japanese is a Painter

The Italian drinks Tea

The Norwegian lives in the first house on the left

The owner of the Green house drinks Coffee

The Green house is on the right of the White house

The Sculptor breeds Snails

The Diplomat lives in the Yellow house

The owner of the middle house drinks Milk

The Norwegian lives next door to the Blue house

The Violinist drinks Fruit juice

The Fox is in the house next to the Doctor's

The Horse is next to the Diplomat's

Who owns the Zebra?  
Who drinks Water?

6

CS486/686 Lecture Slides (c) 2012 P. Poupart

## Example - Street Puzzle

- Symbols:  $V_{in}, V_{ic}, V_{id}, V_{ij}, V_{ia}$

$i \in \{1,2,3,4,5\}$

$n \in \{\text{English, Spaniard, Japanese, Italian, Norwegian}\}$

$c \in \{\text{Red, Green, White, Yellow, Blue}\}$

$d \in \{\text{Tea, Coffee, Milk, Fruit-juice, Water}\}$

$j \in \{\text{Painter, Sculptor, Diplomat, Violinist, Doctor}\}$

$a \in \{\text{Dog, Snails, Fox, Horse, Zebra}\}$

- Example:

$V_{2\text{red}} = \text{true}$ : the 2<sup>nd</sup> house is red

$V_{2\text{red}} = \text{false}$ : the 2<sup>nd</sup> house is not red

7

CS486/686 Lecture Slides (c) 2012 P. Poupart

## Example - Street Puzzle

- Constraint: the Spaniard has a dog

$$\begin{aligned} & (V_{1\text{spaniard}} \Leftrightarrow V_{1\text{dog}}) \wedge (V_{2\text{spaniard}} \Leftrightarrow V_{2\text{dog}}) \wedge \\ & (V_{3\text{spaniard}} \Leftrightarrow V_{3\text{dog}}) \wedge (V_{4\text{spaniard}} \Leftrightarrow V_{4\text{dog}}) \wedge \\ & (V_{5\text{spaniard}} \Leftrightarrow V_{5\text{dog}}) \end{aligned}$$

- Constraint: the green house is on the immediate left of the red house

$$\begin{aligned} & \neg V_{1\text{red}} \wedge (V_{1\text{green}} \Leftrightarrow V_{2\text{red}}) \wedge (V_{2\text{green}} \Leftrightarrow V_{3\text{red}}) \wedge \\ & (V_{3\text{green}} \Leftrightarrow V_{4\text{red}}) \wedge (V_{4\text{green}} \Leftrightarrow V_{5\text{red}}) \wedge \neg V_{5\text{green}} \end{aligned}$$

8

CS486/686 Lecture Slides (c) 2012 P. Poupart

## Example: 4-Queens problem

	1	2	3	4
1				
2				
3				
4				

- Symbols:  $Q_{ij} \ i, j \in \{1, 2, 3, 4\}$

- $Q_{ij} = \text{true}$ : queen at location  $(i, j)$

- $Q_{ij} = \text{false}$ : no queen at location  $(i, j)$

9

CS486/686 Lecture Slides (c) 2012 P. Poupart

## Example: 4-Queens problem

	1	2	3	4
1				
2				
3				
4				

- At least one queen in row 1:

$$Q_{11} \vee Q_{12} \vee Q_{13} \vee Q_{14}$$

- At most one queen in row 1:

$$Q_{11} \Rightarrow \neg Q_{12} \wedge \neg Q_{13} \wedge \neg Q_{14}$$

$$Q_{12} \Rightarrow \neg Q_{11} \wedge \neg Q_{13} \wedge \neg Q_{14}$$

$$Q_{13} \Rightarrow \neg Q_{11} \wedge \neg Q_{12} \wedge \neg Q_{14}$$

$$Q_{14} \Rightarrow \neg Q_{11} \wedge \neg Q_{12} \wedge \neg Q_{13}$$

10

CS486/686 Lecture Slides (c) 2012 P. Poupart

# Knowledge Base (KB)

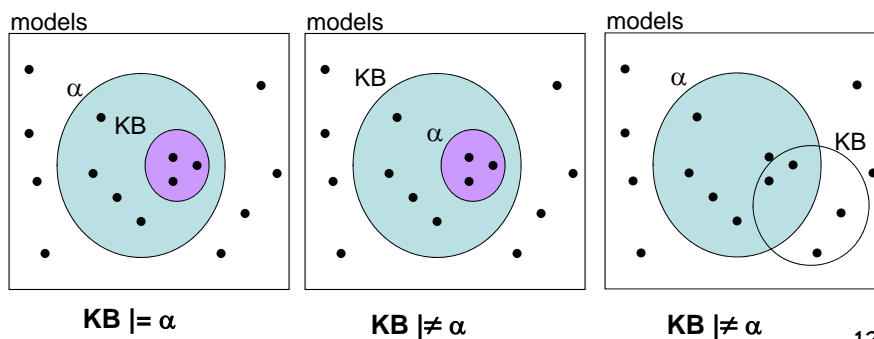
- Knowledge base
  - Encode problem description and any knowledge that could help solve the problem
  - Database of facts and constraints
  - Possible language: propositional logic
- Entailment
  - What can we derive/conclude from KB?
  - $KB \models \alpha$ :  $\alpha$  follows from KB

11

CS486/686 Lecture Slides (c) 2012 P. Poupart

# Semantics

- Entailment:  $KB \models \alpha$ 
  - $\alpha$  is true in all the models in which KB is true



CS486/686 Lecture Slides (c) 2012 P. Poupart

# Inference

- Process of verifying the truth of a formula
- Simple inference algorithm:
  - Build a truth table that enumerates all models
  - Verify that formula is true in all models where KB is true

13

CS486/686 Lecture Slides (c) 2012 P. Poupart

# Truth table

P	Q	R	KB
true	true	true	false
true	true	false	false
true	false	true	false
true	false	false	false
false	true	true	false
false	true	false	false
false	false	true	true
false	false	false	true

KB:

$P \Rightarrow Q,$

$Q \Rightarrow R,$

$\neg Q$

KB  $\models$  P? no

KB  $\models$   $\neg$ P? yes

KB  $\models$  Q? no

KB  $\models$   $\neg$ Q? yes

KB  $\models$  R? no

KB  $\models$   $\neg$ R? no

14

CS486/686 Lecture Slides (c) 2012 P. Poupart

## Efficiency

- Building a truth table is inefficient: exponential in the number of variables
- Alternatives: backtracking, resolution
  - Rewrite "KB  $\models \alpha$ ?" as  
 $KB' = (KB \text{ and } \neg\alpha)$
  - Show that KB' is **not satisfiable** (e.g., there doesn't exist any model for which KB is true, but  $\alpha$  is false)

15

CS486/686 Lecture Slides (c) 2012 P. Poupart

## Convenience

- KB may contain any formula that follows the syntax of our simple logic.
  - Inconvenient: too many possible formula
  - Can we simplify syntax?
- Conjunctive normal form (CNF)
  - Only use  $\wedge, \vee, \neg$
  - Conjunction of clauses, where each clause is a disjunction of (possibly negated) literals
  - Example:  $(V1 \vee V2 \vee V3) \wedge (\neg V1 \vee V2) \wedge (\neg V3 \vee V2)$

16

CS486/686 Lecture Slides (c) 2012 P. Poupart



## Logical equivalence

- How do we transform a KB in CNF?
- Logical equivalence rules
  - $\neg(\neg A) \equiv A$
  - $(A \Rightarrow B) \equiv \neg A \vee B$
  - $(A \Leftrightarrow B) \equiv (A \Rightarrow B) \wedge (B \Rightarrow A)$
  - $\neg(A \vee B) \equiv (\neg A \wedge \neg B)$
  - $\neg(A \wedge B) \equiv (\neg A \vee \neg B)$
  - $(A \vee (B \wedge C)) \equiv ((A \vee B) \wedge (A \vee C))$
- Can every KB be transformed in CNF in polynomial time and space?
  - No: last rule may yield exponentially many clauses

17

CS486/686 Lecture Slides (c) 2012 P. Poupart

## Resolution

- Idea: generate new sentences by implications
  - Unit resolution:  $A \wedge (A \Rightarrow B) \models B$
  - General resolution:  $(A \Rightarrow B) \wedge (B \Rightarrow C) \models (A \Rightarrow C)$
- How do we do this with KB in CNF?
  - Unit resolution:  $A \wedge (\neg A \vee B) \models B$
  - General resolution:  $(\neg A \vee B) \wedge (\neg B \vee C) \models (\neg A \vee C)$
- Algorithm: apply resolution to every possible pair of clauses until
  - Empty clause is generated: false KB
  - No more clauses can be generated: all generated clauses are entailed

18

CS486/686 Lecture Slides (c) 2012 P. Poupart

## Concise Language

- Propositional logic provides a general language to encode deterministic domain knowledge.
- While the encoding is precise and well defined it is often tedious or complicated.
- Simple English sentences often lead to long logical formula.
- **Can we make propositional logic more concise and more natural?**

19

CS486/686 Lecture Slides (c) 2012 P. Poupart

## Recall: 4-Queens problem

	1	2	3	4
1				
2				
3				
4				

- At least one queen in row 1:

$$Q_{11} \vee Q_{12} \vee Q_{13} \vee Q_{14}$$

- At most one queen in row 1:

$$Q_{11} \Rightarrow \neg Q_{12} \wedge \neg Q_{13} \wedge \neg Q_{14}$$

$$Q_{12} \Rightarrow \neg Q_{11} \wedge \neg Q_{13} \wedge \neg Q_{14}$$

$$Q_{13} \Rightarrow \neg Q_{11} \wedge \neg Q_{12} \wedge \neg Q_{14}$$

$$Q_{14} \Rightarrow \neg Q_{11} \wedge \neg Q_{12} \wedge \neg Q_{13}$$

Repeat for every row, column and diagonal.  
This is too tedious! Can we be more concise?

20

CS486/686 Lecture Slides (c) 2012 P. Poupart

# First-order Logic

- World
  - consist of objects and relations
  - Has many objects
- First-order logic:
  - Natural: use predicates to encode relations and constants to encode objects
  - Concise: use quantifiers (e.g.,  $\forall$ ,  $\exists$ ) to talk about many objects simultaneously

21

CS486/686 Lecture Slides (c) 2012 P. Poupart

# First-order Logic Syntax

- Sentence  $\rightarrow$  Predicate(Term, ...) |  
Term = Term |  
(Sentence Connective Sentence) |  
Quantifier Variable, ... Sentence |  
 $\neg$  Sentence
- Term  $\rightarrow$  Function(Term, ...) | Constant | Variable
- Connective  $\rightarrow$   $\Rightarrow$  |  $\wedge$  |  $\vee$  |  $\Leftrightarrow$
- Quantifier  $\rightarrow$   $\forall$  |  $\exists$
- Constant  $\rightarrow$  A |  $X_1$  | John | ...
- Variable  $\rightarrow$  a | x | s | ...
- Predicate  $\rightarrow$  Before | HasColor | Raining | ...
- Function  $\rightarrow$  Mother | LeftLeg | ...

22

CS486/686 Lecture Slides (c) 2012 P. Poupart

## Example: kinship domain

- Elizabeth is the mother of Charles  
Mother(Elizabeth,Charles)
- Charles is the father of William  
Father(Charles,William)
- One's grandmother is the mother of one's parent

$$\forall x,z \exists y \text{ Grandmother}(x,z) \Leftrightarrow \text{Mother}(x,y) \wedge \text{Parent}(y,z)$$

23

CS486/686 Lecture Slides (c) 2012 P. Poupart

## Symbols

- Constant symbols: objects
  - E.g. William, Elizabeth, Charles
- Predicate symbols: relationships
  - Binary: Mother( , ), Grandmother( , )
  - Unary: Female( )
  - Predicates have a truth value
- Function symbols: functions
  - Denote an object
  - E.g.: MotherOf(William) = Elizabeth

24

CS486/686 Lecture Slides (c) 2012 P. Poupart

## Quantifiers

- Universal quantifier:  $\forall$ 
  - For all
  - $\forall x P(x) \equiv P(\text{const1}) \wedge P(\text{const2}) \wedge \dots$
- Existential quantifier:  $\exists$ 
  - There exists
  - $\exists x P(x) \equiv P(\text{const1}) \vee P(\text{const2}) \vee \dots$

25

CS486/686 Lecture Slides (c) 2012 P. Poupart

## Nested Quantifiers

- Order of identical quantifiers doesn't matter
- Brothers are siblings
  - $\forall x \forall y \text{ Brother}(x,y) \Rightarrow \text{siblings}(x,y)$
  - $\forall y \forall x \text{ Brother}(x,y) \Rightarrow \text{siblings}(x,y)$
- Similarly for existential quantifiers
  - $\exists x \exists y P(x,y) \equiv \exists y \exists x P(x,y)$

26

CS486/686 Lecture Slides (c) 2012 P. Poupart

## Nested Quantifiers

- Order of different quantifiers matters
- $\forall x \exists y \text{ Loves}(x,y)$ 
  - Everyone loves someone
  - Conjunction of disjunctions (CNF)
- $\exists x \forall y \text{ Loves}(x,y)$ 
  - There is a person that loves everyone
  - Disjunction of conjunction (DNF)

27

CS486/686 Lecture Slides (c) 2012 P. Poupart

## Connections between $\forall$ and $\exists$

- We only need one of the quantifiers
- De Morgan's rule
  - $\forall x \neg P \equiv \neg \exists x P$        $\neg P \wedge \neg Q \equiv \neg(P \vee Q)$
  - $\neg \forall x P \equiv \exists x \neg P$        $\neg(P \wedge Q) \equiv \neg P \vee \neg Q$
  - $\forall x P \equiv \neg \exists x \neg P$        $P \wedge Q \equiv \neg(\neg P \vee \neg Q)$
  - $\exists x P \equiv \neg \forall x \neg P$        $P \vee Q \equiv \neg(\neg P \wedge \neg Q)$

28

CS486/686 Lecture Slides (c) 2012 P. Poupart

## Equality

- Special relation
- We could define an "Equality" predicate
  - $x = y \equiv \text{Equality}(x,y)$ 
    - True: when  $x$  and  $y$  are the same
    - False: otherwise

29

CS486/686 Lecture Slides (c) 2012 P. Poupart

## Practice: 4-Queens problem

- 4-queens problem in first-order logic
- Predicates:
  - $\text{Queen}( , )$
- Constants:
  - 1, 2, 3, 4 (column and row numbers)
- Column constraints:
  - $\forall i, j_1, j_2 \text{ Queen}(i, j_1) \wedge j_1 \neq j_2 \Rightarrow \neg \text{Queen}(i, j_2)$
  - $\forall i \exists j \text{ Queen}(i, j)$

30

CS486/686 Lecture Slides (c) 2012 P. Poupart

## Practice: street puzzle

- **Predicates:**
  - House(), Person( ), Color( ), Drink( ), Job( ), Animal( )
  - Attr( , ) (attribute of)
- **Constants:**
  - 1, 2, 3, 4, 5 (house number)
  - English, Spaniard, Japanese, Italian, Norwegian
  - Red, Green, White, Yellow, Blue
  - Tea, Coffee, Milk, Juice, Water
  - Painter, Sculptor, Diplomat, Violinist, Doctor
  - Dog, Snails, Fox, Horse, Zebra
- **Function:**
  - Left(x): number of the house to the immediate left of x

31

CS486/686 Lecture Slides (c) 2012 P. Poupart

## Practice: street puzzle

- The Spaniard has a dog
  - $\forall x \text{ House}(x) \Rightarrow (\text{Attr}(x, \text{Spaniard}) \Leftrightarrow \text{Attr}(x, \text{Dog}))$
- the green house is on the immediate left of the red house
  - $\neg \text{Attr}(1, \text{Red}) \wedge$
  - $\neg \text{Attr}(5, \text{Green}) \wedge$
  - $\forall x \text{ House}(x) \wedge x \neq 1 \Rightarrow$   
 $(\text{Attr}(x, \text{Red}) \Leftrightarrow \text{Attr}(\text{Left}(x), \text{Green}))$

32

CS486/686 Lecture Slides (c) 2012 P. Poupart



## Propositional vs first-order logic

- Propositional logic:
  - variables
- First-order logic:
  - Quantifiers, predicates, constants, functions
- Syntactically different!
- Are they equally expressive?
  - Prop logic  $\rightarrow$  1<sup>st</sup>-order logic: yes
  - 1<sup>st</sup>-order logic  $\rightarrow$  prop logic: yes (finite domains)  
no (infinite domains)

33

CS486/686 Lecture Slides (c) 2012 P. Poupart

## Propositional $\rightarrow$ first-order logic

- Variables  $\rightarrow$  predicates
- Indices  $\rightarrow$  constants
- 4-queens problem:
  - $Q_{11} \Rightarrow \neg Q_{12} \wedge \neg Q_{13} \wedge \neg Q_{14}$
  - $Q(1,1) \Rightarrow \neg Q(1,2) \wedge \neg Q(1,3) \wedge \neg Q(1,4)$

34

CS486/686 Lecture Slides (c) 2012 P. Poupart

## First-order $\rightarrow$ propositional logic

- Quantifiers
  - $\forall \rightarrow$  conjunction
  - $\exists \rightarrow$  disjunction
- Predicates  $\rightarrow$  variables
- Constants  $\rightarrow$  indices
- Functions of constants  $\rightarrow$  functions of indices

35

CS486/686 Lecture Slides (c) 2012 P. Poupart

## First-order $\rightarrow$ propositional logic

- $\forall x \text{ House}(x) \wedge x \neq 1 \Rightarrow (\text{Attr}(x, \text{Red}) \Leftrightarrow \text{Attr}(\text{Left}(x), \text{Green}))$
- $\forall x \text{ House}_x \wedge x \neq 1 \Rightarrow (\text{Attr}_{x, \text{Red}} \Leftrightarrow \text{Attr}_{\text{Left}(x), \text{Green}})$
- ~~$(\text{House}_1 \wedge 1 \neq 1 \Rightarrow (\text{Attr}_{1, \text{Red}} \Leftrightarrow \text{Attr}_{\text{Left}(1), \text{Green}})) \wedge$~~   
 ~~$(\text{House}_2 \wedge 2 \neq 1 \Rightarrow (\text{Attr}_{2, \text{Red}} \Leftrightarrow \text{Attr}_{\text{Left}(2), \text{Green}})) \wedge$~~   
 ~~$(\text{House}_3 \wedge 3 \neq 1 \Rightarrow (\text{Attr}_{3, \text{Red}} \Leftrightarrow \text{Attr}_{\text{Left}(3), \text{Green}})) \wedge$~~   
 ~~$(\text{House}_4 \wedge 4 \neq 1 \Rightarrow (\text{Attr}_{4, \text{Red}} \Leftrightarrow \text{Attr}_{\text{Left}(4), \text{Green}})) \wedge$~~   
 ~~$(\text{House}_5 \wedge 5 \neq 1 \Rightarrow (\text{Attr}_{5, \text{Red}} \Leftrightarrow \text{Attr}_{\text{Left}(5), \text{Green}}))$~~

36

CS486/686 Lecture Slides (c) 2012 P. Poupart

# Inference

- Propositionalize KB
  - Run favorite ground inference algorithm (e.g., backtracking (DPLL), resolution)
  - Efficient?
    - No: propositionalizing may yield an exponentially large formula
    - $\forall x \exists y \forall z P(x,y,z) \rightarrow$  conjunction of disjunctions of conjunctions
    - $O(n^m)$  where  $n$  is # of constants and  $m$  is # of quantifiers
- Alternative: lifted inference
  - Work directly with first-order formula

37

CS486/686 Lecture Slides (c) 2012 P. Poupart

# Lifted Inference

- First-order logic:
  - Quantifiers allow us to characterize several objects simultaneously
- Lifted inference:
  - Do inference by reasoning about several terms simultaneously
  - Ground terms only when necessary
  - Hope: determine truth value of goal formula without grounding all terms

38

CS486/686 Lecture Slides (c) 2012 P. Poupart

# Lifted Resolution

Two phases

1. Reduce to lifted conjunctive normal form
2. Resolution with unification

39

CS486/686 Lecture Slides (c) 2012 P. Poupart

# Reduction to Lifted CNF

Six steps

1. Eliminate implications
2. Move negations inwards
3. Standardize variables
4. Skolemize
5. Drop universal quantifiers
6. Distribute  $\vee$  over  $\wedge$

40

CS486/686 Lecture Slides (c) 2012 P. Poupart

## Reduction to Lifted CNF

Example:

$$\forall x [\forall y \text{ Animal}(y) \Rightarrow \text{Loves}(x,y)] \Rightarrow [\exists y \text{ Loves}(y,x)]$$

1. Eliminate implications

$$\forall x [\neg \forall y \neg \text{Animal}(y) \vee \text{Loves}(x,y)] \vee [\exists y \text{ Loves}(y,x)]$$

2. Move negations inwards (apply DeMorgan's rules)

$$\begin{aligned} & \forall x [\exists y \neg(\neg \text{Animal}(y) \vee \text{Loves}(x,y))] \vee [\exists y \text{ Loves}(y,x)] \\ & \forall x [\exists y \neg \neg \text{Animal}(y) \wedge \neg \text{Loves}(x,y)] \vee [\exists y \text{ Loves}(y,x)] \\ & \forall x [\exists y \text{ Animal}(y) \wedge \neg \text{Loves}(x,y)] \vee [\exists y \text{ Loves}(y,x)] \end{aligned}$$

41

CS486/686 Lecture Slides (c) 2012 P. Poupart

## Reduction to Lifted CNF

$$\forall x [\exists y \text{ Animal}(y) \wedge \neg \text{Loves}(x,y)] \vee [\exists y \text{ Loves}(y,x)]$$

3. Standardize variables (use different names)

$$\forall x [\exists y \text{ Animal}(y) \wedge \neg \text{Loves}(x,y)] \vee [\exists z \text{ Loves}(z,x)]$$

4. Skolemize (replace existential variables by new constants or functions)

$$\forall x [\text{Animal}(F(x)) \wedge \neg \text{Loves}(x,F(x))] \vee [\text{Loves}(G(x),x)]$$

42

CS486/686 Lecture Slides (c) 2012 P. Poupart

## Skolemization

- $\exists x P(x)$ 
  - There is at least one term  $x$  that satisfies  $P(x)$
  - We don't care what that term is
- Idea: create a new constant
  - Let  $A$  be a new constant which could correspond to any object, but since we don't care what that object is, just denote it by this new constant.
  - $\exists x P(x) \rightarrow P(A)$

43

CS486/686 Lecture Slides (c) 2012 P. Poupart

## Skolemization

- $\forall x \exists y P(x,y)$ 
  - For each  $x$ , there is at least one term  $y$  that satisfies  $P(x,y)$
  - We don't care what that term is but it may be different for different  $x$
- Idea: create a new function
  - Let  $f$  be a new function that denotes the satisfying term for each  $x$
  - $\forall x \exists y P(x,y) \rightarrow P(x,f(x))$

44

CS486/686 Lecture Slides (c) 2012 P. Poupart

## Reduction to Lifted CNF

$$\forall x [\text{Animal}(F(x)) \wedge \neg \text{Loves}(x, F(x))] \vee [\text{Loves}(G(x), x)]$$

5. Drop universal quantifiers (since all remaining variables are universally quantified)

$$[\text{Animal}(F(x)) \wedge \neg \text{Loves}(x, F(x))] \vee [\text{Loves}(G(x), x)]$$

6. Distribute  $\vee$  over  $\wedge$  (to get a CNF)

$$[\text{Animal}(F(x)) \vee \text{Loves}(G(x), x)] \wedge [\neg \text{Loves}(x, F(x)) \vee \text{Loves}(G(x), x)]$$

45

CS486/686 Lecture Slides (c) 2012 P. Poupart

## Resolution with Unification

- Recall
  - Unit resolution:
    - $A \wedge (A \Rightarrow B) \models B$
    - $A \wedge (\neg A \vee B) \models B$
  - General resolution:
    - $(A \Rightarrow B) \wedge (B \Rightarrow C) \models (A \Rightarrow C)$
    - $(\neg A \vee B) \wedge (\neg B \vee C) \models (\neg A \vee C)$
- Lifted resolution: terms may not match exactly because of variables

46

CS486/686 Lecture Slides (c) 2012 P. Poupart

## Unification

- Variables are like "wild cards" that can be anything
- Find unifier:
  - Unify  $\text{Knows}(\text{John},x)$  and  $\text{Knows}(\text{John},\text{Jane})$   
→  $\text{Knows}(\text{John},\text{Jane})$   $\{x/\text{Jane}\}$
  - Unify  $\text{Knows}(\text{John},x)$  and  $\text{Knows}(y,\text{Bill})$   
→  $\text{Knows}(\text{John},\text{Bill})$   $\{x/\text{Bill}, y/\text{John}\}$
  - Unify  $\text{Knows}(\text{John},x)$  and  $\text{Knows}(y,\text{Mother}(y))$   
→  $\text{Knows}(\text{John},\text{Mother}(\text{John}))$   
 $\{x/\text{Mother}(\text{John}), y/\text{John}\}$

47

CS486/686 Lecture Slides (c) 2012 P. Poupart

## Most General Unifier

- Find unifier that is as general as possible (i.e., preserves as many variables as possible)
- Find most general unifier:
  - Unify  $\text{Knows}(\text{John},x)$  and  $\text{Knows}(y,\text{Mother}(z))$   
→  $\text{Knows}(\text{John},\text{Mother}(z))$   
 $\{y/\text{John}, x/\text{Mother}(z)\}$

48

CS486/686 Lecture Slides (c) 2012 P. Poupart



## Resolution with Unification

- Find clauses with positive and negative terms that can be unified
- Eliminate unifying terms and perform substitutions in remaining terms according to most general unifier
- Unit resolution:
  - $A(f(x),x) \wedge [\neg A(y,g(z)) \vee B(y,z)] \models B(f(g(z)),z)$
  - $\{x/g(z),y/f(g(z))\}$
- General resolution:
  - $[\neg A(x,y) \vee B(y,f(z))] \wedge [\neg B(u,v) \vee C(v,w)] \models$   
 $[\neg A(x,y) \vee C(f(z),w)]$
  - $\{u/y,v/f(z)\}$

49

CS486/686 Lecture Slides (c) 2012 P. Poupart

## Next class

- Markov Logic Networks

50

CS486/686 Lecture Slides (c) 2012 P. Poupart