

# First Order Logic (Predicate Calculus)

CPS 170  
Ronald Parr

## Limitations of Propositional Logic

- Suppose you want to say: All humans are mortal
- For  $\sim 6B$  people, you would need  $\sim 6B$  propositions
- Suppose you want to say that (at least) one person has perfect pitch
- You would need a disjunction of  $\sim 6B$  propositions
- There has to be a better way...

## First Order Logic

- Propositional logic is very restrictive
  - Can't make global statements about objects in the world
  - Workarounds tends to have very large KBs
- First order logic is more expressive
  - Relations, quantification, functions
  - but... inference is trickier

## First Order Syntax

- Sentences
- Atomic sentence predicate(term)
- Terms – functions, constants, variables
- Connectives
- Quantifiers
- Constants
- Variables

## Relations

- Assert relationships between objects
- Examples
  - Loves(Harry, Sally)
  - Between(Canada, US, Mexico)
- Semantics
  - Object and predicate names are mnemonic only
  - Interpretation is imposed from outside
  - Often we imply the “expected” interpretation of predicates and objects with suggested names

## Functions

- Functions are special cases of relations
- Suppose  $R(x_1, x_2, \dots, x_n, y)$  is such that for every value of  $x_1, x_2, \dots, x_n$  there is a unique  $y$
- Then  $R(x_1, x_2, \dots, x_n)$  can be used as a shorthand for  $y$ 
  - Crossed(Right\_leg\_of(Ron), Left\_leg\_of(Ron))
- Remember that the object identified by a function depends upon the interpretation

## Quantification

- For all objects in the world...

$$\forall x \text{happy}(x)$$

- For at least one object in the world...

$$\exists x \text{happy}(x)$$

## Examples

- Everybody loves somebody

$$\forall x \exists y \text{Loves}(x, y)$$

- Everybody loves everybody

$$\forall x \forall y \text{Loves}(x, y)$$

- Everybody loves Raymond

$$\forall x \text{Loves}(x, \text{Raymond})$$

- Raymond loves everybody

$$\forall x \text{Loves}(\text{Raymond}, x)$$

## Equality

- Equality states that two objects are the same
  - `Son_of(Barbara) = Ron`
- Equality is a special relation that holds whenever two objects are the same
- We can imagine that every interpretation comes with its own identity relation
  - `Identical(object27, object58)`

## What's Missing?

- There are many extensions to first order logic
- Higher order logics permit quantification over predicates:

$$\forall x, y (x = y) \Leftrightarrow (\forall p (p(x) \Leftrightarrow p(y)))$$

- Uniqueness
- Extensions typically replace a potentially long series of conjuncts with a single expression

## Inference

- All rules of inference for propositional logic apply to first order logic
- We need extra rules to handle substitution for quantified variables

$\text{SUBST}(\{x / \text{Harry}, y / \text{Sally}\}, \text{Loves}(x, y)) = \text{Loves}(\text{Harry}, \text{Sally})$

## Inference Rules

- Universal Elimination

$$\frac{\forall v : \alpha(v)}{\text{SUBST}(\{v / g\}, \alpha(v))}$$

- How to read this:
  - We have a universally quantified variable  $v$  in  $\alpha$
  - Can substitute any  $g$  for  $v$  and  $\alpha$  will still be true

## Inference Rules

- Existential Elimination

$$\frac{\exists v : \alpha(v)}{\text{SUBST}(\{v/k\}, \alpha(v))}$$

- How to read this:
  - We have a universally quantified variable  $v$  in  $\alpha$
  - Can substitute any  $k$  for  $v$  and  $\alpha$  will still be true
  - IMPORTANT:  $k$  must be a ***previously unused*** constant (***skolem*** constant). Why is this OK?

## Skolemization within Quantifiers

- Skolemizing w/in universal quantifier is tricky
- Everybody loves somebody

$$\forall x \exists y : \text{loves}(x, y)$$

- With Skolem constants, becomes:

$$\forall x : \text{loves}(x, \text{object34752})$$

- Why is this wrong?
- Need to use ***skolem functions***:
$$\forall x : \text{loves}(x, \text{personlovedby}(x))$$

## Inference Rules

- Existential Introduction

$$\frac{\alpha(g)}{\text{SUBST}(\{v/g\}, \exists v : \alpha(v))}$$

- How to read this:
  - We know that the sentence  $\alpha$  is true
  - Can substitute variable  $v$  for any constant  $g$  in  $\alpha$  and (w/existential quantifier) and  $\alpha$  will still be true
  - Why is this OK?

## Generalized Modus Ponens Example

- If  $\text{has\_US\_birth\_certificate}(X)$  then  $\text{natural\_US\_citizen}(X)$
- $\text{has\_US\_birth\_certificate}(\text{Obama})$
- Conclude  $\text{SUBST}(\{\text{Obama}/X\}, \text{natural\_US\_citizen}(X))$
- i.e.,  $\text{natural\_US\_citizen}(\text{Obama})$



## Generalized Modus Ponens

$$\text{SUBST}(\theta, p_i') = \text{SUBST}(\theta, p_i) \forall i$$

$$\frac{p_1', p_2', \dots, p_n', (p_1 \wedge p_2 \wedge \dots \wedge p_n \Rightarrow q)}{\text{SUBST}(\theta, q)}$$

- How to read this:
  - We have an implication which implies q
  - Any consistent substitution of variables on the LHS must yield a valid conclusion on the RHS

## Unification

- Substitution is a non-trivial matter
- We need an algorithm unify:

$$\text{Unify}(p, q) = \theta : \text{Subst}(\theta, p) = \text{Subst}(\theta, q)$$

- Important: Unification replaces variables:

$\exists x \text{Loves}(\text{John}, x)$

$\exists x \text{Hates}(\text{John}, x)$

- Are these the same x?

## Unification Example

$\forall x \text{Knows}(\text{John}, x) \Rightarrow \text{Loves}(\text{John}, x)$

$\text{Knows}(\text{John}, \text{Jane})$

$\forall y \text{Knows}(y, \text{Leonid})$

$\forall y \text{Knows}(y, \text{Mother}(y))$

$\forall x \text{Knows}(x, \text{Elizabeth})$

Note: All unquantified variables are assumed universal from here on.

$\text{Unify}(\text{Knows}(\text{John}, x), \text{Knows}(\text{John}, \text{Jane})) = \{x / \text{Jane}\}$

$\text{Unify}(\text{Knows}(\text{John}, x), \text{Knows}(y, \text{Leonid})) = \{x / \text{Leonid}, y / \text{John}\}$

$\text{Unify}(\text{Knows}(\text{John}, x), \text{Knows}(y, \text{Mother}(y))) = \{y / \text{John}, x / \text{Mother}(\text{John})\}$

$\text{Unify}(\text{Knows}(\text{John}, x), \text{Knows}(x, \text{Elizabeth})) = \{x_1 / \text{Elizabeth}, x_2 / \text{John}\}$

## Most General Unifier

- $\text{Unify}(\text{Knows}(\text{John}, x), \text{Knows}(y, z))$ 
  - $\{y / \text{John}, x / z\}$
  - $\{y / \text{John}, x / z, w / \text{Freda}\}$
  - $\{y / \text{John}, x / \text{John}, z / \text{John}\}$
- When in doubt, we should always return the most general unifier (MGU)
  - MGU makes least commitment about binding variables to constants

## Proof Procedures

- Suppose we have a knowledge base: KB
- We want to prove  $q$
- Forward Chaining
  - Like search: Keep proving new things and adding them to the KB until we are able to prove  $q$
- Backward Chaining
  - Find  $p_1 \dots p_n$  s.t. knowing  $p_1 \dots p_n$  would prove  $q$
  - Recursively try to prove  $p_1 \dots p_n$

## Forward Chaining Example

$\forall x \text{Knows}(\text{John}, x) \Rightarrow \text{Loves}(\text{John}, x)$

$\text{Knows}(\text{John}, \text{Jane})$

$\forall y \text{Knows}(y, \text{Leonid})$

$\forall y \text{Knows}(y, \text{Mother}(y))$

$\forall x \text{Knows}(x, \text{Elizabeth})$

- $\text{Loves}(\text{John}, \text{Jane})$
- $\text{Loves}(\text{John}, \text{Leonid})$
- $\text{Loves}(\text{John}, \text{Mother}(\text{John}))$
- $\text{Loves}(\text{John}, \text{Elizabeth})$

## Forward Chaining

```
Procedure Forward_Chain(KB,p)
  If p is in KB then return
  Add p to KB
  For each  $(p_1 \wedge \dots \wedge p_n \Rightarrow q)$  in KB such that for some i,
  Unify( $p_i, p$ )=q succeeds do
    Find_And_Infer(KB,  $[p_1, \dots, p_{i-1}, p_{i+1}, \dots, p_n]$ , q, q)
  end

Procedure Find_and_Infer(KB, premises, conclusion, q)
  If premises=[] then
    Forward_Chain(KB, Subst(q, conclusion))
  Else for each  $p'$  in KB such that
  Unify( $p', \text{Subst}(q, \text{Head}(\text{premises}))$ )= $q_2$  do
    Find_And_Infer(KB, Tail(premises), conclusion,  $[q, q_2]$ )
  end
```

## A Note About Forward Chaining

- As presented, forward chaining seems undirected
- Can view forward chaining as a search problem
- Can apply heuristics to guide this search
- If you're trying to prove that Barack Obama is a natural born citizen, should you should start by proving that square127 is also a rectangle???

## Backward Chaining Example

$\forall x \text{Knows}(\text{John}, x) \Rightarrow \text{Loves}(\text{John}, x)$

$\text{Knows}(\text{John}, \text{Jane})$

$\forall y \text{Knows}(y, \text{Leonid})$

$\forall y \text{Knows}(y, \text{Mother}(y))$

$\forall x \text{Knows}(x, \text{Elizabeth})$

- Goal:  $\text{Loves}(\text{John}, \text{Jane})?$
- Subgoal:  $\text{Knows}(\text{John}, \text{Jane})$

## Backward Chaining

```
Function Back_Chain(KB,q)
    Back_Chain_List(KB,[q],{})

Function Back_Chain_List(KB,qlist,q)
    If qlist=[] then return q
    q<-head(qlist)
    For each qi' in KB such that qi<-Unify(q,qi') succeeds do
        Answers <- Answers + [q,qi]
    For each (p1^...^pn=>qi') in KB: qi<-Unify(q,qi') succeeds do
        Answers<- Answers+
            Back_Chain_List(KB,Subst(qi,[p1...pn]),[q,qi])
    return union of Back_Chain_List(KB,Tail(qlist),q) for each q in answers
```

## Completeness

$$\forall x P(x) \Rightarrow Q(x)$$

$$\forall x \neg P(x) \Rightarrow R(x)$$

$$\forall x Q(x) \Rightarrow S(x)$$

$$\forall x R(x) \Rightarrow S(x)$$

$$S(A) ???$$

- Problem: Generalized Modus Ponens not complete
- Goal: A sound **and** complete inference procedure for first order logic

## Generalized Resolution

$$\theta = \text{Unify}(p_j, \neg q_k)$$

$$\frac{(p_1 \vee \dots p_j \dots \vee p_m), (q_1 \vee \dots q_k \dots \vee q_n)}{\text{SUBST}(\theta, (p_1 \vee \dots p_{j-1} \vee p_{j+1} \dots \vee p_m \vee q_1 \vee \dots q_{k-1} \vee q_{k+1} \dots \vee q_n))}$$

- If the same term appears in both positive and negative form in two disjunctions, they cancel out when disjunctions are combined

## Generalized Resolution Example

$(\neg P(x) \vee Q(x))$

$(P(x) \vee R(x))$

$(\neg Q(x) \vee S(x))$

$(\neg R(x) \vee S(x))$

$S(A)???$

Example on board/tablet...

## Resolution Properties

- Proof by refutation (asserting negation and resolving to nil) is sound and complete  
(NB: We did not do this in the previous example)
- Resolution is not complete in a generative sense, only in a testing sense
- This is only part of the job
- To use resolution, we must convert everything to a canonical form

## Canonical Form

- Eliminate Implications
- Move negation inwards
- Standardize (apart) variables
- Move quantifiers Left
- Skolemize
- Drop universal quantifiers
- Distribute AND over OR
- Flatten nested conjunctions and disjunctions

## Computational Properties

- We can enumerate the set of all proofs
- We can check if a proof is valid
- First order logic is complete (Gödel)
  
- What if no valid proof exists?
- Inference in first order logic is *semi-decidable*
- Compare with halting problem  
(halting problem is semi-decidable)



## Gödel's Incompleteness Result

- Gödel's incompleteness result is, perhaps, better known
- Incompleteness applies to logical/mathematical systems rich enough to contain numbers and math
  - Need a way of enumerating all valid proofs within the system
  - Need a way of referring to proofs by number
- Construct a Gödel sentence:
  - S: For all  $i$ ,  $i$  is not the number of a proof of the sentence  $j$
  - (Equivalent to saying, there does not exist a proof of sentence  $j$ )
  - Suppose sentence S is sentence  $j$ 
    - If S is false, then we have a contradiction
    - If S is true, then we can't have a proof of it

## Diagonalization

- Incompleteness can be seen as an instance of diagonalization:
  - Define a set  
(Rationals, TMs that halt, theorems that are provable)
  - Use rules of the system to create an impossible object
- Example: Proof that reals are not enumerable (i.e., not countable and therefore larger than the rationals)

## Countability of Rationals

$$X = \frac{n_0 \times 2^0 + n_1 \times 2^1 + n_2 \times 2^2 \dots}{d_0 \times 2^0 + d_1 \times 2^1 + d_2 \times 2^2 \dots}$$

Label	$n_0$	$d_0$	$n_1$	$d_1$	...
0	0	0	0	0	...
1	1	0	0	0	...
2	0	1	0	0	...
3	1	1	0	0	...
...	...	...	...	...	...

## Uncountability of Reals

- Given:

Label	$n_0$	$d_0$	$n_1$	$d_1$	...
0	0	0	0	0	...
1	1	0	0	0	...
2	0	1	0	0	...
3	1	1	0	0	...
...	...	...	...	...	...

- Construct:

Label	$n_0$	$d_0$	$n_1$	$d_1$	...
1	1	0	0	0	...
1	1	1	0	0	...
2	0	1	1	0	...
3	1	1	0	1	...
...	...	...	...	...	...

## Implications of all this

- Sophomoric interpretation: AI is impossible/implausible because there will always be true things that cannot be discovered by logic
- A bit of reality:
  - Incompleteness talks about a system's ability to prove things about itself
  - For any given system, it may be possible to prove things by talking about the system in a more expressive language
  - Relationship of the unprovable to intelligence is murky at best: Are the things you can't justify the things that make you intelligent?
  - Not clear that anything interesting is unprovable in a practical sense (though plenty of interesting things remain unproven)

## First Order Logic Conclusions

- First order logic adds relations and quantification to predicate logic
- Inference in first order logic is, essentially, a generalization of inference in predicate logic
  - Resolution is sound and complete
  - Use of resolution requires:
    - Conversion to canonical form
    - Proof by refutation
- In general, inference in first order logic is semi-decidable
- FOL + basic math is no longer complete