First-order logic

- First-order logic (FOL) models the world in terms of
  - Objects, which are things with individual identities
  - Properties of objects that distinguish them from other objects
  - Relations that hold among sets of objects
  - Functions, which are a subset of relations where there is only one "value" for any given "input"

Ex: Objects: Students, lectures, companies, cars ...

- Relations: Brother-of, bigger-than, outside, part-of, has-color, occurs-after, owns, visits, precedes, ...
- Properties: blue, oval, even, large, ...
- Functions: father-of, best-friend, second-half, one-more-than ...

Sentence → *Atomicsentence*

| (*Sentence Connective Sentence*)

| *Quantifier Variable,. . . Sentence*

| *Sentence*

AtomicSentence → *Predicate(Term, . . .)*

| (*Term* = *Term*

Term -→ *Function(Term, . . .)*

l *Constant*

| *Variable*

Connective → ¬, ∧, ∨, ⇒

Quantifier → ∀, ∃

Constant → *A* (XI (*John* 1 . . .

Variable → *a* | *x* | s | . . .

Predicate → *Before…*

Function → *Mother* | …

Truth in first-order logic

- Sentences are true with respect to a model and an interpretation
- Model contains objects (domain elements) and relations among them
- Interpretation specifies referents for

constant symbols → objects

predicate symbols → relations

function symbols → functional relations

- An atomic sentence *predicate*($term_1$,...,$term_n$) is true iff the objects referred to by $term_1$,...,$term_n$ are in the relation referred to by *predicate*

Entailment

- Entailment means that one thing follows from another:

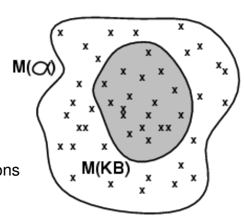
$$KB \models \alpha$$

Knowledge base *KB* entails sentence α if and only if α is true in all worlds where *KB* is true

- E.g., the KB containing "the Greens won" and "the Reds won" entails "Either the Greens or the reds won"
- E.g., x+y = 4 entails 4 = x+y
- Entailment is a relationship between sentences (i.e., syntax) that is based on semantics
- entailment: necessary truth of one sentence given another
-

Models

- Logicians typically think in terms of models, which are formally structured worlds with respect to which truth can be evaluated
- We say *m* is a model of a sentence α if α is true in *m*
- *M(α)* is the set of all models of α
- Then $KB \models \alpha$ iff $M(KB) \subseteq M(\alpha)$
  - E.g. *KB* = Greens won and Reds won α = Greens won



- Think of KB and α as collections of constraints and of models m as possible states. M(KB) are the solutions to KB and M(α) the solutions to α. Then, $KB \models \alpha$ when all solutions to KB are also solutions to α.

Inference

- $KB \vdash_i \alpha$ = sentence α can be derived from *KB* by procedure *i*

i.e. deriving sentences from other sentences

- Soundness: *i* is sound if whenever $KB \vdash_i \alpha$, it is also true that $KB \models \alpha$

i.e. derivations produce only entailed sentences *(no wrong inferences, but maybe not all inferences)*

- Completeness: *i* is complete if whenever $KB \models \alpha$, it is also true that $KB \vdash_i \alpha$

i.e. derivations can produce all entailed sentences (*all inferences can be made, but maybe some wrong extra ones as well)*

Validity and satisfiability

- A sentence is **valid** if it is true in **all models**,
- e.g., *True*, $A \vee \neg A$, $A \Rightarrow A$, $(A \wedge (A \Rightarrow B)) \Rightarrow B$

Validity is connected to inference via the following:
$KB \models \alpha$ if and only if $(KB \Rightarrow \alpha)$ is valid

A sentence is **satisfiable** if it is true in **some model**
e.g., $A \vee B$, C

A sentence is **unsatisfiable** if it is true in **no models**
e.g., $A \wedge \neg A$

Satisfiability is connected to inference via the following:
$KB \models \alpha$ if and only if $(KB \wedge \neg\alpha)$ is unsatisfiable
(there is no model for which KB=true and is false)

Proof Methods in FOL

Three Major Families:

- Resolution
- Forward chaining
- Backward chaining

Some Other inference tools:

Entailment/ Unification/ GMP/ reduction

Inferencing in Predicate Calculus

- Resolution – Refutation
  - Negate goal
  - Convert all pieces of knowledge into clausal form (disjunction of literals)
  - See if contradiction indicated by null clause $\Box$ can be derived
- Forward chaining
  - Given P, $P \to Q$, to infer Q
  - P, match *L.H.S* of
  - Assert Q from *R.H.S*
- Backward chaining
  - Q, Match *R.H.S* of $P \to Q$
  - assert P
  - Check if P exists

Universal instantiation (UI)

- Every instantiation of a universally quantified sentence is entailed by it:

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

for any variable v and ground term g

- E.g., $\forall x\ King(x) \wedge Greedy(x) \Rightarrow Evil(x)$ yields:

$King(John) \wedge Greedy(John) \Rightarrow Evil(John)$
$King(Richard) \wedge Greedy(Richard) \Rightarrow Evil(Richard)$
$King(Father(John)) \wedge Greedy(Father(John)) \Rightarrow Evil(Father(John))$

Existential instantiation (EI)

- For any sentence α, variable *v*, and constant symbol *k* that does not appear elsewhere in the knowledge base:

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

- E.g., $\exists x\ Crown(x) \wedge OnHead(x,John)$ yields:

$$Crown(C_1) \wedge OnHead(C_1,John)$$

provided C_1 is a new constant symbol, called a Skolem constant

Unification

- We can get the inference immediately if we can find a substitution θ such that *King(x)* and *Greedy(x)* match *King(John)* and *Greedy(y)*

θ = {x/John,y/John} works

- Unify(α,β) = θ if αθ = βθ

p	q	θ
Knows(John,x)	Knows(John,Jane)	
Knows(John,x)	Knows(y,OJ)	
Knows(John,x)	Knows(y,Mother(y))	
Knows(John,x)	Knows(x,OJ)	

- Standardizing apart eliminates overlap of variables, e.g., Knows(z_{17},OJ)

Unification

- We can get the inference immediately if we can find a substitution θ such that *King(x)* and *Greedy(x)* match *King(John)* and *Greedy(y)*

θ = {x/John,y/John} works

- Unify(α,β) = θ if αθ = βθ

p	q	θ
Knows(John,x)	Knows(John,Jane)	{x/Jane}}
Knows(John,x)	Knows(y,OJ)	
Knows(John,x)	Knows(y,Mother(y))	
Knows(John,x)	Knows(x,OJ)	

- Standardizing apart eliminates overlap of variables, e.g., Knows(z_{17},OJ)

Unification

- We can get the inference immediately if we can find a substitution θ such that *King(x)* and *Greedy(x)* match *King(John)* and *Greedy(y)*

θ = {x/John,y/John} works

- Unify(α,β) = θ if αθ = βθ

p	q	θ
Knows(John,x)	Knows(John,Jane)	{x/Jane}}
Knows(John,x)	Knows(y,OJ)	{x/OJ,y/John}}
Knows(John,x)	Knows(y,Mother(y))	
Knows(John,x)	Knows(x,OJ)	

- Standardizing apart eliminates overlap of variables, e.g., Knows(z_{17},OJ)

Unification

- We can get the inference immediately if we can find a substitution θ such that *King(x)* and *Greedy(x)* match *King(John)* and *Greedy(y)*

θ = {x/John,y/John} works

- Unify(α,β) = θ if αθ = βθ

p	q	θ
Knows(John,x)	Knows(John,Jane)	{x/Jane}}
Knows(John,x)	Knows(y,OJ)	{x/OJ,y/John}}
Knows(John,x)	Knows(y,Mother(y))	{y/John,x/Mother(John)}}
Knows(John,x)	Knows(x,OJ)	{fail}

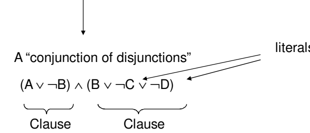
- Standardizing apart eliminates overlap of variables, e.g., Knows(z_{17},OJ)

Conjunction Normal Form (CNF)

We like to prove: $KB \models \alpha$

equivalent to : $KB \wedge \neg\alpha$ *unsatifiable*

We first rewrite $KB \wedge \neg\alpha$ into conjunctive normal form (CNF).



In theory

- Any KB can be converted into CNF.
- In fact, any KB can be converted into CNF-3 using clauses with at most 3 literals.

Example: Conversion to CNF (PC)

$B_{1,1} \Leftrightarrow (P_{1,2} \vee P_{2,1})$

1. Eliminate ⇔, replacing α ⇔ β with (α ⇒ β)∧(β ⇒ α).

   $(B_{1,1} \Rightarrow (P_{1,2} \vee P_{2,1})) \wedge ((P_{1,2} \vee P_{2,1}) \Rightarrow B_{1,1})$

2. Eliminate ⇒, replacing α ⇒ β with ¬α∨ β.

   $(\neg B_{1,1} \vee P_{1,2} \vee P_{2,1}) \wedge (\neg(P_{1,2} \vee P_{2,1}) \vee B_{1,1})$

3. Move ¬ inwards using de Morgan's rules and double-negation: $\neg(\alpha \vee \beta) \equiv \neg\alpha \wedge \neg\beta$

   $(\neg B_{1,1} \vee P_{1,2} \vee P_{2,1}) \wedge ((\neg P_{1,2} \wedge \neg P_{2,1}) \vee B_{1,1})$

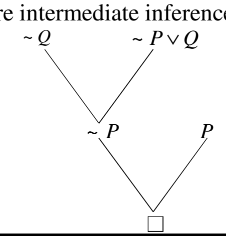
4. Apply distributive law (∧ over ∨) and flatten:

   $(\neg B_{1,1} \vee P_{1,2} \vee P_{2,1}) \wedge (\neg P_{1,2} \vee B_{1,1}) \wedge (\neg P_{2,1} \vee B_{1,1})$

Resolution (PC)

1. P
2. $P \rightarrow Q$ converted to $\sim P \vee Q$
3. $\sim Q$

Draw the resolution tree (actually an inverted tree). Every node is a clausal form and branches are intermediate inference steps.



Resolution Algorithm

- The resolution algorithm tries to prove: $KB \models \alpha$ *equivalent to* $KB \wedge \neg\alpha$ *unsatisfiable*

- Generate all new sentences from KB and the query.
- One of two things can happen:

1. We find $P \wedge \neg P$ which is unsatisfiable. i.e. we can entail the query.
2. We find no contradiction: there is a model that satisfies the sentence $KB \wedge \neg\alpha$ (non-trivial) and hence we cannot entail the query.

Conversion to CNF

- Everyone who loves all animals is loved by someone:

∀x([∀y *Animal*(y) ⇒ *Loves*(x,y)] ⇒ [∃y *Loves*(y,x)])

1. Eliminate biconditionals and implications

∀x([¬∀y (¬*Animal*(y) ∨ *Loves*(x,y))] ∨ [∃y *Loves*(y,x)])

2. Move ¬ inwards:"¬∀x p ≡ ∃x ¬p, ¬ ∃x p ≡ ∀x ¬p"

∀x ([∃y (¬(¬*Animal*(y) ∨ *Loves*(x,y)))] ∨ [∃y *Loves*(y,x)])

∀x ([∃y (¬¬*Animal*(y) ∧ ¬*Loves*(x,y))] ∨ [∃y *Loves*(y,x)])

∀x([∃y (*Animal*(y) ∧ ¬*Loves*(x,y))] ∨ [∃y *Loves*(y,x)])

Conversion to CNF contd.

3. Standardize variables: each quantifier should use a different one

∀x([∃y *Animal*(y) ∧ ¬*Loves*(x,y)] ∨ [∃z *Loves*(z,x)])

4. Skolemize: a more general form of existential instantiation.

Each existential variable is replaced by a Skolem function of the enclosing universally quantified variables:

∀x([*Animal*(*F*(x)) ∧ ¬*Loves*(x,*F*(x))] ∨ *Loves*(*G*(x),x))

5. Drop universal quantifiers:

[*Animal*(*F*(x)) ∧ ¬*Loves*(x,*F*(x))] ∨ *Loves*(*G*(x),x)

6. Distribute ∨ over ∧ :

[*Animal*(*F*(x)) ∨ *Loves*(*G*(x),x)] ∧ [¬*Loves*(x,*F*(x)) ∨ *Loves*(*G*(x),x)]

Resolution in PC

Conjunctive Normal Form (CNF)

conjunction of disjunctions of literals

E.g., (A ∨ ¬B) ∧ (B ∨ ¬C ∨ ¬D)

- Resolution inference rule (for CNF):

$$\frac{\ell_1 \vee \ldots \vee \ell_k, \qquad m_1 \vee \ldots \vee m_n}{\ell_1 \vee \ldots \vee \ell_{s-1} \vee \ell_{s+1} \vee \ldots \vee \ell_k \vee m_1 \vee \ldots \vee m_{j-1} \vee m_{j+1} \vee \ldots \vee m_n}$$

where ℓ_s and m_j are complementary literals.

E.g., $$\frac{P_{1,3} \vee P_{2,2}, \qquad \neg P_{2,2}}{P_{1,3}}$$

Resolution is sound and complete for propositional logic

Resolution in FOL

- Full first-order version:

$$\frac{\ell_1 \vee \cdots \vee \ell_k, \qquad m_1 \vee \cdots \vee m_n}{(\ell_1 \vee \cdots \vee \ell_{i-1} \vee \ell_{i+1} \vee \cdots \vee \ell_k \vee m_1 \vee \cdots \vee m_{j-1} \vee m_{j+1} \vee \cdots \vee m_n)\theta}$$

where `Unify`(ℓ_i, $\neg m_j$) = θ.

The two clauses are assumed to be standardized apart so that they share no variables.

- For example,

$$\frac{\neg Rich(x) \vee Unhappy(x) \qquad Rich(Ken)}{Unhappy(Ken)}$$

with θ = {x/Ken}

A More Concise Version

$$\frac{\bigvee_{i \in A} L_i \qquad \bigvee_{i \in B} L_i}{\bigvee_{i \in C} Subst(\theta, L_i)}$$

$Unify(L_j, \neg L_k)$

$j \in A, k \in B$

$C = (A \cup B) \setminus \{j, k\}$

E.g. for A = {1, 2, 7} first clause is $L_1 \vee L_2 \vee L_7$

Empty Clause means False

- Resolution theorem proving ends
  - When the resolved clause has no literals (empty)
- This can only be because:
  - Two **unit clauses** were resolved
    - One was the negation of the other (after substitution)
  - Example: q(X) and ¬q(X) or: p(X) and ¬p(bob)
- Hence if we see the empty clause
  - This was because there was an inconsistency
  - Hence the proof by refutation

Resolution as Search

- **Initial State**: Knowledge base (KB) of axioms and negated theorem in CNF
- **Operators**: Resolution rule picks 2 clauses and adds new clause
- **Goal Test**: Does KB contain the empty clause?

- Search space of KB states
- We want proof (path) or just checking (artefact)