Queen Mary
University of London

Knowledge Representation and Reasoning

Simon Dixon
Queen Mary University of London

Outline

Queen Mary
University of London

- Knowledge Representation in Logic
  - The Propositional Calculus
  - The First Order Predicate Calculus
- Reasoning
  - Inference Rules to Compute with Calculus Expressions
- Application

Knowledge engineering

Queen Mary
University of London

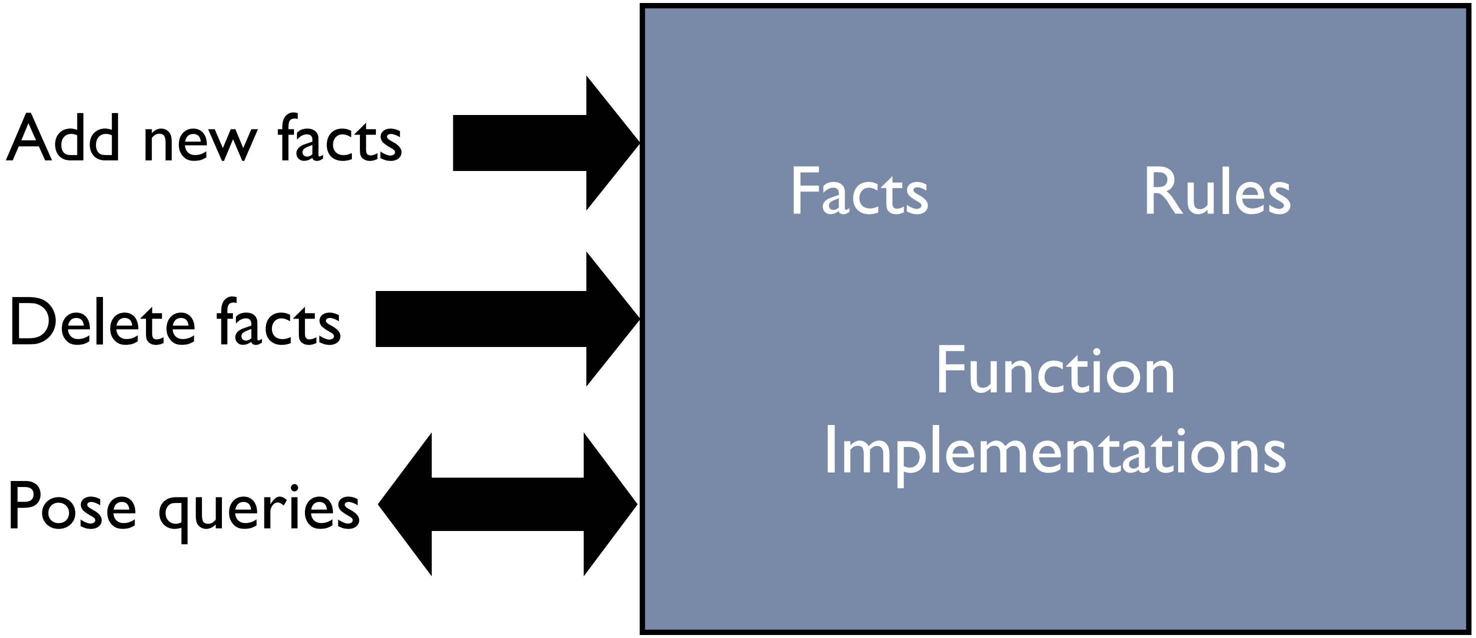
- The role of the Knowledge Engineer is to
  - elicit or otherwise ascertain knowledge
  - represent it in the most appropriate way
  - use it to derive previously unknown facts
    - follow a chain of reasoning from new data to a conclusion (e.g. medical diagnosis)
    - make explicit things that were previously implicit in a system that was too complex for a human to understand all at once
- One way to represent knowledge is using *logic*
- Examples of simple (atomic) logic statements
  - Socrates-is-a-man
  - Man(Socrates)
  - Philosopher(Socrates)
  - Occupation(Socrates, Philosopher)

Forms of atomic sentences

- Most atomic sentences have one of the following forms:
  - Statement
    - e.g. Socrates-is-a-man
  - Property(Object)
    - e.g. Man(Socrates), Dead(Socrates),
    - Perhaps clearer if written IsMan(Socrates), IsDead(Socrates)
  - Relation(Object1, Object2, …)
    - e.g. Occupation(Socrates, Philosopher), Mother(Elizabeth, Charles), LessThan(2, 5)
    - The convention is that Object1 would be the subject of the sentence if expressed in English (Socrates has occupation philosopher; Elizabeth is the mother of Charles; 2 is less than 5)
- In each case, they are sentences, i.e. they say something
  - What the sentence says might be true or false

Knowledge engineering

- Often, in one formalism or another, this will involve maintaining a database of facts that are known to be true and rules that can apply to them



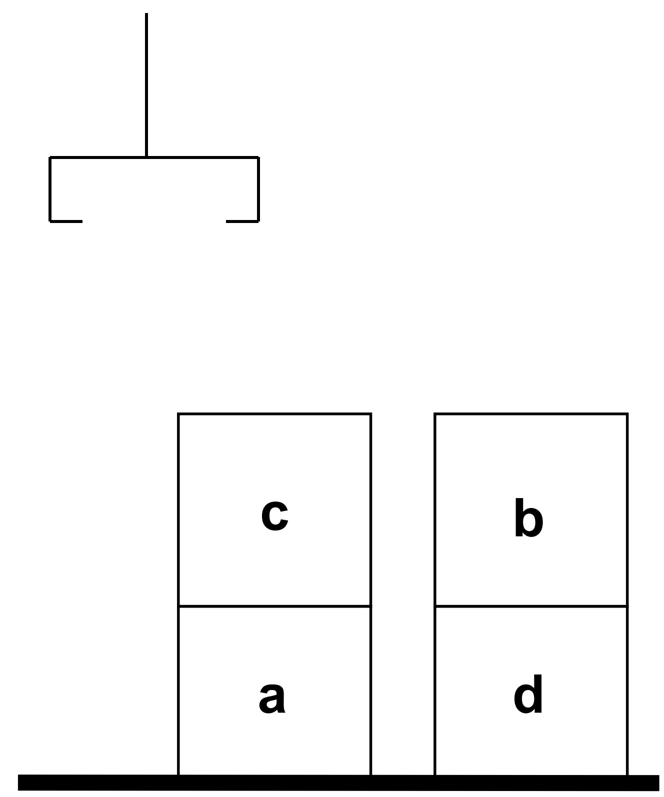
Knowledge engineering

- Quite often, problem formulation in real-world situations is very difficult
  - different experts have different opinions
  - the world is continuous and unpredictable
  - clients don’t really know what they want from you
- A common approach to understanding the issues involved in KE is to use a highly simplified world, and then to generalise with experience
  - a common simplification is the “blocks” world

Example: the Blocks World

Queen Mary University of London

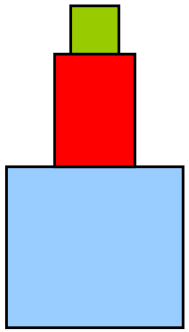
- There is/are
  - a table
  - some distinguishable blocks
  - a robot hand/arm
- Problems are specified by the initial and desired states
- Solutions are expressed as a sequence of actions by the arm
- Predicates
  - On(x,y), On-table(x)
  - Clear(x)
  - Empty-table



Knowledge representation and inference

Queen Mary University of London

- KR should allow us, for a given world, to:
  - Express facts or beliefs using a formal language
    - expressively and unambiguously
- The inference procedure should allow us to:
  - Determine automatically what follows from these facts
    - correctly (soundly) and completely (and tractably)
- Example:
  - Be able to express formally that:
    - “The red block is above the blue block”
    - “The green block is above the red block”
  - Be able to infer:
    - “The green block is above the blue block”
    - “The blocks form a tower”



Example

- Given
  - If it rains in the morning, then I wear my black coat
  - If I wear my black coat, then I wear my black shoes
  - I am not wearing black shoes
- Find out
  - Was it raining this morning?
- Human reasoning:
  - Brown shoes, so no black coat, so it was not raining this morning
    - We want a computer to do that, reliably and in general

Components of a logical calculus

Queen Mary
University of London

- A *formal language*
  - words and *syntactic* rules that tell us how to build up sentences
    - so we can build up more complex statements from simple ones
  - *semantic* mappings that tell us what the words mean
- An *inference procedure* which allows us to compute which sentences are valid *inferences* from other sentences
- There are many different logical calculi; here we study
  - The Propositional Calculus
  - The First Order Predicate Calculus

The Propositional Calculus (PC)

- Each symbol in the Propositional Calculus is either:
  - a *proposition*: a basic, smallest unit of meaning in the calculus
    - e.g. "It-is-raining"
  - a *connective*: for combining propositions into more complex *sentences*
- Two reserved, special propositions
  - True and False
    - with the obvious meanings!
- Convention: propositions begin with upper case letters
  - P, Q, Sunny, etc.
- Connectives use special symbols
  - $\wedge$ (and) , $\vee$ (or) , $\neg$ (not), $\rightarrow$ (implies), $\equiv$ (is equivalent to)

Sentences (syntax) in PC

Queen Mary University of London

- The Sentence is the syntactic unit to which *truth values* can be attached
    - Sentences are also called *Well-Formed Formulae*
  - Every propositional symbol is a sentence. E.g.: True, False, P
  - The negation of a sentence is a sentence. E.g.: $\neg$P, $\neg$False.
  - The conjunction (and) of two sentences is a sentence. E.g.: $P \wedge Q$
  - The disjunction (or) of two sentences is a sentence. E.g.: $P \vee Q$
  - The implication of one sentence by another is a sentence. E.g.: $P \rightarrow Q$
    - Note that implication can also be expressed as $\neg P \vee Q$
  - The equivalence of two sentences is a sentence. E.g.: $P \equiv Q$
    - Note that equivalence can also be expressed as $(P \rightarrow Q) \wedge (Q \rightarrow P)$
    - $\equiv$ is therefore sometimes omitted from the propositional calculus

Semantics (meaning) in PC

- An *interpretation* of a set of sentences is the assignment of a truth value, either T or F, to each propositional symbol (and so to each sentence)
    - The proposition True is always assigned truth value T
    - The proposition False is always assigned truth value F
    - The assignment of negation, $\neg P$, is F iff (if and only if) the assignment of P is T
    - The assignment of conjunction, $P \wedge Q$, is T iff both P and Q are assigned T
    - The assignment of disjunction, $P \vee Q$, is F iff both P and Q are assigned F
    - The assignment of implication, $P \rightarrow Q$, is F iff the assignment of P is T and the assignment of Q is F
    - The assignment of equivalence, $P \equiv Q$, is T iff the assignments of P and Q are the same

Properties of logical connectives

- commutativity
  - $P \lor Q \equiv Q \lor P$
  - $P \land Q \equiv Q \land P$
- associativity
  - $(P \lor Q) \lor R \equiv P \lor (Q \lor R)$
  - $(P \land Q) \land R \equiv P \land (Q \land R)$
- distributivity
  - $P \lor (Q \land R) \equiv (P \lor Q) \land (P \lor R)$
  - $P \land (Q \lor R) \equiv (P \land Q) \lor (P \land R)$

Some useful laws and equivalences

- excluded middle: $P \vee \neg P$
- double negation: $\neg \neg P \equiv P$
- contrapositive: $P \rightarrow Q \equiv \neg Q \rightarrow \neg P$
- de Morgan's laws
  - $\neg (P \vee Q) \equiv \neg P \wedge \neg Q$
  - $\neg (P \wedge Q) \equiv \neg P \vee \neg Q$
- Note order of *operator precedence*
  - $\neg$ precedes $\wedge$ precedes $\vee$
  - $\rightarrow$ are $\equiv$ are complicated: use parentheses
  - Compare with arithmetic operators: –, x, +

Truth tables

- A truth table has all sentences along its top, usually in increasing order of syntactic complexity
  - its rows are all the possible interpretations, one row each
  - how many rows are needed in general?

P	Q	¬P	P ∧ Q	P ∨ Q	P → Q
T	T	F	T	T	T
T	F	F	F	T	F
F	T	T	F	T	T
F	F	T	F	F	T

Truth tables

Queen Mary University of London

- We can prove things using truth tables
  - $\neg P \vee Q \equiv P \rightarrow Q$

P	Q	$\neg P$	$\neg P \vee Q$	$P \rightarrow Q$	$\neg P \vee Q \equiv P \rightarrow Q$
T	T	F	T	T	T
T	F	F	F	F	T
F	T	T	T	T	T
F	F	T	T	T	T

Proofs in propositional calculus

- Problem Description
  - If it rains in the morning, then I wear my black coat
  - If I wear my black coat, then I wear my black shoes
  - I am not wearing black shoes
  - Did it rain this morning?

Proofs in propositional calculus

- Problem Description
  - If it rains in the morning, then I wear my black coat
  - If I wear my black coat, then I wear my black shoes
  - I am not wearing black shoes
  - Did it rain this morning?
- Propositions
  - P: It rained this morning.
  - Q: I am wearing my black coat.
  - R: I am wearing black shoes.

Proofs in propositional calculus

- Problem Description
  - If it rains in the morning, then I wear my black coat
  - If I wear my black coat, then I wear my black shoes
  - I am not wearing black shoes
  - Did it rain this morning?
- Propositions
  - P: It rained this morning.
  - Q: I am wearing my black coat.
  - R: I am wearing black shoes.
- Premises
  - $P \rightarrow Q$
  - $Q \rightarrow R$
  - $\neg R$

Proofs in propositional calculus

- Problem Description
  - If it rains in the morning, then I wear my black coat
  - If I wear my black coat, then I wear my black shoes
  - I am not wearing black shoes
  - Did it rain this morning?
- Propositions
  - P: It rained this morning.
  - Q: I am wearing my black coat.
  - R: I am wearing black shoes.
- Premises
  - $P \rightarrow Q$
  - $Q \rightarrow R$
  - $\neg R$
- Question: P? (Given that the Premises are true, is P true?)

Proof using a truth table

Queen Mary University of London

Propositions			Premises			Trial conclusions	
P	Q	R	$P \rightarrow Q$	$Q \rightarrow R$	$\neg R$	P	$\neg P$
T	T	T	T	T	F	T	F
T	T	F	T	F	T	T	F
T	F	T	F	T	F	T	F
T	F	F	F	T	T	T	F
F	T	T	T	T	F	F	T
F	T	F	T	F	T	F	T
F	F	T	T	T	F	F	T
F	F	F	T	T	T	F	T

Proof using a truth table

Propositions			Premises			Trial conclusions	
P	Q	R	$P \rightarrow Q$	$Q \rightarrow R$	$\neg R$	P	$\neg P$
T	T	T	T	T	F	T	F
T	T	F	T	F	T	T	F
T	F	T	F	T	F	T	F
T	F	F	F	T	T	T	F
F	T	T	T	T	F	F	T
F	T	F	T	F	T	F	T
F	F	T	T	T	F	F	T
F	F	F	T	T	T	F	T

- When all the premises are true, P is false, so it did not rain this morning

First Order Predicate Calculus

- The Propositional Calculus is not very expressive
  - e.g. can't make statements about all of a certain thing
  - or about things that don't exist
  - or about whether things exist
- In the "rains/coat/shoes" example, we had to omit the day on which we checked the premises
- How could we make statements to capture the idea that we'd do this procedure each day?
  - If it rains on Monday morning ...
  - If it rains on Tuesday morning ... etc.

First Order Predicate Calculus

- The First Order Predicate Calculus (FOPC) is a *conservative extension* of the Propositional Calculus (PC)
  - this means that it has all the properties and features of PC
  - and some extra ones
    - constant symbols: stand for objects, the things which sentences are about; written like propositions, but occur in different syntactic positions
    - variables: usually written as lower case single letters, ranging over objects
    - predicate symbols: propositions are now predicates which describe relationships between (and properties of) objects; written like propositions with arguments
    - function symbols: represent mappings between objects and objects; written like predicates
    - existential quantifier $\exists$: "there exists"; always followed by a variable and a sentence
    - universal quantifier $\forall$: "for all"; always followed by a variable and a sentence
- In PC, propositions were predicates that had no arguments

Rain example in FOPC

- Problem description
  - If it rains in the morning [on a particular day], then I wear my black coat [on that day].
  - If I wear my black coat [on a particular day], then I wear black shoes [on that day].
  - I am not wearing black shoes [today].
  - Did it rain in the morning [today]?

Rain example in FOPC

- Problem description
  - If it rains in the morning [on a particular day], then I wear my black coat [on that day].
  - If I wear my black coat [on a particular day], then I wear black shoes [on that day].
  - I am not wearing black shoes [today].
  - Did it rain in the morning [today]?
- Premises:
  - $\forall d$ Rains(d) → BlackCoat(d)
  - $\forall d$ BlackCoat(d) → BlackShoes(d)
  - ¬BlackShoes(Tuesday)

Rain example in FOPC

- Problem description
  - If it rains in the morning [on a particular day], then I wear my black coat [on that day].
  - If I wear my black coat [on a particular day], then I wear black shoes [on that day].
  - I am not wearing black shoes [today].
  - Did it rain in the morning [today]?
- Premises:
  - $\forall d\ Rains(d) \rightarrow BlackCoat(d)$
  - $\forall d\ BlackCoat(d) \rightarrow BlackShoes(d)$
  - $\neg BlackShoes(Tuesday)$
- Question: Rains(Tuesday)?

Rain example in FOPC

- Problem description
  - If it rains in the morning [on a particular day], then I wear my black coat [on that day].
  - If I wear my black coat [on a particular day], then I wear black shoes [on that day].
  - I am not wearing black shoes [today].
  - Did it rain in the morning [today]?
- Premises:
  - $\forall d$ Rains(d) $\rightarrow$ BlackCoat(d)
  - $\forall d$ BlackCoat(d) $\rightarrow$ BlackShoes(d)
  - $\neg$BlackShoes(Tuesday)
- Question: Rains(Tuesday)?
- Note that quantifiers have the lowest precedence, so $\forall x\ P \rightarrow Q$ means $\forall x\ (P \rightarrow Q)$ and not $(\forall x\ P) \rightarrow Q$ (if in doubt, use parentheses)

Function and quantifier examples

- A function maps its arguments to a fixed single value
  - note that functions do not have truth values: they map *between objects*
  - functions are denoted in the same way as predicates
    - you can tell which is which from where they appear: predicates are outermost
  - functions have an *arity*: the number of arguments they take

Function and quantifier examples

Queen Mary
University of London

- A function maps its arguments to a fixed single value
  - note that functions do not have truth values: they map *between objects*
  - functions are denoted in the same way as predicates
    - you can tell which is which from where they appear: predicates are outermost
  - functions have an *arity*: the number of arguments they take
- "A person's mother is that person's parent"
  - $\forall x$ Person(x) → Parent(Mother-of(x), x)
  - Note that a person can only have one mother, so using a function like this is OK

Function and quantifier examples

- A function maps its arguments to a fixed single value
  - note that functions do not have truth values: they map *between objects*
  - functions are denoted in the same way as predicates
    - you can tell which is which from where they appear: predicates are outermost
  - functions have an *arity*: the number of arguments they take
- "A person's mother is that person's parent"
  - $\forall x\ Person(x) \rightarrow Parent(Mother\text{-}of(x), x)$
  - Note that a person can only have one mother, so using a function like this is OK
- "All computers have a mouse connected by USB"
  - $\forall x\ Computer(x) \rightarrow \exists y\ Mouse(y) \wedge USB\text{-}Connection(x, y)$

Function and quantifier examples

- A function maps its arguments to a fixed single value
  - note that functions do not have truth values: they map *between objects*
  - functions are denoted in the same way as predicates
    - you can tell which is which from where they appear: predicates are outermost
  - functions have an *arity*: the number of arguments they take
- "A person's mother is that person's parent"
  - $\forall x$ Person(x) $\rightarrow$ Parent(Mother-of(x), x)
  - Note that a person can only have one mother, so using a function like this is OK
- "All computers have a mouse connected by USB"
  - $\forall x$ Computer(x) $\rightarrow$ $\exists y$ Mouse(y) $\wedge$ USB-Connection(x, y)
- "There is at least one person in this class who thinks"
  - $\exists x$ Person(x) $\wedge$ Registered(x, AIClass) $\wedge$ Thinks(x)

Order and range of quantifiers matters

Queen Mary University of London

- "Every person likes some food"
  - $\forall x\ Person(x) \rightarrow \exists f\ Food(f) \wedge Likes(x, f)$

Order and range of quantifiers matters

- "Every person likes some food"
  - $\forall x \, Person(x) \rightarrow \exists f \, Food(f) \wedge Likes(x, f)$
- "There is a food that every person likes"
  - $\exists f \, \forall x \, Food(f) \wedge Person(x) \rightarrow Likes(x, f)$

Order and range of quantifiers matters

- "Every person likes some food"
  - $\forall x\ Person(x) \rightarrow \exists f\ Food(f) \wedge Likes(x, f)$
- "There is a food that every person likes"
  - $\exists f\ \forall x\ Food(f) \wedge Person(x) \rightarrow Likes(x, f)$
- "Whenever anyone eats some spicy food, they are happy"
  - $\forall x\ \forall f\ Eats(x, f) \wedge Spicy(f) \rightarrow Happy(x)$
    - allowable substitutions for x are people, for f is food (like types in programming languages)
  - $\forall x\ \forall f\ Person(x) \wedge Food(f) \wedge Spicy(f) \wedge Eats(x, f) \rightarrow Happy(x)$
    - no need to worry about allowable substitutions

Order and range of quantifiers matters

- "Every person likes some food"
  - $\forall x\ Person(x) \rightarrow \exists f\ Food(f) \wedge Likes(x, f)$
- "There is a food that every person likes"
  - $\exists f\ \forall x\ Food(f) \wedge Person(x) \rightarrow Likes(x, f)$
- "Whenever anyone eats some spicy food, they are happy"
  - $\forall x\ \forall f\ Eats(x, f) \wedge Spicy(f) \rightarrow Happy(x)$
    - allowable substitutions for x are people, for f is food (like types in programming languages)
  - $\forall x\ \forall f\ Person(x) \wedge Food(f) \wedge Spicy(f) \wedge Eats(x, f) \rightarrow Happy(x)$
    - no need to worry about allowable substitutions
- Compare the following statements about integers. Which is true?
  - $\forall x\ \exists y\ \ x > y$
  - $\exists y\ \forall x\ \ x > y$

Equality

- A very useful extra operator that isn't strictly in FOPC is =
  - i.e., the TEST for equality, like == in Java, not the assignment statement
- The rule for = is that
  - A = A is true for all constants A in the interpretation
  - otherwise, it is false
- We'll use equality in some tutorial questions

Syntax of FOPC

- Terms: correspond with things in the world (like nouns in grammar)
  - Constants
    - e.g., Thursday, Socrates, 25
  - Variables
    - e.g., x
  - Function expressions
    - A function symbol of arity n followed by n terms, enclosed in () and separated by ,
    - e.g., Function(var, AnotherFunction(Thing))

Syntax of FOPC

- Terms: correspond with things in the world (like nouns in grammar)
  - Constants
    - e.g., Thursday, Socrates, 25
  - Variables
    - e.g., x
  - Function expressions
    - A function symbol of arity n followed by n terms, enclosed in () and separated by ,
    - e.g., Function(var, AnotherFunction(Thing))
- Sentences: statements that can be true or false
  - Atomic Sentence
    - A predicate symbol of arity n followed by n terms, enclosed in () and separated by ,
    - Note that n can be 0, so True and False are atomic sentences
  - The result of applying a connective (as in PC) to one or more sentences
  - The result of applying a quantifier (∀, ∃) with its variable to a sentence

Semantics of FOPC: Interpretation

Queen Mary University of London

- Let the *domain* D be a nonempty set of *objects*, which may related in various ways:
    - An n-ary *relation* is a set of n-tuples of elements of D (i.e. those n-tuples for which the relation holds)
        - unary relations represent *properties* of objects
    - An n-ary *function* is a relation between n-tuples and objects in D, which maps each n-tuple to exactly one object

Semantics of FOPC: Interpretation

- Let the *domain* D be a nonempty set of *objects*, which may related in various ways:
  - An n-ary *relation* is a set of n-tuples of elements of D (i.e. those n-tuples for which the relation holds)
    - unary relations represent *properties* of objects
  - An n-ary *function* is a relation between n-tuples and objects in D, which maps each n-tuple to exactly one object
- An *interpretation* over D is an *assignment* of the entities in D to each of the constant, variable, predicate, and function symbols of a predicate calculus expression
  - Each constant is assigned an element of D
  - Each variable is assigned to a nonempty subset of D (*allowable substitutions*)
  - Each function of arity m is defined ($D^m \longmapsto D$)
  - Each predicate of arity n is defined ($D^n \longmapsto \{T,F\}$).

Domains

Syntax	Semantic Domain	World
	Interpretation	

Domains

Queen Mary University of London

Syntax	Semantic Domain	World
		
	Interpretation	

Domains

Syntax

Semantic Domain

Objects:

Predicates:

Interpretation

World

Domains

Queen Mary University of London

Syntax

Semantic Domain

Objects: Edna

Predicates:

Interpretation

World

Domains

Queen Mary
University of London

Syntax	Semantic Domain	World

Objects: Edna

Fido

Predicates:

Interpretation

Domains

Queen Mary University of London

Syntax	Semantic Domain	World

Objects: Edna

Fido

Predicates: DogWalk/3

Interpretation

Domains

Queen Mary
University of London

Syntax

Semantic Domain

Objects: Edna

Fido

Park

Predicates: DogWalk/3

Interpretation

World

Domains

Queen Mary University of London

Syntax	Semantic Domain	World
Constant names: Edna Fido Park	Objects: Edna Fido Park	
Predicate names: DogWalk/3	Predicates: DogWalk/3	

Interpretation

Domains

Queen Mary University of London

Syntax

Constant names: Edna Fido Park

Predicate names: DogWalk/3

DogWalk(Edna, Fido, Park)

Semantic Domain

Objects: Edna

Fido

Park

Predicates: DogWalk/3

Interpretation

World

Domains

Syntax	Semantic Domain	World
Constant names: Edna Fido Park	Objects: Edna	
Predicate names: DogWalk/3	Fido	
	Park	
DogWalk(Edna, Fido, Park)	Predicates: DogWalk/3	

Interpretation

Domains

Queen Mary
University of London

Syntax	Semantic Domain	World
Constant names: Edna Fido Park	Objects: Edna	
Predicate names: DogWalk/3	Fido	
DogWalk(Edna, Fido, Park)	Park	
	Predicates: DogWalk/3	

Interpretation

Domains

Syntax

Constant names: Edna Fido Park

Predicate names: DogWalk/3

DogWalk(Edna, Fido, Park)

Semantic Domain

Objects: Edna

Fido

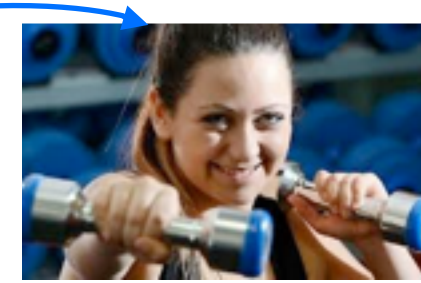
Mehwesh

Park

Predicates: DogWalk/3

Interpretation

World



Domains

Queen Mary
University of London

Syntax

Constant names: Edna Fido Park

Predicate names: DogWalk/3

DogWalk(Edna, Fido, Park)

Semantic Domain

Objects: Edna

Fido

Gym

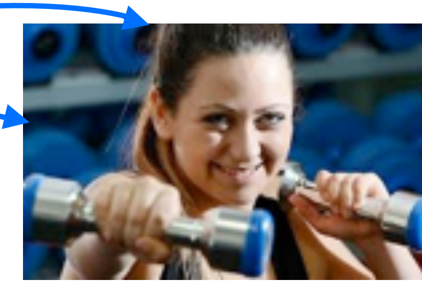
Mehwesh

Park

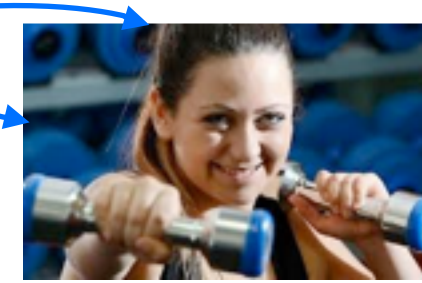
Predicates: DogWalk/3

Interpretation

World



Domains

Syntax	Semantic Domain	World
Constant names: Edna Fido Park	Objects: Edna	
Predicate names: DogWalk/3	Fido	
DogWalk(Edna, Fido, Park)	Gym	
	Mehwesh	
	Park	
	Predicates: DogWalk/3	
	In/2	

Interpretation

Domains

Syntax

Constant names: Edna Fido Park

Predicate names: DogWalk/3

DogWalk(Edna, Fido, Park)

Semantic Domain

Objects: Edna

Fido

Gym

Mehwesh

Park

Weights

Predicates: DogWalk/3

In/2

Interpretation

World

Domains

Queen Mary University of London

Syntax

Constant names: Edna Fido Park

Predicate names: DogWalk/3

DogWalk(Edna, Fido, Park)

Semantic Domain

Objects: Edna

Fido

Gym

Mehwesh

Park

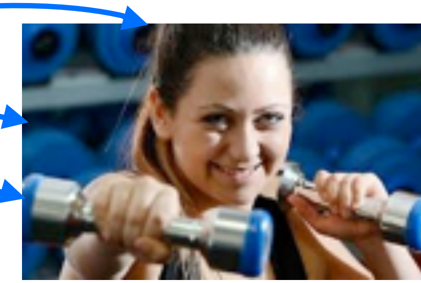
Weights

Predicates: DogWalk/3

In/2 Lifts/2

Interpretation

World



Domains

Queen Mary
University of London

Syntax

Constant names: Edna Fido Park

Predicate names: DogWalk/3

DogWalk(Edna, Fido, Park)

Constant names: Mehwesh Gym Weights

Predicate names: In/2 Lifts/2

Semantic Domain

Objects: Edna

Fido

Gym

Mehwesh

Park

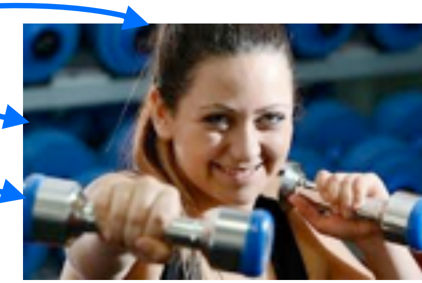
Weights

Predicates: DogWalk/3

In/2 Lifts/2

Interpretation

World



Domains

Queen Mary University of London

Syntax

Constant names: Edna Fido Park

Predicate names: DogWalk/3

DogWalk(Edna, Fido, Park)

Constant names: Mehwesh Gym Weights

Predicate names: In/2 Lifts/2

Lifts(Mehwesh, Weights) $\wedge$ In(Mehwesh, Gym)

Semantic Domain

Objects: Edna

Fido

Gym

Mehwesh

Park

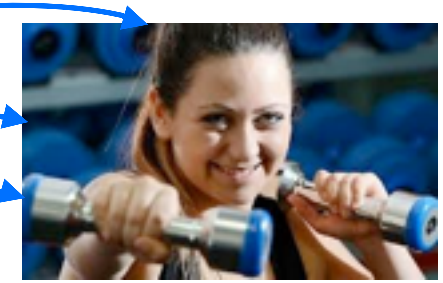
Weights

Predicates: DogWalk/3

In/2 Lifts/2

Interpretation

World



Domains

Queen Mary University of London

Syntax

Constant names: Edna Fido Park

Predicate names: DogWalk/3

DogWalk(Edna, Fido, Park)

Constant names: Mehwesh Gym Weights

Predicate names: In/2 Lifts/2

Lifts(Mehwesh, Weights) ∧ In(Mehwesh, Gym)

Semantic Domain

Objects: Edna

Fido

Gym

Mehwesh

Park

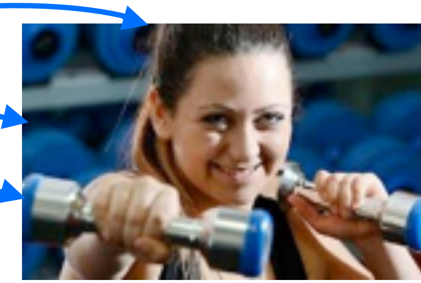
Weights

Predicates: DogWalk/3

In/2 Lifts/2

Interpretation

World



Domains

Queen Mary
University of London

Syntax

Constant names: Edna Fido Park

Predicate names: DogWalk/3

DogWalk(Edna, Fido, Park)

Constant names: Mehwesh Gym Weights

Predicate names: In/2 Lifts/2

Lifts(Mehwesh, Weights) ∧ In(Mehwesh, Gym)

Semantic Domain

Objects: Edna

Dave

Fido

Gym

Mehwesh

Park

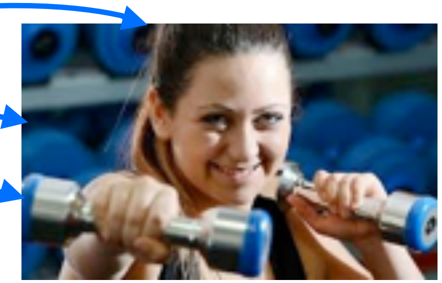
Weights

Predicates: DogWalk/3

In/2 Lifts/2

Interpretation

World



Domains

Queen Mary
University of London

Syntax

Constant names: Edna Fido Park

Predicate names: DogWalk/3

DogWalk(Edna, Fido, Park)

Constant names: Mehwesh Gym Weights

Predicate names: In/2 Lifts/2

Lifts(Mehwesh, Weights) ∧ In(Mehwesh, Gym)

Semantic Domain

Objects: Edna

Dave

Fido

Gym

Mehwesh

Park

Weights

Predicates: Boat/1 DogWalk/3

In/2 Lifts/2

Interpretation

World

Domains

Syntax

Constant names: Edna Fido Park

Predicate names: DogWalk/3

DogWalk(Edna, Fido, Park)

Constant names: Mehwesh Gym Weights

Predicate names: In/2 Lifts/2

Lifts(Mehwesh, Weights) ∧ In(Mehwesh, Gym)

Semantic Domain

Objects: Edna

Dave

Fido

Gym

Mehwesh

Park

Sea

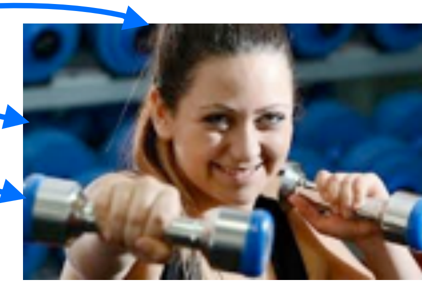
Weights

Predicates: Boat/1 DogWalk/3

In/2 Lifts/2

Interpretation

World



Domains

Syntax

Constant names: Edna Fido Park
Predicate names: DogWalk/3

DogWalk(Edna, Fido, Park)

Constant names: Mehwesh Gym Weights
Predicate names: In/2 Lifts/2

Lifts(Mehwesh, Weights) ∧ In(Mehwesh, Gym)

Semantic Domain

Objects: Edna

Dave

Fido

Gym

Mehwesh

Park

Sea

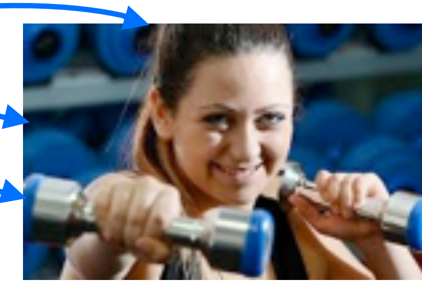
Weights

Predicates: Boat/1 DogWalk/3

In/2 Lifts/2 On/2

Interpretation

World



Domains

Queen Mary
University of London

Syntax

Constant names: Edna Fido Park
Predicate names: DogWalk/3

DogWalk(Edna, Fido, Park)

Constant names: Mehwesh Gym Weights
Predicate names: In/2 Lifts/2

Lifts(Mehwesh, Weights) ∧ In(Mehwesh, Gym)

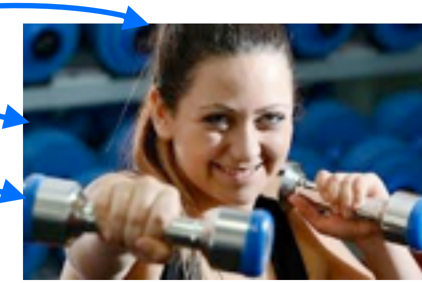
Semantic Domain

Objects: Edna Boat Dave Fido Gym Mehwesh Park Sea Weights

Predicates: Boat/1 DogWalk/3 In/2 Lifts/2 On/2

Interpretation

World



Domains

Syntax

Constant names: Edna Fido Park
Predicate names: DogWalk/3

DogWalk(Edna, Fido, Park)

Constant names: Mehwesh Gym Weights
Predicate names: In/2 Lifts/2

Lifts(Mehwesh, Weights) ∧
In(Mehwesh, Gym)

Constant names: Dave Sea
Predicate names: On/2 Boat/1

Semantic Domain

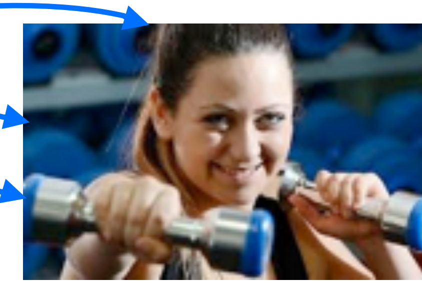
Objects: Edna Boat Dave Fido Gym Mehwesh Park Sea Weights

Predicates: Boat/1 DogWalk/3

In/2 Lifts/2 On/2

Interpretation

World



Domains

Queen Mary
University of London

Syntax

Constant names: Edna Fido Park
Predicate names: DogWalk/3

DogWalk(Edna, Fido, Park)

Constant names: Mehwesh Gym Weights
Predicate names: In/2 Lifts/2

Lifts(Mehwesh, Weights) ∧
In(Mehwesh, Gym)

Constant names: Dave Sea
Predicate names: On/2 Boat/1

$\exists b$ Boat(b) ∧
On(Dave, b) ∧
On(b, Sea)

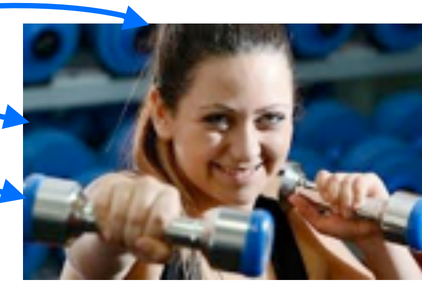
Semantic Domain

Objects: Edna Boat Dave Fido Gym Mehwesh Park Sea Weights

Predicates: Boat/1 DogWalk/3 In/2 Lifts/2 On/2

Interpretation

World



Domains

Queen Mary
University of London

Syntax

Constant names: Edna Fido Park
Predicate names: DogWalk/3

DogWalk(Edna, Fido, Park)

Constant names: Mehwesh Gym Weights
Predicate names: In/2 Lifts/2

Lifts(Mehwesh, Weights) ∧
In(Mehwesh, Gym)

Constant names: Dave Sea
Predicate names: On/2 Boat/1

∃b Boat(b) ∧
On(Dave, b) ∧
On(b, Sea)

Semantic Domain

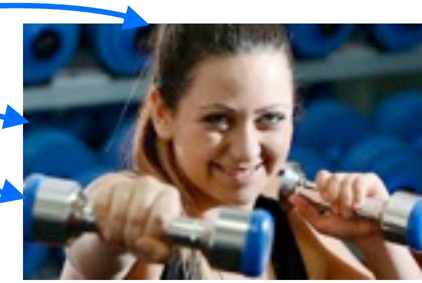
Objects: Edna Boat Dave Fido Gym Mehwesh Park Sea Weights

Predicates: Boat/1 DogWalk/3
In/2 Lifts/2 On/2

Interpretation

DogWalk(Edna, Fido, Park) T
DogWalk(Dave, Fido, Park) F

World



Domains

Queen Mary
University of London

Syntax

Constant names: Edna Fido Park
Predicate names: DogWalk/3

DogWalk(Edna, Fido, Park)

Constant names: Mehwesh Gym Weights
Predicate names: In/2 Lifts/2

Lifts(Mehwesh, Weights) ∧
In(Mehwesh, Gym)

Constant names: Dave Sea
Predicate names: On/2 Boat/1

∃b Boat(b) ∧
On(Dave, b) ∧
On(b, Sea)

Semantic Domain

Objects: Edna Boat Dave Fido Gym Mehwesh Park Sea Weights

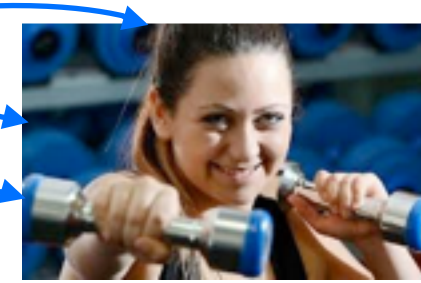
Predicates: Boat/1 DogWalk/3 In/2 Lifts/2 On/2

Interpretation

DogWalk(Edna, Fido, Park)	T
DogWalk(Dave, Fido, Park)	F
Boat(Boat)	T
Boat(Fido)	F

etc.

World



Properties of sentences

- For a predicate calculus sentence, S, and an interpretation, I,
  - I *satisfies* S, if S has a truth value of T under I and at least one variable assignment
  - I *is a model of* S, if I satisfies S for all possible variable assignments in I

Properties of sentences

Queen Mary
University of London

- For a predicate calculus sentence, S, and an interpretation, I,
  - I *satisfies* S, if S has a truth value of T under I and at least one variable assignment
  - I *is a model of* S, if I satisfies S for all possible variable assignments in I
- A sentence is *satisfiable* iff there is at least one interpretation and variable assignment that satisfy it; otherwise it is *unsatisfiable*

Properties of sentences

Queen Mary
University of London

- For a predicate calculus sentence, S, and an interpretation, I,
  - I *satisfies* S, if S has a truth value of T under I and at least one variable assignment
  - I *is a model of* S, if I satisfies S for all possible variable assignments in I
- A sentence is *satisfiable* iff there is at least one interpretation and variable assignment that satisfy it; otherwise it is *unsatisfiable*
- A set of sentences, E, is *satisfiable* iff there is at least one interpretation and variable assignment that satisfies every $S \in E$
  - NB quantification! The same interpretation/variable assignment pair satisfies all S

Properties of sentences

Queen Mary University of London

- For a predicate calculus sentence, S, and an interpretation, I,
  - I *satisfies* S, if S has a truth value of T under I and at least one variable assignment
  - I *is a model of* S, if I satisfies S for all possible variable assignments in I
- A sentence is *satisfiable* iff there is at least one interpretation and variable assignment that satisfy it; otherwise it is *unsatisfiable*
- A set of sentences, E, is *satisfiable* iff there is at least one interpretation and variable assignment that satisfies every $S \in E$
  - NB quantification! The same interpretation/variable assignment pair satisfies all S
- A set of sentences is *inconsistent* iff it is not satisfiable

Properties of sentences

- For a predicate calculus sentence, S, and an interpretation, I,
  - I *satisfies* S, if S has a truth value of T under I and at least one variable assignment
  - I *is a model of* S, if I satisfies S for all possible variable assignments in I
- A sentence is *satisfiable* iff there is at least one interpretation and variable assignment that satisfy it; otherwise it is *unsatisfiable*
- A set of sentences, E, is *satisfiable* iff there is at least one interpretation and variable assignment that satisfies every $S \in E$
  - NB quantification! The same interpretation/variable assignment pair satisfies all S
- A set of sentences is *inconsistent* iff it is not satisfiable
- A sentence is *valid* iff it is satisfiable for all possible interpretations

Proof procedures

Queen Mary
University of London

- A proof procedure consists of
  - a set of inference rules
  - an algorithm for applying the inference rules
    - usually, we start from the thing we want to prove
    - then work "backwards" towards things we already know, such as axioms and theorems

Proof procedures

Queen Mary
University of London

- A proof procedure consists of
    - a set of inference rules
    - an algorithm for applying the inference rules
        - usually, we start from the thing we want to prove
        - then work "backwards" towards things we already know, such as axioms and theorems
- Semantics of logical entailment
    - A sentence, S, *logically follows from*, or *is entailed by*, a set, E, of sentences iff every interpretation and variable assignment that satisfies E also satisfies S

Properties of inference rules

Queen Mary
University of London

- Soundness
  - An set of inference rules is *sound* iff every sentence it infers from a set, E, of sentences logically follows from E
- Completeness
  - An set of inference rules is *complete* iff it can infer every expression that logically follows from a set of sentences

Inference rules

Queen Mary University of London

- Modus Ponens (implication elimination)
  - if we know that P implies Q, and that P is true, then infer Q
  - $(P \wedge (P \rightarrow Q)) \rightarrow Q$

$$\frac{P, \quad P \rightarrow Q}{Q}$$

Inference rules

- Modus Ponens (implication elimination)
  - if we know that P implies Q, and that P is true, then infer Q
  - $(P \wedge (P \rightarrow Q)) \rightarrow Q$

$$\frac{P, \quad P \rightarrow Q}{Q}$$

- Modus Tollens
  - given that P implies Q, and that Q is false, infer $\neg P$
  - $(\neg Q \wedge (P \rightarrow Q)) \rightarrow \neg P$

$$\frac{\neg Q, \quad P \rightarrow Q}{\neg P}$$

Inference rules

- Modus Ponens (implication elimination)
  - if we know that P implies Q, and that P is true, then infer Q
  - $(P \wedge (P \rightarrow Q)) \rightarrow Q$

$$\frac{P, \quad P \rightarrow Q}{Q}$$

- Modus Tollens
  - given that P implies Q, and that Q is false, infer $\neg P$
  - $(\neg Q \wedge (P \rightarrow Q)) \rightarrow \neg P$

$$\frac{\neg Q, \quad P \rightarrow Q}{\neg P}$$

- We also need rules to deal with the other connectives
  - Introduction (adding a connective into a proof sequence)
  - Elimination (removing a connective from a proof sequence)

Inference rules

- Conjunction (And) elimination
  - P is true and Q is true if $P \wedge Q$ is true

$$\frac{P \wedge Q}{P, \quad Q}$$

Inference rules

Queen Mary University of London

- Conjunction (And) elimination
  - P is true and Q is true if P ∧ Q is true

$$\frac{P \wedge Q}{P, \quad Q}$$

- Conjunction (And) introduction
  - P ∧ Q is true if P is true and Q is true

$$\frac{P, \quad Q}{P \wedge Q}$$

Inference rules

Queen Mary University of London

- Conjunction (And) elimination
  - P is true and Q is true if $P \wedge Q$ is true

$$\frac{P \wedge Q}{P, \quad Q}$$

- Conjunction (And) introduction
  - $P \wedge Q$ is true if P is true and Q is true

$$\frac{P, \quad Q}{P \wedge Q}$$

- Universal (For-all) elimination
  - P(A) is true for all constants, A, if $\forall x.P(x)$ is true

$$\frac{\forall x.P(x)}{P(A)}$$

Inference rules

- Conjunction (And) elimination
  - P is true and Q is true if $P \wedge Q$ is true

$$\frac{P \wedge Q}{P, \quad Q}$$

- Conjunction (And) introduction
  - $P \wedge Q$ is true if P is true and Q is true

$$\frac{P, \quad Q}{P \wedge Q}$$

- Universal (For-all) elimination
  - P(A) is true for all constants, A, if $\forall x.P(x)$ is true

$$\frac{\forall x.P(x)}{P(A)}$$

- Universal (For-all) introduction
  - $\forall x.P(x)$ is true, if $P(A_i)$ is true for all constants, A_i

$$\frac{P(A_1), \ldots, P(A_n)}{\forall x.P(x)}$$

Rain example in FOPC revisited

Queen Mary University of London

- Problem description
  - If it rains in the morning [on a particular day], then I wear my black coat [on that day]
  - If I wear my black coat [on a particular day], then I wear my black shoes [on that day]
  - I am not wearing my black shoes [today].
  - Did it rain in the morning [today]?
- Premises:
  - $\forall d$ Rains-in-morning(d) → Black-coat(d)
  - $\forall d$ Black-coat(d) → Black-shoes(d)
  - ¬Black-shoes(Tuesday)
- Question: Rains-in-morning(Tuesday)?

Rain example in FOPC revisited

Queen Mary
University of London

Universal instantiation

$\forall d$ Black-coat(d) → Black-shoes(d)

Modus Tollens

¬ Black-shoes(Tue)

Black-coat(Tue) → Black-shoes(Tue)

$\forall d$ Rains(d) → Black-coat(d)

Universal instantiation

¬ Black-coat(Tue)

Rains(Tue) → Black-coat(Tue)

Modus Tollens

¬Rains(Tue)

- Premises:
  - $\forall d$ Rains(d) → Black-coat(d)
  - $\forall d$ Black-coat(d) → Black-shoes(d)
  - ¬Black-shoes(Tue)

Rain example in FOPC revisited

Queen Mary University of London

Universal instantiation ∀d Black-coat(d)→Black-shoes(d)

Modus Tollens ¬ Black-shoes(Tue) Black-coat(Tue)→Black-shoes(Tue)

∀d Rains(d)→Black-coat(d)

Universal instantiation

¬ Black-coat(Tue)

Rains(Tue)→Black-coat(Tue)

Modus Tollens

¬Rains(Tue)

- Premises:
  - ∀d Rains(d)→Black-coat(d)
  - ∀d Black-coat(d)→Black-shoes(d)
  - ¬Black-shoes(Tue)
- Proof is complicated: which inference rule to use next?
- A simpler approach is better:
  - Resolution Theorem Proving

Past **Exam**ple Question

3(a) Use a truth table to verify that $((A \land B) \rightarrow C) \equiv (\neg A \lor \neg B \lor C)$

[4 marks]

Past Example Question

Queen Mary University of London

3(a) Use a truth table to verify that $((A \wedge B) \rightarrow C) \equiv (\neg A \vee \neg B \vee C)$

[4 marks]

A	B	C	$A \wedge B$	$(A \wedge B) \rightarrow C$	$\neg A$	$\neg B$	$\neg A \vee \neg B \vee C$	LHS $\equiv$ RHS
T	T	T	T	T	F	F	T	T
T	T	F	T	F	F	F	F	T
T	F	T	F	T	F	T	T	T
T	F	F	F	T	F	T	T	T
F	T	T	F	T	T	F	T	T
F	T	F	F	T	T	F	T	T
F	F	T	F	T	T	T	T	T
F	F	F	F	T	T	T	T	T

Past **Exam**ple Question

Queen Mary University of London

3(b) Using the following predicates and their natural language meanings:

cat(x):	x is a cat
dog(x):	x is a dog
owns(x, y):	x owns y
grey(x):	x is grey

express the following sentences in first order logic:

(i) John has a cat.

(ii) Dogs are never grey.

(iii) All of John's cats are grey.

(iv) No dog owner owns any cats. [8 marks]