Adapted from AIMA slides

First-Order Logic

Peter Antal
antal@mit.bme.hu

Outline

- Why FOL?
- Syntax and semantics of FOL
- Knowledge engineering in FOL
- Inference in FOL
  - Reducing first-order inference to propositional inference
  - Unification
  - Generalized Modus Ponens
    - Forward chaining
    - Backward chaining
  - Resolution

Pros and cons for propositional logic

- ☺ Propositional logic is declarative.
- ☺ Propositional logic is compositional:

  meaning of $B_{1,1} \wedge P_{1,2}$ is derived from meaning of $B_{1,1}$ and of $P_{1,2}$

- ☺ Propositional logic allows partial/disjunctive/negated information

  (unlike most data structures and databases)

- ☺ Meaning in propositional logic is context-independent.

  (unlike natural language, where meaning depends on context)

- ☹ Propositional logic has very limited expressive power
  - E.g., cannot say "pits cause breezes in adjacent squares“
    - except by writing one sentence for each square

First-order logic

- Whereas propositional logic assumes the world contains facts,
- first-order logic (like natural language) assumes the world contains
  - Objects: people, houses, numbers, colors, baseball games, wars, …
  - Relations: red, round, prime, brother of, bigger than, part of, comes between, …
  - Functions: father of, best friend, one more than, plus, …

Syntax of FOL: Basic elements

Constants	KingJohn, 2,...
Predicates	Brother, >,...
Functions	Sqrt, LeftLegOf,...
Variables	x, y, a, b,...
Connectives	$\neg, \Rightarrow, \wedge, \vee, \Leftrightarrow$
Equality	$=$
Quantifiers	$\forall, \exists$

Atomic sentences

Atomic sentence = *predicate* $(term_1,...,term_n)$
or $term_1 = term_2$

Term = *function* $(term_1,...,term_n)$
or *constant* or *variable*

- E.g., *Brother(KingJohn,RichardTheLionheart) > (Length(LeftLegOf(Richard)), Length(LeftLegOf(KingJohn)))*

Complex sentences

- Complex sentences are made from atomic sentences using connectives
-

$$\neg S,\ S_1 \wedge S_2,\ S_1 \vee S_2,\ S_1 \Rightarrow S_2,\ S_1 \Leftrightarrow S_2,$$

E.g. *Sibling(KingJohn,Richard)* $\Rightarrow$ *Sibling(Richard,KingJohn)*

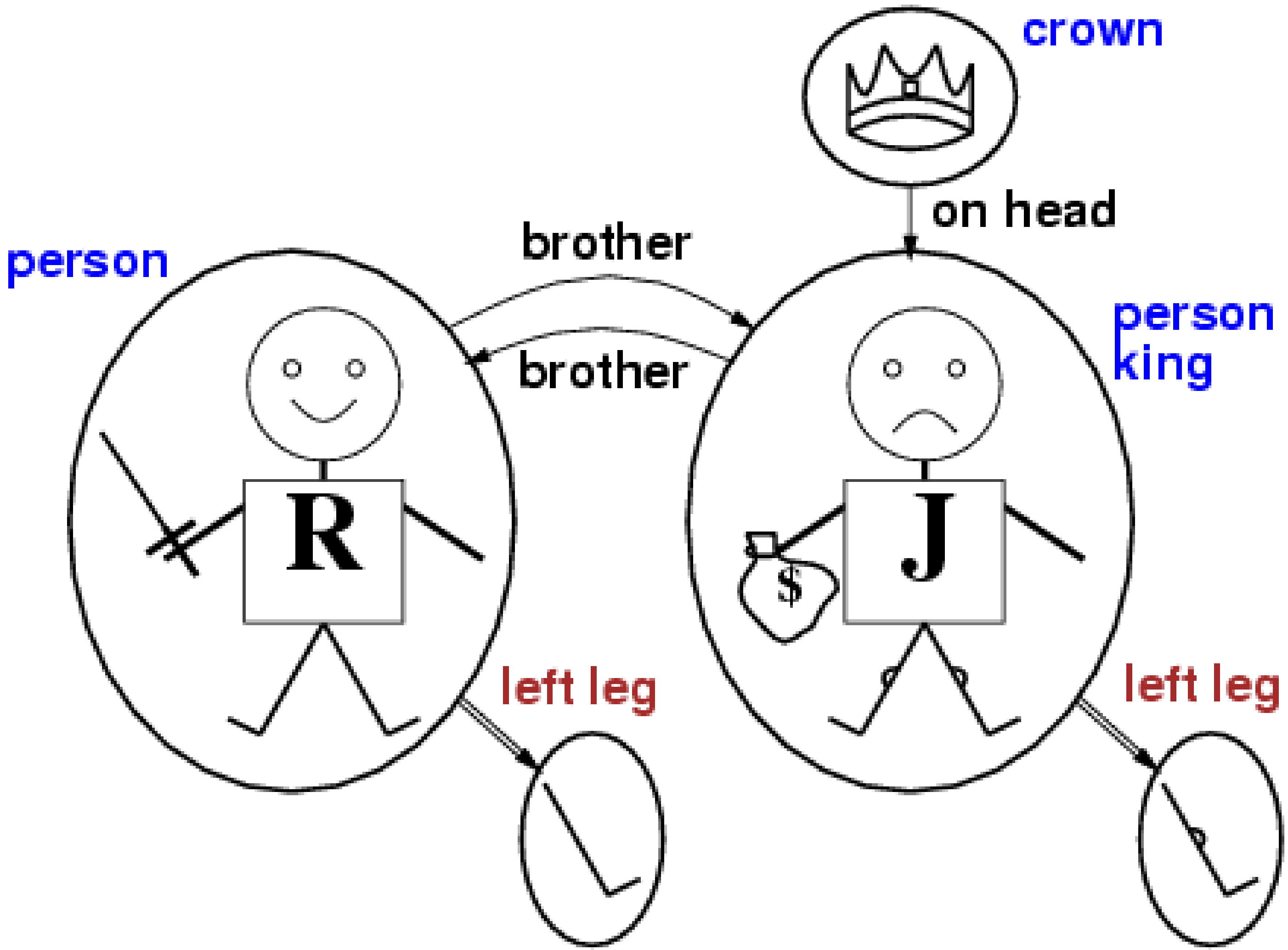
$>(1,2) \vee \leq(1,2)$

$>(1,2) \wedge \neg >(1,2)$

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*($term_1,\ldots,term_n$) is true iff the objects referred to by $term_1,\ldots,term_n$ are in the relation referred to by *predicate*

Models for FOL: Example



Universal quantification

- ∀<*variables*> <*sentence*>
-

Everyone at Y is smart:
$\forall x\ At(x,Y) \Rightarrow Smart(x)$

- ∀x *P* is true in a model *m* iff *P* is true with *x* being each possible object in the model
-
- Roughly speaking, equivalent to the conjunction of instantiations of *P*
- At(KingJohn,NUS) ⇒ Smart(KingJohn)
  ∧ At(Richard,NUS) ⇒ Smart(Richard)
  ∧ At(NUS,NUS) ⇒ Smart(NUS)
  ∧ ...

A common mistake to avoid

- Typically, $\Rightarrow$ is the main connective with $\forall$
-
- Common mistake: using $\wedge$ as the main connective with $\forall$:

  $\forall x\ At(x,Y) \wedge Smart(x)$

  means "Everyone is at Y and everyone is smart"

Existential quantification

- $\exists$<*variables*> <*sentence*>

- Someone at Y is smart:
- $\exists x$ At(x,Y) $\wedge$ Smart(x)$
-
- $\exists x$ *P* is true in a model *m* iff *P* is true with *x* being some possible object in the model
-
- Roughly speaking, equivalent to the disjunction of instantiations of *P*
- At(KingJohn,NUS) $\wedge$ Smart(KingJohn)
  $\vee$ At(Richard,NUS) $\wedge$ Smart(Richard)
  $\vee$ At(NUS,NUS) $\wedge$ Smart(NUS)
  $\vee$ ...

Another common mistake to avoid

- Typically, $\wedge$ is the main connective with $\exists$
- Common mistake: using $\Rightarrow$ as the main connective with $\exists$:
- $$\exists x \; At(x,Y) \Rightarrow Smart(x)$$

  is true if there is anyone who is not at Y!

Properties of quantifiers

- $\forall x \forall y$ is the same as $\forall y \forall x$
- $\exists x \exists y$ is the same as $\exists y \exists x$
- $\exists x \forall y$ is not the same as $\forall y \exists x$
-
- $\exists x \forall y$ Loves(x,y)
  - "There is a person who loves everyone in the world"
  -
- $\forall y \exists x$ Loves(x,y)
  - "Everyone in the world is loved by at least one person"
  -
- Quantifier duality: each can be expressed using the other
- $\forall x$ Likes(x,IceCream) $\neg\exists x \neg$Likes(x,IceCream)
- $\exists x$ Likes(x,Broccoli) $\neg\forall x \neg$Likes(x,Broccoli)

Equality

- $term_1 = term_2$ is true under a given interpretation if and only if $term_1$ and $term_2$ refer to the same object
-
- E.g., definition of *Sibling* in terms of *Parent*:
-

  $\forall x,y\ Sibling(x,y) \Leftrightarrow [\neg(x = y) \wedge \exists m,f\ \neg(m = f) \wedge Parent(m,x) \wedge Parent(f,x) \wedge Parent(m,y) \wedge Parent(f,y)]$

Using FOL

The kinship domain:

- Brothers are siblings
- $\forall x,y \; Brother(x,y) \Leftrightarrow Sibling(x,y)$
- One's mother is one's female parent
- $\forall m,c \; Mother(c) = m \Leftrightarrow (Female(m) \wedge Parent(m,c))$
- "Sibling" is symmetric
- $\forall x,y \; Sibling(x,y) \Leftrightarrow Sibling(y,x)$

Knowledge engineering in FOL

1. Identify the task.
2. Assemble the relevant knowledge.
3. Decide on a vocabulary of predicates, functions, and constants.
4. Encode general knowledge about the domain.
5. Encode a description of the specific problem instance.
6. Pose queries to the inference procedure and get answers.
7. Debug the knowledge base.

Inference in FOL

- Syllogisms
- Reducing first-order inference to propositional inference
- Unification
- Generalized Modus Ponens
- Forward chaining
- Backward chaining
- Resolution

Syllogisms of the First Figure			
	BARBARA		**CELARENT**
A A A	Every B is A. Every C is B. Therefore, every C is A.	E A E	No B is A. Every C is B. Therefore, no C is A.
	DARII		**FERIO**
A I I	Every B is A. Some C is B. Therefore, some C is A.	E I O	No B is A. Some C is B. Therefore, some C is not A.

A universal Affirmative $\forall x.\ B(x) \rightarrow A(x)$
E universal nEgative $\forall x.\ B(x) \rightarrow \neg A(x)$
I partIcular affirmative $\exists x.\ C(x) \wedge B(x)$
O particular nOt affirmative (negative) $\exists x.\ C(x) \wedge \neg B(x)$

BARBARA:

$$\frac{\forall x.\ B(x) \rightarrow A(x) \qquad \forall x.\ C(x) \rightarrow B(x)}{\forall x.\ C(x) \rightarrow A(x)}$$

Syllogisms

BARBARA:

$\forall x.\ B(x) \to A(x)$
$\underline{\forall x.\ C(x) \to B(x)}$
$\forall x.\ C(x) \to A(x)$

DARII:

$\forall x.\ B(x) \to A(x)$
$\underline{\exists x.\ C(x) \land B(x)}$
$\exists x.\ C(x) \land A(x)$

CELARENT:

$\forall x.\ B(x) \to \neg A(x)$
$\underline{\forall x.\ C(x) \to B(x)}$
$\forall x.\ C(x) \to \neg A(x)$

FERIO:

$\forall x.\ B(x) \to \neg A(x)$
$\underline{\exists x.\ C(x) \land B(x)}$
$\exists x.\ C(x) \land \neg A(x)$

Fig. I.

CESARE:

$\forall x.\ B(x) \to \neg A(x)$
$\underline{\forall x.\ C(x) \to A(x)}$
$\forall x.\ C(x) \to \neg B(x)$

CAMESTRES:

$\forall x.\ B(x) \to A(x)$
$\underline{\forall x.\ C(x) \to \neg A(x)}$
$\forall x.\ C(x) \to \neg B(x)$

FESTIMO:

$\forall x.\ B(x) \to \neg A(x)$
$\underline{\exists x.\ C(x) \land A(x)}$
$\exists x.\ C(x) \land \neg B(x)$

BAROCO:

$\forall x.\ B(x) \to A(x)$
$\underline{\exists x.\ C(x) \land \neg A(x)}$
$\exists x.\ C(x) \land \neg B(x)$

Fig. II.

érvelési hiba

DARAPTI:

$\forall x.\ C(x) \to A(x)$
$\underline{\forall x.\ C(x) \to B(x)}$
$\exists x.\ B(x) \land A(x)$

FELAPTON:

$\forall x.\ C(x) \to \neg A(x)$
$\underline{\forall x.\ C(x) \to B(x)}$
$\exists x.\ B(x) \land \neg A(x)$

DISAMIS:

$\exists x.\ C(x) \land A(x)$
$\underline{\forall x.\ C(x) \to B(x)}$
$\exists x.\ B(x) \land A(x)$

DATISI:

$\forall x.\ C(x) \to A(x)$
$\underline{\exists x.\ C(x) \land B(x)}$
$\exists x.\ B(x) \land A(x)$

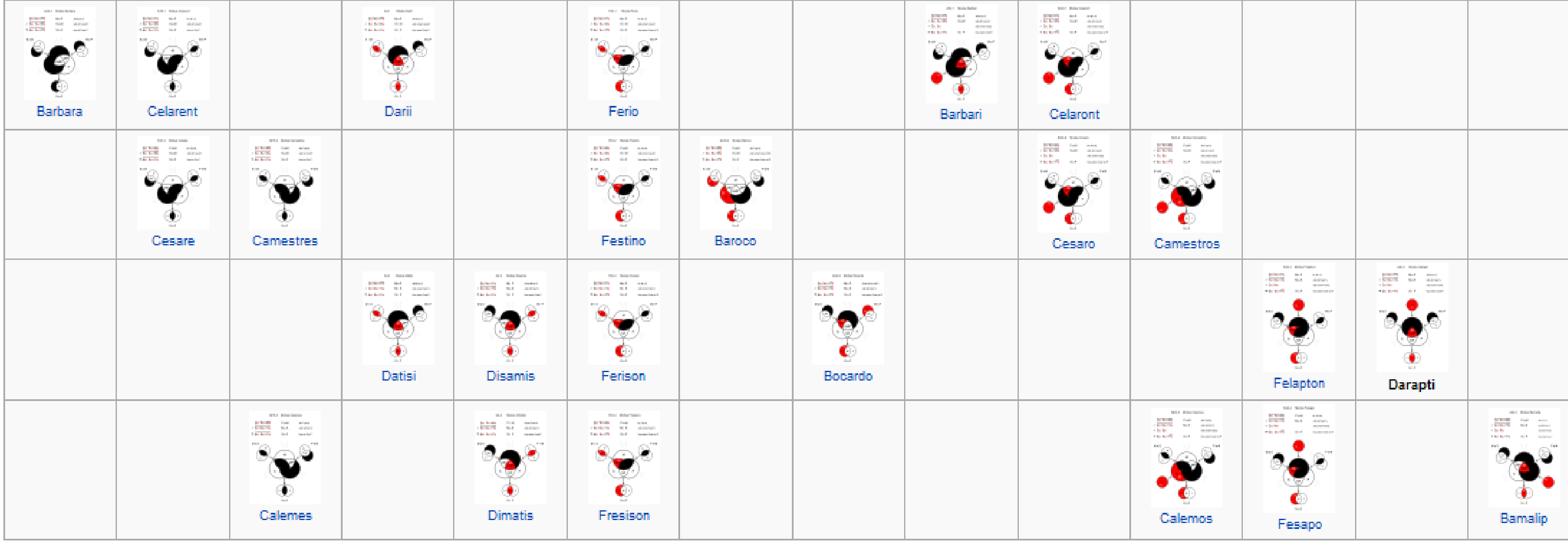
BOCARDO:

$\exists x.\ C(x) \land \neg A(x)$
$\underline{\forall x.\ C(x) \to B(x)}$
$\exists x.\ B(x) \land \neg A(x)$

Fig. III.

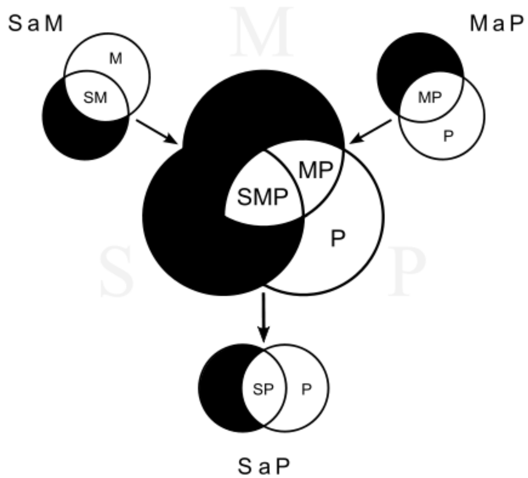
FERISON: $\forall x.\ C(x) \to \neg A(x)$
$\underline{\exists x.\ C(x) \land B(x)}$
$\exists x.\ B(x) \land \neg A(x)$

Fig. IV.



AAA-1 Modus Barbara

$\overline{\exists x: Mx \wedge \overline{Px}}$	M a P	All M are P,
$\wedge\ \overline{\exists x: Sx \wedge \overline{Mx}}$	S a M	and all S are M;
$\Rightarrow \overline{\exists x: Sx \wedge \overline{Px}}$	S a P	thus all S are P.



Reduction to propositional inference

Suppose the KB contains just the following:

$\forall$x King(x) $\wedge$ Greedy(x) $\Rightarrow$ Evil(x)
King(John)
Greedy(John)
Brother(Richard,John)

- Instantiating the universal sentence in all possible ways, we have:
  King(John) $\wedge$ Greedy(John) $\Rightarrow$ Evil(John)
  King(Richard) $\wedge$ Greedy(Richard) $\Rightarrow$ Evil(Richard)
  King(John)
  Greedy(John)
  Brother(Richard,John)

- The new KB is propositionalized: proposition symbols are
-

King(John), Greedy(John), Evil(John), King(Richard), etc.

Reduction contd.

- Every FOL KB can be propositionalized so as to preserve entailment
- (A ground sentence is entailed by new KB iff entailed by original KB)
- Idea: propositionalize KB and query, apply resolution, return result
- Problem: with function symbols, there are infinitely many ground terms,
  - e.g., *Father*(*Father*(*Father*(*John*)))

Reduction contd.

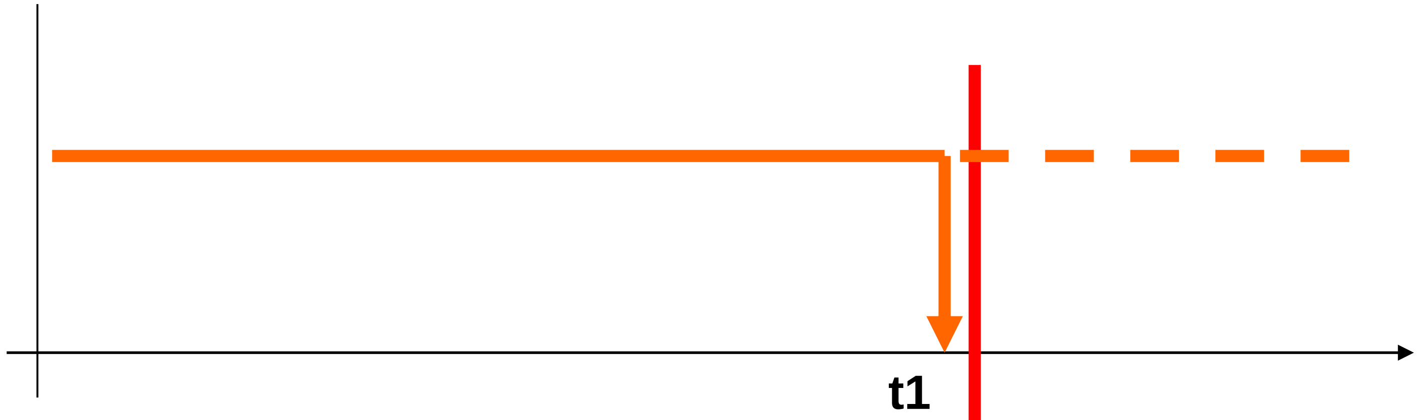
Theorem: Herbrand (1930). If a sentence α is entailed by an FOL KB, it is entailed by a finite subset of the propositionalized KB

Idea: For $n = 0$ to ∞ do
create a propositional KB by instantiating with depth-n terms
see if α is entailed by this KB

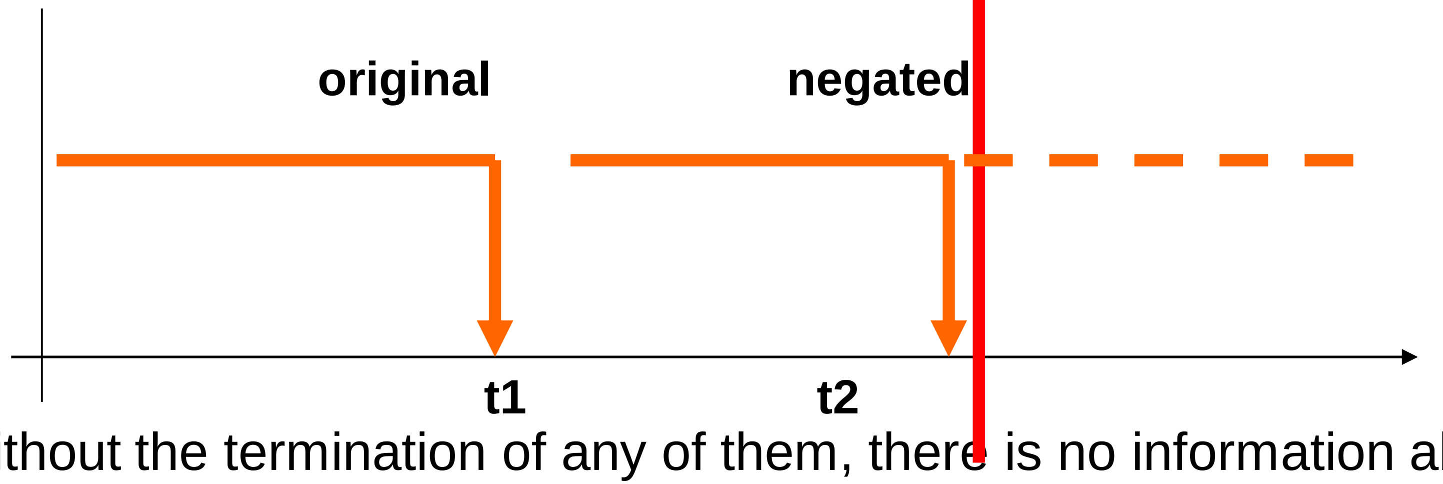
Problem: works if α is entailed, loops if α is not entailed

Theorem: Turing (1936), Church (1936) Entailment for FOL is semidecidable (algorithms exist that say yes to every entailed sentence, but no algorithm exists that also says no to every nonentailed sentence.)

Semidecidability in FOL: effect of finite time on proof



conclusion: if not proved, then false!?



Without the termination of any of them, there is no information about provability/truth.

Problems with propositionalization

- Propositionalization seems to generate lots of irrelevant sentences.
- E.g., from:
-
-

$\forall$x King(x) $\wedge$ Greedy(x) $\Rightarrow$ Evil(x)
King(John)
$\forall$y Greedy(y)
Brother(Richard,John)

- it seems obvious that *Evil*(*John*), but propositionalization produces lots of facts such as *Greedy*(*Richard*) that are irrelevant
-
- With *p* *k*-ary predicates and *n* constants, there are $p \cdot n^k$ instantiations.

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)*
-

$\theta = \{x/John, y/John\}$ works

- $Unify(\alpha,\beta) = \theta$ if $\alpha\theta = \beta\theta$
-

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)	

- 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)

Unification

- To unify *Knows(John,x)* and *Knows(y,z)*,
- θ = {y/John, x/z } or θ = {y/John, x/John, z/John}
- The first unifier is more general than the second.
-
- There is a single most general unifier (MGU) that is unique up to renaming of variables.
- MGU = { y/John, x/z }

Generalized Modus Ponens (GMP)

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

where $p_i'\theta = p_i\,\theta$ for all i

p_1' is *King*(*John*) p_1 is *King*(*x*)

p_2' is *Greedy*(*y*) p_2 is *Greedy*(*x*)

θ is {x/John,y/John} q is *Evil*(*x*)

q θ is *Evil*(*John*)

- GMP used with KB of definite clauses (exactly one positive literal)
- All variables assumed universally quantified
-

Resolution: brief summary

- 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`$(\ell_i, \neg m_j) = \theta$.

- 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 $\theta = \{x/Ken\}$

- Apply resolution steps to CNF(KB $\wedge \neg\alpha$); complete for FOL

Monotonicity

- If $KB_1 \models a$, then $(KB_1 \cup KB_2) \models a$
- Old theorems are not invalidated by additional axioms.
- Robotics:
  - Inferred results remains valid after expanding the knowledge-base with new facts from observations.
- Non-monotonic logics
  - truth-maintenance systems
  - default logic..

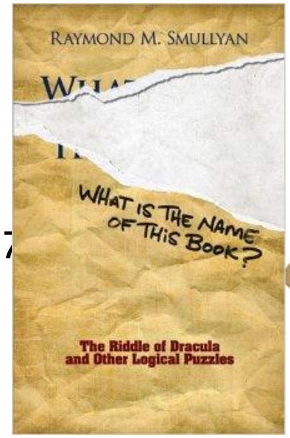
Abductive inference/reasoning

- C.S.Pierce: inference of the most pragmatical explanation for an observation.
- Types of inference
  - Deduction: model→observation
  - Induction: observation(s) → model → observation
    - observation(s) → model
    - observation(s) → [model →] observation
  - Abduction: observation(s) → model
  - Transduction: observation(s) → observation
  - Causal: intervention → effect
  - Counterfactual: (observation/intervention→effect) →(imagery intervention → imagery effect)
- Related to abduction
  - theories of explanation
  - philosophy of science
  - theories of belief change in artificial intelligence
- Subtypes of abduction
  - Common sense
  - Scientific (Ockham's razor)
  - Logical
  - Probabilistic (most probable explanation)
  - Causal (necessary and sufficient cause)

Summary

- First-order logic:
  - objects and relations are semantic primitives
  - syntax: constants, functions, predicates, equality, quantifiers
- Inference
  - Resolution (CNF-based)
  - Semi-decidable

- Suggested reading:
  - Puzzles
    - http://www.greylabyrinth.com/puzzle/puzzle102
    - http://www.greylabyrinth.com/puzzle/puzzle107
  - Interview with R. M. Smullyan
    - http://www.doverpublications.com/mathsci/0227/news.html
  - R. M. Smullyan: *What Is the Name of This Book?*, 197



The Secret Chamber

While excavating an ancient Puzzlanian crypt, you discover an unusual column. The column has four narrow holes bored into it, all at the same height, evenly spaced around the column, just large enough for a human hand. Reading the inscriptions above and below the holes, you realize that this column is part of a complex mechanism that will open a secret chamber. Out of sight within each hole is a switch that can either be up or down; when all the switches are in the same position, the secret chamber will open before you. The column is small enough that you can reach all the way around, so using both hands you could flip any two switches at the same time. Here's the tricky part: As soon as your hand leaves a hole, the column will rapidly spin for a random number of quarter rotations. If you're not careful, you might lose a hand. But you can flip two switches at once, then quickly pull both hands out at the same time. What strategy can you use to open the chamber in a finite, and preferably small, number of attempts? Assume you cannot discriminate between holes after a spin without reaching into the column.

http://www.greylabyrinth.com/puzzle/puzzle102