First-order logic

Chapter 8

Outline

- Why FOL?
- Syntax and semantics of FOL
- Fun with sentences
- Wumpus world in FOL

Thinking about formal languages

- Programming languages = formal languages. Most widely used type…
  - Have "facts" = data structures and contents
  - But!
    - The manipulation of facts is "hard-wired" in domain-specific procedures
    - They are *procedural*, not *declarative*.
  - Declarative: Facts and rules for manipulating stated independently.
    - → Domain-*independent* reasoning system…could be applied to any facts.

- Propositional Logic: Pros and cons
  - ✓ Propositional logic is declarative: pieces of syntax correspond to facts
  - ✓ Propositional logic allows partial/disjunctive/negated information (unlike most data structures and databases)
  - ✓ Propositional logic is compositional:
    - meaning of $B1,1 \wedge P1,2$ is derived from meaning of B1,1 and of P1,2
  - ✓ Meaning in propositional logic is context-independent
    - (unlike natural language, where meaning depends on context)
  - ◆ Propositional logic has very limited expressive power (unlike natural language)
    - E.g., cannot say "pits cause breezes in adjacent squares" except by writing one sentence for each square

First Order Logics

- Propositional Logics: Assumes the world contains facts (only)
  - Individual propositional symbols. May be true or false. Not parameterized.
  - $P_{1,1}$ $\neg B_{1,2}$ $B_{1,1} \Rightarrow P_{1,2} \vee P_{2,1}$

- First Order Logic: More like natural language. Contains:
  - Objects: people, houses, numbers, theories, Ronald McDonald, colors, baseball games, wars, centuries . . .
  - Relations:
    - Propositions: Facts about one object.
      - red, round, bogus, prime, multistoried . . .,
    - Many to many relations: Relate whole groups of objects
      - brother of, bigger than, inside, part of, has color, occurred after, owns, comes between, . . .
  - Functions:
    - Subset of relations: relate multiple “inputs” to a single “output”
    - father of, best friend, third inning of, one more than, end of

- And there are other logics as well:
  - Temporal, Probabilistic, Fuzzy

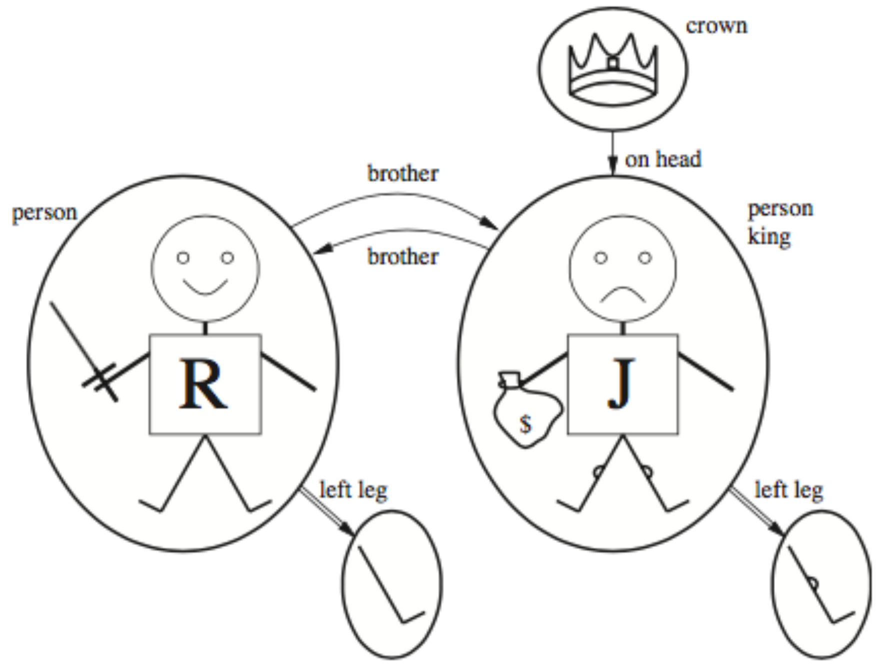
Logics: Quick Big Picture Overview

Language	Ontological Commitment	Epistemological Commitment
Propositional logic	facts	true/false/unknown
First-order logic	facts, objects, relations	true/false/unknown
Temporal logic	facts, objects, relations, times	true/false/unknown
Probability theory	facts	degree of belief
Fuzzy logic	facts + degree of truth	known interval value

Syntax of FOL: Basics

- Models in propositional logics: Link symbols to truth values

- Models in FOLs: Objects!
  - Domain of a FOL model: the set of objects it contains. (must be >0)
  - Objects represent entities that exist in the world.
    - Kings, swords, big toes, wind, rain, etc.



  - Objects can be related in various ways
    - This is the "1st order" aspect of the logic! Reason about relationships...
      - E.g. "Brother" relation={ <john,doug>, <john, ben>, etc. ...   tuples }
    - Unary relations = properties
      - Person(john), hungry(jack)
    - Some relations are functions: only one value.
      - Father(son, papa).
      - Sqrt(16,4)

Syntax of FOL: Closer look

- Three kinds of symbols on FOL:
  - Constant symbols: Stand for objects
    - MyCar, Richard, Kittycat
  - Predicate symbols: stand for relations
    - Brother, ParkedAt, Sparkly, Fat
  - Function symbols: stand for functions
    - Leftleg, Father

- Where does "truth" come from?
  - Propositional logic: simpler → Symbols refer to world features. T/F
  - FOL: Model must provide necessary information to determine truth value
    - Has its set of Constant (Object), Predicate (relations), and function symbols
    - Also has *interpretation*: what world objects/relations specified by above symbols.
  - Example interpretations:
    - "John" refers to John Georgas, "Brother" refers to Brotherhood relation
    - "John" refers to a donor liver in Chad, "Brother" refers to "smaller than".
  - Logics are not truth! Always dependent on human interpretation!
  - Concept of "intended interpretation" = "the obvious one"

Syntax of FOL: Making Sentences

- Logical symbols can be combined into sentences
  - Just like propositional logic. To describe a possible world (model).

- Anatomy of sentences in FOL:
  - Term: Logical expression that refers to an object
    - Constant symbols are terms → named reference to object
    - Could also have descriptive reference to object (we don't name everything!)
      - LeftLeg(John) refers to an (anonymous) object that is John's left leg.
    - Generally: complex terms = functor(term1, term2, term3...)
      - Functor refers to some function in model, terms refer to objects related by function
      - The interpretation of model clarifies/fixes the referent of each term.

  - Atomic Sentences: state facts in the model
    - Brother (John, Richard) → intended interpretation = "Richard is brother of John"
    - Could be more complex/nested: Married(Father(John), Mother(John)).
    - *Sentence is true in a given model, if the specified relation holds in that model*

  - Complex Sentences: Can use the usual logical connectives to compound
    - $\neg$King(Richard) $\Rightarrow$ King(John)
    - $\neg$Brother(LeftLeg(Richard), John)
    - King(Richard) $\vee$ King(John)

Entailment in FOL

- Our goal remains to show/prove/compute *entailment.*
  - KB |= α → show that statement α is true in all models where KB is true.
  - Model checking in propositional logic: generate all possible models, check em.
  - Technically still works perfect in FOL...except model space is HUGE.

- How many models exist for a give world in FOL?
  - By definition: every possible combination of every possible assignment.
  - So: Model space = *all* permutations of *all* factors in an FOL world:
    - For each number of domain elements n from 1 to ∞
    - For each k-ary predicate P_k in the vocabulary For each possible k-ary relation on n objects
    - For each constant symbol C in the vocabulary
    - For each choice of referent for C from n objects . . .

- Computing entailment by enumerating FOL models is not feasible!
  - Model-checking is not an option to compute entailment
  - Need more focused inference-based reasoning!

Universal Quantification

- FOL power! We can make broad statements about the world!
  - State that a logical sentence holds for *all possible instantiations of referents!*

- Format: ∀(variables) (logical sentence, with variables)
  - States: sentences is true for all possible bindings of given variables.
  - $\forall x$ *EnrolledIn(cs470, x)* ⇒ Smart(x)

- Truth: $\forall x \; P$ is true in model m iff P is true with x bound to every possible object in the model.
  - So wait: Above statement true only if it evaluates to true with x bound to "Frank"...and with x bound to "projector" holds? Huh?
    - Yes. Note that it's an *implication* → true *except when* premise false and RHS true.
- Caution:
  - Typically ' ⇒ ' is the main connective with ∀
  - Common mistake: Using ∧ as the main connective
    - ∀x *EnrolledIn(cs470, x)* ∧ Smart(x)
    - "Everyone is enrolled in CS470" and "everyone is smart"

Existential Quantification

- State: a sentence holds for *at least one instantiation of referents!*

- Format: $\exists$ (variables) (logical sentence, with variables)
  - States: sentence is true for *at least one* binding of given variables.
  - $\exists x$ *EnrolledIn(cs470, x)* $\wedge$ Smart(x)

- Truth: $\exists x$ *P* is true in model m iff P is true with x bound to every possible object in the model.
  - So wait: Above statement true only if it evaluates to true with x bound to *some* possible object in the model.
- Caution:
  - Typically '$\wedge$ ' is the main connective with $\forall$
  - Common mistake: Using $\Rightarrow$ as the main connective instead
    - $\exists x$ *EnrolledIn(cs470, x)* $\Rightarrow$ Smart(x)
    - "The existence of someone enrolled in cs470 implies that someone is smart"
    - Ok, but: implication is false only if LHS false and RHS true. True all other times.
    - Above is also true if there is anyone who is NOT enrolled in cs470!

Properties of Quantifiers

- Quantifiers can be nested...but be careful of meaning!
- $\forall x \, \forall y$ is the same as $\forall y \, \forall x$
  - *Usually we just write* $\forall x,y$

- $\exists x \, \exists y$ is the same as $\exists y \, \exists x$
  - *Usually write* $\exists x,y$

- $\exists x \, \forall y$ 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"

- Often good to parenthesize to emphasize quantifier meaning
  - $\forall$x ($\exists$y Loves(x,y))
    - "Everyone loves at least one person"
    - Note: there's nothing saying that x and y can't be bound to same object!

Fun with logic sentences...

- Brothers are siblings

Fun with sentences

Brothers are siblings

$\forall\ x, y\ Brother(x, y) \quad \Rightarrow \ Sibling(x, y)$.

"Sibling" is symmetric

Fun with sentences

Brothers are siblings

$\forall\ x, y\ Brother(x, y) \quad \Rightarrow \ Sibling(x, y).$

"Sibling" is symmetric

$\forall\ x, y\ \ Sibling(x, y) \ \Leftrightarrow\ Sibling(y, x).$

One's mother is one's female parent

Fun with sentences

Brothers are siblings

$\forall\ x, y\ \ Brother(x, y)\ \Rightarrow\ Sibling(x, y).$

“Sibling” is symmetric

$\forall\ x, y\ \ Sibling(x, y)\ \Leftrightarrow\ Sibling(y, x).$

One’s mother is one’s female parent

$\forall\ x, y\ \ Mother(x, y)\ \Leftrightarrow\ (Female(x) \wedge Parent(x, y)).$

A first cousin is a child of a parent’s sibling

Fun with sentences

Brothers are siblings

$\forall x, y \;\; Brother(x, y) \;\Rightarrow\; Sibling(x, y)$.

"Sibling" is symmetric

$\forall x, y \;\; Sibling(x, y) \;\Leftrightarrow\; Sibling(y, x)$.

One's mother is one's female parent

$\forall x, y \;\; Mother(x, y) \;\Leftrightarrow\; (Female(x) \wedge Parent(x, y))$.

A first cousin is a child of a parent's sibling

$\forall x, y \;\; FirstCousin(x, y) \;\Leftrightarrow\; \exists p, ps \;\; Parent(p, x) \wedge Sibling(ps, p) \wedge$
$Parent(ps, y)$

Some closing details

- Equality
  - Problem: The same object could be bound to multiple names.
  - Ex: ∃ x,y Brother(x, Richard) ∧ Brother(y, Richard)
    - "Richard has two brothers"?? No!
    - x, y could be bound to same object → true in models where R has one brother!
  - Solution: need a way to constrain what variables could (or not) refer to.
  - Equality symbol (=) signifies that two terms refer to same object
    - x = y → the object referred to by x is the same on as y.
    - ¬(x = y) → the two are NOT equal. Usually write: x ≠ y
  - So now: ∃ x,y Brother(x, Richard) ∧ Brother(y, Richard) ∧ x ≠ y

- Database Semantics
  - Problem: Brother(John, Richard) ∧ Brother(Jeff, Richard)
    - "john" and "jeff" could be bound to same object! → add john ≠ jeff
    - But sentence still true in models where there are tons of *other* brothers.
  - Non-intuitive and cumbersome. Could constrain semantics a bit...
    - Unique-names assumption: Every constant refers to distinct object.
    - Closed World assumption: atomic sentences (facts) not known to be true are false.
    - Domain Closure: models contain only objects named by constants (no hidden)
  - Not strictly FOL...but often used (e.g. in systems like Prolog)

Using FOL to infer entailments

- Want to:
  - Establish a KB. Express our known axioms about the world
  - Put in new information. New knowledge, percepts.
  - Ask if the current KB entails some logic sentence (query) α

- Use Tell/Ask model for KB agents introduced at start
  - TELL (KB, *King(John)*)
  - TELL (KB, $\forall x$ *King(x)* ⇒ *Person(x))*
  - ASK (KB, *King(John))*
  - Or the real power: ask quantified questions (i.e. with variables)
  - ASK(KB, $\exists$ x Person(x))
    - "Does there exist a person?"
    - Is true...but could be true *many times over*. → could be many bindings for x
  - ASKVARS(KB, Person(x))
    - Modified semantics: return a stream/list of all possible bindings
    - Only works for H-clause KBs → list of *specific* bindings.
    - Person(Sue) Person(Maggie) is true in FOL...but can't yield binding for Person(x)

A simple example

The Mogul World

Start with some rules about the world:

- If you have a rich parent, then you're rich
- If you are ruthless and have powerful friends then you are rich
- Rich people are rotten
- The children of rich people are rotten.

Then we need some facts:

- Trump is rich
- Tim is a powerful friend to Joe and Maggie
- Trump has a child named Ivanka
- Joe is ruthless but Maggie is not

Show some queries being resolved.

- Is Trump rich?   Ivanka?
- Who is a rotten person?
- Who is rich?

Summary: First Order Logic

- Maintains best features of knowledge-based reasoning intro'd in Ch7
  - K-rep is declarative, compositional, context-independent, unambiguous
- FOL is *far* more powerful than propositional logic
  - Reasoning about objects, properties, and their relations. Not just T/F facts
  - Increased power: sufficient to encode Wumpus world (and many others)
- Syntax is similar to what to that of prop. logic (and most other logics)
  - Simple atomic terms ... that can be combined into complex sentences
  - Uses all of the standard logical ops... Plus universal/existential quantification
- A model in FOL is:
  - A set of objects, predicates, functions...plus...
  - An interpretation that connects these meaningless symbols to world objects
- Developing a KB in FOL requires:
  - careful domain analysis to identify relevant objects/predicates/fns for domain
  - Careful encoding of domain axioms and facts into FOL

α β $\subseteq$ $\neg$ $\Rightarrow$ $\models$ $\wedge$ $\vee$
$\Leftrightarrow$