Artificial Intelligence Programming

First-order Logic

Cindi Thompson

Department of Computer Science

University of San Francisco

Representation

- Propositional Logic has several nice features
  - Lets us easily express disjunction and negation
    - “There is a pit in (1,1) or in (2,2)”
    - “There is not a pit in (1,1)”
    - This is hard to do in C/Java/Python - variables can only take on a single value.
    - There’s no obvious way to *assign* x the value “3 or 4” or “some value less than 10”.
  - Separates declarative knowledge from inference procedures
  - Compositional
    - The meaning of a sentence is a function of the meaning of its parts.

Expressivity

- We would like the sorts of structures that are useful in programming languages. In particular, we would like:
  - Objects: Wumpi, pits, gold, vacuum cleaners, etc.
  - Variables: how do we talk about objects without knowing their names?

. . .

Expressivity

- We would like the sorts of structures that are useful in programming languages. In particular, we would like:
  - Objects, Variables
  - Relations: These can include:
    - Unary relations (or properties): Smelly(Wumpus), Shiny(Gold), Sleepy(student), etc.
    - Binary relations: Brother-of(Bart, Lisa), Holding(Agent, Gold), After(Tuesday, Monday)
    - n-ary relations: Simpsons(Homer, Marge, Bart, Lisa)
    - These are sometimes called *predicates*
  - Functions: Father-of(Bart) = Homer, Boss-of(Homer) = Mr_Burns, etc.
- *First-order logic* gives us all of this.

Models in first-order logic

- Recall that a model is the set of “possible worlds” for a collection of sentences.
- In propositional logic, this meant truth assignments to facts.
- In FOL, models have objects in them.
- The *domain* of a model is the set of objects in that world.
- For example, the Simpsons model might have the domain
  - {Marge, Homer, Lisa, Bart, Maggie}
- We can then specify relations and functions between these objects
  - Married-to(Marge, Homer), Baby(Maggie), Father(Bart) = Homer

Models in FOL, Cont'd

Remember, models also contain enough information to determine the truth value of a given sentence.

- Thus we need an *interpretation* that specifies which objects, relations, and functions are referred to by the constant, predicate, and function symbols.
- There are an infinite number of such models since they can contain an infinite number of objects.

More on this when we discuss inference - can't just enumerate models to check our "logic" on entailment.

Terms and sentences

- A *term* is an expression that refers to a single object.
  - Bart, Lisa, Homer
  - We can also use functions as terms - Saxophone(Lisa) refers to the object that is Lisa's saxophone
- An *atomic sentence* consists of a function (aka predicate) applied to terms
  - Brother-of(Lisa, Bart), Married(Homer, Marge), Married(Mother(Lisa), Father(Bart))
  - Plays(Lisa, Saxophone(Lisa))

Terms and sentences

- A Complex sentence uses logical connectives $\neg, \vee, \wedge, \Rightarrow, \Leftrightarrow$ to join atomic sentences.
  - $\neg BrotherOf(Homer, Bart)$,
  - $MotherOf(Lisa, Marge) \Rightarrow MotherOf(Bart, Marge)$
  - $Oldest(Bart) \vee Oldest(Lisa)$
- We can also use equality to relate objects: $Homer = Father(Bart)$

Quantifiers and variables

- Often, it's not enough to make a statement about particular objects. Instead, we want to make a statement about some or all objects.
  - "All of the Simpsons are yellow."
  - "At least one of the Simpsons is a baby."
  - Quantifiers allow us to do this.
  - $\forall$ is the symbol for universal quantification
    - It means that a sentence holds for every object in the domain.
    - $\forall x Simpson(x) \Rightarrow Yellow(x)$

Quantifiers and variables

- $\exists$ is the symbol for existential quantification
  - It means that the sentence is true for at least one element in the domain.
  - $\exists x Female(x) \wedge PlaysSaxophone(x)$
  - What would happen if I said $\exists x Female(x) \Rightarrow PlaysSaxophone(x)$?

Quantifiers

- In general, $\Rightarrow$ makes sense with $\forall$ ($\wedge$ is usually too strong).
- $\wedge$ makes sense with $\exists$ ($\Rightarrow$ is generally too weak.)
- Some examples:
    - One of the Simpsons works at a nuclear plant.
    - All of the Simpsons are cartoon characters.
    - There is a Simpson with blue hair and a green dress.
    - There is a Simpson who doesn't have hair.

Nesting quantifiers

- Often, we'll want to express more complex quantifications. For example, "every person has a mother"
  - $\forall x \exists y Mother(x, y)$
  - Notice the scope - for each x, a different y is (potentially) chosen.
- What if we said $\exists y \forall x Mother(x, y)$?
- this is not a problem when nesting quantifiers of the same type.
- $\forall x \forall y \; BrotherOf(x, y) \Rightarrow SiblingOf(x, y)$ and $\forall y \forall x \; BrotherOf(x, y) \Rightarrow SiblingOf(x, y)$ are equivalent.
- We often write that as $\forall x, y \; BrotherOf(x, y) \Rightarrow SiblingOf(x, y)$

Negation

- We can negate quantifiers
  - $\neg \forall x \, yellow(x)$ says that it is not true that everyone is yellow.
  - $\exists x \, \neg yellow(x)$ has the same meaning - there is someone who is not yellow.
  - $\neg \exists x \, daughterOf(Bart, x)$ says that there does not exist anyone who is Bart's daughter.
  - $\forall x \, \neg daughterOf(Bart, x)$ says that for all individuals they are not Bart's daughter.
- In fact, we can use DeMorgan's rules with quantifiers just like with $\wedge$ and $\vee$.

More examples

- A husband is a male spouse
  - $\forall x, y\ husband(x, y) \Leftrightarrow spouse(x, y) \wedge male(x)$
- Two siblings have a parent in common
  - $\forall x, y\ sibling(x, y) \Leftrightarrow$
    $\neg(x = y) \wedge \exists p Parent(x, p) \wedge Parent(y, p)$
- Everyone who goes to Moe's likes either Homer or Barney (but not both)
  - $\forall x\ goesTo(Moes, x) \Rightarrow$
    $(Likes(x, Homer) \Leftrightarrow \neg Likes(x, Barney))$

More examples

- Everyone knows someone who is angry at Homer.
  - $\forall x \exists y\, knows(x, y) \wedge angryAt(y, Homer)$
- Everyone who works at the power plant is scared of Mr. Burns
  - $\forall x\, worksAt(PowerPlant, x) \Rightarrow scaredOf(x, Burns)$

Audience Participation

- Everyone likes Lisa.
- Someone who works at the power plant doesn't like Homer. (both ways)
- Bart, Lisa, and Maggie are Marge's only children.
- People who go to Moe's are depressed.
- There is someone in Springfield who is taller than everyone else.
- When a person is fired from the power plant, they go to Moe's
- Everyone loves Krusty except Sideshow Bob
- Only Bart skateboards to school
- Someone with large feet robbed the Quickie-mart.

Representing useful knowledge

- We can use FOL to represent class/subclass information, causality, existence, and disjoint sets.
- Example: Let's suppose we are interested in building an agent that can help recommend music.
- We want it to be able to reason about musical artists, songs, albums, and genres.
- It would be tedious to enter every bit of information about every artist; instead, we'll enter some rules and let our agent derive entailed knowledge.

Music example

- $\forall x\ Genre(x, Punk) \rightarrow Genre(x, Rock)$ - subclass: all Punk songs are Rock songs.
- $Member(JohnLennon, Beatles) \wedge Member(PaulMcCartney, Beatles) \wedge Member(GeorgeHarrison, Beatles) \wedge Member(RingoStarr, Beatles) \wedge \forall x\ Member(x, Beatles) \rightarrow x \in \{John, Paul, George, Ringo\}$ - exclusive membership: John, Paul, George, and Ringo are the Beatles.
- $PerformedBy(Beatles, WhiteAlbum)$ The WhiteAlbum is a Beatles album
- $\forall x, y, z\ Member(x, y) \wedge PerformedBy(y, z) \rightarrow PlayedOn(x, z)$ if someone is a member of a group, and that group performed an album, then that person played on that album.

Music example

- $Genre(HolidaysInTheSun, Punk)$ - "Holidays in the Sun" is a Punk song.
- $\forall x\ Genre(x, Rock) \rightarrow Likes(Bob, x)$ Bob likes all rock songs.
  - We should be able to infer that Bob will like "Holidays in the Sun"
- $\forall w, x, y, z\ Likes(x, y) \wedge Member(z, y) \wedge PerformedBy(z, w) \rightarrow Likes(x, w)$ - If someone likes a band Y, and Z is a member of band Y, then that person will like albums (W) performed by person Z.

We'll look at how inference in FOL works after the midterm…