

# **Artificial Intelligence**

## **First-Order Logic**

Instructor: Vincent Conitzer

# Limitations of propositional logic

- So far we studied propositional logic
- Some English statements are hard to model in propositional logic:
- “If your roommate is wet because of rain, your roommate must not be carrying **any** umbrella”
- Pathetic attempt at modeling this:
- RoommateWetBecauseOfRain =>  
(NOT(RoommateCarryingUmbrella0) AND  
NOT(RoommateCarryingUmbrella1) AND  
NOT(RoommateCarryingUmbrella2) AND ...)

# Problems with propositional logic

- No notion of **objects**
- No notion of **relations among objects**
- RoommateCarryingUmbrella0 is instructive **to us**, suggesting
  - there is an object we call Roommate,
  - there is an object we call Umbrella0,
  - there is a relationship Carrying between these two objects
- Formally, none of this meaning is there
  - Might as well have replaced RoommateCarryingUmbrella0 by P

# Elements of first-order logic

- **Objects:** can give these names such as Umbrella0, Person0, John, Earth, ...
- **Relations:** Carrying(., .), IsAnUmbrella(.)
  - Carrying(Person0, Umbrella0),  
IsUmbrella(Umbrella0)
  - Relations with one object = **unary relations** = **properties**
- **Functions:** Roommate(.)
  - Roommate(Person0)
- **Equality:** Roommate(Person0) = Person1

# Things to note about functions

- It could be that we have a separate name for Roommate(Person0)
- E.g., Roommate(Person0) = Person1
- ... but we do not **need** to have such a name
  
- A function can be applied to any object
- E.g., Roommate(Umbrella0)

# Reasoning about many objects at once

- **Variables:**  $x, y, z, \dots$  can refer to multiple objects
- New operators “for all” and “there exists”
  - **Universal quantifier** and **existential quantifier**
- for all  $x$ :  $\text{CompletelyWhite}(x) \Rightarrow$   
 $\text{NOT}(\text{PartiallyBlack}(x))$ 
  - Completely white objects are never partially black
- there exists  $x$ :  $\text{PartiallyWhite}(x) \text{ AND}$   
 $\text{PartiallyBlack}(x)$ 
  - There exists some object in the world that is partially white and partially black

# Practice converting English to first-order logic

- “John has Jane’s umbrella”
- $\text{Has}(\text{John}, \text{Umbrella}(\text{Jane}))$
- “John has an umbrella”
- there exists  $y$ :  $(\text{Has}(\text{John}, y) \text{ AND } \text{IsUmbrella}(y))$
- “Anything that has an umbrella is not wet”
- for all  $x$ :  $((\text{there exists } y: (\text{Has}(x, y) \text{ AND } \text{IsUmbrella}(y))) \Rightarrow \text{NOT}(\text{IsWet}(x)))$
- “Any person who has an umbrella is not wet”
- for all  $x$ :  $(\text{IsPerson}(x) \Rightarrow ((\text{there exists } y: (\text{Has}(x, y) \text{ AND } \text{IsUmbrella}(y))) \Rightarrow \text{NOT}(\text{IsWet}(x))))$

# More practice converting English to first-order logic

- “John has at least two umbrellas”
- there exists  $x$ : (there exists  $y$ : (Has(John,  $x$ ) AND IsUmbrella( $x$ ) AND Has(John,  $y$ ) AND IsUmbrella( $y$ ) AND NOT( $x=y$ )))
- “John has at most two umbrellas”
- for all  $x, y, z$ : ((Has(John,  $x$ ) AND IsUmbrella( $x$ ) AND Has(John,  $y$ ) AND IsUmbrella( $y$ ) AND Has(John,  $z$ ) AND IsUmbrella( $z$ ))  $\Rightarrow$  ( $x=y$  OR  $x=z$  OR  $y=z$ ))



# Even more practice converting English to first-order logic...

- “Duke’s basketball team defeats any other basketball team”
- for all  $x$ :  $((\text{IsBasketballTeam}(x) \text{ AND NOT}(x=\text{BasketballTeamOf}(\text{Duke}))) \Rightarrow \text{Defeats}(\text{BasketballTeamOf}(\text{Duke}), x))$
- “Every team defeats some other team”
- for all  $x$ :  $(\text{IsTeam}(x) \Rightarrow (\text{there exists } y: (\text{IsTeam}(y) \text{ AND NOT}(x=y) \text{ AND Defeats}(x,y))))$

## More realistically...

- “Any basketball team that defeats Duke’s basketball team in one year will be defeated by Duke’s basketball team in a future year”
- for all  $x, y$ : (IsBasketballTeam( $x$ ) AND IsYear( $y$ ) AND DefeatsIn( $x$ , BasketballTeamOf(Duke),  $y$ )) => there exists  $z$ : (IsYear( $z$ ) AND IsLaterThan( $z, y$ ) AND DefeatsIn(BasketballTeamOf(Duke),  $x$ ,  $z$ ))

# Is this a tautology?

- “Property P implies property Q, or property Q implies property P (or both)”
- for all x:  $((P(x) \Rightarrow Q(x)) \text{ OR } (Q(x) \Rightarrow P(x)))$
- $(\text{for all } x: (P(x) \Rightarrow Q(x))) \text{ OR } (\text{for all } x: (Q(x) \Rightarrow P(x)))$

# Relationship between universal and existential

- for all  $x$ :  $a$
- is equivalent to
- $\text{NOT}(\text{there exists } x: \text{NOT}(a))$

# Something we cannot do in first-order logic

- We are **not** allowed to reason in general about relations and functions
- The following would correspond to **higher-order logic** (which is more powerful):
- “If John is Jack’s roommate, then any property of John is also a property of Jack’s roommate”
- $(\text{John}=\text{Roommate}(\text{Jack})) \Rightarrow \text{for all } p: (p(\text{John}) \Rightarrow p(\text{Roommate}(\text{Jack})))$
- “If a property is inherited by children, then for any thing, if that property is true of it, it must also be true for any child of it”
- $\text{for all } p: (\text{IsInheritedByChildren}(p) \Rightarrow (\text{for all } x, y: ((\text{IsChildOf}(x,y) \text{ AND } p(y)) \Rightarrow p(x))))$

# Axioms and theorems

- **Axioms**: basic facts about the domain, our “initial” knowledge base
- **Theorems**: statements that are logically derived from axioms

# SUBST

- SUBST replaces one or more variables with something else
- For example:
  - $\text{SUBST}(\{x/\text{John}\}, \text{IsHealthy}(x) \Rightarrow \text{NOT}(\text{HasACold}(x)))$   
gives us
  - $\text{IsHealthy}(\text{John}) \Rightarrow \text{NOT}(\text{HasACold}(\text{John}))$

# Instantiating quantifiers

- From
- for all  $x$ :  $a$
- we can obtain
- $\text{SUBST}(\{x/g\}, a)$
  
- From
- there exists  $x$ :  $a$
- we can obtain
- $\text{SUBST}(\{x/k\}, a)$
- where  $k$  is a constant that does not appear elsewhere in the knowledge base (**Skolem constant**)
- Don't need original sentence anymore



# Instantiating existentials after universals

- for all  $x$ : there exists  $y$ :  $\text{IsParentOf}(y,x)$
- WRONG: for all  $x$ :  $\text{IsParentOf}(k, x)$
- RIGHT: for all  $x$ :  $\text{IsParentOf}(k(x), x)$
- Introduces a new function (**Skolem function**)
- ... again, assuming  $k$  has not been used previously

# Generalized modus ponens

- for all  $x$ : Loves(John,  $x$ )
  - John loves every thing
- for all  $y$ : (Loves( $y$ , Jane)  $\Rightarrow$  FeelsAppreciatedBy(Jane,  $y$ ))
  - Jane feels appreciated by every thing that loves her
- Can infer from this:
- FeelsAppreciatedBy(Jane, John)
  
- Here, we used the substitution  $\{x/\text{Jane}, y/\text{John}\}$ 
  - Note we used different variables for the different sentences
- General UNIFY algorithms for finding a good substitution

# Keeping things as general as possible in unification

- Consider EdibleByWith
  - e.g., EdibleByWith(Soup, John, Spoon) – John can eat soup with a spoon
- for all x: for all y: EdibleByWith(Bread, x, y)
  - Anything can eat bread with anything
- for all u: for all v: (EdibleByWith(u, v, Spoon) => CanBeServedInBowlTo(u,v))
  - Anything that is edible with a spoon by something can be served in a bowl to that something
- Substitution: {x/z, y/Spoon, u/Bread, v/z}
- Gives: for all z: CanBeServedInBowlTo(Bread, z)
- Alternative substitution {x/John, y/Spoon, u/Bread, v/John} would only have given CanBeServedInBowlTo(Bread, John), which is not as general

# Resolution for first-order logic

- for all  $x$ : ( $\text{NOT}(\text{Knows}(\text{John}, x)) \text{ OR } \text{IsMean}(x) \text{ OR } \text{Loves}(\text{John}, x)$ )
  - John loves everything he knows, with the possible exception of mean things
- for all  $y$ : ( $\text{Loves}(\text{Jane}, y) \text{ OR } \text{Knows}(y, \text{Jane})$ )
  - Jane loves everything that does not know her
- What can we unify? What can we conclude?
- Use the substitution:  $\{x/\text{Jane}, y/\text{John}\}$
- Get:  $\text{IsMean}(\text{Jane}) \text{ OR } \text{Loves}(\text{John}, \text{Jane}) \text{ OR } \text{Loves}(\text{Jane}, \text{John})$
- Complete (i.e., if not satisfiable, will find a proof of this), **if** we can remove literals that are duplicates after unification
  - Also need to put everything in **canonical form** first

# Notes on inference in first-order logic

- Deciding whether a sentence is entailed is **semidecidable**: there are algorithms that will eventually produce a proof of any entailed sentence
- It is not **decidable**: we cannot always conclude that a sentence is not entailed

# (Extremely informal statement of) Gödel's Incompleteness Theorem

- First-order logic is not rich enough to model basic arithmetic
- For any consistent system of axioms that is rich enough to capture basic arithmetic (in particular, mathematical induction), there exist true sentences that cannot be proved from those axioms

# A more challenging exercise

- Suppose:
  - There are exactly 3 objects in the world,
  - If  $x$  is the spouse of  $y$ , then  $y$  is the spouse of  $x$  (spouse is a function, i.e., everything has a spouse)
- Prove:
  - Something is its own spouse

# More challenging exercise

- there exist  $x, y, z$ :  $(\text{NOT}(x=y) \text{ AND } \text{NOT}(x=z) \text{ AND } \text{NOT}(y=z))$
- for all  $w, x, y, z$ :  $(w=x \text{ OR } w=y \text{ OR } w=z \text{ OR } x=y \text{ OR } x=z \text{ OR } y=z)$
- for all  $x, y$ :  $((\text{Spouse}(x)=y) \Rightarrow (\text{Spouse}(y)=x))$
- for all  $x, y$ :  $((\text{Spouse}(x)=y) \Rightarrow \text{NOT}(x=y))$  (*for the sake of contradiction*)
- *Try to do this on the board...*



# Umbrellas in first-order logic

- You know the following things:
  - You have exactly one other person living in your house, who is wet
  - If a person is wet, it is because of the rain, the sprinklers, or both
  - If a person is wet because of the sprinklers, the sprinklers must be on
  - If a person is wet because of rain, that person must not be carrying any umbrella
  - There is an umbrella that “lives in” your house, which is not in its house
  - An umbrella that is not in its house must be carried by some person who lives in that house
  - You are not carrying any umbrella
- Can you conclude that the sprinklers are on?

# Theorem prover on the web

- <https://webspass.spass-prover.org/>
- `begin_problem(TinyProblem).`
- `list_of_descriptions.`
- `name({*TinyProblem*}).`
- `author({*CPS570*}).`
- `status(unknown).`
- `description({*Just a test*}).`
- `end_of_list.`
- `list_of_symbols.`
- `predicates[(F,1),(G,1)].`
- `end_of_list.`
- `list_of_formulae(axioms).`
- **`formula(exists([U],F(U))).`**
- **`formula(forall([V],implies(F(V),G(V)))).`**
- `end_of_list.`
- `list_of_formulae(conjectures).`
- **`formula(exists([W],G(W))).`**
- `end_of_list.`
- `end_problem.`

# Theorem prover on the web...

- `begin_problem(ThreeSpouses).`
- `list_of_descriptions.`
- `name(*ThreeSpouses*).`
- `author(*CPS570*).`
- `status(unknown).`
- `description(*Three Spouses*).`
- `end_of_list.`
- `list_of_symbols.`
- `functions[spouse].`
- `end_of_list.`
- `list_of_formulae(axioms).`
- **`formula(exists([X],exists([Y],exists([Z],and(not(equal(X,Y)),and(not(equal(X,Z)),not(equal(Y,Z)))))))).`**
- **`formula(forall([W],forall([X],forall([Y],forall([Z],or(equal(W,X),or(equal(W,Y),or(equal(W,Z),or(equal(X,Y),or(equal(X,Z),equal(Y,Z)))))))))).`**
- **`formula(forall([X],forall([Y],implies(equal(spouse(X),Y),equal(spouse(Y),X)))).`**
- `end_of_list.`
- `list_of_formulae(conjectures).`
- **`formula(exists([X],equal(spouse(X),X))).`**
- `end_of_list.`
- `end_problem.`

# Theorem prover on the web...

- begin\_problem(TwoOrThreeSpouses).
- list\_of\_descriptions.
- name({\*TwoOrThreeSpouses\*}).
- author({\*CPS570\*}).
- status(unknown).
- description({\*TwoOrThreeSpouses\*}).
- end\_of\_list.
- list\_of\_symbols.
- functions[spouse].
- end\_of\_list.
- list\_of\_formulae(axioms).
- **formula(exists([X],exists([Y],not(equal(X,Y))))).**
- **formula(forall([W],forall([X],forall([Y],forall([Z],or(equal(W,X),or(equal(W,Y),or(equal(W,Z),or(equal(X,Y),or(equal(X,Z),equal(Y,Z)))))))))).**
- **formula(forall([X],forall([Y],implies(equal(spouse(X),Y),equal(spouse(Y),X))))).**
- end\_of\_list.
- list\_of\_formulae(conjectures).
- **formula(exists([X],equal(spouse(X),X))).**
- end\_of\_list.
- end\_problem.

# Theorem prover on the web...

- `begin_problem(FiveSpouses).`
- `list_of_descriptions.`
- `name({*FiveSpouses*}).`
- `author({*CPS570*}).`
- `status(unknown).`
- `description({*Five Spouses*}).`
- `end_of_list.`
- `list_of_symbols.`
- `functions[spouse].`
- `end_of_list.`
- `list_of_formulae(axioms).`
- **`formula(exists([X],exists([Y],exists([Z],exists([V],exists([W],and(not(equal(X,Y)),and(not(equal(X,Z)),and(not(equal(Y,Z)),and(not(equal(X,V)),and(not(equal(Y,V)),and(not(equal(Z,V)),and(not(equal(X,W)),and(not(equal(Y,W)),and(not(equal(Z,W)),not(equal(V,W)))))))))))))))).`**
- **`formula(forall([W],forall([X],forall([Y],forall([Z],forall([U],forall([V],or(equal(W,X),or(equal(W,Y),or(equal(W,Z),or(equal(X,Y),or(equal(X,Z),or(equal(Y,Z),or(equal(X,U),or(equal(Y,U),or(equal(Z,U),or(equal(W,U),or(equal(X,V),or(equal(Y,V),or(equal(Z,V),or(equal(W,V),or(equal(U,V)))))))))))))))))))).`**
- `formula(forall([X],forall([Y],implies(equal(spouse(X),Y),equal(spouse(Y),X))))).`
- `end_of_list.`
- `list_of_formulae(conjectures).`
- `formula(exists([X],equal(spouse(X),X))).`
- `end_of_list.`
- `end_problem.`

# Theorem prover on the web...

- `begin_problem(Umbrellas).`
- `list_of_descriptions.`
- `name({*Umbrellas*}).`
- `author({*CPS570*}).`
- `status(unknown).`
- `description({*Umbrellas*}).`
- `end_of_list.`
- `list_of_symbols.`
- `functions[(House,1),(You,0)].`
- `predicates[(Person,1),(Wet,1),(WetDueToR,1),(WetDueToS,1),(SprinklersOn,0),(Umbrella,1),(Carrying,2),(NotAtHome,1)].`
- `end_of_list.`
- `list_of_formulae(axioms).`
- `formula(forall([X],forall([Y],implies(and(Person(X),and(Person(Y),and(not(equal(X,You)),and(not(equal(Y,You)),and(equal(House(X),House(You))),equal(House(Y),House(You))))))),equal(X,Y))))).`
- `formula(exists([X],and(Person(X),and(equal(House(You),House(X)),and(not(equal(X,You)),Wet(X)))))).`
- `formula(forall([X],implies(and(Person(X),Wet(X)),or(WetDueToR(X),WetDueToS(X))))).`
- `formula(forall([X],implies(and(Person(X),WetDueToS(X)),SprinklersOn))).`
- `formula(forall([X],implies(and(Person(X),WetDueToR(X)),forall([Y],implies(Umbrella(Y),not(Carrying(X,Y)))))).`
- `formula(exists([X],and(Umbrella(X),and(equal(House(X),House(You)),NotAtHome(X))))).`
- `formula(forall([X],implies(and(Umbrella(X),NotAtHome(X)),exists([Y],and(Person(Y),and(equal(House(X),House(Y)),Carrying(Y,X)))))).`
- `formula(forall([X],implies(Umbrella(X),not(Carrying(You,X))))).`
- `end_of_list.`
- `list_of_formulae(conjectures).`
- `formula(SprinklersOn).`
- `end_of_list.`
- `end_problem.`

# Applications

- Some serious novel mathematical results proved
- Verification of hardware and software
  - Prove outputs satisfy required properties for all inputs
- Synthesis of hardware and software
  - Try to prove that there exists a program satisfying such and such properties, **in a constructive way**
- Also: contributions to planning (up next)