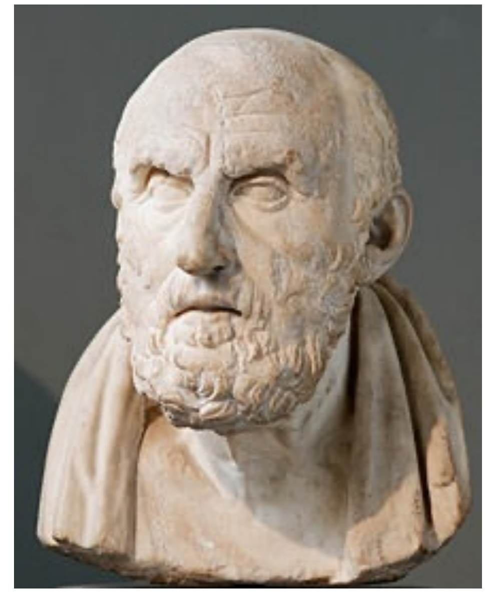
# Propositional and First-Order Logic

Mark Hasegawa-Johnson



March 2023



By Unknown artist - Marie-Lan Nguyen (2011), Public Domain, https://commons.wikimedia.org/w/index.php?curid=1447238

# Outline

- Propositional Logic
- First-Order Logic
- Quantification

# Propositional Logic

- “Propositions” are statements that can be either True or False
  - P=“an iguana is an animal with scales”
  - Q=“an iguana is an animal that breathes air”
  - R=“an iguana is a reptile”
- Propositional logic studies the relationships among propositions.

# Notation

Propositional logic uses a letter to denote each proposition, and introduces the following five symbols:

- $\neg$ (not)
- $\wedge$ (and)
- $\vee$ (or)
- $\Longrightarrow$ (implies)
- $\Leftrightarrow$ (equivalent)

Using these symbols, you can combine several propositions to form one proposition. For example,

# Propositional Logic

- Propositional logic is the study of how given propositions can be combined to prove new propositions.
- For example, consider the proposition $P \wedge Q \implies R$, "if an iguana has scales and breathes air, then it is a reptile." This proposition is only false if P and Q are both true, but R is false:

| $P$ | $Q$ | $R$ | $P \wedge Q \implies R$ |
|---|---|---|---|
| $F$ | $F$ | $F$ | $T$ |
| $F$ | $F$ | $T$ | $T$ |
| $F$ | $T$ | $F$ | $T$ |
| $F$ | $T$ | $T$ | $T$ |
| $T$ | $F$ | $F$ | $T$ |
| $T$ | $F$ | $T$ | $T$ |
| $T$ | $T$ | $F$ | $F$ |
| $T$ | $T$ | $T$ | $T$ |

# Outline

- Propositional Logic
- First-Order Logic
- Quantification

# First Order Logic

- Propositional logic says that propositions can be constructed from other propositions
- First-order logic says propositions can also be constructed by applying predicates to constants

# Predicates, Constants, Variables, Propositions, and Rules

- A **<u>predicate</u>** is like a function, that can be applied to some **<u>variables</u>**.
  - $BreathesAir(x)$ is true if and only if x breathes air.
- A **<u>constant</u>** is a particular object in the real world, which can be the value of the argument of a function:
  - $reptiles$ is a constant
- A **<u>proposition</u>** is a predicate applied to a constant
  - $BreathesAir(reptiles)$ is true if and only if reptiles breathes air.
- A **<u>rule</u>** is an implication or equivalence that's true for all values of its variable
  - $BreathesAir(x) \wedge Scales(x) \Longrightarrow Reptile(x)$: everything that breathes air and has scales is a reptile.

# Theorem Proving

An automatic theorem-prover uses a database of known facts and known rules to prove a theorem. For example, suppose we know that:

- Iguanas have scales: $Scales(iguanas)$
- Iguanas breathe air: $BreathesAir(iguanas)$
- Anything that breathes air and has scales is a reptile: $BreathesAir(x) \wedge Scales(x) \Longrightarrow Reptile(x)$

And suppose we want to prove that:

- Iguanas are reptiles: $Reptile(iguanas)$

# Theorem Proving by Forward-Chaining

- Forward-chaining is the process of applying rules to facts in order to prove more facts.
- For example, let's start by combining these two facts:

$$BreathesAir(iguanas) \wedge Scales(iguanas)$$

- Now let's apply this rule:

$$BreathesAir(x) \wedge Scales(x) \Longrightarrow Reptile(x)$$

- The result: we have proven that:

$$Reptile(iguanas)$$

# Theorem-Proving by Forward-Chaining

Notice that, when we're forward-chaining, each step of the process just expands the set of available facts. If we start with the following database of facts:

$$BreathesAir(iguanas) \wedge Scales(iguanas)$$

… and if we apply the rule $BreathesAir(x) \wedge Scales(x) \Longrightarrow Reptile(x)$, then the database can only get larger. It becomes this:

$$BreathesAir(iguanas) \wedge Scales(iguanas) \wedge Reptile(iguanas)$$

Forward-chaining just keeps going, until the fact we want is part of the database, or until we can't prove any more facts.

# Outline

- Propositional Logic
- First-Order Logic
- Quantification

# Quantification

- It is sometimes useful to express compound propositions that are true for some values of their variables, but not all.
- To do this, we introduce two new symbols, called quantifiers:
- ∃ (there exists)
  - Suppose P is the proposition $\mathrm{P} = \exists x{:}\, F(x)$
  - Then $P = T$ if and only if, for at least one value of the variable $x$, $F(x) = T$
- ∀ (for all)
  - Suppose P is the proposition $\mathrm{P} = \forall x{:}\, F(x)$
  - Then $P = T$ if and only if, for all values of the variable $x$, $F(x) = T$

# Colonel West (Example from the textbook)

| English | First-Order Logic Notation |
|---|---|
| It is a crime for Americans to sell weapons to hostile nations. | $\forall x: \exists y, z:\ American(x) \wedge Weapon(y) \wedge Sells(x, y, z) \wedge Hostile(z) \Longrightarrow Criminal(x)$ |
| Colonel West sold missiles to Ganymede. | $\exists x: Sells(west, x, ganymede) \wedge Missile(x)$ |
| Colonel West is American. | $American(west)$ |
| Ganymede is an enemy of America. | $Enemy(ganymede, america)$ |
| Missiles are weapons. | $\forall x: Missile(x) \Longrightarrow Weapon(x)$ |
| An enemy of America is a hostile nation. | $\forall x: Enemy(x, america) \Longrightarrow Hostile(x)$ |

# Automatic Theorem Proving

| First-Order Logic Notation |
|---|
| $American(x) \wedge Weapon(y) \wedge Sells(x, y, z) \wedge Hostile(z) \Longrightarrow Criminal(x)$ |
| $\exists x, Missile(x) \wedge Sells(west, x, ganymede)$ |
| $American(west)$ |
| $Enemy(ganymede, america)$ |
| $Missile(x) \Longrightarrow Weapon(x)$ |
| $Enemy(x, america) \Longrightarrow Hostile(x)$ |

Can we prove the theorem:

$Criminal(west)$?

# Actions that a Theorem Prover can Take

- **Universal Instantiation**:
  - given the sentence $\forall x, Function(x)$,
  - for any known constant $C$,
  - it is possible to generate the sentence $Function(C)$.
- **Existential Instantiation**:
  - given the proposition $\exists x, Function(x)$,
  - if no known constant $A$ is known to satisfy $Function(A)$, then
  - it is possible to define a new, otherwise unspecified constant $B$, and
  - to generate the sentence $Function(B)$.
- **Generalized Modus Ponens:**
  - Given the sentence $p_1(x_1) \wedge p_2(x_2) \wedge \ldots \wedge p_n(x_n) \Longrightarrow q(x_1, \ldots, x_n)$, and
  - given the sentences $p_1(C_1), \ldots, p_n(C_n)$ for any constants $C_1, \ldots, C_n$,
  - it is possible to generate the sentence $q(C_1, \ldots, C_n)$

# Automatic Theorem Proving Example

- **Existential Instantiation**:
  - Input: $\exists x, Missile(x) \wedge Sells(West, x, Ganymede)$
  - Output: $Missile(M) \wedge Sells(West, M, Ganymede)$
- **Generalized Modus Ponens:**
  - Input: $Missile(M)$ **and** $Missile(x) \Longrightarrow Weapon(x)$
  - Output: $Weapon(M)$
- **Generalized Modus Ponens:**
  - Input: $Enemy(Ganymede, America)$ **and** $Enemy(x, America) \Longrightarrow Hostile(x)$
  - Output: $Hostile(Ganymede)$
- **Generalized Modus Ponens:**
  - $American(x) \wedge Weapon(y) \wedge Sells(x, y, z) \wedge Hostile(z) \Longrightarrow Criminal(x)$
  - Input: $(x)$ **and** $American(West), Weapon(M), Sells(West, M, Ganymede), Hostile(Ganymede)$
  - Output: $Criminal(West)$

# Quiz

- Try the quiz: https://us.prairielearn.com/pl/course_instance/129874/assessment/2334390

# Outline

- Propositional Logic
  - $\neg$ (not) , $\wedge$ (and) , $\vee$ (or), $\Longrightarrow$ (implies), $\Leftrightarrow$ (equivalent)
- First-Order Logic
  - A proposition is a predicate applied to a constant
  - A rule is an implication or equivalence that's true for all values of its variables
- Quantification
  - $\exists x: F(x)$ means that, for at least one value of the variable $x$, $F(x) = T$
  - $\forall x: F(x)$ means that, for all values of the variable $x$, $F(x) = T$