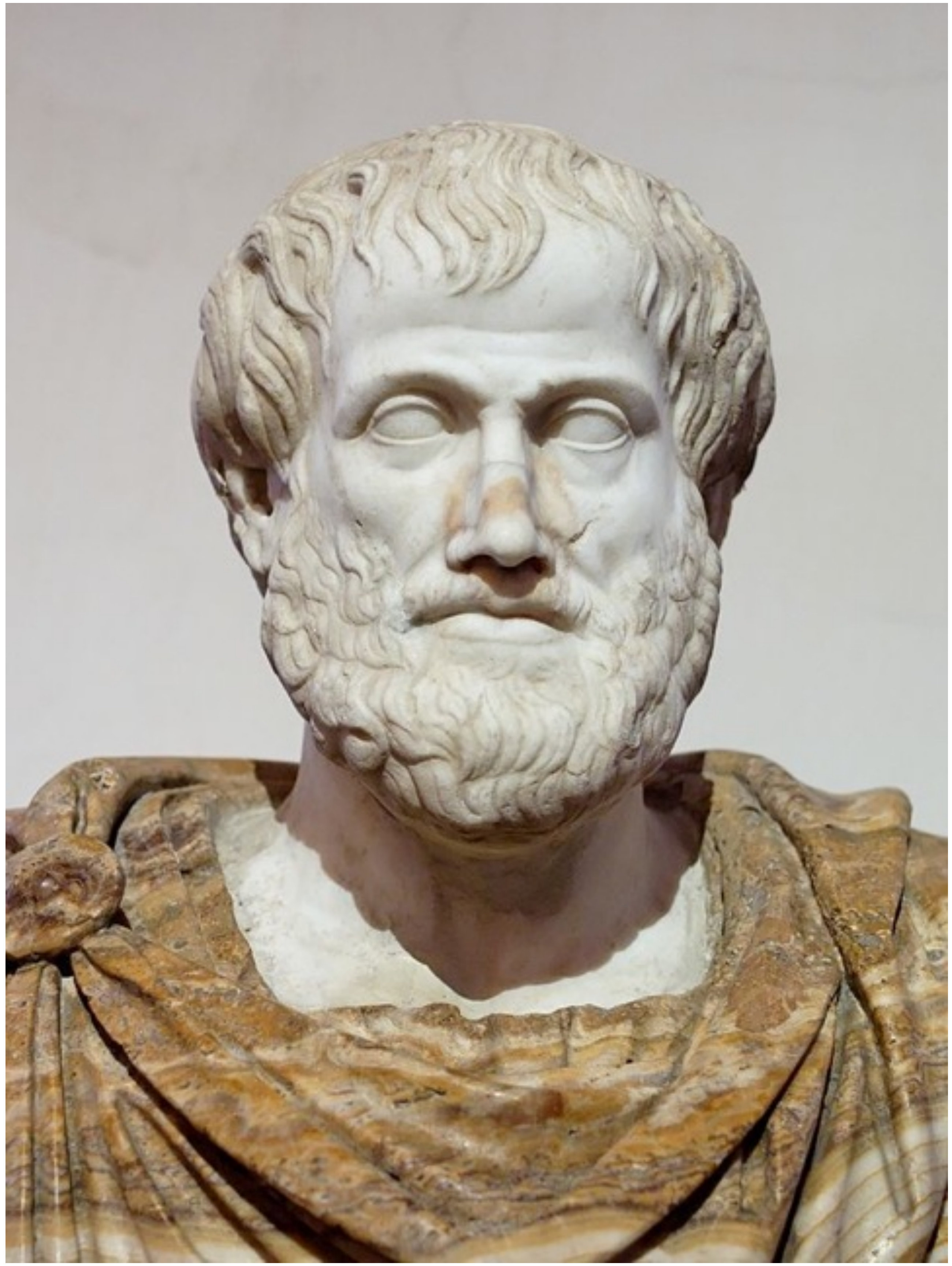
First-Order Logic

George Konidaris
gdk@cs.duke.edu

DUKE COMPUTER SCIENCE

Spring 2016



First-Order Logic

More sophisticated representation language.

World can be described by:

objects

$Color(\cdot)$
functions

$Adjacent(\cdot, \cdot)$
$IsApple(\cdot)$
relations

First-Order Logic

Objects:

- A "thing in the world"
  - Apples
  - Red
  - The Internet
  - Team Edward
  - Reddit
- A *name* that references something.
- Cf. a *noun*.

$$MyApple271$$

$$TheInternet$$

$$CompSci270$$

$$Ennui$$

First-Order Logic

Functions:

- Operator that maps object(s) to single object.
  - $ColorOf(\cdot)$
  - $ObjectNextTo(\cdot)$
  - $SocialSecurityNumber(\cdot)$
  - $DateOfBirth(\cdot)$
    $Spouse(\cdot)$

$$ColorOf(MyApple271) = Red$$

First-Order Logic

Relations (also called *predicates*):

Like a function, but returns *True* or *False* - holds or does not.

- $IsApple(\cdot)$
- $ParentOf(\cdot, \cdot)$
- $BiggerThan(\cdot, \cdot)$
- $HasA(\cdot, \cdot)$

These are like *verbs* or *verb phrases*.

First-Order Logic

We can build up complex sentences using logical connectives, as in propositional logic:

- $Fruit(X) \implies Sweet(X)$
- $Food(X) \implies (Savory(X) \lor Sweet(X))$
- $ParentOf(Bob, Alice) \land ParentOf(Alice, Humphrey)$
- $Fruit(X) \implies Tasty(X) \lor (IsTomato(X) \land \neg Tasty(X))$

Predicates can appear where a propositions appear in propositional logic, but *functions cannot*.

Models for First-Order Logic

Recall from Propositional Logic!

A *model* is a formalization of a "world":

- Set the value of every variable in the KB to *True* or *False*.
- 2^n models possible for n propositions.

Proposition	**Value**
Cold	False
Raining	False
Cloudy	False
Hot	False

Proposition	**Value**
Cold	True
Raining	False
Cloudy	False
Hot	False

...

Proposition	**Value**
Cold	True
Raining	True
Cloudy	True
Hot	True

Models for First-Order Logic

DUKE COMPUTER SCIENCE

The situation is much more complex for FOL.

A model in FOL consists of:

- A set of objects.
- A set of functions + *values for all inputs*.
- A set of predicates + *values for all inputs*.

Models for First-Order Logic

Consider:

Objects

$Orange$

$Apple$

Predicates

$IsRed(\cdot)$

$HasVitaminC(\cdot)$

Functions

$OppositeOf(\cdot)$

Example model:

Predicate	**Argument**	**Value**
$IsRed$	$Orange$	$False$
$IsRed$	$Apple$	$True$
$HasVitaminC$	$Orange$	$True$
$HasVitaminC$	$Apple$	$True$

Function	**Argument**	**Return**
$OppositeOf$	$Orange$	$Apple$
$Opposite$	$Apple$	$Orange$

Knowledge Bases in FOL

A KB is now:

- A set of objects.
- A set of predicates.
- A set of functions.
- A set of sentences using the predicates, functions, and objects, and **asserted to be true**.

Objects

$Orange$

$Apple$

Predicates

$IsRed(\cdot)$

$HasVitaminC(\cdot)$

Functions

$OppositeOf(\cdot)$

$IsRed(Apple)$

$HasVitaminC(Orange)$

Knowledge Bases in FOL

Listing everything is tedious …

- Especially when general relationships hold.

We would like a way to say more general things about the world than explicitly listing truth values for each object.

Quantifiers

New weapon:

- **Quantifiers.**

Quantifiers allow us to make generic statements about properties that hold for the *entire collection of objects* in our KB.

Natural way to say things like:

- All fish have fins.
- All books have pages.
- There is a textbook about AI.

Key idea: **variable + binding rule**.

Existential Quantifiers

There exists object(s) such that a sentence holds.

$$\exists x, IsPresident(x)$$

"there exists"

sentence using variable

temporary variable

Existential Quantifiers

Examples:

- $\exists x, Person(x) \wedge Name(x, George)$
- $\exists x, Car(x) \wedge ParkedIn(x, E23)$
- $\exists x, Course(x) \wedge Prerequisite(x, CS270)$

Universal Quantifiers

A sentence holds for all object(s).

$$\forall x, HasStudentNumber(x) \implies Person(x)$$

"for every"

sentence
using variable

temporary variable

Universal Quantifiers

Examples

- $\forall x, Fruit(x) \implies Tasty(x)$

- $\forall x, Bird(x) \implies Feathered(x)$

$$\forall x, Book(x) \rightarrow HasAuthor(x)$$

Quantifiers

Difference in strength:

- Universal quantifier is **very strong**.
  - So use **weak sentence**.

$$\forall x, Bird(x) \implies Feathered(x)$$

- Existential quantifier is **very weak**.
  - So use **strong sentence**.

$$\exists x, Car(x) \land ParkedIn(x, E23)$$

Compound Quantifiers

$$\forall x, \exists y, Person(x) \implies Name(x, y)$$

"every person has a name"

Common Pitfalls

$$\forall x, Bird(x) \land Feathered(x)$$

Common Pitfalls

$$\exists x, Car(x) \implies ParkedIn(x, E23)$$

Examples

...

Inference in First-Order Logic

Ground term, or **literal** - an actual object:

$$MyApple12$$

vs. a **variable**:

$$x$$

If you have only ground terms, you can convert to a propositional representation and proceed from there.

$$IsTasty(Apple) : IsTastyApple$$

Instantiation

DUKE COMPUTER SCIENCE

Getting rid of variables: **instantiate** a variable to a literal.

Universally quantified:

$$\forall x, Fruit(x) \implies Tasty(x)$$

$$Fruit(Apple) \implies Tasty(Apple)$$

$$Fruit(Orange) \implies Tasty(Orange)$$

$$Fruit(MyCar) \implies Tasty(MyCar)$$

$$Fruit(TheSky) \implies Tasty(TheSky)$$

For every object in the KB, just write out the rule with the variables substituted.

Instantiation

Existentially quantified:

- Invent a new name **(Skolem constant)**

$$\exists x, Car(x) \land ParkedIn(x, E23)$$

$$Car(C) \land ParkedIn(C, E23)$$

- Name cannot be one you've already used.
- Rule can then be discarded.

PROLOG

PROgramming in LOGic (Colmerauer, 1970s)

- General-purpose AI programming language
- Based on First-Order Logic
- Declarative

- Use centered in Europe and Japan
- Fifth-Generation Computer Project

- Some parts of Watson (pattern matching over NLP)
- Often used as component of a system.