

First Order Logic

Dr Pooja Agarwal

Department of Computer Science & Engineering

Outline

- Quantifiers
 - Universal quantification (∀)
 - Existential quantification (∃)
 - Nested quantifiers
 - Connections between ∀ and ∃
- Equality



Universal quantification (∀)

PES UNIVERSITY ONLINE

∀: Universal Quantifiers

 \forall x : for all x

"All kings are persons," is written in first-order logic as

 $\forall x \text{ King}(x) \Rightarrow \text{Person}(x)$

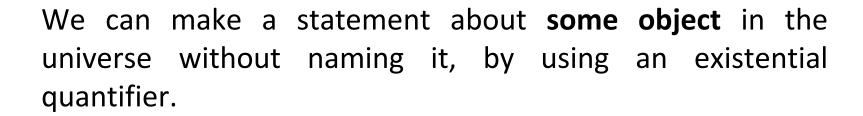
symbol x is called a variable.

 $\forall x P$

Existential quantification (∃)

∃: Existential Quantifiers

∃ x : there exist x



Example
 King John has a crown on his head

 $\exists x Crown(x) \land OnHead(x, John)$





Nested quantifiers

PES UNIVERSITY ONLINE

- To express more complex sentences using multiple quantifiers.
- The simplest case is where the quantifiers are of the same type.
- Example:"Brothers are siblings" can be written as
- $\forall x \forall y \text{ Brother } (x, y) \Rightarrow \text{Sibling}(x, y)$

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

Nested quantifiers - Example



1. "Everybody loves somebody"

$$\forall x \exists y Loves(x, y)$$

$$\forall$$
 x (\exists y Loves(x, y))

2. "There is someone who is loved by everyone," we write

$$\exists y \forall x Loves(x, y)$$

$$\exists y (\forall x Loves(x, y))$$

Nested quantifiers



When two quantifiers are used with the same variable name.

Example:

 \forall x (Crown(x) \lor (\exists x Brother (Richard, x)))

x in Brother(Richard, x) is existentially quantified.

 \forall x (Crown(x) \lor (\exists z Brother (Richard, z)))

Connections between ∀ and ∃



The two quantifiers are actually intimately connected with each other, through negation.

Declaring

"Everyone dislikes Mango is **the same as** asserting there does not exist someone who likes Mango, and vice versa"

 $\forall x \neg Likes(x, Mango)$ is equivalent to $\neg \exists x Likes(x, Mango)$

Connections between ∀ and ∃

PES UNIVERSITY ONLINE

We can go one step further:

"Everyone likes ice cream"

means that there is no one who does not like ice cream:

 \forall x Likes(x, IceCream) is equivalent to $\neg\exists$ x \neg Likes(x, IceCream)

Connections between ∀ and ∃



The De Morgan rules for quantified and unquantified sentences are as follows:

$$\forall x \neg P \equiv \neg \exists x P$$
 $\neg \forall x P \equiv \exists x \neg P$
 $\forall x P \equiv \neg \exists x \neg P$
 $\forall x P \equiv \neg \exists x \neg P$
 $\exists x P \equiv \neg \forall x \neg P$
 $\neg (P \lor Q) \equiv \neg P \land \neg Q$
 $\neg (P \land Q) \equiv \neg P \lor \neg Q$
 $P \land Q \equiv \neg (\neg P \lor \neg Q)$

 $PVQ \equiv \neg(\neg P \land \neg Q)$

Equality

PES UNIVERSITY ONLINE

- First-order logic includes one more way to make atomic sentences, other than using a predicate and terms.
- We can use the **equality symbol** to signify the two terms refer to the same object.

Example:

Father (John)=Henry

To say that Richard has at least two brothers, we would write

 $\exists x, y \text{ Brother } (x, \text{Richard }) \land \text{Brother } (y, \text{Richard }) \land \neg (x=y)$

Symbols and interpretations

Syntax of FOL

```
Sentence \rightarrow AtomicSentence \mid ComplexSentence
          AtomicSentence \rightarrow Predicate \mid Predicate(Term,...) \mid Term = Term
         ComplexSentence \rightarrow (Sentence) \mid [Sentence]
                                       \neg Sentence
                                       Sentence \wedge Sentence
                                       Sentence \lor Sentence
                                       Sentence \Rightarrow Sentence
                                       Sentence \Leftrightarrow Sentence
                                       Quantifier Variable, ... Sentence
                        Term \rightarrow Function(Term, ...)
                                       Constant
                                        Variable
                  Quantifier \rightarrow \forall \mid \exists
                   Constant \rightarrow A \mid X_1 \mid John \mid \cdots
                    Variable \rightarrow a \mid x \mid s \mid \cdots
                   Predicate \rightarrow True \mid False \mid After \mid Loves \mid Raining \mid \cdots
                   Function \rightarrow Mother \mid LeftLeg \mid \cdots
OPERATOR PRECEDENCE : \neg, =, \land, \lor, \Rightarrow, \Leftrightarrow
```





THANK YOU

Pooja Agarwal
Department of Computer Science & Engineering
poojaagarwal@pes.edu