# Lecture 26

## Chapter 3: first-order logic FOL

Goal: To understand what is meant by the term 'universally valid formulae' in order to understand Turing's work on the Entscheidungsproblem.

PL has atoms - we now split those atoms.

Names These pick out specific individuals like:

2, $\pi$, Darth Vader, Tigger, ...

We shall use the term constants to mean the same as names

For us, names will be $a, b, c, \ldots$ or $a_1, a_2, a_3, \ldots$

Variables / These locate the places into which names can be slotted. For us, variables will be $x, y, z, \ldots, x_1, x_2, x_3, \ldots$

Predicates

1-place predicates $P(x)$, $R(x)$, ...

2-place predicates $P(x, y)$, $R(x, y)$ ...

More generally, n-place predicates or n-ary predicates

We say that a predicate has <u>arity n</u> if it has exactly n slots — marked by variables — into which names can be slotted.

<u>Examples</u>

(1) 'x is green' is a 1-place predicate

Replace x by a name 'grass is green'
and we get a statement (that will be T or F depending on the name).

(2) 'x is the father of y' is a 2-place predicate.

If we replace x by 'Darth Vader' and y by 'Luke Skywalker' we get the statement: 'Darth Vader is the father of Luke Skywalker' which is — spoiler alert — T.

(3) 'x is between y and z' is a 3-place predicate

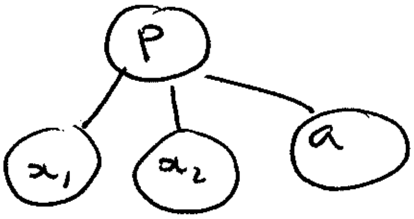
'Theresa is between a rock and a hard place'.

> For simplicity, our predicates will be 1-place or 2-place

An <u>atomic formula</u> (these replace atoms in PL) is a predicate whose slots are filled with either names or variables.

<u>Example</u> $P(x_1, x_2, a)$ is an atomic formula

with parse tree




---

## What do predicates do?

$\boxed{\text{1-place predicates describe sets.}}$

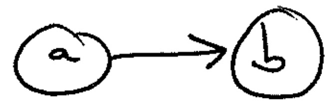
If $P(x)$ is a 1-place predicate then

$$\mathbb{P} = \{a : P(a)\}$$

is the set of all a s.t. $P(a)$ is true.

## 2-place predicates describe binary relations

An ordered pair $(a, b)$ ⇝ picture



first component (of $a$), second component (of $b$)

← the set of all ordered pairs $(a,b)$ s.t. $P(a,b)$ is T.

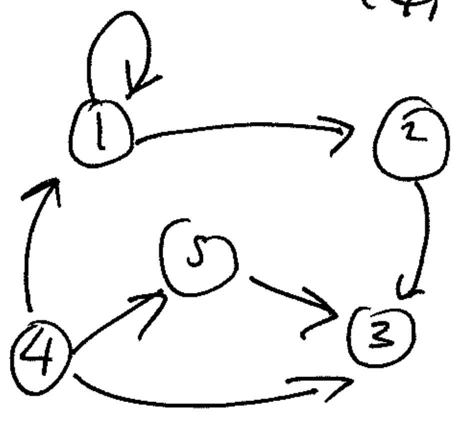
$$\Pi = \{ (a,b) : P(a,b) \}$$

is a set of ordered pairs. Sets of ordered pairs are called **binary relations**

Binary relations can be represented by means of **directed graphs**

**Example**

$$\Pi = \{ (1,1), (1,2), (2,3), (4,1), (4,3), (4,5), (5,3) \}$$



For convenience only – subsets denoted by upper case Latin letters, binary relations by lower case Greek letters.

Binary relations are v. common in maths

Examples

1. $x, y \in \mathbb{N}$, $x \mid y$ means $x = yz$ for some $z \in \mathbb{N}$.

2. $\leq, <, \geq, >$ on sets of numbers.

3. $\subseteq$

4. $\in$.

5. $\equiv$

# Structures

A structure is a non-empty set, $D$, called a domain equipped with one or more subsets, binary relations etc.

## Example

(1) $(\mathbb{N}, \leq, 1)$ is a structure.

(2) $(D, P)$ where $D$ is the set of all people and $P$ is the binary relation 'is the father of'.

$\boxed{\text{FOL is a language that will enable us to talk about structures}}$

# Quantification

$(\forall x)$ is read 'for all $x$' (like $\wedge$)

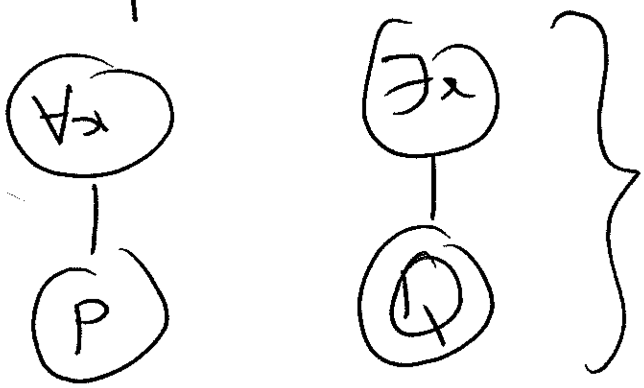
$(\exists x)$ is read 'there exists a $x$'

or 'there exists at least one $x$' (like $\vee$)

Also, $(\forall x), (\forall y), (\forall z), \ldots$

$(\exists x), (\exists y), (\exists z), \ldots$

Regard quantifiers as unary connectives




parse trees.

$(\forall x)P \qquad (\exists x)Q$