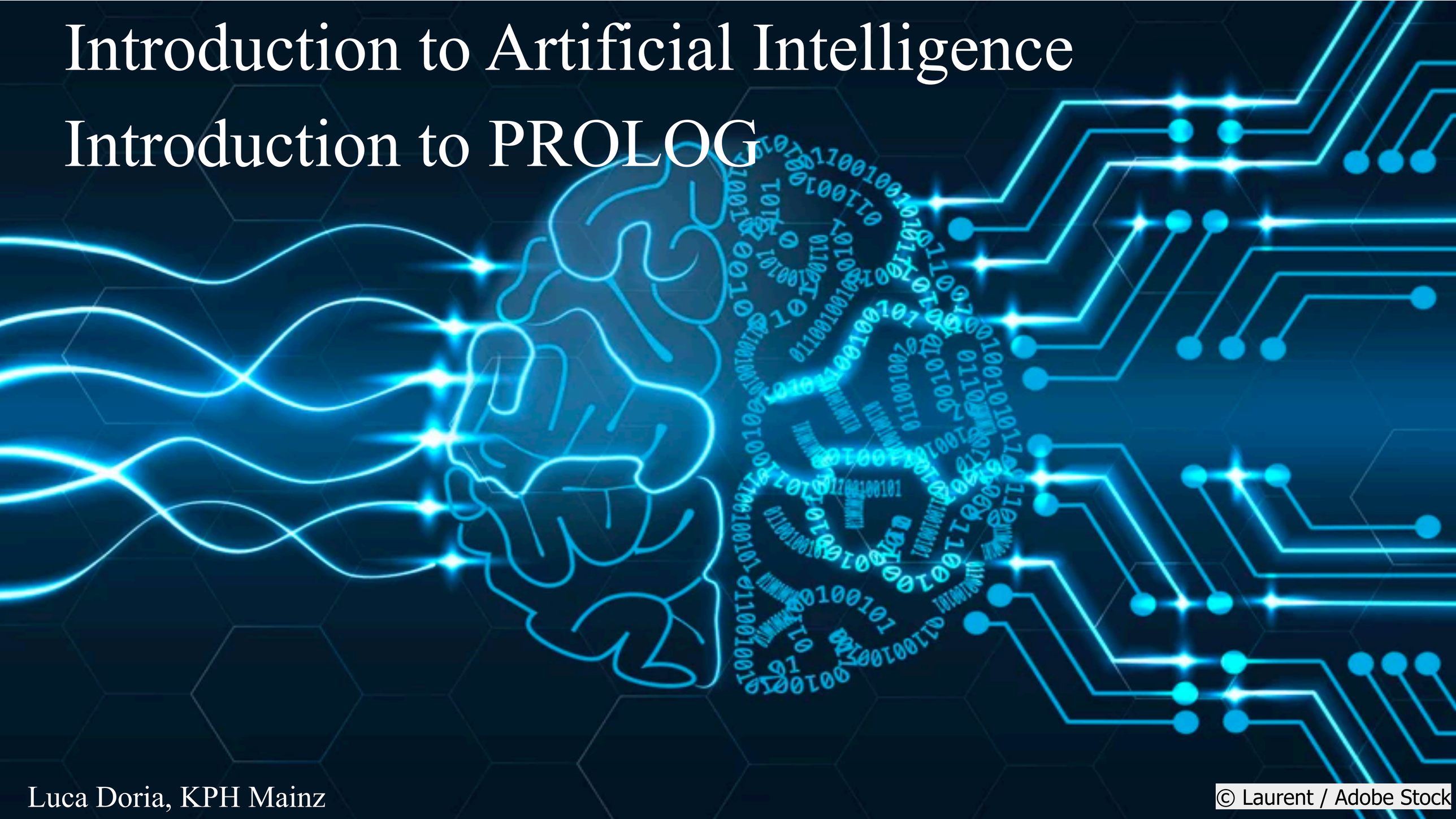
Introduction to Artificial Intelligence

Introduction to PROLOG



Luca Doria, KPH Mainz

A new kind of agent: logical agents

PROLOG

- **Developed in 1972** by **Alain Colmerauer** and **Philippe Roussel** in Marseille, France.
- Based on first-order predicate logic and Horn clauses.
- Name stands for PROgramming in LOGic.
- One of the oldest logic programming languages.
- Became widely used in the 1980s, especially in AI research and expert systems.
- Based on declarative programming: you describe *what* you want, not *how* to do it.
- Programs are written as **facts** and **rules**.
- Uses **logical inference** and **backtracking** to resolve queries.
- A typical Prolog program is a **database of knowledge** + **queries**.

Basic Syntax

Facts: likes(john, pizza).

Note the “period” at the end of a statement

Rules: friend(X, Y) **:-** likes(X, Y), likes(Y, X).

Rule declaration operator

Queries: ?- friend(jack, Who).

A PROLOG implementation (there are many)

https://www.swi-prolog.org/

Call the prompt: swipl
Exit: CTRL-D
Input a program as txt file (3 alternatives):

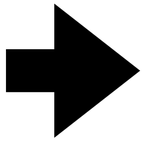
1) ['my_program.txt'].
2) From the shell prompt: swipl -s filename.pl
3) consult('my_program.txt').

Example Program 1

```
% Facts
parent(john, mary).
parent(john, david).
parent(susan, mary).
parent(susan, david).
parent(david, emily).
parent(lisa, emily).

% Rules
father(X, Y) :- parent(X, Y), male(X).
mother(X, Y) :- parent(X, Y), female(X).
grandparent(X, Y) :- parent(X, Z), parent(Z, Y).

% Gender
male(john).
male(david).
female(susan).
female(mary).
female(emily).
female(lisa).
```



ASK:

Who is the grandparent of Emily?

```
grandparent(X, emily).
```

Is John Mary's father?

```
?- father(john, mary).
```

Who are Mary's parents?

```
?- parent(X, mary).
```

Example Program 2: Peano Arithmetic

```
% Definition of natural numbers
nat(0).              %predicate that defines what it means to be a natural number
nat(s(X)) :- nat(X).

% Addition in Peano arithmetic
add(0, Y, Y).
add(s(X), Y, s(Z)) :- add(X, Y, Z).

% Peano representation:
% 1 = s(0), 2 = s(s(0)), 3 = s(s(s(0)))

————————————————————————————

% 1 + 2 = 3
add(s(0), s(s(0)), R).

%2 + X = 3
add(s(s(0)), X, s(s(s(0)))).
```

Example Program 2: Geometry

```
% point(Name, X, Y)
point(a, 1, 1).
point(b, 2, 2).
point(c, 3, 3).
point(d, 1, 2).
point(e, 2, 3).

% collinear(P1, P2, P3) is true if points P1, P2, P3 are collinear.
collinear(P1, P2, P3) :-
    point(P1, X1, Y1),
    point(P2, X2, Y2),
    point(P3, X3, Y3),
    % Determinant
    0 is X1*(Y2 - Y3) + X2*(Y3 - Y1) + X3*(Y1 - Y2).
```

Ask e.g. : collinear(a,b,c).

The **is** operator in Prolog **evaluates the arithmetic expression on the right side** and **unifies** it with the value on the left.

Three points form a valid triangle if they are not collinear and:

$$\text{Area} = \frac{1}{2} \times \begin{vmatrix} x_1 & y_1 & 1 \\ x_2 & y_2 & 1 \\ x_3 & y_3 & 1 \end{vmatrix} = 0$$

so the determinant cannot be ≠0

Example Program 3: An Investigation

Imagine this following case for Sherlock Holmes:

- There are three suspects: Watson, Moriarty, and Lestrade.
- Only one of them committed the crime and Sherlock must deduce who did it.
- He has the following clues:
    - The culprit is either Moriarty or a person owning a dog.
    - Watson owns a cat.
    - Lestrade owns a dog.
    - Moriarty was out if town at the time of the crime.
    - The criminal does not own a cat.

Too difficult: the investigator needs PROLOG…

Example Program 3: An Investigation

```
% suspect(Name)
suspect(watson).
suspect(moriarty).
suspect(lestrade).

% owns_pet(Person, Pet)
owns_pet(watson, cat).
owns_pet(lestrade, dog).
owns_pet(moriarty, none).

% alibi(Person) - Person has an alibi
alibi(moriarty).

% culprit(X) - X committed the crime
culprit(X) :- suspect(X), (X = moriarty; owns_pet(X, dog)), \+ alibi(X), \+ owns_pet(X, cat).
```

X must be a suspect

Either X is Moriarty **or** it owns a pet

X does not have an alibi

X does not own a cat

This operator means: **"it is not provable that…".** Basically a **negation**.

Ask e.g. : culprit(X).

```
% Clue 1: Culprit is Moriarty or owns a dog

% Clue 4: Moriarty has alibi, so he can't be culprit

% Clue 5: Culprit does not own a cat
```