# CS339: Abstractions and Paradigms for Programming

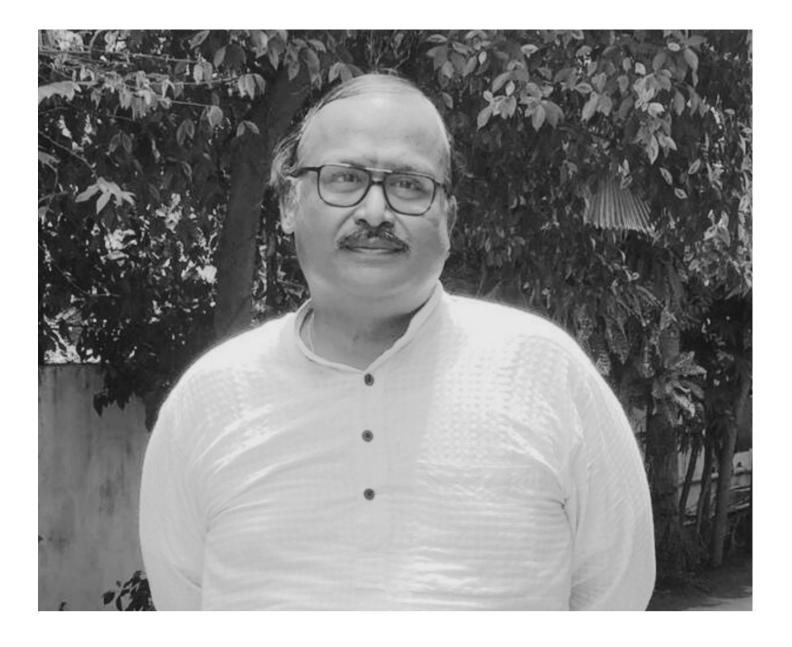
Logic Paradigm (Cont.)

#### Manas Thakur

CSE, IIT Bombay



Autumn 2024



Our beloved Vinay Wardhan sir has passed away.

He made his mark on thousands of students as a career coach and as an inspiration. Please remember him in your prayers.

Thank you for touching our lives, sir. You'll be missed.

। ॐ शांति ॥

Vinay Wardhan, Director, Career Path Pvt Ltd





Manas Thakur

You Pour Love, You Get Love.

The World is Circular.

#### From the homework

- ➤ Define a sibling relationship for our family tree:
  - ➤ What if you did this?
    - $\succ$  sibling(X, Y) :- parent(W, X), parent (W, Y).
  - ➤ The right way:
    - $\succ$  sibling(X, Y) :- parent(W, X), parent (W, Y), X \= Y.

> Prolog supports relational operators.



# How does the Prolog engine work?

- ➤ Program = logic + control.
  - ➤ Logic is specified by user; control is managed by runtime.

- ➤ Given a query:
  - ➤ Consult the facts and rules in top-down order.
  - ➤ Try to instantiate variables in the RHS of rules.
  - ➤ Report instantiated values that satisfy the predicates resultant from the query.



#### Resolution and Unification

- ➤ **Resolution.** If h is the head of a Horn clause and it matches with one of the terms of another Horn clause, then that term can be replaced by h.
- ➤ **Unification.** A pattern-matching process that determines what particular instantiations can be made to variables while making a series of simultaneous resolutions.
  - ➤ Which resolutions are simultaneous?
    - ➤ Those that satisfy the given set of predicates.
- > Example: ?- parent(brandon, bran).



### Do they unify?

- **>** a & a ✓
- > a & b ×
- > a & A ✓ A = a
- $\rightarrow$  f(x,y) & A  $\checkmark$  A = f(x,y)
- > f(a,b) & g(a,b) X
- $\rightarrow$  f(X,b,c) & f(a,X,c)  $\times$

# Searching and Backtracking

When the path ahead is not nice, say even mice, that backtracking is wise.

- ➤ Basic idea of logic paradigm:
  - ➤ Search through the solution space while trying to unify variables with values, till you get a solution.
  - ➤ If no further resolution can be done, then backtrack and try a different instantiation.
- ➤ Example: ?- parent(brandon, bran).
- ➤ Observe: Multiple solutions are possible.
  - > Example: ?- grandparent(rickard, Whoall).



#### Lists

- ➤ List: the basic data structure in Prolog.
  - ➤ [Head | Tail]

- > [lists, wont, leave, you, in, app]
  - ➤ Head: lists
  - ➤ Tail: [wont, leave, you, in, app]
  - ➤ What about [H1, H2 | T]?



### Operations on lists

➤ Concatenate two lists:

```
append([], X, X).
append([Head | Tail], Y, [Head | Z]) :- append(Tail, Y, Z).
```

- ➤ Prefix: prefix(X, Z) :- append(X, Y, Z).
- > Suffix: suffix(Y, Z) :- append(X, Y, Z).
- ➤ Membership:

```
member(X, [X | _]).
member(X, [_ | Y]):- member(X, Y).
```





#### But we had learnt numbers before lists in Scheme!

➤ Compute the factorial of a number in Prolog:

- ➤ The infix operator *is* forces the instantiation of a variable by performing arithmetic operations.
- $\rightarrow$  Is N > 0 important?
  - ➤ Try removing it!



# Let's see another example

Find the max of two numbers.

```
\max(X, Y, Y) :- X =< Y.

\max(X, Y, X) :- X > Y.
```

- ➤ **Observe:** The interpreter waits to try more solutions, even though the cases are mutually exclusive.
  - ➤ What we want: Abort searching if the first case is true.
- > The same reasoning for the wait during grandparent (rickard, bran).

