Prolog: Exercising Control

Course: CS40002
Instructor: Dr. Pallab Dasgupta

Department of Computer Science & Engineering
Indian Institute of Technology Kharagpur

Eight Queens Problem

```
solution( Queens ) :-
   permutation( [1,2,3,4,5,6,7,8], Queens ),
   safe( Queens ).

permutation( [ ], [ ] ).
permutation( [Head | Tail], Permlist ) :-
                permutation( Tail, PermTail ),
                del( Head, Permlist, PermTail ).
```

Eight Queens Problem (Contd.)

```
safe( [ ] ).
safe( [Queen | Others] ) :-
    safe( Others ), noattack( Queen, Others, 1 ).

noattack( _, [ ], _ ).
noattack( Y, [ Y1 | Ylist ], Xdist ) :-
    Y1 – Y =\= Xdist, Y – Y1 =\= Xdist,
    Dist1 is Xdist + 1, noattacks( Y, Ylist, Dist1 ).
```

Cuts – for controlling backtracking

```
C :- P, Q, R, !, S, T, U.
C :- V.
A :- B, C, D
?- A
```

- Backtracking within the goal list P, Q, R
- As soon as the cut is reached:
  - All alternatives of P, Q, R are suppressed.
  - The clause C:- V will also be discarded
  - Backtracking possible within S, T, U.
  - No effect within A :- B, C, D, that is, backtracking within B, C, D remains active.

Examples

- Finding the maximum of two numbers

  If X >= Y then Max = X, otherwise Max = Y.

  ```
  max( X, Y, X ) :- X >= Y, !.
  max( X, Y, Y ).
  ```

- Adding an element into a list without duplication

  ```
  add( X, L, L ) :- member( X, L ), !.
  add( X, L, [X | L] ).
  ```

Negation as failure

- Frodo likes all jewellery except rings

```
likes( frodo, X ) :-  ring( X ), !, fail.
likes( frodo, X ) :-  jewellery( X ).
```

- The “different” predicate:

```
different( X, X ) :- !, fail.
different( X, Y ).
```

Quicksort

```
quicksort( [ ], [ ] ).

quicksort( [ X | Tail ], sorted ) :-
    split( X, Tail, Small, Big ),
    quicksort( Small, SortedSmall ),
    quicksort( Big, SortedBig ),
    conc( SortedSmall, [ X | SortedBig ], Sorted ).
```

Quicksort

```
split( X, [ ], [ ], [ ] ).

split( X, [ Y | Tail ], [ Y | Small ], Big ) :-
        gt( X, Y ), !, split( X, Tail, Small, Big ).

split( X, [ Y | Tail ], Small, [ Y | Big ] ) :-
                        split( X, Tail, Small, Big ).
```