CSEN403: Concepts of Programming Languages
Logic Programming II

Prof Dr. Slim Abdennadher
Dr. Nada Sharaf

Spring Semester 2021

Term Matching in Prolog (Unification)

- Aims at making two terms the same thing
- Variables might be bound to values
- In orer for two atoms to match, **they have to be the same atom**
  - `a=a`
  - `a\=b`
- A variable can match any Prolog term
  - `X= a.`
  - `X= timing(tuesday, 3).`
- Two complex terms can match iff
  1. They have the same functor name
  2. They have the same number of arguments
  3. Every two corresponding arguments can match
  4. `f(X) = f(a).`

How does Prolog work?

- Logic Programming is non-deterministic
- Prolog is deterministic.

Deterministic vs. Non-Deterministic

Non-Deterministic
Arbitrary choose fact, subgoal and rule

Deterministic
Chooses facts, subgoals and rules according to a specific order

Search Technique

```
male(prince_charles).

parent(queen_elizabeth_ii, prince_charles).
parent(prince_philip, prince_charles).
parent(queen_elizabeth_ii, princess_anne).
parent(prince_philip, princess_anne).
parent(queen_elizabeth_ii, prince_andrew).
```

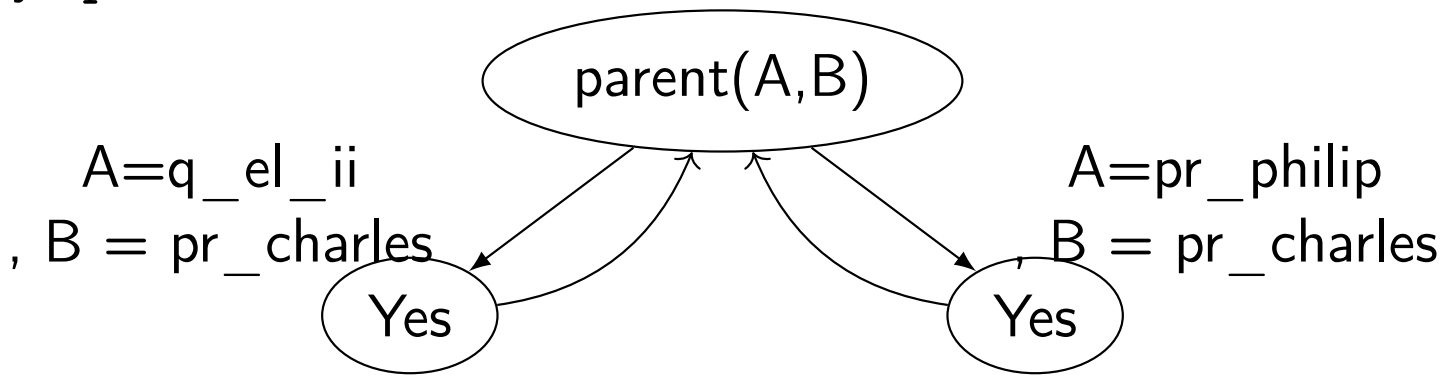
```
?- parent(X,Y).
```

1. Search trough facts and (head) of rules.
2. Do the necessary bindings.
3. File is searched in a top-bottom approach.
4. Subgoals are explored in left-to-right approach

Example I

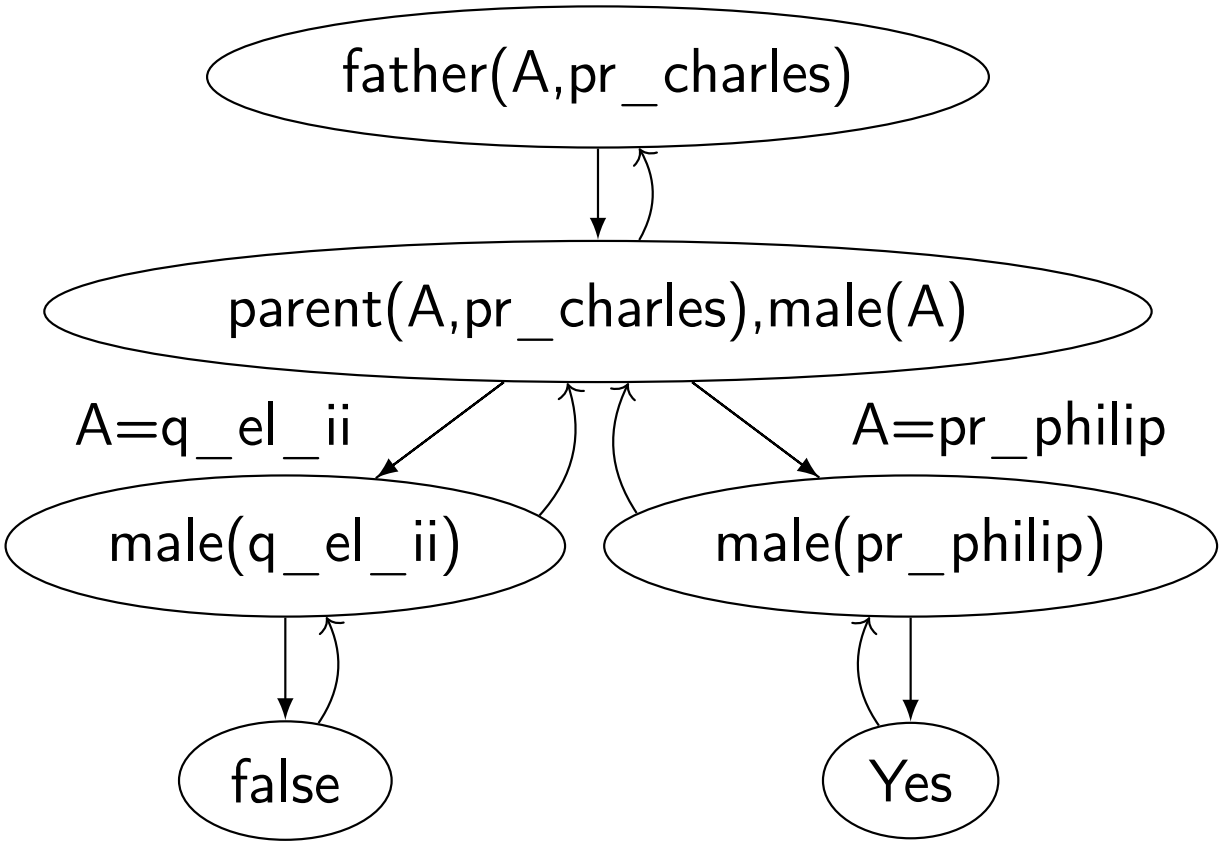
```
male(prince_philip).

parent(q_el_ii, pr_charles).
parent(pr_philip, pr_charles).
```

Query: `parent(A,B).`



Example II

```
male(prince_philip).
parent(q_el_ii, pr_charles). parent(pr_philip, pr_charles).
father(X,Y):- parent(X,Y), male(X).
```



Example 3

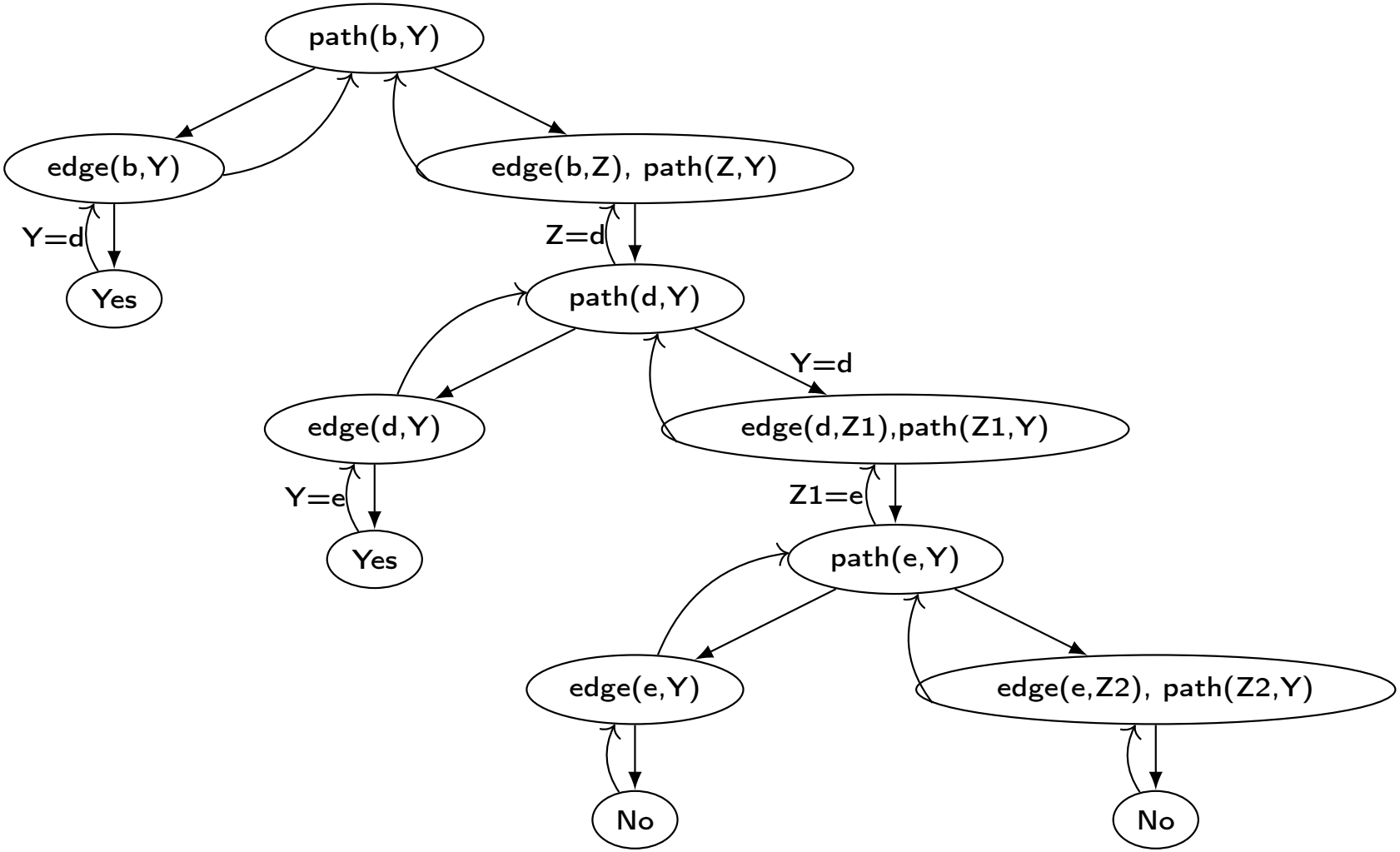
```
edge(a,b).
edge(a,c).
edge(b,d).
edge(c,d).
edge(d,e).
```

When is `path(X,Y)` true, write the rules

```
path(X,Y):- edge(X,Y).
path(X,Y):- edge(X,Z), path (Z,Y).
```

What is the result of the query `path(b,Y)` (Use slido code W586)

path(b,Y)


path(b,Y)
edge(b,Y)
edge(b,Z), path(Z,Y)
Y=d
Yes
Z=d
path(d,Y)
Y=d
edge(d,Y)
edge(d,Z1),path(Z1,Y)
Y=e
Yes
Z1=e
path(e,Y)
edge(e,Y)
edge(e,Z2), path(Z2,Y)
No
No

Back to Family

Example: X is an ancestor of Y if

- X is a parent of Y or
- X is a parent of Z and Z is a ancestor of Y

```
ancestor(X,Y):- parent(X,Y).
ancestor(X,Y):-
                parent(X,Z),
                ancestor(Z,Y).
```

Arithmetic in Prolog

- By default, Prolog will never evaluate an expression.
- `?-X= 2 + 3`
- A special operator has to be used for evaluating e.g. `is`
- `?- X is 2 + 3.`
- `is` evaluates the rihgt hand side only
- `2 + 3 is X.`
- Is this correct? `p(X,Y):- p(X-1,Y).`

Arithmetic Operators

- X + Y,
- X - Y,
- X * Y,
- X / Y (float division),
- X // Y (integer division),
- X mod Y (remainder)

Logical Operators

- X < Y, X =< Y, X >= Y, X > Y
- X = Y: succeeds if X matches with Y
- X \= Y: if X = Y would fail, this succeeds
- X == Y: compares only, does not match
- X \== Y: negation of X == Y
- X =:= Y: evaluates both sides and compares them numerically
- X =\= Y: similar to =:= but tests for inequality

Factorial

```
fact(0,1).
fact(X,Y):-
        X>0,
        X1 is X-1,
        fact(X1,Y1),
        Y is X*Y1.
```

Fibonacci

As a Mathematical Function:

$$f(x) = \begin{cases} 1 & : \quad \text{if } x = 0 \\ 1 & : \quad \text{if } x = 1 \\ f(x-1) + f(x-2) & : \quad \text{if } x > 1 \end{cases}$$

As a prolog predicate ???

```
fib(0,1).
fib(1,1).
fib(X,Y):-
            X>1,
            X1 is X-1,
            X2 is X-2,
            fib(X1,Y1),
            fib(X2,Y2),
            Y is Y1 + Y2.
```

Thank you