University of Applied Sciences and Arts Northwestern Switzerland
School of Business

Rule-Based Systems: Logic Programming

Reasoning Example: first try

```
parent(peter, paul).                                    (F1)
parent(paul, mary).                                     (F2)

ancestor(X, Y) :- ancestor(X, Z), parent(Z, Y).         (R1)
ancestor(A, B) :- parent(A, B).                         (R2)

?- ancestor(peter, paul)
?- ancestor(peter, mary)
?- ancestor(peter, carl)
```

Reasoning Example: infinite loop with Q1

```
?- ancestor(peter, paul)

        L = {ancestor(peter, paul)}

R1:     L = {ancestor(peter, Z1), parent(Z1, paul)}

R1:           L = {ancestor(peter, Z2), parent(Z2, Z1),
                   parent(Z1, paul)}

R1:                 L = {ancestor(peter, Z3),
                         parent(Z3, Z2),
                         parent(Z2, Z1),
                         parent(Z1, paul)}
```

Infinite loop;
not expected answer („true“)!

Reasoning Example: next try

```
parent(peter, paul).                                    (F1)
parent(paul, mary).                                     (F2)

ancestor(A, B) :- parent(A, B).                         (R1)
ancestor(X, Z) :- ancestor(X, Y), parent(Y, Z).         (R2)

?- ancestor(peter, paul)
?- ancestor(peter, mary)
?- ancestor(peter, carl)
```

Reasoning Example (next try): Q1 works

```
?- ancestor(peter, paul)

        L = {ancestor(peter, paul)}

R1:     L = {parent(peter, paul)}

F1:           L = {}
```

Reasoning Example (next try): Q2 works too

```
?- ancestor(peter, mary)
        L = {ancestor(peter, mary)}
R1:     L = {parent(peter, mary)} FAIL
R2:     L = {ancestor(peter, Z1), parent(Z1, mary)}
R1:           L = {parent(peter, Z1), parent(Z1, mary)}
F1{Z1/paul}:        L = {parent(paul, mary)}
F2:                       L = {}
```

Reasoning Example (next try): Q3 ends up in infinite loop

```
?- ancestor(peter, carl)
        L = {ancestor(peter, carl)}
R1:     L = {parent(peter, carl)} FAIL
R2:     L = {ancestor(peter, Z1), parent(Z1, carl)}
R1:             L = {parent(peter, Z1), parent(Z1, carl)}
F1{Z1/paul}:            L = {parent(paul, carl)} FAIL
R2:             L = {ancestor(peter, Z2), parent(Z2, Z1),
                     parent(Z1, carl)}
R1:                     L = {parent(peter, Z2), parent(Z2, Z1),
                          parent(Z1, carl)}
F1{Z2/paul}:                    L = {parent(paul, Z1),
                                 parent(Z1, carl)} FAIL
R2:                     L = {ancestor(peter, Z3), parent(Z3, Z2),
                     parent(Z2, Z1), parent(Z1, carl)}
```

Infinite Loop;
not expected answer („false“)!

Reasoning Example (final try)

```
parent(peter, paul).                                    (F1)
parent(paul, mary).                                     (F2)

ancestor(A, B) :- parent(A, B).                         (R1)
ancestor(X, Y) :- parent(X, Z), ancestor(Z, Y).         (R2)

?- ancestor(peter, paul)
?- ancestor(peter, mary)
?- ancestor(peter, carl)
```

Reasoning Example (final try): Q1 works

```
?- ancestor(peter, paul)

        L = {ancestor(peter, paul)}

R1:     L = {parent(peter, paul)}

F1:           L = {}
```

Reasoning Example (final try): Q2 works too

```
?- ancestor(peter, mary)

        L = {ancestor(peter, mary)}

R1:     L = {parent(peter, mary)} FAIL

R2:     L = {parent(peter,Z1), ancestor(Z1, mary)}

F1{Z1/paul}:  L = {ancestor(paul, mary)}

R1:                 L = {parent(paul, mary)}

F2:                         L = {}
```

Reasoning Example (3/3): Q3 also works!!

```
?- ancestor(peter, carl)
        L = {ancestor(peter, carl)}
R1:     L = {parent(peter, carl)} FAIL
R2:     L = {parent(peter, Z1), ancestor(Z1, carl)}
F1{Z1/paul}:     L = {ancestor(paul, carl)}
R1:                  L = {parent(paul, carl)} FAIL
R2:                  L = {parent(paul,Z2), ancestor(Z2, carl)}
F2{Z2/mary}:                 L = {ancestor(mary, carl)}
R1:                                L = {parent(mary, carl)} FAIL
R2:                                L = {parent(mary,Z3),
                                        ancestor(Z3, carl)} FAIL

FAIL
```