What if P has rules?

The program (*P*):

```
father(fred,pebbles).
father(bamm-bamm,roxy).
father(barney,bamm-bamm).
father(bamm-bamm,chip).
mother(pebbles,roxy).
mother(pebbles,chip).
mother(wilma,pebbles).
mother(betty,bamm-bamm).
grandmother(X,Y) :-
 mother(X,Z),parent(Z,Y).
parent(X,Y) :-
mother(X,Y).
parent(X,Y) :-
father(X,Y).
```

?-

resolvent =

```
Initialize resolvent to goal G (the query)
while resolvent not empty do
    choose a goal A from the resolvent
    choose a ground instance of a clause
        A' :- B1,…,Bn from program P
        such that A and A' are identical
        (if no such goal and clause exist, exit
        the while loop)
    replace A by B1,…,Bn in the resolvent
If the resolvent is empty, then output yes,
   else output no
```

What if P has rules?

The program (*P*):

```
father(fred,pebbles).
father(bamm-bamm,roxy).
father(barney,bamm-bamm).
father(bamm-bamm,chip).
mother(pebbles,roxy).
mother(pebbles,chip).
mother(wilma,pebbles).
mother(betty,bamm-bamm).
grandmother(X,Y) :-
 mother(X,Z),parent(Z,Y).
parent(X,Y) :-
mother(X,Y).
parent(X,Y) :-
father(X,Y).
```

?- grandmother(betty,roxy).

resolvent =

```
Initialize resolvent to goal G (the query)
while resolvent not empty do
    choose a goal A from the resolvent
    choose a ground instance of a clause
        A' :- B1,…,Bn from program P
        such that A and A' are identical
        (if no such goal and clause exist, exit
        the while loop)
    replace A by B1,…,Bn in the resolvent
If the resolvent is empty, then output yes,
   else output no
```

What if P has rules?

The program (*P*):

```
father(fred,pebbles).
father(bamm-bamm,roxy).
father(barney,bamm-bamm).
father(bamm-bamm,chip).
mother(pebbles,roxy).
mother(pebbles,chip).
mother(wilma,pebbles).
mother(betty,bamm-bamm).
grandmother(X,Y) :-
 mother(X,Z),parent(Z,Y).
parent(X,Y) :-
mother(X,Y).
parent(X,Y) :-
father(X,Y).
```

?- grandmother(betty,roxy).

resolvent =

```
Initialize resolvent to goal G (the query)
while resolvent not empty do
    choose a goal A from the resolvent
    choose a ground instance of a clause
        A' :- B1,…,Bn from program P
        such that A and A' are identical
        (if no such goal and clause exist, exit
        the while loop)
    replace A by B1,…,Bn in the resolvent
If the resolvent is empty, then output yes,
   else output no
```

What if P has rules?

The program (*P*):

```
father(fred,pebbles).
father(bamm-bamm,roxy).
father(barney,bamm-bamm).
father(bamm-bamm,chip).
mother(pebbles,roxy).
mother(pebbles,chip).
mother(wilma,pebbles).
mother(betty,bamm-bamm).
grandmother(X,Y) :-
 mother(X,Z),parent(Z,Y).
parent(X,Y) :-
mother(X,Y).
parent(X,Y) :-
father(X,Y).
```

?- grandmother(betty,roxy).

resolvent = grandmother(betty,roxy)

```
Initialize resolvent to goal G (the query)
while resolvent not empty do
    choose a goal A from the resolvent
    choose a ground instance of a clause
        A' :- B1,…,Bn from program P
        such that A and A' are identical
        (if no such goal and clause exist, exit
        the while loop)
    replace A by B1,…,Bn in the resolvent
If the resolvent is empty, then output yes,
   else output no
```

What if P has rules?

The program (*P*):

```
father(fred,pebbles).
father(bamm-bamm,roxy).
father(barney,bamm-bamm).
father(bamm-bamm,chip).
mother(pebbles,roxy).
mother(pebbles,chip).
mother(wilma,pebbles).
mother(betty,bamm-bamm).
grandmother(X,Y) :-
 mother(X,Z),parent(Z,Y).
parent(X,Y) :-
mother(X,Y).
parent(X,Y) :-
father(X,Y).
```

?- grandmother(betty,roxy).

resolvent = grandmother(betty,roxy)

```
Initialize resolvent to goal G (the query)
while resolvent not empty do
    choose a goal A from the resolvent
    choose a ground instance of a clause
        A' :- B1,…,Bn from program P
        such that A and A' are identical
        (if no such goal and clause exist, exit
        the while loop)
    replace A by B1,…,Bn in the resolvent
If the resolvent is empty, then output yes,
   else output no
```

What if P has rules?

The program (*P*):

```
father(fred,pebbles).
father(bamm-bamm,roxy).
father(barney,bamm-bamm).
father(bamm-bamm,chip).
mother(pebbles,roxy).
mother(pebbles,chip).
mother(wilma,pebbles).
mother(betty,bamm-bamm).
grandmother(X,Y) :-
 mother(X,Z),parent(Z,Y).
parent(X,Y) :-
mother(X,Y).
parent(X,Y) :-
father(X,Y).
```

?- grandmother(betty,roxy).

resolvent = grandmother(betty,roxy)

```
Initialize resolvent to goal G (the query)
while resolvent not empty do
    choose a goal A from the resolvent
    choose a ground instance of a clause
        A' :- B1,…,Bn from program P
        such that A and A' are identical
        (if no such goal and clause exist, exit
        the while loop)
    replace A by B1,…,Bn in the resolvent
If the resolvent is empty, then output yes,
  else output no
```

What if P has rules?

The program (*P*):

```
father(fred,pebbles).
father(bamm-bamm,roxy).
father(barney,bamm-bamm).
father(bamm-bamm,chip).
mother(pebbles,roxy).
mother(pebbles,chip).
mother(wilma,pebbles).
mother(betty,bamm-bamm).
grandmother(X,Y) :-
 mother(X,Z),parent(Z,Y).
parent(X,Y) :-
mother(X,Y).
parent(X,Y) :-
father(X,Y).
```

?- grandmother(betty,roxy).

resolvent = grandmother(betty,roxy)

```
Initialize resolvent to goal G (the query)
while resolvent not empty do
    choose a goal A from the resolvent
    choose a ground instance of a clause
        A' :- B1,…,Bn from program P
        such that A and A' are identical
        (if no such goal and clause exist, exit
        the while loop)
    replace A by B1,…,Bn in the resolvent
If the resolvent is empty, then output yes,
   else output no
```

What if P has rules?

The program (*P*):

```
father(fred,pebbles).
father(bamm-bamm,roxy).
father(barney,bamm-bamm).
father(bamm-bamm,chip).
mother(pebbles,roxy).
mother(pebbles,chip).
mother(wilma,pebbles).
mother(betty,bamm-bamm).
grandmother(X,Y) :-
 mother(X,Z),parent(Z,Y).
parent(X,Y) :-
mother(X,Y).
parent(X,Y) :-
father(X,Y).
```

?- grandmother(betty,roxy).

resolvent = grandmother(betty,roxy)

```
Initialize resolvent to goal G (the query)
while resolvent not empty do
    choose a goal A from the resolvent
    choose a ground instance of a clause
        A' :- B1,…,Bn from program P
        such that A and A' are identical
        (if no such goal and clause exist, exit
        the while loop)
    replace A by B1,…,Bn in the resolvent
If the resolvent is empty, then output yes,
   else output no
```

What if P has rules?

The program (*P*):

```
father(fred,pebbles).
father(bamm-bamm,roxy).
father(barney,bamm-bamm).
father(bamm-bamm,chip).
mother(pebbles,roxy).
mother(pebbles,chip).
mother(wilma,pebbles).
mother(betty,bamm-bamm).
grandmother(X,Y) :-
 mother(X,Z),parent(Z,Y).
parent(X,Y) :-
mother(X,Y).
parent(X,Y) :-
father(X,Y).
```

?- grandmother(betty,roxy).

resolvent = grandmother(betty,roxy)

```
Initialize resolvent to goal G (the query)
while resolvent not empty do
    choose a goal A from the resolvent
    choose a ground instance of a clause
        A' :- B1,…,Bn from program P
        such that A and A' are identical
        (if no such goal and clause exist, exit
        the while loop)
    replace A by B1,…,Bn in the resolvent
If the resolvent is empty, then output yes,
   else output no
```

OOPS! Time to modify things a bit.

What if P has rules?

The program (*P*):

```
father(fred,pebbles).
father(bamm-bamm,roxy).
father(barney,bamm-bamm).
father(bamm-bamm,chip).
mother(pebbles,roxy).
mother(pebbles,chip).
mother(wilma,pebbles).
mother(betty,bamm-bamm).
grandmother(X,Y) :-
 mother(X,Z),parent(Z,Y).
parent(X,Y) :-
mother(X,Y).
parent(X,Y) :-
father(X,Y).
```

?- grandmother(betty,roxy).

resolvent = grandmother(betty,roxy)

```
Initialize resolvent to goal G (the query)
while resolvent not empty do
    choose a goal A from the resolvent
    choose a clause
        A' :- B1,…,Bn from program P
        such that A and A' unify
        (if no such goal and clause exist, exit
        the while loop)
    replace A by B1,…,Bn in the resolvent
If the resolvent is empty, then output yes,
   else output no
```

What if P has rules?

The program (*P*):

```
father(fred,pebbles).
father(bamm-bamm,roxy).
father(barney,bamm-bamm).
father(bamm-bamm,chip).
mother(pebbles,roxy).
mother(pebbles,chip).
mother(wilma,pebbles).
mother(betty,bamm-bamm).
grandmother(X,Y) :-
 mother(X,Z),parent(Z,Y).
parent(X,Y) :-
mother(X,Y).
parent(X,Y) :-
father(X,Y).
```

?- grandmother(betty,roxy).

resolvent = grandmother(betty,roxy)

```
Initialize resolvent to goal G (the query)
while resolvent not empty do
    choose a goal A from the resolvent
    choose a clause
        A' :- B1,…,Bn from program P
        such that A and A' unify
        (if no such goal and clause exist, exit
        the while loop)
    replace A by B1,…,Bn in the resolvent
If the resolvent is empty, then output yes,
  else output no
```

What if P has rules?

The program (*P*):

```
father(fred,pebbles).
father(bamm-bamm,roxy).
father(barney,bamm-bamm).
father(bamm-bamm,chip).
mother(pebbles,roxy).
mother(pebbles,chip).
mother(wilma,pebbles).
mother(betty,bamm-bamm).
grandmother(X,Y) :-
 mother(X,Z),parent(Z,Y).
parent(X,Y) :-
mother(X,Y).
parent(X,Y) :-
father(X,Y).
```

?- grandmother(betty,roxy).

resolvent = grandmother(betty,roxy)

```
Initialize resolvent to goal G (the query)
while resolvent not empty do
    choose a goal A from the resolvent
    choose a clause
        A' :- B1,…,Bn from program P
        such that A and A' unify
        (if no such goal and clause exist, exit
        the while loop)
    replace A by B1,…,Bn in the resolvent
If the resolvent is empty, then output yes,
  else output no
```

What if P has rules?

The program (*P*):

```
father(fred,pebbles).
father(bamm-bamm,roxy).
father(barney,bamm-bamm).
father(bamm-bamm,chip).
mother(pebbles,roxy).
mother(pebbles,chip).
mother(wilma,pebbles).
mother(betty,bamm-bamm).
grandmother(X,Y) :-
 mother(X,Z),parent(Z,Y).
parent(X,Y) :-
mother(X,Y).
parent(X,Y) :-
father(X,Y).
```

?- grandmother(betty,roxy).

resolvent = grandmother(betty,roxy)

```
Initialize resolvent to goal G (the query)
*while* resolvent not empty *do*
    choose a goal *A* from the resolvent
    choose a clause
        *A'* :- *B1*,…,*Bn* from program *P*
        such that *A* and *A'* unify
        (if no such goal and clause exist, exit
        the while loop)
    replace *A* by *B1*,…,*Bn* in the resolvent
        (after performing the appropriate
        substitutions)
*If* the resolvent is empty, *then* output *yes*,
    *else* output *no*
```

What if P has rules?

The program (*P*):

```
father(fred,pebbles).
father(bamm-bamm,roxy).
father(barney,bamm-bamm).
father(bamm-bamm,chip).
mother(pebbles,roxy).
mother(pebbles,chip).
mother(wilma,pebbles).
mother(betty,bamm-bamm).
grandmother(betty,roxy) :-
 mother(betty,Z),parent(Z,roxy).
parent(X,Y) :- mother(X,Y).
parent(X,Y) :- father(X,Y).
```

?- grandmother(betty,roxy).

resolvent = grandmother(betty,roxy)

```
Initialize resolvent to goal G (the query)
while resolvent not empty do
    choose a goal A from the resolvent
    choose a clause
        A' :- B1,…,Bn from program P
        such that A and A' unify
        (if no such goal and clause exist, exit
        the while loop)
    replace A by B1,…,Bn in the resolvent
        (after performing the appropriate
        substitutions)
If the resolvent is empty, then output yes,
  else output no
```

What if P has rules?

The program (*P*):

```
father(fred,pebbles).
father(bamm-bamm,roxy).
father(barney,bamm-bamm).
father(bamm-bamm,chip).
mother(pebbles,roxy).
mother(pebbles,chip).
mother(wilma,pebbles).
mother(betty,bamm-bamm).
grandmother(betty,roxy) :-
 mother(betty,Z),parent(Z,roxy).
parent(X,Y) :- mother(X,Y).
parent(X,Y) :- father(X,Y).
```

?- grandmother(betty,roxy).

resolvent = mother(betty,Z),
parent(Z,roxy)

```
Initialize resolvent to goal G (the query)
while resolvent not empty do
    choose a goal A from the resolvent
    choose a clause
        A' :- B1,…,Bn from program P
        such that A and A' unify
        (if no such goal and clause exist, exit
        the while loop)
    replace A by B1,…,Bn in the resolvent
        (after performing the appropriate
        substitutions)
If the resolvent is empty, then output yes,
   else output no
```

What if P has rules?

The program (*P*):

```
father(fred,pebbles).
father(bamm-bamm,roxy).
father(barney,bamm-bamm).
father(bamm-bamm,chip).
mother(pebbles,roxy).
mother(pebbles,chip).
mother(wilma,pebbles).
mother(betty,bamm-bamm).
grandmother(X,Y) :-
 mother(X,Z),parent(Z,Y).
parent(X,Y) :-
mother(X,Y).
parent(X,Y) :-
father(X,Y).
```

?- grandmother(betty,roxy).

resolvent = mother(betty,Z),
            parent(Z,roxy)

```
Initialize resolvent to goal G (the query)
while resolvent not empty do
    choose a goal A from the resolvent
    choose a clause
        A' :- B1,…,Bn from program P
        such that A and A' unify
        (if no such goal and clause exist, exit
        the while loop)
    replace A by B1,…,Bn in the resolvent
        (after performing the appropriate
        substitutions)
If the resolvent is empty, then output yes,
   else output no
```

What if P has rules?

The program (*P*):

```
father(fred,pebbles).
father(bamm-bamm,roxy).
father(barney,bamm-bamm).
father(bamm-bamm,chip).
mother(pebbles,roxy).
mother(pebbles,chip).
mother(wilma,pebbles).
mother(betty,bamm-bamm).
grandmother(X,Y) :-
 mother(X,Z),parent(Z,Y).
parent(X,Y) :-
mother(X,Y).
parent(X,Y) :-
father(X,Y).
```

?- grandmother(betty,roxy).

```
resolvent = mother(betty,Z),
            parent(Z,roxy)
```

```
Initialize resolvent to goal G (the query)
while resolvent not empty do
    choose a goal A from the resolvent
    choose a clause
        A' :- B1,…,Bn from program P
        such that A and A' unify
        (if no such goal and clause exist, exit
        the while loop)
    replace A by B1,…,Bn in the resolvent
        (after performing the appropriate
        substitutions)
If the resolvent is empty, then output yes,
   else output no
```

What if P has rules?

The program (*P*):

```
father(fred,pebbles).
father(bamm-bamm,roxy).
father(barney,bamm-bamm).
father(bamm-bamm,chip).
mother(pebbles,roxy).
mother(pebbles,chip).
mother(wilma,pebbles).
mother(betty,bamm-bamm).
grandmother(X,Y) :-
 mother(X,Z),parent(Z,Y).
parent(X,Y) :-
mother(X,Y).
parent(X,Y) :-
father(X,Y).
```

?- grandmother(betty,roxy).

```
resolvent = mother(betty,Z),
            parent(Z,roxy)
```

```
Initialize resolvent to goal G (the query)
while resolvent not empty do
    choose a goal A from the resolvent
    choose a clause
        A' :- B1,…,Bn from program P
        such that A and A' unify
        (if no such goal and clause exist, exit
        the while loop)
    replace A by B1,…,Bn in the resolvent
        (after performing the appropriate
        substitutions)
If the resolvent is empty, then output yes,
  else output no
```

What if P has rules?

The program (*P*):

```
father(fred,pebbles).
father(bamm-bamm,roxy).
father(barney,bamm-bamm).
father(bamm-bamm,chip).
mother(pebbles,roxy).
mother(pebbles,chip).
mother(wilma,pebbles).
mother(betty,bamm-bamm).
grandmother(X,Y) :-
 mother(X,Z),parent(Z,Y).
parent(X,Y) :-
mother(X,Y).
parent(X,Y) :-
father(X,Y).
```

?- grandmother(betty,roxy).

resolvent = mother(betty,Z),
            parent(Z,roxy)

```
Initialize resolvent to goal G (the query)
*while* resolvent not empty *do*
    choose a goal *A* from the resolvent
    choose a clause
        *A'* :- *B1*,…,*Bn* from program *P*
        such that *A* and *A'* unify
        (if no such goal and clause exist, exit
        the while loop)
    replace *A* by *B1*,…,*Bn* in the resolvent
        (after performing the appropriate
        substitutions)
*If* the resolvent is empty, *then* output *yes*,
  *else* output *no*
```

What if P has rules?

The program (*P*):

```
father(fred,pebbles).
father(bamm-bamm,roxy).
father(barney,bamm-bamm).
father(bamm-bamm,chip).
mother(pebbles,roxy).
mother(pebbles,chip).
mother(wilma,pebbles).
mother(betty,bamm-bamm).
grandmother(X,Y) :-
 mother(X,Z),parent(Z,Y).
parent(X,Y) :-
mother(X,Y).
parent(X,Y) :-
father(X,Y).
```

?- grandmother(betty,roxy).

resolvent = mother(betty,Z),
parent(Z,roxy)

```
Initialize resolvent to goal G (the query)
while resolvent not empty do
    choose a goal A from the resolvent
    choose a clause
        A' :- B1,…,Bn from program P
        such that A and A' unify
        (if no such goal and clause exist, exit
        the while loop)
    replace A by B1,…,Bn in the resolvent
        (after performing the appropriate
        substitutions)
If the resolvent is empty, then output yes,
  else output no
```

What if P has rules?

The program (*P*):

```
father(fred,pebbles).
father(bamm-bamm,roxy).
father(barney,bamm-bamm).
father(bamm-bamm,chip).
mother(pebbles,roxy).
mother(pebbles,chip).
mother(wilma,pebbles).
mother(betty,bamm-bamm).
grandmother(X,Y) :-
 mother(X,Z),parent(Z,Y).
parent(X,Y) :-
mother(X,Y).
parent(X,Y) :-
father(X,Y).
```

?- grandmother(betty,roxy).

resolvent = mother(betty,Z),
                      parent(Z,roxy)

```
Initialize resolvent to goal G (the query)
while resolvent not empty do
    choose a goal A from the resolvent
    choose a clause
        A' :- B1,…,Bn from program P
        such that A and A' unify
        (if no such goal and clause exist, exit
        the while loop)
    replace A by B1,…,Bn in the resolvent
        (after performing the appropriate
        substitutions)
If the resolvent is empty, then output yes,
    else output no
```

What if P has rules?

The program (*P*):

```
father(fred,pebbles).
father(bamm-bamm,roxy).
father(barney,bamm-bamm).
father(bamm-bamm,chip).
mother(pebbles,roxy).
mother(pebbles,chip).
mother(wilma,pebbles).
mother(betty,bamm-bamm).
grandmother(X,Y) :-
 mother(X,Z),parent(Z,Y).
parent(X,Y) :-
mother(X,Y).
parent(X,Y) :-
father(X,Y).
```

?- grandmother(betty,roxy).

resolvent = mother(betty,bamm-bamm),
parent(bamm-bamm,roxy)

```
Initialize resolvent to goal G (the query)
while resolvent not empty do
    choose a goal A from the resolvent
    choose a clause
        A' :- B1,…,Bn from program P
        such that A and A' unify
        (if no such goal and clause exist, exit
        the while loop)
    replace A by B1,…,Bn in the resolvent
        (after performing the appropriate
        substitutions)
If the resolvent is empty, then output yes,
    else output no
```

What if P has rules?

The program (*P*):

```
father(fred,pebbles).
father(bamm-bamm,roxy).
father(barney,bamm-bamm).
father(bamm-bamm,chip).
mother(pebbles,roxy).
mother(pebbles,chip).
mother(wilma,pebbles).
mother(betty,bamm-bamm).
grandmother(X,Y) :-
 mother(X,Z),parent(Z,Y).
parent(X,Y) :-
mother(X,Y).
parent(X,Y) :-
father(X,Y).
```

?- grandmother(betty,roxy).

resolvent = parent(bamm-bamm,roxy)

```
Initialize resolvent to goal G (the query)
while resolvent not empty do
    choose a goal A from the resolvent
    choose a clause
        A' :- B1,…,Bn from program P
        such that A and A' unify
        (if no such goal and clause exist, exit
        the while loop)
    replace A by B1,…,Bn in the resolvent
        (after performing the appropriate
        substitutions)
If the resolvent is empty, then output yes,
    else output no
```

What if P has rules?

The program (*P*):

```
father(fred,pebbles).
father(bamm-bamm,roxy).
father(barney,bamm-bamm).
father(bamm-bamm,chip).
mother(pebbles,roxy).
mother(pebbles,chip).
mother(wilma,pebbles).
mother(betty,bamm-bamm).
grandmother(X,Y) :-
 mother(X,Z),parent(Z,Y).
parent(X,Y) :-
mother(X,Y).
parent(X,Y) :-
father(X,Y).
```

?- grandmother(betty,roxy).

resolvent = parent(bamm-bamm,roxy)

```
Initialize resolvent to goal G (the query)
*while* resolvent not empty *do*
    choose a goal A from the resolvent
    choose a clause
        A' :- B1,…,Bn from program P
        such that A and A' unify
        (if no such goal and clause exist, exit
        the while loop)
    replace A by B1,…,Bn in the resolvent
        (after performing the appropriate
        substitutions)
*If* the resolvent is empty, *then* output *yes*,
    *else* output *no*
```

What if P has rules?

The program (*P*):

```
father(fred,pebbles).
father(bamm-bamm,roxy).
father(barney,bamm-bamm).
father(bamm-bamm,chip).
mother(pebbles,roxy).
mother(pebbles,chip).
mother(wilma,pebbles).
mother(betty,bamm-bamm).
grandmother(X,Y) :-
 mother(X,Z),parent(Z,Y).
parent(X,Y) :-
mother(X,Y).
parent(X,Y) :-
father(X,Y).
```

?- grandmother(betty,roxy).

resolvent = parent(bamm-bamm,roxy)

```
Initialize resolvent to goal G (the query)
while resolvent not empty do
    choose a goal A from the resolvent
    choose a clause
        A' :- B1,…,Bn from program P
        such that A and A' unify
        (if no such goal and clause exist, exit
        the while loop)
    replace A by B1,…,Bn in the resolvent
        (after performing the appropriate
        substitutions)
If the resolvent is empty, then output yes,
    else output no
```

What if P has rules?

The program (*P*):

```
father(fred,pebbles).
father(bamm-bamm,roxy).
father(barney,bamm-bamm).
father(bamm-bamm,chip).
mother(pebbles,roxy).
mother(pebbles,chip).
mother(wilma,pebbles).
mother(betty,bamm-bamm).
grandmother(X,Y) :-
 mother(X,Z),parent(Z,Y).
parent(X,Y) :-
mother(X,Y).
parent(X,Y) :-
father(X,Y).
```

?- grandmother(betty,roxy).

resolvent = parent(bamm-bamm,roxy)

```
Initialize resolvent to goal G (the query)
while resolvent not empty do
    choose a goal A from the resolvent
    choose a clause
        A' :- B1,…,Bn from program P
        such that A and A' unify
        (if no such goal and clause exist, exit
        the while loop)
    replace A by B1,…,Bn in the resolvent
        (after performing the appropriate
        substitutions)
If the resolvent is empty, then output yes,
    else output no
```

What if P has rules?

The program (*P*):

```
father(fred,pebbles).
father(bamm-bamm,roxy).
father(barney,bamm-bamm).
father(bamm-bamm,chip).
mother(pebbles,roxy).
mother(pebbles,chip).
mother(wilma,pebbles).
mother(betty,bamm-bamm).
grandmother(X,Y) :-
 mother(X,Z),parent(Z,Y).
parent(X,Y) :-
mother(X,Y).
parent(X,Y) :-
father(X,Y).
```

?- grandmother(betty,roxy).

resolvent = parent(bamm-bamm,roxy)

```
Initialize resolvent to goal G (the query)
while resolvent not empty do
    choose a goal A from the resolvent
    choose a clause
        A' :- B1,…,Bn from program P
        such that A and A' unify
        (if no such goal and clause exist, exit
        the while loop)
    replace A by B1,…,Bn in the resolvent
        (after performing the appropriate
        substitutions)
If the resolvent is empty, then output yes,
    else output no
```

What if P has rules?

The program (*P*):

```
father(fred,pebbles).
father(bamm-bamm,roxy).
father(barney,bamm-bamm).
father(bamm-bamm,chip).
mother(pebbles,roxy).
mother(pebbles,chip).
mother(wilma,pebbles).
mother(betty,bamm-bamm).
grandmother(X,Y) :-
 mother(X,Z),parent(Z,Y).
parent(X,Y) :-
mother(X,Y).
parent(X,Y) :-
father(X,Y).
```

?- grandmother(betty,roxy).

resolvent = parent(bamm-bamm,roxy)

```
Initialize resolvent to goal G (the query)
while resolvent not empty do
    choose a goal A from the resolvent
    choose a clause
        A' :- B1,…,Bn from program P
        such that A and A' unify
        (if no such goal and clause exist, exit
        the while loop)
    replace A by B1,…,Bn in the resolvent
        (after performing the appropriate
        substitutions)
If the resolvent is empty, then output yes,
    else output no
```

What if P has rules?

The program (*P*):

```
father(fred,pebbles).
father(bamm-bamm,roxy).
father(barney,bamm-bamm).
father(bamm-bamm,chip).
mother(pebbles,roxy).
mother(pebbles,chip).
mother(wilma,pebbles).
mother(betty,bamm-bamm).
grandmother(X,Y) :-
 mother(X,Z),parent(Z,Y).
parent(X,Y) :-
mother(X,Y).
parent(X,Y) :-
father(X,Y).
```

?- grandmother(betty,roxy).

resolvent = parent(bamm-bamm,roxy)

```
Initialize resolvent to goal G (the query)
while resolvent not empty do
    choose a goal A from the resolvent
    choose a clause
        A' :- B1,…,Bn from program P
        such that A and A' unify
        (if no such goal and clause exist, exit
        the while loop)
    replace A by B1,…,Bn in the resolvent
        (after performing the appropriate
        substitutions)
If the resolvent is empty, then output yes,
    else output no
```

What if P has rules?

The program (*P*):

```
father(fred,pebbles).
father(bamm-bamm,roxy).
father(barney,bamm-bamm).
father(bamm-bamm,chip).
mother(pebbles,roxy).
mother(pebbles,chip).
mother(wilma,pebbles).
mother(betty,bamm-bamm).
grandmother(X,Y) :-
 mother(X,Z),parent(Z,Y).
parent(X,Y) :-
mother(X,Y).
parent(bamm-bamm,roxy) :-
 father(bamm-bamm,roxy).
```

?- grandmother(betty,roxy).

resolvent = parent(bamm-bamm,roxy)

```
Initialize resolvent to goal G (the query)
while resolvent not empty do
    choose a goal A from the resolvent
    choose a clause
        A' :- B1,…,Bn from program P
        such that A and A' unify
        (if no such goal and clause exist, exit
        the while loop)
    replace A by B1,…,Bn in the resolvent
        (after performing the appropriate
        substitutions)
If the resolvent is empty, then output yes,
    else output no
```

What if P has rules?

The program (*P*):

```
father(fred,pebbles).
father(bamm-bamm,roxy).
father(barney,bamm-bamm).
father(bamm-bamm,chip).
mother(pebbles,roxy).
mother(pebbles,chip).
mother(wilma,pebbles).
mother(betty,bamm-bamm).
grandmother(X,Y) :-
 mother(X,Z),parent(Z,Y).
parent(X,Y) :-
mother(X,Y).
parent(bamm-bamm,roxy) :-
 father(bamm-bamm,roxy).
```

?- grandmother(betty,roxy).

resolvent = father(bamm-bamm,roxy)

```
Initialize resolvent to goal G (the query)
while resolvent not empty do
    choose a goal A from the resolvent
    choose a clause
        A' :- B1,…,Bn from program P
        such that A and A' unify
        (if no such goal and clause exist, exit
        the while loop)
    replace A by B1,…,Bn in the resolvent
        (after performing the appropriate
        substitutions)
If the resolvent is empty, then output yes,
    else output no
```

What if P has rules?

The program (*P*):

```
father(fred,pebbles).
father(bamm-bamm,roxy).
father(barney,bamm-bamm).
father(bamm-bamm,chip).
mother(pebbles,roxy).
mother(pebbles,chip).
mother(wilma,pebbles).
mother(betty,bamm-bamm).
grandmother(X,Y) :-
 mother(X,Z),parent(Z,Y).
parent(X,Y) :-
mother(X,Y).
parent(X,Y) :-
father(X,Y).
```

?- grandmother(betty,roxy).

resolvent = father(bamm-bamm,roxy)

```
Initialize resolvent to goal G (the query)
while resolvent not empty do
    choose a goal A from the resolvent
    choose a clause
        A' :- B1,…,Bn from program P
        such that A and A' unify
        (if no such goal and clause exist, exit
        the while loop)
    replace A by B1,…,Bn in the resolvent
        (after performing the appropriate
        substitutions)
If the resolvent is empty, then output yes,
    else output no
```

What if P has rules?

The program (*P*):

```
father(fred,pebbles).
father(bamm-bamm,roxy).
father(barney,bamm-bamm).
father(bamm-bamm,chip).
mother(pebbles,roxy).
mother(pebbles,chip).
mother(wilma,pebbles).
mother(betty,bamm-bamm).
grandmother(X,Y) :-
 mother(X,Z),parent(Z,Y).
parent(X,Y) :-
mother(X,Y).
parent(X,Y) :-
father(X,Y).
```

?- grandmother(betty,roxy).

resolvent = father(bamm-bamm,roxy)

```
Initialize resolvent to goal G (the query)
while resolvent not empty do
    choose a goal A from the resolvent
    choose a clause
        A' :- B1,…,Bn from program P
        such that A and A' unify
        (if no such goal and clause exist, exit
        the while loop)
    replace A by B1,…,Bn in the resolvent
        (after performing the appropriate
        substitutions)
If the resolvent is empty, then output yes,
    else output no
```

What if P has rules?

The program (*P*):

```
father(fred,pebbles).
father(bamm-bamm,roxy).
father(barney,bamm-bamm).
father(bamm-bamm,chip).
mother(pebbles,roxy).
mother(pebbles,chip).
mother(wilma,pebbles).
mother(betty,bamm-bamm).
grandmother(X,Y) :-
 mother(X,Z),parent(Z,Y).
parent(X,Y) :-
mother(X,Y).
parent(X,Y) :-
father(X,Y).
```

?- grandmother(betty,roxy).

resolvent = father(bamm-bamm,roxy)

```
Initialize resolvent to goal G (the query)
while resolvent not empty do
    choose a goal A from the resolvent
    choose a clause
        A' :- B1,…,Bn from program P
        such that A and A' unify
        (if no such goal and clause exist, exit
        the while loop)
    replace A by B1,…,Bn in the resolvent
        (after performing the appropriate
        substitutions)
If the resolvent is empty, then output yes,
    else output no
```

What if P has rules?

The program (*P*):

```
father(fred,pebbles).
father(bamm-bamm,roxy).
father(barney,bamm-bamm).
father(bamm-bamm,chip).
mother(pebbles,roxy).
mother(pebbles,chip).
mother(wilma,pebbles).
mother(betty,bamm-bamm).
grandmother(X,Y) :-
 mother(X,Z),parent(Z,Y).
parent(X,Y) :-
mother(X,Y).
parent(X,Y) :-
father(X,Y).
```

?- grandmother(betty,roxy).

resolvent = father(bamm-bamm,roxy)

```
Initialize resolvent to goal G (the query)
*while* resolvent not empty *do*
    choose a goal *A* from the resolvent
    choose a clause
        *A'* :- *B1,…,Bn* from program *P*
        such that *A* and *A'* unify
        (if no such goal and clause exist, exit
        the while loop)
    replace *A* by *B1,…,Bn* in the resolvent
        (after performing the appropriate
        substitutions)
*If* the resolvent is empty, *then* output *yes*,
    *else* output *no*
```

What if P has rules?

The program (*P*):

```
father(fred,pebbles).
father(bamm-bamm,roxy).
father(barney,bamm-bamm).
father(bamm-bamm,chip).
mother(pebbles,roxy).
mother(pebbles,chip).
mother(wilma,pebbles).
mother(betty,bamm-bamm).
grandmother(X,Y) :-
 mother(X,Z),parent(Z,Y).
parent(X,Y) :-
mother(X,Y).
parent(X,Y) :-
father(X,Y).
```

?- grandmother(betty,roxy).

resolvent = father(bamm-bamm,roxy)

```
Initialize resolvent to goal G (the query)
while resolvent not empty do
    choose a goal A from the resolvent
    choose a clause
        A' :- B1,…,Bn from program P
        such that A and A' unify
        (if no such goal and clause exist, exit
        the while loop)
    replace A by B1,…,Bn in the resolvent
        (after performing the appropriate
        substitutions)
If the resolvent is empty, then output yes,
    else output no
```

What if P has rules?

The program (*P*):

```
father(fred,pebbles).
father(bamm-bamm,roxy).
father(barney,bamm-bamm).
father(bamm-bamm,chip).
mother(pebbles,roxy).
mother(pebbles,chip).
mother(wilma,pebbles).
mother(betty,bamm-bamm).
grandmother(X,Y) :-
 mother(X,Z),parent(Z,Y).
parent(X,Y) :-
mother(X,Y).
parent(X,Y) :-
father(X,Y).
```

?- grandmother(betty,roxy).

resolvent = father(bamm-bamm,roxy)

```
Initialize resolvent to goal G (the query)
while resolvent not empty do
    choose a goal A from the resolvent
    choose a clause
        A' :- B1,…,Bn from program P
        such that A and A' unify
        (if no such goal and clause exist, exit
        the while loop)
    replace A by B1,…,Bn in the resolvent
        (after performing the appropriate
        substitutions)
If the resolvent is empty, then output yes,
    else output no
```

What if P has rules?

The program (*P*):

```
father(fred,pebbles).
father(bamm-bamm,roxy).
father(barney,bamm-bamm).
father(bamm-bamm,chip).
mother(pebbles,roxy).
mother(pebbles,chip).
mother(wilma,pebbles).
mother(betty,bamm-bamm).
grandmother(X,Y) :-
 mother(X,Z),parent(Z,Y).
parent(X,Y) :-
mother(X,Y).
parent(X,Y) :-
father(X,Y).
```

?- grandmother(betty,roxy).

resolvent =

```
Initialize resolvent to goal G (the query)
while resolvent not empty do
    choose a goal A from the resolvent
    choose a clause
        A' :- B1,…,Bn from program P
        such that A and A' unify
        (if no such goal and clause exist, exit
        the while loop)
    replace A by B1,…,Bn in the resolvent
        (after performing the appropriate
        substitutions)
If the resolvent is empty, then output yes,
    else output no
```

What if P has rules?

The program (*P*):

```
father(fred,pebbles).
father(bamm-bamm,roxy).
father(barney,bamm-bamm).
father(bamm-bamm,chip).
mother(pebbles,roxy).
mother(pebbles,chip).
mother(wilma,pebbles).
mother(betty,bamm-bamm).
grandmother(X,Y) :-
 mother(X,Z),parent(Z,Y).
parent(X,Y) :-
mother(X,Y).
parent(X,Y) :-
father(X,Y).
```

?- grandmother(betty,roxy).

resolvent =

```
Initialize resolvent to goal G (the query)
while resolvent not empty do
    choose a goal A from the resolvent
    choose a clause
        A' :- B1,…,Bn from program P
        such that A and A' unify
        (if no such goal and clause exist, exit
        the while loop)
    replace A by B1,…,Bn in the resolvent
        (after performing the appropriate
        substitutions)
If the resolvent is empty, then output yes,
   else output no
```

What if P has rules?

The program (*P*):

```
father(fred,pebbles).
father(bamm-bamm,roxy).
father(barney,bamm-bamm).
father(bamm-bamm,chip).
mother(pebbles,roxy).
mother(pebbles,chip).
mother(wilma,pebbles).
mother(betty,bamm-bamm).
grandmother(X,Y) :-
 mother(X,Z),parent(Z,Y).
parent(X,Y) :-
mother(X,Y).
parent(X,Y) :-
father(X,Y).
```

?- grandmother(betty,roxy).

resolvent =

```
Initialize resolvent to goal G (the query)
while resolvent not empty do
    choose a goal A from the resolvent
    choose a clause
        A' :- B1,…,Bn from program P
        such that A and A' unify
        (if no such goal and clause exist, exit
        the while loop)
    replace A by B1,…,Bn in the resolvent
        (after performing the appropriate
        substitutions)
If the resolvent is empty, then output yes,
   else output no
```

What if P has rules?

The program (*P*):

```
father(fred,pebbles).
father(bamm-bamm,roxy).
father(barney,bamm-bamm).
father(bamm-bamm,chip).
mother(pebbles,roxy).
mother(pebbles,chip).
mother(wilma,pebbles).
mother(betty,bamm-bamm).
grandmother(X,Y) :-
 mother(X,Z),parent(Z,Y).
parent(X,Y) :-
mother(X,Y).
parent(X,Y) :-
father(X,Y).
```

```
?- grandmother(betty,roxy).
yes
resolvent =
```

```
Initialize resolvent to goal G (the query)
while resolvent not empty do
    choose a goal A from the resolvent
    choose a clause
        A' :- B1,…,Bn from program P
        such that A and A' unify
        (if no such goal and clause exist, exit
        the while loop)
    replace A by B1,…,Bn in the resolvent
        (after performing the appropriate
        substitutions)
If the resolvent is empty, then output yes,
   else output no
```

Questions?