CS61A Lecture 27

2011-08-04

Colleen Lewis

Run (query) and tell it some facts

```
STk> (load "query.scm")
okay
STk> (query)
;;; Query input:
(assert! (colleen likes cookies))

Assertion added to data base.

;;; Query input:
```

Like (mce): it starts an infinite loop

Tell the system facts

Query system goal:

- Fill in the variables in the query (with all possible sets of values)
- NEVER fill in the variables with an inconsistent set of variables

Query: (?who is the parent of bart)

```
(abraham is the parent of herb)
(abraham is the parent of homer)

(mona is the parent of herb)
(mona is the parent of homer)

(clancy is the parent of marge)
(clancy is the parent of patty)
(clancy is the parent of selma)

(jackeline is the parent of marge)
(jackeline is the parent of patty)
(jackeline is the parent of selma)

(rule (?a is the car of (?a . ?b)))
```

```
(homer is the parent of bart)
(homer is the parent of lisa)
(homer is the parent of maggie)

(marge is the parent of bart)
(marge is the parent of lisa)
(marge is the parent of maggie)

(selma is the parent of ling)

(female mona)
(female jackie)
(female marge)
(female patty)
(female selma)
(female lisa)
(female maggie)
(female ling)
```

Query: (?who is the parent of bart)

TWO facts match:

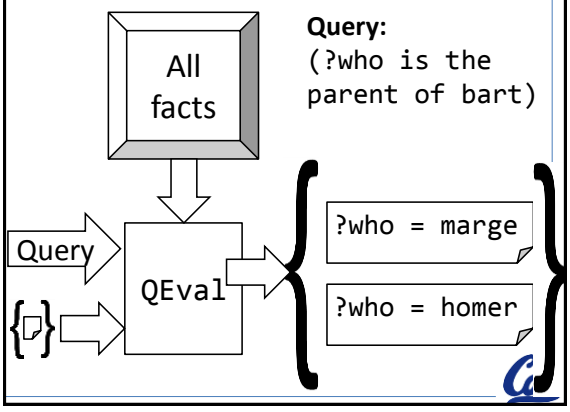
(?who	is	the	parent	of	bart)	;query
(homer	is	the	parent	of	bart)	;fact

?who = homer

These are frames!

(?who	is	the	parent	of	bart)	;query
(marge	is	the	parent	of	bart)	;fact

?who = marge



Query:
(?who is the parent of bart)

Query: (marge is the parent of ?x)

(abraham is the parent of herb)
(abraham is the parent of homer)

(mona is the parent of herb)
(mona is the parent of homer)

(clancy is the parent of marge)
(clancy is the parent of patty)
(clancy is the parent of selma)

(jackeline is the parent of marge)
(jackeline is the parent of patty)
(jackeline is the parent of selma)

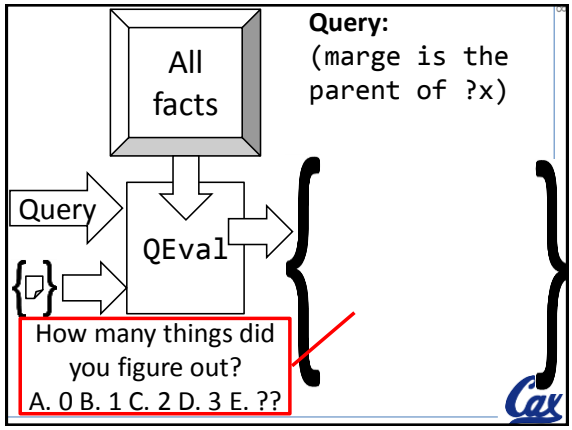
(rule (?a is the car of (?a . ?b))))

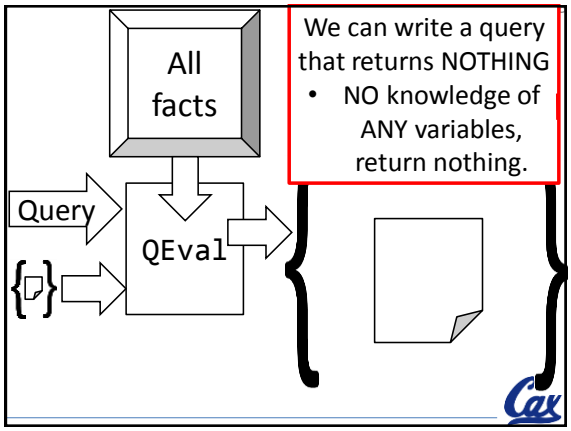
(homer is the parent of bart)
(homer is the parent of lisa)
(homer is the parent of maggie)

(marge is the parent of bart)
(marge is the parent of lisa)
(marge is the parent of maggie)

(selma is the parent of ling)

(female mona)
(female jackie)
(female marge)
(female patty)
(female selma)
(female lisa)
(female maggie)
(female ling)





Write a query that returns 0 results

(abraham is the parent of herb)
(abraham is the parent of homer)

(mona is the parent of herb)
(mona is the parent of homer)

(clancy is the parent of marge)
(clancy is the parent of patty)
(clancy is the parent of selma)

(jackeline is the parent of marge)
(jackeline is the parent of patty)
(jackeline is the parent of selma)

(rule (?a is the car of (?a . ?b))))

(homer is the parent of bart)
(homer is the parent of lisa)
(homer is the parent of maggie)

(marge is the parent of bart)
(marge is the parent of lisa)
(marge is the parent of maggie)

(selma is the parent of ling)

(female mona)
(female jackie)
(female marge)
(female patty)
(female selma)
(female lisa)
(female maggie)
(female ling)

Query: (female ?who)

(abraham is the parent of herb)
(abraham is the parent of homer)

(mona is the parent of herb)
(mona is the parent of homer)

(clancy is the parent of marge)
(clancy is the parent of patty)
(clancy is the parent of selma)

(jackeline is the parent of marge)
(jackeline is the parent of patty)
(jackeline is the parent of selma)

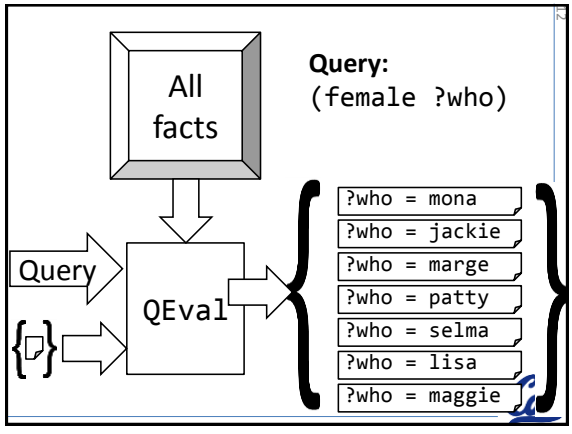
(rule (?a is the car of (?a . ?b))))

(homer is the parent of bart)
(homer is the parent of lisa)
(homer is the parent of maggie)

(marge is the parent of bart)
(marge is the parent of lisa)
(marge is the parent of maggie)

(selma is the parent of ling)

(female mona)
(female jackie)
(female marge)
(female patty)
(female selma)
(female lisa)
(female maggie)
(female ling)



Query: (and
(?who is the parent of bart)
(female ?who))

```
(abraham is the parent of herb)
(abraham is the parent of homer)

(mona is the parent of herb)
(mona is the parent of homer)

(homer is the parent of bart)
(homer is the parent of lisa)
(homer is the parent of maggie)

(marge is the parent of bart)
(marge is the parent of lisa)
(marge is the parent of maggie)

(selma is the parent of ling)

(female mona)
(female jackie)
(female marge)
(female patty)
(female selma)
(female lisa)
(female maggie)
(female ling)
```

Query Goals:
- Fill in the variables in the query (with all possible sets of values)
- NEVER fill in the variables with an inconsistent set of variables

Query: (and
(?who is the parent of bart);q1
(female ?who)) ;q2

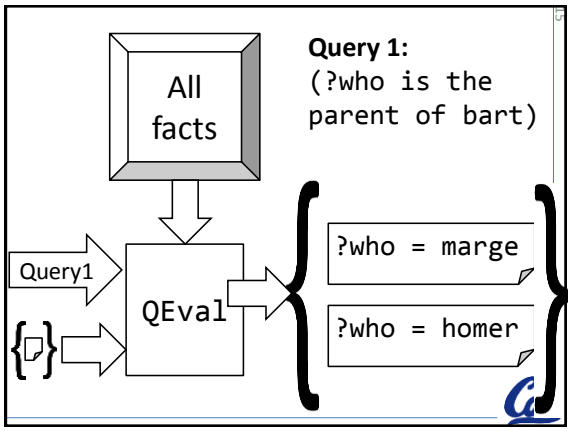
Left with TWO facts:

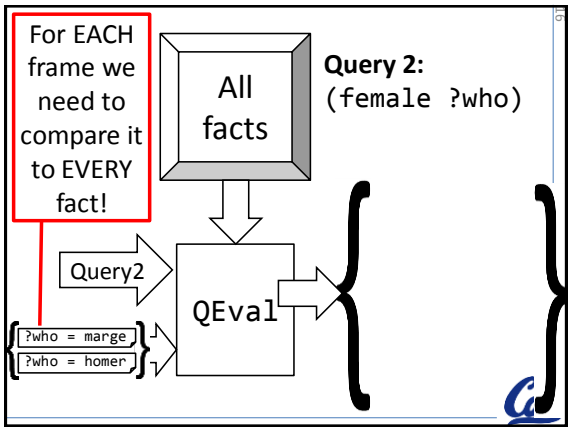
(?who	is	the	parent	of	bart)	;query
(homer	is	the	parent	of	bart)	;fact

?who = homer

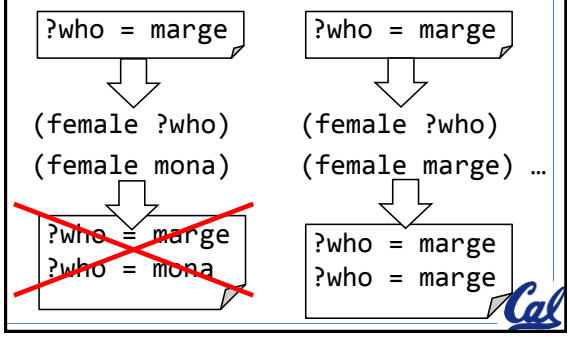
(?who	is	the	parent	of	bart)	;query
(marge	is	the	parent	of	bart)	;fact

?who = marge



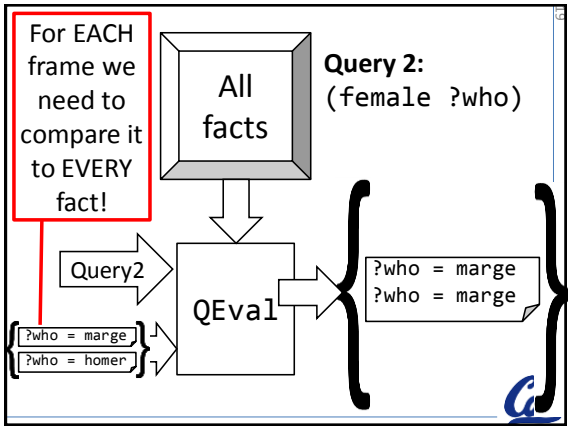


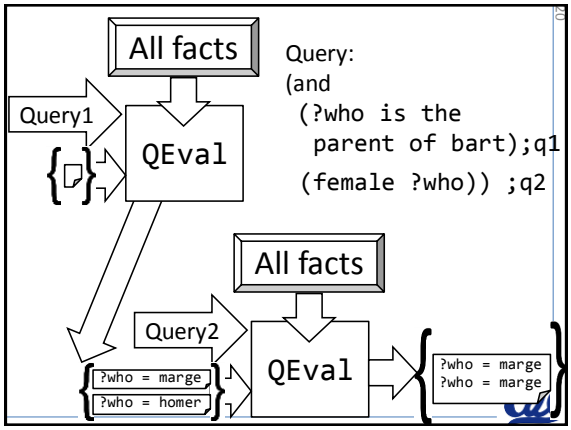
Query: (and
(?who is the parent of bart);q1
(female ?who)) ;q2



Query: (and
(?who is the parent of bart);q1
(female ?who)) ;q2

?who = marge	?who = homer
~~(female mona)~~	~~(female mona)~~
~~(female jackie)~~	~~(female jackie)~~
(female marge)	~~(female marge)~~
~~(female patty)~~	~~(female patty)~~
~~(female selma)~~	~~(female selma)~~
~~(female lisa)~~	~~(female lisa)~~
~~(female maggie)~~	~~(female maggie)~~
~~(female ling)~~	~~(female ling)~~



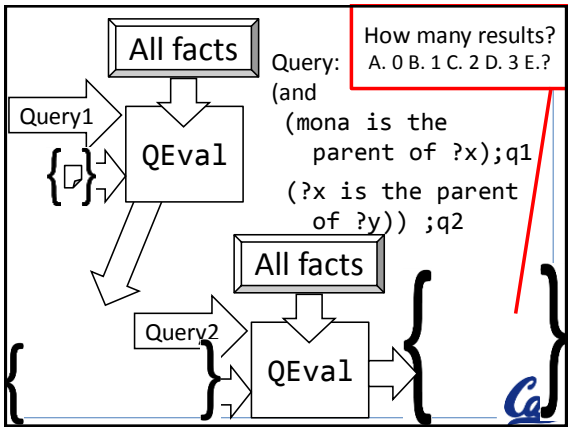


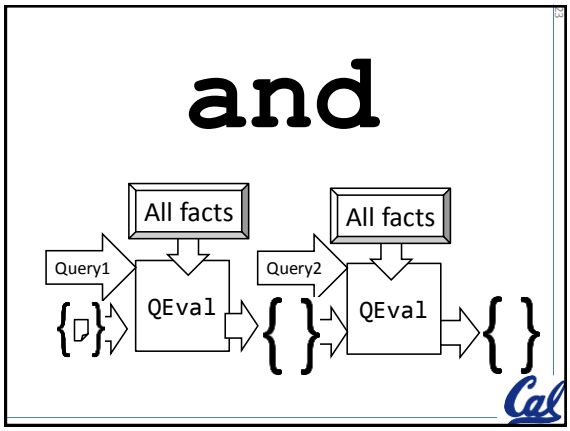
(and
(?who is the parent of bart)
(female ?who))

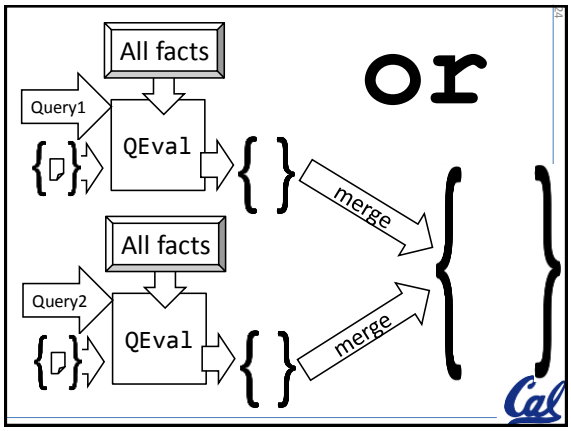
?who = marge
?who = marge

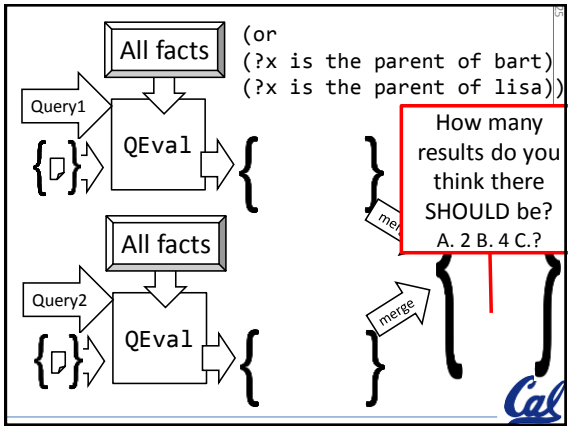
;;; Query results:
(and
(marge is the parent of bart)
(female marge))

Query Goals:
- Fill in the variables in the query (with all possible sets of values)
- NEVER fill in the variables with an inconsistent set of variables





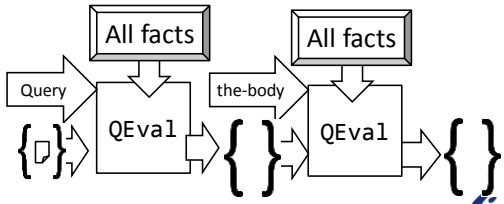





```
;;; Query input:
(or (?x is the parent of bart)
    (?x is the parent of ling))
;;; Query results:
```

rules

```
;;; Query input:
(assert! (rule <the-rule> <the-body>))
When a rule (a type of fact) matches, do this:
```



rules

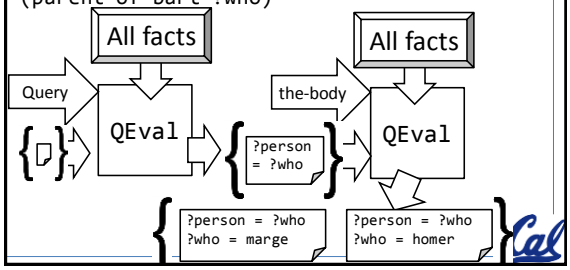
```
;;; Query input:
(assert!
  (rule (parent-of-bart ?person)
      (?person is the parent of bart)))
```

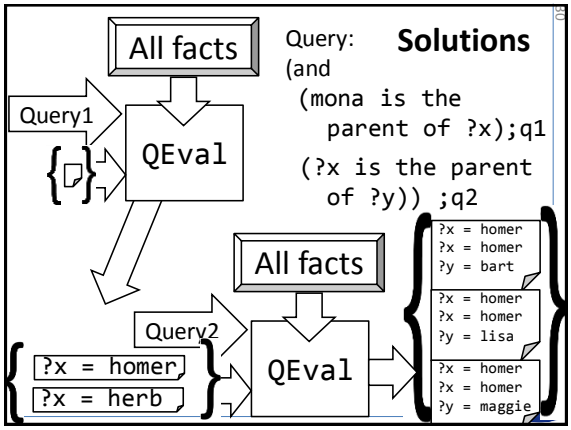
the-rule

the-body

rules

```
  (rule (parent-of-bart ?person)
      (?person is the parent of bart)))
;;; Query input:
(parent-of-bart ?who)
```





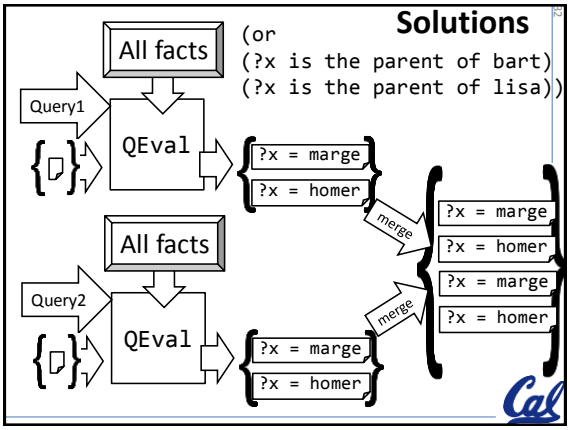
```
;;; Query input:
(and  (mona is the parent of ?x)
      (?x is the parent of ?y))
;;; Query results:
(and  (mona is the parent of homer)
      (homer is the parent of bart))
(and  (mona is the parent of homer)
      (homer is the parent of lisa))
(and  (mona is the parent of homer)
      (homer is the parent of maggie))
```

Solutions

?x = homer
?x = homer
?y = bart

?x = homer
?x = homer
?y = lisa

?x = homer
?x = homer
?y = maggie




```
;;; Query input:
(or (?x is the parent of bart)
    (?x is the parent of lisa))
;;; Query results:
(or (marge is the parent of bart)   ?x = marge
    (marge is the parent of lisa))
(or (homer is the parent of bart)   ?x = homer
    (homer is the parent of lisa))
(or (marge is the parent of bart)   ?x = marge
    (marge is the parent of lisa))
(or (homer is the parent of bart)   ?x = homer
    (homer is the parent of lisa))
```

```
;;; Query input:
(or (?x is the parent of bart)
    (?x is the parent of ling))
;;; Query results:
(or (marge is the parent of bart)   ?x = marge
    (marge is the parent of ling))
(or (homer is the parent of bart)   ?x = homer
    (homer is the parent of ling))
(or (selma is the parent of bart)   ?x = selma
    (selma is the parent of ling))
```

Solutions