Prolog lecture 5

Go to:

http://etc.ch/kG4y

Or scan the barcode

# Today's discussion

Videos

- Cut
- Negation
- Databases

Q: Why do the question numbers in the 'all exercises in a single PDF' jump by two each lecture? (Sorry, I know this question is useless but it's been bugging me).

A: Oops. Thanks for the report. I'll fix it (next year so as not to confuse things)

Q: is gnd() special in prolog or is it just a frequently used naming convention?

Q: is gnd() special in prolog or is it just a frequently used naming convention?

A: swipl has ground(X) which is true if X is a ground term. gnd() is just a compound term. Don't use ground(X) in the exam...

Q: All the methods taught so far (like generate and test) don't seem too efficient computationally. In the exam should we think of more complex logic to do so?

Q: All the methods taught so far (like generate and test) don't seem too efficient computationally. In the exam should we think of more complex logic to do so?

A: If the exam question is interested in efficiency it will say so...past questions have not asked this. You make generate and test more efficient by generating better!

Q: With drawing out the execution traces - it's pretty difficult to understand them if you look back at them. How can we convey it in an exam?

Q: With drawing out the execution traces - it's pretty difficult to understand them if you look back at them. How can we convey it in an exam?

A: I can't remember an exam question where I asked for a search tree to be drawn out. Instead a question might ask for what happens: e.g. what results do you get

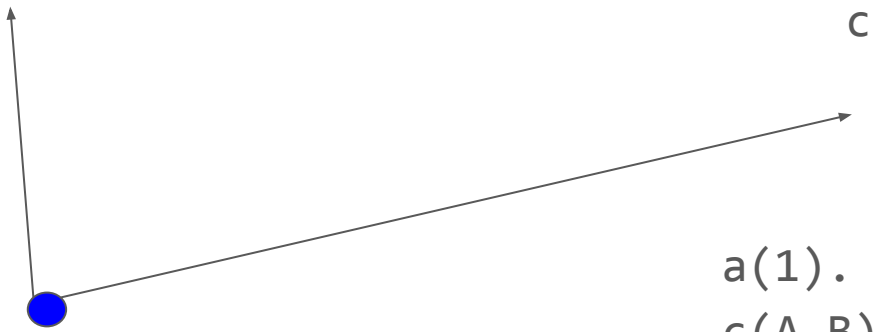
# y2011p3q8: what happens if you ask c(A,B)?

```
a(1).
a(a).
b(3).
b(a).
c(A,B) :- b(B),!,a(A).
c(X,_) :- a(X),b(X).
```

```
a(1). a(a). b(3). b(a).
c(A,B) :- b(B),!,a(A).
c(X,_) :- a(X),b(X).
```

```
c(A,B)
```

```
c(A,B) :- b(B),!,a(A).
```

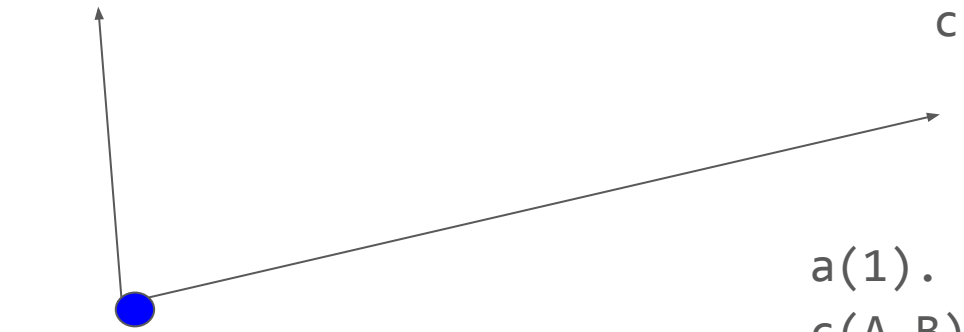


```
c(A,B)
```

```
c(X,_) :- a(X),b(X).
```

```
a(1). a(a). b(3). b(a).
c(A,B) :- b(B),!,a(A).
c(X,_) :- a(X),b(X).
```

```
c(A1,B1) :- b(B1),!,a(A1).
```

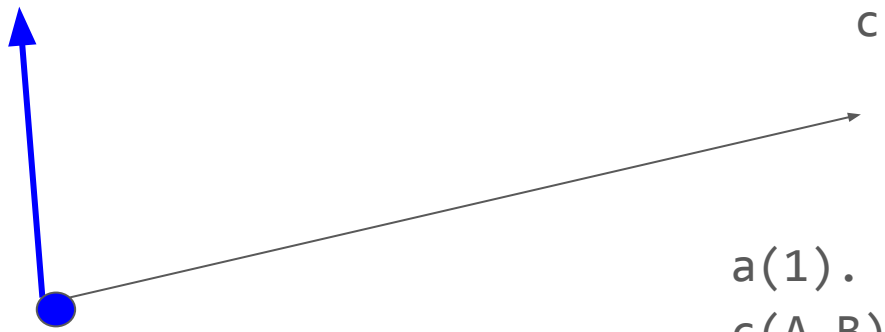


```
c(A,B)
```

```
c(X1,_) :- a(X1),b(X1).
```

```
a(1). a(a). b(3). b(a).
c(A,B) :- b(B),!,a(A).
c(X,_) :- a(X),b(X).
```

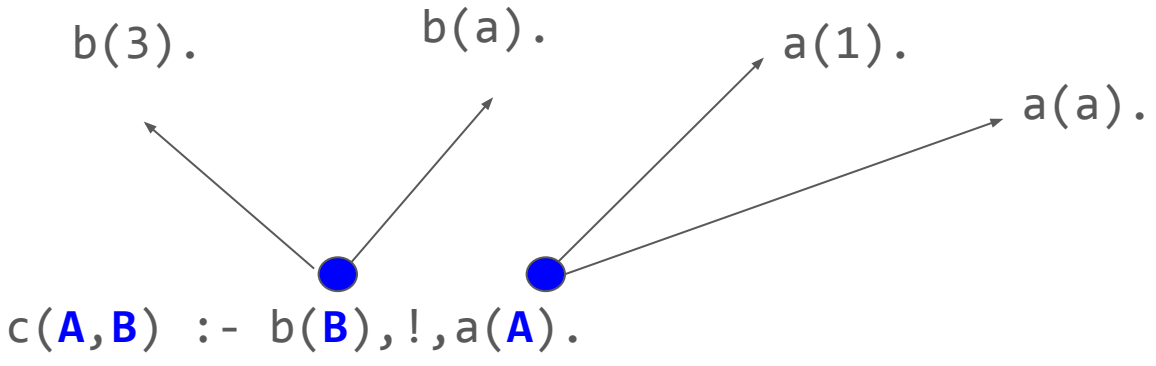
```
c(A,B) :- b(B),!,a(A).
```

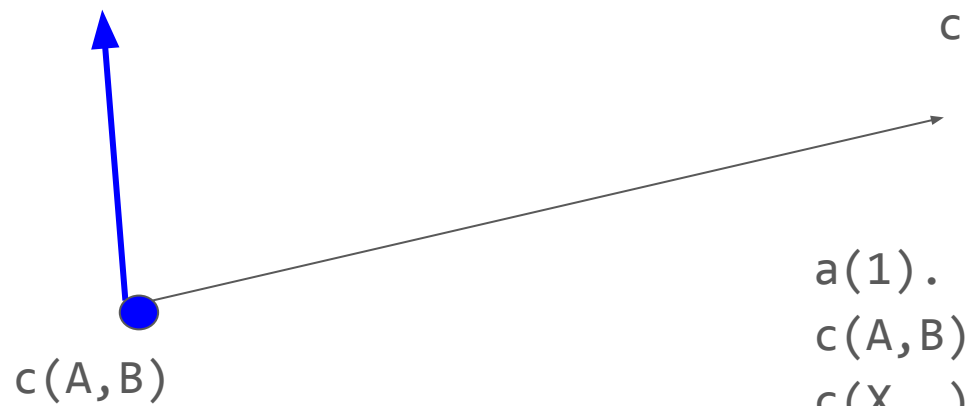


```
c(X1,_) :- a(X1),b(X1).
```

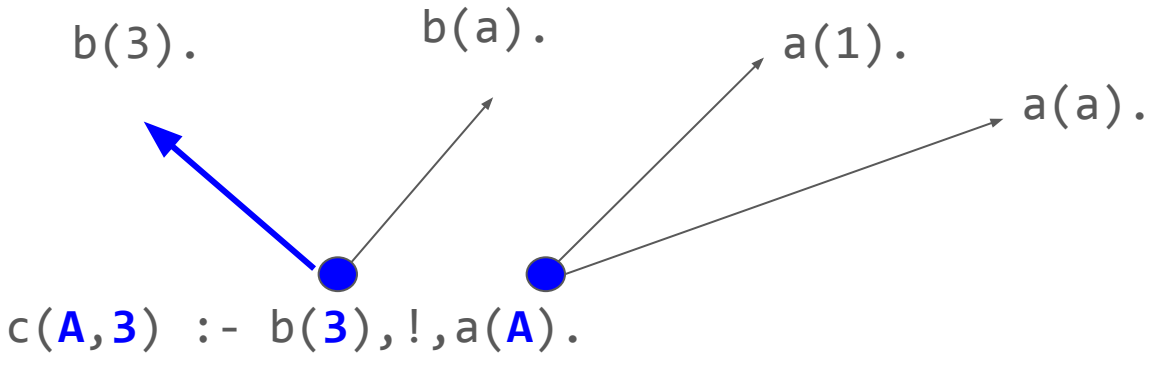
```
c(A,B)
```

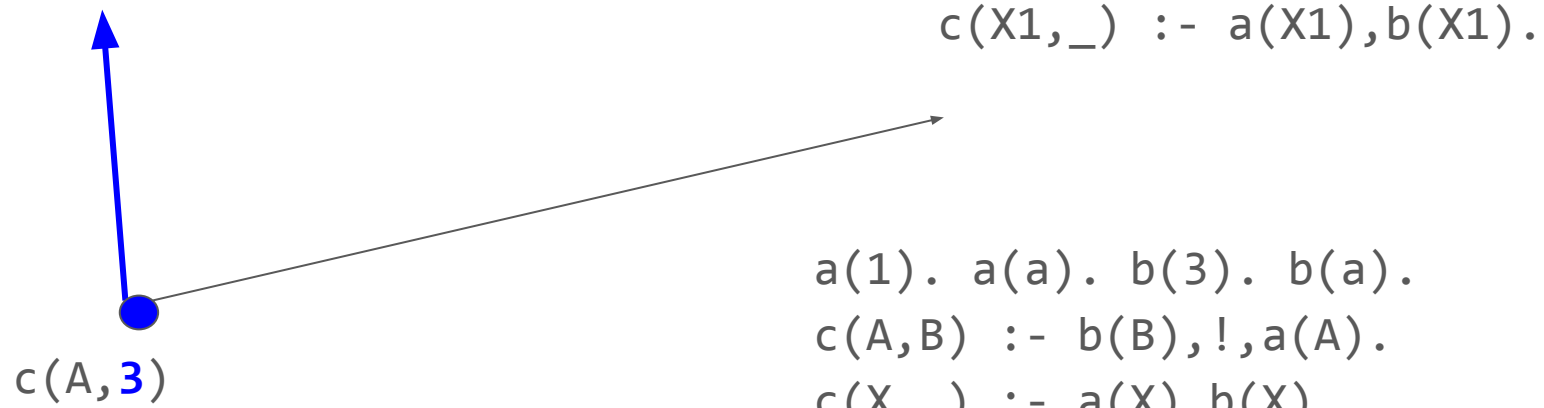
```
a(1). a(a). b(3). b(a).
c(A,B) :- b(B),!,a(A).
c(X,_) :- a(X),b(X).
```



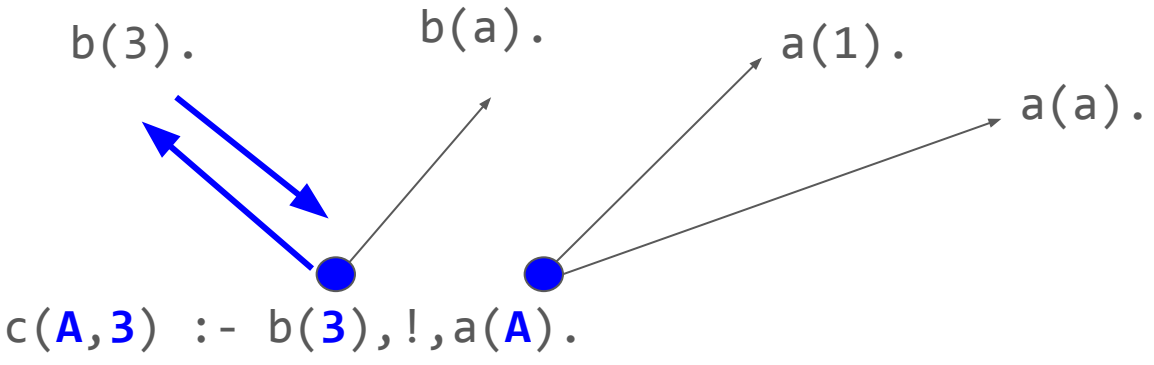



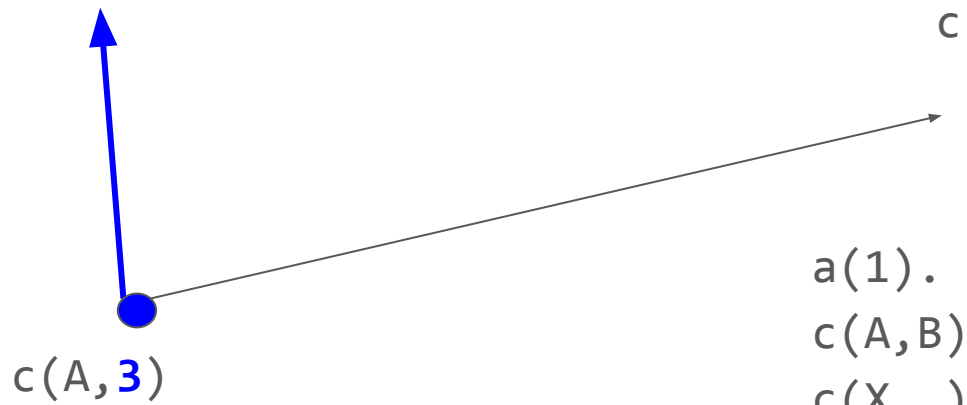

```
a(1). a(a). b(3). b(a).
c(A,B) :- b(B),!,a(A).
c(X,_) :- a(X),b(X).
```



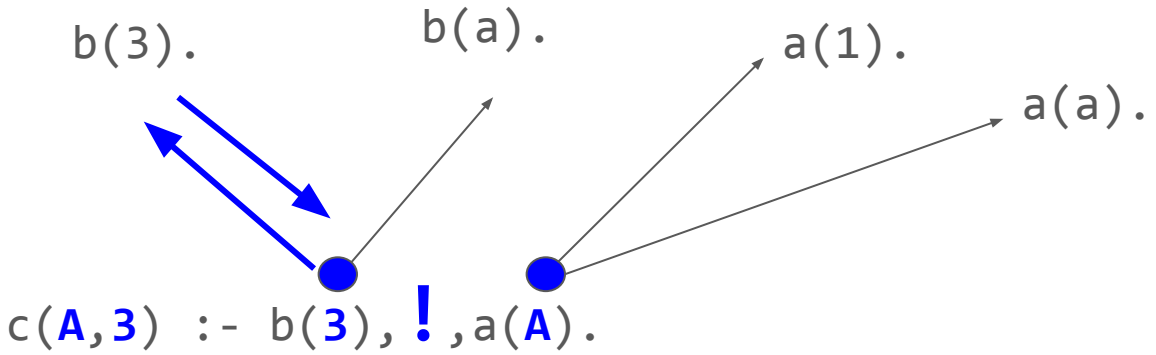



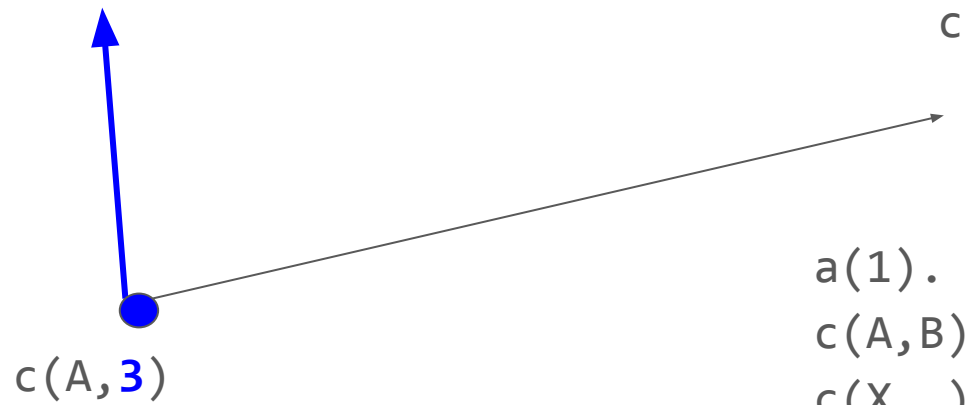

```
a(1). a(a). b(3). b(a).
c(A,B) :- b(B),!,a(A).
c(X,_) :- a(X),b(X).
```




```
a(1). a(a). b(3). b(a).
c(A,B) :- b(B),!,a(A).
c(X,_) :- a(X),b(X).
```

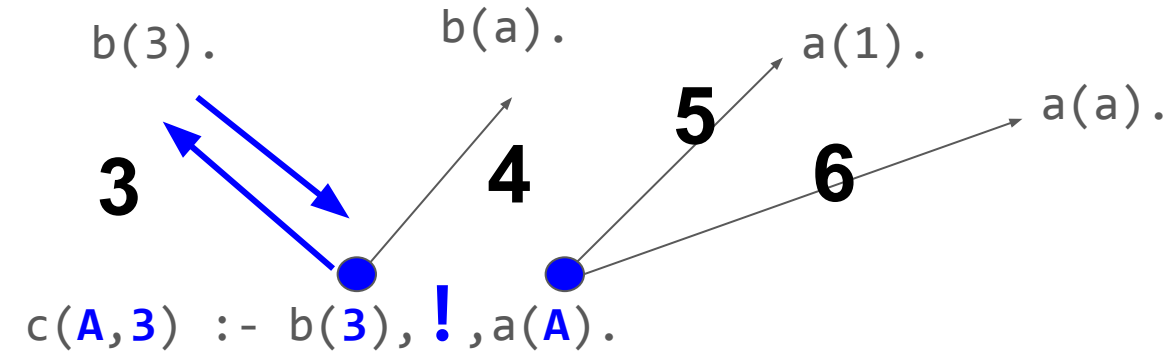


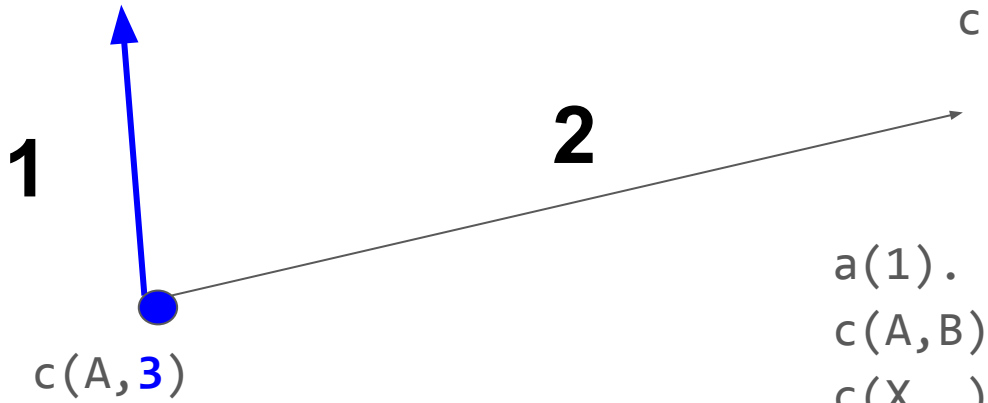

```
a(1). a(a). b(3). b(a).
c(A,B) :- b(B),!,a(A).
c(X,_) :- a(X),b(X).
```

http://etc.ch/kG4y

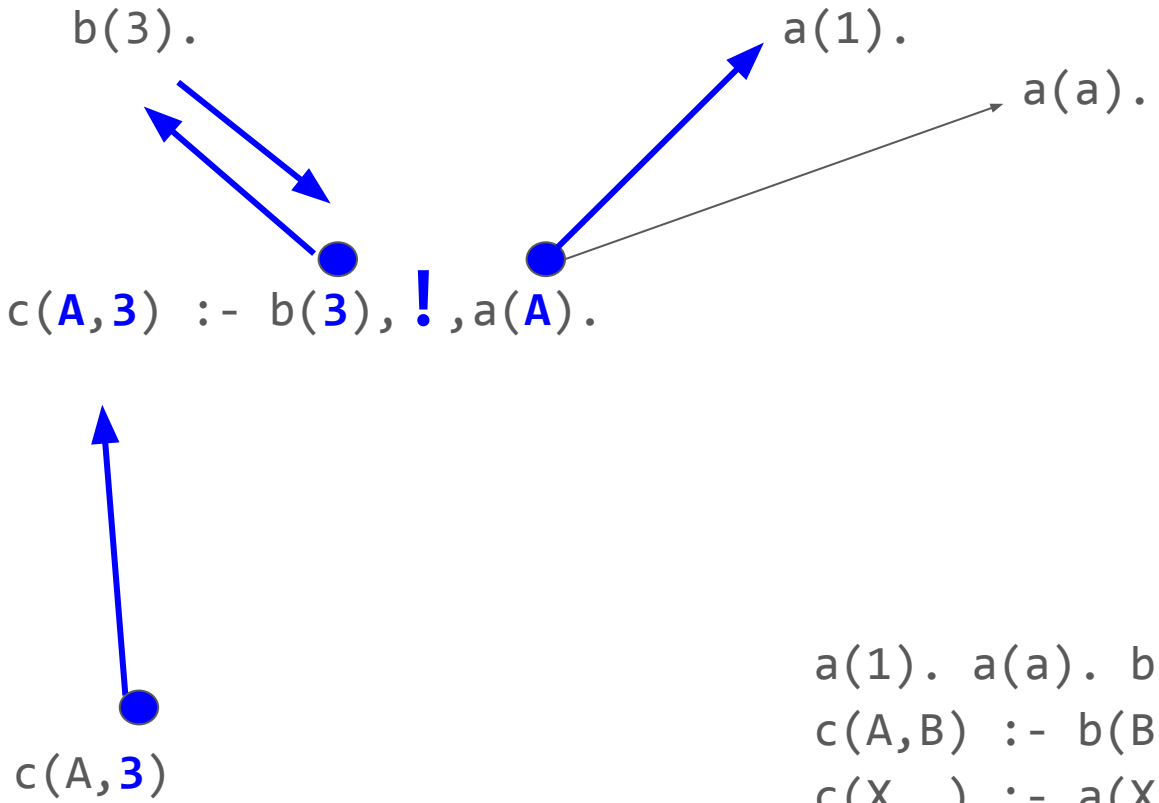
DirectPoll



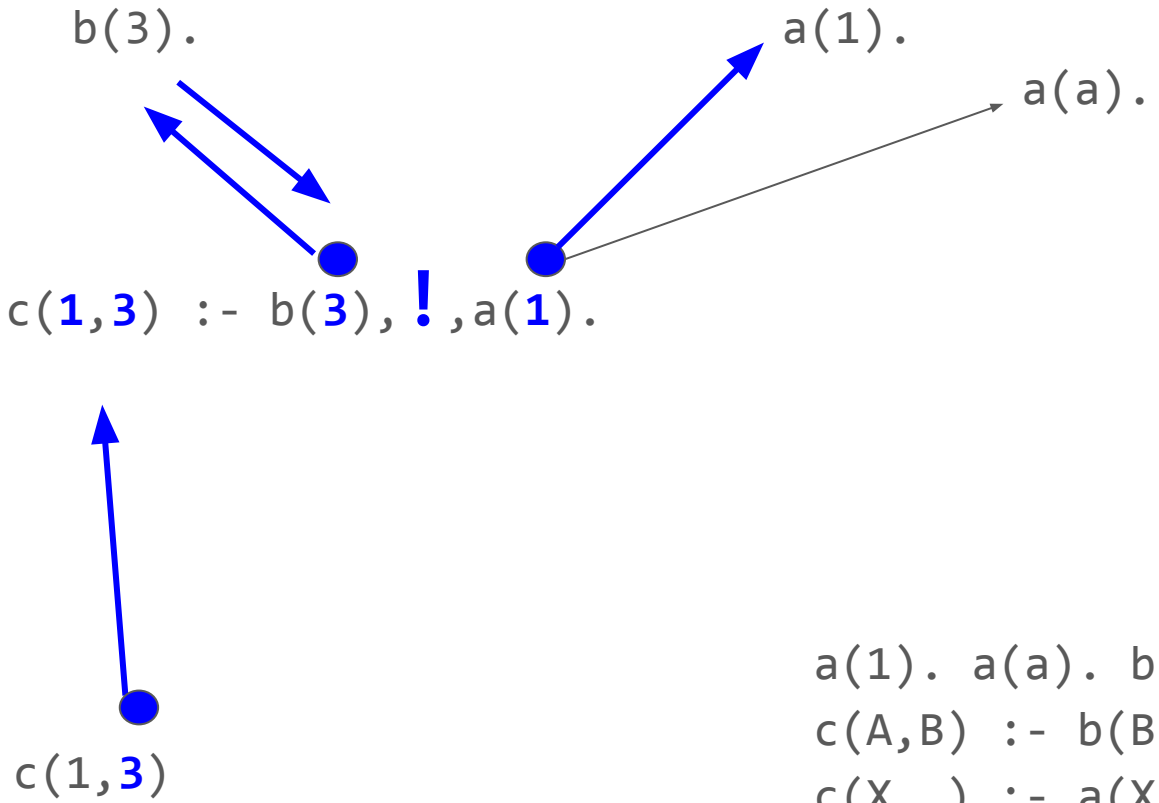

Which of these edges are removed by crossing the cut?



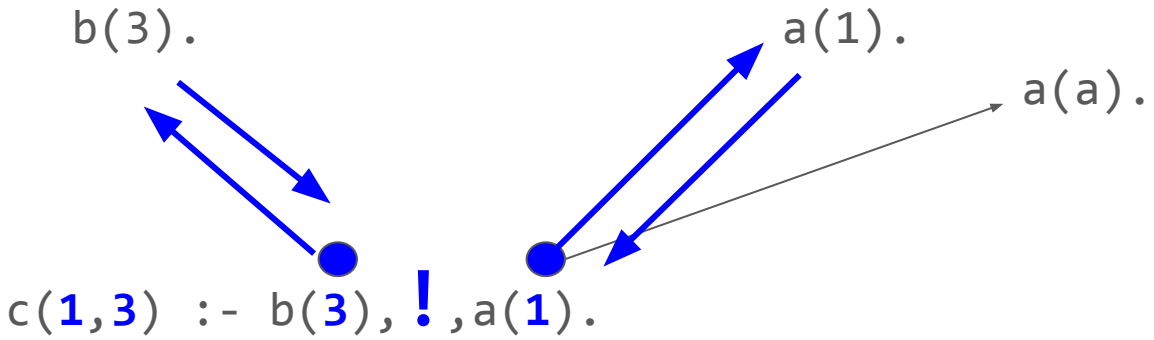

```
a(1). a(a). b(3). b(a).
c(A,B) :- b(B),!,a(A).
c(X,_) :- a(X),b(X).
```

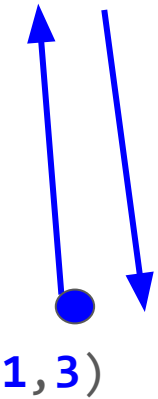



```
a(1). a(a). b(3). b(a).
c(A,B) :- b(B),!,a(A).
c(X,_) :- a(X),b(X).
```



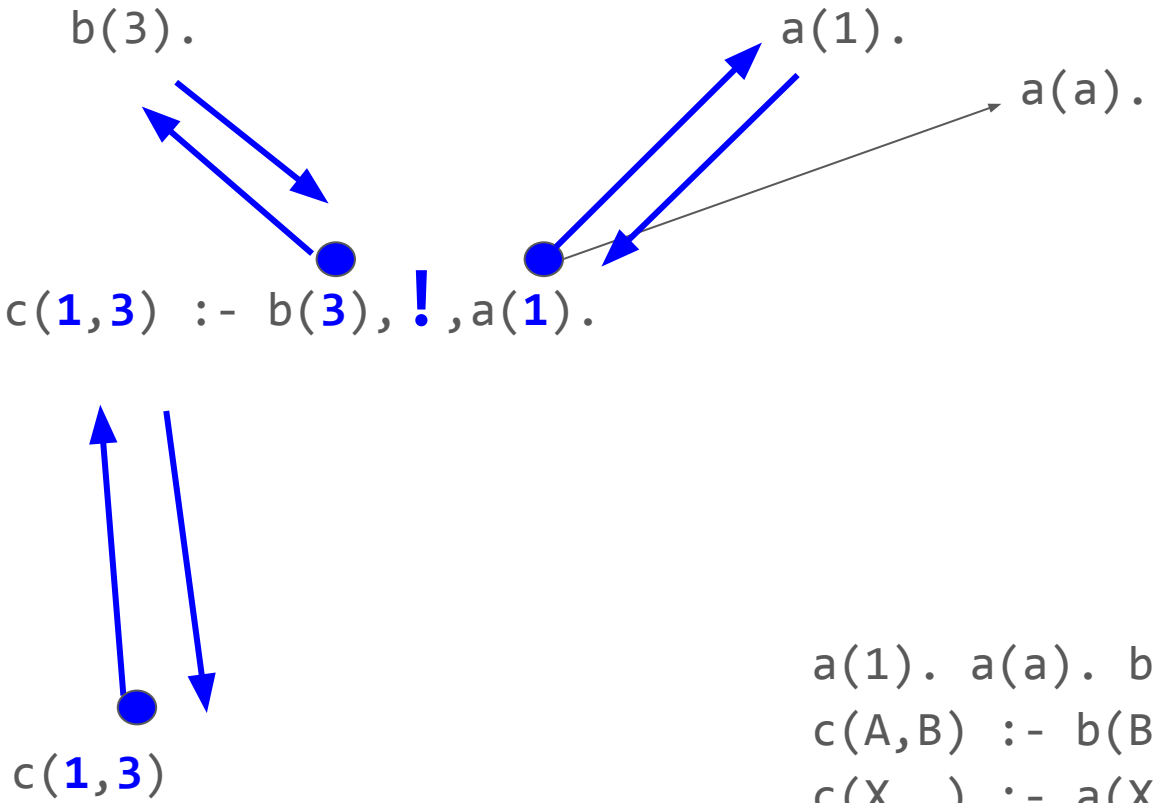

```
a(1). a(a). b(3). b(a).
c(A,B) :- b(B),!,a(A).
c(X,_) :- a(X),b(X).
```




```
a(1). a(a). b(3). b(a).
c(A,B) :- b(B),!,a(A).
c(X,_) :- a(X),b(X).
```

http://etc.ch/kG4y




What happens if we backtrack now?

```
a(1). a(a). b(3). b(a).
c(A,B) :- b(B),!,a(A).
c(X,_) :- a(X),b(X).
```

# Challenge: everything

```
a(ham).
a(eggs).
a(cheese).
a(bread).
```

Vote when finished or stuck

Write a predicate `everything(A)` which succeeds if A is a list containing all X such that a(X) is true.

http://etc.ch/8WDC

DirectPoll

# Implement not(X)

Vote when done

http://etc.ch/8WDC

# Next time

Videos

- Countdown
- Graph search

Ask questions at www.slido.com with event code E508