

Practice Problems

Disclaimer


These practice problems are for your practice alone; I make no claims of their representation on the exam, or their relevance to exam material. Any and all exercises listed herein are simply my best guess of the type of problems you'll encounter and what you should be expected to perform on a test. They may or may not represent the difficulty of any question encountered on the exam.

Note: since FOL inference was not on the midterm (as it usually is) you should also review the inference examples from the practice midterm (click here) (<http://cs.ucla.edu/~forns/classes/winter-2015/cs-161/midterm-practice.html#fol>)

Here's a smattering of problems I think might be on the exam. Have at it, and let me know if you have any questions!

Planning Problems

Example

 [BLOCKS 1] Try the following variant of the block-moving problem from the book, (page 371); determine what sequence of actions reaches the goal state from the initial state.

As a reminder, this problem has blocks resting on a table surface, or on top of one another; the goal is to stack them in a particular order.

In particular, we have the following FOL predicates:

- $\text{Block}(x)$: indicates that argument x is a block
- $\text{Clear}(x)$: indicates that argument x can have something stacked on top of it
- $\text{On}(b, x)$: indicates that argument b is stacked on top of argument x

```

; Move block b from x to y
Action(Move(b, x, y)
  Preconditions:
    On(b, x)  $\wedge$  Clear(b)  $\wedge$  Clear(y)  $\wedge$ 
    Block(b)  $\wedge$  Block(y)  $\wedge$ 
    (b  $\neq$  x)  $\wedge$  (b  $\neq$  y)  $\wedge$  (x  $\neq$  y)

  Effect:
    On(b, y)  $\wedge$  Clear(x)  $\wedge$   $\neg$ On(b, x)  $\wedge$   $\neg$ Clear(y)
)

; Move block b onto the table from x
Action(MoveToTable(b, x)
  Predonditions:
    On(b, x)  $\wedge$  Clear(b)  $\wedge$  Block(b)  $\wedge$  (b  $\neq$  x)

  Effect:
    On(b, Table)  $\wedge$  Clear(x)  $\wedge$   $\neg$ On(b, x)
)

```

Using the above actions, find an action sequence starting at the following initial state to the goal state:


```

; Constants:
A, B, C, Table

Init(
  On(A, Table)  $\wedge$  On(B, Table)  $\wedge$  On(C, A)  $\wedge$ 
  Block(A)  $\wedge$  Block(B)  $\wedge$  Block(C)  $\wedge$  Clear(B)  $\wedge$  Clear(C)
)

Goal(
  On(A, Table)  $\wedge$  On(B, A)  $\wedge$  On(C, B)
)

```

 Click for solution.

OK hotshot, that was too easy... but now that you're warmed up...

Example

☑ [BLOCKS 2] Let's try the block moving problem, but add a degree of difficulty: now, there is only room on the table to hold 3 stacks of blocks, designated table spots T1, T2, T3. We will also have to modify the actions:

```
; Move block b from x to y
; [!] NOTE: x and y may now be table slots, which
; also have Clear predicate designators if a block
; may be placed upon them
Action(Move(b, x, y)
  Preconditions:
    On(b, x) ^ Clear(b) ^ Clear(y) ^
    Block(b) ^
    (b ≠ x) ^ (b ≠ y) ^ (x ≠ y)

  Effect:
    On(b, y) ^ Clear(x) ^ ¬On(b, x) ^ ¬Clear(y)
)
```

Using the above action and initial state:

```
; Constants:
A, B, C, D, E, T1, T2, T3

Init(
  On(A, T1) ^ On(E, A) ^ On(C, T2) ^ On(B, C) ^ On(D, B) ^
  Block(A) ^ Block(B) ^ Block(C) ^ Block(D) ^ Block(E) ^
  Clear(D) ^ Clear(E) ^ Clear(T3)
)

Goal(
  On(A, T3) ^ On(B, A) ^ On(C, B) ^ On(D, C) ^ On(E, D)
)
```

🔗 Click for solution.

Alright that was a little obnoxious and grueling... so why not aim for more obnoxious and grueling with a planning graph?

Your book starts the planning graph for the Tire Changing problem, but doesn't show some mutex connections... why don't we complete them?

Example

☑ Follow the planning graph for the tire changing problem up to State level S1:

```
; Constants:
Flat, Spare, Axle, Trunk, Ground

Init(Tire(Flat) ^ Tire(Spare) ^ At(Flat, Axle) ^ At(Spare, Trunk))
Goal(At(Spare, Axle))

Action(Remove(obj, loc),
  Preconditions:
    At(obj, loc)
  Effect:
    ¬At(obj, loc) ^ At(obj, Ground)
)

Action(PutOn(t, Axle),
  Preconditions:
    Tire(t) ^ At(t, Ground) ^ ¬At(Flat, Axle)
  Effect:
    ¬At(t, Ground) ^ At(t, Axle)
)

Action(LeaveOvernight(),
  Preconditions:
    Effect: ¬At(Spare, Ground) ^ ¬At(Spare, Axle) ^ ¬At(Spare, Trunk) ^
           ¬At(Flat, Ground) ^ ¬At(Flat, Axle) ^ ¬At(Flat, Trunk)
)
```

Example

☑ [TIRE 1] Step One: diagram the planning graph's initial state (HINT: remember to explicitly list the closed world assumptions!)

⚠ NOTE: For simplicity's sake, we will not model the Tire(x) nor the ¬At(Flat, Trunk) fluents, since those have no chance of changing.

❓ Click for solution.

Example

✔ [TIRE 2] Step Two: diagram the planning graph's first Action level, A0 (without mutex links, for now):

🔗 Click for solution.

Example

✔ [TIRE 3] Step Three: add mutex links between actions in A0:

🔗 Click for solution.

Example

✔ [TIRE 4] Step Four: diagram the second state level S1:

🔗 Click for solution.

Example

✔ [TIRE 5] Step Five: add mutex links between fluents of state S1:

🔗 Click for solution.

How about a couple of nice, easy, conceptual questions after that?

❓ Under what condition does the planning graph stop generating new levels?

❓ If the planning graph "levels off," the goal state's fluents exist in the final state level, and they are not mutex, are we guaranteed that a solution exists to the planning problem?

Example

✓ [REGRESSION 1] Use regression planning to derive an action sequence that will attain the given goal state from the given initial state.

⚠ **Remember!** For states g and actions a

$$g' = (g - ADD(a)) \cup Preconditions(a)$$

```

; Constants:
Andrew, Dyer, Charlotte

; Predicates:
X(arg), Y(arg), Z(arg)

; Actions:
Action(A(arg1, arg2, arg3)
  Pre:
     $\neg X(\text{arg1}) \wedge Y(\text{arg2}) \wedge Z(\text{arg3})$ 
  Effect:
    Z(arg1)
)

Action(B(arg1)
  Pre:
    Z(arg1)  $\wedge$  X(arg1)
  Effect:
     $\neg Z(\text{arg1}) \wedge Y(\text{arg1})$ 
)

Action(C(arg1)
  Pre:
     $\neg X(\text{arg1})$ 
  Effect:
    X(arg1)
)

; Initial state:
Y(Andrew)  $\wedge$  Z(Charlotte)

; Goal:
Y(Dyer)

```

You might find it convenient to fill out the following table to track your solution steps:

g	Action	g'	Bindings Θ
(e.g.) \emptyset	(e.g.) Test(Dyer)	(e.g.) $Y(\text{Andrew}) \wedge X(\text{Charlotte})$	(e.g.) $\{\text{arg1} / \text{Dyer}\}$

 Click for solution

Probability Calculus

Get your theorems and rules ready kids, it's probability time!

Why don't we start with a cheat sheet on stuff you're going to need:

Product Rule:

$$\begin{aligned} Pr(A, B) &= Pr(A|B) * Pr(B) \\ &= Pr(B|A) * Pr(A) \end{aligned}$$

Chain Rule (Recursive application of Product Rule):

$$Pr(A, B, C) = Pr(A|B, C) * Pr(B|C) * Pr(C)$$

Law of Total Probability

$$Pr(A) = \sum_{\forall b \in B} Pr(A|b) * Pr(b)$$

Bayes' Conditioning

$$Pr(A|B) = \frac{Pr(A, B)}{Pr(B)}$$

Bayes' Rule

$$Pr(A|B) = \frac{Pr(B|A) * Pr(A)}{Pr(B)}$$

🔗 [PROB1] If the product rule tells us that

$$Pr(A, B) = Pr(A|B) * Pr(B)$$

...then in the general case accounting for evidence E, can we say that

$$Pr(A, B|E) \stackrel{?}{=} Pr(A|B, E) * Pr(B|E)$$

2 [PROB2] Prove the **generalized Bayes Rule** stating:

$$Pr(A|B, E) = \frac{Pr(B|A, E) * Pr(A|E)}{Pr(B|E)}$$

2 [PROB3] Assuming that:

$$C \perp A$$

$$A \perp\!\!\!\perp B|C$$

prove that

$$Pr(A, B, C) = Pr(A) * Pr(B, C)$$

Example

✓ Read the following problem description, and for each probability mentioned, formalize it into a $Pr(x | y)$ statement (for example) and then solve for the correct quantity. Assume two binary variables: Test and Disease, as described by the problem.

A rare disease is found in about 2% of those tested for it. The test returns an end result that is either positive (Test) or negative (\neg Test). The problem is that the tests are not perfect such that 3% of people without the disease will test positive and 1% of people with the disease will test negative.

If a patient tests negative, what is the probability that they **DO NOT have the disease?**

2 Click for solution.

Example

✓ Use the following joint probability distribution to compute the requested quantities.

X	Y	Z	Pr(X, Y, Z)
0	0	0	0.10
0	0	1	0.12
0	1	0	0.25
0	1	1	0.15
1	0	0	0.18
1	0	1	0.05
1	1	0	0.13
1	1	1	0.02

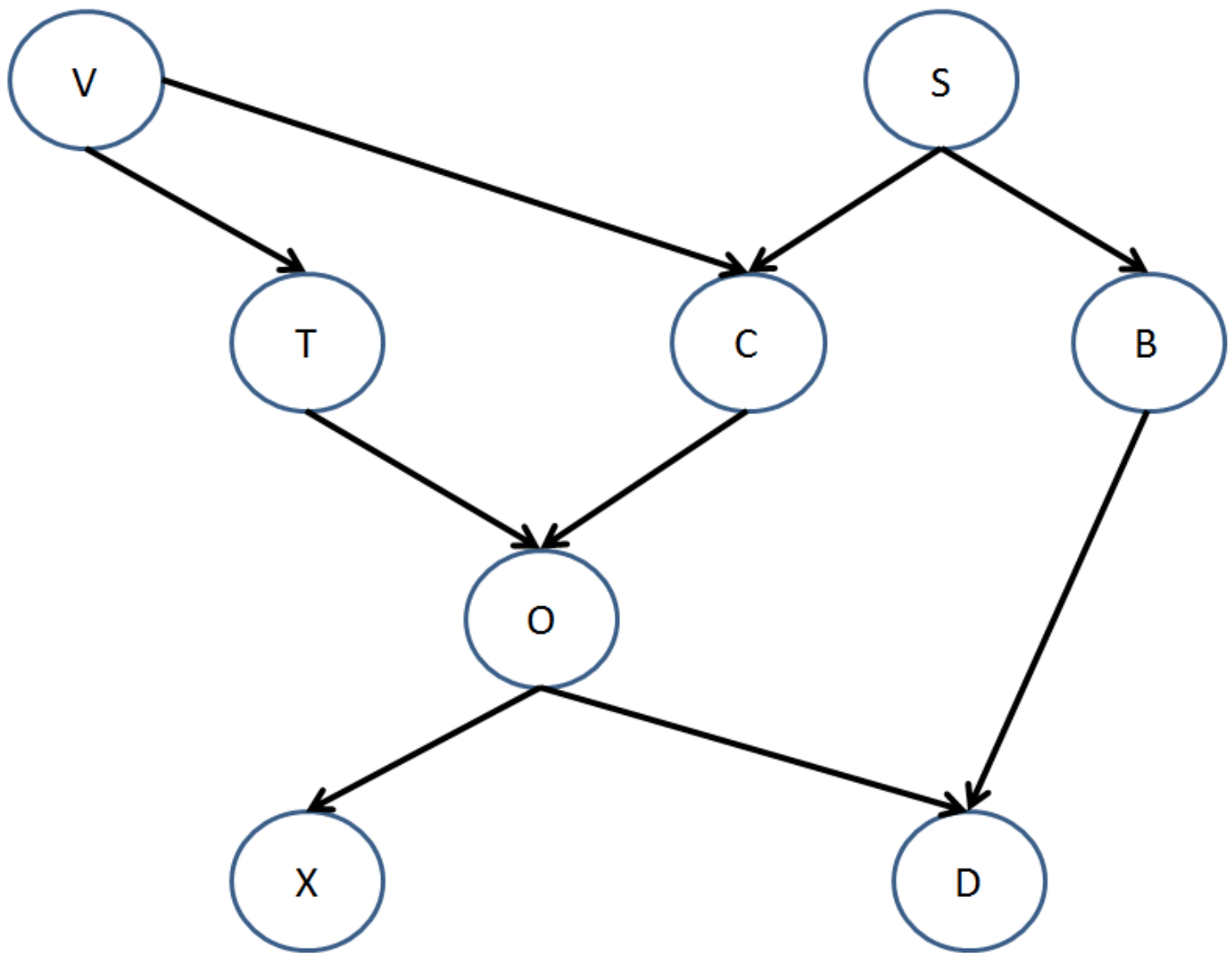
❓ What is $\Pr(X = 1)$?

❓ What is $\Pr(X = 0, Y = 0)$?

❓ What is $\Pr(Z = 0, Y = 1 \mid X = 0)$?

Bayesian Networks

Observe the structure of the following Bayesian Network and use it to answer the following questions about independence relations.

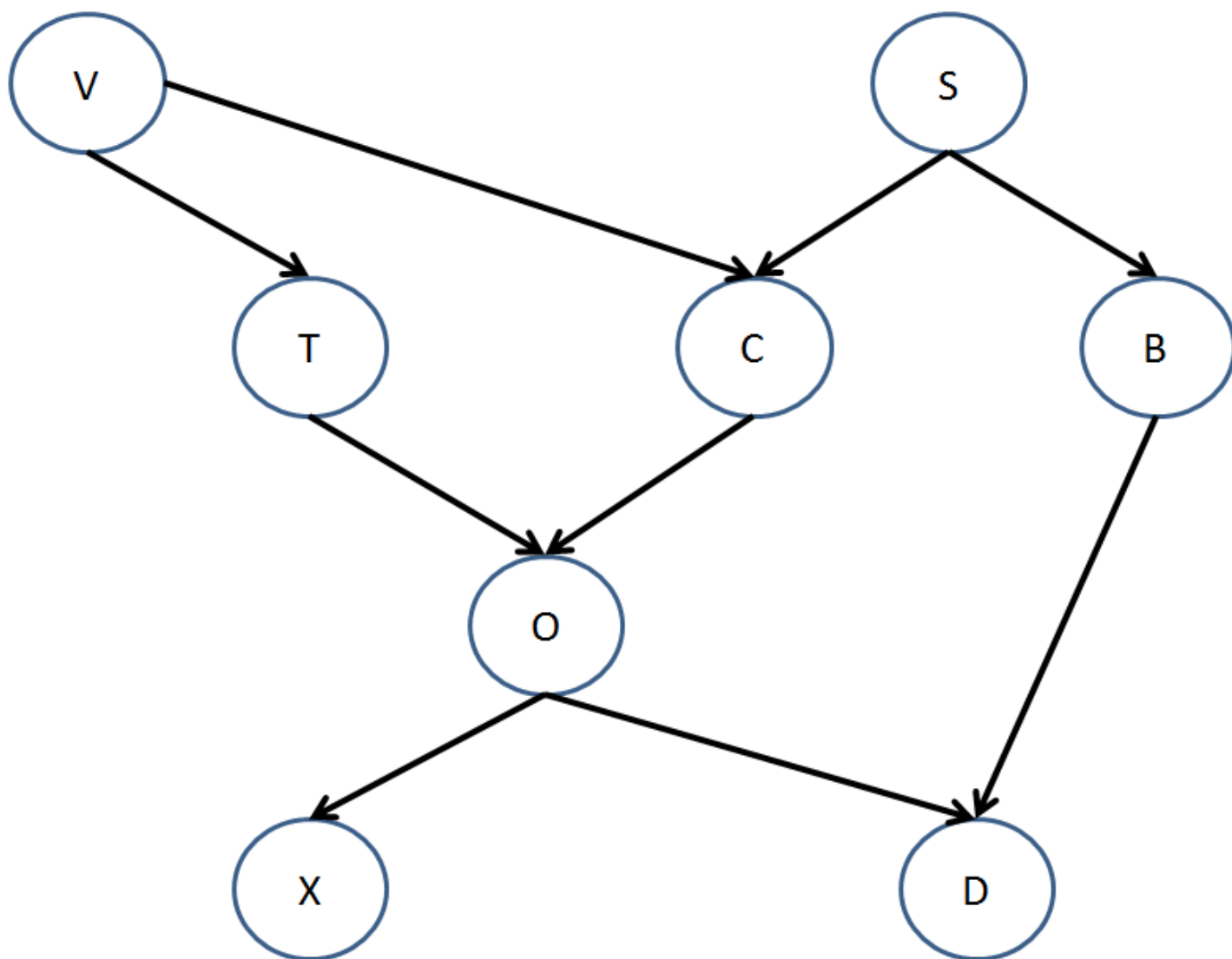
**Example**

☑ [BN 1] List the Markovian assumptions implicit within the network.

🔍 Click for solution.

Example

☑ [BN 2] Use the same Bayesian Network (replicated below for your scrolling convenience) to answer the following d-separation questions.



X	Z	Y	d-sep(X, Z, Y)?
V	\emptyset	S	???
V	D	S	???

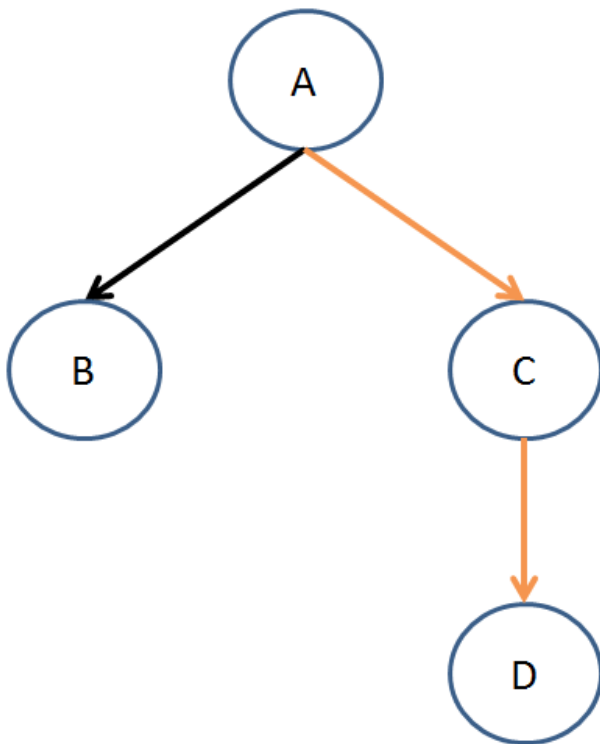
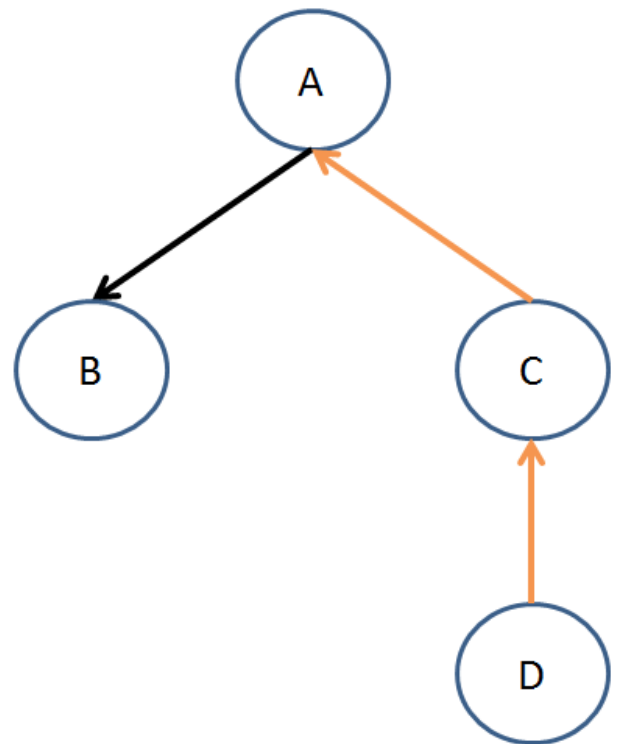
X	Z	Y	d-sep(X, Z, Y)?
T	???	B	Yes!
T	C, S	B	???
X	C	B	???
V, B	T, C	X	???
V	T, C, ???	O	No!

🔗 Click for solution.

📌 Two Bayesian Networks are said to be **observationally equivalent** if they exhibit the same independence and conditional independence relationships between variables.

Example

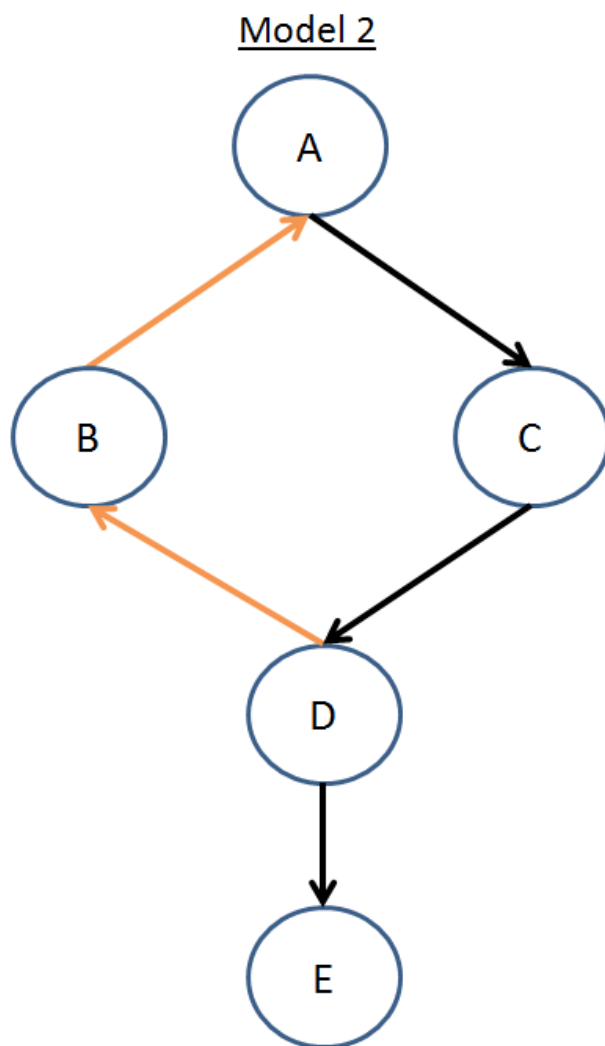
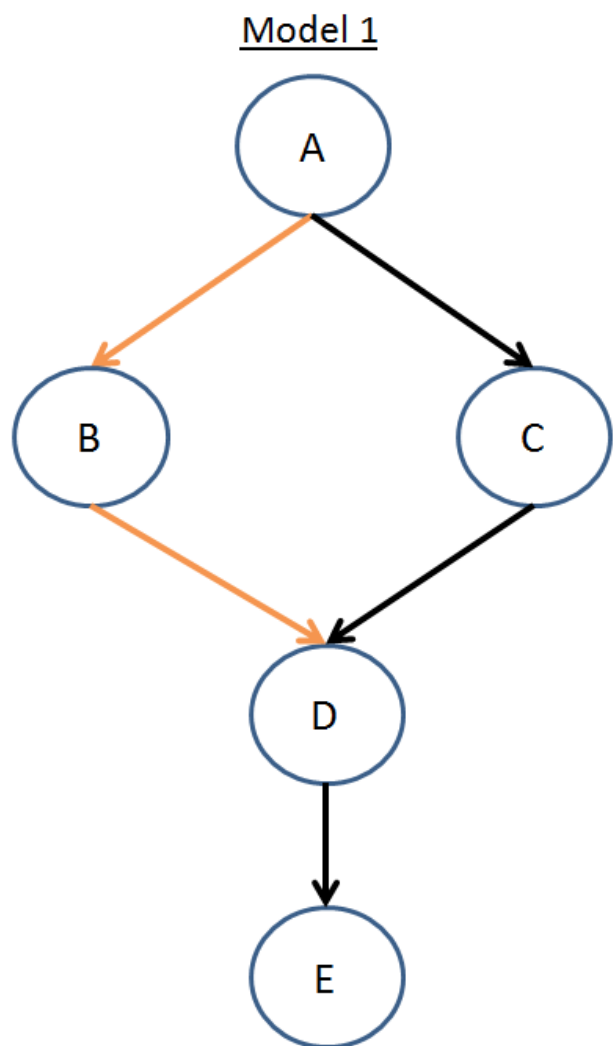
🔗 [BN 3] Are the following two models observationally equivalent?

Model 1Model 2

🔍 Click for solution.

Example

🔍 [BN 4] Are the following two models observationally equivalent?

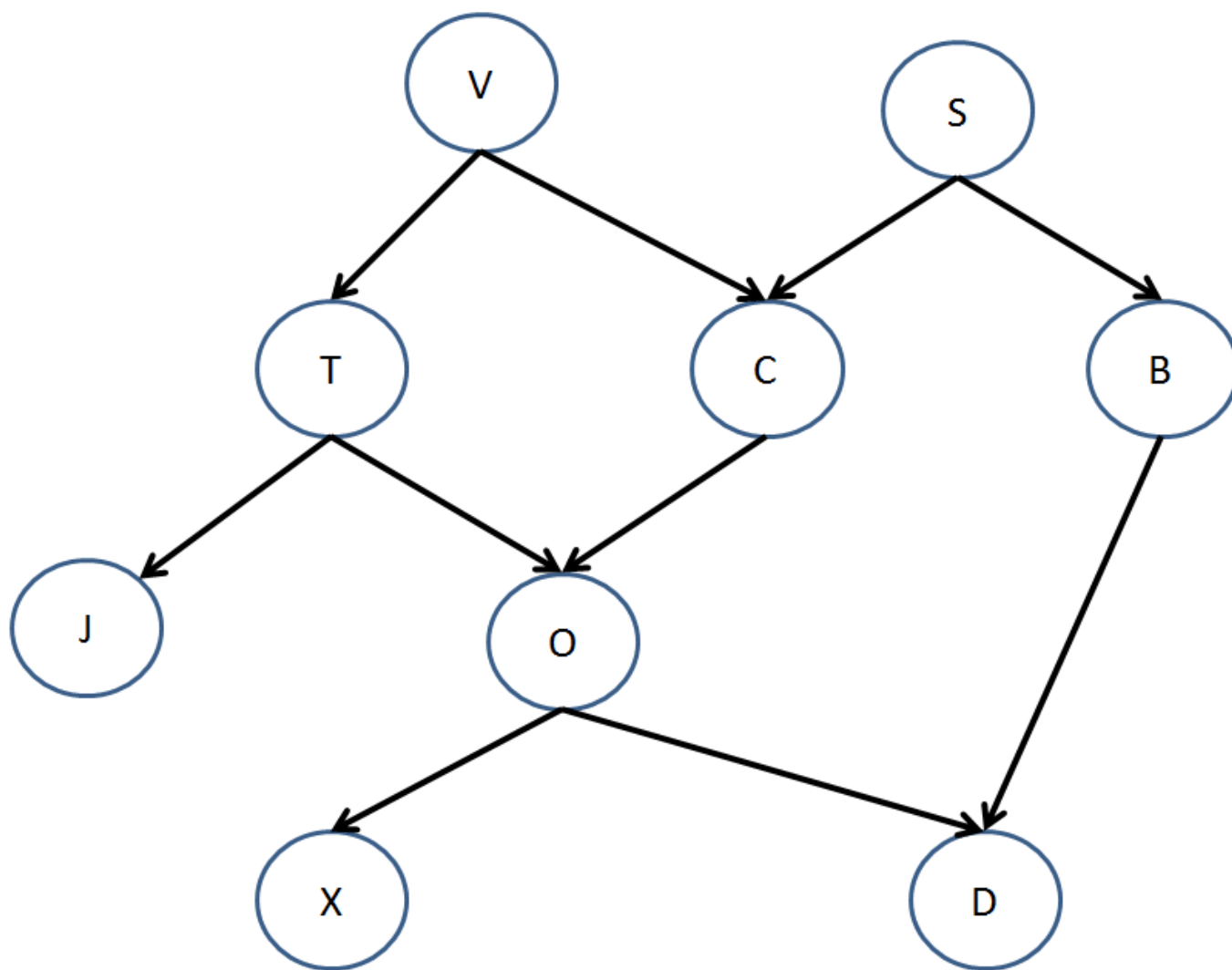


Click for solution.

i A **Markov Blanket** for a node in a Bayesian Network (indicated $MB(A)$ for node A) is the set of all nodes that, if given, render A independent from ALL other nodes in the network. A Markov Blanket consists of a node's parents, children, and spouses.

Example

Observe the following Bayesian Network and answer the following questions about Markov Blankets:



🔍 [BN 5] What is the Markov Blanket for Node C?

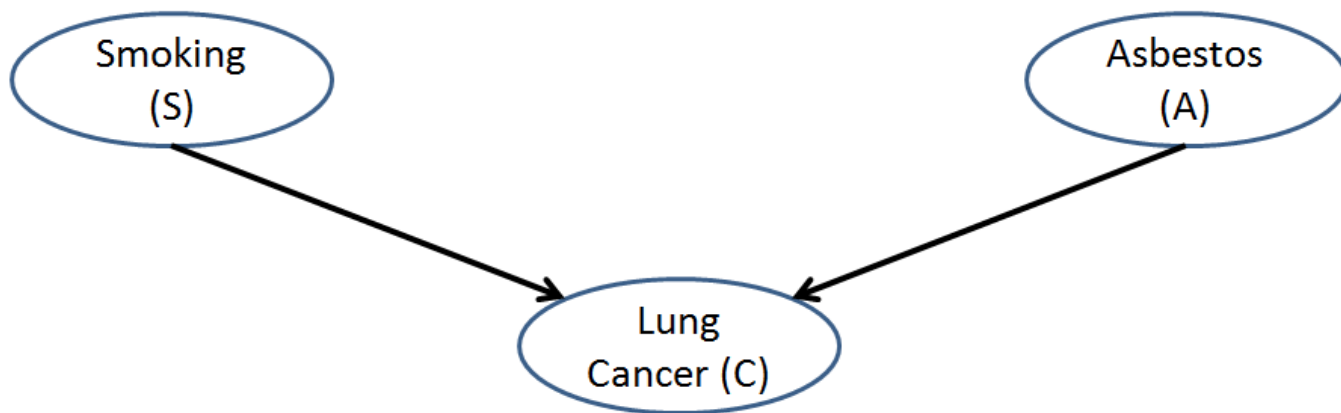
🔍 [BN 6] What is the Markov Blanket for Node O?

Example

🔍 Compute $\Pr(S = 1 \mid A = 1)$ using the Bayesian Network and CPTs below.

S	Pr(S)
0	0.7
1	0.3

A	Pr(A)
0	0.9
1	0.1



🔗 Click for solution.

Example

🔗 Compute $\Pr(A = 1 \mid B = 0)$. DO NOT construct the entire joint distribution in your solution.

As a reminder, here are some "at a glance" steps to enumeration inference:

1. Identify your query (Q), evidence (e), and hidden variables (V) amongst all variables in the network (X).
2. Formulate your query expression in terms of the network CPTs, by the formula:

$$\Pr(Q|e) = \alpha \sum_{v \in V} \Pr(Q, e, v)$$

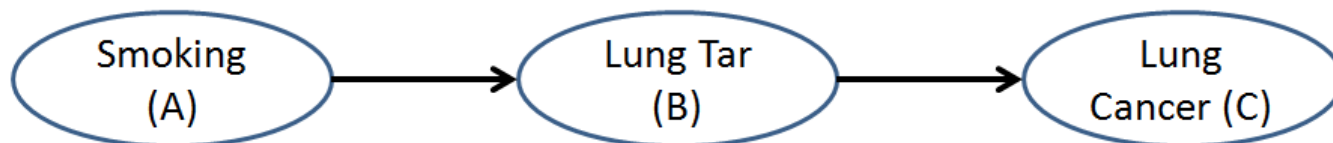
3. [Optional] (But time saving) Re-arrange summations and factors to reduce number of multiplications (like when we pulled out $\Pr(A)$ above)
4. Solve for

$$\alpha = 1/\Pr(e)$$

and substitute into (2) above!

A	Pr(A)
0	0.4
1	0.6

B	C	Pr(C B)
0	0	0.5
0	1	0.5
1	0	0.7
1	1	0.3



A	B	Pr(B A)
0	0	0.8
0	1	0.2
1	0	0.1
1	1	0.9

🔗 Click for solution (using Enumeration Inference).

🔗 Click for solution (using Elimination Inference).

Whew! That one was intricate! But now that you know what you're doing... care to try it again? :)

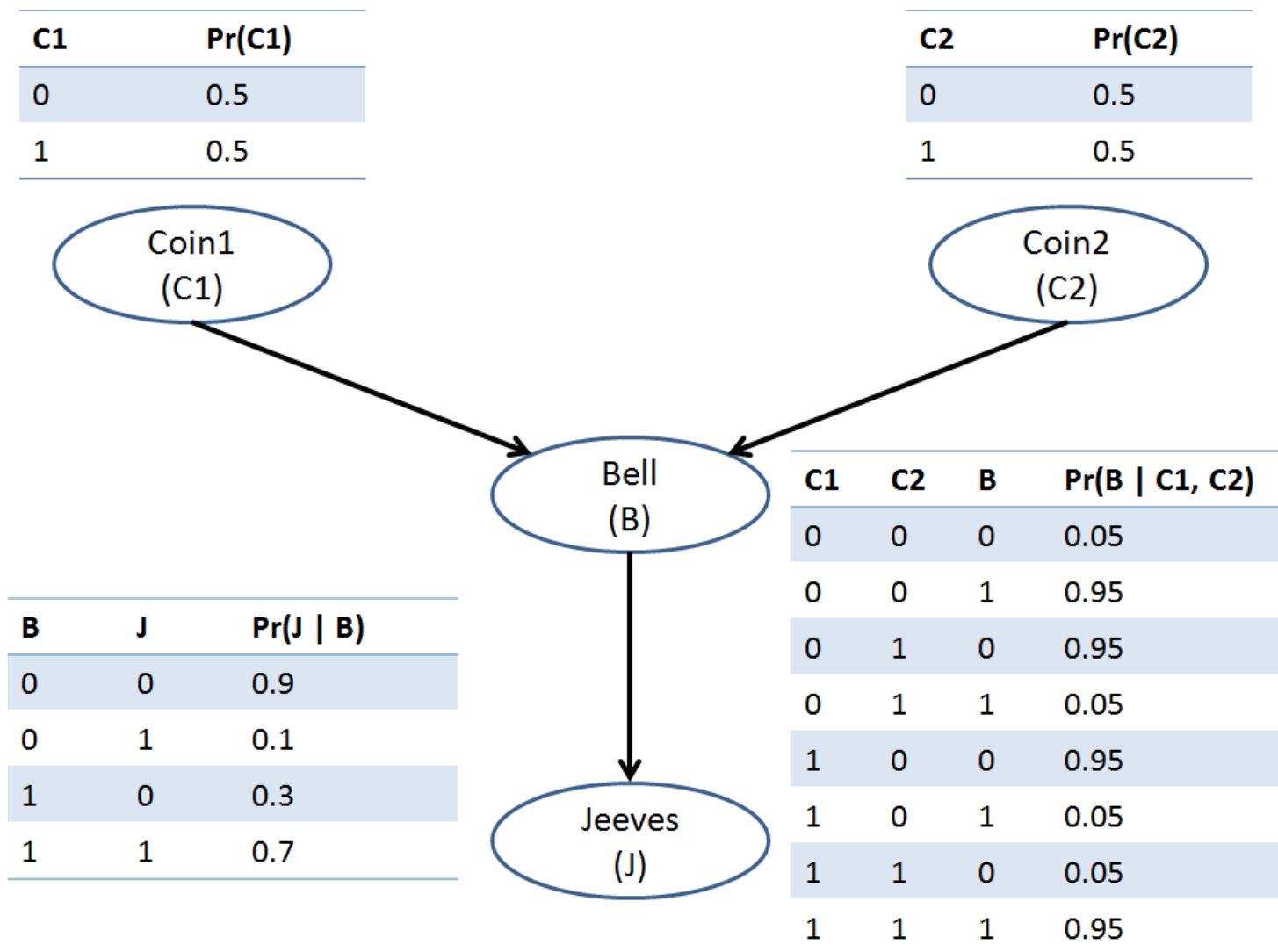
(by the way, recall that Elimination Inference and Enumeration Inference will yield the same answer -- their procedures are just slightly different).

Example

🔗 Read the following problem description and then compute the following quantity: $\Pr(C_1 = 1 \mid J = 1, C_2 = 1)$

Remember our example about a bell ringing whenever 2 coins both came up the same (both heads or both tails)? Well, let's introduce a little noise into the system: Now, whenever both coin flips are the same, there is a 95% chance that the bell will ring, and a 5% chance that it will ring under all other scenarios. Furthermore, our butler, Jeeves, who is still hard of hearing, will come serve us tea 70% of the time the bell rings, and 10% of the time that it does not ("Was that the bell I heard?! Maybe..."). This scenario is represented by the Bayesian network and CPTs below.

Given that Jeeves did indeed bring us tea ($J = 1$) and coin 2 came up heads ($C2 = 1$), what is the probability that coin 1 also came up heads?



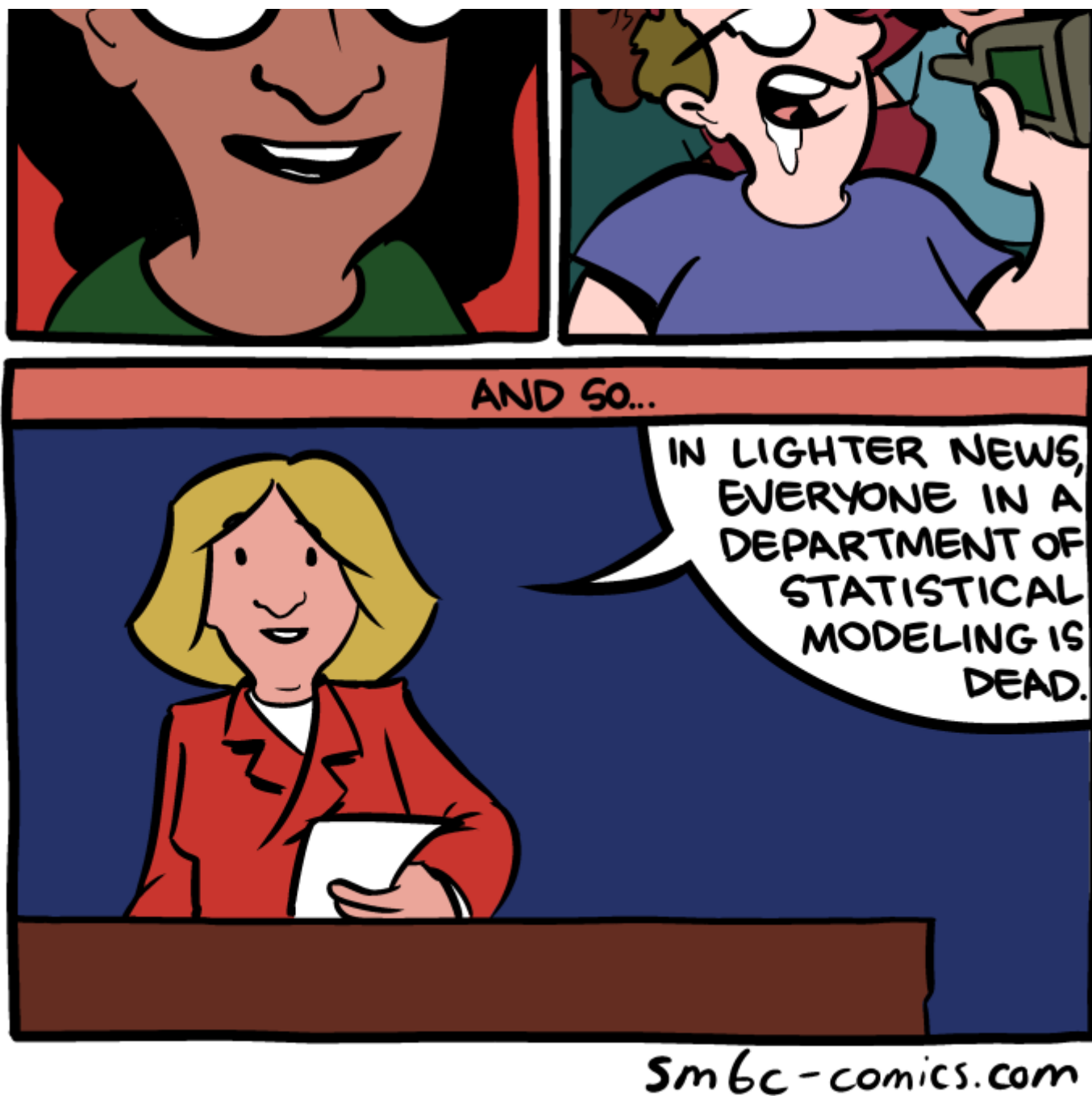
🔗 Click for solution (using Elimination Inference).

If you weren't able to do that one, don't fret too much... I'm banking it's harder than what you'll see on the final but who knows!

If you did get it, then high five! (yourself, since I'm not there, but picture that I am)

For your hard work so far, here's something Bayesian related from Saturday Morning Breakfast Cereal:





(<http://www.smbc-comics.com/?id=3366#comic>)

Decision Theory

Example

☑ For each of the following sets of preferences and utility functions, determine if all preference axioms are satisfied and that the utilities abide by the preference ordering.

; Example 1:
; Preferences:

A > B
B > D
E > C
D > C
B ~ C

🔗 Click for solution.

; Example 2:
; Preferences & Utilities:

A > B
B ≥ D
E > C
B ~ C

U(A) = 100
U(B) = 50
U(C) = 50
U(D) = 50
U(E) = 75

🔗 Click for solution.

Example

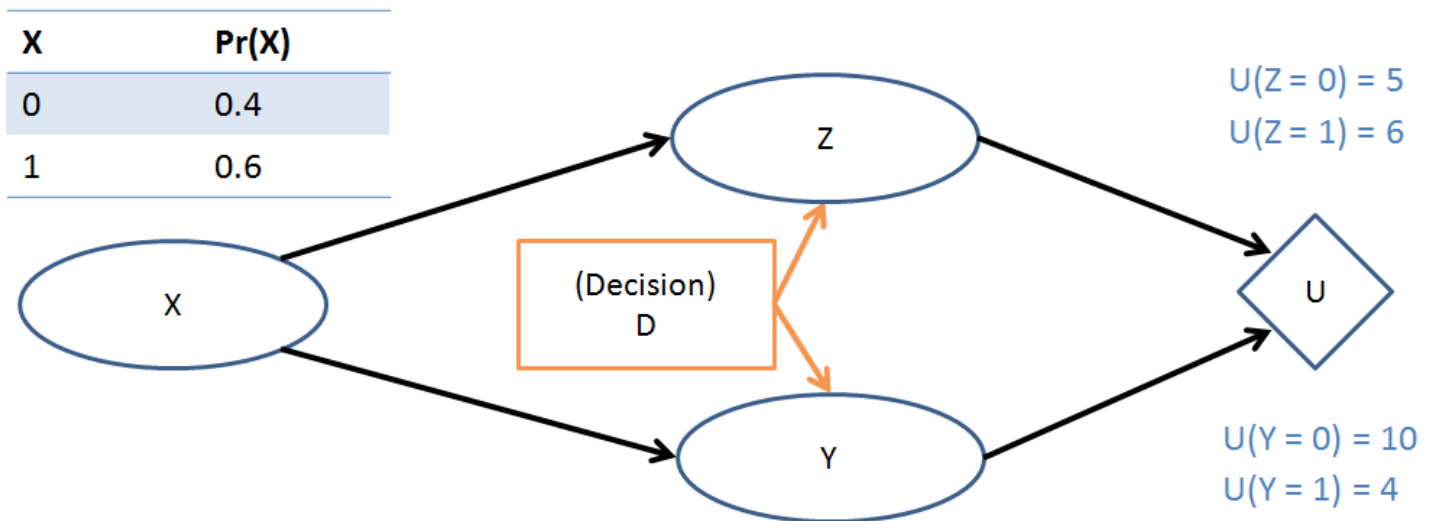
☑ Compute the expected utility of actions 1 and 2 below; which has the maximum?

```
; Action 1 corresponds to L1:  
L1 = [0.01, S1; 0.50, S2; 0.25, L3; 0.24, L4]  
  
; Action 2 corresponds to L2:  
L2 = [0.25, S2; 0.4, L4; 0.35, S3]  
  
L3 = [0.3, S4; 0.7, S5]  
L4 = [0.5, S2; 0.5, S3]  
  
; Utilities:  
U(S1) = 100  
U(S2) = 10  
U(S3) = 5  
U(S4) = 3  
U(S5) = 1
```

🔗 Click for solution.

Example

✔ Use the following decision network to determine which action (of two choices A1 and A2 for choice node D) is the most preferable **given that X is observed to be 1**. Assume the scoring function is mere addition between (probability) weighted utility values.



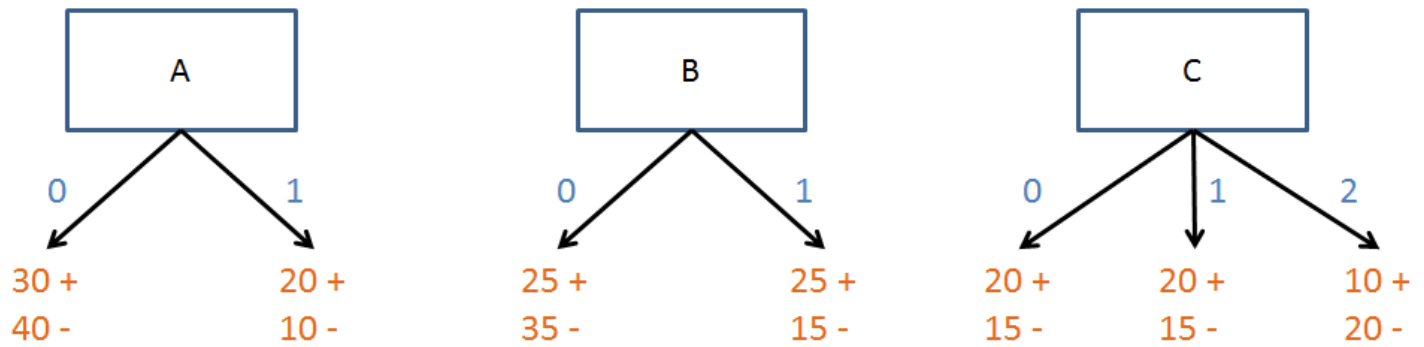
D	X	Z	Pr(Z X, D)
A1	0	0	0.1
A1	0	1	0.9
A1	1	0	0.7
A1	1	1	0.3
A2	0	0	0.5
A2	0	1	0.5
A2	1	0	0.4
A2	1	1	0.6

D	X	Y	Pr(Y X, D)
A1	0	0	0.5
A1	0	1	0.5
A1	1	0	0.45
A1	1	1	0.55
A2	0	0	0.25
A2	0	1	0.75
A2	1	0	0.2
A2	1	1	0.8

Click for solution.

Example

Use the following decision tree single-node splits to determine which has the least expected entropy for classification X.



N = 100	A = 0	A = 1	B = 0	B = 1	C = 0	C = 1	C = 2
X = yes	30	20	25	25	20	20	10
X = no	40	10	35	15	15	15	20
Totals	70	30	60	40	35	35	30

Click for solution.

Example

☑ Determine whether or not the following patient who complains of weight loss, a cough, and NO headache might have lung cancer (which is more likely? L or -L?)