6.891 Review Problems

5/16/01

1 Bayes' Nets

I am a professor trying to predict the performance of my class on an exam. After much thought, it is apparent that the students who do well are those that studied and do not have a headache when they take the exam. My vast medical knowledge leads me to believe that headaches can only be caused by being tired or by having the flu. Studying, the flu, and being tired are pairwise independent.

a) Model 3 best models the relationship.

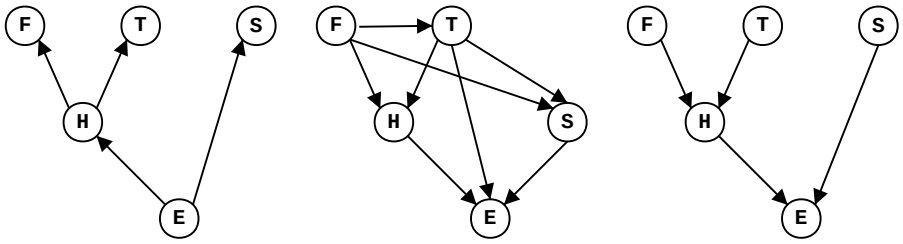
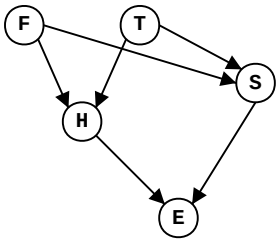


Figure 1: From left to right, models 1, 2, and 3

b) The first model makes flu, tiredness, and studying only conditionally independent (the first two conditioned on H and the third conditioned on E). The second model has the right relations, but many unnecessary dependencies. If the situation is accurately described in the problem, it will produce the same joint distribution as the third model because the dependencies will have no effect (for example, $P(S|F,T)$ will be the same as $P(S)$).

c) If we assume that we need to hold only one value, $P(X = true)$ to represent the apriori probability of a single variable, then we need 2^n entries for a conditional probability table for a node with n parents. So, the original network has a complexity of $1+1+1+4+4 = 11$ and the new network has a complexity of $1+1+4+4+4 = 14$. So, the size of the information

necessary to store the network has increased by about 27%, but we have gained only a little more accuracy.



d)

$$\begin{aligned}
P(\neg E|F) &= \sum_{H,S} P(\neg E|H,S,F)P(H,S|F) \\
&= \sum_{H,S} P(\neg E|H,S)P(H,S|F) \\
&= \sum_{H,S} P(\neg E|H,S)P(H|F,S)P(S|F) \\
&= \sum_{H,S} P(\neg E|H,S)P(H|F,S)P(S) \\
&= \sum_{H,S} P(\neg E|H,S)P(S)\sum_{T} P(H|F,T)P(T)
\end{aligned}$$

e)

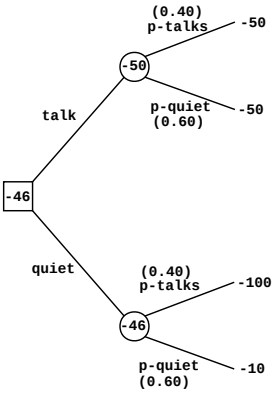
$$\begin{aligned}
P(S|E) &= \frac{P(E|S)P(S)}{P(E)} \\
P(E|S) &= \sum_{H} P(E|S,H)P(H) \\
&= \sum_{H} P(E|S,H)\sum_{F,T} P(H|F,T)P(F)P(T) \\
P(E) &= \sum_{H,S} P(E|S,H)P(S)\sum_{F,T} P(H|F,T)P(F)P(T)
\end{aligned}$$

2 True & False

1. False.
2. True.
3. False.
4. True.

5. False.
6. False.
7. False.
8. True.
9. True.

3 The Prisoner's Dilemma



The decision tree shows that the best choice is to keep quiet - the expected value is higher than that for talking. If we want to find out how much trust in our partner is necessary make keeping quiet the better choice, we need to find out for what value of $P(\text{p} - \text{quiet})$ the expected value of keeping quiet equals -50.

$$\begin{aligned} -100 * (1 - P(\text{p} - \text{quiet})) + (-10 * P(\text{p} - \text{quiet})) &= -50 \\ -100 + 100P(\text{p} - \text{quiet}) - 10P(\text{p} - \text{quiet}) &= -50 \\ -100 + 90P(\text{p} - \text{quiet}) &= -50 \\ 90P(\text{p} - \text{quiet}) &= 50 \\ P(\text{p} - \text{quiet}) &= \frac{5}{9} \approx .55 \end{aligned}$$

So as long as I believe that the probability of my partner keeping quiet is $\frac{5}{9}$ or better, choosing to keep quiet is the best strategy.

4 Searching

a) A*-search would be a good strategy. Straight-line distance is known to be an admissible heuristic and should be a good approximation of the

distance remaining to our goal from any point. And, we know that A*-search will return the shortest path, allowing us to beat out our trading competitors.

b) I can still use the straight-line distances. Heuristics are admissible so long as they do not *overestimate* the distance to a goal.

5 Overfitting

In regression and in crafting generative models, we can make our model fit the training points too closely, which often means that we are modeling noise. Or we start fitting with a model that is more complex than is necessary to fit the points. In either case, future data are likely to not fit our model as well as they would have fit a simpler model of the training data. The standard approach is to add a penalty to the error criterion based on the regularity or the complexity of the model. This will make the best-fit calculation balance the complexity of the model against the error it produces on the training data, and reduce the likelihood of overfitting.

In classification models, we can encounter a similar problem. The classifier will separate the training data precisely, but the boundary may not generalize well. We can resist overfitting by using simpler models, either with fewer features or (in multi-layer nets) with fewer hidden nodes. Other possibilities are to use support vector machines to find the maximal-margin separator, which should be more resistant to error, or to add some noise to the training data to make the learned classifier more robust.

6 Planning

There are no constraints on what order the actions must be executed. They are all in the same layer, which indicates that they can be performed in parallel.

7 GraphPlan Vs. Partial Order Planner

GraphPlan will return a two-layer plan, either fizz followed by fuzz, or vice-versa. POP, on the other hand, will return a single-layer plan, indicating that fizz and fuzz can be performed in parallel. Why does this difference occur? GraphPlan declares actions with inconsistent effects mutex, while POP does not. So, POP's solutions are more expressive (encompass more valid possibilities) than do GraphPlan's solutions.

8 More GraphPlan

Yes, this is an admissible heuristic because it will always underestimate the distance to a solution (no solution can be nearer than the first layer where all

of the solution propositions are not mutex).

9 Bayesian Networks

Which of the following conditional independence assumptions are true?

1. False.
2. True.
3. False.
4. True.
5. False.
6. False.
7. False.
8. False.

Neither network is equivalent to the original one. The second network can encode the same joint probability because it's conditional independence relations are a subset of the original network's relations.

10 Automated Inference

The algorithm attempts to avoid calculating unnecesary information. It is easier to understand what is going on if we look at the description and derivation, rather than the pseudocode. α and β indicate normalizing factors that can be computed with information from the known probability tables and the return values of previously made recursive calls.

$$P(A|L) = \alpha P(L|A)P(A)$$

$$\begin{aligned} P(L|A) &= \beta[P(L|C)\,[P(C|AB)P(B) + P(C|A\neg B)P(\neg B)] \\ &\quad +P(L|\neg C)\,[P(\neg C|AB)P(B) + P(\neg C|A\neg B)P(\neg B)]] \end{aligned}$$

Now, we know every value in this equation except for $P(L|C)$, so we make a recursive call to calculate it.

$$\begin{aligned} P(L|C) &= \beta[P(L|D)[P(D|CE)P(E) + P(D|C\neg E)P(\neg E)] \\ &\quad +P(L|\neg D)[P(\neg D|CE)P(E) + P(\neg D|C\neg E)P(\neg E)]] \end{aligned}$$

Again, all of these values can be looked up in the network's conditional probability tables, except for $P(L|D)$.

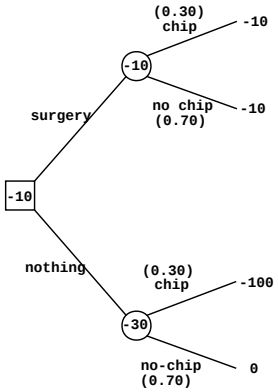
$$\begin{aligned} P(L|D) &= \beta[P(L|J)[P(J|DK)P(K) + P(J|D\neg K)P(\neg K)] \\ &\quad + P(L|\neg J)[P(\neg J|DK)P(K) + P(\neg J|D\neg K)P(\neg K)]] \end{aligned}$$

Finally, we have arrived at a point where every element of the equation can be directly looked up in a known conditional probability table. Notice that we avoided calculating information for the F-G-H chain, or for I. These help to make our calculation more efficient than the brute-force method of reconstructing the entire joint probability table.

11 Sampling

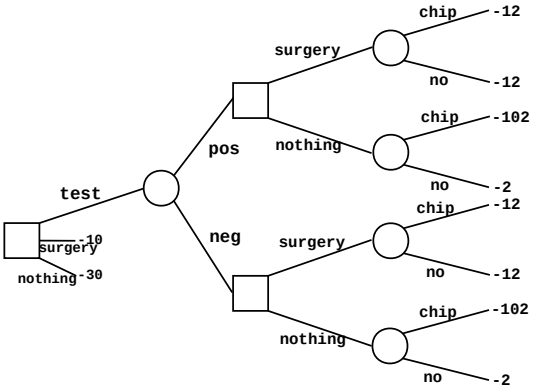
You would use sampling-based inference in a very large Bayesian network or one with undirected cycles that make exact inference procedures infeasible. However, in cases where you wish to evaluate the probability of events with very small probability, sampling is not an effective strategy.

12 Medical Decisions



Clearly, surgery is the best decision. The value of perfect information is 7 because it would allow me to skip surgery (have zero cost) 70% of the time.

$$P(\text{chip}) * 0 + P(\neg\text{chip}) * (-10) = -3$$



It's too hard to fit the math on the diagram, so here are the relevant probabilities and expected values. Abbreviations are: p - positive test, n - negative test, bc - bone chip, s - surgery.

$$
\begin{aligned}
P(p|bc) = x \qquad & P(n|bc) = (1-x) \\
P(p|\neg bc) = y \qquad & P(n|\neg bc) = (1-y) \\
P(p) &= P(p|bc)P(bc) + P(p|\neg bc)P(\neg bc) \\
&= 0.3x + 0.7y \\
P(n) &= P(n|bc)P(bc) + P(n|\neg bc)P(\neg bc) \\
&= 0.3(1-x) + 0.7(1-y) \\
&= 1 - 0.3x - 0.7y \\
P(bc|p) &= \frac{P(p|bc)P(bc)}{P(p)} \\
&= \frac{0.3x}{0.3x + 0.7y} \\
P(\neg bc|p) &= \frac{P(p|\neg bc)P(\neg bc)}{P(p)} \\
&= \frac{0.7y}{0.3x + 0.7y} \\
P(bc|n) &= \frac{P(n|bc)P(bc)}{P(n)} \\
&= \frac{0.3 - 0.3x}{1 - 0.3x - 0.7y} \\
P(\neg bc|n) &= \frac{P(n|\neg bc)P(\neg bc)}{P(n)} \\
&= \frac{0.7 - 0.7y}{1 - 0.3x - 0.7y} \\
E[s|p] &= -12P(bc|p) + -12P(\neg bc|p) \\
&= -12
\end{aligned}
$$

$$
\begin{aligned}
E[\neg s|p] &= -102P(bc|p) + -2P(\neg bc|p) \\
&= \frac{-30.6x - 1.4y}{0.3x + 0.7y} \\
E[s|n] &= -12 \\
E[\neg s|n] &= -102P(bc|n) + -2P(\neg bc|n) \\
&= \frac{-32 + 30.6x + 1.4y}{1 - 0.3x - 0.7y}
\end{aligned}
$$

Now it gets ugly. Let's refer to our decision to take the test as D.

$$
\begin{aligned}
E[D] &= \max(E[s|p], E[\neg s|p])P(p) + \max(E[s|n], E[\neg s|n])P(n) \\
E[D] &= \max(-12, E[\neg s|p])P(p) + \max(E[-12, E[\neg s|n])P(n)
\end{aligned}
$$

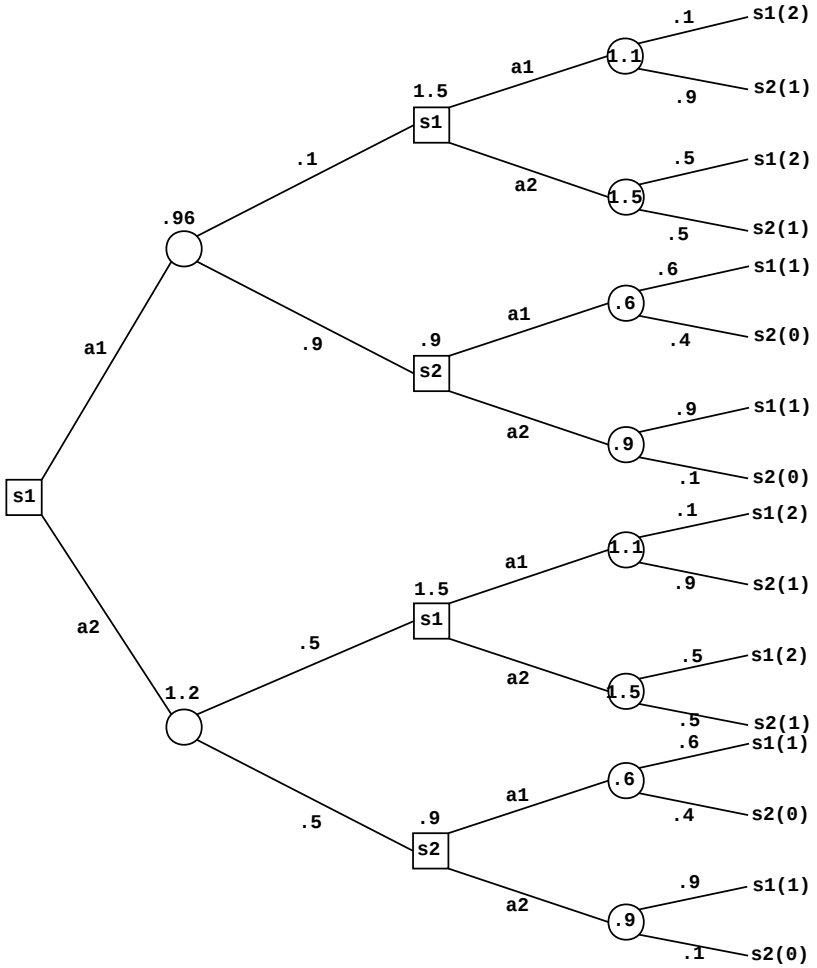
Clearly, if we elect to have surgery no matter what, we are going to get an expected value of -12, which is less than our previous max of -10. So we need to at least not elect to have surgery some time. If we always elect not to have surgery, then we are going to end up with a worse expected value than our uninformed "no surgery" choice. Unless the test is completely useless, we should elect to have surgery if it detects bone spurs and elect not to have surgery if it does not.

$$
\begin{aligned}
E[D] &= -12P(p) + E[\neg s|n]P(n) \\
&= -12(0.3x + 0.7y) + \frac{(-32 + 30.6x + 1.4y)}{(1 - 0.3x - 0.7y)}(1 - 0.3x - 0.7y) \\
&= -3.6x - 8.4y - 32 + 30.6x + 1.4y \\
&= 27x - 7y - 32 > -10 \\
27x - 7y &> 22 \\
x &> (0.259y + 0.815)
\end{aligned}
$$

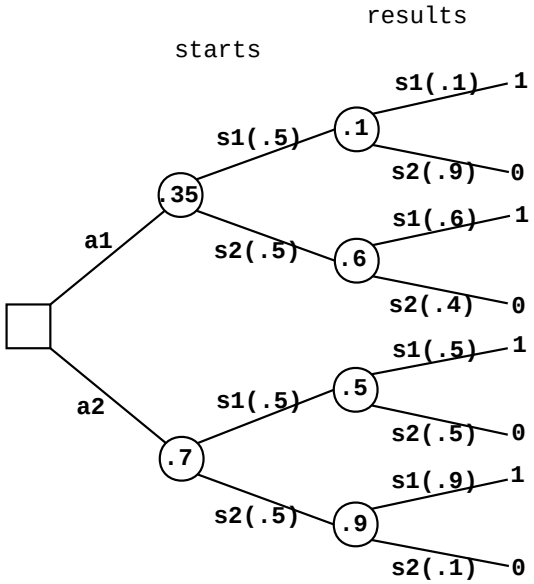
Any values of x and y which satisfy this relationship (and, of course, which are less than or equal to 1, so they are valid probabilities) will give us a better expected value than just deciding on surgery without having the examination.

13 Markov Decision Processes

The decision tree for the optimal action strategy over two steps, starting in state s_1.



The decision tree for the optimal one-step strategy, assuming no knowledge of the starting state. Clearly, we should take a_2.



And now, the value functions. Because this is a simple system, we can calculate them directly and avoid value-iteration.

$$\begin{aligned} V(s_1) &= 1 + \gamma \max(.1V(s_1) + .9V(s_2), .5V(s_1) + .5V(s_2)) \\ &= 1 + .9[.5V(s_1) + .5V(s_2)] \\ &= 1 + .45V(s_1) + .45V(s_2) \\ V(s_2) &= 0 + \gamma \max(.6V(s_1) + .4V(s_2), .9V(s_1) + .1V(s_2)) \\ &= .9[.9V(s_1) + .1V(s_2)] \\ &= .81V(s_1) + .09V(s_2) \end{aligned}$$

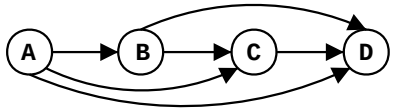
So, we have two unknowns, and two linear equations. So we can easily solve them and get $V(s_1) \approx 6.69$ and $V(s_2) \approx 5.96$.

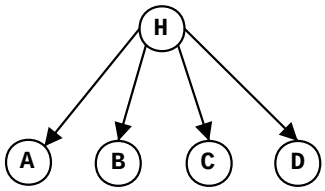
14 Some Other Questions

1. If we do not observe any occurences of B this value is undefined. Also, if there are no occurences of (A and B) or of not (A and B), then we can get a 0 probability, which can sometimes cause trouble later.

2. Reversal might lower the error in a single step, but deletion and addition as separate steps might require increasing the error temporarily. Gradient descent algorithms will not select moves that increase the measured error, so we need to make reversal an atomic action. Another alternative would be a stochastic search method (such as simulated annealing) which can sometimes take non-improving steps.

3. The four-element network has a complexity of 15, the five-element network requires only 9 conditional probability values. This illustrates the use of a hidden variable to simplify a network.





4. This network is only ever going to generate a single output. Call it z. Then

$$E = \sum_i (z - y^i)^2 \ .$$

So

$$\frac{dE}{dz} = \sum_i 2(z - y^i)z(1 - z) \ .$$

E is minimized when the derivative is zero; that happens, uninterestingly at $z = 0$ and $z = 1$. Let's set it to zero and see what else happens (using the proportion of 1's and 0's among the y^i):

$$\begin{aligned} 80(z-1) + 20(z-0) &= 0 \\ 100z &= 80 \\ z &= 0.8 \end{aligned}$$

5. Let's assume that the event of prisoner A being executed is represented by A, and the event of being pardoned is represented by $\neg A$.

   So, at first there are three possibilities: A is executed, and B and C are pardoned; B is executed, while A and C are pardoned; or C is executed, while A and B are pardoned. All other possible combinations of events are impossible and have probability 0.

   When the guard returns and tells A that prisoner B has been pardoned, what is the probability that A will be executed?

$$
\begin{aligned}
P(A|\neg B) &= \frac{P(A,\neg B)}{P(\neg B)} \\
&= \frac{P(A,\neg B,\neg C)}{P(A,\neg B,\neg C)+P(C,\neg A,\neg B)} \\
&= \frac{1/3}{1/3+1/3} \\
&= 0.5
\end{aligned}
$$

So, now that you know that B is being pardoned, the probability that you will be executed is 50%. Before, your probability of execution was only $\frac{1}{3}$. Weird, huh? This is similar to another famous probability scenario known as the "Monty Hall" problem.

6. If $f(X) = 0$ or $f(X) = 1$, meaning that $X \cdot W$ was greater than b or less than $-b$, there is no gradient. In the other cases, we can derive the following gradient equation.

$$
\begin{aligned}
s &= ((1/2b)(X \cdot W + b) - d)^2 \\
\frac{ds}{dW_i} &= ((1/2b)(X \cdot W + b) - d)(X_i/b)
\end{aligned}
$$

The problem is that points that are mislabeled, but produce a 0 or a 1 will produce large error, but no gradient. So the classifier will not be able to reduce the error of these cases by gradient descent.

7. Let the vector of weights from the inputs to the first unit be w_1 and the vector of weights input the second unit be w_2. Let $v = \{g_1(w_1 \cdot x); x\}$.

$$
\begin{aligned}
E &= (f-d)^2 \\
E &= (g_2(w_2 \cdot v) - d)^2 \\
\frac{dE}{dw_{2n}} &= 2(g_2 - d) \cdot (g_2(1-g_2)) \cdot v_n \\
\frac{dE}{dw_{1n}} &= 2(g_2 - d) \cdot (g_2(1-g_2)) \cdot w_{20} g_1(1-g_1) \cdot x_n
\end{aligned}
$$

8. First example: looks good, let's try resolution.

$$
\begin{aligned}
&\neg((((\exists x)P(x)) \rightarrow Q(A)) \rightarrow ((\forall x)(P(x) \rightarrow Q(A)))) \\
&\neg(\neg(\neg((\exists x)P(x)) \vee Q(A)) \vee ((\forall y)(\neg P(y) \vee Q(A))))
\end{aligned}
$$

$$((\neg\exists x.P(x)) \vee Q(A)) \wedge \exists y.(P(y) \wedge \neg Q(A))$$
$$(\forall x.\neg P(x) \vee Q(A)) \wedge \exists y.(P(y) \wedge \neg Q(A))$$

1. $\neg P(x) \vee Q(A)$
2. $P(fred)$
3. $\neg Q(A)$
4. $\neg P(x)$ (3,1)
5. False (4,2, fred/x)

Second example: looks good, let's try resolution.

$$\neg(((\forall x.P(x)) \rightarrow Q(A)) \rightarrow (\exists x.(P(x) \rightarrow Q(A))))$$
$$((\forall x.P(x)) \rightarrow Q(A)) \wedge \neg\exists x.(P(x) \rightarrow Q(A))$$
$$(\neg(\forall x.P(x)) \vee Q(A)) \wedge \forall x.\neg(\neg P(x) \vee Q(A))$$
$$(\exists x.\neg P(x) \vee Q(A)) \wedge \forall x.(P(x) \wedge \neg Q(A))$$
$$(\neg P(fred) \vee Q(A)) \wedge P(x) \wedge \neg Q(A)$$

1. $\neg P(fred) \vee Q(A)$
2. $P(x)$
3. $\neg Q(A)$
4. $\neg P(fred)$ (3,1)
5. False (4,2)

Third example: Hmmm, I'm a bit suspicious. Maybe we can find a counterexample.

$$Q(a) = false$$
$$P(fred) = true$$
$$P(ned) = false$$
$$\exists x.(P(x) \rightarrow Q(a)) \rightarrow \forall x.(P(x) \rightarrow Q(a))$$

We need a situation in which the left-hand side is true and the right-hand side is false. Consider if we satisfy the left-hand side with x=ned. Because $P(ned) = false$, the left-hand side will be true despite the fact the Q(a) is always false. But, because the right-hand side is universally quantified, it is not true, because P(fred) is true, which will make $P(fred) \rightarrow Q(a)$ false. So, the entire expression is false because we have a true left-hand side and a false right-hand side.

9. $\neg\text{On}(x, z) \vee \neg\text{Above}(z, y) \vee \text{Above}(x, y)$

10. The first-order logic description is:

$$\forall x.(P(x) \rightarrow B(x)) \rightarrow \forall y.(\neg P(y) \rightarrow G(y))$$
$$\forall x.(B(x) \vee G(x)) \wedge (\neg B(x) \vee \neg G(x))$$
$$(\exists x.\neg P(x)) \rightarrow (\forall y.P(y) \rightarrow B(y))$$
$$P(O1)$$
$$\neg P(O2)$$

Which turns into the following clausal form statements:

1. $P(F) \vee P(y) \vee G(y)$
2. $\neg B(F) \vee P(y) \vee G(y)$
3. $B(x) \vee G(x)$
4. $\neg B(x) \vee \neg G(x)$
5. $P(x) \vee \neg P(y) \vee B(y)$
6. $P(O1)$
7. $\neg P(O2)$

   And then we can prove that there is a green object:

8. $\neg G(x)$
9. $\neg B(F) \vee G(O2)$ (2,7)
10. $G(F) \vee G(O2)$ (3,9)
11. $G(O2)$ (8,10)
12. False (8,11)

11. (a) Optimize threatens HappyCustomer because it deletes BugFree.
    (b) HaveProgram is not linked to optimize, and HavePackaging (requirement of Ship) is unsupported.
    (c) Any plan that places debug after optimize and both before ship is acceptable.