- 14.1 (Thomas Nakagawa) Suppose we have a bag of three biased coins *a*, *b* and *c*, which have probability of coming up heads of 20%, 60% and 80% respectively. We draw one of the three coins randomly and flip it three times to get outcomes *X1*, *X2* and *X3*.
  - Draw a Bayesian network corresponding to this setup and define the relevant CPTs.
  - Calculate which coin is most likely to have been drawn if the flips come up HHT.

14.1

a. With the random variable C denoting which coin $\{a, b, c\}$ we drew, the network has C at the root and X_1, X_2, and X_3 as children.

The CPT for C is:

C	$P(C)$
a	1/3
b	1/3
c	1/3

The CPT for X_i given C are the same, and equal to:

C	X_1	$P(C)$
a	*heads*	0.2
b	*heads*	0.6
c	*heads*	0.8

b. The coin most likely to have been drawn from the bag given this sequence is the value of C with greatest posterior probability $P(C|2 \text{ heads}, 1 \text{ tails})$. Now,

$$\begin{aligned} P(C|2 \text{ heads}, 1 \text{ tails}) &= P(2 \text{ heads}, 1 \text{ tails}|C)P(C)/P(2 \text{ heads}, 1 \text{ tails}) \\ &\propto P(2 \text{ heads}, 1 \text{ tails}|C)P(C) \\ &\propto P(2 \text{ heads}, 1 \text{ tails}|C) \end{aligned}$$

where in the second line we observe that the constant of proportionality $1/P(2 \text{ heads}, 1 \text{ tails})$ is independent of C, and in the last we observe that $P(C)$ is also independent of the value of C since it is, by hypothesis, equal to $1/3$.

From the Bayesian network we can see that X_1, X_2, and X_3 are conditionally independent given C, so for example

$$\begin{aligned} & P(X_1 = tails, X_2 = heads, X_3 = heads|C = a) \\ &= P(X_1 = tails|C = a)P(X_2 = heads|C = a)P(X_3 = heads|C = a) \\ &= 0.8 \times 0.2 \times 0.2 = 0.032 \end{aligned}$$

Note that since the CPTs for each coin are the same, we would get the same probability above for any ordering of 2 heads and 1 tails. Since there are three such orderings, we have

$$P(2heads, 1tails|C = a) = 3 \times 0.032 = 0.096.$$

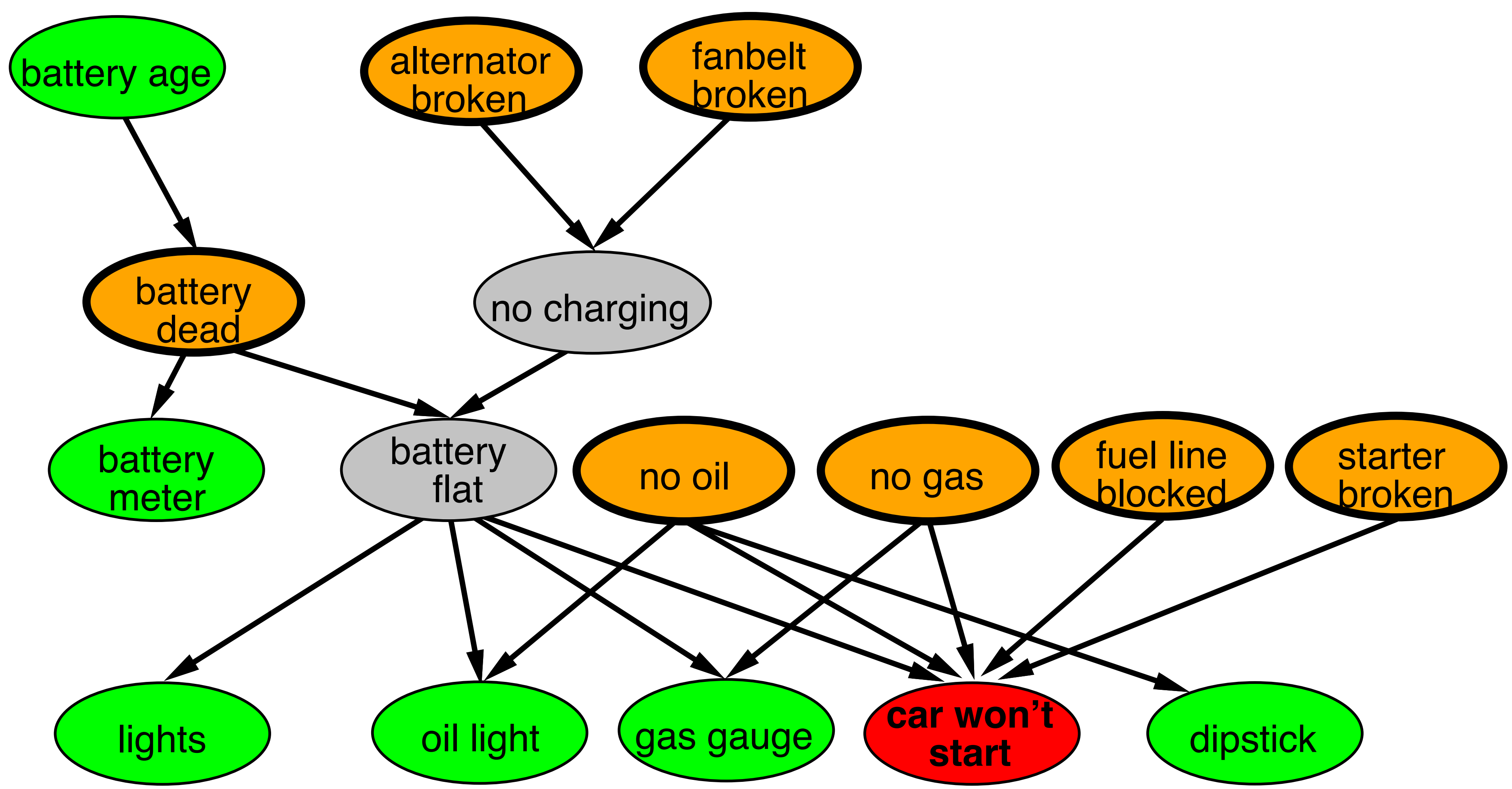
Similar calculations to the above find that

$$\begin{aligned} P(2heads, 1tails|C = b) &= 0.432 \\ P(2heads, 1tails|C = c) &= 0.384 \end{aligned}$$

showing that coin b is most likely to have been drawn.

Alternatively, one could directly compute the value of $P(C|2 \text{ heads}, 1 \text{ tails})$.

- 14.8 (Lijun (Julie) Zhu) Consider the network for car diagnosis.
  - Extend the network with boolean variables for IcyWeather and StarterMotor.
  - Give reasonable CPTs for all the nodes.
  - How many independent numbers do your network's tables contain? In contrast, how many independent numbers are contained in the joint probability distribution for eight boolean variables, assuming no known conditional independencies?
  - The conditional distribution for Starts could be described as a noisy-AND distribution. Define this family in general and relate it to the noisy-OR distribution.



14.8 Adding variables to an existing net can be done in two ways. Formally speaking, one should insert the variables into the variable ordering and rerun the network construction process from the point where the first new variable appears. Informally speaking, one never really builds a network by a strict ordering. Instead, one asks what variables are direct causes or influences on what other ones, and builds local parent/child graphs that way. It is usually easy to identify where in such a structure the new variable goes, but one must be very careful to check for possible induced dependencies downstream.

a. $IcyWeather$ is not caused by any of the car-related variables, so needs no parents. It directly affects the battery and the starter motor. $StarterMotor$ is an additional precondition for $Starts$. The new network is shown in Figure S14.1.

b. Reasonable probabilities may vary a lot depending on the kind of car and perhaps the personal experience of the assessor. The following values indicate the general order of magnitude and relative values that make sense:

- A reasonable prior for IcyWeather might be 0.05 (perhaps depending on location and season).
- $P(Battery|IcyWeather) = 0.95$, $P(Battery|\neg IcyWeather) = 0.997$.
- $P(StarterMotor|IcyWeather) = 0.98$, $P(Battery|\neg IcyWeather) = 0.999$.
- $P(Radio|Battery) = 0.9999$, $P(Radio|\neg Battery) = 0.05$.
- $P(Ignition|Battery) = 0.998$, $P(Ignition|\neg Battery) = 0.01$.
- $P(Gas) = 0.995$.
- $P(Starts|Ignition, StarterMotor, Gas) = 0.9999$, other entries 0.0.
- $P(Moves|Starts) = 0.998$.

c. With 8 Boolean variables, the joint has $2^8 - 1 = 255$ independent entries.

d. Given the topology shown in Figure S14.1, the total number of independent CPT entries is 1+2+2+2+2+1+8+2= 20.

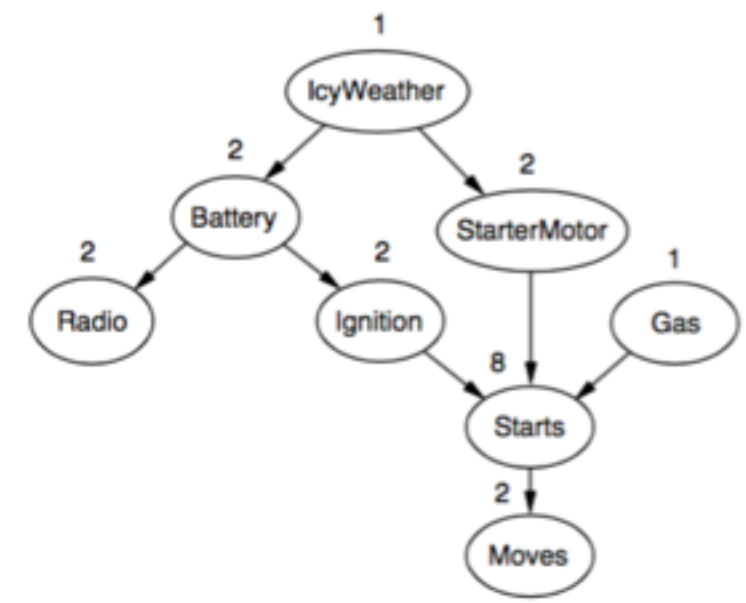


Figure S14.1 Car network amended to include $IcyWeather$ and $StarterMotorWorking$ (SMW).

e. The CPT for $Starts$ describes a set of nearly necessary conditions that are together almost sufficient. That is, all the entries are nearly zero except for the entry where all the conditions are true. That entry will be not quite 1 (because there is always some other possible fault that we didn't think of), but as we add more conditions it gets closer to 1. If we add a $Leak$ node as an extra parent, then the probability is exactly 1 when all parents are true. We can relate noisy-AND to noisy-OR using de Morgan's rule: $A \wedge B \equiv \neg(\neg A \vee \neg B)$. That is, noisy-AND is the same as noisy-OR except that the polarities of the parent and child variables are reversed. In the noisy-OR case, we have

$$P(Y = true|x_1, \ldots, x_k) = 1 - \prod_{\{i: x_i = true\}} q_i$$

where q_i is the probability that the *presence* of the ith parent *fails* to cause the child to be *true*. In the noisy-AND case, we can write

$$P(Y = true|x_1, \ldots, x_k) = \prod_{\{i: x_i = false\}} r_i$$

where r_i is the probability that the *absence* of the ith parent *fails* to cause the child to be *false* (e.g., it is magically bypassed by some other mechanism).

- 14.10 (Farzin Ahmed) The probit distribution describes the probability distribution for a Boolean child given a single continuous parent.
  - How might the definition be extended to cover multiple continuous parents?
  - How might it be extended to handle a multi-valued child variable? consider both cases where the child's values are ordered (e.g. selecting a gear while driving) and cases where they are unordered (as in selecting bus/train/car to get to work). Hint: consider ways to divide the possible values into two sets, to mimic a Boolean child.

14.10

a. With multiple continuous parents, we must find a way to map the parent value vector to a single threshold value. The simplest way to do this is to take a linear combination of the parent values.

b. For ordered values $y_1 < y_2 < \cdots < y_d$, we assume some unobserved continuous dependent variable y^* that is normally distributed conditioned on the parent variables, and define cutpoints c_j such that $Y = y_j$ iff $c_{j-1} \leq y^* \leq c_j$. The probability of this event is given by subtracting the cumulative distributions at the adjacent cutpoints.

The unordered case is not obviously meaningful if we insist that the relationship between parents and child be mediated by a single, real-valued, normally distributed variable.

- 14.15 (Robin Qumsieh)
  - Perform variable elimination for the query *P(Burglary | JohnCalls = true, MaryCalls = true)*.
  - Count the number of arithmetic operations performed, and compare to the number performed by the enumeration algorithm.
  - Suppose a network has the form of a *chain*: A sequence of Boolean variables *X1..XN* where *Parents(Xi) = {X(i-1)}*. What is the complexity of computing *P(X1 | XN = true)* using enumeration? Variable elimination?

14.15 This question definitely helps students get a solid feel for variable elimination. Students may need some help with the last part if they are to do it properly.

a.

$$
\begin{aligned}
&P(B|j,m) \\
&= \alpha P(B) \sum_e P(e) \sum_a P(a|b,e)P(j|a)P(m|a) \\
&= \alpha P(B) \sum_e P(e) \left[.9 \times .7 \times \begin{pmatrix} .95 & .29 \\ .94 & .001 \end{pmatrix} + .05 \times .01 \times \begin{pmatrix} .05 & .71 \\ .06 & .999 \end{pmatrix} \right] \\
&= \alpha P(B) \sum_e P(e) \begin{pmatrix} .598525 & .183055 \\ .59223 & .0011295 \end{pmatrix} \\
&= \alpha P(B) \left[.002 \times \begin{pmatrix} .598525 \\ .183055 \end{pmatrix} + .998 \times \begin{pmatrix} .59223 \\ .0011295 \end{pmatrix} \right] \\
&= \alpha \begin{pmatrix} .001 \\ .999 \end{pmatrix} \times \begin{pmatrix} .59224259 \\ .001493351 \end{pmatrix} \\
&= \alpha \begin{pmatrix} .00059224259 \\ .0014918576 \end{pmatrix} \\
&\approx \langle .284, .716 \rangle
\end{aligned}
$$

b. Including the normalization step, there are 7 additions, 16 multiplications, and 2 divisions. The enumeration algorithm has two extra multiplications.

c. To compute $\mathbf{P}(X_1|X_n = true)$ using enumeration, we have to evaluate two complete binary trees (one for each value of X_1), each of depth $n-2$, so the total work is $O(2^n)$. Using variable elimination, the factors never grow beyond two variables. For example, the first step is

$$
\begin{aligned}
&\mathbf{P}(X_1|X_n = true) \\
&= \alpha \mathbf{P}(X_1) \ldots \sum_{x_{n-2}} P(x_{n-2}|x_{n-3}) \sum_{x_{n-1}} P(x_{n-1}|x_{n-2}) P(X_n = true|x_{n-1}) \\
&= \alpha \mathbf{P}(X_1) \ldots \sum_{x_{n-2}} P(x_{n-2}|x_{n-3}) \sum_{x_{n-1}} \mathbf{f}_{X_{n-1}}(x_{n-1}, x_{n-2}) \mathbf{f}_{X_n}(x_{n-1}) \\
&= \alpha \mathbf{P}(X_1) \ldots \sum_{x_{n-2}} P(x_{n-2}|x_{n-3}) \mathbf{f}_{\overline{X_{n-1}}X_n}(x_{n-2})
\end{aligned}
$$

The last line is isomorphic to the problem with $n-1$ variables instead of n; the work done on the first step is a constant independent of n, hence (by induction on n, if you want to be formal) the total work is $O(n)$.

d. Here we can perform an induction on the number of nodes in the polytree. The base case is trivial. For the inductive hypothesis, assume that any polytree with n nodes can be evaluated in time proportional to the size of the polytree (i.e., the sum of the CPT sizes). Now, consider a polytree with $n+1$ nodes. Any node ordering consistent with the topology will eliminate first some leaf node from this polytree. To eliminate any leaf node, we have to do work proportional to the size of its CPT. Then, *because the network is a polytree*, we are left with *independent* subproblems, one for each parent. Each subproblem takes total work proportional to the sum of its CPT sizes, so the total work for $n+1$ nodes is proportional to the sum of CPT sizes.