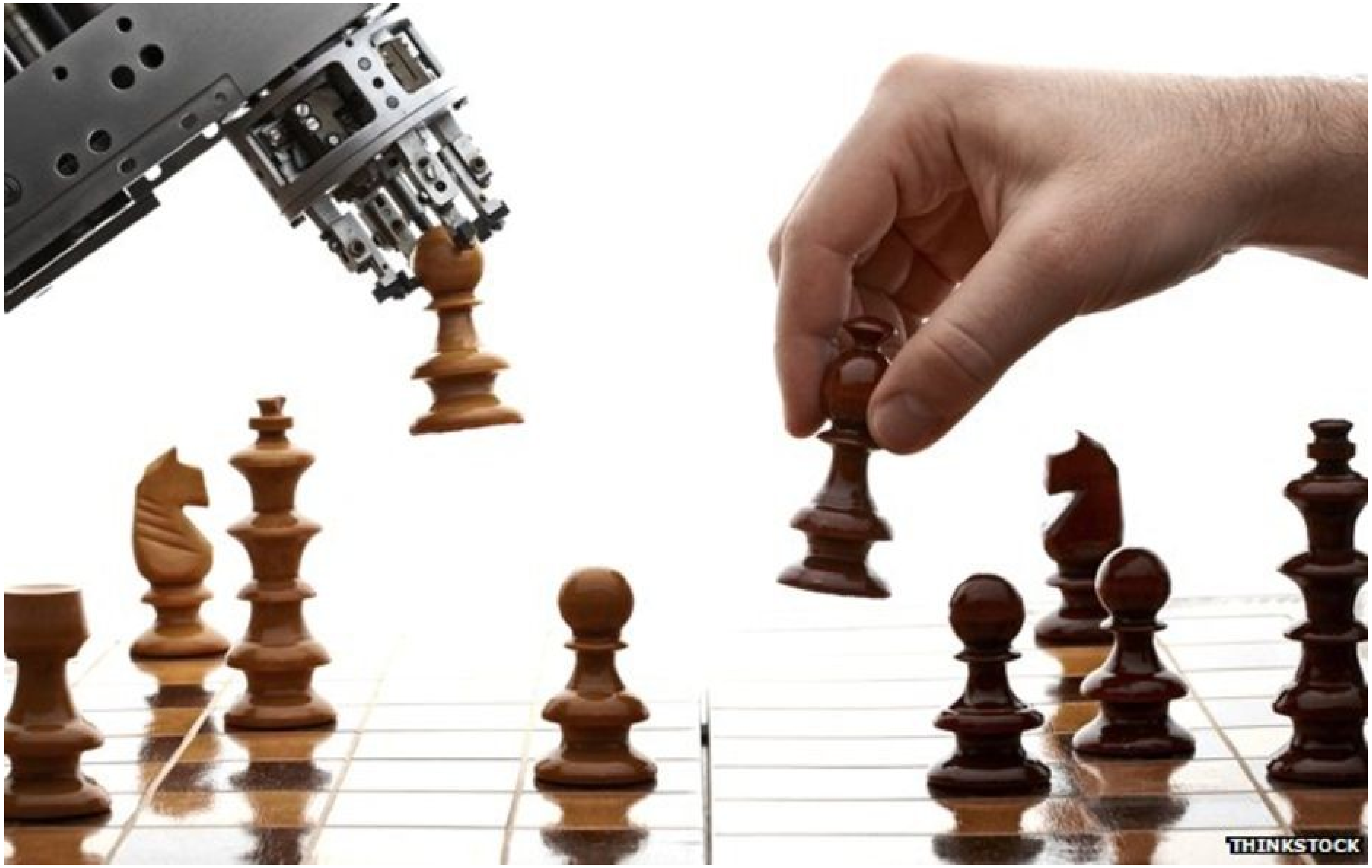


Introduction to Artificial Intelligence

COSC 4550 / COSC 5550

Professor Cheney

9/27/17

to give me teaching feedback!

https://goo.gl/forms/W3MbDeQYH5xWyeU13

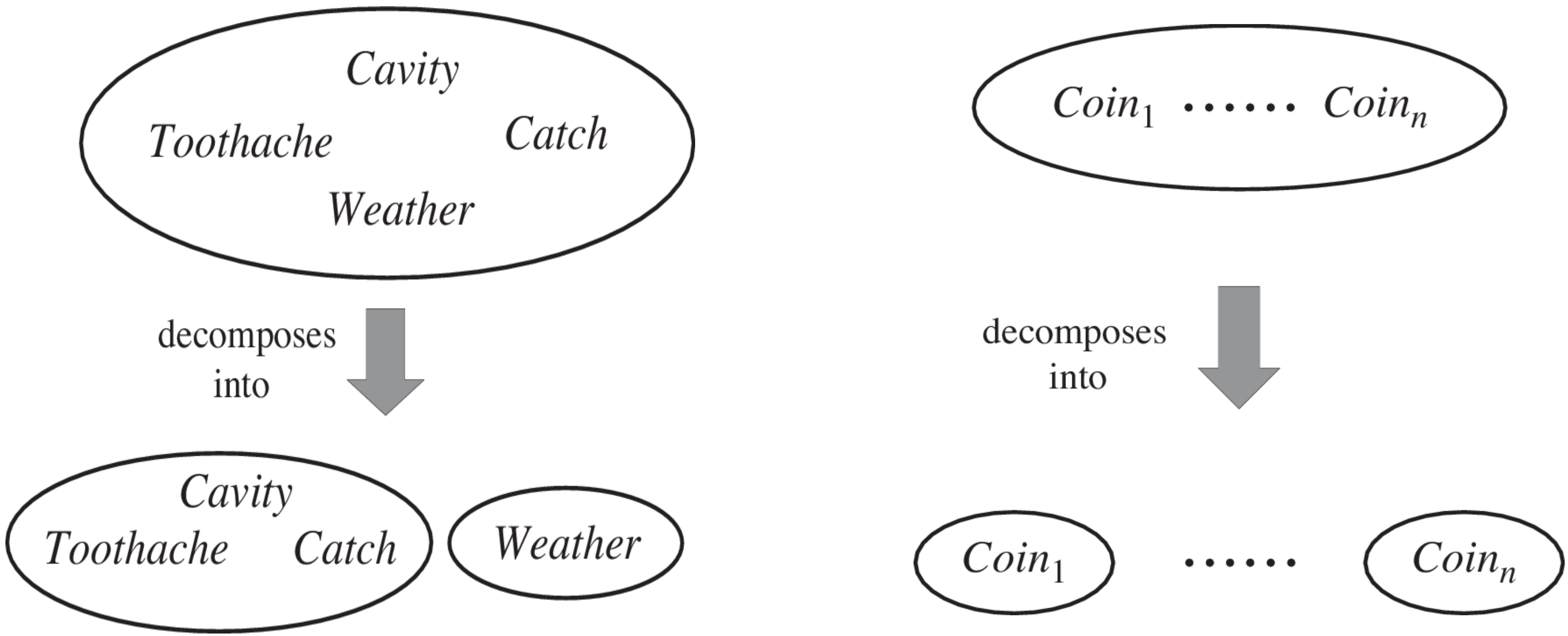
Bayes' rule told us how to update a belief, given one additional piece of evidence

(recall that we assumed the robot was seeing just one sensor value at a time in the last example)

what if we have multiple observations for a single cause?

naive Bayes

recall from the lecture on independence...



$$P(coin_1, coin_2, \ldots, coin_n) = P(coin_1) * P(coin_2) * \ldots * P(coin_n)$$

(since we can assume all the coin flips are independent)

$P(\text{cell}, \text{sensor}_1, \text{sensor}_2, \ldots, \text{sensor}_n) \neq P(\text{cell}) * P(\text{sensor}_1) * P(\text{sensor}_2) * \ldots * P(\text{sensor}_n)$

can't assume independence!
because the grid cell location of the robot
is causing the sensors to give certain readings

but what if we said that the grid cell location was the cause
for all of the sensor readings, and that each of the senors
were independent **given** that you knew what cell you were in?

$$P(\text{cell}, \text{sensor}_1, \text{sensor}_2, \ldots, \text{sensor}_n)$$
$$= P(\text{cell}) * P(\text{sensor}_1 \mid \text{cell}) * P(\text{sensor}_2 \mid \text{cell}) * \ldots * P(\text{sensor}_n \mid \text{cell})$$

$$= P(\text{cause}) * P(\text{effect}_1 \mid \text{cause}) * P(\text{effect}_2 \mid \text{cause}) * \ldots * P(\text{effect}_n \mid \text{cause})$$
$$= P(\text{cause}) * \prod_{i=1\ldots n} P(\text{effect}_i \mid \text{cause})$$

$$P(\text{cause}, \text{effect}_1, \text{effect}_1, \ldots, \text{effect}_1) = P(\text{cause}) * \prod_{i=1\ldots n} P(\text{effect}_i \mid \text{cause})$$

naive Bayes

makes the naive assumption that all effects are independent, conditioned on a already knowing the value of another variable (presumable common cause for them all)

“conditional independence”

surprisingly effective in practice

Can we come up with something less naive?

(making less assumptions about independence of events, allowing multiple causes to affect multiple observations)

Bayesian network

(belief network)

In this example, you are out of town and have asked two of your neighbors (John and Mary) to check on your house and call you if your security alarm goes off. But it turns out your alarm is not perfect, and it's motion detector can also be set off if the house starts shaking because of an earthquake. Also, your neighbors don't always call (only) when there is an alarm

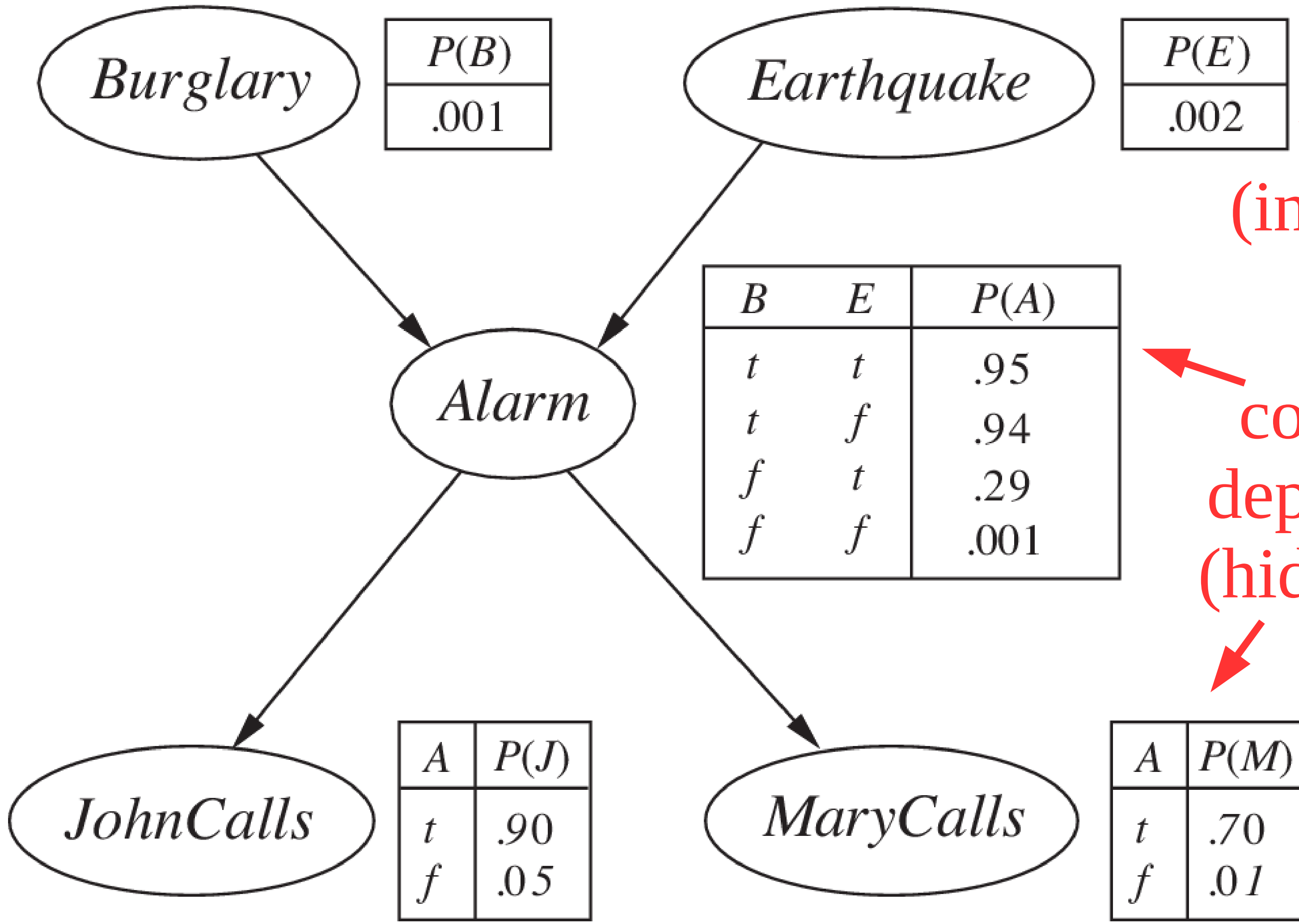
what's the probability you get a phone call?

= P(JohnCalls, alarm, burglar, earthquake)
+ P(JohnCalls, alarm, burglar, ~earthquake)
+ P(JohnCalls, alarm, ~burglar, earthquake)
+ P(JohnCalls, alarm, ~burglar, ~earthquake)
+ P(JohnCalls, ~alarm, burglar, earthquake)
+ P(JohnCalls, ~alarm, burglar, ~earthquake)
…
+ P(MaryCalls, ~alarm, ~burglar, earthquake)
+ P(MaryCalls, ~alarm, ~burglar, ~earthquake)

wow, that's complicated… let's just draw a picture

(i.e. let's draw a graphical model of the dependencies)

Bayes net (or) belief net



independent variable "cause" (input layer)

conditional dependencies (hidden layer)

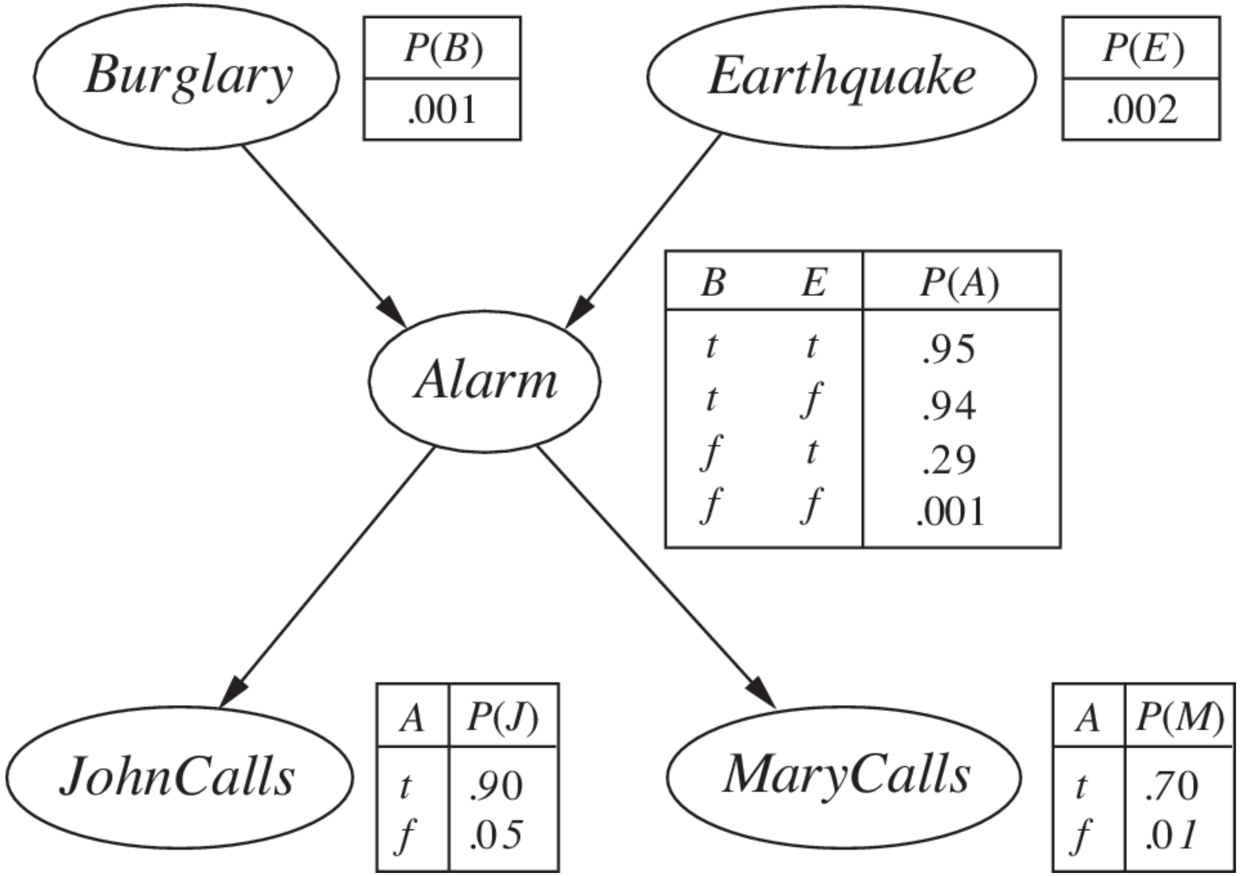
"effect" (output layer)

side note:

belief nets look very similar to neural networks!

(you know the inputs nodes, and propagate their information forward to find the values of hidden and output nodes)

Deep Belief Networks were some of the most popular early version of Deep Neural Networks



P(B)
.001

P(E)
.002

B	E	P(A)
t	t	.95
t	f	.94
f	t	.29
f	f	.001

A	P(J)
t	.90
f	.05

A	P(M)
t	.70
f	.01

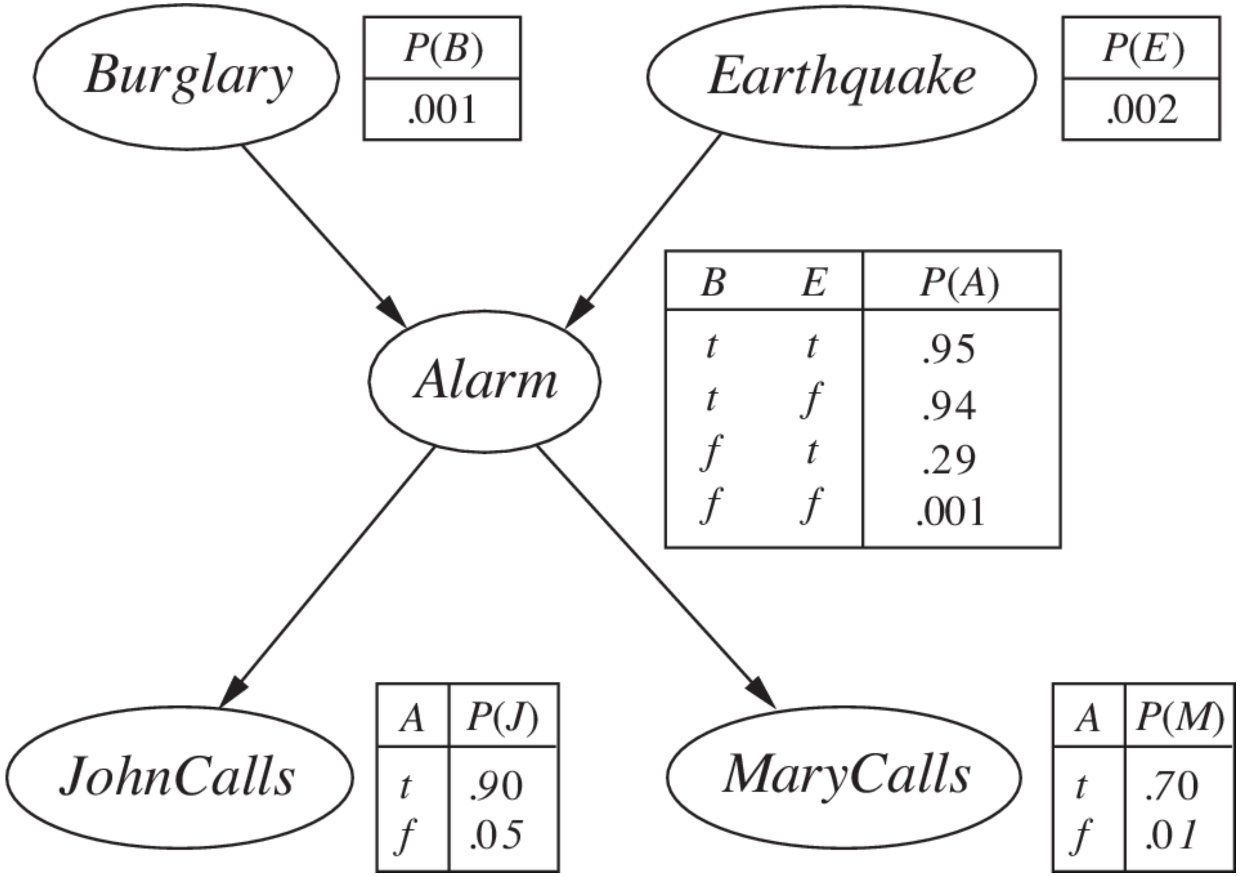
note that the conditional probabilities only depend on nodes in the layer before them!

(or generally nodes with arrows leading to them)

for example:

P(MaryCalls | JohnCalls, Alarm, Burglar, Earthquake) = P(MaryCalls | Alarm)

note that: P(Alarm) still depends on Burglar and Earthquake,
but P(MaryCalls) does not gain any information by including them,
given that we already know if the Alarm went off or not



breaking up the full joint probabilities with conditional independence allows us to make the computations tractable

full joint probability: $2^n = 2^5 = 32$ values (n variables)

this Bayes net: 10 values

general Bayes net: $n*2^k$ (k dependencies/node)

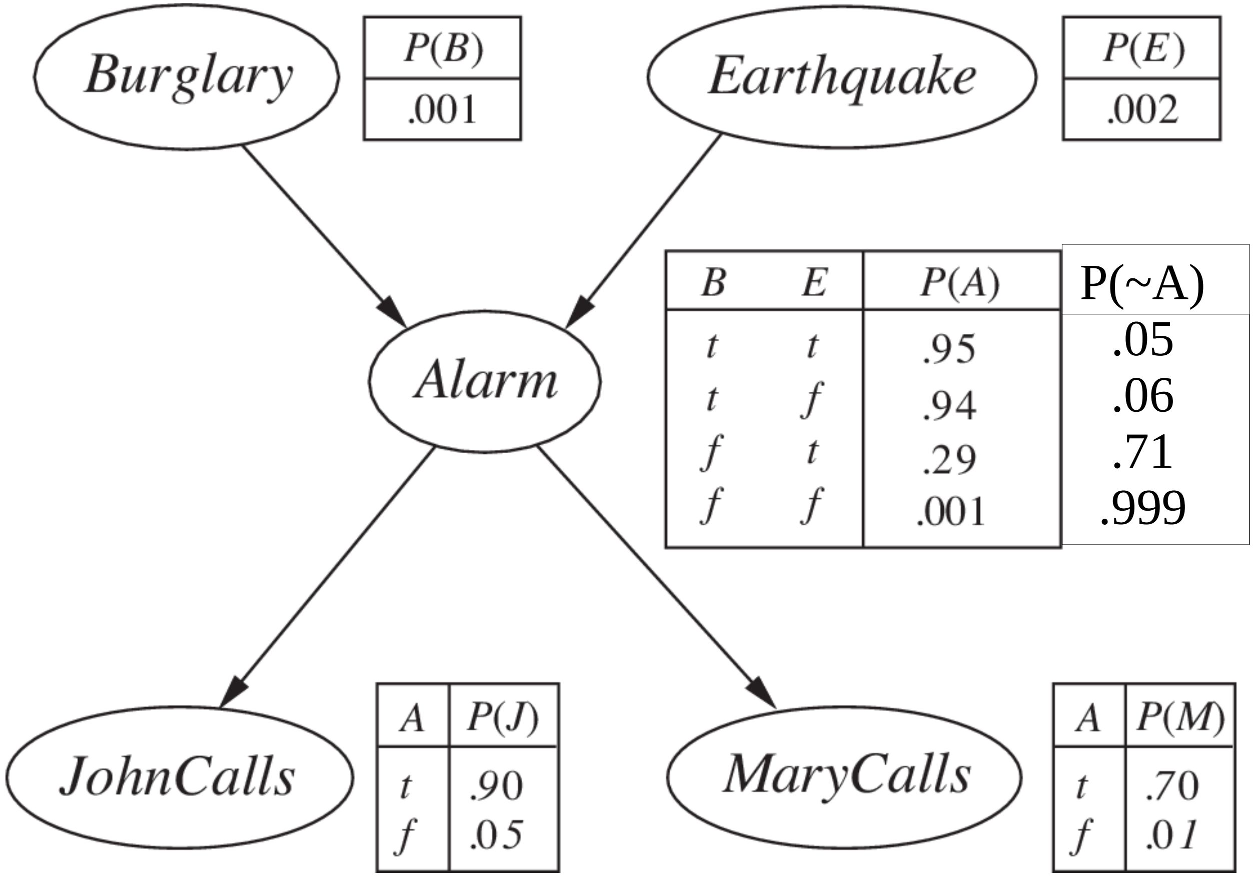
worst-case Bayes net: $n*2^n$ (fully connected network)

Bayes net: $n*2^k$
full joint probability: 2^n

e.g. 30 nodes with 5 dependencies each

Bayes net: $30*2^5 = 960$ entries

full joint probability: $2^{30} = 1{,}000{,}000{,}000$



P(B)
.001

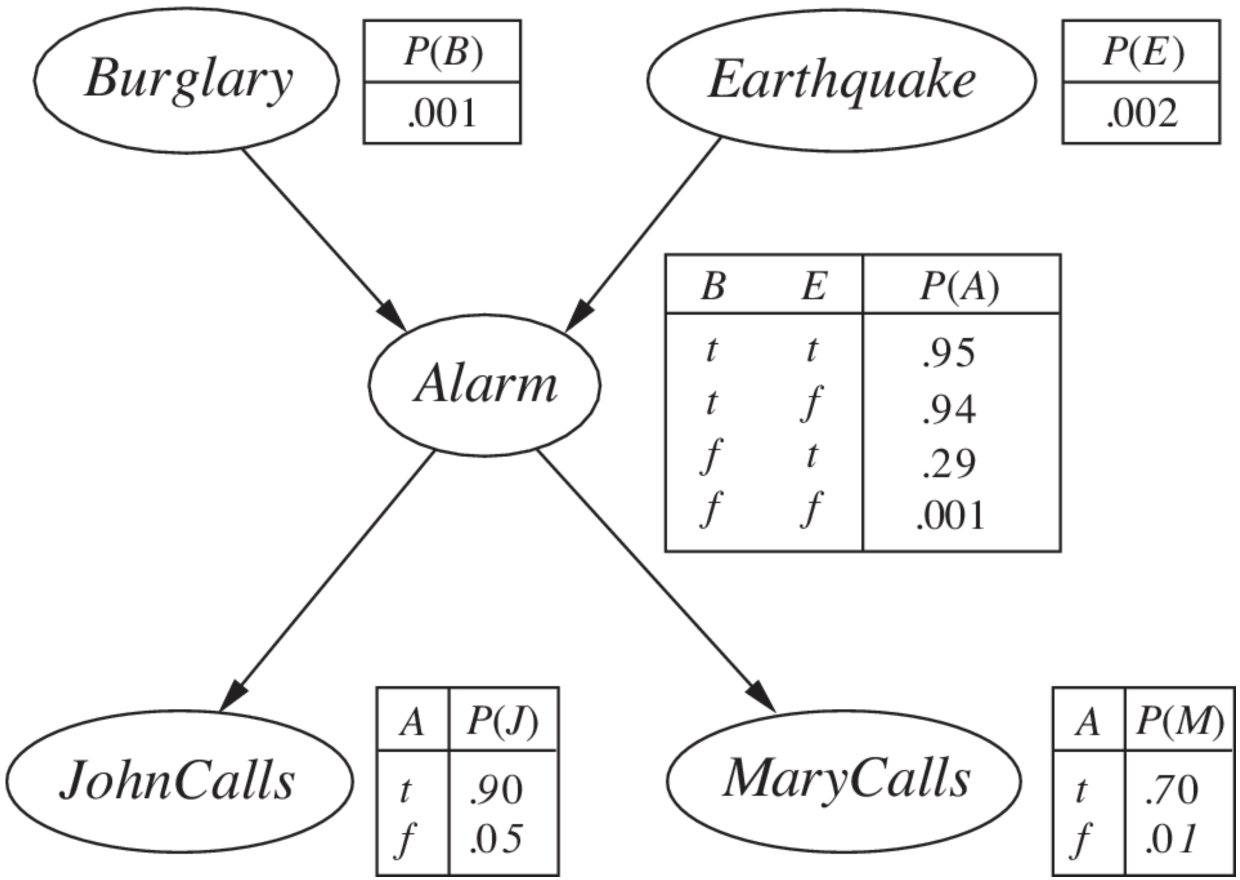
P(E)
.002

B	E	P(A)	P(~A)
t	t	.95	.05
t	f	.94	.06
f	t	.29	.71
f	f	.001	.999

A	P(J)
t	.90
f	.05

A	P(M)
t	.70
f	.01

none of the boxes sum to 1… what gives?



example:

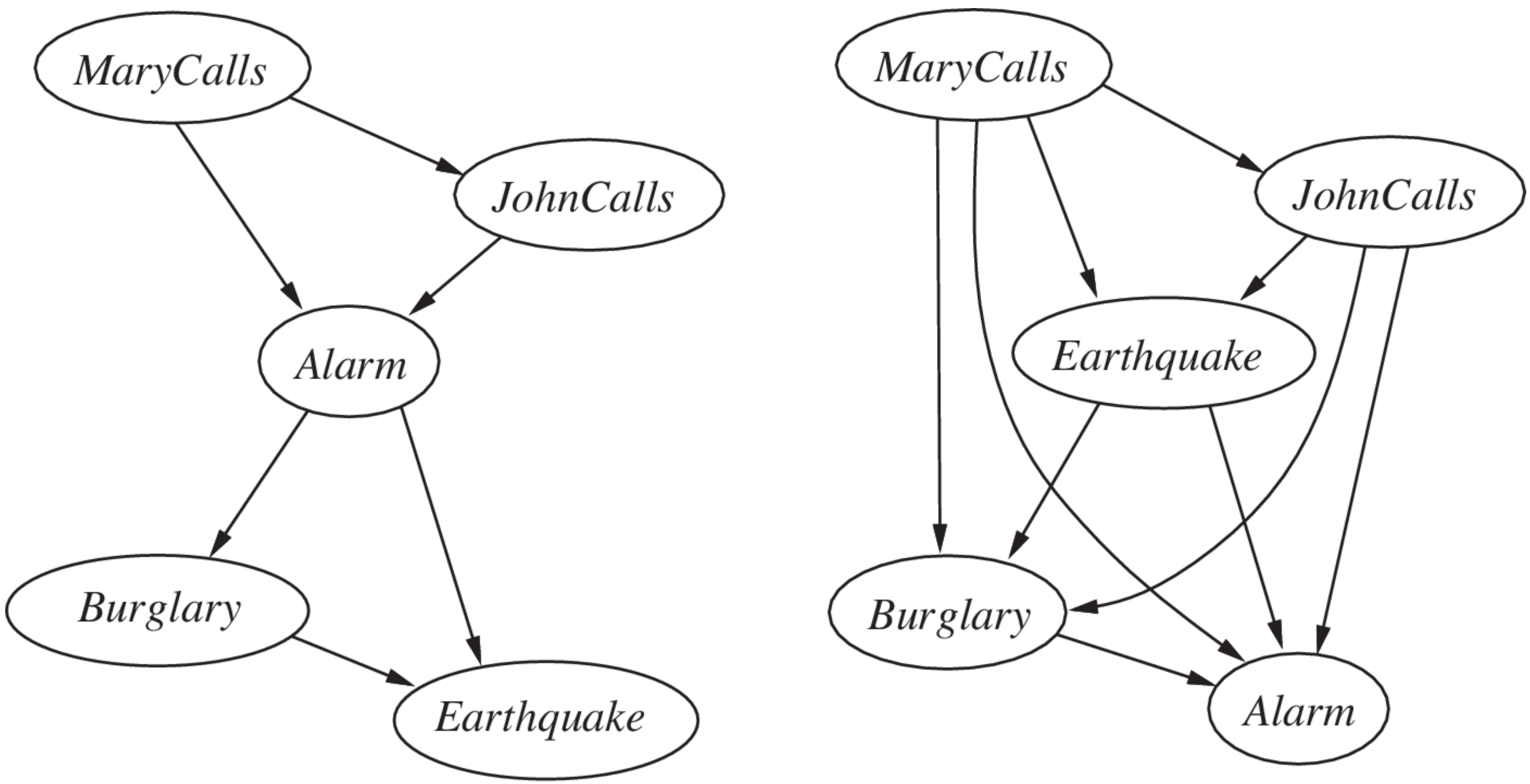
what's the probability that the alarm sounds and both John and Mary call, but there was neither an earthquake nor a burglary?

$$
\begin{aligned}
& P(j, m, a, \sim b, \sim e) \\
&= P(j \mid a) * P(m \mid a) * P(a \mid \sim b, \sim e) * P(\sim b) * P(\sim e) \\
&= 0.9 * 0.7 * 0.001 * 0.999 * 0.998 \\
&= 0.00628
\end{aligned}
$$

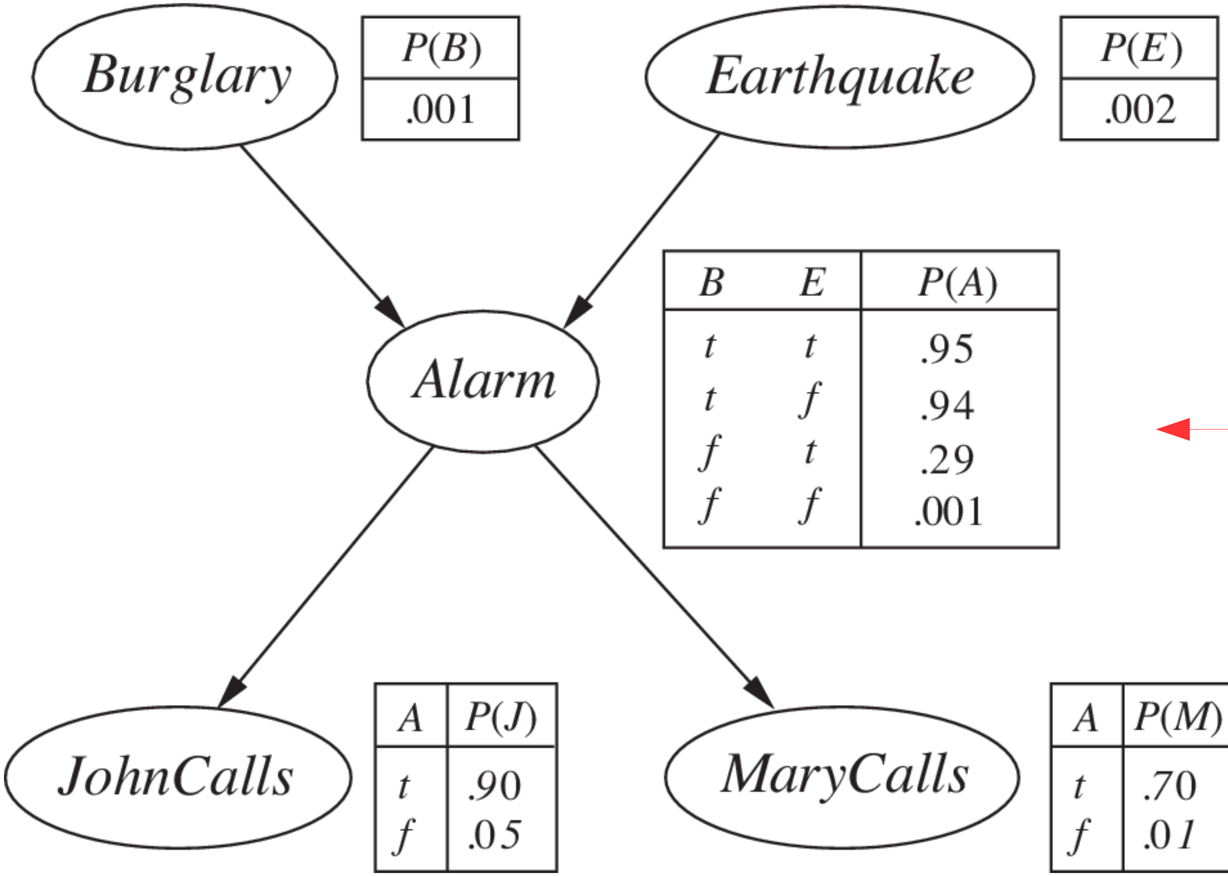
node introduction order matters (see book for full example)

e.g. we said that P(Earthquake) doesn't direct affect P(MaryCalls), given that we already knew if the Alarm went off or not

but… if the Earthquake node is added to the network before the Alarm node, then P(MaryCalls) can still gain information from it



(a) (b)



conditional probability tables assume discrete variables (T or F)

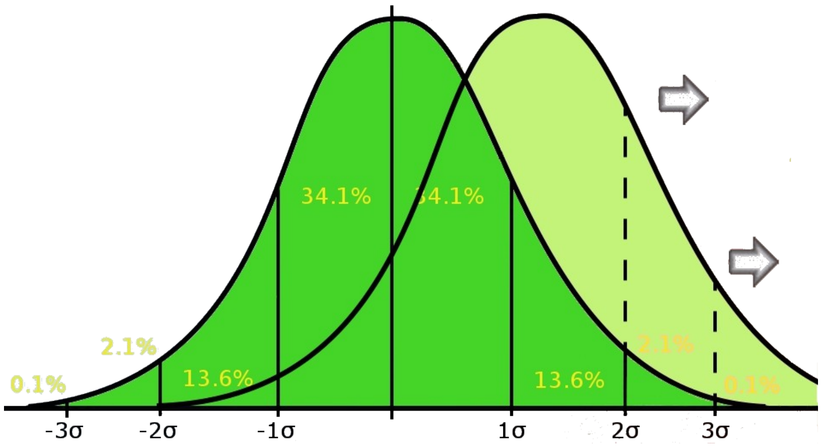
what about continuous variables… ?

(option 1)
discretize continuous variables to discrete ones

(option 2)
use a continuous function to calculate the probability of a node based on its dependencies

often a Gaussian function $g(x) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{1}{2}\left(\frac{x-\mu}{\sigma}\right)^2}$

mean of Gaussian is linear function of dependencies



$$P(c|h) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{1}{2}\left(\frac{c-(a\times h+b)}{\sigma}\right)^2}$$

final projects

idea & outlines due in less than a month!

(step 1)
pick a method we've covered
(already or by the project plan due date)

(step 2)
pick an application that you care about
(a fun game to play, or a question in your field/passion)

(step 3)
apply it to solve this new problem!

(step 1)

pick a method and application that we've covered
(e.g. search on pacman or vacuum-bot)

(step 2)

think of way that our implementation of it sucked
(what would be a more efficient search algorithm/feature set?)

(step 3)

try out your change, and see if it works better!

(step 4)

make an awesome and entertaining (and informative!) YouTube video about what you've done to share with the class (and the world)

also...

(step 2.5)
tell me about what you're going to do
(in writing, criteria to come later)

to make sure that what you're doing is new enough to be cool
and to make sure it's do-able enough for you to finish

optionally...

(step 5)
keep working on the project after class end
turn your video/project into a real scientific paper
and submit it for publication at a conference!

(if you're interested in this, I recommend
evolutionary algorithms and deep learning)

Examples of class projects!

https://www.youtube.com/channel/UCeiDaur181A2lI4CBufSlXA

https://www.youtube.com/user/EvolvingRobots/videos