15-381: Artificial Intelligence

Bayesian networks: Construction and inference

Gaussian (Normal)

- If I look at the height of women in country xx, it will look approximately Gaussian
- Small random noise errors, look Gaussian/Normal

- Distribution:

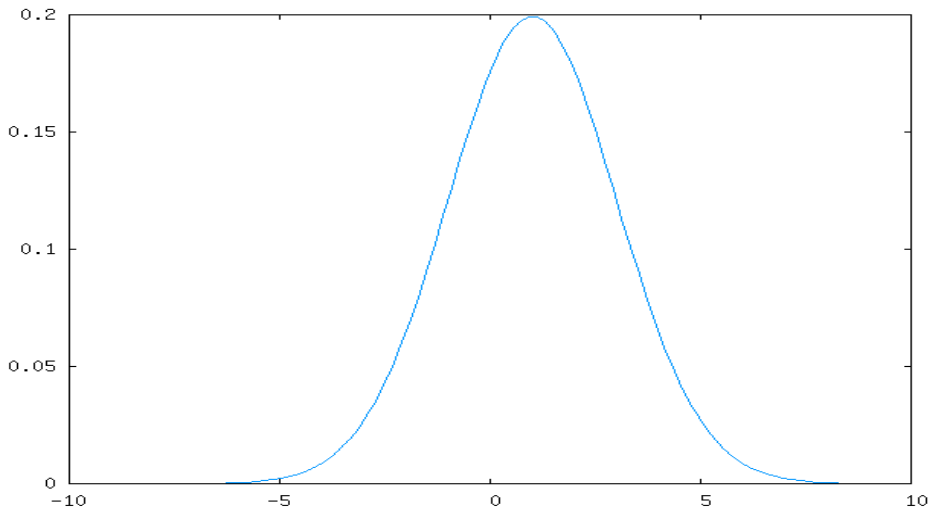
$$x \sim N(\mu, \sigma^2)$$

$$f(x) = \frac{1}{\sqrt{2\pi}\sigma} e^{\frac{-(x-\mu)^2}{2\sigma^2}}$$

- Mean/var

$$E[x] = \mu$$

$$Var(x) = \sigma^2$$



Why Do People Use Gaussians

- Central Limit Theorem: (loosely)
  - Sum of a large number of IID random variables is approximately Gaussian

Multivariate Gaussians

- Distribution for vector x

$$x = (x_1, \ldots, x_N)^T, \quad x \sim N(\mu, \Sigma)$$

- PDF:

$$f(x) = \frac{1}{(2\pi)^{\frac{N}{2}} |\Sigma|^{\frac{1}{2}}} e^{-\frac{1}{2}(x-\mu)^T \Sigma^{-1} (x-\mu)}$$

$$E[x] = \mu = (E[x_1], \ldots, E[x_N])^T$$

$$Var(x) \rightarrow \Sigma = \begin{pmatrix} Var(x_1) & Cov(x_1, x_2) & \ldots & Cov(x_1, x_N) \\ Cov(x_2, x_1) & Var(x_2) & \ldots & Cov(x_2, x_N) \\ \vdots & & \ddots & \vdots \\ Cov(x_N, x_1) & Cov(x_N, x_2) & \ldots & Var(x_N) \end{pmatrix}$$

Multivariate Gaussians

$$f(x) = \frac{1}{(2\pi)^{\frac{N}{2}}|\Sigma|^{\frac{1}{2}}} e^{-\frac{1}{2}(x-\mu)^T\Sigma^{-1}(x-\mu)}$$

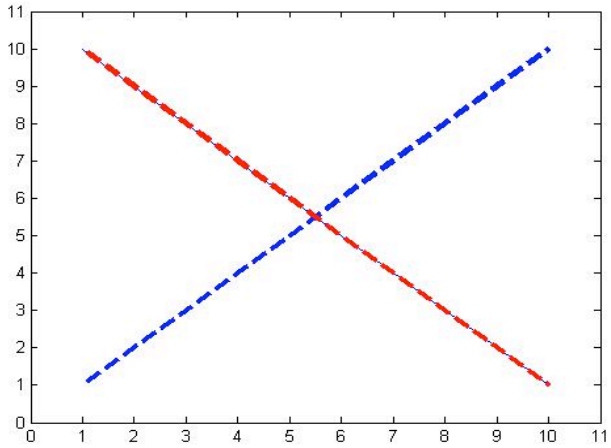
$$E[x] = \mu = (E[x_1], \ldots, E[x_N])^T$$

$$Var(x) \rightarrow \Sigma = \begin{pmatrix} Var(x_1) & Cov(x_1, x_2) & \ldots & Cov(x_1, x_N) \\ Cov(x_2, x_1) & Var(x_2) & \ldots & Cov(x_2, x_N) \\ \vdots & & \ddots & \vdots \\ Cov(x_N, x_1) & Cov(x_N, x_2) & \ldots & Var(x_N) \end{pmatrix}$$

$$\mathrm{cov}(x_1, x_2) = \frac{1}{n}\sum_{i=1}^{n}(x_{1,i} - \mu_1)(x_{2,i} - \mu_2)$$

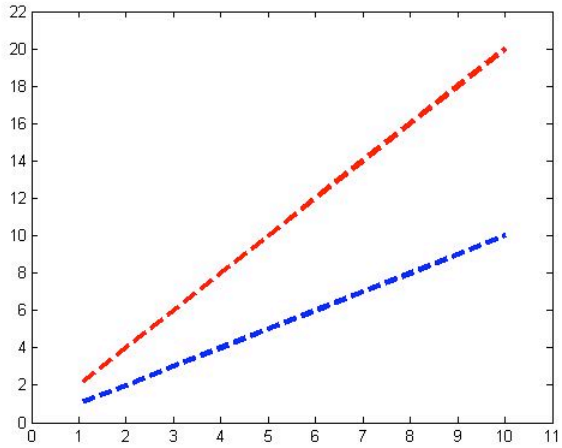
Covariance examples

Anti-correlated



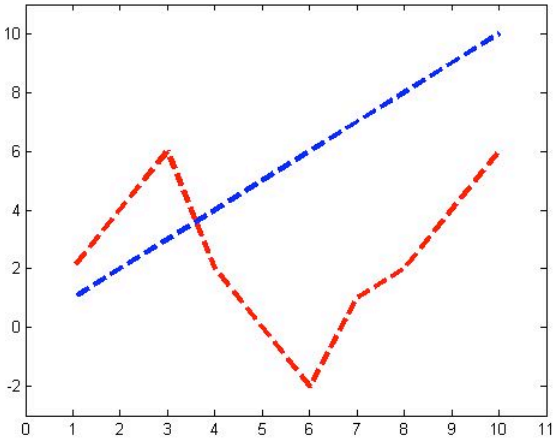
Covariance: -9.2

Correlated



Covariance: 18.33

Independent (almost)



Covariance: 0.6

Independence

- In some cases the additional information does not help

  P(slept) = 0.5

  P(slept | rain = 1) = 0.5

- In this case, the extra knowledge about rain does not change our prediction
- Slept and rain are independent!

Liked movie	Slept	raining	P
1	1	1	0.05
1	0	1	0.1
0	0	1	0.025
0	1	1	0.075
1	1	0	0.15
1	0	0	0.3
0	0	0	0.075
0	1	0	0.225

Independence (cont.)

- Notation: $P(S \mid R) = P(S)$
- Using this we can derive the following:
  - $P(\neg S \mid R) = P(\neg S)$
  - $P(S,R) = P(S)P(R)$
  - $P(R \mid S) = P(R)$

Independence

- Independence allows for easier models, learning and inference
- For our example:
  - P(raining, slept movie) = P(raining)P(slept movie)
  - Instead of 4 by 2 table (4 parameters), only 2 are required
  - The saving is even greater if we have many more variables …
- In many cases it would be useful to assume independence, even if its not the case

Conditional independence

- Two dependent random variables may become independent when conditioned on a third variable:

  $P(A,B \mid C) = P(A \mid C)\, P(B \mid C)$

- Example

  P(liked movie) = 0.5

  P(slept) = 0.4

  P(liked movie, slept) = 0.1

  P(liked movie | long) = 0.4

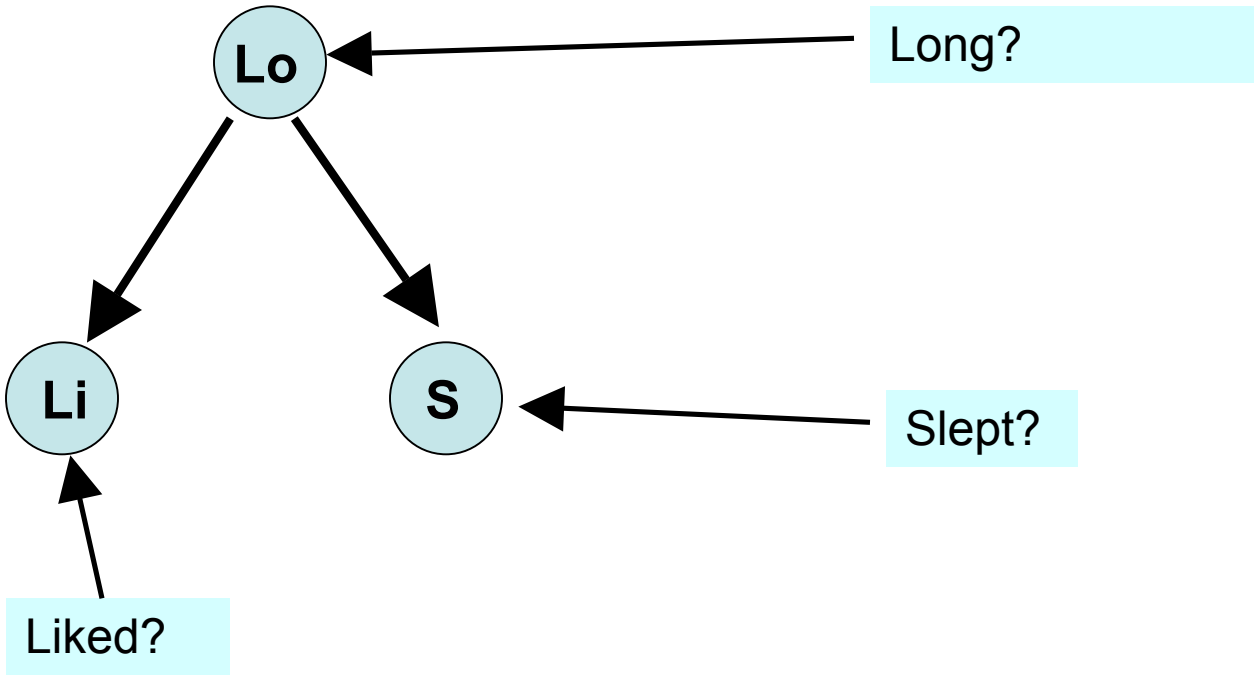
  P(slept | long) 0.6

  P(slept, like movie | long) = 0.24

Given knowledge of length, the two other variables become independent

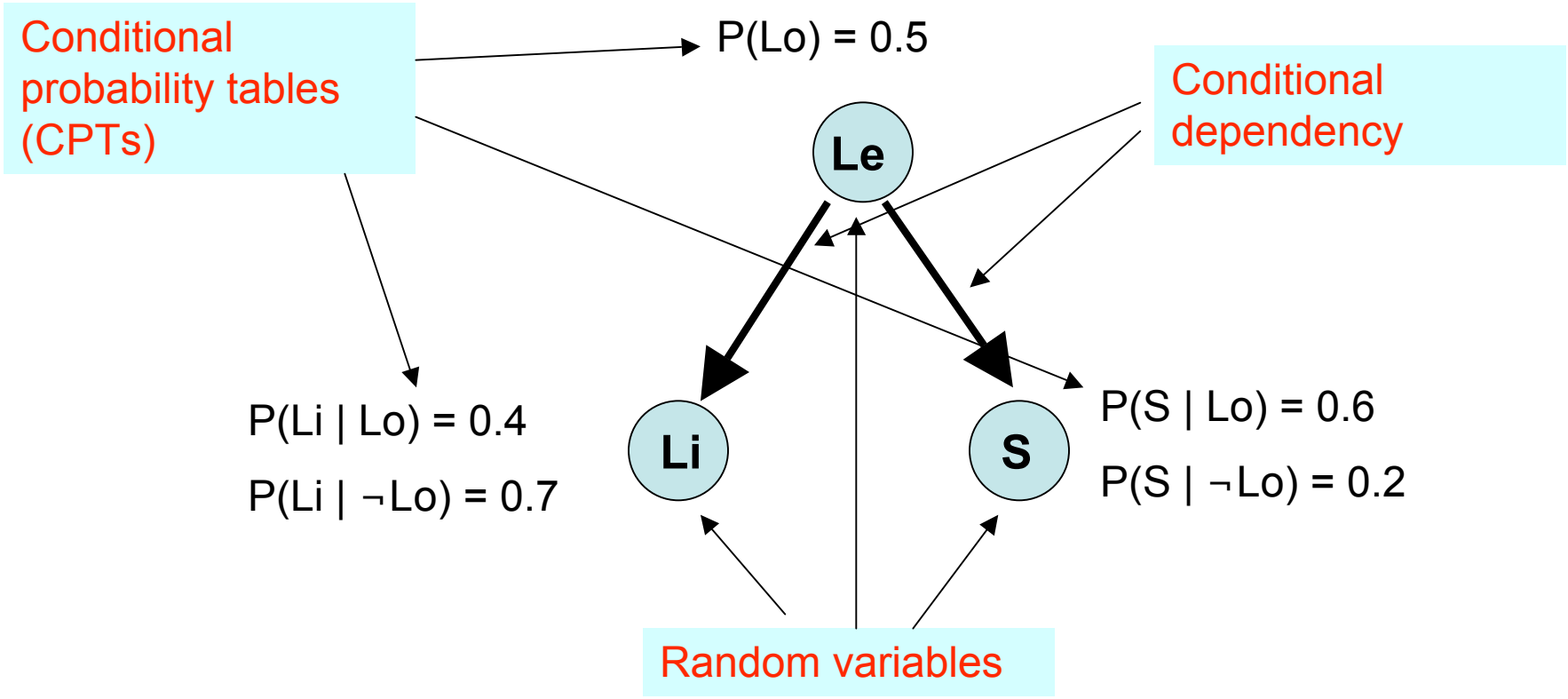
Bayesian networks

- Bayesian networks are *directed graphs* with nodes representing *random variables* and edges representing *dependency assumptions*



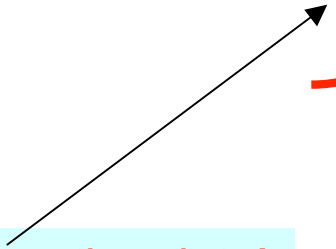
Bayesian networks: Notations

Bayesian networks are directed acyclic graphs.



Constructing a Bayesian network

- How do we go about constructing a network for a specific problem?
- Step 1: Identify the random variables
- Step 2: Determine the conditional dependencies
- Step 3: Populate the CPTs



Can be learned from observation data!

A example problem

- An alarm system

  B – Did a burglary occur?

  E – Did an earthquake occur?

  A – Did the alarm sound off?

  M – Mary calls

  J – John calls

- How do we reconstruct the network for this problem?

Factoring joint distributions

- Using the chain rule we can always factor a joint distribution as follows:

  $P(A,B,E,J,M) =$

  $P(A \mid B,E,J,M)\ P(B,E,J,M) =$

  $P(A \mid B,E,J,M)\ P(B \mid E,J,M)\ P(E,J,M) =$

  $P(A \mid B,E,J,M)\ P(B \mid E, J,M)\ P(E \mid J,M)\ P(J,M)$

  $P(A \mid B,E,J,M)\ P(B \mid E, J,M)\ P(E \mid J,M) P(J \mid M) P(M)$

- This type of conditional dependencies can also be represented graphically.

A Bayesian network

P(A | B,E,J,M) P(B | E, J,M) P(E | J,M)P(J | M)P(M)

Number of parameters:

A: 2^4

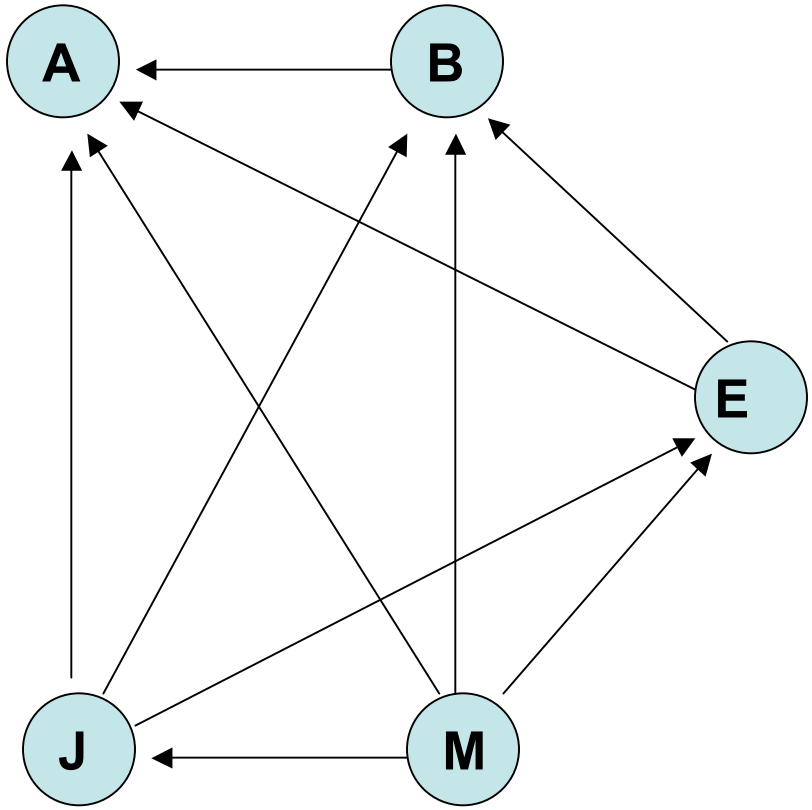
B: 2^3

E: 4

J: 2

M: 1

A total of 31 parameters



A better approach

- An alarm system

  B – Did a burglary occur?

  E – Did an earthquake occur?

  A – Did the alarm sound off?

  M – Mary calls

  J – John calls

- Lets use our knowledge of the domain!

Reconstructing a network

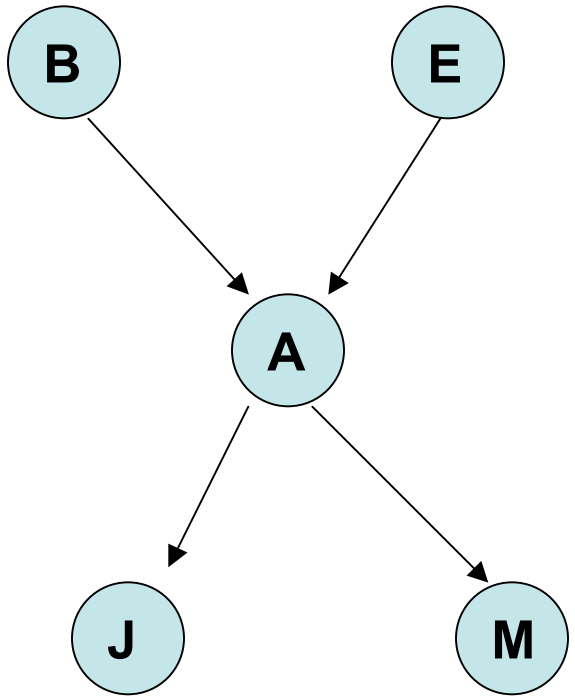
B – Did a burglary occur?

E – Did an earthquake occur?

A – Did the alarm sound off?

M – Mary calls

J – John calls



Reconstructing a network

Number of parameters:

A: 4

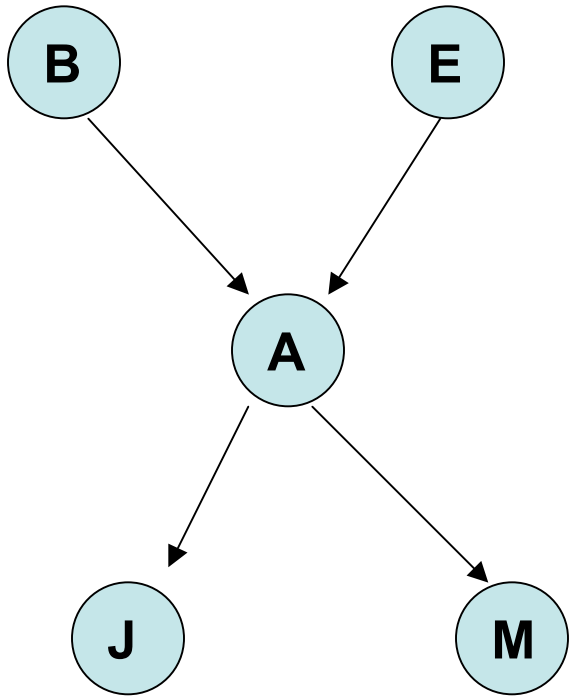
B: 1

E: 1

J: 2

M: 2

A total of 10 parameters



By relying on domain knowledge we saved 21 parameters!

Constructing a Bayesian network: Revisited

- Step 1: Identify the random variables
- Step 2: Determine the conditional dependencies

  - Select on ordering of the variables

  - Add them one at a time

  - For each new variable X added select the minimal subset of nodes as parents such that X is independent from all other nodes in the current network given its parents.
- Step 3: Populate the CPTs

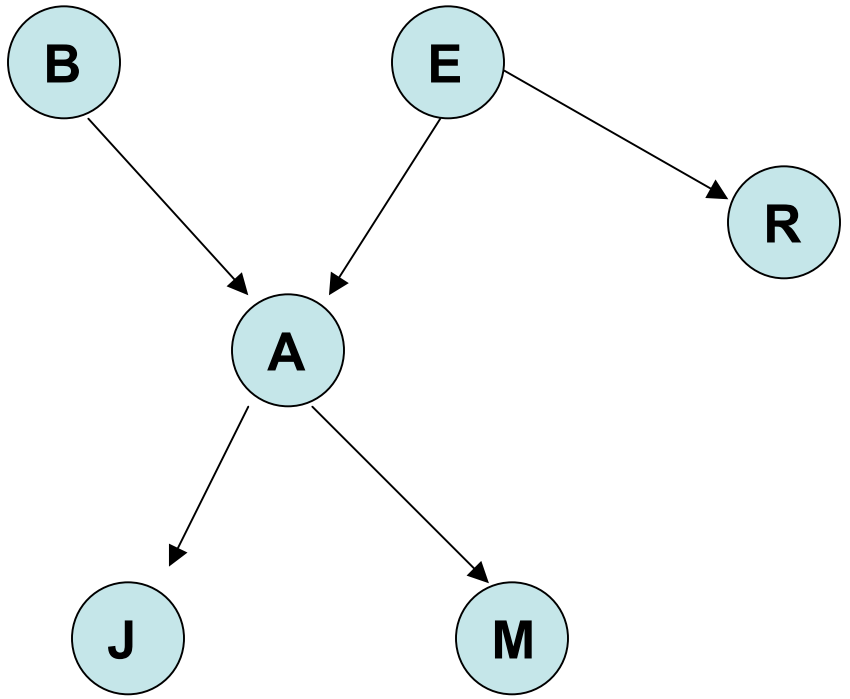
  - We will discuss this when we talk about density estimations

Reconstructing a network

Suppose we wanted to add a new variable to the network:

R – Did the radio announce that there was an earthquake?

How should we insert it?



Bayesian network: Inference

- Once the network is constructed, we can use algorithms for inferring the values of unobserved variables.
- For example, in our previous network the only observed variables are the phone call and the radio announcement. However, what we are really interested in is whether there was a burglary or not.
- How can we determine that?

Inference

- Lets start with a simpler question
  - How can we compute a joint distribution from the network?
  - For example, P(B,¬E,A,J, ¬M)?
- Answer:
  - That's easy, lets use the network

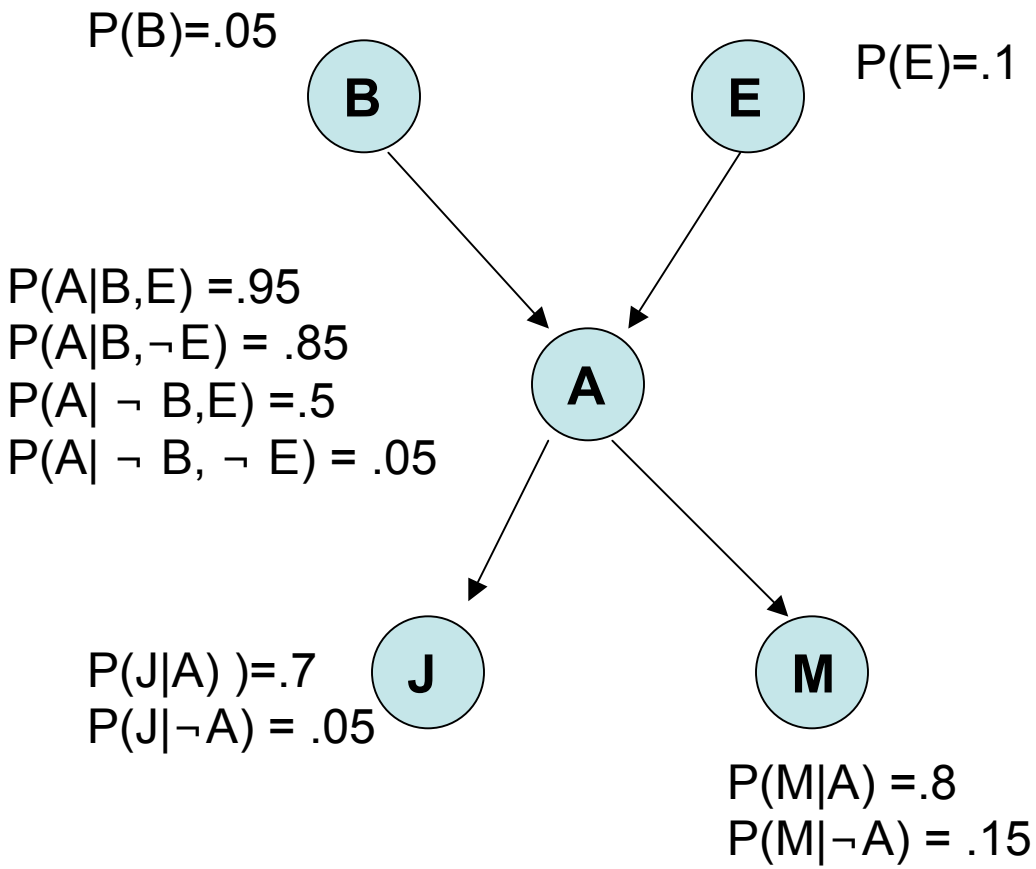
Computing: P(B,¬E,A,J, ¬M)

P(B,¬E,A,J, ¬M) =

P(B)P(¬E)P(A | B, ¬E)
P(J | A)P(¬M | A)

= 0.05*0.9*.85*.7*.2

= 0.005355



Computing: P(B,¬E,A,J, ¬M)

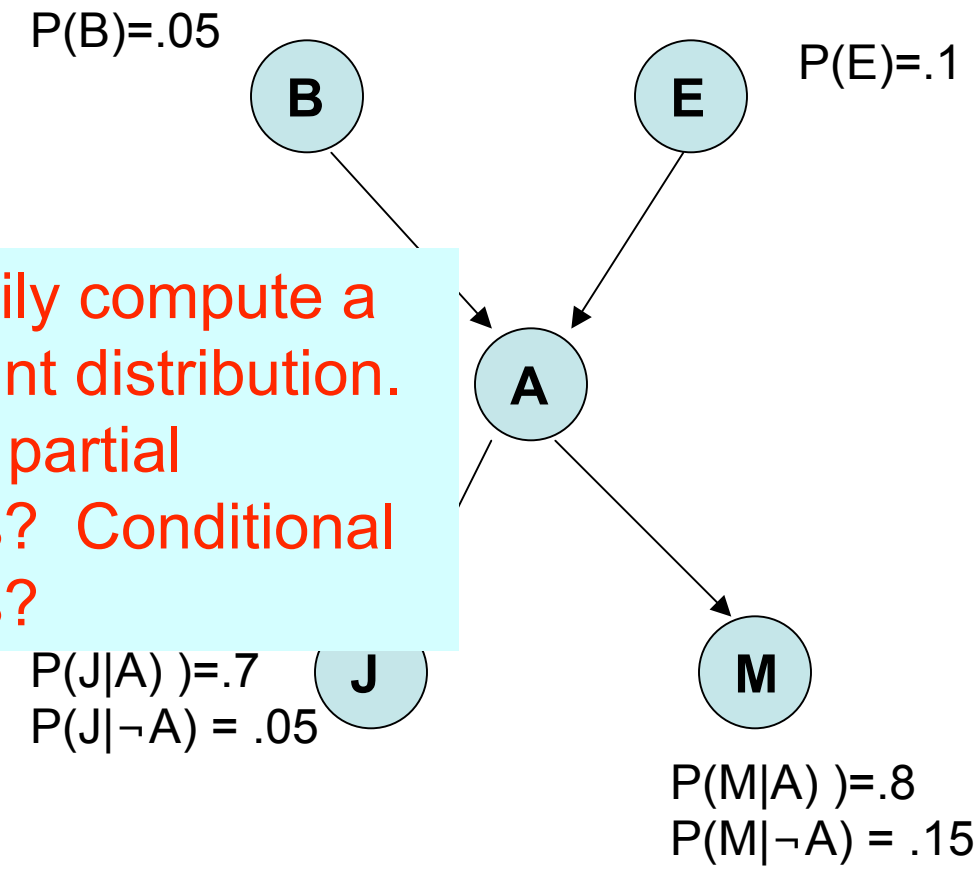
P(B,¬E,A,J, ¬M) =

P(B)P(¬E)P(A | B, ¬E)
P(J | A)P(¬M | A)

= 0.05*0.9*.85*.7* ?

= 0.005355

We can easily compute a complete joint distribution. What about partial distributions? Conditional distributions?

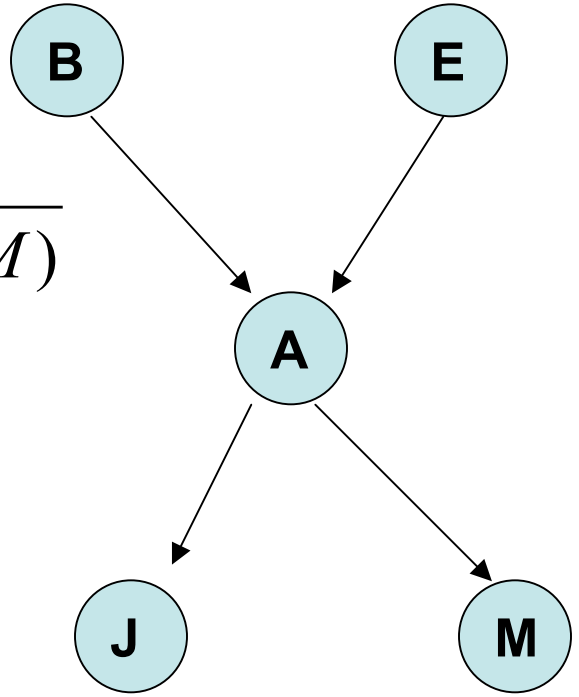


Inference

- We are interested in queries of the form:
  P(B | J,¬M)
- This can also be written as a joint:

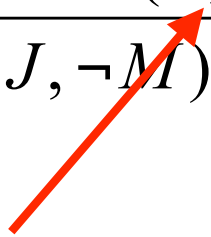
$$P(B \mid J, \neg M) = \frac{P(B, J, \neg M)}{P(B, J, \neg M) + P(\neg B, J, \neg M)}$$

- How do we compute the new joint?



Computing partial joints

$$P(B \mid J, \neg M) = \frac{P(B, J, \neg M)}{P(B, J, \neg M) + P(\neg B, J, \neg M)}$$



Sum all instances with these settings (the sum is over the possible assignments to the other two variables, E and A)

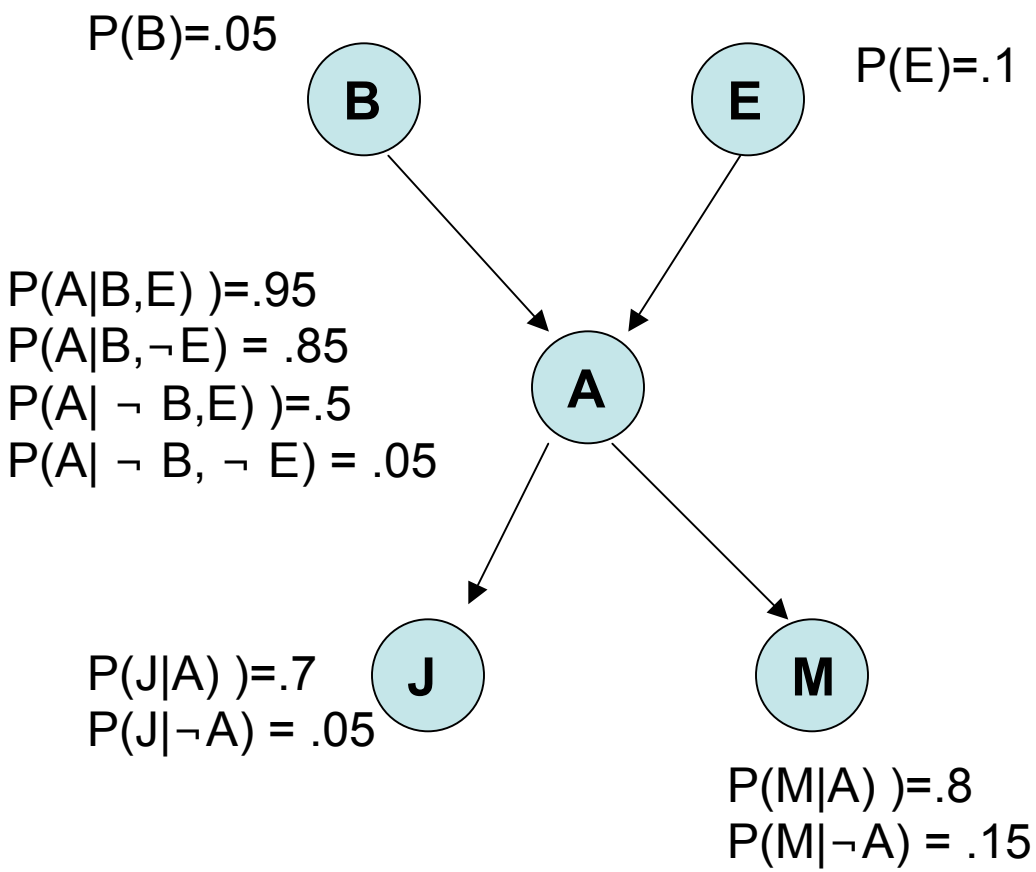
Computing: P(B,J, ¬M)

P(B,J, ¬M) =

P(B,J, ¬M,A,E)+

P(B,J, ¬M, ¬ A,E) +
P(B,J, ¬M,A, ¬ E) +
P(B,J, ¬M, ¬ A, ¬ E) =

0.0007+0.00001+0.005+0.
0003 = 0.00601



Computing partial joints

$$P(B \mid J, \neg M) = \frac{P(B, J, \neg M)}{P(B, J, \neg M) + P(\neg B, J, \neg M)}$$

Sum all instances with these settings (the sum is over the possible assignments to the other two variables, E and A)

- This method can be improved by re-using calculations (similar to dynamic programming)
- Still, the number of possible assignments is exponential in the unobserved variables?
- That is, unfortunately, the best we can do. General querying of Bayesian networks is NP-complete

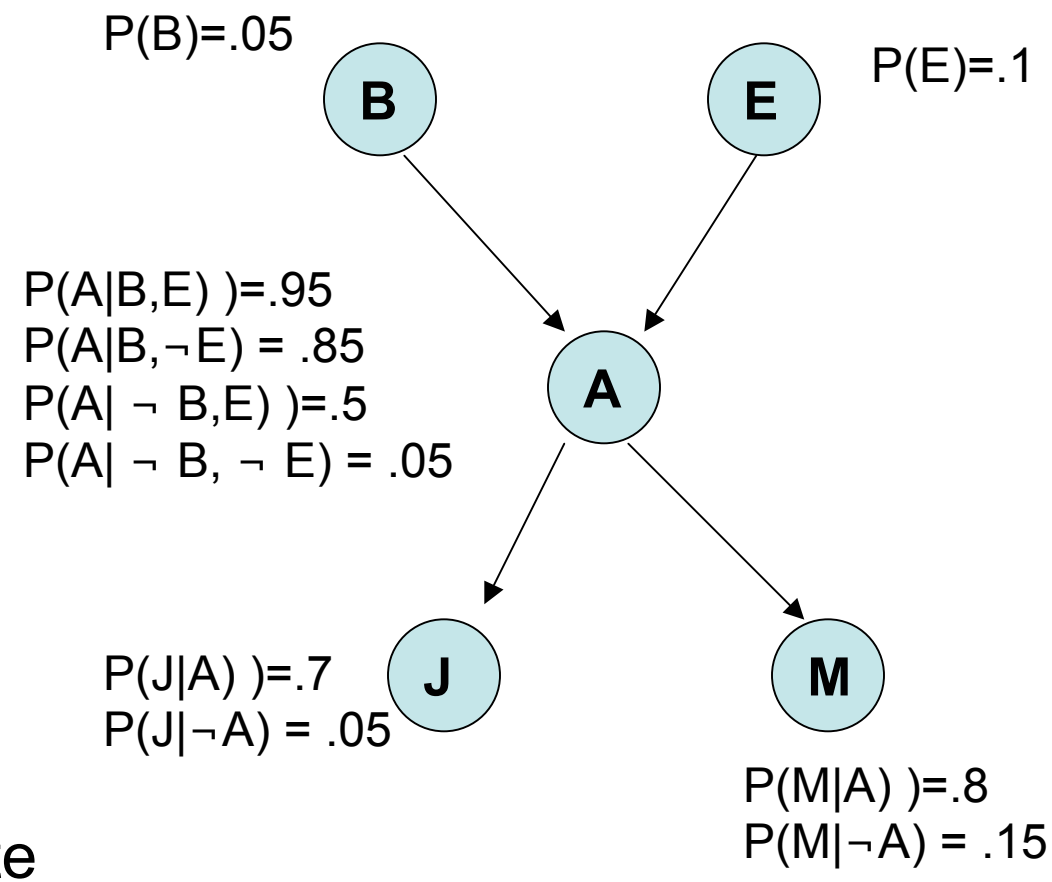
Variable elimination

P(B,J, ¬M) =

P(B,J, ¬M,A,E)+

P(B,J, ¬M, ¬ A,E) +
P(B,J, ¬M,A, ¬ E) +
P(B,J, ¬M, ¬ A, ¬ E) =

0.0007+0.00001+0.005+0.
0003 = 0.00601



- Reuse computations rather than recompute probabilities

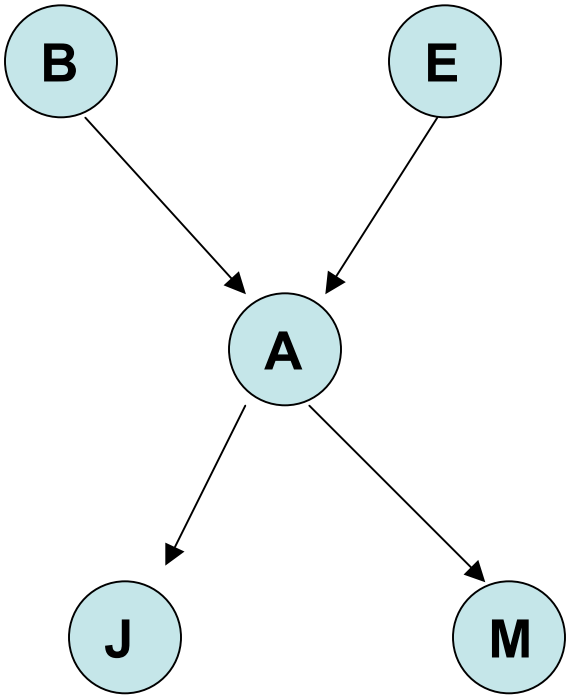
Computing: P(B,J, ¬M)

P(B,J, ¬M) =

P(B,J, ¬M,A,E)+

P(B,J, ¬M, ¬ A,E) +
P(B,J, ¬M,A, ¬ E) +
P(B,J, ¬M, ¬ A, ¬ E) =

$$\sum_a \sum_e P(B)P(e)P(a \mid B,e)P(M \mid a)P(J \mid a)$$



Store as a function of a and use whenever necessary (no need to recompute each time)

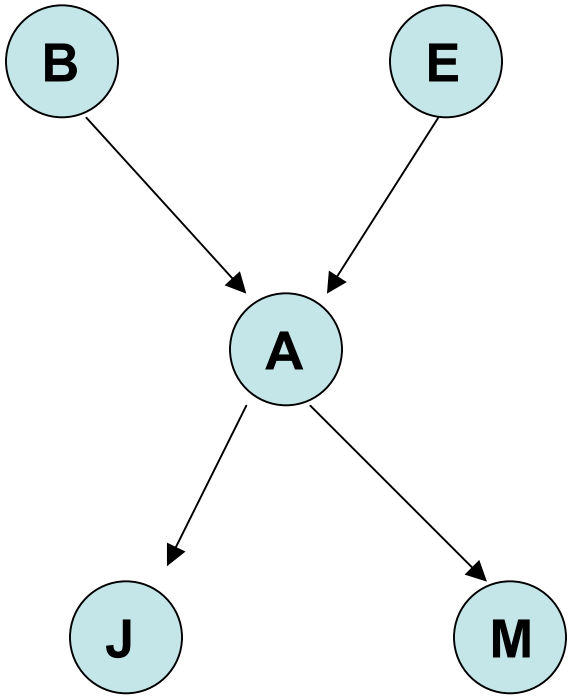
Variable elimination

$$P(B,J,M) = \sum_a \sum_e P(B)P(e)P(a \mid B,e)P(M \mid a)P(J \mid a)$$

$$= P(B)\sum_e P(e)\sum_a P(a \mid B,e)P(M \mid a)P(J \mid a)$$

Set: $f_M(A) = \begin{pmatrix} P(M \mid A) \\ P(M \mid \neg A \end{pmatrix}$

$$f_J(A) = \begin{pmatrix} P(J \mid A) \\ P(J \mid \neg A \end{pmatrix}$$



Variable elimination

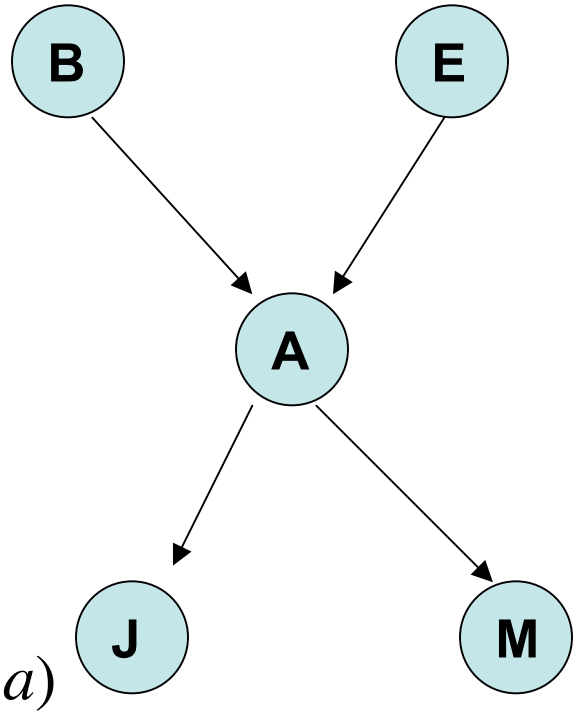
$$P(B,J,M) = \sum_a \sum_e P(B)P(e)P(a \mid B,e)P(M \mid a)P(J \mid a)$$

$$= P(B)\sum_e P(e)\sum_a P(a \mid B,e)P(M \mid a)P(J \mid a)$$

Set: $f_M(A) = \begin{pmatrix} P(M \mid A) \\ P(M \mid \neg A \end{pmatrix}$

$$f_J(A) = \begin{pmatrix} P(J \mid A) \\ P(J \mid \neg A \end{pmatrix}$$

$$P(B,J,M) = P(B)\sum_e P(e)\sum_a P(a \mid B,e) f_M(a) f_J(a)$$



Variable elimination

$$= P(B)\sum_{e} P(e)\sum_{a} P(a \mid B,e) f_M(a) f_J(a)$$

Lets continue with these functions:

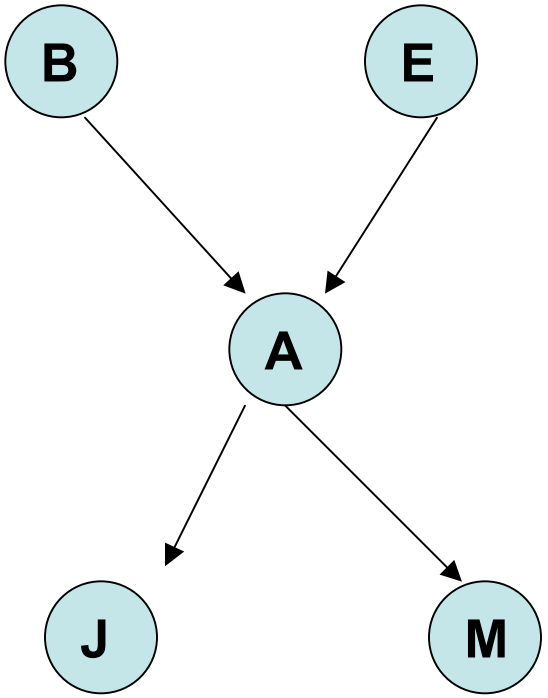
$$f_A(a,B,e) = P(a \mid B,e)$$

We can now define the following function:

$$f_{A,J,M}(B,e) = \sum_{a} f_A(a,B,e) f_J(a) f_M(a)$$

And so we can write:

$$P(B,J,M) = P(B)\sum_{e} P(e) f_{A,J,M}(B,e)$$



Variable elimination

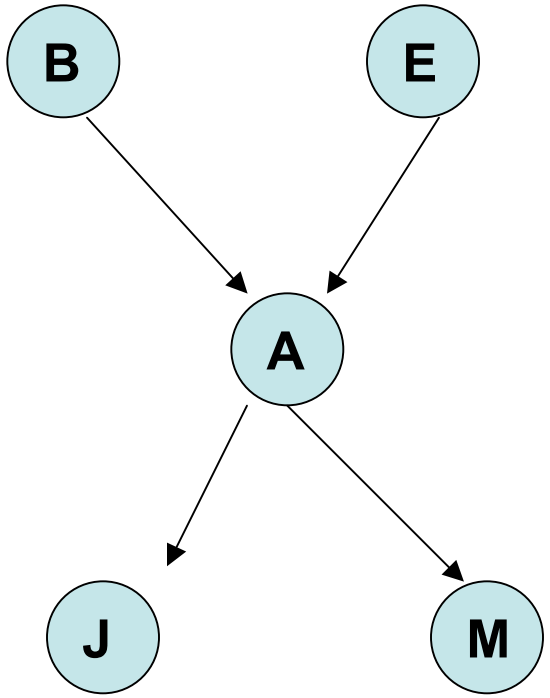
$$P(B,J,M) = P(B)\sum_e P(e) f_{A,J,M}(B,e)$$

Lets continue with another function:

$$f_{E,A,J,M}(B) = \sum_e P(e) f_{A,J,M}(B,e)$$

And finally we can write:

$$P(B,J,M) = P(B) f_{E,A,J,M}(B)$$

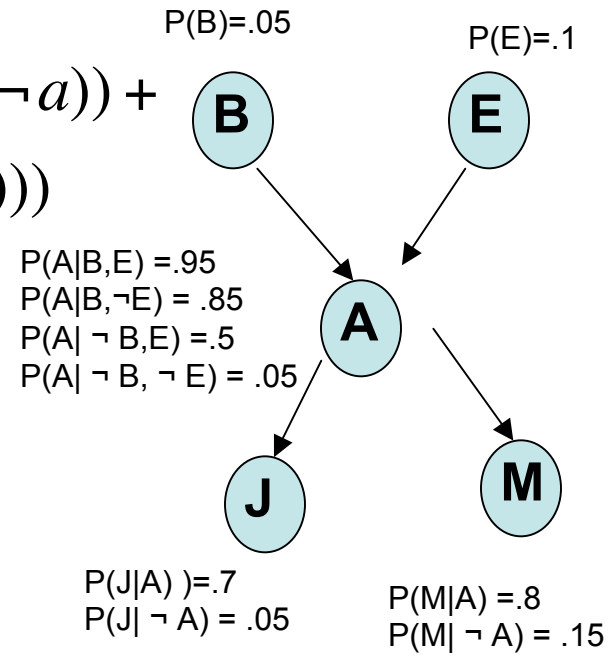


Example

$$P(B,J,M) = P(B) f_{E,A,J,M}(B)$$

$$= 0.05 \sum_e P(e) f_{A,J,M}(B,e) = 0.05(0.1 f_{A,J,M}(B,e) + 0.9 f_{A,J,M}(B,\neg e))$$

$$0.05(0.1(0.95 f_J(a) f_M(a) + 0.05 f_J(\neg a) f_M(\neg a)) +$$
$$0.9(.85 f_J(a) f_M(a) + .15 f_J(\neg a) f_M(\neg a)))$$



Final computation (normalization)

$$P(B \mid J, \neg M) = \frac{P(B, J, \neg M)}{P(B, J, \neg M) + P(\neg B, J, \neg M)}$$

Algorithm

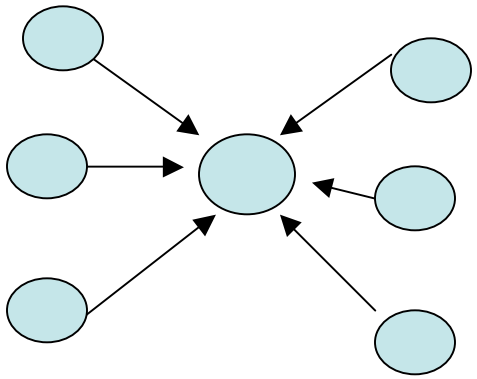
- *e* - evidence (the variables that are known)
- *vars* - the conditional probabilities derived from the network in reverse order (bottom up)
- For each *var* in *vars*

  - *factors* <- make_factor (*var,e*)

  - if *var* is a hidden variable then create a new factor by summing out *var*
- Compute the product of all factors
- Normalize

Computational complexity

- We are reusing computations so we are reducing the running time.
- However, there are still cases in which this algorithm we lead to exponential running time.
- Consider the case of $f_x(y_1 \ldots y_n)$. When factoring x out we would need to account for all possible values of the y's.

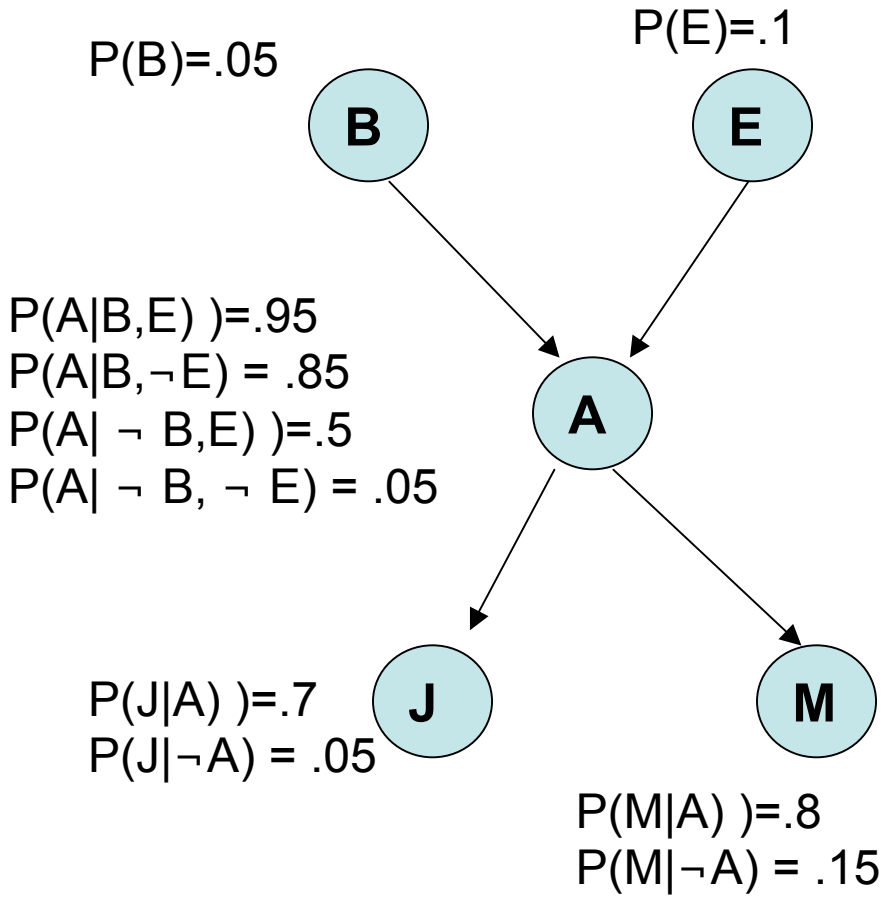


Stochastic inference

- We can easily sample the joint distribution to obtain possible instances

1. Sample the free variable
2. For every other variable:
   - If all parents have been sampled, sample based on conditional distribution

We end up with a new set of assignments for B,E,A,J and M which are a random sample from the joint

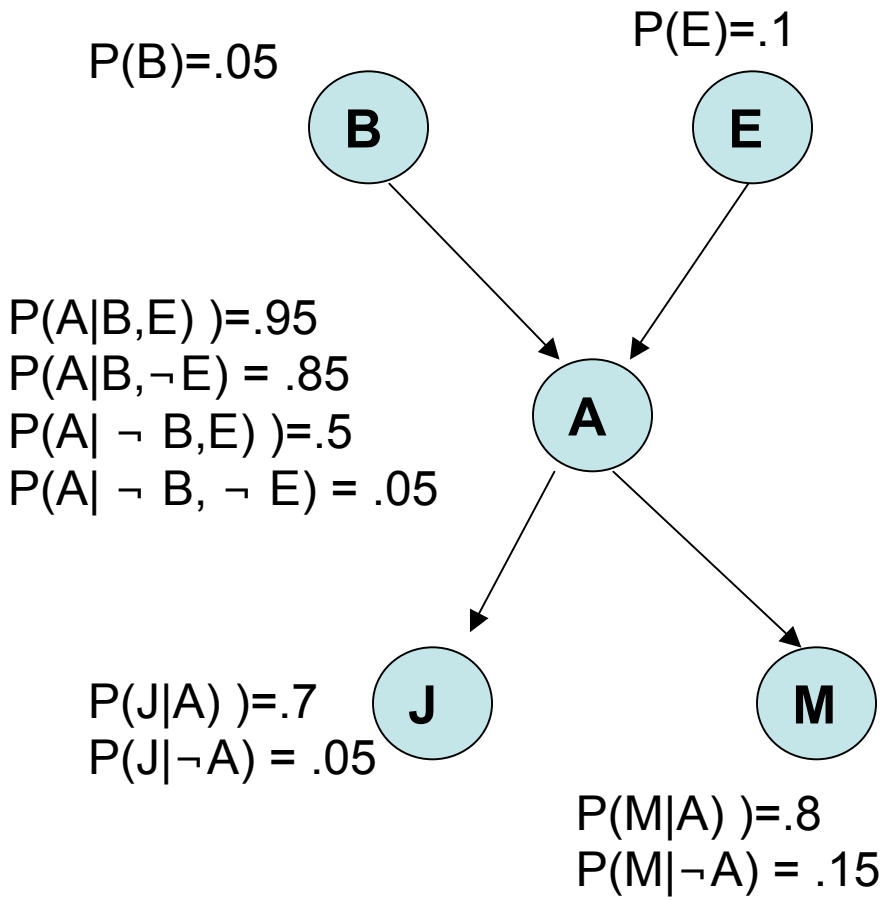


Stochastic inference

- We can easily sample the joint distribution to obtain possible instances

1. Sample the free variable
2. For every other variable:
   - If all parents have been sampled, sample based on conditional distribution

Its always possible to carry out this sampling procedure, why?



Using sampling for inference

- Lets revisit our problem: Compute P(B | J,¬M)
- Looking at the samples we can cound:

  - N: total number of samples

  - N_c : total number of samples in which the condition holds (J,¬M)

  - N_B: total number of samples where the joint is true (B,J,¬M)
- For a large enough N

  - N_c / N ≈ P(J,¬M)

  - N_B / N ≈ P(B,J,¬M)
- And so, we can set

$$P(B \mid J,\neg M) = P(B,J,\neg M) / P(J,\neg M) \approx N_B / N_c$$

Using sampling for inference

- Lets revisit our problem: Compute P(B | J,¬M)
- Looking at the samples we can count:
  - *N*: total number o
  - N_c : total number
  - N_B: total number
- For a large enoug
  - N_c / N ≈ P(J,¬M)
  - N_B / N ≈ P(B,J,¬M)
- And so, we can set

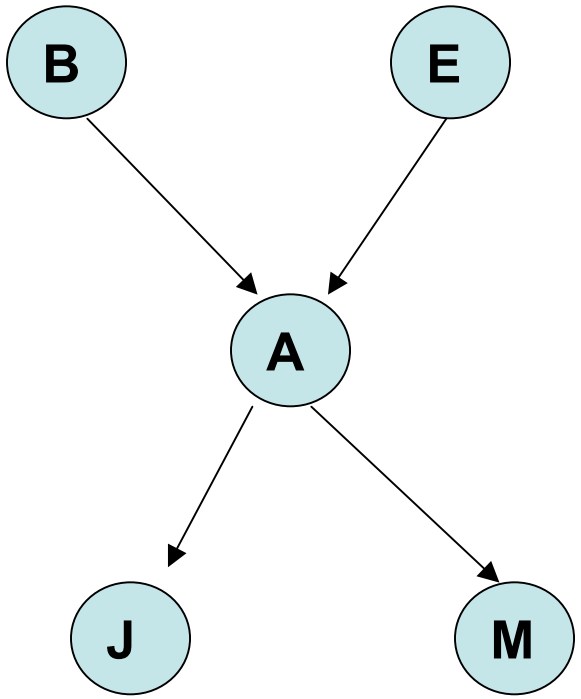
$$P(B \mid J,\neg M) = P(B,J,\neg M) / P(J,\neg M) \approx N_B / N_c$$

Problem: What if the condition rarely happens?

We would need lots and lots of samples, and most would be wasted

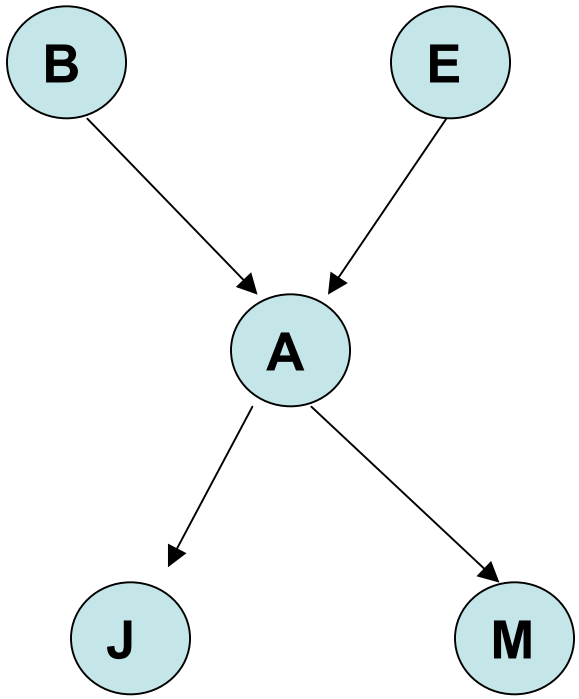
Weighted sampling

- Compute P(B | J,¬M)
- We can manually set the value of J to 1 and M to 0
- This way, all samples will contain the correct values for the conditional variables
- Problems?



Weighted sampling

- Compute P(B | J,¬M)
- Given an assignment to parents, we assign a value of 1 to J and 0 to M.
- We record the *probability* of this assignment ($w = p_1 * p_2$) and we weight the new joint sample by w



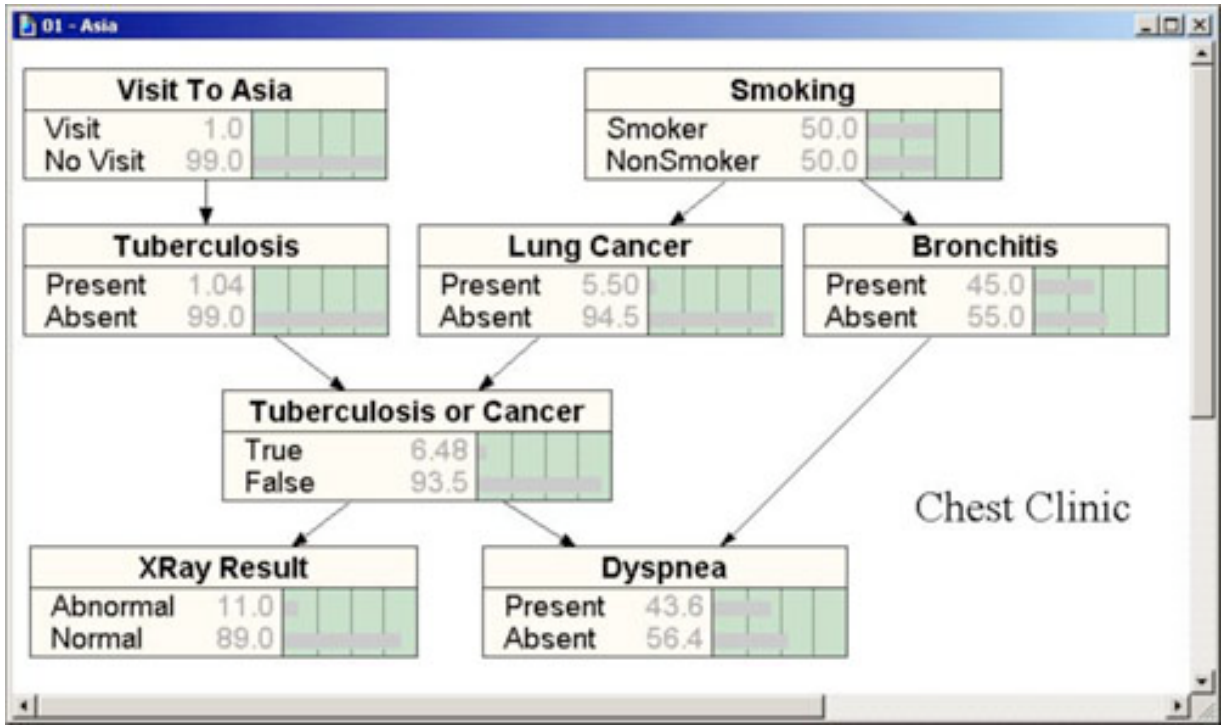
Weighted sampling algorithm for computing P(B | J,¬M)

- Set $N_B, N_c = 0$
- Sample the joint setting the values for *J* and *M*, compute the weight, *w,* of this sample
- $N_c = N_c + w$
- If $B = 1$, $N_B = N_B + w$

- After many iterations, set

$P(B \mid J, \neg M) = N_B / N_c$

Bayesian networks for cancer detection



Important points

- Bayes rule
- Joint distribution, independence, conditional independence
- Attributes of Bayesian networks
- Constructing a Bayesian network
- Inference in Bayesian networks

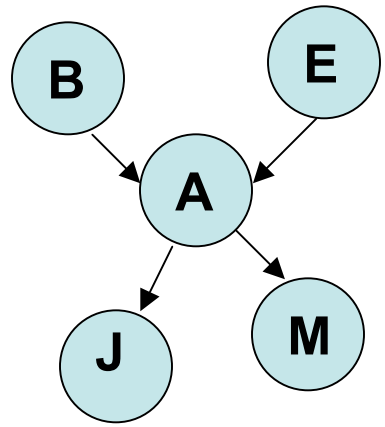
Other inference methods

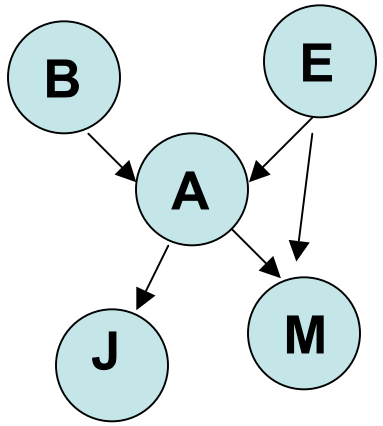
- Convert network to a polytree

  - In a polytree no two nodes have more than one path between them

  - We can convert arbitrary networks to a polytree by clustering (grouping) nodes. For such a graph there is a algorithm which is linear in the number of nodes

  - However, converting into a polytree can result in an exponential increase in the size of the CPTs





Inference in Bayesian networks if NP complete (sketch)

- Reduction from 3SAT
- Recall: 3SAT, find satisfying assignments to the following problem: (a ∨ b ∨ c) ∧ (d ∨ ¬ b ∨ ¬ c) …

What is P(Y=1)?

$P(x_i=1) = 0.5$

$P(x_i=1) = (x_1 \vee x_2 \vee x_3)$

$P(Y=1) = (x_1 \wedge x_2 \wedge x_3 \wedge x_4)$