Lecture 14: Belief (Bayes) Networks

- What kinds of questions do we ask a belief network?
- Exact inference: variable elimination
- Conditional independence in Bayes nets

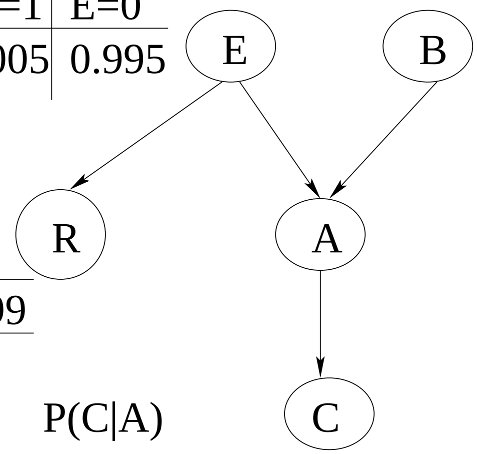
Recall from last week: Belief (Bayesian) Networks

P(E)

E=1	E=0
0.005	0.995

P(B)

B=1	B=0
0.01	0.99



P(R|E)

	R=1	R=0
E=0	0.0001	0.9999
E=1	0.65	0.35

P(A|B,E)

	A=1	A=0
B=0,E=0	0.001	0.999
B=0,E=1	0.3	0.7
B=1,E=0	0.8	0.2
B=1,E=1	0.95	0.05

P(C|A)

	C=1	C=0
A=0	0.05	0.95
A=1	0.7	0.3

- The nodes represent random variables
- The arcs represent “influences”
- At each node, we have a conditional probability distribution for the corresponding variable <u>*given its parents*</u>

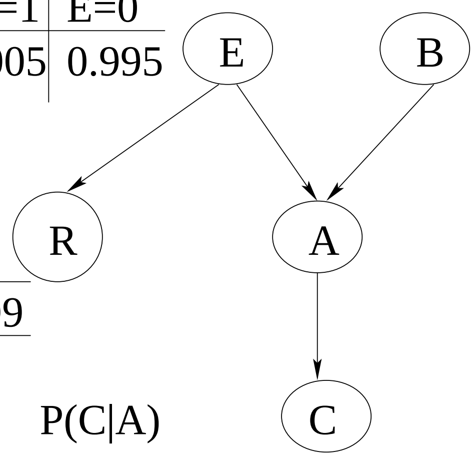
Example

P(E)

E=1	E=0
0.005	0.995

P(B)

B=1	B=0
0.01	0.99



P(R|E)

	R=1	R=0
E=0	0.0001	0.9999
E=1	0.65	0.35

P(A|B,E)

	A=1	A=0
B=0,E=0	0.001	0.999
B=0,E=1	0.3	0.7
B=1,E=0	0.8	0.2
B=1,E=1	0.95	0.05

P(C|A)

	C=1	C=0
A=0	0.05	0.95
A=1	0.7	0.3

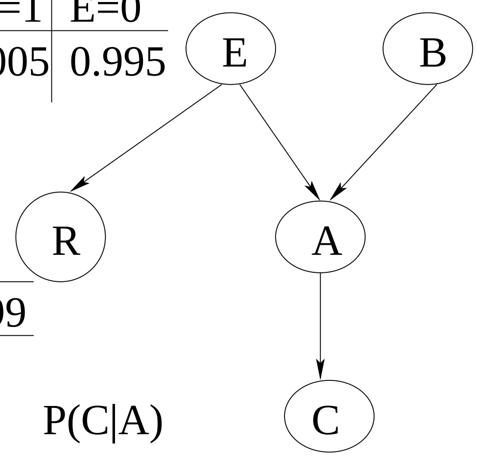
How do you compute $P(E = 1, A = 1, R = 1, B = 0, C = 0)$?

Queries

Graphical models can answer questions about the underlying probability distribution:

- *Unconditional probability queries*: What is the probability of a given value assignment for a subset of variables Y? $P(Y)$
- *Conditional probability queries*: What is the probability of different value assignments for query variables Y given evidence about variables Z? $P(Y|Z = z)$
- *Maximum a posteriori (MAP) queries*: given evidence $Z = z$, what is the most likely assignment of values to the query variables Y:$MAP(Y|Z = z) = \arg\max_y P(Y = y|Z = z)$

Example



P(E)

E=1	E=0
0.005	0.995

P(B)

B=1	B=0
0.01	0.99

P(R|E)

	R=1	R=0
E=0	0.0001	0.9999
E=1	0.65	0.35

P(A|B,E)

	A=1	A=0
B=0,E=0	0.001	0.999
B=0,E=1	0.3	0.7
B=1,E=0	0.8	0.2
B=1,E=1	0.95	0.05

P(C|A)

	C=1	C=0
A=0	0.05	0.95
A=1	0.7	0.3

How do you compute $P(B|C = 1)$?

Examples of MAP queries

- In speech recognition:

  Given a speech signal, *determine* the sequence of words most likely to have generated the signal.

- In text processing:

  Given a paragraph, *determine* what the most likely topic is.

- In medical diagnosis:

  Given a patient, *determine* the most probable diagnosis.

- In robotics:

  Given sensor readings, *determine* the most probable location of the robot.

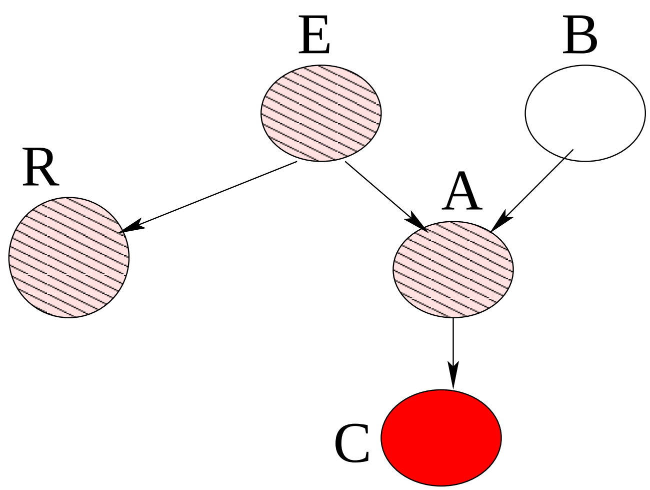
Complexity of inference

- Given a Bayesian network and a random variable X, deciding whether $P(X = x) > 0$ is NP-hard.

  Why?

- *Bad news:* there is no general inference procedure that will work efficiently for all network configurations
- *Good news:* for particular families of networks, inference can be done efficiently.

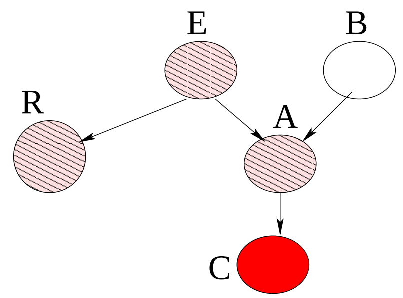
Example



P(B|C=1)=?

$$P(B|C=1) = \frac{P(B, C=1)}{P(C=1)}$$

Naive solution



$$\begin{aligned} P(B=1,C=1) &= \sum_{a,r,e} P(A=a,R=r,E=e,B=1,C=1) \\ &= \sum_{a,r,e} P(r|e)P(e)P(a|e,B=1)P(C=1|a)P(B=1) \end{aligned}$$

$$\begin{aligned} P(B=0,C=1) &= \sum_{a,r,e} P(A=a,R=r,E=e,B=0,C=1) \\ &= \sum_{a,r,e} P(r|e)P(e)P(a|e,B=0)P(C=1|a)P(B=0) \end{aligned}$$

Then $P(C=1) = P(B=1,C=1) + P(B=0,C=1)$.

A better solution

- Re-arrange the sums slighty:

$$\begin{aligned} P(B,C=1) &= \sum_{a,r,e} P(r|e)P(e)P(a|e,B)P(C=1|a)P(B) \\ &= \sum_{a,e} P(e)P(a|e,B)P(B)P(C=1|a)\sum_{r} P(r|e) \end{aligned}$$

- Replace: $\sum_r P(r|e) = m_R(e)$. The notation means: obtained by summing out over R, only depends on variable e.

  (Note that $m_R(e) = 1$, but ignore that for the moment.)

- Now we have:

$$P(B,C=1) = \sum_{a}\sum_{e} P(e)P(a|e,B)P(C=1|a)P(B)m_R(e)$$

- Repeat with other hidden variables (A,E)

Instead of $O(2^n)$ factors, we have to sum over $O(2^k n)$ factors

Basic idea of variable elimination

- We impose an ordering over the variables, with the *query variable* coming *last*
- We maintain a list of “factors”, which depend on given variables
- We *sum* over the variables in the *order* in which they appear in the list
- We *memorize* the result of *intermediate computations*
- This is a kind of *dynamic programming*

A bit of notation

- Let X_i an evidence variable with observed value $\hat{x}_i$
- Let the *evidence potential* be an indicator function:

$$\delta(x_i, \hat{x}_i) = 1 \text{ if and only if } X_i = \hat{x}_i$$

This way, we can turn conditionals into sums as well, e.g.

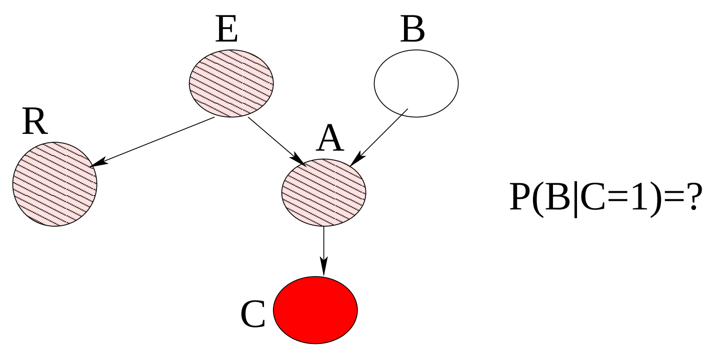
$$P(r|E=1) = \sum_e P(r|e)\delta(e,1)$$

- This is convenient as a notation, but not efficient as a practical implementation

Variable elimination algorithm

1. Pick a *variable ordering* with query variable Y at the end of the list
2. Initialize the *active factor list* with the conditional probability distributions (tables) in the Bayes net
3. Add to the active factor list the evidence potentials $\delta(e, \hat{e})$, for all evidence variables E
4. For $i = 1$ to n
   (a) Take the next variable X_i from the ordering.
   (b) Take all the factors that have X_i as an argument off the active factor list, and multiply them, then sum over all values of X_i, creating a new factor m_{X_i}
   (c) Put m_{X_i} on the active factor list

Example



1. Pick a variable ordering: R, E, C, A, B.
2. Initialize the active factor list and introduce the evidence:

   List: $P(R|E), P(E), P(B), P(A|E, B), P(C|A), \delta(C, 1)$

3. Eliminate R: take $P(R|E)$ off the list, compute $m_R(e) = \sum_r P(r|e)$.

   List: $P(E), P(B), P(A|E, B), P(C|A), \delta(C, 1), m_R(E)$

Example (continued)

4. Eliminate E: $m_E(a,b) = \sum_e P(e)P(a|e,b)m_R(e)$

   List: $P(B), P(C|A), \delta(C,1), m_E(A,B)$

5. Eliminate C: $m_C(a) = \sum_c P(c|a)\delta(C,1)$

   List: $P(B), m_E(A,B), m_C(A)$

6. Eliminate A: $m_A(b) = \sum_a m_E(a,b)m_C(a)$

   List: $P(B), m_A(B)$

7. We compute the answers for $B=1$ and $B=0$, which are $P(B=1)m_A(B=1)$ and $P(B=0)m_A(B=0)$ respectively.

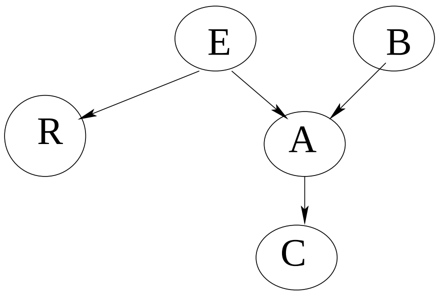
   This is the answer we are looking for!

Complexity of variable elimination

- We need at most $O(n)$ multiplications to create one entry in a factor (where n is the total number of variables)
- If m is the maximum number of values that a variable can take, a factor depending on k variables will have $O(m^k)$ entries
- So it is important to have *small factors*!
- But the size of the factors depends on the ordering of the variables!
- Choosing an optimal ordering is NP-complete

DAGs and independencies

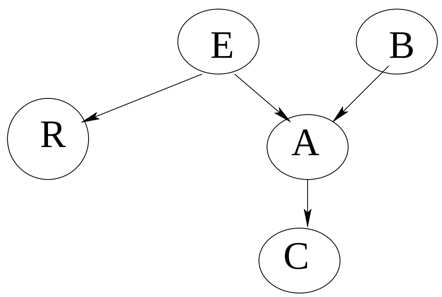
- Given a graph G, what sort of independence assumptions does it imply? E.g. Consider the alarm network:



- In general the *lack of an edge* corresponds to lack of a variable in the conditional probability distribution at a node
- But there are other independencies between variables as well:
  - Is E independent f B?
  - Is R independent of A?
- What variables are independent or conditionally independent in general?

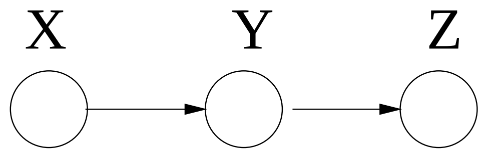
Implied independencies

- Independencies are important because they can help us answer queries more efficiently
- E.g. Suppose that we want to know $P(R|B)$. Do we really need to sum over all values of A, C, E?



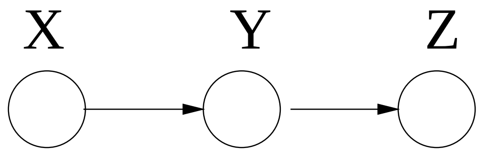
- Given a Bayes net structure G, and evidence for variables Y, what can we say about the sets of variables X and Z?
  – Evidence will propagate along paths in the graph
  – If it reaches both X and Y, then they are *not* independent.

A simple case: Indirect connection



- We interpret the lack of an edge between X and Z as a conditional independence: $P(Z|X,Y) = P(Z|Y)$ and same for X. Is this justified?

A simple case: Indirect connection



- We interpret the lack of an edge between X and Z as a conditional independence: $P(Z|X,Y) = P(Z|Y)$ and same for X. Is this justified?
- Based on the graph structure, we have:

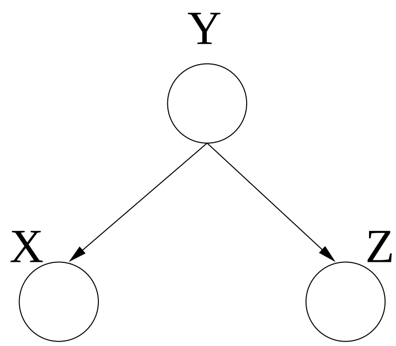
$$P(X,Y,Z) = P(X)P(Y|X)P(Z|Y)$$

- Hence, we have:

$$P(Z|X,Y) = \frac{P(X,Y,Z)}{P(X,Y)} = \frac{P(X)P(Y|X)P(Z|Y)}{P(X)P(Y|X)} = P(Z|Y)$$

- Edges that are *present* do *not imply dependence*.
- Edges that are *missing* do *imply independence*.

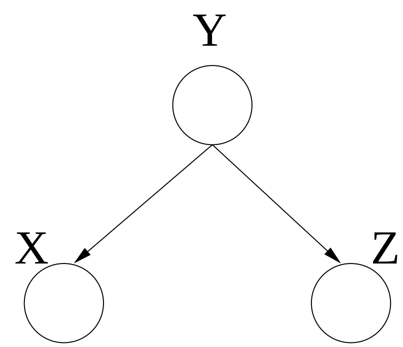
A more interesting case: Common cause



- Again, we interpret the lack of edge between X and Z as conditional independence given Y. Why is this true?

- This is a *hidden variable* scenario: if Y is unknown, then X and Z could appear to be dependent on each other

A more interesting case: Common cause

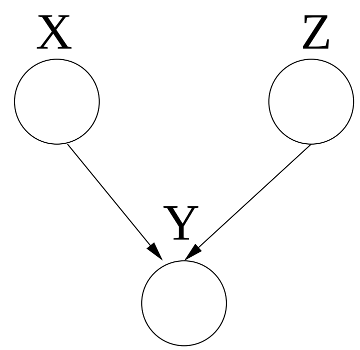


- Again, we interpret the lack of edge between X and Z as conditional independence given Y. Why is this true?

$$P(Z|X,Y) = \frac{P(X,Y,Z)}{P(X,Y)} = \frac{P(Y)P(X|Y)P(Z|Y)}{P(X|Y)P(Y)} = P(Z|Y)$$

- This is a *hidden variable* scenario: if Y is unknown, then X and Z could appear to be dependent on each other

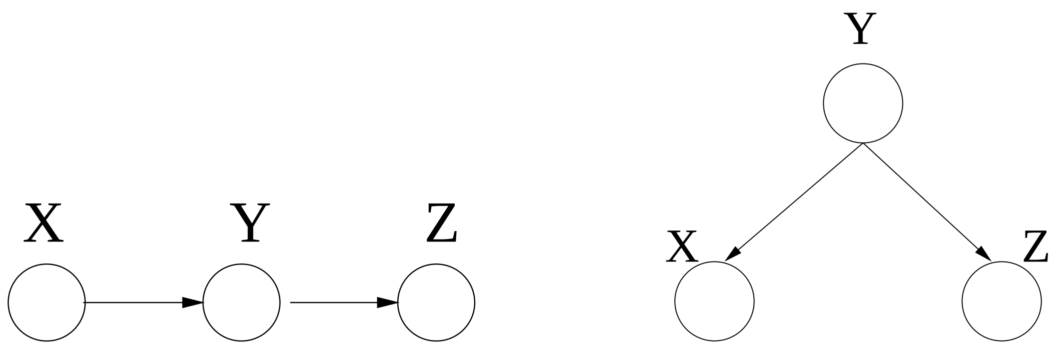
The most interesting case: V-structure



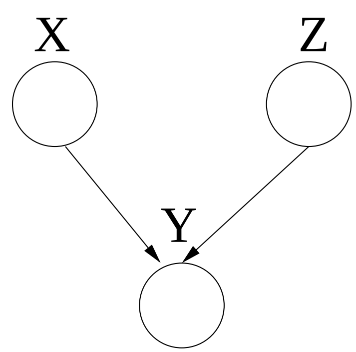
- In this case, the lacking edge between X and Z is a statement of *marginal independence*
- In this case, once we know the value of Y, X is *not independent* of Z

  (You can check that $P(Z|X, Y)$ does not simplify)
- This is the case of *explaining away* when there are multiple, competing explanations.

Summary of the three cases



In both cases, the path between X and Z is open if Y is unknown, but blocked if Y is known



In this case, the path between X and Z is blocked if Y is unknown but open if Y is known

Bayes ball algorithm

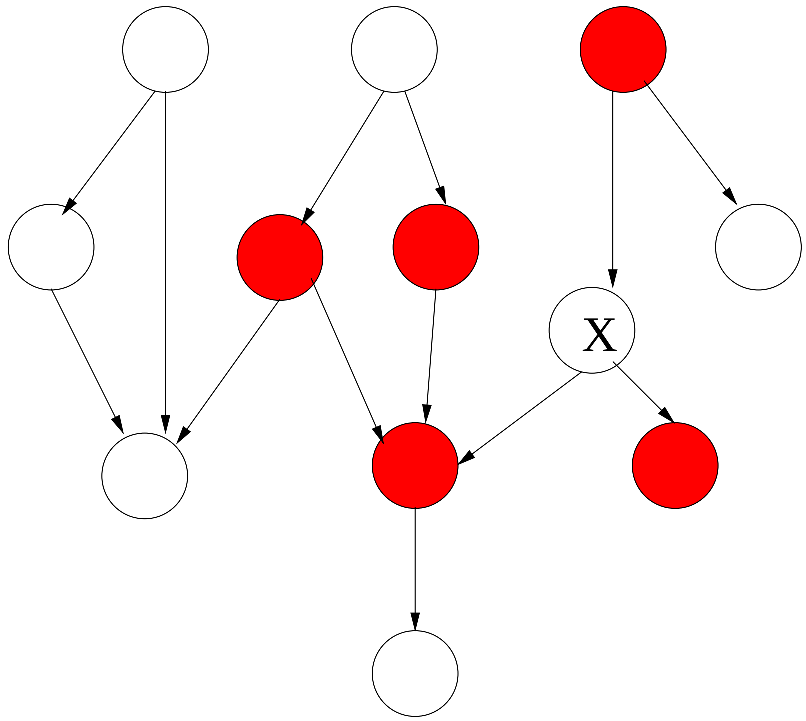
- How can we know whether X is independent of Z given Y for a general Bayes net with corresponding graph G?
- Algorithm (Pearl):
  - Shade all nodes in the evidence set Y
  - Put balls in all the nodes in X, and we let them bounce around the graph according to the rules from the three base cases
  - Note that the balls can go in any direction along an edge!
  - If any ball reaches some node in Z, then the conditional independence assertion is *not* true.

Independence in a general Bayes net

- Any network can be treated as a collection made from these three base cases.
- Bayes ball can be used to assert the conditional independence of different nodes given evidence
- In general, a node will be independent of the rest of the network given:
  - its parents
  - its children
  - its ”spouses” (other parents of its children)

  These form the *Markov blanket* of the node.

Example of Markov blanket



The red nodes are the Markov blanket for X

Summary of inference in Bayes nets

- The complexity of inference depends a lot on the structure of the network
  - Inference can be done efficiently (polynomial time) for tree-structured networks
  - In the worst case, inference is NP-complete
- The best exact inference algorithm converts the network to a tree, then does exact inference on the tree
- In practice, for large nets, approximate inference methods work much better
- More about this in COMP-526.