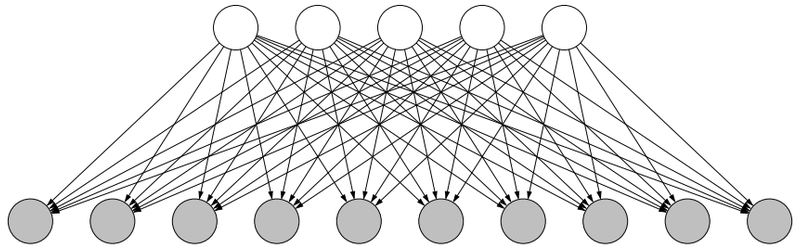
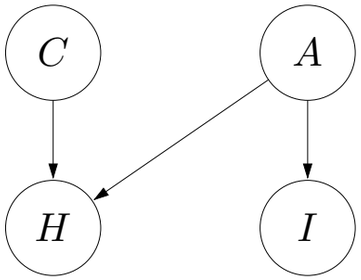
Bayesian networks: probabilistic inference



- In this module, I will talk about a strategy for performing probabilistic inference in general Bayesian networks.

Review: Bayesian network

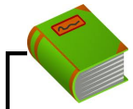


Random variables:

cold C, allergies A, cough H, itchy eyes I

Joint distribution:

$$\mathbb{P}(C = c, A = a, H = h, I = i) = p(c)p(a)p(h \mid c, a)p(i \mid a)$$



Definition: Bayesian network

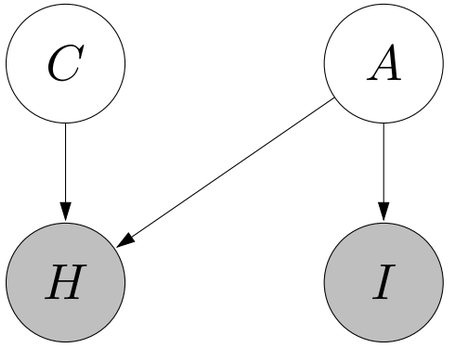
Let $X = (X_1, \ldots, X_n)$ be random variables.

A **Bayesian network** is a directed acyclic graph (DAG) that specifies a joint distribution over X as a product of local conditional distributions, one for each node:

$$\mathbb{P}(X_1 = x_1, \ldots, X_n = x_n) \stackrel{\text{def}}{=} \prod_{i=1}^{n} p(x_i \mid x_{\text{Parents}(i)})$$

- Recall that a Bayesian network is given by (i) a set of random variables, (ii) directed edges between those variables capturing qualitative dependencies, (iii) local conditional distributions of each variable given its parents which captures these dependencies quantitatively, and (iv) a joint distribution which is produced by multiplying all the local conditional distributions together.

Review: probabilistic inference



Question: $\mathbb{P}(C \mid H = 1, I = 1)$

Input

Bayesian network: $\mathbb{P}(X_1, \ldots, X_n)$

Evidence: $E = e$ where $E \subseteq X$ is subset of variables

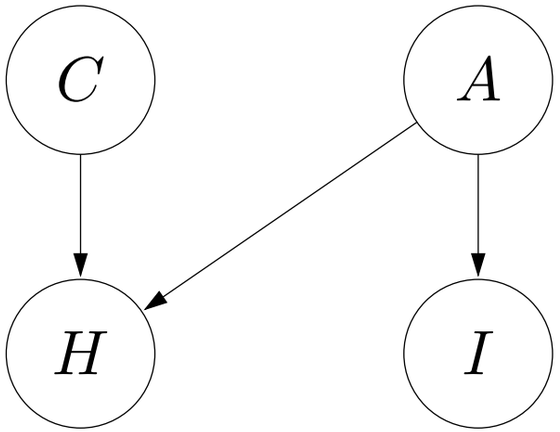
Query: $Q \subseteq X$ is subset of variables

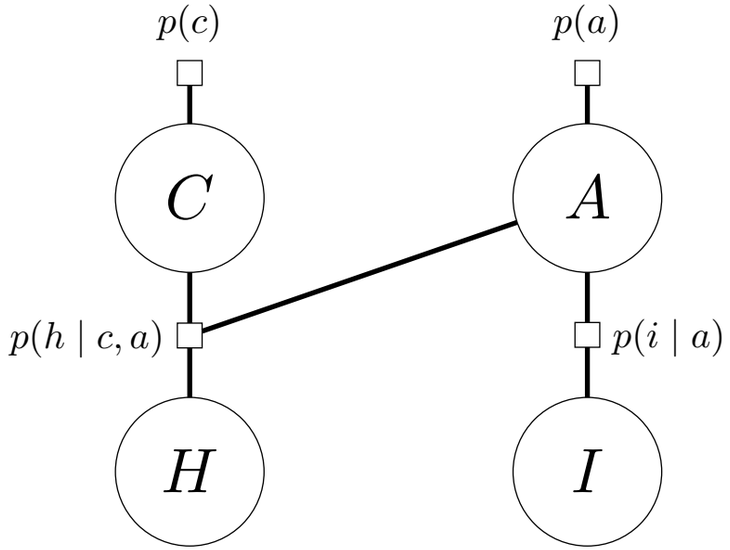
Output

$\mathbb{P}(Q \mid E = e)$ $\longleftrightarrow$ $\mathbb{P}(Q = q \mid E = e)$ for all values q

- Given the joint distribution representing your probabilistic database, you can answer all sorts of questions on it using probabilistic inference.
- Given a set of evidence variables and values, a set of query variables, we want to compute the probability of the query variables given the evidence, marginalizing out all other variables.

Reduction to Markov networks





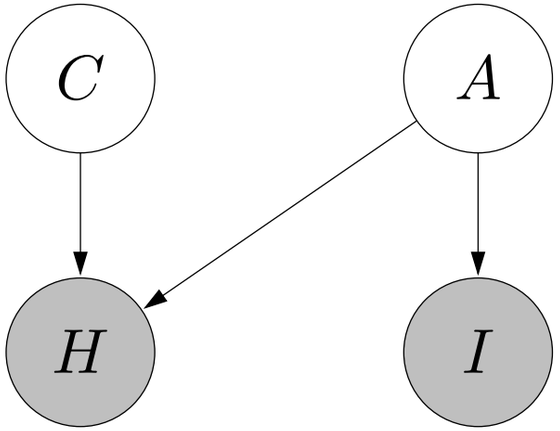
$$\mathbb{P}(C = c, A = a, H = h, I = i) = \frac{1}{Z} p(c)p(a)p(h \mid c, a)p(i \mid a)$$

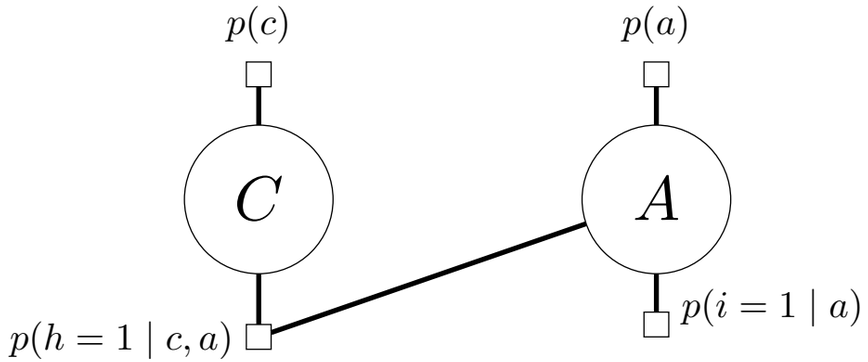
Bayesian network = Markov network with normalization constant $Z = 1$

Reminder: single factor that connects **all** parents!

- Our overarching strategy for performing inference in Bayesian networks is to convert them into Markov networks.
- Recall that the joint distribution is just the product of all the local conditional distributions. The local conditional distributions (e.g., $p(a \mid b, e)$) are all non-negative so they can be interpreted as simply factors in a factor graph.
- Recall that a Markov network defines the joint distribution as the product of all the factors divided by some normalization constant Z. But in this case, $Z = 1$ because the factors are local conditional distributions of a Bayesian network! Put it another way, Bayesian networks are just instances of Markov networks where the normalization constant $Z = 1$.
- It's important to remember that there is a single factor that connects all the parents. Don't let the directed graph in the Bayesian network deceive you into thinking that there are two factors, one per arrow, which is a common mistake.
- Now we can run any inference algorithm for Markov networks (e.g., Gibbs sampling) on this so-called Markov network and obtain quantities such as $\mathbb{P}(H = 1)$. But there is one important thing that's missing, which is the ability to condition on evidence...

Conditioning on evidence





Markov network:

$$\mathbb{P}(C = c, A = a \mid H = 1, I = 1) = \frac{1}{Z} p(c) p(a) p(h = 1 \mid c, a) p(i = 1 \mid a)$$

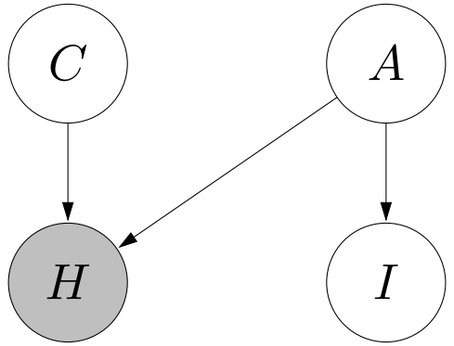
Bayesian network with evidence = Markov network with $Z = \mathbb{P}(H = 1, I = 1)$

Solution: run any inference algorithm for Markov networks (e.g., Gibbs sampling)!

[demo]

- Suppose we condition on evidence $H = 1$ and $I = 1$.
- We can define a new Markov network over the remaining variables (C and A) by simply plugging in the values to H and I. The normalization constant Z is the sum over all values of C and A, which is no longer 1, but rather the probability of the evidence $\mathbb{P}(H = 1, I = 1)$.
- To understand why this relationship holds, recall that the desired conditional probability is the joint probability over the marginal probability. The factors simply represent the joint probability, and thus the normalization constant must be the marginal probability.
- Now we can again run any inference algorithm for Markov networks (e.g., Gibbs sampling), and this allows us to do probabilistic inference in any Bayesian network.
- In the demo, we will run Gibbs sampling to compute $\mathbb{P}(C = 1 \mid H = 1, I = 1)$, and we see that it converges to the right answer (0.13).

Leveraging additional structure: unobserved leaves



Markov network:

$$\mathbb{P}(C = c, A = a, I = i \mid H = 1) = \frac{1}{Z} p(c)p(a)p(h = 1 \mid c, a)p(i \mid a),$$

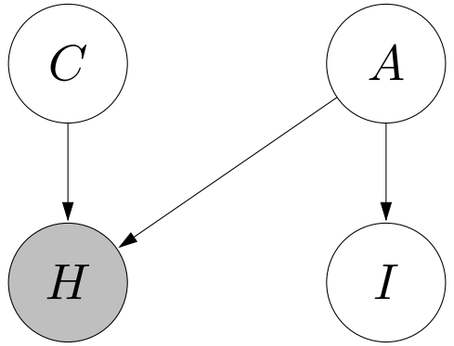
where $Z = \mathbb{P}(H = 1)$

Question: $\mathbb{P}(C = 1 \mid H = 1)$

Can we reduce the Markov network before running inference?

- We could stop there, but there are two more ways we can leverage the structure of Bayesian networks to optimize things a bit.
- Suppose we are now just conditioning on $H = 1$. As before we can form a Markov network over the remaining variables.
- But what if we knew we were only interested in $\mathbb{P}(C = 1 \mid H = 1)$?
- Is there a way to reduce the size of the Markov network before running inference?

Leveraging additional structure: unobserved leaves



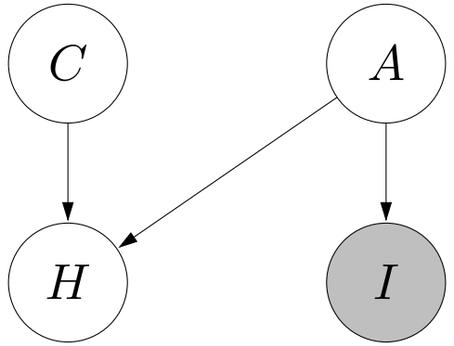
Markov network:

$$\begin{aligned}\mathbb{P}(C = c, A = a \mid H = 1) &= \textstyle\sum_i \mathbb{P}(C = c, A = a, I = i \mid H = 1) \\ &= \textstyle\sum_i \frac{1}{Z} p(c)p(a)p(h = 1 \mid c, a)p(i \mid a) \\ &= \frac{1}{Z} p(c)p(a)p(h = 1 \mid c, a) \textstyle\sum_i p(i \mid a) \\ &= \frac{1}{Z} p(c)p(a)p(h = 1 \mid c, a)\end{aligned}$$

Throw away any unobserved leaves before running inference!

- The answer is yes.
- Let us try marginalizing out I. We expand using the definition of marginal probability, definition of the Bayesian network, pushing the $\sum_i$ inwards past factors that don't depend on i, and noting that $\sum_i p(i \mid a) = 1$ by definition of local conditional distributions.
- But if we stare at the last equation, it is what we would have gotten if we had just ignored I in the first place!
- The general principle here is that marginalization of any unobserved leaf node produces 1, and thus all such nodes can be simply ignored. And we can keep on iterating this until all leaves are observed.
- This is practically very useful because it means that whenever we have a large Bayesian network, we might be able to remove large swaths of the network.
- This property establishes a bridge between marginalization (algebraic operations, usually involves hard work) with removal (graph operations, usually more intuitive).

Leveraging additional structure: independence



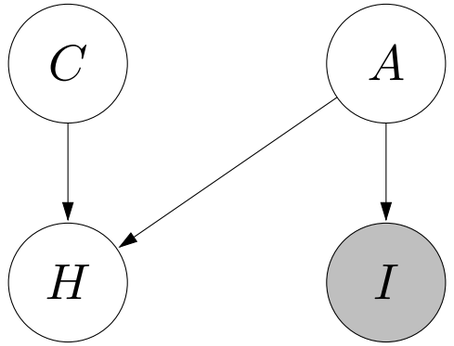
Markov network:

$$\begin{aligned}
\mathbb{P}(C = c \mid I = 1) &= \textstyle\sum_{a,h} \mathbb{P}(C = c, A = a, H = h \mid I = 1) \\
&= \textstyle\sum_{a,h} \frac{1}{Z} p(c) p(a) p(h \mid c, a) p(i = 1 \mid a) \\
&= \textstyle\sum_{a} \frac{1}{Z} p(c) p(a) p(i = 1 \mid a) \\
&= p(c) \textstyle\sum_{a} \frac{1}{Z} p(a) p(i = 1 \mid a) \\
&= p(c)
\end{aligned}$$

Throw away any disconnected components before running inference!

- There is another type of structure we can exploit, which is not specific to Bayesian networks, but shows up generally in Markov networks.
- Suppose we now condition on $I = 1$. Let us expand the marginal probability into the joint probability, expand into the local conditional probabilities, marginalize out the unobserved leaf H using the same idea we just discussed,
- Now at this point, C is completely disconnected from A and I. Algebraically, we can pull $p(c)$ out of the expression.
- We have this mess involving a and i, but this quantity does not depend on c so it is a constant. In this case, we know this constant must be 1 because both $p(c)$ and the LHS are probability distributions.
- So we can throw away any disconnected components. Note that it is advantageous to do this after removing all unobserved leaves, because removing those leaves can help disconnect the graph, as it did in this example.
- Now we have a Markov network, and we would run a standard inference algorithm on it. But in this case, it only has one factor which is already a local probability distribution, so we're done.

Summary



- Condition on evidence (e.g., $I = 1$)
- Throw away unobserved leaves (e.g., H)
- Throw away disconnected components (e.g., A and I)
- Define Markov network out of remaining factors
- Run your favorite inference algorithm (e.g., manual, Gibbs sampling)

- In summary, we tackled the problem of how to perform probabilistic inference in Bayesian networks, by reducing the problem to that of inference in Markov networks.
- To prepare the Markov network, we condition on the evidence (substitute the values into the factors), throw away any unobserved leaves, and throw away any disconnected components.
- Then we just define the Markov network over the remaining factors. If the resulting Markov network is small enough, we can do inference manually. Otherwise, we can run an algorithm like Gibbs sampling.