Introduction to Artificial Intelligence

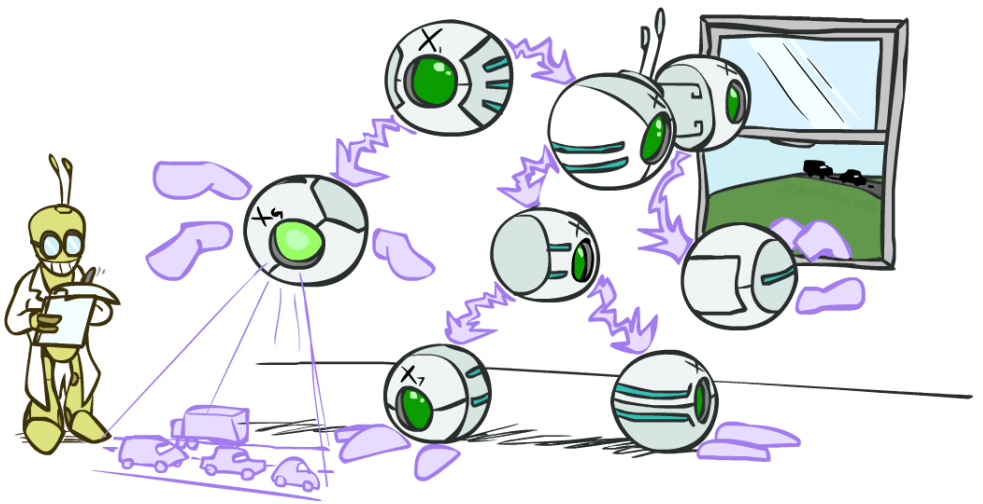
Lecture 5: Probabilistic reasoning

Prof. Gilles Louppe
g.louppe@uliege.be

LIÈGE université

Today

- Bayesian networks
  - Semantics
  - Construction
  - Independence relations
- Inference
- Parameter learning



—

Image credits: CS188, UC Berkeley.

Representing uncertain knowledge

The explicit representation of the joint probability distribution grows exponentially with the number of variables.

Independence and conditional independence assumptions reduce the number of probabilities that need to be specified. They can be represented explicitly in the form of a Bayesian network.

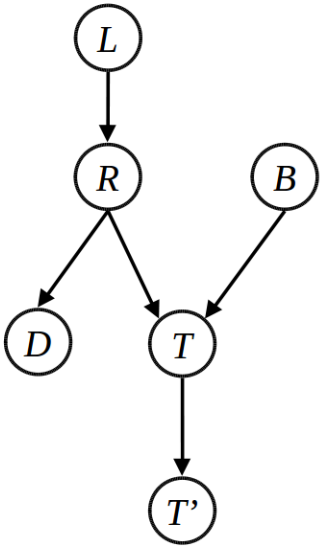
Bayesian networks

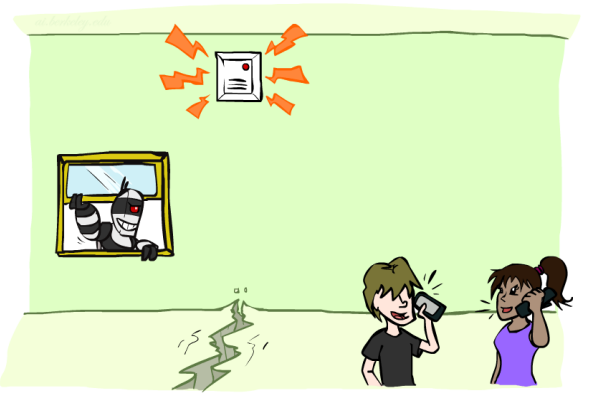
A Bayesian network is a **directed acyclic graph** where

- each node corresponds to a random variable;
  - observed or unobserved
  - discrete or continuous
- each edge is directed and indicates a direct probabilistic dependency between two variables;
- each node X_i is annotated with a conditional probability distribution

$$\mathbf{P}(X_i|\text{parents}(X_i))$$

that defines the distribution of X_i given its parents in the network.



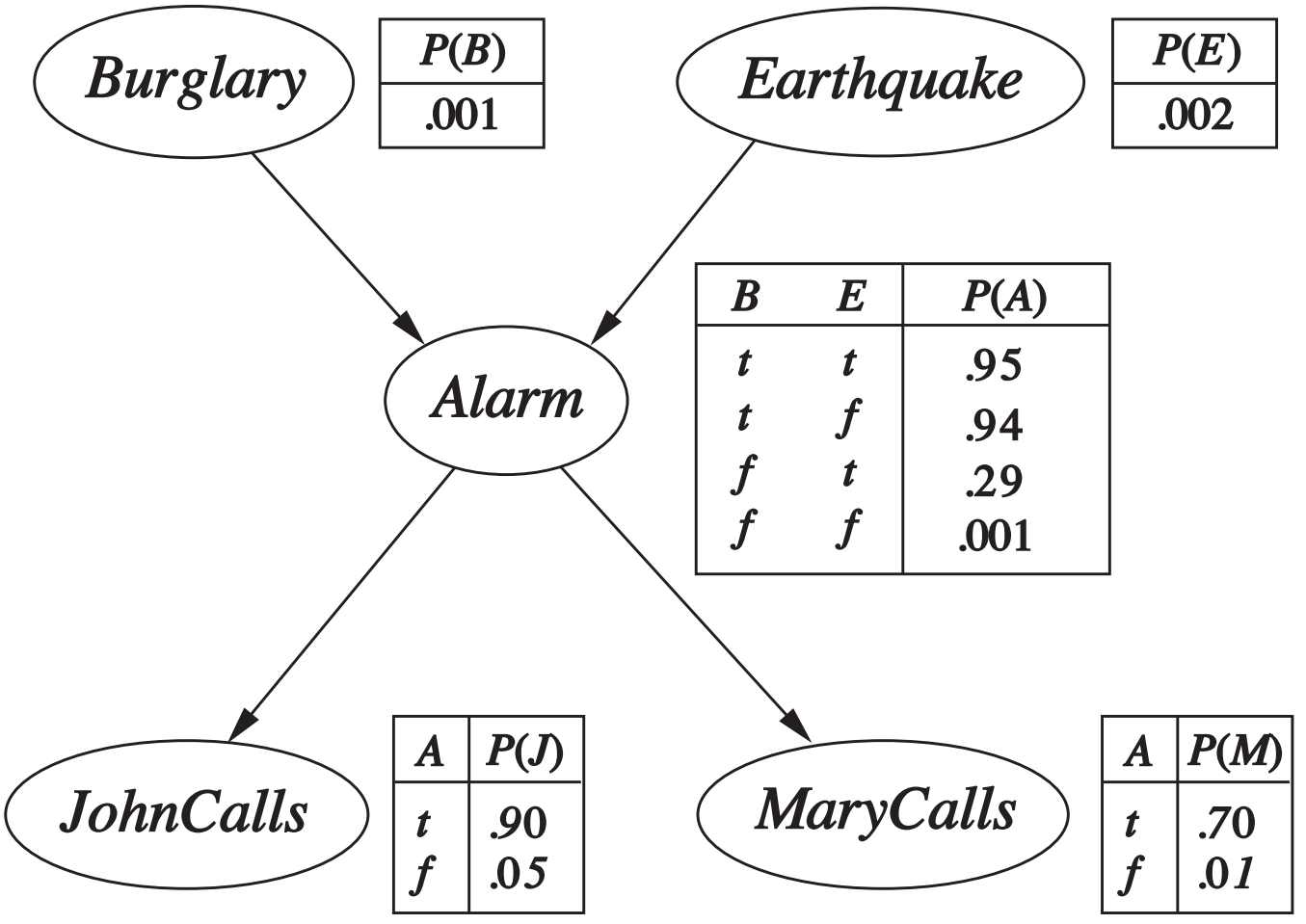


Example 1

- Variables: Burglar, Earthquake, Alarm, JohnCalls, MaryCalls.
- The network topology can be defined from domain knowledge:
  - A burglar can set the alarm off
  - An earthquake can set the alaram off
  - The alarm can cause Mary to call
  - The alarm can cause John to call

—

Image credits: CS188, UC Berkeley.


Burglary
P(B)
.001
Earthquake
P(E)
.002
Alarm
B E P(A)
t t .95
t f .94
f t .29
f f .001
JohnCalls
A P(J)
t .90
f .05
MaryCalls
A P(M)
t .70
f .01

Semantics

A Bayesian network implicitly encodes the full joint distribution as a product of local distributions, that is

$$P(x_1, ..., x_n) = \prod_{i=1}^{n} P(x_i | \text{parents}(X_i)).$$

Proof:

- By the chain rule, $P(x_1, ..., x_n) = \prod_{i=1}^{n} P(x_i | x_1, ..., x_{i-1})$.
- Provided that we assume conditional independence of X_i with its predecessors in the ordering given the parents, and provided $\text{parents}(X_i) \subseteq \{X_1, ..., X_{i-1}\}$, we have

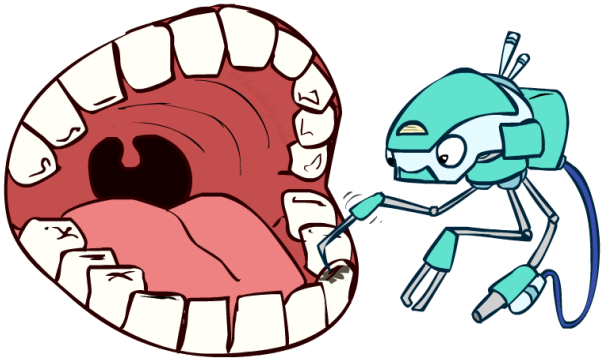
$$P(x_i | x_1, ..., x_{i-1}) = P(x_i | \text{parents}(X_i)).$$

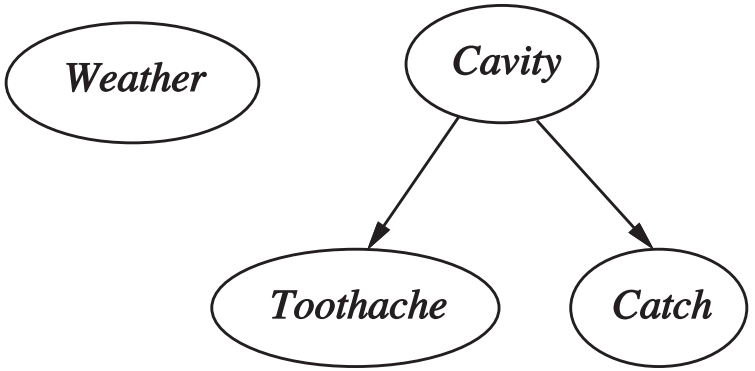
- Therefore, $P(x_1, ..., x_n) = \prod_{i=1}^{n} P(x_i | \text{parents}(X_i))$.

Example 1 (continued)

$$\begin{aligned} P(j, m, a, \neg b, \neg e) &= P(j|a)P(m|a)P(a|\neg b, \neg e)P(\neg b)P(\neg e) \\ &= 0.9 \times 0.7 \times 0.001 \times 0.999 \times 0.998 \\ &\approx 0.00063 \end{aligned}$$

Example 2



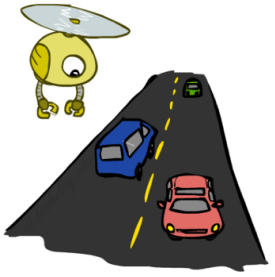


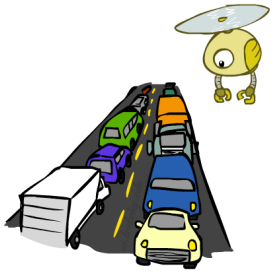
The dentist's scenario can be modeled as a Bayesian network with four variables, as shown on the right.

By construction, the topology of the network encodes conditional independence assertions. Each variable is independent of its non-descendants given its parents:

- Weather is independent of the other variables.
- Toothache and Catch are conditionally independent given Cavity.

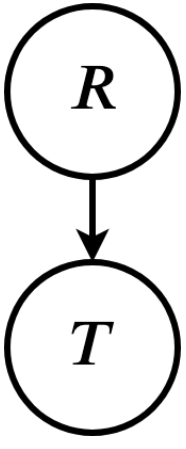
—
Image credits: CS188, UC Berkeley.





Example 3

Edges may correspond to causal relations.



$\mathbf{P}(R)$

R	P
r	0.25
¬r	0.75

$\mathbf{P}(T|R)$

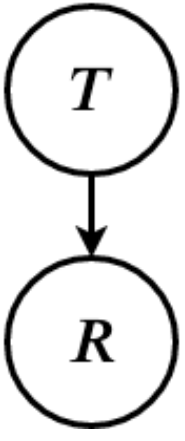
R	T	P
r	t	0.75
r	¬t	0.25
¬r	t	0.5
¬r	¬t	0.5

—

Image credits: CS188, UC Berkeley.

Example 3 (bis)

... but edges need not be causal!



$\mathbf{P}(T)$

T	P
t	$9/16$
$\neg$t	$7/16$

$\mathbf{P}(R|T)$

T	R	P
t	r	$1/3$
t	$\neg$r	$2/3$
$\neg$t	r	$1/7$
$\neg$t	$\neg$r	$6/7$

—

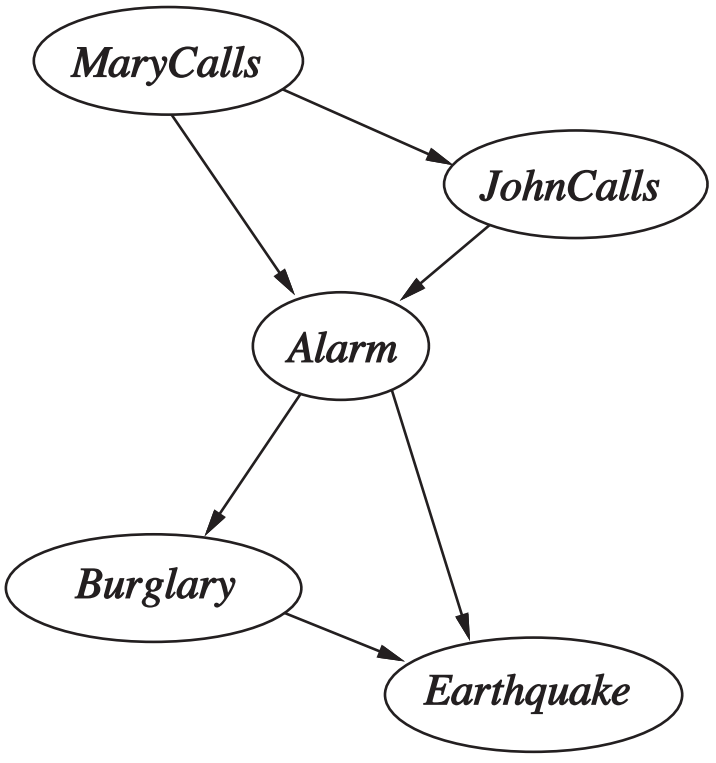
Image credits: CS188, UC Berkeley.

Construction

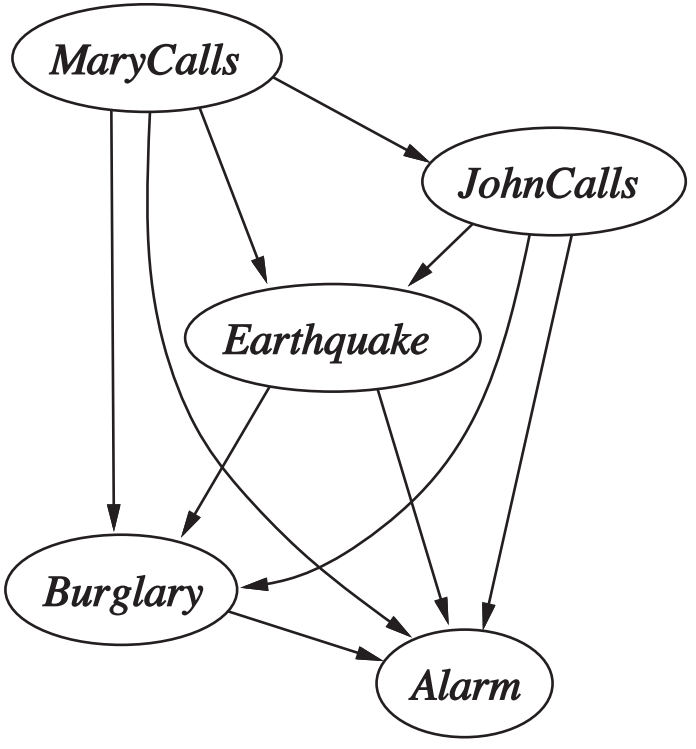
Bayesian networks can be constructed in any order, provided that the conditional independence assertions are respected.

Algorithm

1. Choose some ordering of the variables $X_1, ..., X_n$.
2. For $i = 1$ to n:
    1. Add X_i to the network.
    2. Select a minimal set of parents from $X_1, ..., X_{i-1}$ such that $P(x_i | x_1, ..., x_{i-1}) = P(x_i | \text{parents}(X_i))$.
    3. For each parent, insert a link from the parent to X_i.
    4. Write down the CPT.



(a)

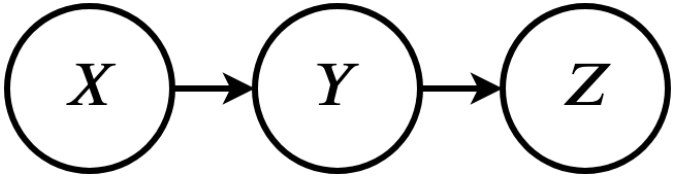


(b)

Do these networks represent the same distribution? Are they as compact?

Independence relations

Since the topology of a Bayesian network encodes conditional independence assertions, it can be used to answer questions about the independence of variables given some evidence.



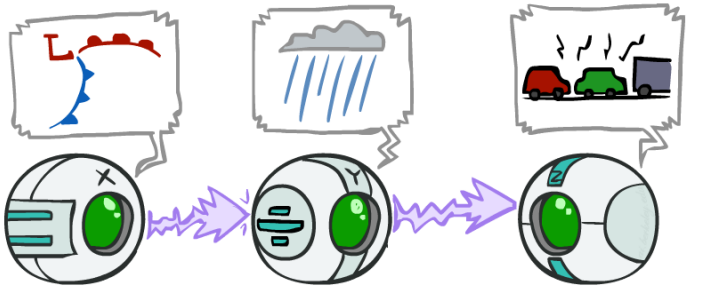
Example: Are X and Z necessarily independent?

Cascades

Is X independent of Z? No.

Counter-example:

- Low pressure causes rain causes traffic, high pressure causes no rain causes no traffic.
- In numbers:
  - $P(y|x) = 1$,
  - $P(z|y) = 1$,
  - $P(\neg y|\neg x) = 1$,
  - $P(\neg z|\neg y) = 1$



X: low pressure, Y: rain, Z: traffic.

$$P(x, y, z) = P(x)P(y|x)P(z|y)$$

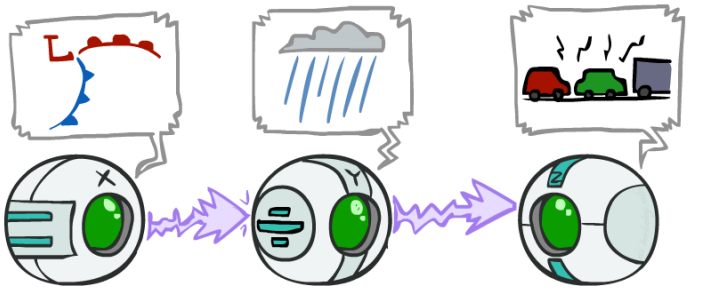
—

Image credits: CS188, UC Berkeley.

Is X independent of Z, given Y? Yes.

$$\begin{aligned} P(z|x,y) &= \frac{P(x,y,z)}{P(x,y)} \\ &= \frac{P(x)P(y|x)P(z|y)}{P(x)P(y|x)} \\ &= P(z|y) \end{aligned}$$

We say that the evidence along the cascade blocks the influence.



X: low pressure, Y: rain, Z: traffic.

$$P(x,y,z) = P(x)P(y|x)P(z|y)$$

—

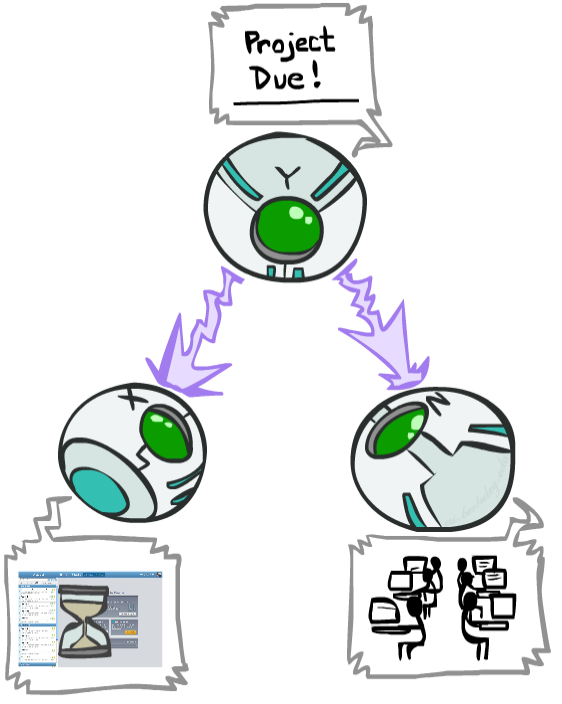
Image credits: CS188, UC Berkeley.

Common parent

Is X independent of Z? No.

Counter-example:

- Project due causes both forums busy and lab full.
- In numbers:
  - $P(x|y) = 1$,
  - $P(\neg x|\neg y) = 1$,
  - $P(z|y) = 1$,
  - $P(\neg z|\neg y) = 1$



X: forum busy, Y: project due, Z: lab full.

$$P(x, y, z) = P(y)P(x|y)P(z|y)$$

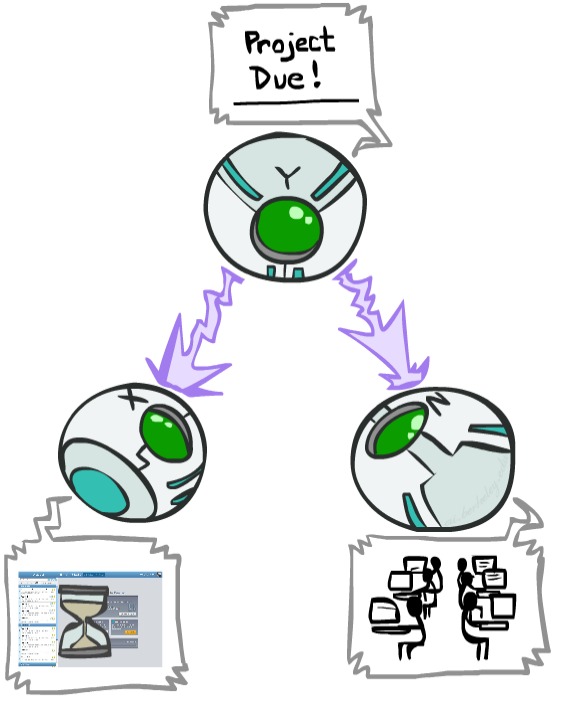
—

Image credits: CS188, UC Berkeley.

Is X independent of Z, given Y? Yes

$$\begin{aligned} P(z|x,y) &= \frac{P(x,y,z)}{P(x,y)} \\ &= \frac{P(y)P(x|y)P(z|y)}{P(y)P(x|y)} \\ &= P(z|y) \end{aligned}$$

Observing the parent blocks the influence between the children.



X: forum busy, Y: project due, Z: lab full.

$$P(x,y,z) = P(y)P(x|y)P(z|y)$$

—

Image credits: CS188, UC Berkeley.

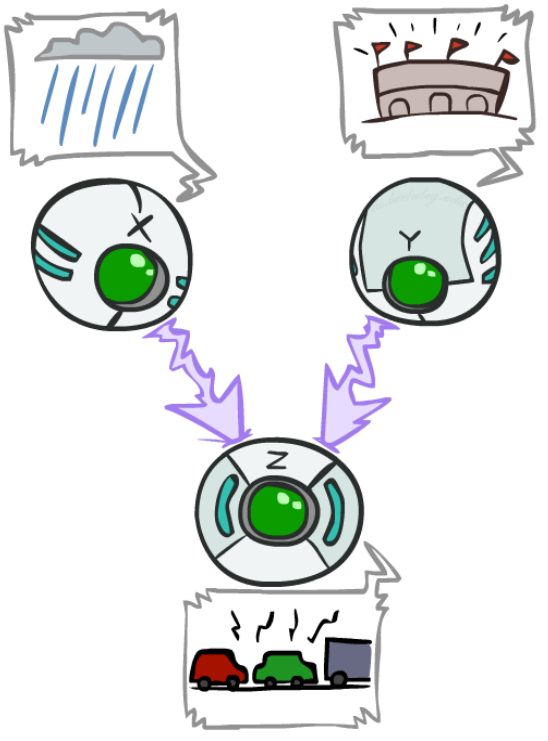
v-structures

Are X and Y independent? Yes.

- The ballgame and the rain cause traffic, but they are not correlated.
- (Prove it!)

Are X and Y independent given Z? No!

- Seeing traffic puts the rain and the ballgame in competition as explanation.
- This is backwards from the previous cases. Observing a child node activates influence between parents.



X: rain, Y: ballgame, Z: traffic.

$$P(x, y, z) = P(x)P(y)P(z|x, y)$$

—

Image credits: CS188, UC Berkeley.

d-separation

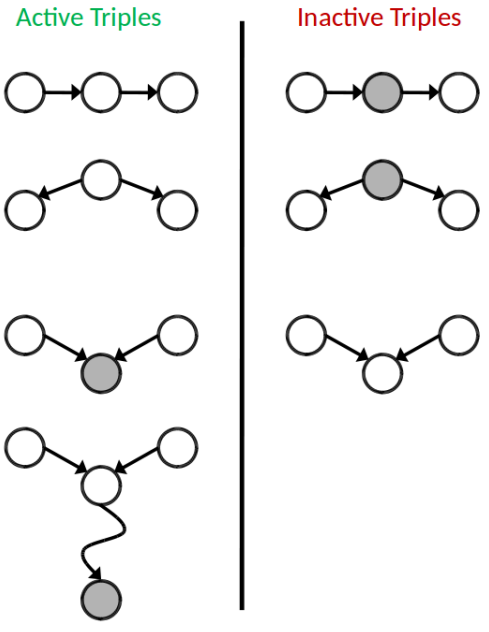
Let us assume a complete Bayesian network. Are X_i and X_j conditionally independent given evidence $Z_1 = z_1, ..., Z_m = z_m$?

Consider all (undirected) paths from X_i to X_j:

- If one or more active path, then independence is not guaranteed.
- Otherwise (i.e., all paths are inactive), then independence is guaranteed.

A path is active if each triple along the path is active:

- Cascade $A \rightarrow B \rightarrow C$ where B is unobserved (either direction).
- Common parent $A \leftarrow B \rightarrow C$ where B is unobserved.
- v-structure $A \rightarrow B \leftarrow C$ where B or one of its descendents is observed.

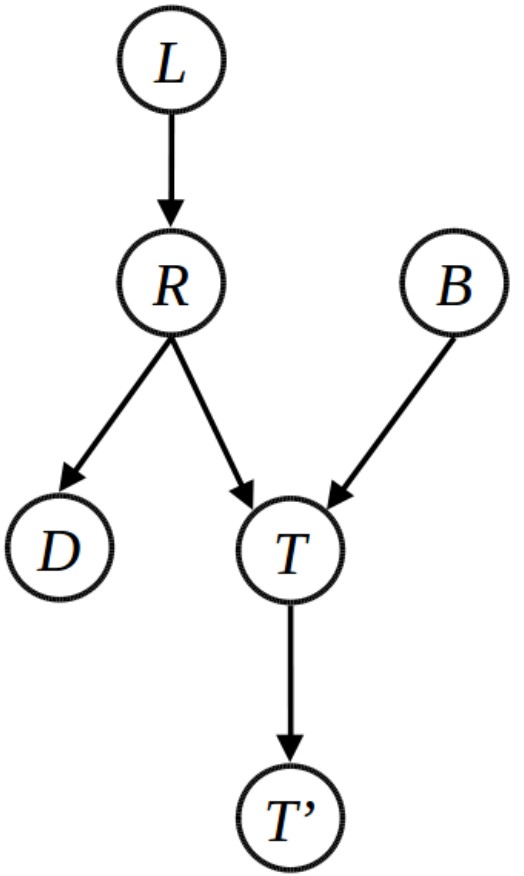


—

Image credits: CS188, UC Berkeley.

Example

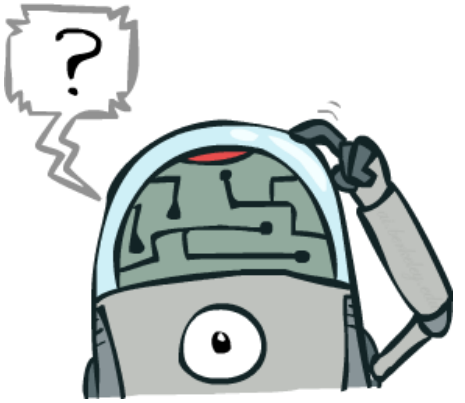
- $L \perp T' | T$?
- $L \perp B$?
- $L \perp B | T$?
- $L \perp B | T'$?
- $L \perp B | T, R$?



Inference

Inference is concerned with the problem **computing a marginal and/or a conditional probability distribution** from a joint probability distribution:

Simple queries:	$\mathbf{P}(X_i \mid e)$
Conjunctive queries:	$\mathbf{P}(X_i, X_j \mid e) = \mathbf{P}(X_i \mid e)\mathbf{P}(X_j \mid X_i, e)$
Most likely explanation:	$\arg\max_q P(q \mid e)$
Optimal decisions:	$\arg\max_a \mathbb{E}_{p(s' \mid s,a)}\left[V(s')\right]$



—

Image credits: CS188, UC Berkeley.

Inference by enumeration

Start from the joint distribution $\mathbf{P}(Q, E_1, ..., E_k, H_1, ..., H_r)$.

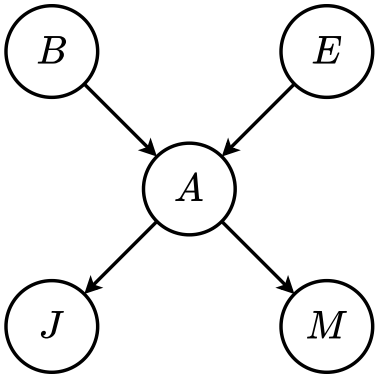
1. Select the entries consistent with the evidence $E_1, ..., E_k = e_1, ..., e_k$.
2. Marginalize out the hidden variables to obtain the joint of the query and the evidence variables:

$$\mathbf{P}(Q, e_1, ..., e_k) = \sum_{h_1, ..., h_r} \mathbf{P}(Q, h_1, ..., h_r, e_1, ..., e_k).$$

3. Normalize:

$$Z = \sum_q P(q, e_1, ..., e_k)$$

$$\mathbf{P}(Q|e_1, ..., e_k) = \frac{1}{Z}\mathbf{P}(Q, e_1, ..., e_k)$$

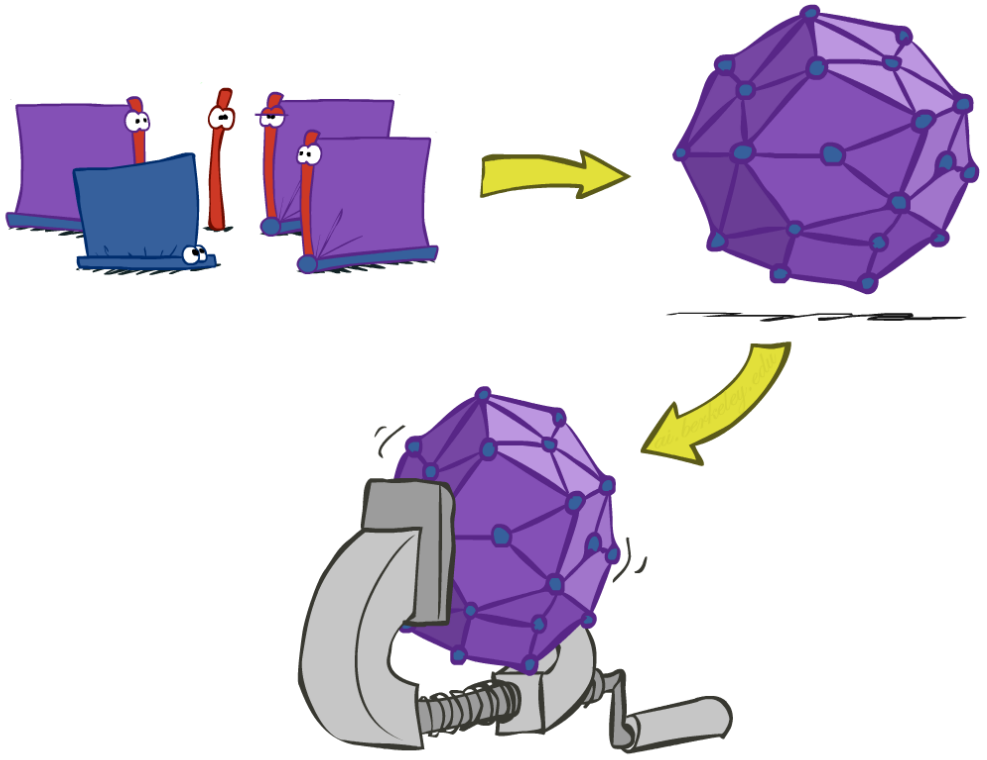


Consider the alarm network and the query $\mathbf{P}(B|j,m)$. We have

$$\begin{aligned}\mathbf{P}(B|j,m) &= \frac{1}{Z}\sum_e\sum_a \mathbf{P}(B,j,m,e,a) \\ &\propto \sum_e\sum_a \mathbf{P}(B,j,m,e,a).\end{aligned}$$

Using the Bayesian network, the full joint entries can be rewritten as the product of CPT entries

$$\mathbf{P}(B|j,m) \propto \sum_e\sum_a \mathbf{P}(B)P(e)\mathbf{P}(a|B,e)P(j|a)P(m|a).$$



Inference by enumeration is slow because the whole joint distribution is joined up before summing out the hidden variables.

—

Image credits: CS188, UC Berkeley.

Factors that do not depend on the variables in the summations can be factored out, which means that marginalization does not necessarily have to be done at the end, hence saving some computations.

For the alarm network, we have

$$\begin{aligned}\mathbf{P}(B|j,m) &\propto \sum_e \sum_a \mathbf{P}(B)P(e)\mathbf{P}(a|B,e)P(j|a)P(m|a) \\ &= \mathbf{P}(B)\sum_e P(e)\sum_a \mathbf{P}(a|B,e)P(j|a)P(m|a).\end{aligned}$$

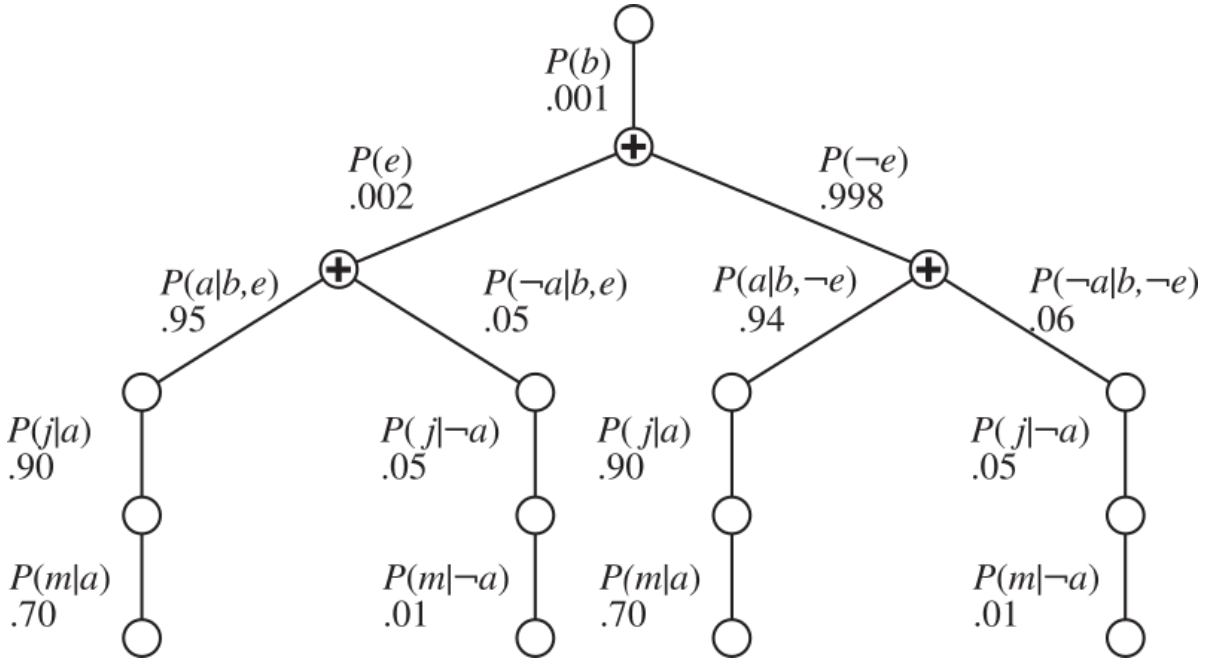
function ENUMERATION-ASK(X, **e**, bn) **returns** a distribution over X
  inputs: X, the query variable
    e, observed values for variables **E**
    bn, a Bayes net with variables $\{X\} \cup \mathbf{E} \cup \mathbf{Y}$ /* **Y** = *hidden variables* */

  Q(X) ← a distribution over X, initially empty
  for each value x_i of X **do**
    Q(x_i) ← ENUMERATE-ALL(bn.VARS, $\mathbf{e}_{x_i}$)
      where $\mathbf{e}_{x_i}$ is **e** extended with $X = x_i$
  return NORMALIZE(**Q**(X))

function ENUMERATE-ALL($vars$, **e**) **returns** a real number
  if EMPTY?($vars$) **then return** 1.0
  Y ← FIRST($vars$)
  if Y has value y in **e**
    then return $P(y \mid parents(Y)) \times$ ENUMERATE-ALL(REST($vars$), **e**)
    else return $\sum_y P(y \mid parents(Y)) \times$ ENUMERATE-ALL(REST($vars$), $\mathbf{e}_y$)
      where $\mathbf{e}_y$ is **e** extended with $Y = y$

Same complexity as DFS: $O(n)$ in space, $O(d^n)$ in time.

Evaluation tree for $P(b|j,m)$



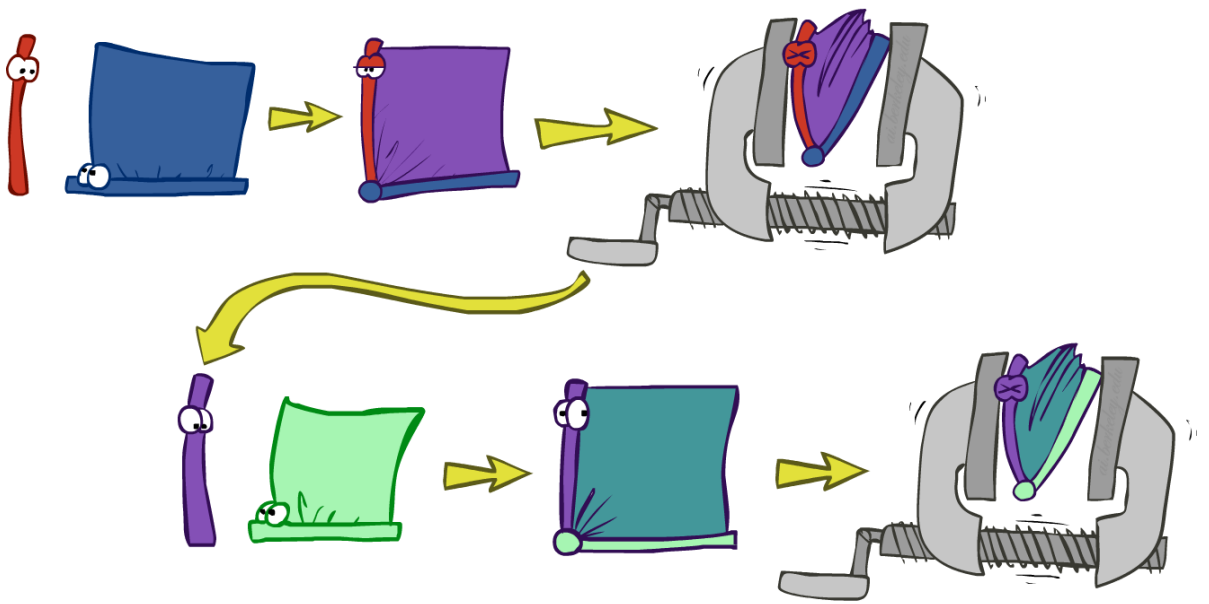
Despite the factoring, inference by enumeration is still inefficient. There are repeated computations!

- e.g., $P(j|a)P(m|a)$ is computed twice, once for e and once for $\neg e$.
- These can be avoided by storing intermediate results.

Inference by variable elimination

The **Variable Elimination** algorithm carries out summations right-to-left and stores intermediate factors to avoid recomputations. The algorithm interleaves:

- Joining sub-tables
- Eliminating hidden variables



—

Image credits: CS188, UC Berkeley.

Variable Elimination

Query: $\mathbf{P}(Q|e_1, ..., e_k)$.

1. Start with the initial factors (the local CPTs, instantiated by the evidence).
2. While there are still hidden variables:
    1. Pick a hidden variable H
    2. Join all factors mentioning H
    3. Eliminate H
3. Join all remaining factors
4. Normalize

Factors

- Each factor $\mathbf{f}_i$ is a multi-dimensional array indexed by the values of its argument variables. E.g.:

$$\mathbf{f}_4 = \mathbf{f}_4(A) = \begin{pmatrix} P(j|a) \\ P(j|\neg a) \end{pmatrix} = \begin{pmatrix} 0.90 \\ 0.05 \end{pmatrix}$$

$$\mathbf{f}_4(a) = 0.90$$

$$\mathbf{f}_4(\neg a) = 0.5$$

- Factors are initialized with the CPTs annotating the nodes of the Bayesian network, conditioned on the evidence.

Join

The pointwise product $\times$, or join, of two factors $\mathbf{f}_1$ and $\mathbf{f}_2$ yields a new factor $\mathbf{f}_3$.

- Exactly like a database join!
- The variables of $\mathbf{f}_3$ are the union of the variables in $\mathbf{f}_1$ and $\mathbf{f}_2$.
- The elements of $\mathbf{f}_3$ are given by the product of the corresponding elements in $\mathbf{f}_1$ and $\mathbf{f}_2$.

A	B	$\mathbf{f}_1(A,B)$
T	T	.3
T	F	.7
F	T	.9
F	F	.1

B	C	$\mathbf{f}_2(B,C)$
T	T	.2
T	F	.8
F	T	.6
F	F	.4

A	B	C	$\mathbf{f}_3(A,B,C)$
T	T	T	$.3 \times .2 = .06$
T	T	F	$.3 \times .8 = .24$
T	F	T	$.7 \times .6 = .42$
T	F	F	$.7 \times .4 = .28$
F	T	T	$.9 \times .2 = .18$
F	T	F	$.9 \times .8 = .72$
F	F	T	$.1 \times .6 = .06$
F	F	F	$.1 \times .4 = .04$

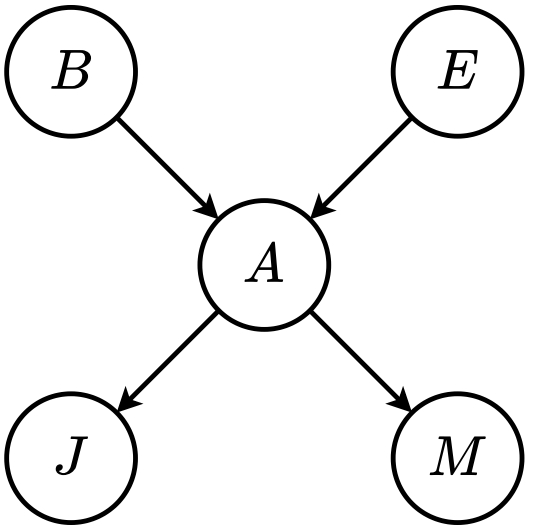
Figure 14.10 Illustrating pointwise multiplication: $\mathbf{f}_1(A,B) \times \mathbf{f}_2(B,C) = \mathbf{f}_3(A,B,C)$.

Elimination

Summing out, or eliminating, a variable from a factor is done by adding up the sub-arrays formed by fixing the variable to each of its values in turn.

For example, to sum out A from $\mathbf{f}_3(A, B, C)$, we write:

$$\begin{aligned}\mathbf{f}(B, C) &= \sum_a \mathbf{f}_3(a, B, C) = \mathbf{f}_3(a, B, C) + \mathbf{f}_3(\neg a, B, C) \\ &= \begin{pmatrix} 0.06 & 0.24 \\ 0.42 & 0.28 \end{pmatrix} + \begin{pmatrix} 0.18 & 0.72 \\ 0.06 & 0.04 \end{pmatrix} = \begin{pmatrix} 0.24 & 0.96 \\ 0.48 & 0.32 \end{pmatrix}\end{aligned}$$



Run the variable elimination algorithm for the query $\mathbf{P}(B|j,m)$.

Relevance

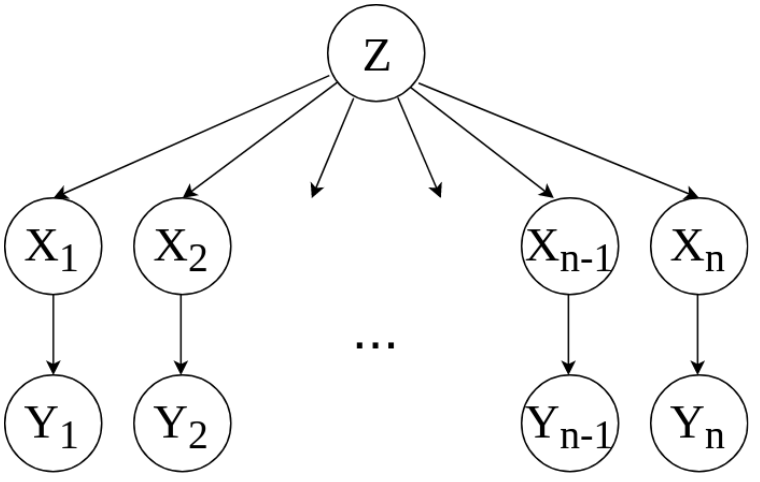
Consider the query $\mathbf{P}(J|b)$:

$$\mathbf{P}(J|b) \propto P(b) \sum_e P(e) \sum_a P(a|b,e) \mathbf{P}(J|a) \sum_m P(m|a)$$

- $\sum_m P(m|a) = 1$, therefore M is irrelevant for the query.
- In other words, $\mathbf{P}(J|b)$ remains unchanged if we remove M from the network.

Theorem. H is irrelevant for $\mathbf{P}(Q|e)$ unless $H \in \mathbf{ancestors}(\{Q\} \cup E)$.

Complexity



Consider the query $\mathbf{P}(X_n|y_1, ..., y_n)$.

Work through the two elimination orderings:

- $Z, X_1, ..., X_{n-1}$
- $X_1, ..., X_{n-1}, Z$

What is the size of the maximum factor generated for each of the orderings?

- Answer: 2^{n+1} vs. 2^2 (assuming boolean values)

The computational and space complexity of variable elimination is determined by the largest factor.

- The elimination ordering can greatly affect the size of the largest factor.
- The optimal ordering is NP-hard to find. There is no known polynomial-time algorithm to find it.

Approximate inference

Exact inference is intractable for most probabilistic models of practical interest. (e.g., involving many variables, continuous and discrete, undirected cycles, etc).

We must resort to approximate inference algorithms:

- Sampling methods: produce answers by repeatedly generating random numbers from a distribution of interest.
- Variational methods: formulate inference as an optimization problem.
- Belief propagation methods: formulate inference as a message-passing algorithm.
- Machine learning methods: learn an approximation of the target distribution from training examples.

Parameter learning

When modeling a domain, we can choose a probabilistic model specified as a Bayesian network. However, specifying the individual probability values is often difficult.

A workaround is to use a parameterized family $\mathbf{P}(X|\theta)$ (sometimes also noted $\mathbf{P}_\theta(X)$) of models, and estimate the parameters θ from data.

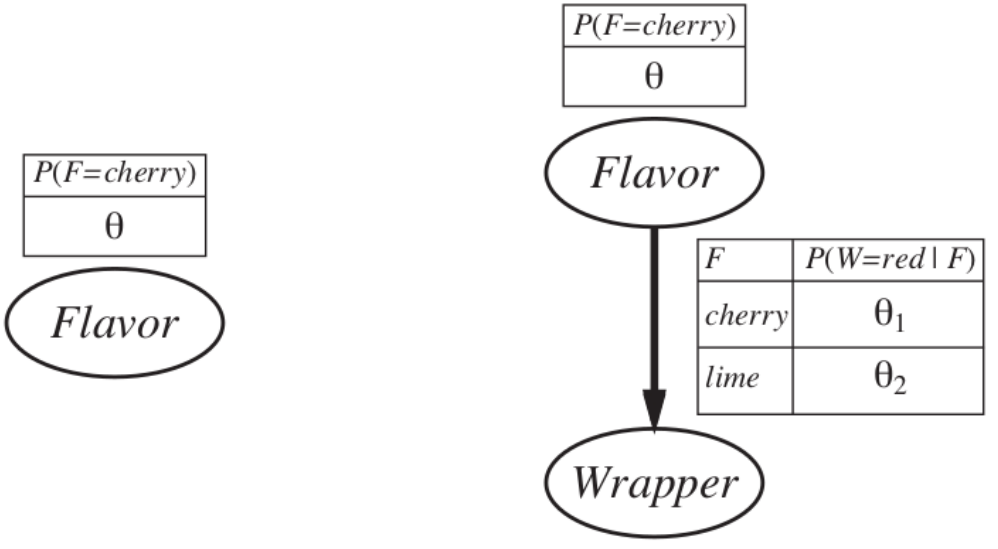


Figure 20.2 (a) Bayesian network model for the case of candies with an unknown proportion of cherries and limes. (b) Model for the case where the wrapper color depends (probabilistically) on the candy flavor.

Maximum likelihood estimation

Suppose we have a set of N i.i.d. observations $\mathbf{d} = \{x_1, ..., x_N\}$.

The likelihood of the parameters θ is the probability of the data given the parameters

$$P(\mathbf{d}|\theta) = \prod_{j=1}^{N} P(x_j|\theta).$$

The maximum likelihood estimate (MLE) θ^* of the parameters is the value of θ that maximizes the likelihood

$$\theta^* = \arg\max_{\theta} P(\mathbf{d}|\theta).$$

In practice,

1. Write down the log-likelihood $L(\theta) = \log P(\mathbf{d}|\theta)$ of the parameters θ.
2. Write down the derivative $\frac{\partial L}{\partial \theta}$ of the log-likelihood of the parameters θ.
3. Find the parameter values θ^* such that the derivatives are zero (and check whether the Hessian is negative definite).

Case (a)

What is the fraction θ of cherry candies?

Suppose we unwrap N candies, and get c cherries and $l = N - c$ limes. These are i.i.d. observations, therefore

$$P(\mathbf{d}|\theta) = \prod_{j=1}^{N} P(x_j|\theta) = \theta^c (1-\theta)^l.$$

Maximize this w.r.t. θ, which is easier for the log-likelihood and leads to

$$\begin{aligned} L(\mathbf{d}|\theta) &= \log P(\mathbf{d}|\theta) = c \log \theta + l \log(1-\theta) \\ \frac{\partial L(\mathbf{d}|\theta)}{\partial \theta} &= \frac{c}{\theta} - \frac{l}{1-\theta} = 0. \end{aligned}$$

Hence $\theta = \frac{c}{N}$.

Case (b)

Red and green wrappers depend probabilistically on flavor. E.g., the likelihood for a cherry candy in green wrapper is

$$\begin{aligned} &P(\text{cherry}, \text{green}|\theta, \theta_1, \theta_2) \\ &= P(\text{cherry}|\theta, \theta_1, \theta_2)P(\text{green}|\text{cherry}, \theta, \theta_1, \theta_2) \\ &= \theta(1-\theta_1). \end{aligned}$$

The likelihood for the parameters, given N candies, r_c red-wrapped cherries, g_c green-wrapped cherries, etc., is

$$\begin{aligned} P(\mathbf{d}|\theta, \theta_1, \theta_2) &= \theta^c(1-\theta)^l\theta_1^{r_c}(1-\theta_1)^{g_c}\theta_2^{r_l}(1-\theta_2)^{g_l} \\ L &= c\log\theta + l\log(1-\theta) + \\ &\quad r_c\log\theta_1 + g_c\log(1-\theta_1) + \\ &\quad r_l\log\theta_2 + g_l\log(1-\theta_2). \end{aligned}$$

The derivatives of L yield

$$\begin{aligned}
\frac{\partial L}{\partial \theta} &= \frac{c}{\theta} - \frac{l}{1-\theta} = 0 \Rightarrow \theta = \frac{c}{c+l} \\
\frac{\partial L}{\partial \theta_1} &= \frac{r_c}{\theta_1} - \frac{g_c}{1-\theta_1} = 0 \Rightarrow \theta_1 = \frac{r_c}{r_c+g_c} \\
\frac{\partial L}{\partial \theta_2} &= \frac{r_l}{\theta_2} - \frac{g_l}{1-\theta_2} = 0 \Rightarrow \theta_2 = \frac{r_l}{r_l+g_l}.
\end{aligned}$$

In case (a), if we unwrap 1 candy and get 1 cherry, what is the MLE? How confident are we in this estimate?

- With small datasets, maximum likelihood estimation can lead to overfitting.
- The MLE does not provide a measure of uncertainty about the parameters.

Bayesian parameter learning

We can treat parameter learning as a **Bayesian inference** problem:

- Make the parameters θ random variables and treat them as hidden variables.
- Specify a prior distribution $\mathbf{P}(\theta)$ over the parameters.
- Then, as data arrives, update our beliefs about the parameters to obtain the posterior distribution $\mathbf{P}(\theta|\mathbf{d})$.

How does Figure 20.2 (a) should be updated?

Case (a)

What is the fraction θ of cherry candies?

We assume a Beta prior

$$P(\theta) = \text{Beta}(\theta|a, b) = \frac{1}{Z}\theta^{a-1}(1-\theta)^{b-1}$$

where Z is a normalization constant.

Then, observing a cherry candy yields the posterior

$$\begin{aligned} P(\theta|\text{cherry}) &\propto P(\text{cherry}|\theta)P(\theta) \\ &= \theta\text{Beta}(\theta|a, b) \\ &= \theta(1-\theta)^{b-1}\theta^{a-1}(1-\theta)^{b-1} \\ &= \theta^{a}(1-\theta)^{b-1} \\ &= \text{Beta}(\theta|a+1, b). \end{aligned}$$

(demo)

Case (b)

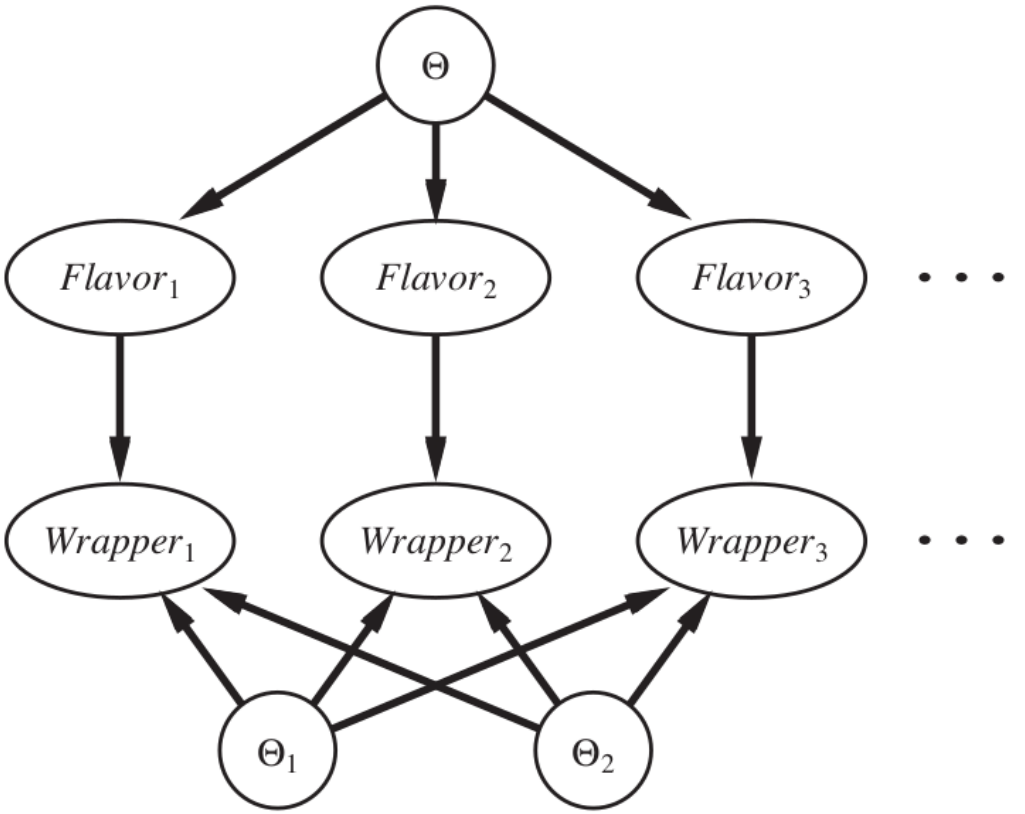


Figure 20.6 A Bayesian network that corresponds to a Bayesian learning process. Posterior distributions for the parameter variables Θ, Θ_1, and Θ_2 can be inferred from their prior distributions and the evidence in the $Flavor_i$ and $Wrapper_i$ variables.

Maximum a posteriori estimation

When the posterior cannot be computed analytically, we can use maximum a posteriori (MAP) estimation, which consists in approximating the posterior with the point estimate θ^* that maximizes the posterior distribution, i.e.,

$$\theta^* = \arg\max_{\theta} P(\theta|\mathbf{d}) = \arg\max_{\theta} P(\mathbf{d}|\theta)P(\theta).$$

Summary

- A Bayesian Network specifies a full joint distribution. BNs are often exponentially smaller than an explicitly enumerated joint distribution.
- The topology of a Bayesian network encodes conditional independence assumptions between random variables.
- Inference is the problem of computing a marginal and/or a conditional probability distribution from a joint probability distribution.
    - Exact inference is possible for simple Bayesian networks, but is intractable for most probabilistic models of practical interest.
    - Approximate inference algorithms are used in practice.
- Parameters of a Bayesian network can be learned from data using maximum likelihood estimation or Bayesian inference.

The end.