Machine Intelligence

6. Reasoning Under Uncertainty, Part III: Inference in Bayesian networks

Putting the Machinery to Practical Use

Álvaro Torralba

AALBORG UNIVERSITET

Fall 2022

Thanks to Thomas D. Nielsen and Jörg Hoffmann for slide sources

Our Agenda for This Topic

→ Our treatment of the topic "Probabilistic Reasoning" consists of Chapters 4-6.

- **Chapter 4**: All the basic machinery at use in Bayesian networks.

  → Sets up the framework and basic operations.

- **Chapter 5**: Bayesian networks: What they are and how to build them.

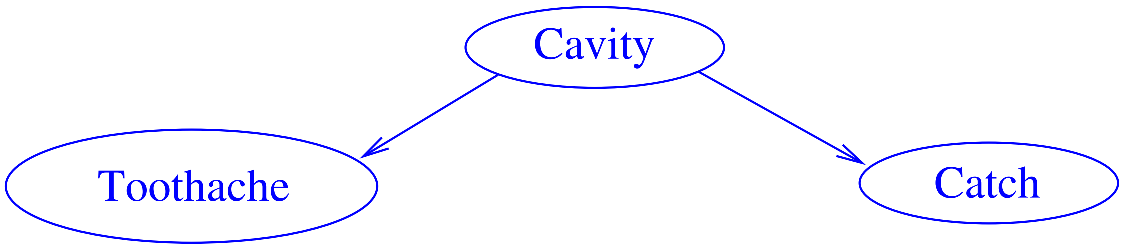
  → The most wide-spread and successful practical framework for probabilistic reasoning.

- **This Chapter**: Bayesian networks: how to use them.

  → How to use Bayesian Networks to answer our questions.

Reminder: Our Machinery

1. **Graph captures variable dependencies:** (Variables $X_1, \dots, X_n$)



$\rightarrow$ Given evidence $\mathbf{e}$, want to know $\mathbf{P}(X \mid e)$. Remaining vars: $\mathbf{Y}$.

2. **Normalization+Marginalization:**

$$\mathbf{P}(X \mid \mathbf{e}) = \alpha \mathbf{P}(X, \mathbf{e}) = \alpha \sum_{\mathbf{y} \in \mathbf{Y}} \mathbf{P}(X, \mathbf{e}, \mathbf{y})$$

$\rightarrow$ A sum over atomic events!

3. **Chain rule:** $X_1, \dots, X_n$ consistently with dependency graph.

$$\mathbf{P}(X_1, \dots, X_n) = \mathbf{P}(X_n \mid X_{n-1}, \dots, X_1) * \mathbf{P}(X_{n-1} \mid X_{n-2}, \dots, X_1) * \dots * \mathbf{P}(X_1)$$

4. **Exploit conditional independence:** Instead of $\mathbf{P}(X_i \mid X_{i-1}, \dots, X_1)$, we can use $\mathbf{P}(X_i \mid Parents(X_i))$.

$\rightarrow$ Bayesian networks!

Reminder: Recovering the Full Joint Probability Distribution

"A Bayesian network is a methodology for representing the full joint probability distribution."

$\rightarrow$ How to recover the full joint probability distribution $\mathbf{P}(X_1, \dots, X_n)$ from $BN = (\{X_1, \dots, X_n\}, E)$?

Chain rule: For any ordering $X_1, \dots, X_n$, we have:

$$\mathbf{P}(X_1, \dots, X_n) = \mathbf{P}(X_n \mid X_{n-1}, \dots, X_1)\mathbf{P}(X_{n-1} \mid X_{n-2}, \dots, X_1) \dots \mathbf{P}(X_1)$$

Choose $X_1, \dots, X_n$ consistent with BN: $X_j \in Parents(X_i) \implies j < i$.

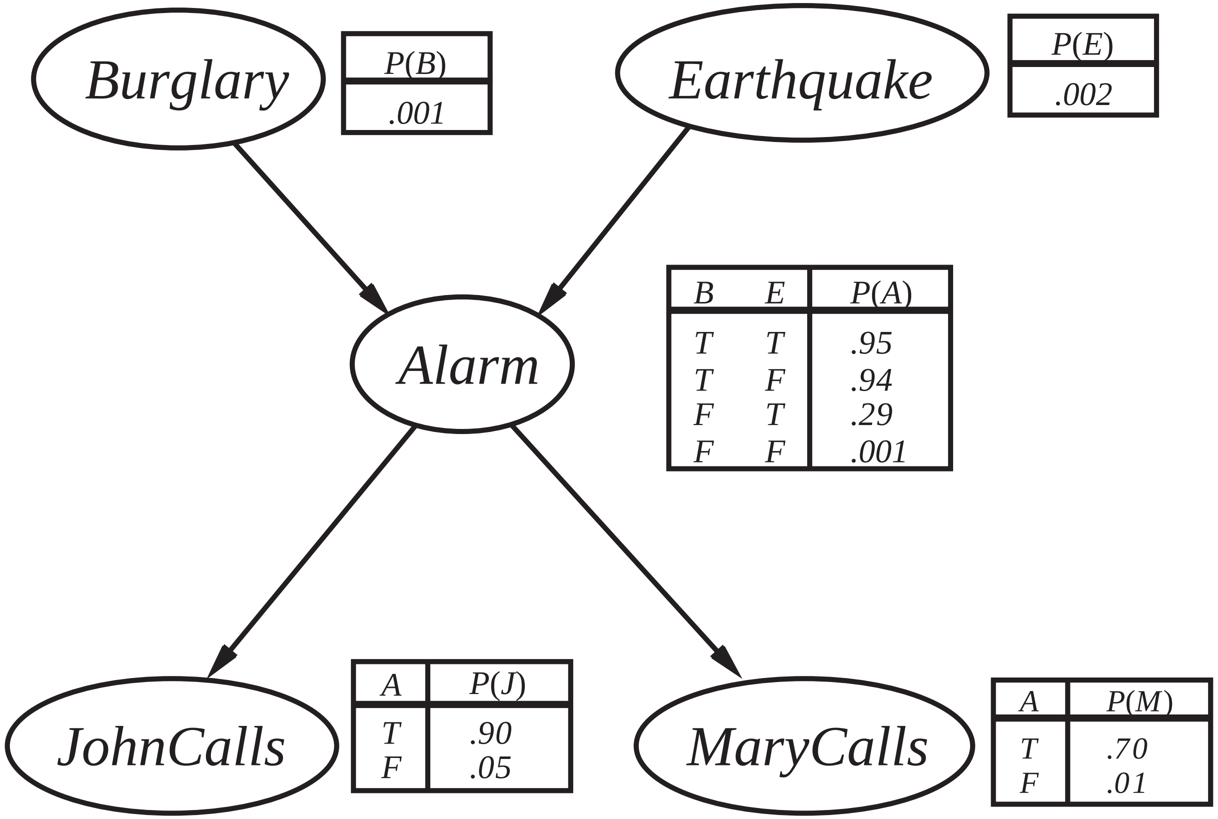
Exploit conditional independence: With BN assumption (A), instead of $\mathbf{P}(X_i \mid X_{i-1} \dots, X_1)$ we can use $\mathbf{P}(X_i \mid Parents(X_i))$:

$$\mathbf{P}(X_1, \dots, X_n) = \Pi_{i=1}^{n}\mathbf{P}(X_i \mid Parents(X_i))$$

The distributions $\mathbf{P}(X_i \mid Parents(X_i))$ are given by BN assumption (B).

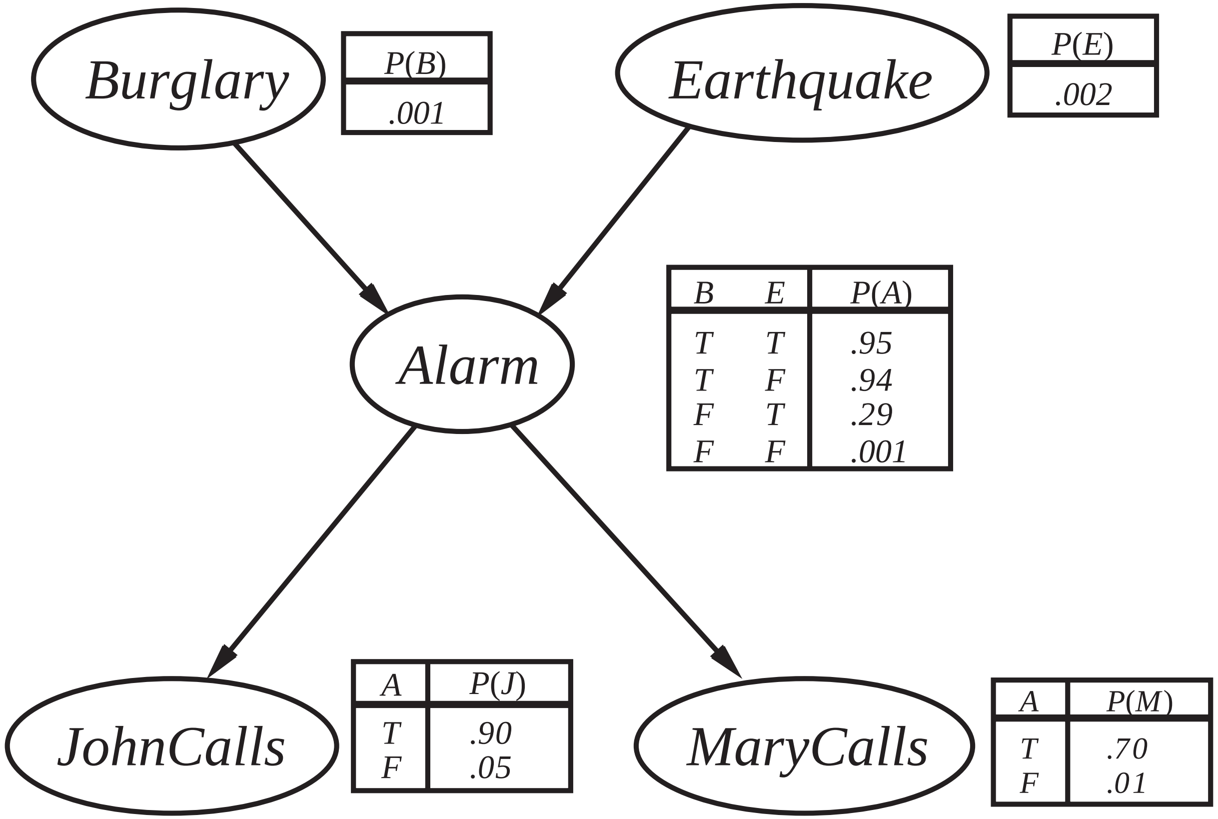
$\rightarrow$ Same for atomic events $P(x_1, \dots, x_n)$.

Reminder: Recovering a Probability for John, Mary, and the Alarm



$$P(j, m, a, \neg b, \neg e) =$$
$$=$$
$$=$$

Recovering a Probability for John, Mary, and the Alarm



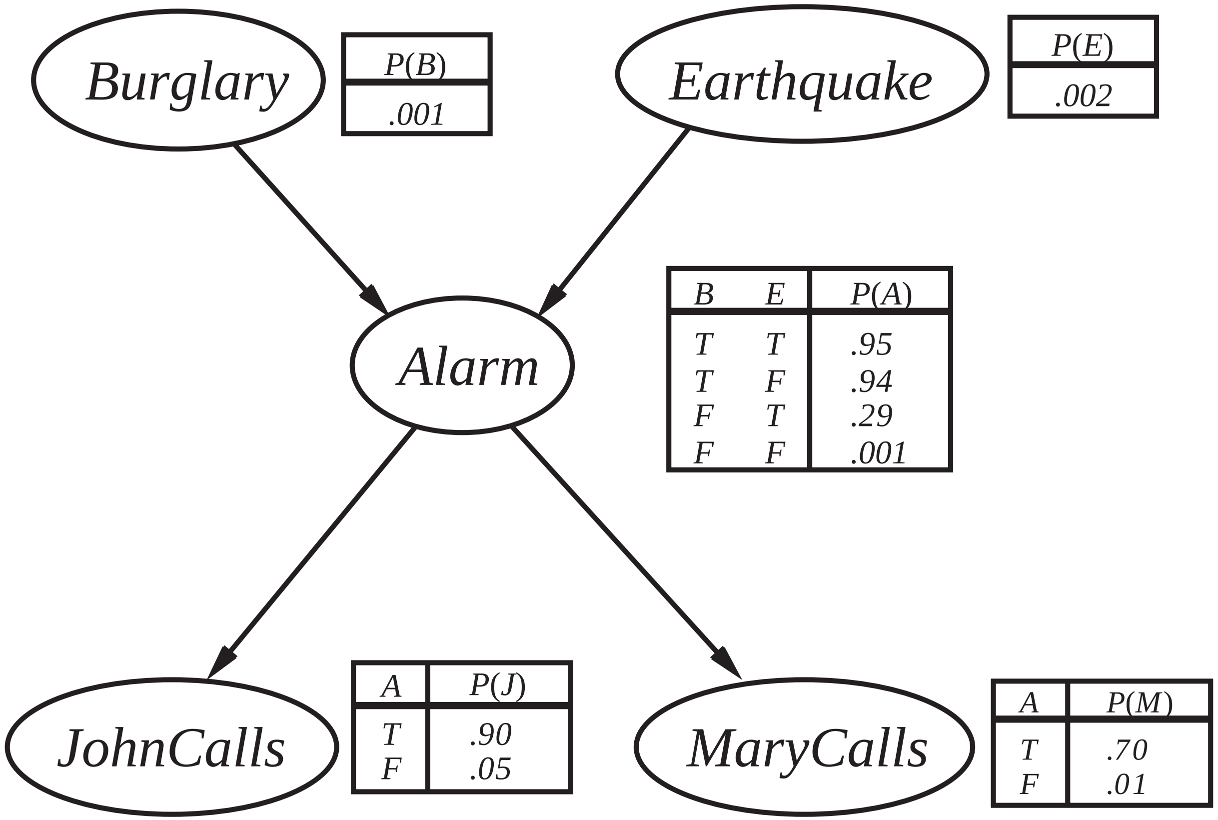
$$
\begin{aligned}
P(j, m, a, \neg b, \neg e) &= \\
&= \\
&=
\end{aligned}
$$

Our Agenda for This Chapter

- **Probabilistic Inference Tasks**: What problems do we want to solve?
  →Before doing algorithms that answer, we need to understand what the questions are about!
- **Exact Inference: Naive Enumeration**: A basic method to solve the probabilistic inference tasks
  →A first basic method we can use (but does not scale very well)
- **Exact Inference: Variable Elimination**: A more clever way of using the structure of the Bayesian Network to our advantage!
  →How to obtain the exact answer to our questions !
- **Naive Bayes Models**: A very simple Bayesian model that can always be computed quickly
  →A very simple case !
- **Approximate Inference via Sampling:** What to do when computing the exact answer is too expensive?
  →We can approximate the solution via sampling methods!

Inference for Mary and John

→ Observe evidence variables and draw conclusions on query variables.



What is $\mathbf{P}(Burglary \mid johncalls)$?

What is $\mathbf{P}(Burglary \mid johncalls, marycalls)$?

Probabilistic Inference Tasks in Bayesian Networks

Definition (Probabilistic Inference Task). *Given random variables $X_1, \dots, X_n$, a probabilistic inference task consists of a set $\mathbf{X} \subseteq \{X_1, \dots, X_n\}$ of query variables, a set $\mathbf{E} \subseteq \{X_1, \dots, X_n\}$ of evidence variables, and an event $\mathbf{e}$ that assigns values to $\mathbf{E}$. We wish to compute the posterior probability distribution $\mathbf{P}(\mathbf{X} \mid \mathbf{e})$.*

Notes:

- $\mathbf{Y} := \{X_1, \dots, X_n\} \setminus (\mathbf{X} \cup \mathbf{E})$ are the hidden variables.
- We assume that a BN for $X_1, \dots, X_n$ is given.
- In the remainder, for simplicity, $\mathbf{X} = \{X\}$ is a singleton.

Example: In $\mathbf{P}(Burglary \mid johncalls, marycalls)$, $X = Burglary$, $\mathbf{e} = johncalls, marycalls$, and $\mathbf{Y} = \{Alarm, EarthQuake\}$.

Simplifying the Problem by Using Normalization

According to the definition of conditional probability:

$$P(X \mid \mathbf{E} = \mathbf{e}) = \frac{P(X, \mathbf{E} = \mathbf{e})}{P(\mathbf{E} = \mathbf{e})}$$

It is sufficient to compute for each $x \in D_X$ the value

$$P(X = x, \mathbf{E} = \mathbf{e}).$$

Together with

$$P(\mathbf{E} = \mathbf{e}) = \sum_{x \in D_X} P(X = x, \mathbf{E} = \mathbf{e})$$

this gives the desired posterior distribution.

$\rightarrow$ To predict the desired conditional probability $P(X \mid \mathbf{E} = \mathbf{e})$, it suffices for us to compute the probability of (non-atomic) events: $P(X \mid \mathbf{E} = \mathbf{e})$!

Notation: As $\alpha = \frac{1}{\sum_{x \in D_X} P(X=x,\mathbf{E}=\mathbf{e})}$ is easily derived from $P(X \mid \mathbf{E} = \mathbf{e})$, we simply write $P(X \mid \mathbf{E} = \mathbf{e}) = \alpha P(X \mid \mathbf{E} = \mathbf{e})$ instead of $\frac{P(X,\mathbf{E}=\mathbf{e})}{P(\mathbf{E}=\mathbf{e})}$.

Marginalization: Inference as Summation

Want to compute: $P(X = x, \mathbf{E} = \mathbf{e})$

Problem: We do not know the value of hidden variables Y

We sum over all possible values of the hidden variables!

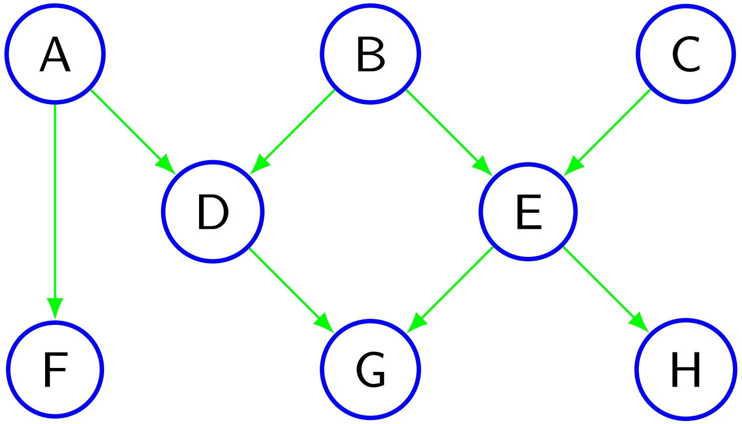
Let X be the variable of interest, $\mathbf{E}$ the evidence variables, and $\mathbf{Y} = Y_1, \dots, Y_l$ the remaining variables in the network not belonging to $X \cup \mathbf{E}$. Then

$$P(X = x, \mathbf{E} = \mathbf{e}) =$$

Simplified notation:

$$P(x, \mathbf{e}) =$$

Naive Enumeration Example



Find $P(B|a,f,g,h) = \frac{P(B,a,f,g,h)}{P(a,f,g,h)}$

We can if we have access to $P(A,B,C,D,E,F,G,H)$

$\rightarrow$ Reminder: We can recover the full joint probability distribution from our BN!

Inserting evidence we get:

$$P(B \mid a,f,g,h) = \alpha \cdot \sum_{C,D,E} P(a,B,C,D,E,f,g,h)$$

and

$$\frac{1}{\alpha} = P(a,f,g,h) = \sum_{\mathrm{B}} P(B,a,f,g,h)$$

Inference by Enumeration: The Principle (A Reminder!)

Given evidence $\mathbf{e}$, want to know $\mathbf{P}(X \mid \mathbf{e})$. Hidden variables: $\mathbf{Y}$.

1. **Bayesian network** BN **captures variable dependencies.**

2. **Normalization+Marginalization.**

$$\mathbf{P}(X \mid \mathbf{e}) = \alpha \mathbf{P}(X, \mathbf{e}); \text{ if } \mathbf{Y} \neq \emptyset \text{ then } \mathbf{P}(X \mid \mathbf{e}) = \alpha \textstyle\sum_{\mathbf{y} \in \mathbf{Y}} \mathbf{P}(X, \mathbf{e}, \mathbf{y})$$

$\rightarrow$ Recover the summed-up probabilities $\mathbf{P}(X, \mathbf{e}, \mathbf{y})$ from BN!

3. **Chain rule.** Order $X_1, \dots, X_n$ consistent with BN.

$$\mathbf{P}(X_1, \dots, X_n) = \mathbf{P}(X_n \mid X_{n-1}, \dots, X_1) \mathbf{P}(X_{n-1} \mid X_{n-2}, \dots, X_1) \dots \mathbf{P}(X_1)$$

4. **Exploit conditional independence.** Instead of $\mathbf{P}(X_i \mid X_{i-1}, \dots, X_1)$, use $\mathbf{P}(X_i \mid Parents(X_i))$.

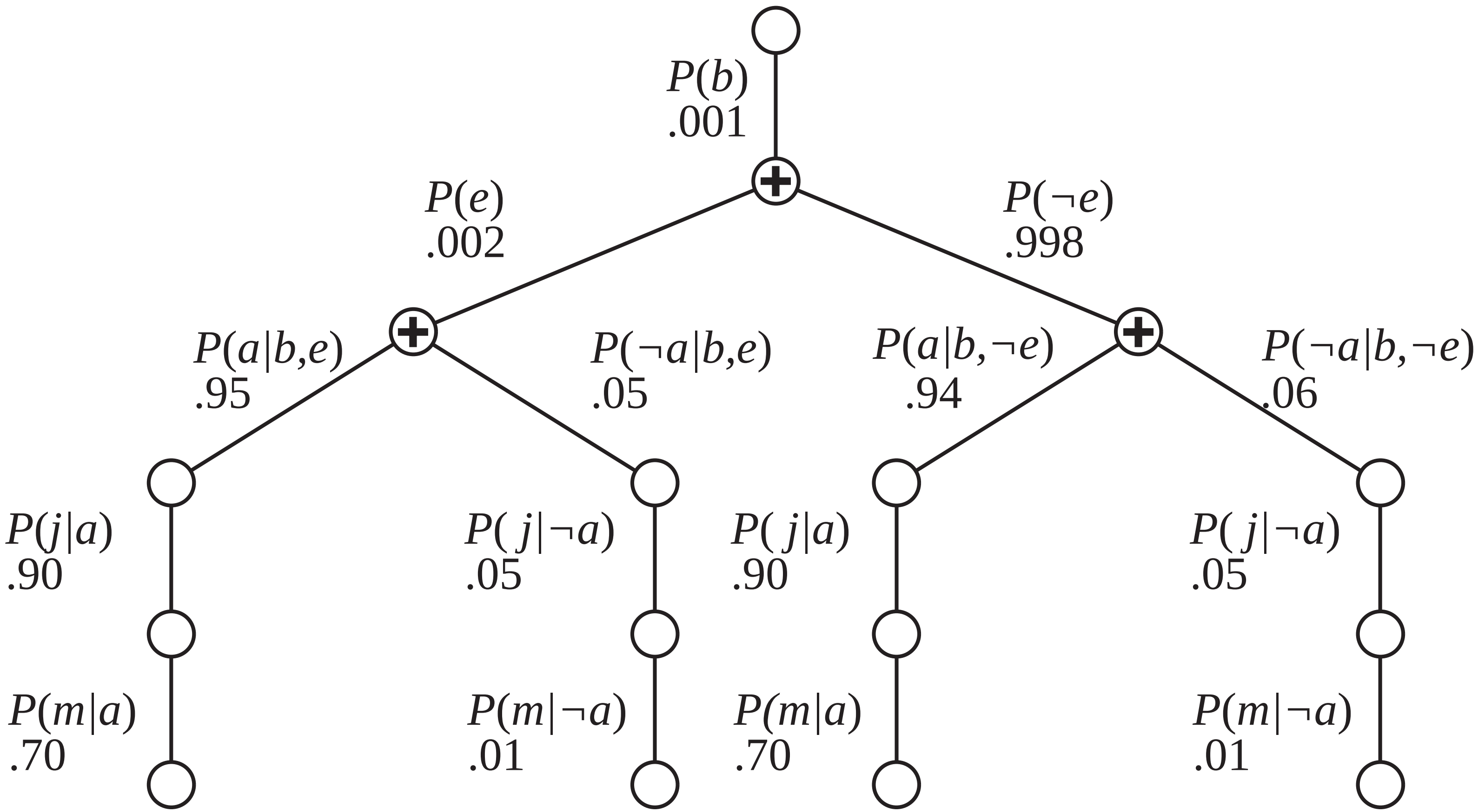
$\rightarrow$ Given a Bayesian network BN, probabilistic inference tasks can be solved as sums of products of conditional probabilities from BN.

$\rightarrow$ Sum over all value combinations of hidden variables.

Inference by Enumeration: John and Mary

Inference by Enumeration: John and Mary, ctd.

The Evaluation of $P(b \mid j, m)$, as a "Search Tree"



$\rightarrow$ Inference by enumeration = a tree with "sum nodes" branching over values of hidden variables, and with non-branching "multiplication nodes".

Inference by Enumeration: Pseudo-Code

→ With $bn.\textsc{Vars}$ being a variable ordering consistent with bn:

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* */

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

---

function ENUMERATE-ALL($vars$, **e**) **returns** a real number
  if EMPTY?($vars$) **then return** 1.0
  $Y \leftarrow$ 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 $Y = y$

Inference by Enumeration: Properties

Inference by Enumeration:

- Evaluates the tree in a depth-first manner.
- Space Complexity: Linear in the number of variables.
- Time Complexity:

Bad News: Not in general.

- Probabilistic inference is **#P**-hard.
- **#P** is harder than **NP** (i.e., **NP** $\subseteq$ **#P**).

But: Variable Elimination.

- Improves on inference by enumeration through (A) avoiding repeated computation, and (B) avoiding irrelevant computation.
- In some special cases, variable elimination runs in polynomial time.

Variable Elimination: Sketch of Ideas

(A) Avoiding repeated computation: Evaluate expressions from right to left, storing all intermediate results. For query $P(B \mid j, m)$:

1. CPTs of BN yield factors (probability tables): $\mathbf{P}(B \mid j, m) =$
$$\alpha \underbrace{\mathbf{P}(B)}_{\mathbf{f}_1(B)} \sum_{v_E} \underbrace{P(v_E)}_{\mathbf{f}_2(E)} \sum_{v_A} \underbrace{\mathbf{P}(v_A \mid B, v_E)}_{\mathbf{f}_3(A,B,E)} \underbrace{P(j \mid v_A)}_{\mathbf{f}_4(A)} \underbrace{P(m \mid v_A)}_{\mathbf{f}_5(A)}$$

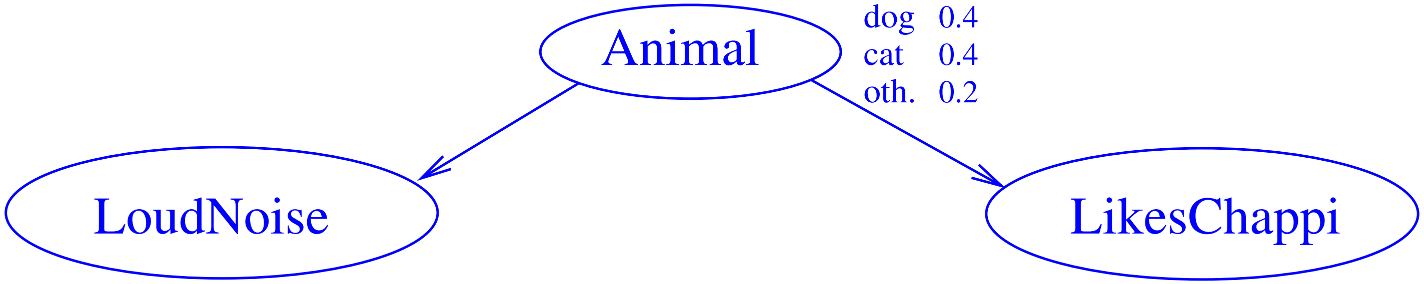
2. Then the computation is performed in terms of factor product and summing out variables from factors: $\mathbf{P}(B \mid j, m) =$
$$\alpha\, \mathbf{f}_1(B) \times \sum_{v_E} \mathbf{f}_2(E) \times \sum_{v_A} \mathbf{f}_3(A, B, E) \times \mathbf{f}_4(A) \times \mathbf{f}_5(A)$$

(B) Avoiding irrelevant computation: Repeatedly remove hidden variables that are leaf nodes. For query $P(JohnCalls \mid burglary)$:

$$\mathbf{P}(J \mid b) = \alpha P(b) \sum_{v_E} P(v_E) \sum_{v_A} P(v_A \mid b, v_E) \mathbf{P}(J \mid v_A) \sum_{v_M} P(v_M \mid v_A)$$

$\rightarrow$ The rightmost sum equals 1 and can be dropped.

Questionnaire



	P(A)
dog	0.4
cat	0.4
oth.	0.2

A	P(LN|A)
dog	0.7
cat	0.2
oth.	0.01

A	P(LC|A)
dog	0.8
cat	0.1
oth.	0.7

Question!

Say BN is the Bayesian network above. How can we compute $P(dog \mid loudnoise)$?

$P(d \mid ln) =$

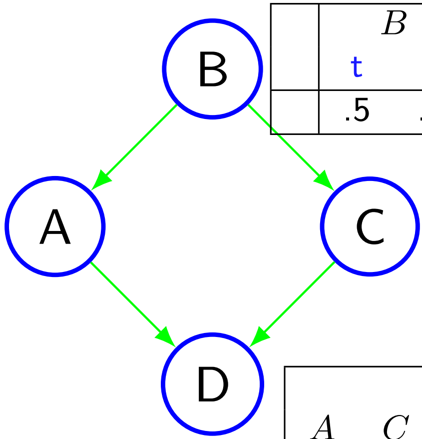
(A): $\alpha \sum_{v_{LC}} P(v_{LC} \mid d) P(ln \mid d)$

(B): $\alpha \sum_{v_{LC}} P(v_{LC} \mid d) P(ln \mid d) P(d)$

(C): $\alpha P(ln \mid d) P(d) \sum_{v_{LC}} P(v_{LC} \mid d)$

(D): $\alpha P(ln \mid d) P(d)$

Example



	B: t	B: f
	.5	.5

B	A: t	A: f
t	.7	.3
f	.1	.9

B	C: t	C: f
t	.7	.3
f	.2	.8

A	C	D: t	D: f
t	t	.9	.1
t	f	.7	.3
f	t	.8	.2
f	f	.4	.6

Naive Enumeration:

$$P(A, D = f) =$$

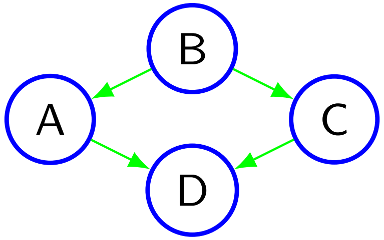
$$\sum_{b \in \{t,f\}} \sum_{c \in \{t,f\}} P(B = b, A, C = c, D = f) =$$

$$\sum_{b \in \{t,f\}} \sum_{c \in \{t,f\}} P(B = b)P(A \mid B = b)P(C = c \mid B = b)P(D = f \mid A, C = c)$$

Observation 1: $P(B = b)P(A \mid B = b)$ does not depend on C, so

$$= \sum_{b \in \{t,f\}} P(B = b)P(A \mid B = b) \sum_{c \in \{t,f\}} P(C = c \mid B = b)P(D = f \mid A, C = c)$$

Example continued



	B	
	t	f
	.5	.5

	A	
B	t	f
t	.7	.3
f	.1	.9

	C	
B	t	f
t	.7	.3
f	.2	.8

		D	
A	C	t	f
t	t	.9	.1
t	f	.7	.3
f	t	.8	.2
f	f	.4	.6

Observation 2: Let's precompute first factor F_1

$$\sum_b P(B=b)P(A \mid B=b) \underbrace{\sum_c P(C=c \mid B=b)P(D=f \mid A, C=c)}_{F_1} =$$

$$\sum_b P(B=b)P(A \mid B=b)F_1(B=b, A) = F_2(A)$$

Is F_1 a single numerical value?

Variable Elimination

The procedure operates on **factors**: functions of subsets of variables

Definition (Factor). A factor $F(V_1, \dots V_k)$ is a function mapping each combination of values of the variables $V_1, \dots, V_k$ to a number.

$\rightarrow$ Factors are not necessarily CPTs as numbers do not need to sum up to 1.

Required operations on factors:

- Restriction (setting selected variables to specific values)
- Multiplication (join tables and multiply the probabilities)
- Marginalization (summing out selected variables)

Restriction (of variable D to value $D = f$):

A	C	$F(A,C,D)$ D = t	$F(A,C,D)$ D = f
t	t	.9	.1
t	f	.7	.3
f	t	.8	.2
f	f	.4	.6

$\mapsto$

A	C	$F_2(A,C)$
t	t	.1
t	f	.3
f	t	.2
f	f	.6

Variable Elimination

The procedure operates on **factors**: functions of subsets of variables

Definition (Factor). A factor $F(V_1, \dots V_k)$ is a function mapping each combination of values of the variables $V_1, \dots, V_k$ to a number.

$\rightarrow$ Factors are not necessarily CPTs as numbers do not need to sum up to 1.

Required operations on factors:

- Restriction (setting selected variables to specific values)
- Multiplication (join tables and multiply the probabilities)
- Marginalization (summing out selected variables)

Multiplication + Marginalization (of variable C):

B	C	$F_1(B,C)$
t	t	.7
t	f	.3
f	t	.2
f	f	.8

A	C	$F_2(A,C)$
t	t	.1
t	f	.3
f	t	.2
f	f	.6

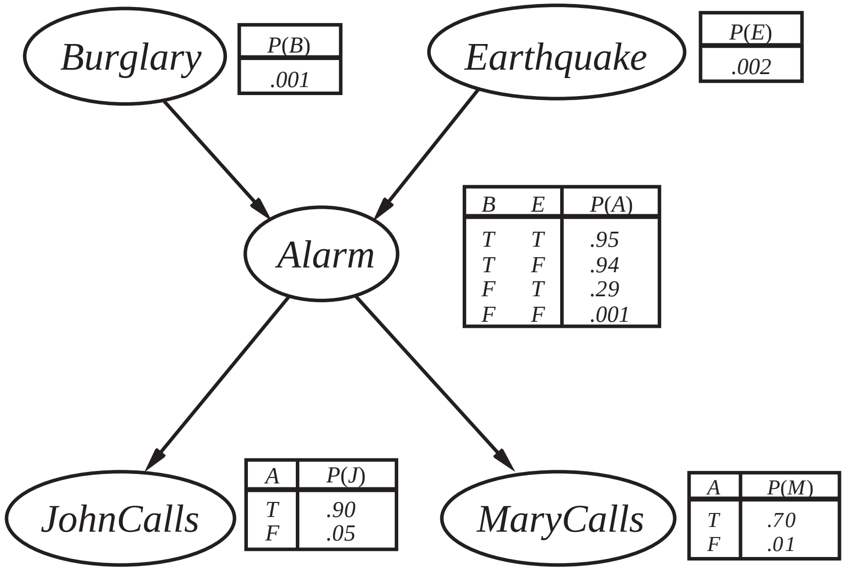
$\mapsto$

b	a	$F(B,A)$
t	t	$.7 \cdot .1 + .3 \cdot .3 = .16$
t	f	$.7 \cdot .2 + .3 \cdot .6 = .32$
f	t	$.2 \cdot .1 + .8 \cdot .3 = .26$
f	f	$.2 \cdot .2 + .8 \cdot .6 = .52$

Variable Elimination Algorithm

1. Factors = CPT in BN
2. For all variables in the evidence, restrict all factors with the observed value
3. Fix any order of the remaining variables, $X_1, \ldots, X_n$.
4. **for** $i := 1, \ldots, n$ **do**
   a. $\mathcal{F} = \{F \mid F \in \text{Factors}, F \text{ depends on } X_i\}$
   b. $T = \pi_{F \in \mathcal{F}}$ (Compute product of factors)
   c. $F_{new} = \text{Marginalize}(T, X_i)$
   d. Factors = Factors $\setminus \mathcal{F} \cup \{N\}$ (Replace all factors $\mathcal{F}$ by new factor N)

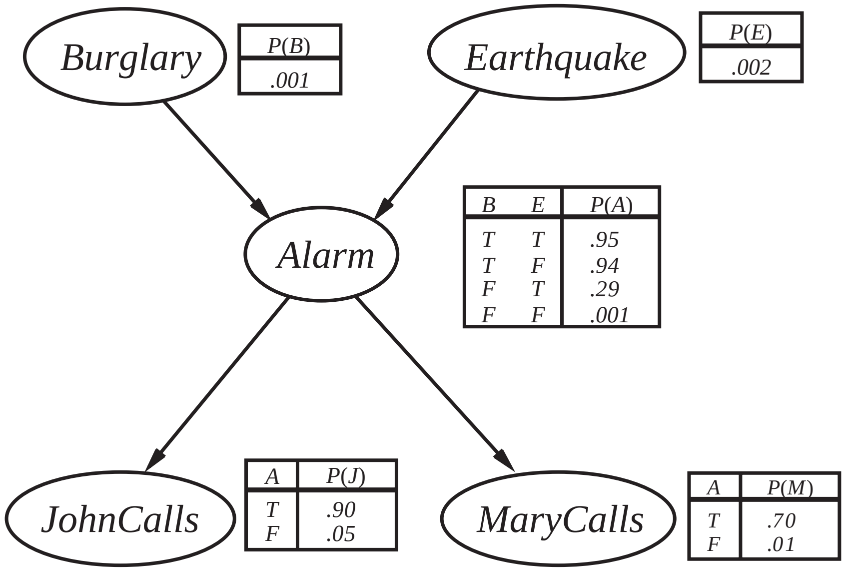
Variable Elimination Algorithm: Alarm Example



Bad ordering for computing $P(M, B = t)$: A, J, E

1. Factors = CPT in BN
2. For all variables in the evidence, restrict all factors with the observed value
3. Fix any order of the remaining variables, $X_1, \dots, X_n$.
4. **for** $i := 1, \dots, n$ **do**
   a. $\mathcal{F} = \{F \mid F \in$ Factors, F depends on $X_i\}$
   b. $T = \pi_{F \in \mathcal{F}}$ (Compute product of factors)
   c. $F_{new} =$ Marginalize(T, X_i)
   d. Factors = Factors $\setminus \mathcal{F} \cup \{N\}$ (Replace all factors $\mathcal{F}$ by new factor N)

Alarm Example



Bad ordering for computing $P(MC, B = t)$: A, J, E

$$\sum_{eq \in \{t,f\}} \sum_{jc \in \{t,f\}} \sum_{a \in \{t,f\}} P(B = t)P(EQ = eq)P(A = a \mid B = t, EQ = eq)P(JC = jc \mid A = a)P(MC \mid A = a) =$$

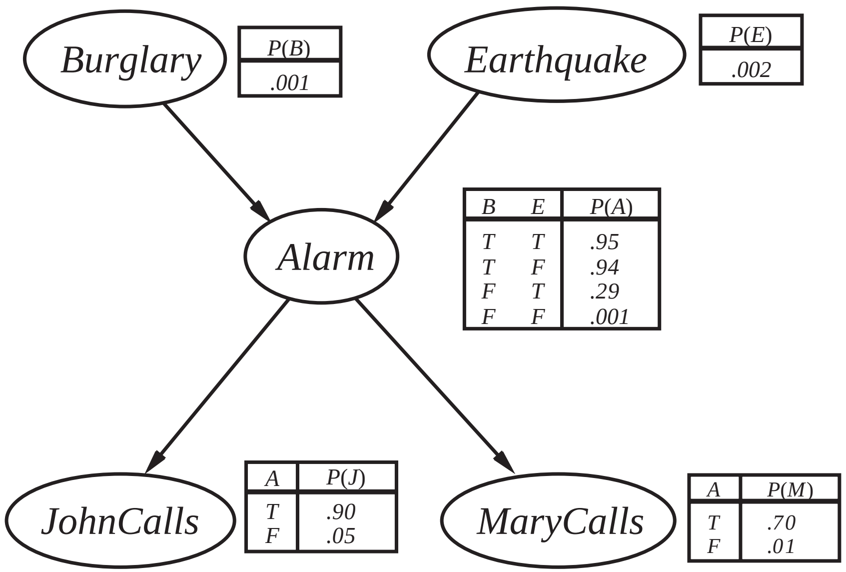
$$\sum_{eq \in \{t,f\}} \sum_{jc \in \{t,f\}} P(B = t)P(EQ = eq)F_1(eq, jc, MC) =$$

$$\sum_{eq \in \{t,f\}} P(B = t)P(EQ = eq)F_2(eq, MC) =$$

$$P(B = t)F_3(MC)$$

Largest factor (F_1) is function of 3 variables!

Alarm Example continued



Question!

What's the maximum number of columns of the largest table generated by variable elimination under ordering: J,E,A?

(A): 1 (B): 2

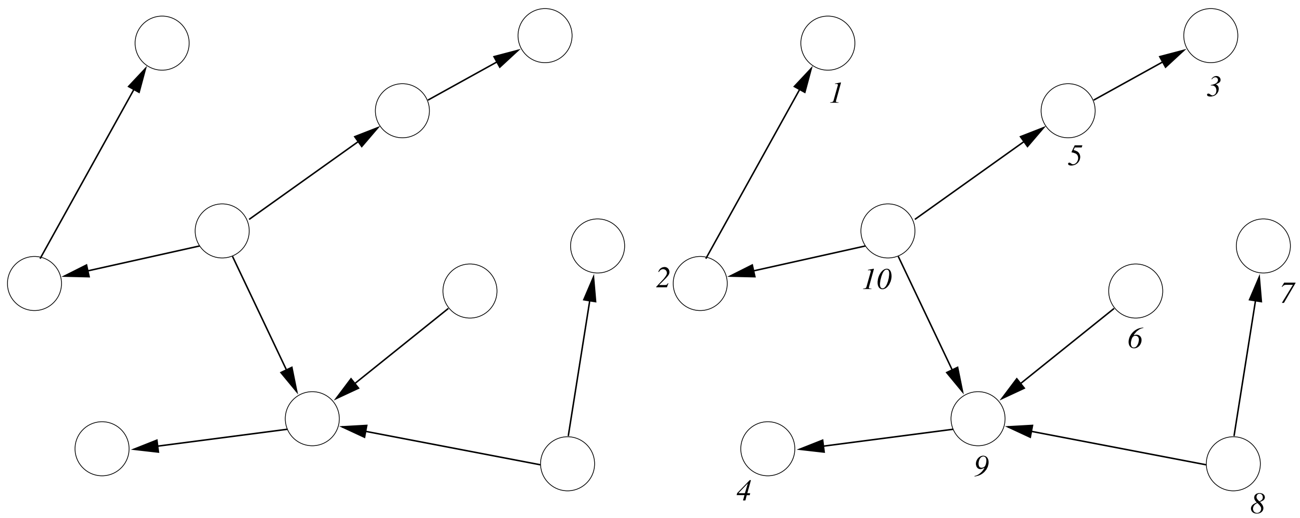
(C): 3 (D): 4

Alarm Example continued

Variable Elimination Runtime

An important easy case:

- A graph is called singly connected, or a polytree, if there is at most one undirected path between any two nodes in the graph.



→ Is our BN for Mary & John a polytree?

For singly connected network: any elimination order that "peels" variables from outside will only create factors of only one variable.

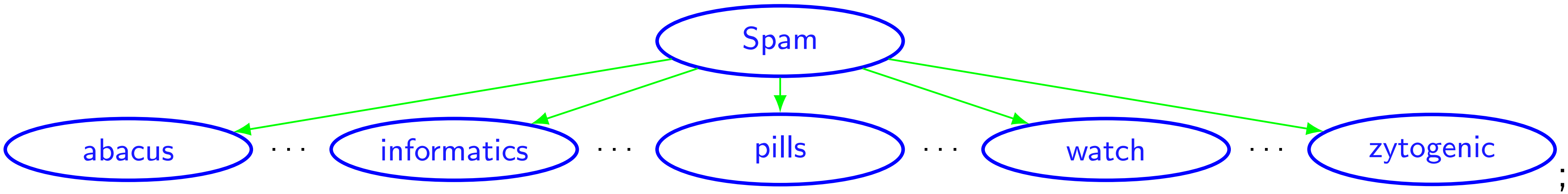
The complexity of inference is therefore linear in the total size of the network (= combined size of all conditional probability tables).

Naive Bayes Model

Example: Spam filter

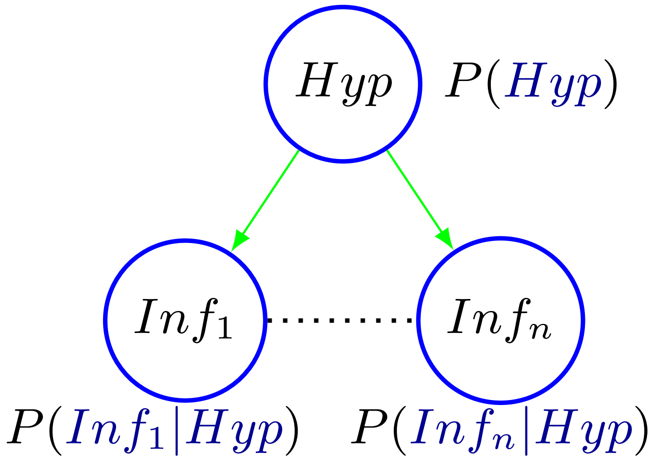
- A single query variable: Spam
- Many observable features (e.g. words appearing in the body of the message): abacus,. . . ,informatics, pills, . . . , watch,. . . , zytogenic

Network Structure:



- Inference with large number of variables possible
- Essentially how Thunderbird spam filter works

Naïve Bayes models



We want the posterior probability of the hypothesis variable Hyp given the observations $\{\text{Inf}_1 = e_1, \ldots, \text{Inf}_n = e_n\}$:

$$P(\text{Hyp}|\text{Inf}_1 = e_1, \ldots, \text{Inf}_n = e_n) = \frac{P(\text{Inf}_1 = e_1, \ldots, \text{Inf}_n = e_n|\text{Hyp})P(\text{Hyp})}{P(\text{Inf}_1 = e_1, \ldots, \text{Inf}_n = e_n)}$$
$$= \alpha \cdot P(\text{Inf}_1 = e_1|\text{Hyp}) \cdot \ldots \cdot P(\text{Inf}_n = e_n|\text{Hyp})P(\text{Hyp})$$

Note: The model assumes that the information variables are independent given the hypothesis variable.

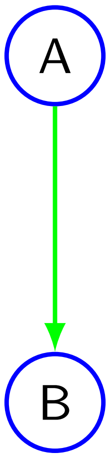
Sampling Inference

Using a BN as a Sample Generator

Observation: can use Bayesian network as random generator that produces states $\mathbf{X} = \mathbf{x}$ according to distribution P defined by the network.

$\rightarrow$ We just sample a random value for each variable always picking a value for parents(X) before picking a value for X

Example:



	A: t	A: f
	.2	.8

A	B: t	B: f
t	.7	.3
f	.4	.6

- Generate random numbers r_A, r_B uniformly from [0,1].
- Set $A = t$ if $r_A \leq .2$ and $A = f$ else.
- Depending on the value of A and r_B set B to t or f.

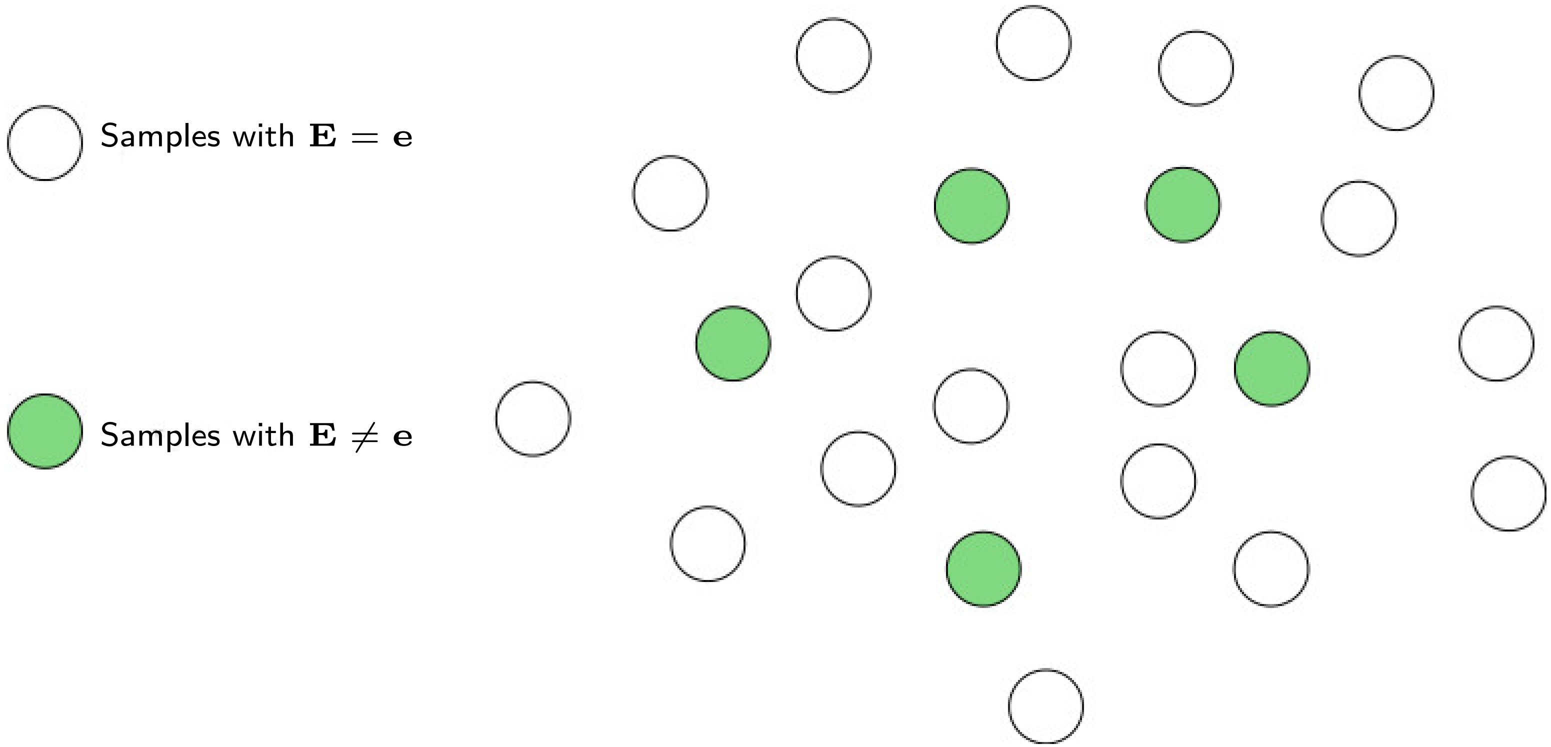
Random generation of one state: linear in size of network.

Sampling Inference

Approximate Inference from Samples

To compute an approximation of $P(\mathbf{E} = \mathbf{e})$ ($\mathbf{E}$ a subset of the variables in the Bayesian network):

- generate a (large) number of random states
- count the frequency of states in which $\mathbf{E} = \mathbf{e}$.



Accuracy

Hoeffding Bound

- p: true probability $P(\mathbf{E} = \mathbf{e})$
- s: estimate for p from sample of size n
- ϵ: an error bound > 0.

Then

$$P(|s - p| > \epsilon) \leq 2e^{-2n\epsilon^2}$$

Required Sample Size

To obtain an estimate that with probability at most δ has an error greater than ϵ, it is sufficient to take

$$n = -ln(\delta/2)/(2\epsilon^2) \text{ samples.}$$

Example

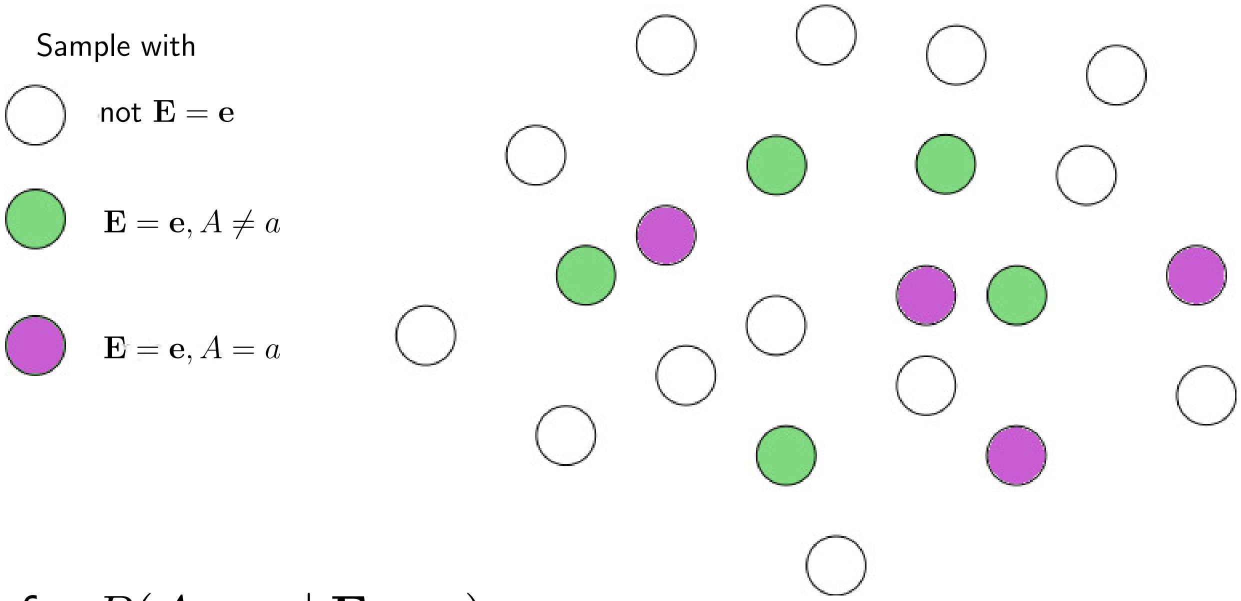
To get an error ϵ of less than 0.1 in 95% of the cases ($\delta = 0.05$), we need:

$$n > -ln(0.05/2)/(2 \cdot 0.1^2) \approx 184 \text{ samples}$$

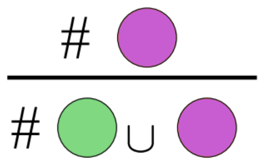
How many samples do we need if the error should be less than 0.01? 18444 samples

Rejection Sampling

The simplest approach: **Rejection Sampling**



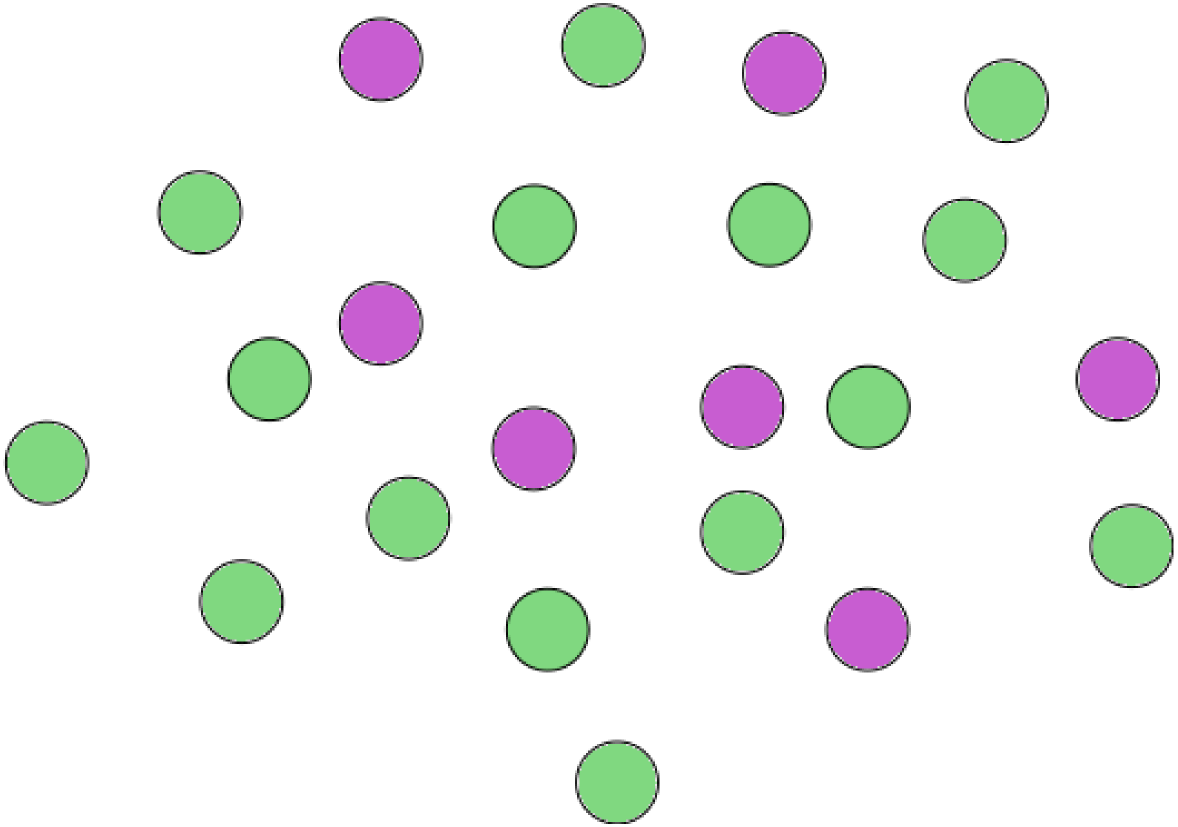
Approximation for $P(A = a \mid \mathbf{E} = \mathbf{e})$:



Sampling from the conditional distribution

Problem with rejection sampling: samples with $\mathbf{E} \neq \mathbf{e}$ are useless!

Ideally: would draw samples directly from the conditional distribution $P(\mathbf{A} \mid \mathbf{E} = \mathbf{e})$.



A Wrong Sampling Method

First idea (not to be followed)

1. Fix evidence variables to their observed states.
2. Sample from non-evidence variables.
3. Count frequency as before

Problem: This gives a sampling distribution

$$\prod_{X \in \mathbf{X} \setminus \mathbf{E}} P(X \mid \mathrm{pa}(X) \setminus \mathbf{E}, \mathrm{pa}(X) \cap \mathbf{E})$$

somewhere between $P(\mathbf{X})$ and $P(\mathbf{X} \mid \mathbf{e})$.

Likelihood Weighting

We would like to sample from

$$P(\mathbf{X}, \mathbf{e}) = \underbrace{\prod_{X \in \mathbf{X} \setminus \mathbf{E}} P(X \mid \mathrm{pa}(X) \setminus \mathbf{E}, \mathrm{pa}(X) \cap \mathbf{E})}_{\text{Part 1}} \cdot \underbrace{\prod_{E \in \mathbf{E}} P(E = e \mid \mathrm{pa}(E) \setminus \mathbf{E}, \mathrm{pa}(E) \cap \mathbf{E})}_{\text{Part 2}}$$

So instead weigh each generated sample with a weight corresponding to Part 2.

Estimate $P(X = x \mid \mathbf{e})$ as

$$\hat{P}(X = x \mid \mathbf{e}) = \frac{\sum_{sample:X=x} w(sample)}{\sum_{sample} w(sample)},$$

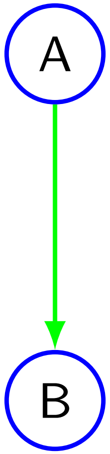
where

$$w(sample) = \prod_{E \in \mathbf{E}} P(E = e \mid \mathrm{pa}(E) = \pi) \qquad \text{(Part 2)}$$

and π is the values of $\mathrm{pa}(E)$ under $sample$ and $\mathbf{e}$.

Likelihood Sampling: Example

Sample state where B=t



	A: t	A: f
	.2	.8

A	B: t	B: f
t	.7	.3
f	.4	.6

- Initialize weight $W = 1$
- Generate random number r_A uniformly from [0,1].
- Set $A = t$ if $r_1 \leq .2$ and $A = f$ else.
- Set B to f. Update weight depending on the value of A to: $P(B = f \mid A = a)$

So:

- 20% of the time we sample $\langle A = t, B = f\rangle$ with weight
- and 80% of the time we sample $\langle A = f, B = f\rangle$ with weight

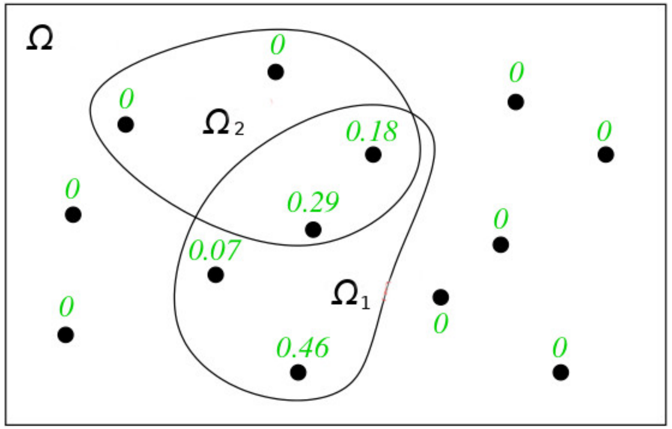
Importance Sampling I

Importance sampling

Likelihood weighting is an instance of importance sampling, where

- samples are weighted and can come from (almost) any proposal distribution.

- $\mathbf{S}$: the set of all variables defining possible worlds (includes the variables A and $\mathbf{E}$).
- Possible worlds then are tuples $\mathbf{s}$ of values



$$P(A = a \mid \mathbf{E} = \mathbf{e}) = \sum_{\mathbf{s} \in \Omega} P(A = a \mid \mathbf{S} = \mathbf{s}) P(\mathbf{S} = \mathbf{s} \mid \mathbf{E} = \mathbf{e})$$

Observe that:

- $P(\mathbf{S} = \mathbf{s} \mid \mathbf{E} = \mathbf{e})$ are the "green numbers"
- $P(A = a \mid \mathbf{S} = \mathbf{s})$ is 0 or 1, depending on whether $A = a$ in $\mathbf{s}$.

Importance Sampling II

If $\mathbf{s}_1, \dots, \mathbf{s}_n$ are sampled according to $P(\mathbf{S} = \mathbf{s} \mid \mathbf{E} = \mathbf{e})$, then

$$P(A = a \mid \mathbf{E} = \mathbf{e}) = \sum_{\mathbf{s} \in \Omega} P(A = a \mid \mathbf{S} = \mathbf{s}) P(\mathbf{S} = \mathbf{s} \mid \mathbf{E} = \mathbf{e}) \approx \frac{1}{n} \sum_{i=1}^{n} P(A = a \mid \mathbf{S} = \mathbf{s}_i)$$

Let Q be any probability distribution according to which we can sample possible worlds $\mathbf{s}_i$ (called a **proposal distribution**). Then:

$$\sum_{\mathbf{s} \in \Omega} P(A = a \mid \mathbf{S} = \mathbf{s}) P(\mathbf{S} = \mathbf{s} \mid \mathbf{E} = \mathbf{e}) = \sum_{\mathbf{s} \in \Omega} P(A = a \mid \mathbf{S} = \mathbf{s}) \frac{P(\mathbf{S} = \mathbf{s} \mid \mathbf{E} = \mathbf{e})}{Q(\mathbf{S} = \mathbf{s})} Q(\mathbf{S} = \mathbf{s})$$

If $\mathbf{s}_1, \dots, \mathbf{s}_n$ are sampled according to Q, then the right side is approximated by

$$\frac{1}{n} \sum_{i=1}^{n} P(A = a \mid \mathbf{S} = \mathbf{s}_i) \frac{P(\mathbf{S} = \mathbf{s}_i \mid \mathbf{E} = \mathbf{e})}{Q(\mathbf{S} = \mathbf{s}_i)}$$

which then is also an approximation of $P(A = a \mid \mathbf{E} = \mathbf{e})$.

Importance Sampling III

Importance Sampling

- Generate random samples $\mathbf{s}_i$ according to some proposal distribution Q.
- Estimate $P(A = a \mid \mathbf{E} = \mathbf{e})$ by $\frac{1}{n}\sum_{i=1}^{n} P(A = a \mid \mathbf{S} = \mathbf{s}_i)\frac{P(\mathbf{S}=\mathbf{s}_i|\mathbf{E}=\mathbf{e})}{Q(\mathbf{S}=\mathbf{s}_i)}$

Observations and Issues

- $P(A = a \mid \mathbf{S} = \mathbf{s}_i)$ is still only 0-1-valued
- $P(\mathbf{S} = \mathbf{s}_i \mid \mathbf{E} = \mathbf{e})$ is usually easy to compute, because $\mathbf{s}_i$ contains a value for all variables.
- $P(\mathbf{S} = \mathbf{s}_i \mid \mathbf{E} = \mathbf{e}) = 0$ if $\mathbf{s}_i$ does not satisfy $\mathbf{E} = \mathbf{e}$, i.e. samples that do not comply with the evidence don't count.
- The best approximation is obtained when Q is close to (identical to) $P(\mathbf{S} \mid \mathbf{E} = \mathbf{e})$.

Summary

- Bayesian networks (BN) are a wide-spread tool to model uncertainty, and to reason about it. A BN represents conditional independence relations between random variables. It consists of a graph encoding the variable dependencies, and of conditional probability tables (CPTs).
- Probabilistic inference requires to compute the probability distribution of a set of query variables, given a set of evidence variables whose values we know. The remaining variables are hidden.
- Inference by enumeration takes a BN as input, then applies Normalization+Marginalization, the Chain rule, and exploits conditional independence. This can be viewed as a tree search that branches over all values of the hidden variables.
- Variable elimination avoids unnecessary computation. It runs in polynomial time for poly-tree BNs. In general, exact probabilistic inference is **#P**-hard. Approximate probabilistic inference methods exist.
- When exact inference is infeasible, approximate inference can be used to obtaain estimates faster.
- Approximate inference can be done via sampling from the distribution the BN represents. We can apply a weighting function to correct our estimations when we are interested in a diffrent distribution.

Topics We Didn't Cover Here

- More advance inference by sampling: A whole zoo of methods for doing this exists.
- Clustering: Pre-combining subsets of variables to reduce the runtime of inference.
- Compilation to SAT: More precisely, to "weighted model counting" in CNF formulas. Model counting extends DPLL with the ability to determine the number of satisfying interpretations. Weighted model counting allows to define a mass for each such interpretation (= the probability of an atomic event).
- Dynamic BN: BN with one slice of variables at each "time step", encoding probabilistic behavior over time.
- Relational BN: BN with predicates and object variables.
- First-order BN: Relational BN with quantification, i.e., probabilistic logic. E.g., the BLOG language developed by Stuart Russel and co-workers.

Reading

- *Chapter 8: Reasoning with Uncertainty* from the book "Artificial Intelligence:Foundations of Computational Agents" (2nd edition). In particular:
  - Section 8.4 "Probabilistic Inference"
  - Section 8.4.1 "Variable Elimination for Belief Networks"
  - Section 8.6 "Stochastic Simulation"
  - Section 8.6.5 "Importance Sampling"

For a further reading on the topic you can also read:

- *Chapter 14: Probabilistic Reasoning* from the book "Artificial Intelligence:A Modern Approach (4th edition)