Bayesian Networks

BY: MOHAMAD ALSABBAGH

Outlines

- Introduction
  - Bayes Rule
- Bayesian Networks (BN) Representation
  - Size of a Bayesian Network
- Inference via BN
- BN Learning
- Dynamic BN

Introduction

- Conditional Probability: $P(x|y) = \frac{P(x,y)}{P(y)}$
- Product Rule: $P(x,y) = P(x|y)P(y)$
- Chain Rule: $$\begin{aligned} P(X_1, X_2, \ldots X_n) &= P(X_1)P(X_2|X_1)P(X_3|X_1,X_2)\ldots \\ &= \prod_{i=1}^{n} P(X_i|X_1,\ldots,X_{i-1}) \end{aligned}$$
- Independence: $\forall x, y : P(x,y) = P(x)P(y)$
- Conditional Independence: $\forall x, y, z : P(x,y|z) = P(x|z)P(y|z)$

Bayes Rule

$$P(A|B) = \frac{P(B|A)P(A)}{P(B)}$$

$$P(Cause|Evidence) = \frac{P(Evidence|Cause)P(Cause)}{P(Evidence)}$$

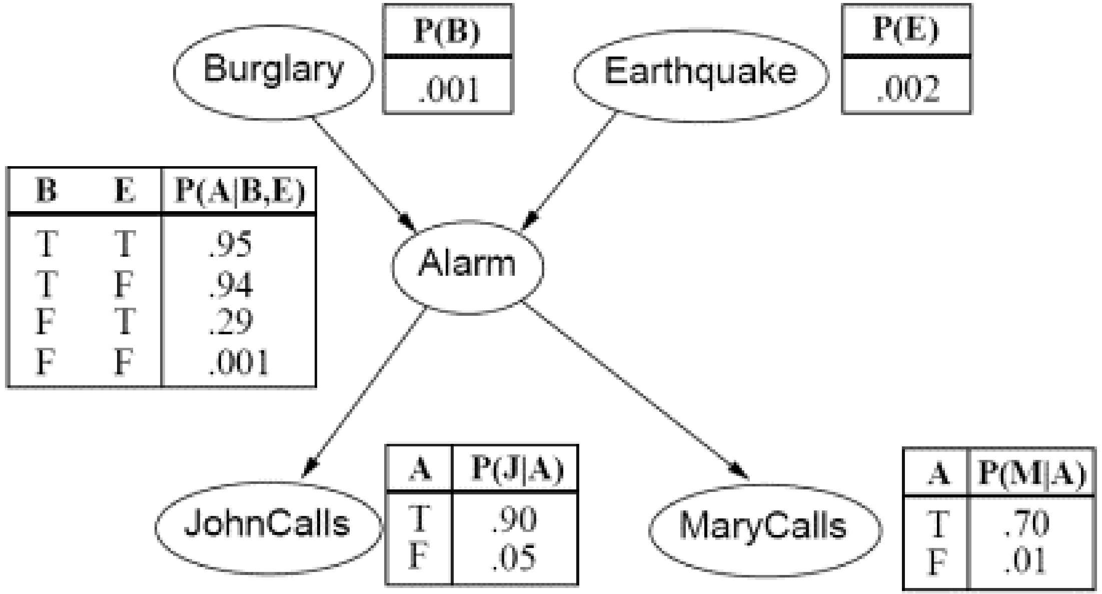
Thomas Bayes
(1701 - 1761)

Bayesian Networks (BN) Representation

- A directed, acyclic graph (DAG)
- One node per random variable.
- Each Node is a conditional distribution represented by a conditional probability table (CPT) given its parents.
- BN encodes joint distribution efficiently:
  - As a product of local conditional distribution

$$P(x_1, x_2, \ldots x_n) = \prod_{i=1}^{n} P(x_i | parents(X_i))$$

Bayesian Networks (BN) Representation

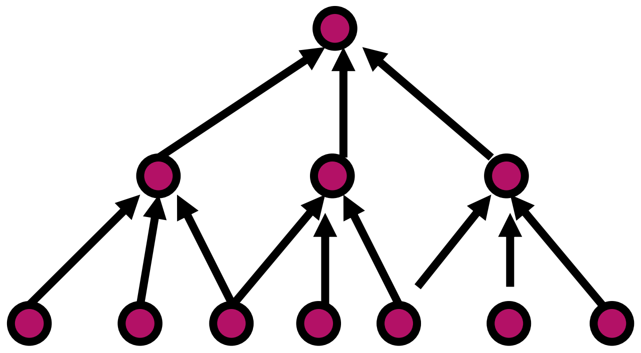


$$P(x_1, x_2, \ldots x_n) = \prod_{i=1}^{n} P(x_i | parents(X_i))$$

- Examples:
  - $P(b, \neg e, a, m, \neg j) = P(b) * P(\neg e) * P(a \mid b, \neg e) * P(m \mid a) * P(\neg j | a)$
    $= 0.001 * 0.998 * 0.94 * 0.7 * 0.1 = 0.0000656684$
  - $P(b, \neg e, a, m, j) = 0.001 * 0.998 * 0.94 * 0.7 * 0.9 = 0.0005910156$

Size of a Bayesian Network

- Full joint distribution over N Boolean variables table requires 2^N numbers in the table.
- BN with N nodes and up to k parents representation size is $O(N * 2^K)$
- Benefits:
  - Provide a huge saving in space
  - Easier to calculate local CPTs
  - Faster to answer queries
- For N = 11 and k = 3
  BN size is 88 vs 2048 numbers in CPT
- N = 30 and k = 5
- BN requires 960 and full joint distribution requires over billion.



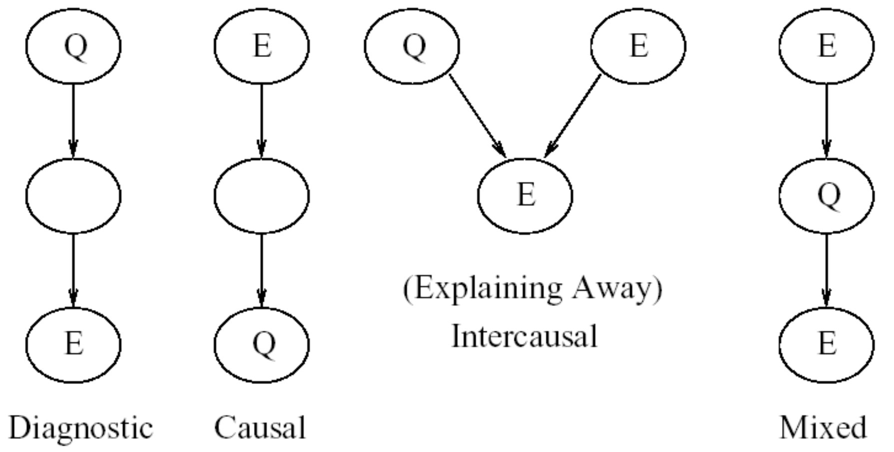
k = 3

Inference via BN

- What is Inference?
- Exact Inference in BN:
  - Enumeration
  - Variable Elimination
- Approximate Inference in BN:
  - Sampling

What is Inference?

- Inference: Compute posterior probability distribution for a set of query variables given some observed event.
- Q query variable, E evidence variable
- Examples (Alarm BN):
  - P(b | j, m) (diagnostic)
  - P(e | m) (diagnostic)
  - P(m | e) (causal)
  - P(a | m, b) (Mixed)
  - P(b | a, e) (inter-causal)



Exact Inference in BN

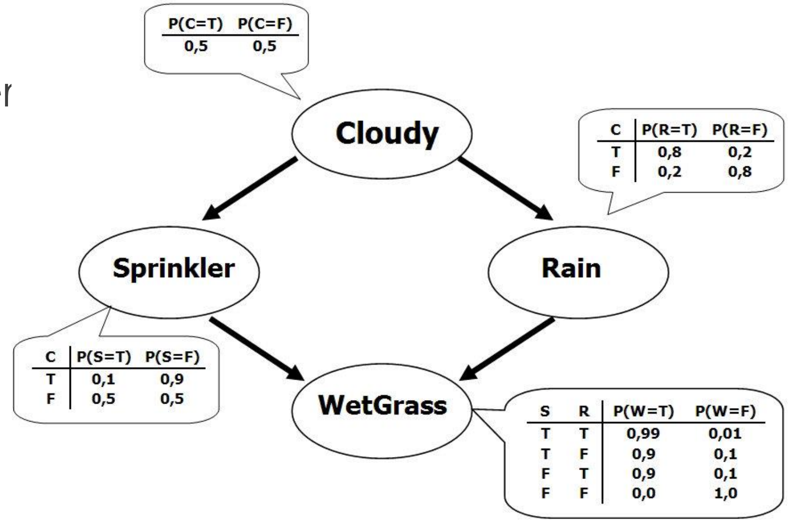
- Enumeration:
  - Summing terms from the full join distribution
- Examples:
  - P(b | j, m) = 0.284
  - P(b | j, m) = $\frac{P(b,j,m)}{P(j,m)} = \alpha\, P(b,j,m) = \alpha\, P(b,j,m,A,E) = \alpha\, \sum_a \sum_e P(b,j,m,a\ e) = \alpha * 0.00059224$
  - P(¬b | j, m) = $\alpha * 0.0014919$
  - P(b | j, m)+P(¬b | j, m) = 1 ➔ $\alpha \approx 479.53532$

Exact Inference in BN

- Variable Elimination:
  - Improve Enumeration Algorithm by eliminating repeated calculations.
  - Store intermediate results.
  - Elimination order of hidden variables matters.
  - Every variable that is not an ancestor of a query variable or an evidence variable is irrelevant to the query
- Algorithm:
  - Query = $P(Q|E_1 = e_1, \ldots E_k = e_k)$
  - Local CPTs (but instantiated by evidence)
  - While there are still hidden variables (not Q or E_i)
    - Pick hidden variable H
    - Join all factors mentioning H
    - Eliminate (sum out) H
  - Join all remaining factors and normalize

Approximate Inference in BN

- Sampling:
  - Sampling is a lot like repeated simulation
  - Generate N random samples to compute approximate posterior probability.
- Why:
  - Getting samples is faster than computing the right answer
  - Learning: get samples from a distribution we don't know.



Approximate Inference in BN

- Sampling in BN:
  - Prior Sampling
    - Each variable is sampled according to the condition distribution given.
    - $P(x_1, \ldots, x_m) = N_{PS}(x_1, \ldots, x_m)/N$
  - Rejection Sampling
    - No point keeping all samples around.
    - P (C | s) same as before but reject samples which don't have S = s (sprinkler evidence)
  - Likelihood Weighting:
    - Avoids inefficiency from rejecting samples by generating samples that are consistent with the evidence e.

Approximate Inference in BN

- Sampling in BN:
  - Markov chain Monte Carlo (Gibbs Sampling):
    - Generates each sample from a previous sample by doing a random modification.
    - It is conditional on the current values of the variables in the Markov blanket.
    - The algorithm wanders randomly around the state space flipping one variable at a time but keeping evidence variable fixed.
  - Gibbs Algorithm:
    - Fix evidence R= r (as an example)
    - Initialize other variables randomly
    - Repeat on non-evidence variable.

BN Learning

- In practical settings BN is unknown and we need to use data to learn.
- Given training data (prior knowledge), we need to estimate the graph topology (network structure) and the parameters in joint distribution.
- Learning the structure is harder than BN parameters.
- Possible cases of the problem:

Case	**BN Structure**	**Observability**	**Proposed Learning Method**
1	Known	Full	Maximum likelihood estimate
2	Known	Partial	EM (Expectation Maximization) MCMC
3	Unknown	Full	Search through model space
4	Unknown	Partial	EM + Search through model space

Dynamic BN

- DBN is a BN that represents a temporal probability.
- In general each time slice of DBN can have any numbers of variables X_t and evidence variables.
- Model structure & parameters don't change overtime.
- Inference:
  - Filtering: $P(X_t \mid e_{1:t})$
  - Prediction: $P(X_{t+k} \mid e_{1:t})$; $k > 0$
  - Smoothing: $P(X_k \mid e_{1:t})$; $0 \leq k \leq t$
  - Most likely explanation: (given sequence of observation we want to find best states)
  - Exact inference
    - Variable elimination
  - Approximate inference
    - Particle filtering (an improvement on Likelihood Weighting)
    - MCMC

Dynamic BN

- Special cases of DBN:
  - Each HMM is a DBN
    - Discrete State Variables
    - Used to model sequences of events.
    - Single state variable and single evidence variable
  - Each DBN can be converted to HMM by combining all state variables to mega variable with all possible cases.
  - DBN with 20 Boolean states and 3 parents as max the transition model for it will require only 160 probabilities while corresponding HMM needs 2^{40} or ~trillion in transition model.

Dynamic BN

- Special cases of DBN:
  - Every Kalman Filters is a DBN
    - Continuous State Variables, with Gaussian Distribution
      - Gaussian distribution is fully defined by its mean and variance
    - Used to model noisy continuous observations
    - Example: predict a motion of a bird in a Jungle.
  - Not every DBN can be converted to Kalman Filter.
    - DBN allow no-linear distribution, that require both discrete and continues variables
which Kalman doesn't allow.

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

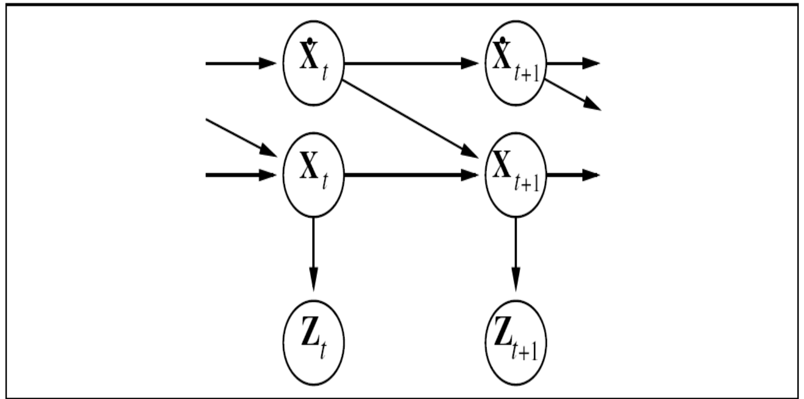


Figure 15.7 Bayesian network structure for a linear dynamical system with position $\mathbf{X}_t$, velocity $\dot{\mathbf{X}}_t$, and position measurement $\mathbf{Z}_t$.

Dynamic BN Constructing

- Required information
  - Prior distributions over state variables $P(X_0)$
  - The transition model $P(X_{t+1} | X_t)$
  - The sensor or observation model $P(E_t | X_t)$

Further resources

- Tools (Belief and Decision Networks)
  - http://www.aispace.org/downloads.shtml
- Books:
  - Artificial Intelligence A Modern Approach by Russell & Norvig.

Summary

- BN become extremely popular models.
- BN used in many applications like:
  - Machine Learning
  - Speech Recognition
  - Bioinformatics
  - Medical diagnosis
  - Weather forecasting
- BN is intuitively appealing and convenient for representation of both causal and probabilistic semantics.

Q&A

Thank you