Bayesian Networks

Fabrizio Riguzzi

•Acknowledgments: some slides from
- Andrew Moore's tutorials http://www.autonlab.org/tutorials/
- Irina Rish and Moninder Singh's tutorial http://www.research.ibm.com/people/r/rish/

Summary

- Conditional independence
- Definition of Bayesian network
- Inference
- Learning
- Markov networks

Domain Modeling

- We use a set of random variables to describe the domain of interest
- Example: home intrusion detection system, variables:
  - Earthquake E, values e_1=no, e_2=moderate, e_3=severe
  - Burglary B, values: b_1=no, b_2=yes through door, b_3=yes through window
  - Alarm A, values a_1=no, a_2=yes
  - Neighbor call N, values n_1=no, n_2=yes

Inference

- We would like to answer the following questions
  - What is the probability of a burglary through the door? (compute $P(b_2)$, belief computation)
  - What is the probability of a burglary through the door given that the neighbor called ? (compute $P(b_2|n_2)$, belief updating)

Inference

- What is the probability of a burglary through the door given that there was a moderate earthquake and the neighbor called ? (compute $P(b_2|n_2,e_2)$, belief updating)
- What is the probability of a burglary through the door and of the alarm ringing given that there was a moderate earthquake and the neighbor called ? (compute $P(a_2,b_2|n_2,e_2)$, belief updating)
- What is the most likely value for burglary given that the neighbor called ($\mathrm{argmax}_b\ P(b|n_2)$, belief revision)

Types of Problems

- Diagnosis: P(cause|symptom)=?
- Prediction: P(symptom|cause)=?
- Classification: $\mathrm{argmax}_{class}\ P(class|data)$

Inference

- In general, we want to compute the probability P(**q**|**e**)
  - of a query **q** (assignment of values to a set of variables **Q**)
  - given the evidence **e** (assignment of values to a set of variables **E**)

Joint Probability Distribution

- The **joint probability distribution** (jpd) of a set of variables **U** is given by P(**u**) for all values **u**
- For our example
  - **U**={E,B,A,N}
  - We have the jpd if we know P(**u**)=P(e,b,a,n) for all the possible values e, b, a, n.

Inference

- If we know the jpd, we can answer all the possible queries:

$$P(\mathbf{q}|\mathbf{e}) = \frac{P(\mathbf{q},\mathbf{e})}{P(\mathbf{e})}$$
$$= \frac{\sum_{\mathbf{x}, \mathbf{X}=\mathbf{U}\setminus\mathbf{Q}\setminus\mathbf{E}} P(\mathbf{x},\mathbf{q},\mathbf{e})}{\sum_{\mathbf{y}, \mathbf{Y}=\mathbf{U}\setminus\mathbf{E}} P(\mathbf{y},\mathbf{e})}$$

Computational Cost

- If we have n binary variables (|**U**|=n), knowing the jpd requires storing $O(2^n)$ different values.
- Even if we had the space to store all the 2^n different values, computing P(**q**|**e**) would require $O(2^n)$ operations
- Impractical for real world domains
- How to avoid the space and time problems? Use conditional independence assertions

Conditional Independence

- **X**, **Y**, **Z** vectors of multivalued variables
- **X** and **Y** are **conditionally independent** given **Z** if

$$\forall \boldsymbol{x}, \boldsymbol{y}, \boldsymbol{z} : P(\boldsymbol{y}, \boldsymbol{z}) > 0 \rightarrow P(\boldsymbol{x} | \boldsymbol{y}, \boldsymbol{z}) = P(\boldsymbol{x} | \boldsymbol{z})$$

- We write $I<\mathbf{X},\mathbf{Z},\mathbf{Y}>$
- Special case: **X** and **Y** are **independent** if

$$\forall \boldsymbol{x}, \boldsymbol{y} : P(\boldsymbol{y}) > 0 \rightarrow P(\boldsymbol{x} | \boldsymbol{y}) = P(\boldsymbol{x})$$

- We write $I<\mathbf{X},\{\},\mathbf{Y}>$

Chain Rule

- n random variables $X_1,\ldots,X_n$
- Let $\mathbf{U}=\{X_1,\ldots,X_n\}$
- Joint event $\mathbf{u}=(x_1,\ldots,x_n)$
- Chain rule:

$$\begin{aligned} P(\boldsymbol{u}) &= P(x_{1,} \ldots, x_n) \\ &= P(x_n | x_{n-1} \ldots, x_1) \ldots P(x_2 | x_1) P(x_1) \\ &= \prod_{i=1}^{n} P(x_i | x_{i-1} \ldots, x_1) \end{aligned}$$

Conditional Independence

- $\mathbf{\Pi}_i$ is a subset of $\{X_{i-1},\ldots,X_1\}$ such that
- X_i is conditionally independent of $\{X_{i-1},\ldots,X_1\}\backslash\mathbf{\Pi}_i$ given $\mathbf{\Pi}_i$

  $P(x_i|x_{i-1}\ldots,x_1)=P(x_i|\boldsymbol{\pi_i})$ whenever $P(x_{i-1},\ldots,x_1)>0$
- where $\boldsymbol{\pi}_i$ is a set of values for $\mathbf{\Pi}_i$
- $\mathbf{\Pi}_i$ **parents** of X_i

Conditional Independence

- Knowing $\mathbf{\Pi}_i$ for all i we could write

$$\begin{aligned}
P(\boldsymbol{u}) &= P(x_{1,}\ldots,x_n)\\
&= P(x_n|x_{n-1}\ldots,x_1)\ldots P(x_2|x_1)P(x_1)\\
&= P(x_n|\boldsymbol{\pi_n})\ldots P(x_2|\boldsymbol{\pi_2})P(x_1|\boldsymbol{\pi_1})\\
&= \prod_{i=1}^{n} P(x_i|\boldsymbol{\pi_i})
\end{aligned}$$

Conditional Independence

- In order to compute P(**u**) we have to store

$$P\left(x_i \middle| \boldsymbol{\pi_i}\right)$$

- for all values x_i and $\boldsymbol{\pi}_i$
- $P(x_i \mid \boldsymbol{\pi}_i)$: Conditional probability table
- If $\mathbf{\Pi}_i$ is much smaller than the set $\{X_{i-1},\ldots,X_1\}$, then we have huge savings
- If k is the maximum number of parents of a variable, then storage is $O(n2^k)$ instead of $O(2^n)$

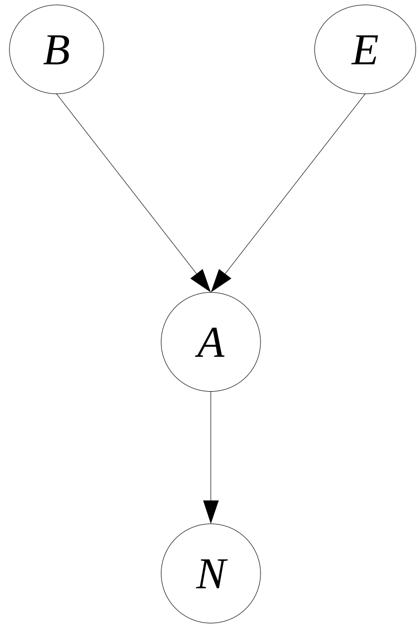
Graphical Representation

- We can represent the conditional independence assertions using a directed graph with a node per variable
- $\mathbf{\Pi}_i$ is the set of parents of X_i
- The graph is acyclic

Example Network

- Variable order: E,B,A,N
- Independencies

$$P(e)$$
$$P(b|e) = P(b)$$
$$P(a|b,e) = P(a|b,e)$$
$$P(n|a,b,e) = P(n|a)$$



Conditional Probability Tables

- Earthquake E, e_1=no, e_2=moderate, e_3=severe
- Burglary B, : b_1=no, b_2=yes through door, b_3=yes through window
- Alarm A, a_1=no, a_2=yes
- Neighbor call N, n_1=no, n_2=yes

P(B)	
B=no	0,7
B=door	0,1
B=windows	0,2

P(E)	
E=no	0,6
E=moderate	0,2
E=severe	0,2

P(A|EB)	no,no	no,do	no,wi	mo,no	mo,do	mo,wi	se,no	se,do	se,wi
no	0,99	0,1	0,2	0,8	0,08	0,1	0,7	0,05	0,07
yes	0,01	0,9	0,8	0,2	0,92	0,9	0,3	0,95	0,93

P(N|A)	A=no	A=yes
N=no	0,9	0,05
N=yes	0,1	0,95

Bayesian Network

- A **Bayesian network** [Pearl 85] (BN) B is a couple (G,Θ) where
  - G is a directed acyclic graph (DAG) (V,E) where
    - V is a set of vertices $\{X_1,...,X_n\}$
    - E is a set of edges, i.e. A set of couples (X_i,X_j)
    - $<X_1,...,X_n>$ is a topological sort of G, i.e. $(X_i,X_j)\in E \Rightarrow i<j$
  - Θ is a set of conditional probability tables (cpts)

$$\{\theta_{x_i|\boldsymbol{\pi}_i}\in R \mid i=1,\ldots,n,\ x_i\in X_i,\ \boldsymbol{\pi}_i\in \Pi_i\}$$

  - where $\mathbf{\Pi}_i$ is the set of parents of X_i

Bayesian Network

- BNs are also called belief networks or directed acyclic graphical models

Bayesian Network

- A BN (G,Θ) **represents** a jpd P iff
  - given its parents in G, each variable is independent of its other predecessors

$$P(x_i|x_{i-1}\ldots,x_1)=P(x_i|\boldsymbol{\pi_i})$$

  - $\theta_{x_i|\boldsymbol{\pi}_i}=P(x_i|\boldsymbol{\pi}_i)$ for all i and $\boldsymbol{\pi}_i$
- In this case

$$P(x_1,\ldots,x_n)=\prod_{i=1}^{n} P(x_i|\boldsymbol{\pi_i})$$
$$=\prod_{i=1}^{n}\theta_{x_i|\boldsymbol{\pi}_i}$$

How to Build a Bayesian Network

- Choose an ordering X_1 .. X_n for the variables
- For i = 1 to n:
  - Add X_i node to the network
  - Set $\boldsymbol{\Pi}_i$ to be a minimal subset of $\{X_1 \ldots X_{i-1}\}$ such that we have conditional independence of X_i and all other members of $\{X_1 \ldots X_{i-1}\}$ given $\boldsymbol{\Pi}_i$
  - Assign a value to $P(x_i|\boldsymbol{\pi}_i)$ for all the values of x_i and $\boldsymbol{\pi}_i$

Building a Bayesian Network

- Usually the expert considers a variable X as a child of Y if Y is a **direct cause** of X
- Correlation and causality are related but are **not** the same thing
  - See the book [Pearl 00]

Pathfinder system [Suermondt et al. 90]

- Diagnostic system for lymph-node diseases.
- 60 diseases and 100 symptoms and test-results.
- 14,000 probabilities
- Expert consulted to make net.
- 8 hours to determine variables.
- 35 hours for net topology.
- 40 hours for probability table values.

Pathfinder system [Suermondt et al. 90]

- Pathfinder is now outperforming the world experts in diagnosis.
- Being extended to several dozen other medical domains.

How to Tell Independence

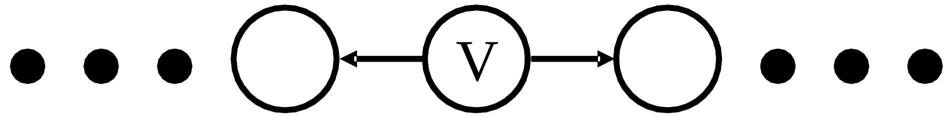
- There is a relatively simple algorithm for determining whether two variables in a Bayesian network are conditionally independent: **d-separation**.
- Definition: *X* and *Z* are **d-separated** by a set of evidence variables *E* iff every undirected path from *X* to *Z* is “blocked”, where a path is “blocked” iff one or more of the following conditions is true: ...

Blocked Path

There exists a variable *V* on the path such that

it **is** in the evidence set *E*

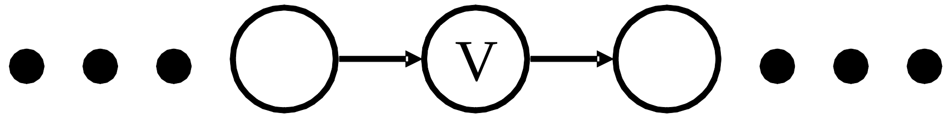
the arcs putting *V* in the path are "tail-to-tail"



Or, there exists a variable *V* on the path such that

it **is** in the evidence set *E*

the arcs putting *V* in the path are "tail-to-head"



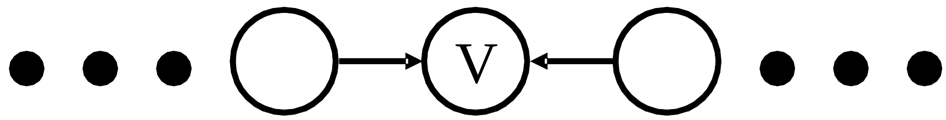
Blocked Path

- … Or, there exists a variable *V* on the path such that

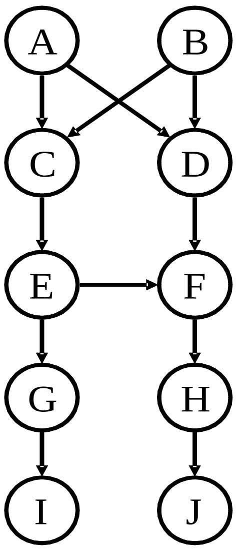
it **is NOT** in the evidence set *E*

neither are any of its descendants

the arcs putting *V* on the path are "head-to-head"

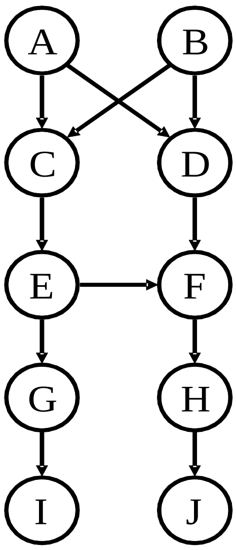


Example



- I<C, {}, D>?
- I<C, {A}, D>?
- I<C, {A, B}, D>?
- I<C, {A, B, J}, D>?
- I<C, {A, B, E, J}, D>?

Example



- I<C, {}, D>?No
- I<C, {A}, D>?No
- I<C, {A, B}, D>?Yes
- I<C, {A, B, J}, D>?No
- I<C, {A, B, E, J}, D>?Yes

Inference with Bayesian Networks

- With a Bayesian Network we save space, do we also save time?
- Do we have to use the formula

$$P(\mathbf{q}|\mathbf{e}) = \frac{\sum_{\mathbf{x},\, \mathbf{X}=\mathbf{U}\setminus\mathbf{Q}\setminus\mathbf{E}} P(\mathbf{x},\mathbf{q},\mathbf{e})}{\sum_{\mathbf{y},\, \mathbf{Y}=\mathbf{U}\setminus\mathbf{E}} P(\mathbf{y},\mathbf{e})}$$

- to compute P(**q**|**e**)?

Inference with Bayesian Networks

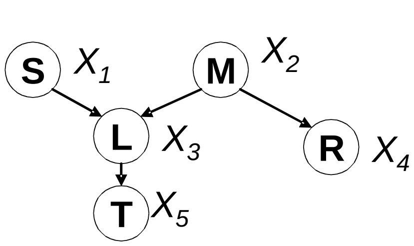
- There are quicker algorithms
  - Exact methods for polytrees
    - Belief propagation
  - Exact methods for general networks
    - Junction tree
    - Variable elimination
  - Approximate methods for general networks:
    - Stochastic simulation
    - Loopy belief propagation
    - Variational methods,

Complexity of Inference

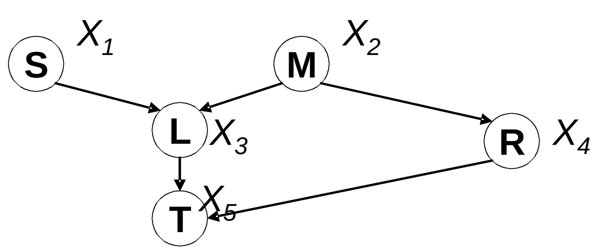
- Exact inference with BN is #P-complete
- #P-complete: a special case of NP-complete problems
  - The answer to a #P-complete problem is the number of solutions to a NP-complete problem

Polytrees

A polytree is a directed acyclic graph in which no two nodes have more than one path between them.



A polytree



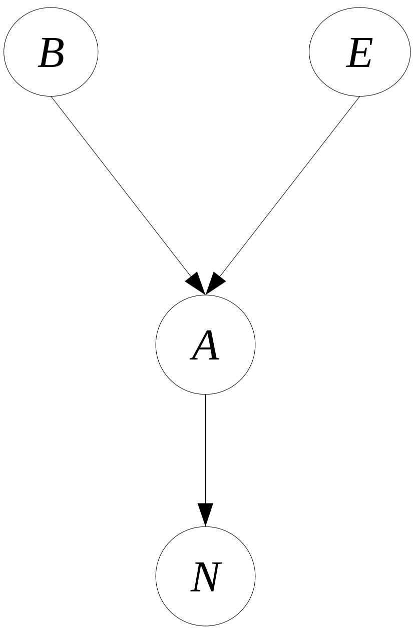
Not a polytree

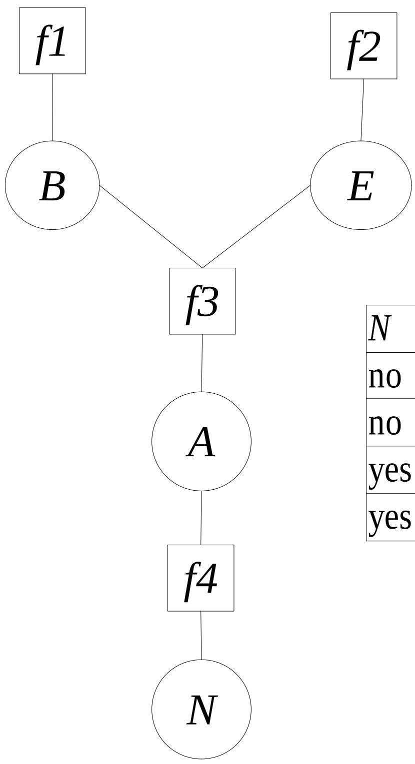
- i.e. There are no cycles in the corresponding undirected graph

Belief Propagation [Pearl 88]

- Best presented over Factor Graphs
- A **Factor Graph** is a bipartite graph (V,F,E) where vertices V index the variables, the vertices F index the families (factors), and edges E are connected between V and F
- A factor, given the values of the variables involved in the factor, returns a non-negative number.
- A family in a BN can be seen as a factor

Example Network





N	*A*	*f4*
no	no	0,9
no	yes	0,05
yes	no	0,1
yes	yes	0,05

Messages

- The message from a variable node X to a neighbor factor node f is

$$\mu_{X \to f}(x) = \prod_{h \in nb(X) \setminus X} \mu_{h \to X}(x)$$

- where nb(X) is the set of neighbor of X, the set of factors X appears in
- The message from a factor to a variable is

$$\mu_{f \to X}(x) = \sum_{\neg\{X\}} \left(f(\mathbf{x}) \prod_{Y \in nb(f) \setminus X} \mu_{Y \to f}(y) \right)$$

- Where nb(f) is the set of arguments of f and the sum is over all of these except X

Belief

- The unnormalized belief of each variable X_i in iteration k can be computed from the equation

$$b_i(x_i) = \prod_{f \in nb(X_i)} \mu_{f \to X_i}(x_i)$$

- For example, if X_1 has 3 values x_{11}, x_{12}, x_{13}, their probabilities are
- $B=b_1(x_{11})+b_1(x_{12})+b_1(x_{13})$
- $P(x_{11})=b_1(x_{11})/B$ $P(x_{12})=b_1(x_{12})/B$ $P(x_{13})=b_1(x_{13})/B$

Incorporation of Evidence

- For each factor f, for each combination of values of the arguments that is incompatible with the evidence, f(**x**) is set to 0
- Example: evidence N=yes, factor f4 becomes

N	*A*	*f4*
no	no	0,9
no	yes	0,05
yes	no	0,1
yes	yes	0,05

N	*A*	*f4*
no	no	0
no	yes	0
yes	no	0,1
yes	yes	0,05

Algorithm

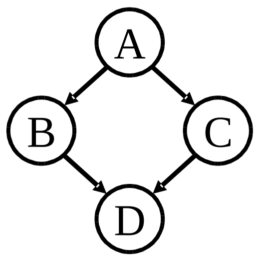
- Initialize all messages to 1 or randomly
- Loop
  - Select an arc
  - Compute the value of the message on the arc
- Until the messages do not change anymore
- If the network is a polytree, this algorithm converges
- Various strategies for selecting the arc to update

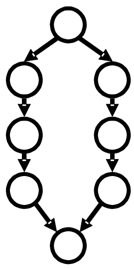
Message schedules

- The order in which messages are updated
- Asynchronous schedules: messages are updated sequentially, one arc at a time
- Synchronous schedules: all messages are updated in parallel.
- Flooding (asynchronous): messages are passed from each variable to each corresponding factor and back at each step
- The most widely used and generally best-performing method

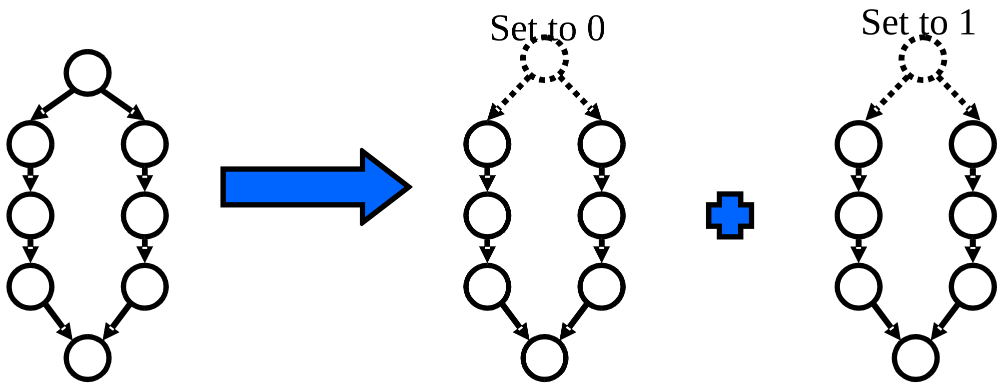
General Networks

- Networks that have a cycle in their undirected version
- Three possibilities
  - Conditioning
  - Clustering
  - Approximations



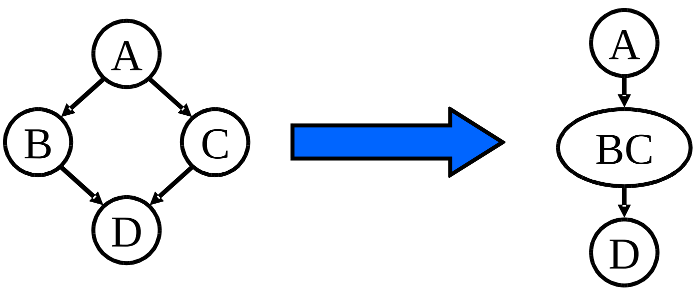


Conditioning



Clustering

- Group together variables so that the resulting network is a polytree and use belief propagation



- Problem: how to find a good clustering?

Join Trees

- Technique for clustering variables
- Steps:
  - Obtain an undirected version of the network
  - Perform a graph operation on it (triangulation)
  - Each clique is a compound variable
  - Add direction to the edges

Junction Tree

- The resulting inference algorithm [Lauritzen, Spiegelhalter 1988] is called
  - Junction tree algorithm (jt), or
  - Clique propagation

Approximate Methods

- Stochastic simulation:
  - Generate N samples from BN
  - Count: $N_{\mathbf{e}}$: samples that satisfy **e**, $N_{\mathbf{qe}}$ samples that satisfy **q,e**
  - $P(q|\mathbf{e})=N_{\mathbf{qe}}/N_{\mathbf{e}}$
- Loopy belief propagation:
  - bp in networks with cycles
  - Experiments have shown that it converges also in network with cycles, often to good quality solutions

Stochastic Simulation

- Let $X_1,\ldots,X_n$ be a topological sort of the variables
- For i=1 to n
  - Find parents, if any, of X_i. Call them $X_{p(i,1)}$, $X_{p(i,2)}$, … $X_{p(i,p(i))}$.
  - Recall the values that those parents were randomly given: $x_{p(i,1)}$, $x_{p(i,2)}$, …$x_{p(i,p(i))}$.
  - Look up in the cpt for:

    $P(X_i=x_i \mid X_{p(i,1)}=x_{p(i,1)},X_{p(i,2)}=x_{p(i,2)}\ldots X_{p(i,p(i))}=x_{p(i,p(i))})$
  - Randomly choose x_i according to this probability

Problems in Building BN

- Assessing conditional independence is not always easy for humans
- Usually done on the basis of causal information
- Assigning a number to each cpt entry is also difficult for humans

Problems in Building BN

- Often we do not have an expert but we are given a set of observations $D=\{\mathbf{u}^1,...\mathbf{u}^N\}$
- $\mathbf{u}^j$ is an assignment to all the variables $\mathbf{U}=\{X_1,...,X_n\}$
- How to infer the parameters and/or the structure from D?

Learning

- We want to find a BN over **U** such that the probability of the data P(D) is maximized
- P(D) is also called the **likelihood** of the data
- We assume that all the samples are **independent and identically distributed** (iid) so

$$P(D) = \prod_{i=1}^{N} P(\boldsymbol{u}^{\boldsymbol{i}})$$

- Often the natural log of P(D) **(log likelihood**) is considered

$$\log P(D) = \sum_{i=1}^{N} \log P(\boldsymbol{u}^{\boldsymbol{i}})$$

Learning BN

- Tasks
  - Computing the parameters given a fixed structure or
  - finding the structure and the parameters
- Properties of data:
  - complete data: in each data vectors $\mathbf{u}^j$, the values of all the variables are observed
  - incomplete data

Parameter Learning from Complete Data

- Parameters to be learned

$$\theta_{x_i|\boldsymbol{\pi}_i} = P(x_i|\boldsymbol{\pi}_i)$$

- for all x_i, $\boldsymbol{\pi}_i$, i=1,...,n
- The values of the parameters that maximize the likelihood can be computed in closed form

Maximum Likelihood Parameters

- Given by relative frequency
- If $N_{\mathbf{y}}$ be the number of vectors of D where **Y**=**y**.

$$\theta_{x_i|\boldsymbol{\pi}_i} = \frac{N_{x_i,\boldsymbol{\pi}_i}}{N_{\boldsymbol{\pi}_i}}$$

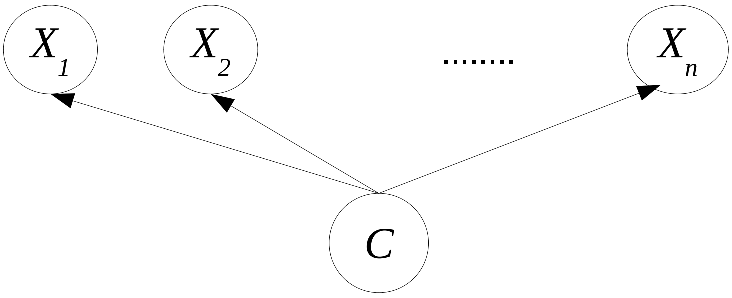
- Counting: for each i, for each value $\boldsymbol{\pi}_i$ we must collect

$$C_{\boldsymbol{\pi}_i} = \langle N_{x_i^1,\boldsymbol{\pi}_i}, \ldots, N_{x_i^{v_i},\boldsymbol{\pi}_i} \rangle$$

- where v_i is the number of values of X_i

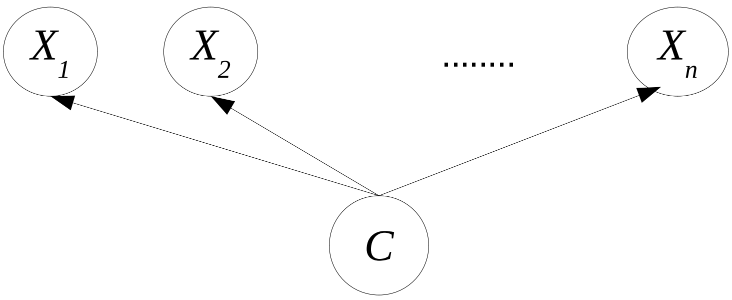
Naive Bayes Special Case

- We want to perform classification
- One variable C represents the class
- The variables **X** represent the attributes
- Model:



- X_i independent from X_j given C

Naive Bayes Special Case



- Conditional probability tables (case of Boolean variables):

	C=true	C=false
X_i=true	P(X_i=true|C=true)	P(X_i=true|C=false)
X_i=false	P(X_i=false|C=true)	P(X_i=false|C=false)

Example

No	Outlook	Temp	Humid	Windy	Class
D1	sunny	mild	normal	T	P
D2	sunny	hot	high	T	N
D3	sunny	hot	high	F	N
D4	sunny	mild	high	F	N
D5	sunny	cool	normal	F	P
D6	overcast	mild	high	T	P
D7	overcast	hot	high	F	P
D8	overcast	cool	normal	T	P
D9	overcast	hot	normal	F	P
D10	rain	mild	high	T	N
D11	rain	cool	normal	T	N
D12	rain	mild	normal	F	P
D13	rain	cool	normal	F	P
D14	rain	mild	high	F	P

	C=P	C=N
Humid=normal	6/9=0.66666	1/5=0.2
Humid=high	3/9=0.33333	4/5=0.8

Queries

- Computing the probability of a class given values for the attributes: $P(c|x_1,\ldots,x_n)$

$$P(c|x_{1,},\ldots,x_n)=\frac{P(c,x_{1,},\ldots,x_n)}{P(x_1\ldots x_n)}=\frac{P(x_{1,},\ldots,x_n|c)P(c)}{P(x_1\ldots x_n)}$$

- Since the attributes are independent given the class

$$P(c|x_{1,},\ldots,x_n)=\frac{P(x_1|c)\ldots P(x_n|c)P(c)}{P(x_1\ldots x_n)}$$

Example

- We want to classify <Outlook=sunny,Temp=cool,Humid=high,Windy=T>
- We have to compute
  P(Class=P|Outlook=sunny,Temp=cool,Humid=high,Windy=T)
- We compute only the parameters we need

  P(Class=P)=9/14=0.64
  P(Class=N)=5/14=0.36
  P(Outlook=sunny | Class=P)=2/9=0.222
  P(Outlook=sunny | Class=N)=3/5=0.6
  P(Temp=cool | Class=P)=3/9=0.333
  P(Temp=cool | Class=N)=1/5=0.2
  P(Humid=high | Class=P)=3/9=0.333
  P(Humid=high | Class=N)=4/5=0.8
  P(Windy=T | Class=P)=3/9=0.33
  P(Windy=T | Class=N)=3/5=0.6

Example

P(Class=P,Outlook=sunny,Temp=cool,Humid=high,Windy=T) = 0.0053

P(Class=N,Outlook=sunny,Temp=cool,Humid=high,Windy=T) = 0.0206

- We can compute P(Outlook=sunny,Temp=cool,Humid=high,Windy=T) by marginalization:

P(Outlook=sunny,Temp=cool,Humid=high,Windy=T)=

P(Class=P,Outlook=sunny,Temp=cool,Humid=high,Windy=T) +

P(Class=N,Outlook=sunny,Temp=cool,Humid=high,Windy=T)=

0.0053+0.0206=0.0259

- So

P(Class=P|Outlook=sunny,Temp=cool,Humid=high,Windy=T)=0.0053/0.0259=
0.205

P(Class=P|Outlook=sunny,Temp=cool,Humid=high,Windy=T)=0.0206/0.0259=
0.795

Structure Learning from Complete Data

- Perform a local search in the space of possible structures
- HGC algorithm [Heckerman, Geiger, Chickering 95]:
  - Start with a structure BestG' (possibly empty)
  - Repeat
    - BestG=BestG'
    - Let Ref={G|G is obtained from BestG' by adding, deleting or reversing an arc}
    - Let BestG'=argmax$_{G'}$ {score(G)|G $\in$ Ref}
  - while score(BestG')-score(BestG)>0

Structure Score

$$score(G) = \log P(D|G)$$

$$P(D|G) = \int \rho(D, \Theta|G)\, d\Theta$$
$$= \int P(D|\Theta, G)\, \rho(\Theta|G)\, d\Theta$$

- where

$$\rho(\Theta|G) = \prod_{i,\pi_i} \rho(\theta_{\pi_i})$$
$$\theta_{\pi_i} = \langle \theta_{x_i^1|\pi_i}, \ldots, \theta_{x_i^{v_i}|\pi_i} \rangle$$

- and $\rho(\theta_{\pi i})$ is the prior density of the vector $\theta_{\pi i}$

Prior Density of the Parameters

- A common choice for the form of the prior density is the **Dirichlet probability density**
- In this case $\rho(\theta_{\boldsymbol{\pi} i})$ is described by v_i parameters

$$C'_{\pi_i} = \langle N'_{x_i^1, \boldsymbol{\pi}_i}, \ldots, N'_{x_i^{v_i}, \boldsymbol{\pi}_i} \rangle$$

- Prior counts: it is as if we had previously observed some data on which the counts are $N'_{xi,\boldsymbol{\pi}i}$

Structure Score

- If the priors for the parameters are Dirichlet, then the score is called BD (for Bayesian Dirichlet) and

$$BD(G) = \sum_i BD_i(G)$$

- where $BD_i(G)$ depends only on C_i and C'_i, the counts for the family of X_i

$$C_i = \langle C_{\pi_i^1}, \ldots, C_{\pi_i^{r_i}} \rangle$$
$$C'_i = \langle C'_{\pi_i^1}, \ldots, C'_{\pi_i^{r_i}} \rangle$$

Structure Score

$$BD_i(G) = \sum_{\pi_i} \log \frac{\Gamma(N_{\pi_i})}{\Gamma(N_{\pi_i} + N'_{\pi_i})} + \sum_{x_i} \log \frac{\Gamma(N_{x_i,\pi_i} + N'_{x_i,\pi_i})}{\Gamma(N_{x_i,\pi_i})}$$

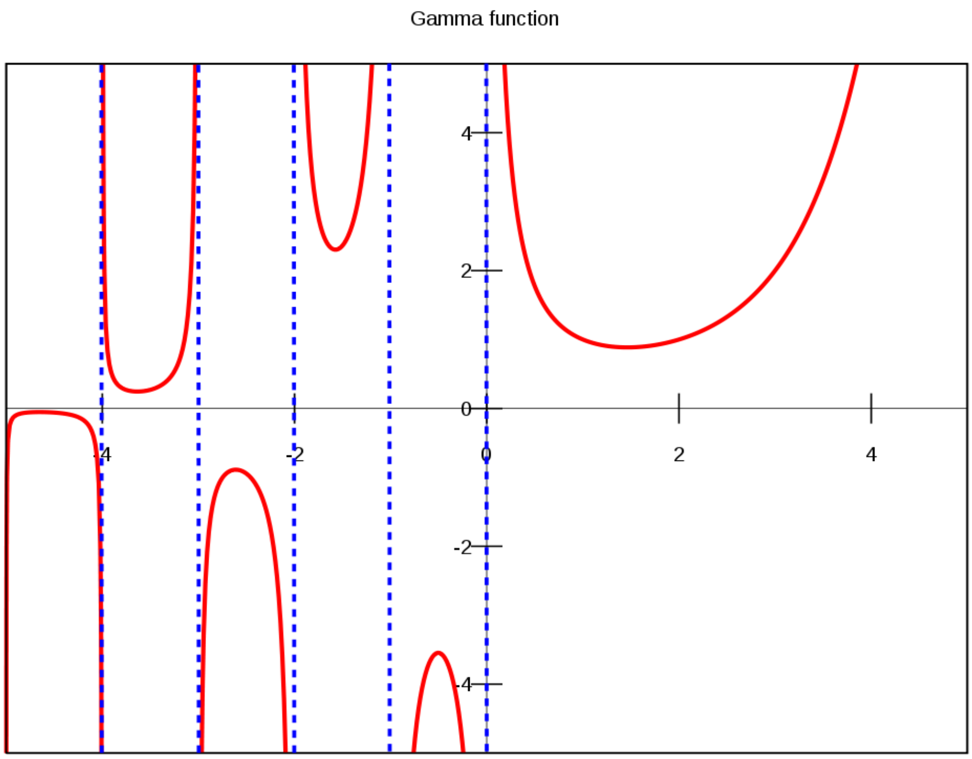
- Where Γ is the Gamma function, an extension of the factorial function with its argument shifted down by 1, to real and complex numbers. That is, if n is a positive integer:

$$\Gamma(n) = (n-1)!$$

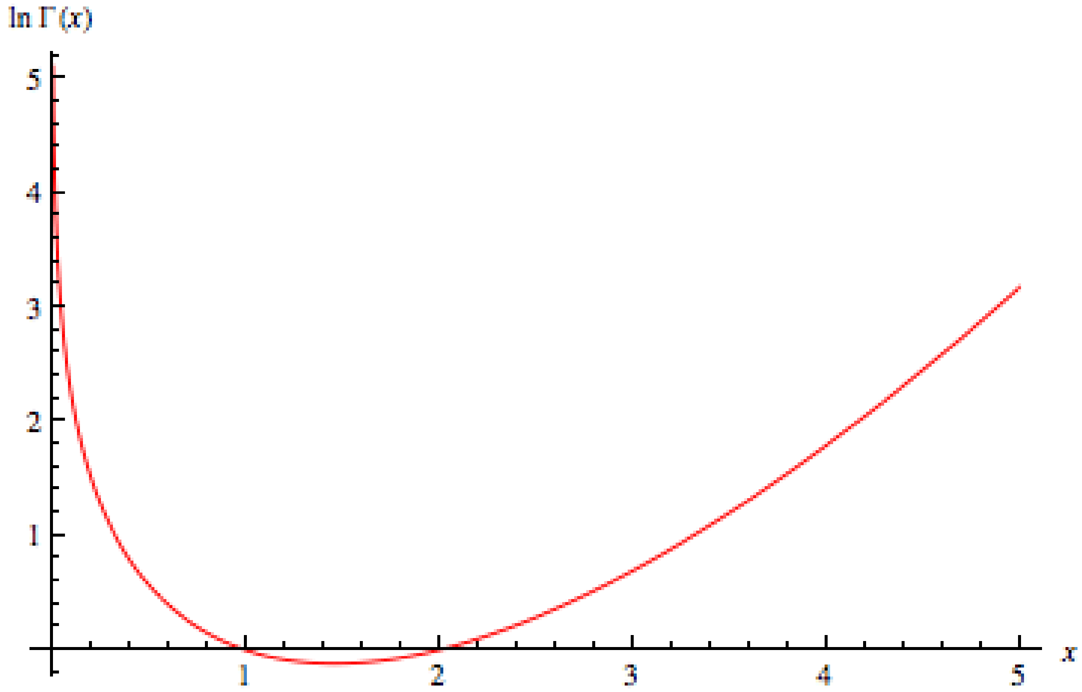
- otherwise

$$\Gamma(z) = \int_0^\infty t^{z-1} e^{-t}\, dt$$

Gamma Function



Log Gamma Function



Structure Score

- BD(G) is **decomposable**:
  - It can be computed independently for each family
- Each edge operation involves
  - 1 family (addition, deletion) or
  - 2 families (reversal)
- BD(G) can be quickly computed from BD(BestG') by changing only the score of the affected families

Parameter Learning from Incomplete Data

- The maximum likelihood parameters cannot be computed in closed form
- An iterative algorithm is necessary: the EM algorithm
- Finds a (possibly) local maximum of the likelihood

EM Algorithm

- Initialize the parameters at random Θ
- Repeat
  - Expectation step:
    - compute the probability P(y|e) of each value y of the missing attributes using (G,Θ) and inference
  - Compute Θ by maximum likelihood on D'
    - Relative frequency for each family
    - If a variable Y is unobserved in an example e that matches $x_i,\boldsymbol{\pi}_i$, instead of adding 1 to $N_{x_i,\boldsymbol{\pi}_i}$ we add P(y|e)

Structure Learning from Incomplete Data

- There is no decomposable score
- HGC would not be efficient
- Structural EM:
  - Start with a structure BestG' (possibly empty)
  - Repeat
    - BestG=BestG'
    - Compute the parameters of BestG with EM
    - Optimize a lower bound of the likelihood of the observed data
    - Let BestG' the optimum
  - Until no improvement

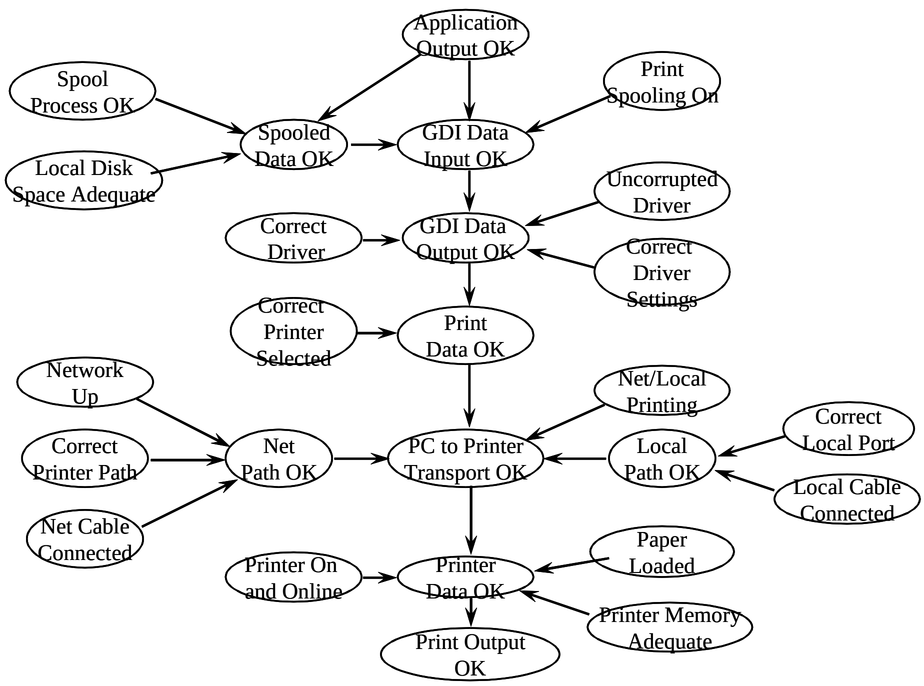
Applications of BN

- Monitoring of emergency care patients
- Model of barley crops yield
- Diagnosis of carpal tunnel syndrome
- Insulin dose adjustment (DBN) in diabetes .
- Predicting hails in northern Colorado
- Evaluating insurance applications

Applications of BN

- Deciding on the amount of fungicides to be used against attack of mildew in wheat
- Assisting experts of electromyography
- Pedigree of breeding pigs
- Modeling the biological processes of a water purification plant
- Printer troubleshooting (Microsoft Windows)

Printer Troubleshooting (Windows 95)



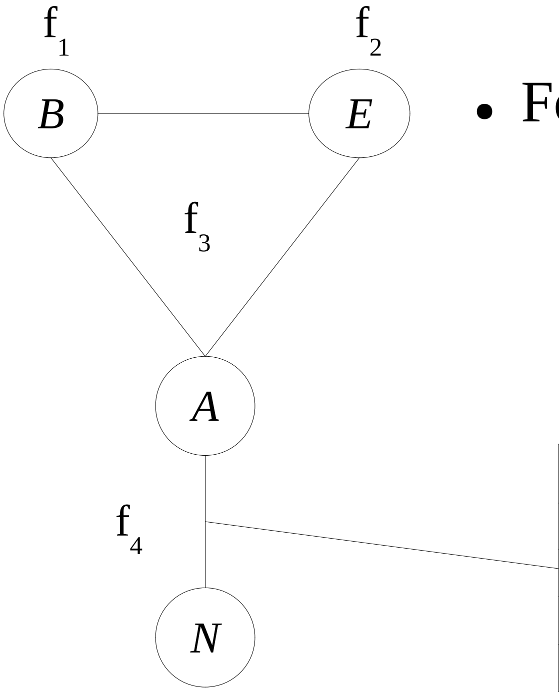
Applications

- Office Assistant in MS Office ("smiley face")
  - Bayesian network based free-text help facility
  - help based on past experience (keyboard/mouse use) and task user is doing currently

Markov Networks (MN)

- Approach alternative to BN
- Also called Random Fields, undirected graphical models
- Undirected graph
- Conditional independence represented by graph separation
- Probability distribution as the product of a set of **potentials** or **factors** (functions of a subset of variables) divided by a normalization constant
- Potentials over cliques

Example



- Four potentials f_1, f_2, f_3 and f_4

N	*A*	f_4
no	no	0,9
no	yes	0,05
yes	no	0,1
yes	yes	0,05

Markok Networks

- Probability

$$P(\boldsymbol{u}) = \frac{\prod_c f_c(\boldsymbol{x_c})}{Z}$$

$$Z = \sum_{\boldsymbol{u}} \prod_c f_c(\boldsymbol{x_c})$$

- Z is called **partition function**, ensures that the probabilities sum to 1

Loglinear Models

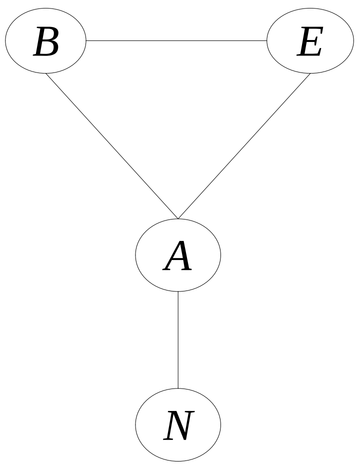
- If all the potentials are >0, they can be represented as exponential functions, i.e., f_4 can be represented as $f_4=\exp(w_4\, F_4)$
- where F_4 is any real function of f_4 arguments and w_4 is a real weight. Then

$$P(\boldsymbol{u}) = \frac{\exp \sum_c w_c F_c(\boldsymbol{x_c})}{Z}$$

$$Z = \sum_{\boldsymbol{u}} \exp \sum_c w_c F_c(\boldsymbol{x_c})$$

How to tell independence

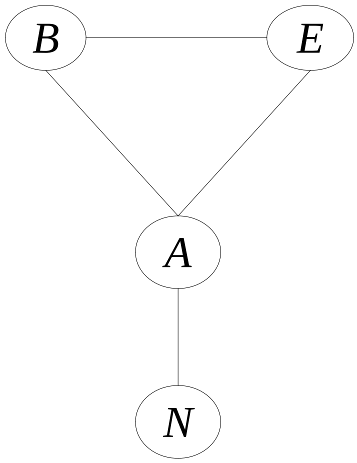
- Definition: **X** and **Y** are **independent** given a set of variables **Z** (I<**X**,**Z**,**Y**>) iff every path from **X** to **Y** passes through a variable of **Z**



- I<B,{},N>?
- I<B, A, N>?
- I<B, E, N>?
- I<{B,E}, A, N>?

How to tell independence

- Definition: **X** and **Y** are **independent** given a set of variables **Z** (I<**X**,**Z**,**Y**>) iff every path from **X** to **Y** passes through a variable of **Z**



- I<B,{},N> No
- I<B, A, N> Yes
- I<B, E, N> No
- I<{B,E}, A, N>Yes

Markov Network

- Inference:
  - Algorithms similar to those for BN (bp, cp, ve, ss...)
  - Same complexity
- MN can represent some independencies that BN can not represent and vice versa
- Advantage: we do not have to avoid cycles
- Disadvantage: MN parameters are more difficult to interpret

BN Software

- List of BN software
  http://www.cs.ubc.ca/~murphyk/Software/bnsoft.html\
- BNT: inference and learning, Matlab, open source
- MSBNx: inference, by Microsoft, free closed source
- OpenBayes: inference and learning, Python, open source
- BNJ: inference and learning, Java, open source
- Weka: learning, Java, open source

Resources

- Daphne Koller, Nir Friedman, Probabilistic graphical models: principles and techniques, MIT Press: 2009, ISBN 978-0-262-01319-2
- Probabilistic Reasoning in Intelligent Systems by Judea Pearl. Morgan Kaufmann: 1998.
- Probabilistic Reasoning in Expert Systems by Richard Neapolitan. Wiley: 1990.
- List of BN Models and Datasets
  http://www.cs.huji.ac.il/labs/compbio/Repository/

References

- [Pearl 85] Pearl, J., "Bayesian Networks: a Model of Self-Activated Memory for Evidential Reasoning," UCLA CS Technical Report 850021, Proceedings, Cognitive Science Society, UC Irvine, 329-334, August 15-17, 1985.
- [Pearl 00] Pearl, J., Causality: Models, Reasoning, and Inference, Cambridge University Press, 2000
- [Suermondt et al. 90] Henri Jacques Suermondt, Gregory F. Cooper, David Heckerman, “A combination of cutset conditioning with clique-tree propagation in the Pathfinder system”, UAI '90.

References

- [Pearl 88] Judea Pearl, Probabilistic Reasoning in Intelligent Systems, Morgan Kaufmann: 1998.
- [Lauritzen, Spiegelhalter 1988]
- [Heckerman, Geiger, Chickering 95] D. Heckerman, D. Geiger, D. M. Chickering: “Learning Bayesian Networks: The Combination of Knowledge and Statistical Data”, Machine Learning, 20(3), 1995