BASICS OF BNs

Basics of Bayesian networks

- Reasoning under uncertainty
- Conditional independence
- D-separation
- Bayesian networks: formal definition
- Building BNs

Reasoning under uncertainty

Advantages of BNs

- Explicit representation of the uncertain knowledge
  - Graphical, intuitive, closer to a world repres.
- Deal with uncertainty for reasoning *and* decision-making
- Founded on probability theory, provide a clear semantics and a sound theoretical foundation
- Both data and experts can be used to construct the model
- Current and huge development
- Support the expert; do not try to replace him

Reasoning under uncertainty

Modularity

- The joint probability distribution (JPD) (global model) is specified via marginal and conditional distributions (local models), taking into account conditional independence relationships among variables

- This modularity:
  - Provides an easy maintenance
  - Reduces the number of parameters needed for the JPD
    - Estimation/elicitation is easier
    - Reduction of the storing needs
    - Efficient reasoning (inference)

Conditional independence

The joint probability distribution (JPD)

- Dealing with a JPD
  - m_1 diseases $D_1,\dots,D_{m1}$
  - m_2 symptoms $S_1,\dots,S_{m2}$
  - Represent $P(D_1,\dots,D_{m1},S_1,\dots,S_{m2})$, with $2^{m1+m2}-1$ parameters
  - E.g.: $m_1=30$, $m_2=10$, need of $2^{40}-1\approx 10^{12}$
- That's *complete* dependence: intractable in practice

Conditional independence

Independence

- With mutual independence, only specify $P(D_1),\ldots,P(S_{m2})$
  - m1+m2 parameters (linear) instead of 2^{m1+m2} (exponential)
- Unfortunately, it rarely holds in most domains
- Fortunately, there are some conditional independencies. Exploit them (representation and inference)

Conditional independence

Conditional independence

- Independence (marginal) $P(x|y) = P(x)$ $\Leftrightarrow$ $P(x,y) = P(x)P(y)$

  sets of vars

- Conditional independence of X and Y given Z

  3 disjoint sets of variables

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

  for all possible values x,y,z

  $\Leftrightarrow$

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

  Intuitively, whenever Z=z, the information Y=y does not influence on the probability of x

  Notation: $I_P(X,Y|Z)$

Conditional independence

Example A (Alarm [Pearl'88])

- R burglary and T earthquake are not uncommon in LA. Alarm A may be caused by R or T

$$I_P(R,T|\emptyset)$$

➡ R and T are independent without any knowledge (burglary is not a sign of earthquake and vice versa)

➡ R and T are dependent if we know that the alarm went off (to know R will explain the evidence obtained about A and then it will confirm or discard T as the cause of A. And vice versa)

$$\neg I_P(R,T|A)$$

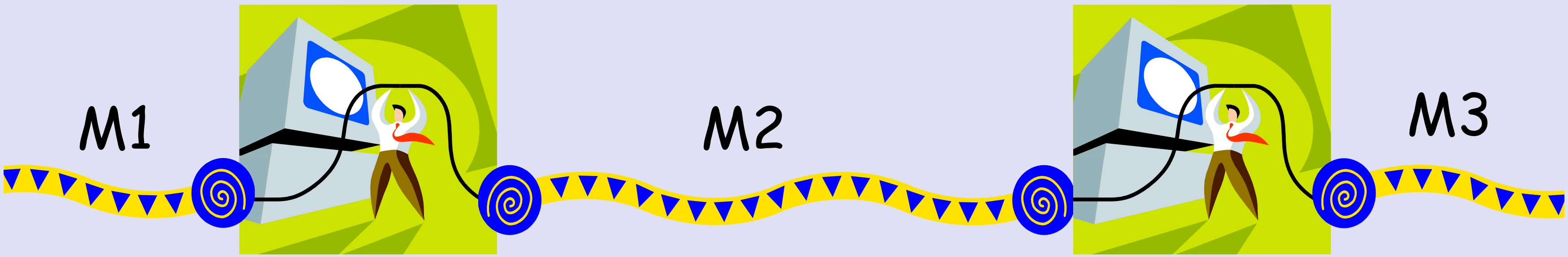
With further knowledge, we go from independence to dependence

Conditional independence

Example B

- Send a message M1 through a transmitter. It is received as M2 and it is then sent through other transmitter. It is received finally as M3.

  Transmitters have noise that modifies messages



→ M1 & M3 are dependent without any knowledge $\neg I_P(M1, M3|\emptyset)$

→ M1 y M3 are independent given M2 $I_P(M1, M3|M2)$

With further knowledge, we go from dependence to independence

Further factorizing the JPD

Chain rule $P(X_1, X_2, \ldots, X_n) = \prod_{i=1}^{n} P(X_i|X_1, \ldots, X_{i-1})$ and factorization via c.i.

- About $P(X_i|X_1, \ldots, X_{i-1})$:
  - Domain knowledge usually allows one to identify a subset $pa(X_i) \subseteq \{X_1, \ldots, X_{i-1}\}$ such that
    - Given $pa(X_i)$, X_i is independent of all variables in $\{X_1, \ldots, X_{i-1}\} \setminus pa(X_i)$, i.e.

$$P(X_i|X_1, \ldots, X_{i-1}) = P(X_i|pa(X_i))$$

$$\Rightarrow \quad P(X_1, X_2, \ldots, X_n) = \prod_{i=1}^{n} P(X_i|pa(X_i))$$

Joint distribution factorized

- The number of parameters might be substantially reduced

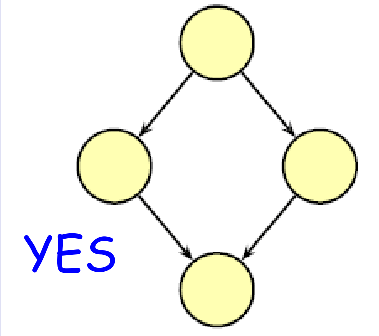
BNs

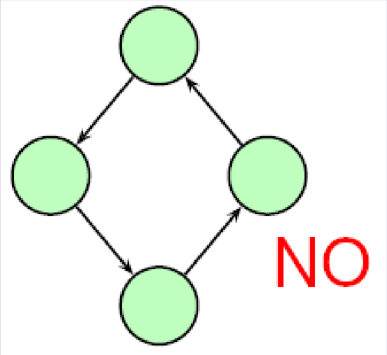
Informal definition: 2 components in a BN

- Qualitative part: a directed acyclic graph (DAG)

  Nodes = variables

  Arcs = direct dependence relations (otherwise it indicates absence of direct dependence; there may be indirect dependences and independences)





Not necessarily causality

- Quantitative part: a set of conditional probabilities that determine a unique JPD

BNs: nodes

Target node

Parents

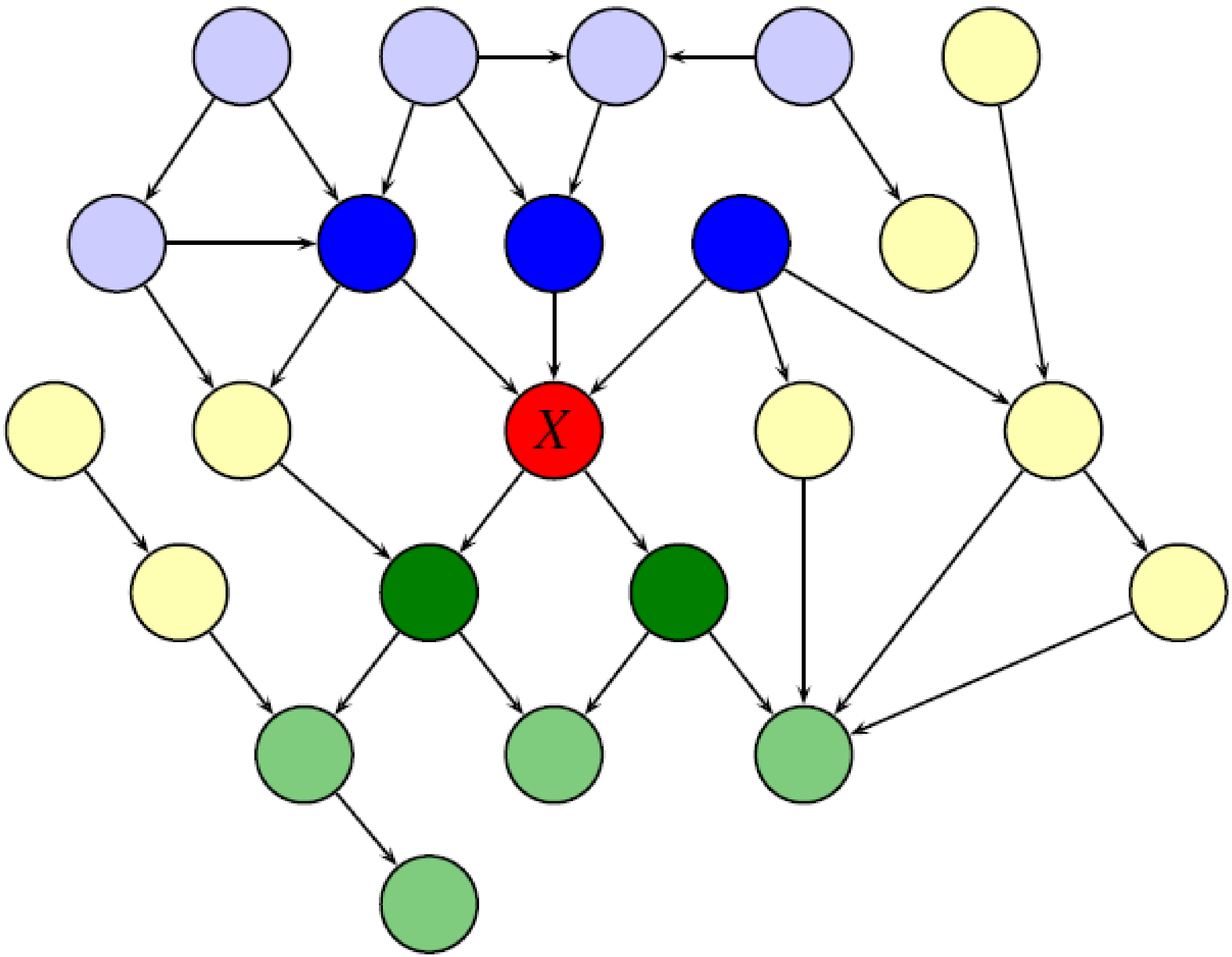
Ancestors

Children

Descendants

Rest

Family



BNs: arcs (types of independence)

Independences in a BN

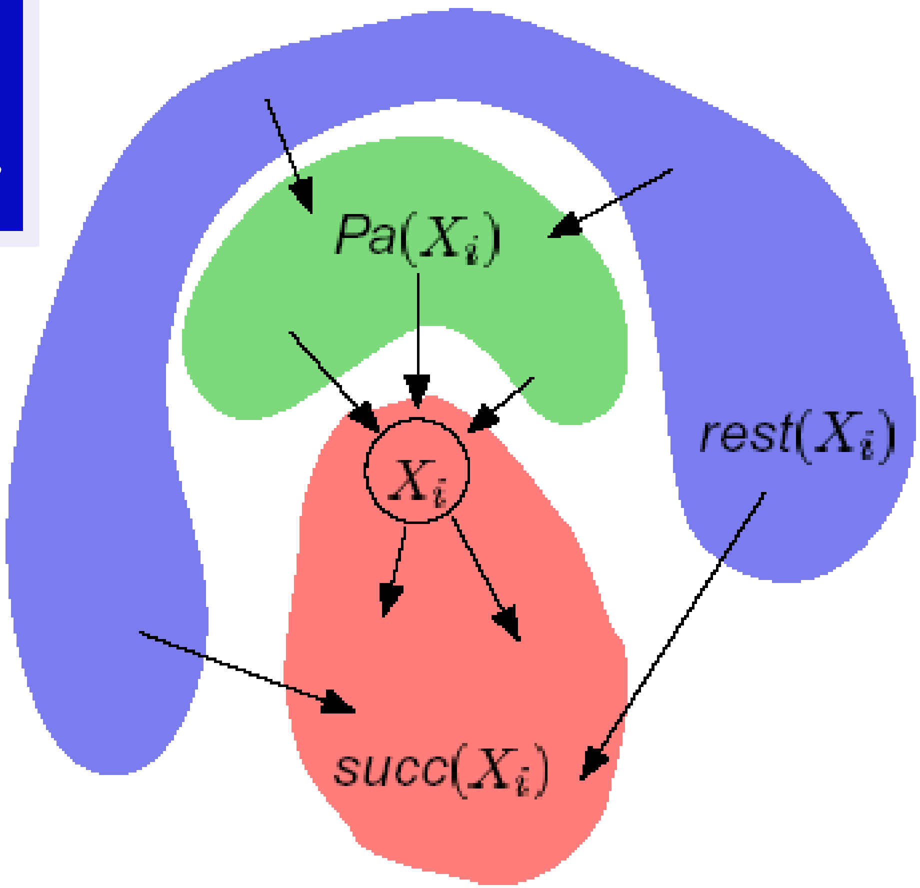
- A BN represents a set of independences
- Distinguish:
  - Basic independences: we should take care of verifying them when constructing the net
  - Derived independences: from the previous independences, by using the properties of the independence relations

Check them by means of the u-separation (or d-separation) criterion

Basic independences

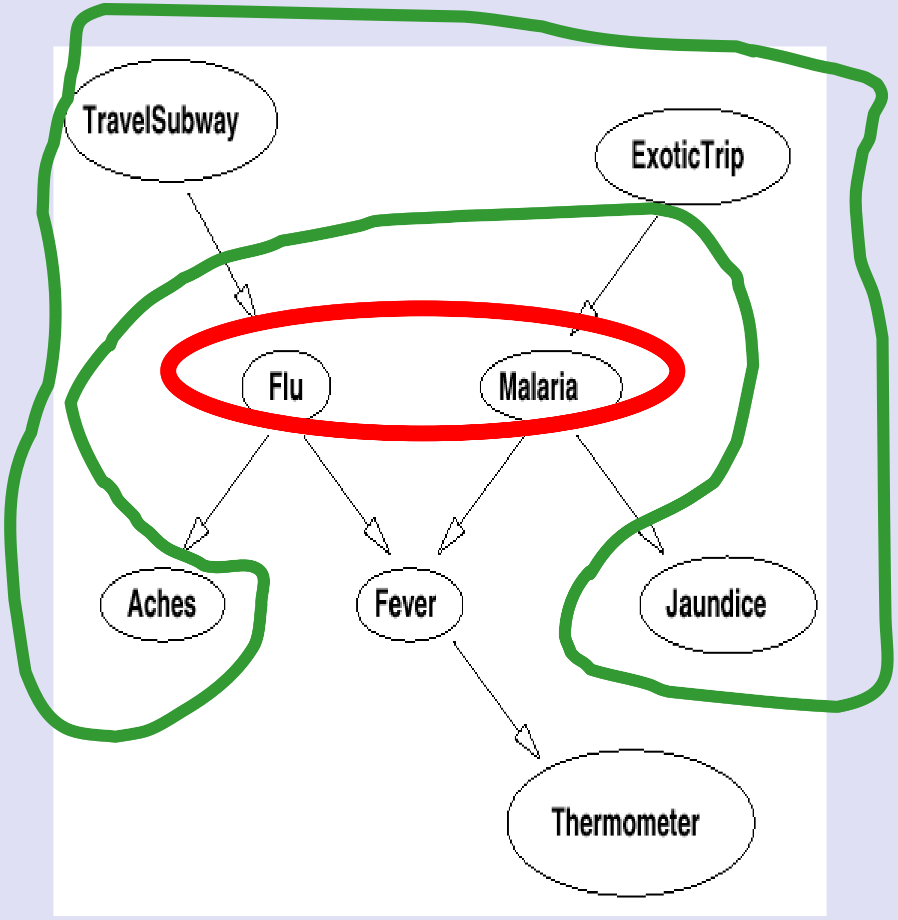
Basic independence: Markov condition or (local) Markov property

X_i c.i. of its non-descendants, given its parents $Pa(X_i)$



Basic independences

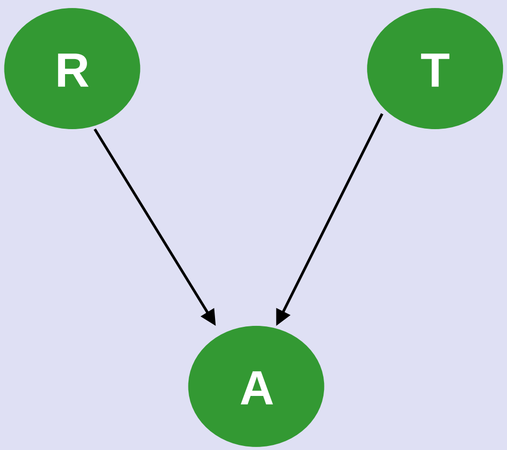
Example



Fever is c.i. of **Jaundice** given **Malaria** and **Flu**

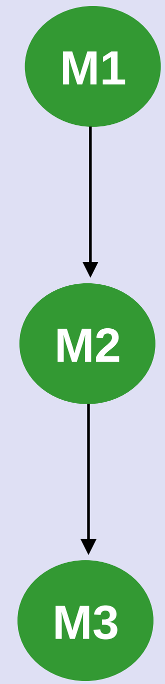
Basic independences

Examples A and B



$I_P(R, T|\emptyset)$

Non-descend.



$I_P(M1, M3|M2)$

Markov condition and JPD factorization

Factorizing the JPD

- Use the chain rule and the Markov condition

$$P(X_1, X_2, \ldots, X_n) = \prod_{i=1}^{n} P(X_i | X_1, \ldots, X_{i-1})$$

$$I_P(X, \text{non-desc} | Pa(X))$$

- Let $X_1,\ldots,X_n$ be an ancestral ordering (parents appear before their children in the sequence). It always exists (DAG)

- Using that ordering in the chain rule, in $\{X_1, \ldots, X_{i-1}\}$ there are non-descendants of X_i, and we have

$$P(X_i | X_1, \ldots, X_{i-1}) = P(X_i | pa(X_i))$$

Markov condition and JPD factorization

Factorizing the JPD

- Therefore, we can recover the JPD by using the following factorization:

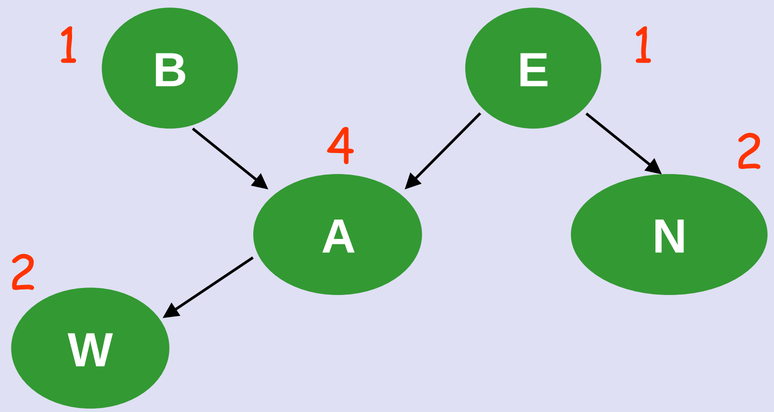
$$P(X_1, X_2, \ldots, X_n) = \prod_{i=1}^{n} P(X_i | pa(X_i))$$

MODEL CONSTRUCTION EASIER:

- Only store local distributions at each node
- Fewer parameters to assign and more naturally
- Inference easier

Example of savings

With all binary variables:



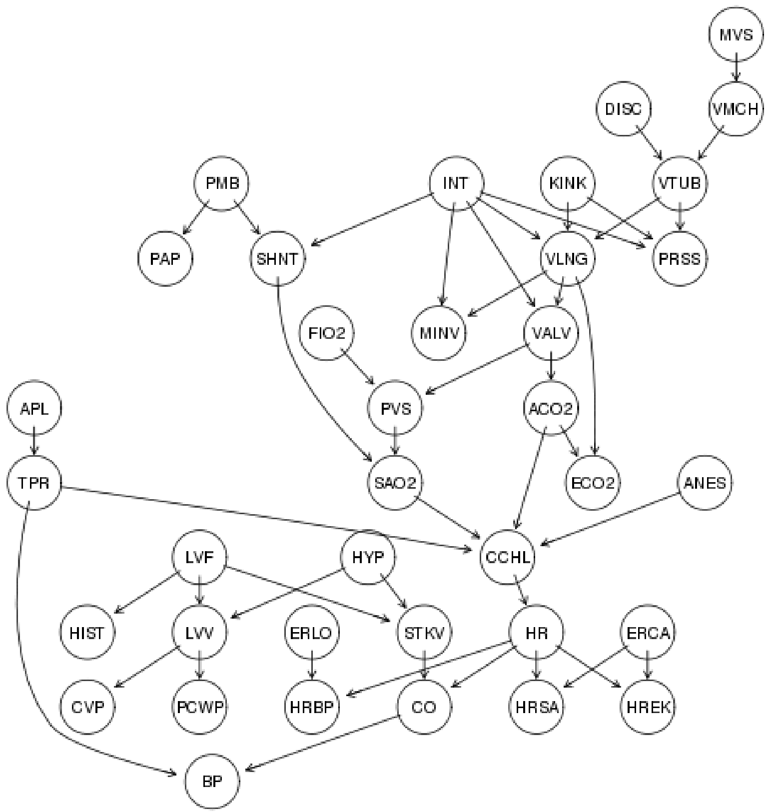
- $32=2^5$ probabilities for the JPD
- 20 with the factorization in the BN (10 in fact):

$$P(B,E,A,N,W) = P(W|A)P(A|B,E)P(N|E)P(B)P(E)$$

Example of savings

BN *Alarm* for monitoring ICU patients

- 2^{37} probabilities for the JPD vs. 509 in BN



Independencies derived from u-separation

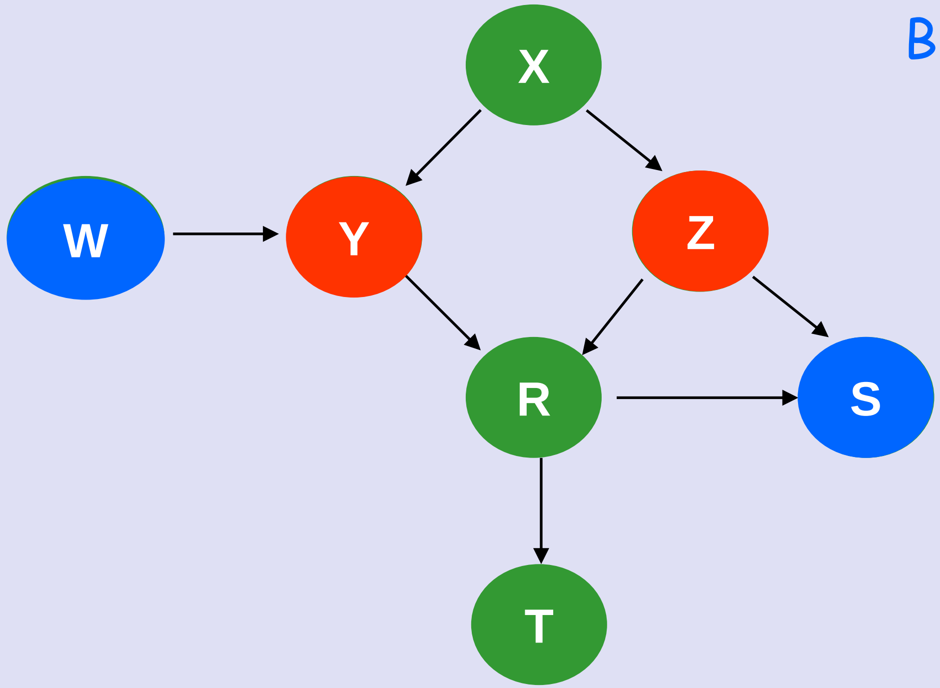
u-separation

- Obtain the minimum graph containing **X**,**Y**,**Z** and their ancestors (ancestral graph)
- The subgraph obtained is moralized (add a link between parents with children in common) and remove direction of arcs
- **Z** u-separates **X** and **Y** whenever **Z** is in all paths between **X** and **Y**

$$\mathrm{X} \perp \mathrm{Y} | \mathrm{Z}$$

Independencies derived from u-separation

u-separation

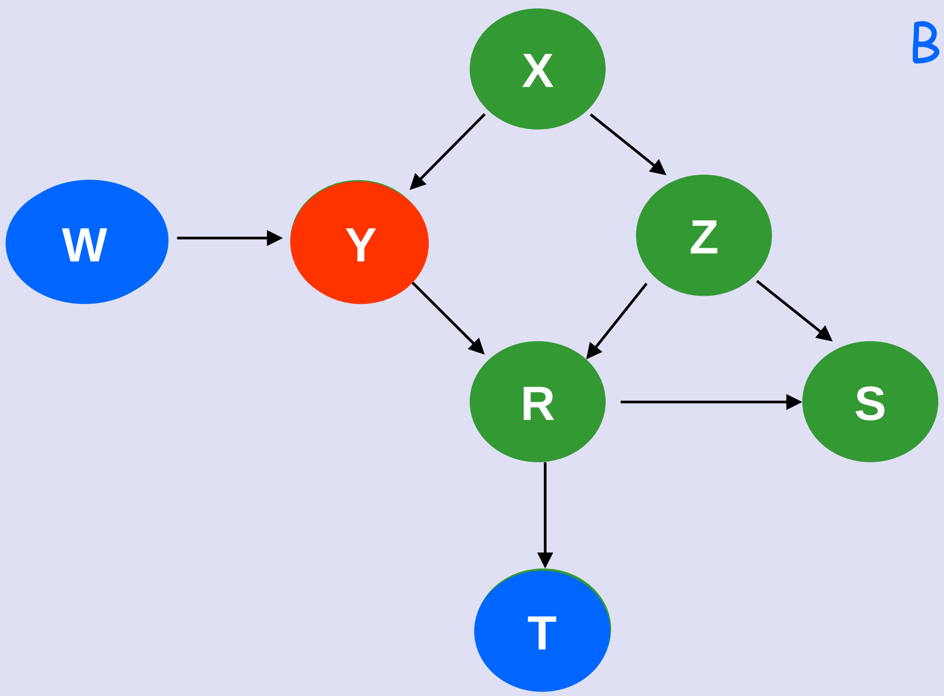


Blue d-separated by red?

$W \perp S \mid \{Y,Z\}$?

Independencies derived from u-separation

u-separation



Blue d-separated by red?

$W \perp T \mid Y$?

Joining the two parts

Separation Theorem [Verma and Pearl'90, Neapolitan'90]

- Let P be a prob. distribution of the variables in V and G=(V,E) a DAG.
  (G,P) holds the Markov condition iff

$$\mathbf{X} \perp_{\mathbf{G}} \mathbf{Y}|\mathbf{Z} \Longrightarrow I_P(\mathbf{X},\mathbf{Y}|\mathbf{Z}) \quad \forall \mathbf{X},\mathbf{Y},\mathbf{Z} \subseteq V$$

u-separation defined by G

c.i. defined by P

disjoint

- Graph G represents all dependences of P

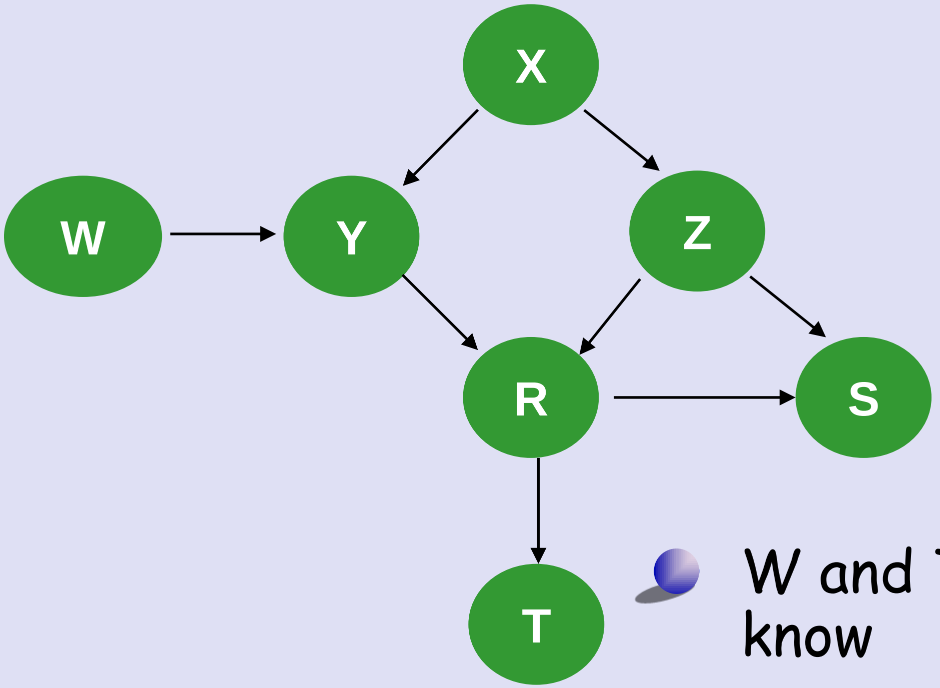
$$\neg I_P(\mathbf{X},\mathbf{Y}|\mathbf{Z}) \Rightarrow \neg(\mathbf{X} \perp_{\mathbf{G}} \mathbf{Y}|\mathbf{Z})$$

- Some independences of P may be not identified by u-separation in G

Joining the two parts

In the example...

- W and S are c.i. given {Y,Z}



- W and T given Y??: we don't know

Joining the two parts

Correspondence graph-model

- A DAG may be viewed as maps of independences of P:

  - D-Map of P (dependences): independent variables are d-separated in the graph
  - I-Map of P (independences): u-separated variables in the graph are independent
  - P-Map of P (perfect): I-map and D-map

- P-Map is not always possible (P is faithful to the DAG)

Definition of BN

Formal definition

- Let P be a JPD over $V=\{X_1,\ldots,X_n\}$.

A **BN** is a tuple (G,P), where G=(V,E) is a DAG such that:

- Each node of G represents a variable of V
- The Markov condition is held
- Each node has associated a local prob. distribution such that $P(X_i|pa(X_i))$,

(taking an ancestral ordering)

quantitative part

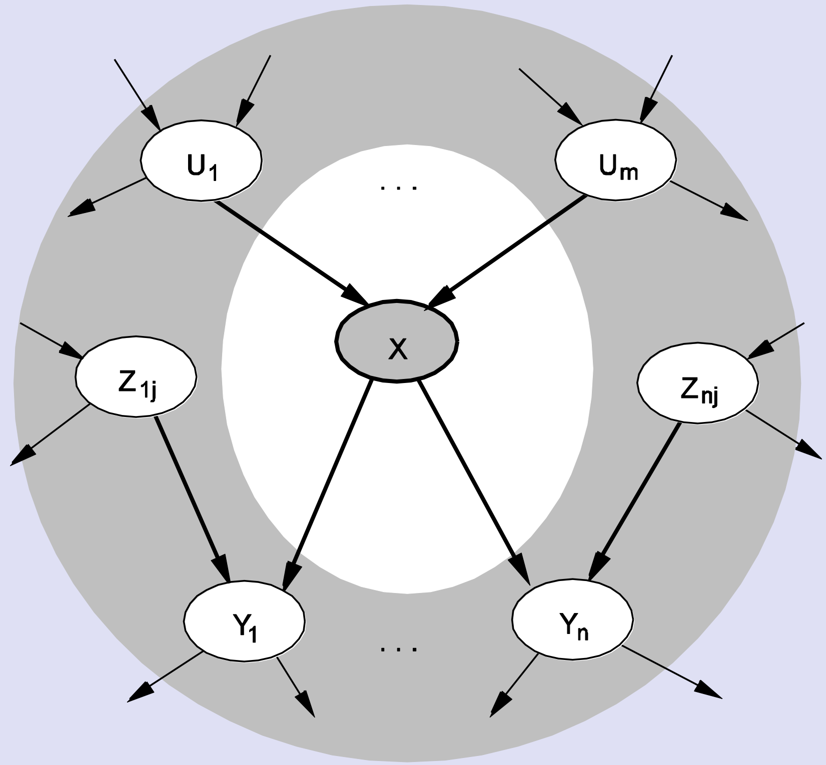
$$P(X_1,\ldots,X_n)=\prod_{i=1}^{n}P(X_i|pa(X_i))$$

- u-separated variables in the graph are independent (G is a minimal I-map of P)

Definition of BN

The global Markov property

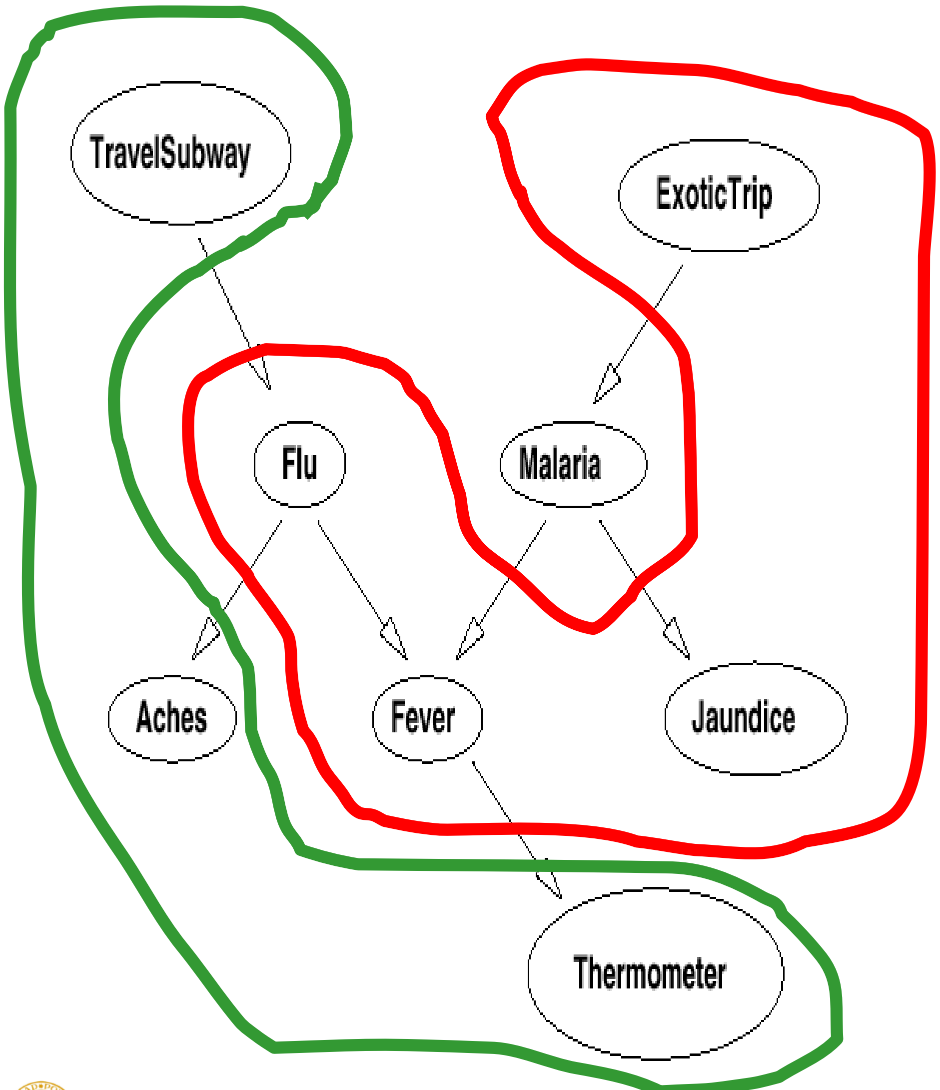
Set of nodes that makes X c.i. of the rest of the network:



- A node is c.i. of all other nodes in the BN, **given its** parents, children and children's parents (spouses)

  -its Markov blanket-

Definition of BN

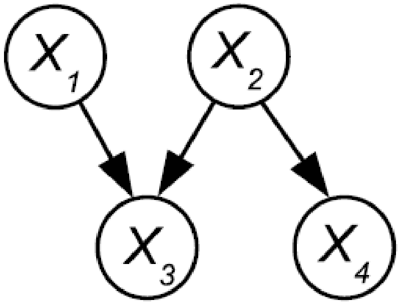


Malaria is c.i. of Aches, TravelSubway and Thermometer given ExoticTrip, Jaundice, Fever and Flu

What about continuous variables?

GAUSSIAN BNs

- All variables are continuous
- All conditional distributions as (linear) Gaussians



$$
\begin{aligned}
f_1(x_1) &\sim \mathcal{N}(\mu_1, v_1) \\
f_2(x_2) &\sim \mathcal{N}(\mu_2, v_2) \\
f_3(x_3|x_1, x_2) &\sim \mathcal{N}(\mu_3 + \beta_{31}(x_1 - \mu_1) + \beta_{32}(x_2 - \mu_2), v_3) \\
f_4(x_4|x_2) &\sim \mathcal{N}(\mu_4 + \beta_{42}(x_2 - \mu_2), v_4)
\end{aligned}
$$

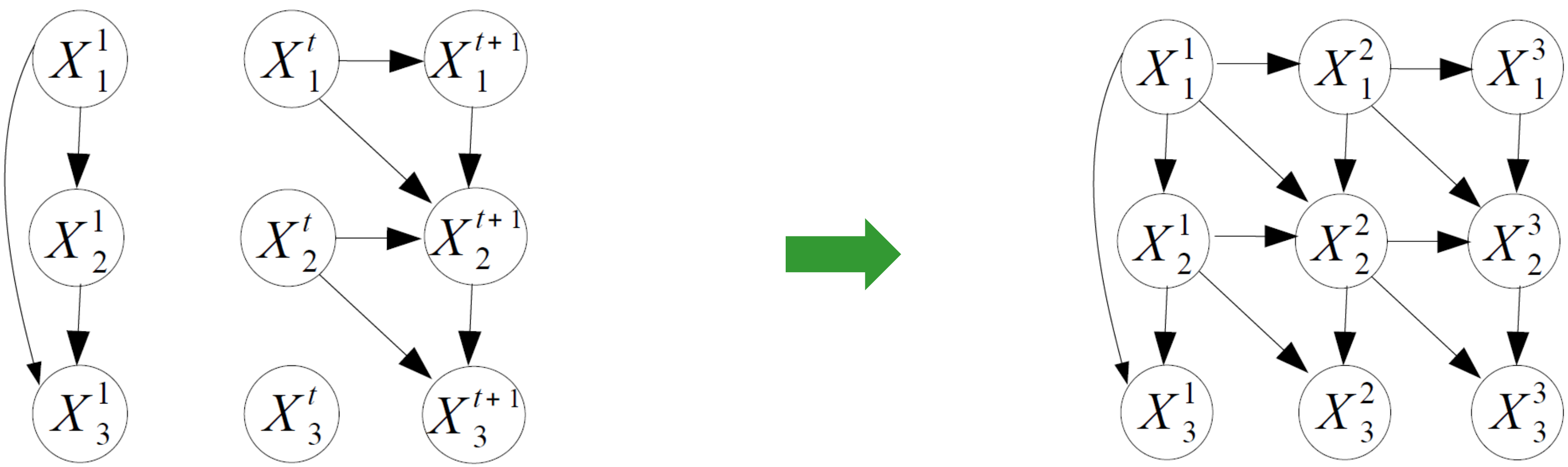
- Define the JPD $\mathcal{N}(\boldsymbol{x}|\boldsymbol{\mu}, \boldsymbol{\Sigma})$
- (Inference in closed form)

- Other: kernel, MTE, MoP, MoTBF...

What about dynamic systems?

- DYNAMIC BNs: Time slices (with identical BNs)
- Transition arcs toward future

$$\mathbf{X}^t = (X_1^t, \ldots, X_n^t),\ t = 1, \ldots, T$$



Prior BN Transition BN Unrolled

$$\boxed{P(\mathbf{X}^1)\prod_{t=2}^{T} P(\mathbf{X}^t|\mathbf{X}^{t-1}) = P(\mathbf{X}^1, \ldots, \mathbf{X}^T)}$$

$$P(\mathbf{X}^t|\mathbf{X}^{t-1}, \ldots, \mathbf{X}^1) = P(\mathbf{X}^t|\mathbf{X}^{t-1})$$

Stationarity and first-order Markov assumptions

Building a BN

Expert /from data /both

- Manual with the aid of an expert in the domain

Causal mechanisms $\xrightarrow{\text{modelisation}}$ Causal graph $\xrightarrow{\text{probabilities}}$ Bayesian net

  - Build it in the causal direction: BNs simpler and efficient

- Learning from a database

Database $\xrightarrow{\text{algorithm}}$ Bayesian net

- A combination (experts $\rightarrow$ structure; database $\rightarrow$ probabilities)

Resources

- BN repositories:
  http://www.bnlearn.com/bnrepository/
  http://www.cs.huji.ac.il/~galel/Repository/

- Much information:
  http://www.cs.ualberta.ca/~greiner/bn.html#applic

- Coursera (D. Koller @ Stanford): "Probabilistic graphical models": https://www.coursera.org/specializations/probabilistic-graphical-models

INFERENCE in BNS

Inference in Bayesian networks

Types of queries

Exact inference:

- Brute-force computation
- Variable elimination algorithm
- Message passing algorithm

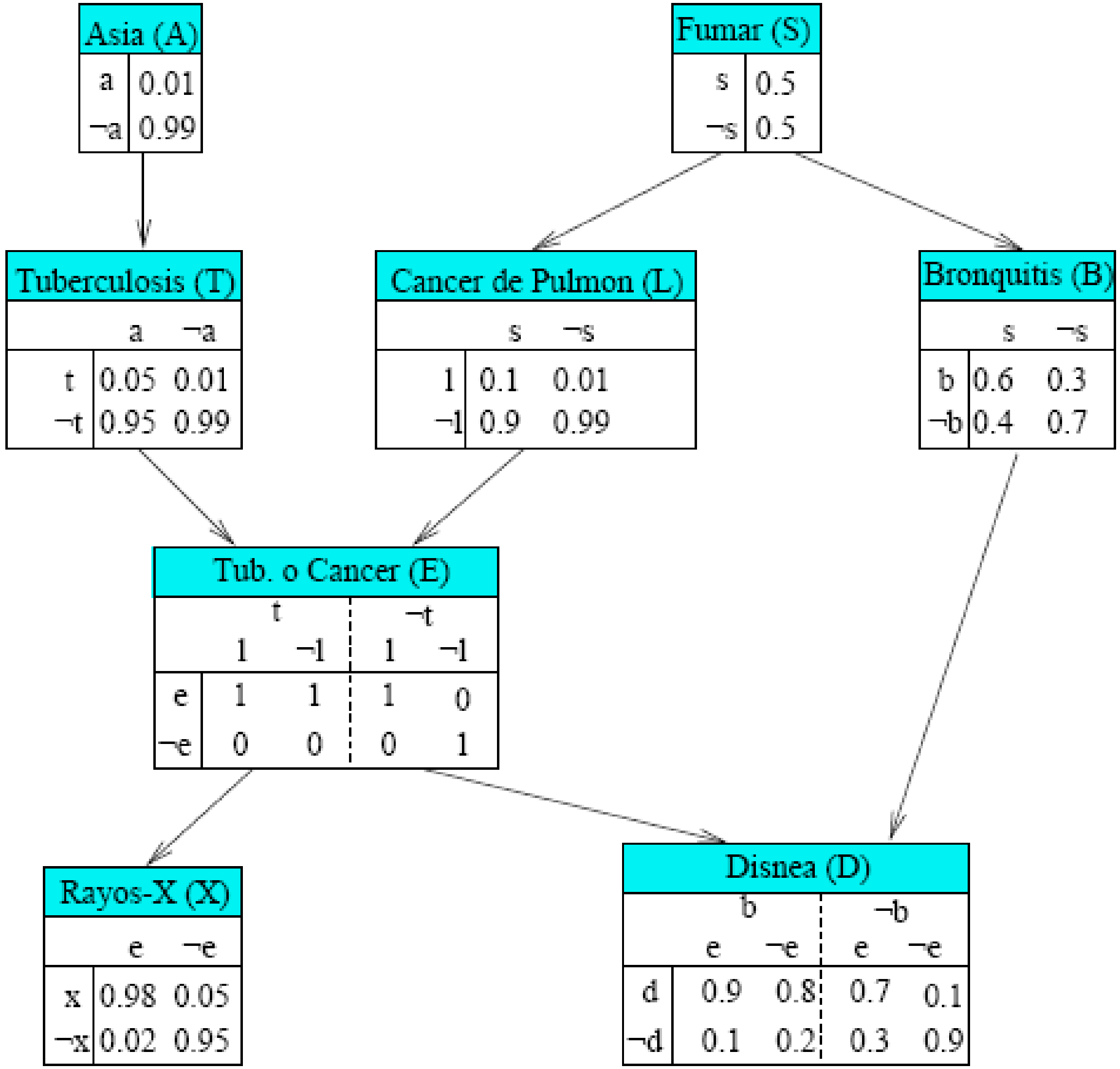
Approximate inference:

- Logic sampling
- Likelihood weighting

Example: Asia BN [Lauritzen & Spiegelhalter'88]

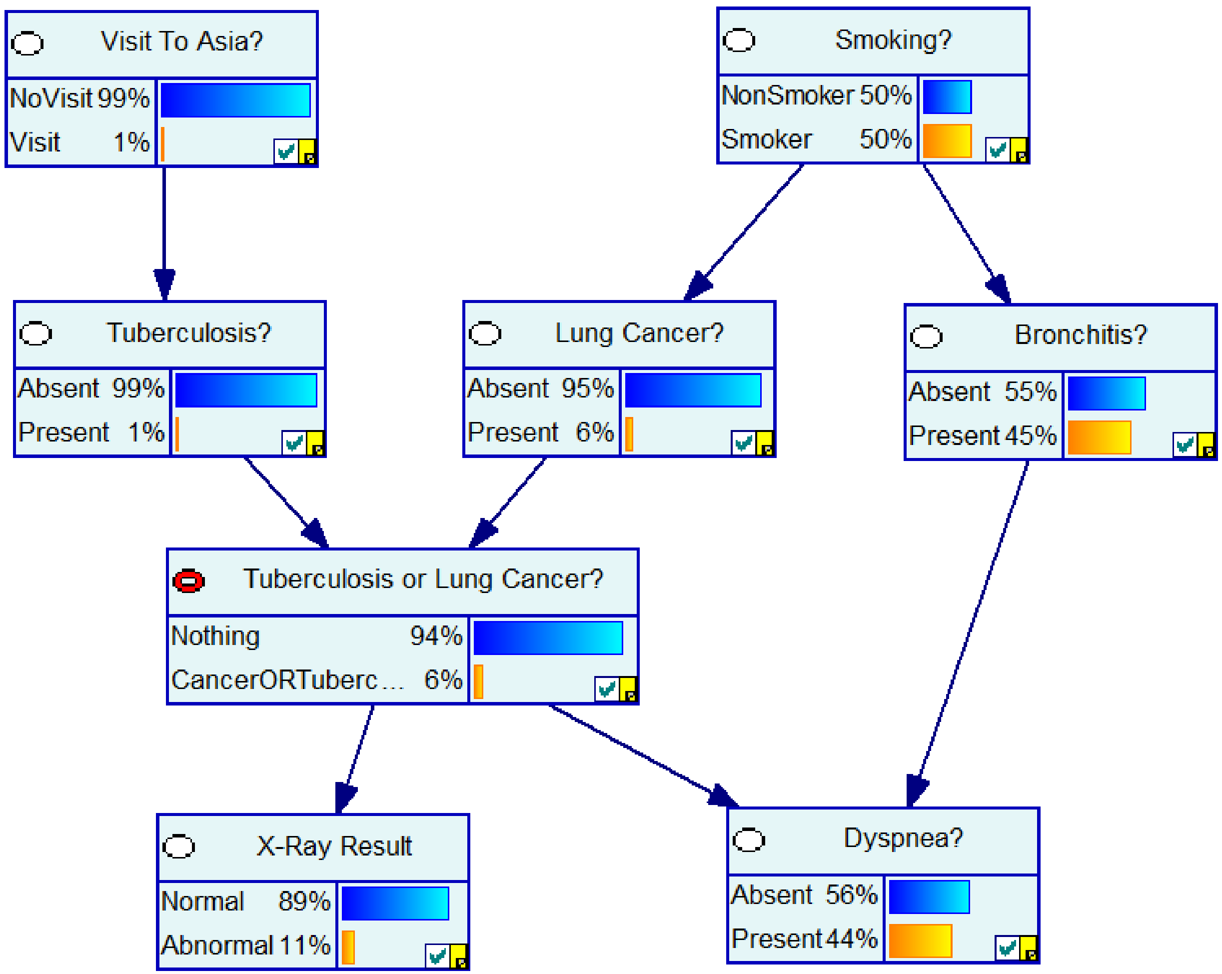
- Physician wants to diagnose her patients w.r.t. 3 diseases
  - Tuberculosis
  - Lung cancer
  - Bronchitis
- Causes or risk factors:
  - Recent Visit to Asia increases the chances of Tuberculosis
  - Smoking is a risk factor for both Lung cancer and Bronchitis
- Symptoms:
  - Dyspnea (shortness-of-breath) may be due to Tuberculosis, Lung cancer, Bronchitis, none of them, or more than one of them
  - Chest X-Ray. Neither symptom discriminates between Lung cancer and Tuberculosis

Example: Asia BN [Lauritzen & Spiegelhalter'88]

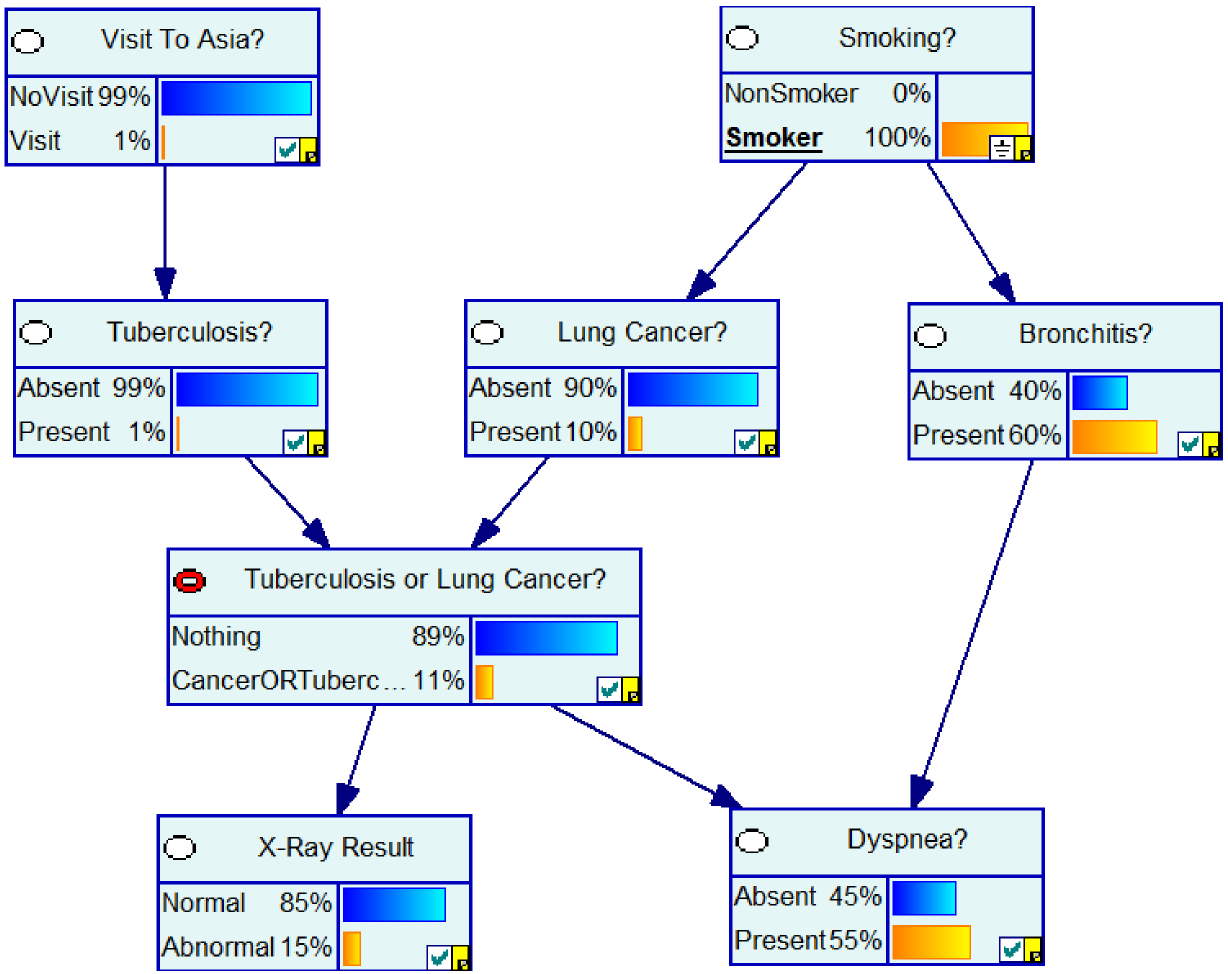


Example: Asia BN [Lauritzen & Spiegelhalter'88]

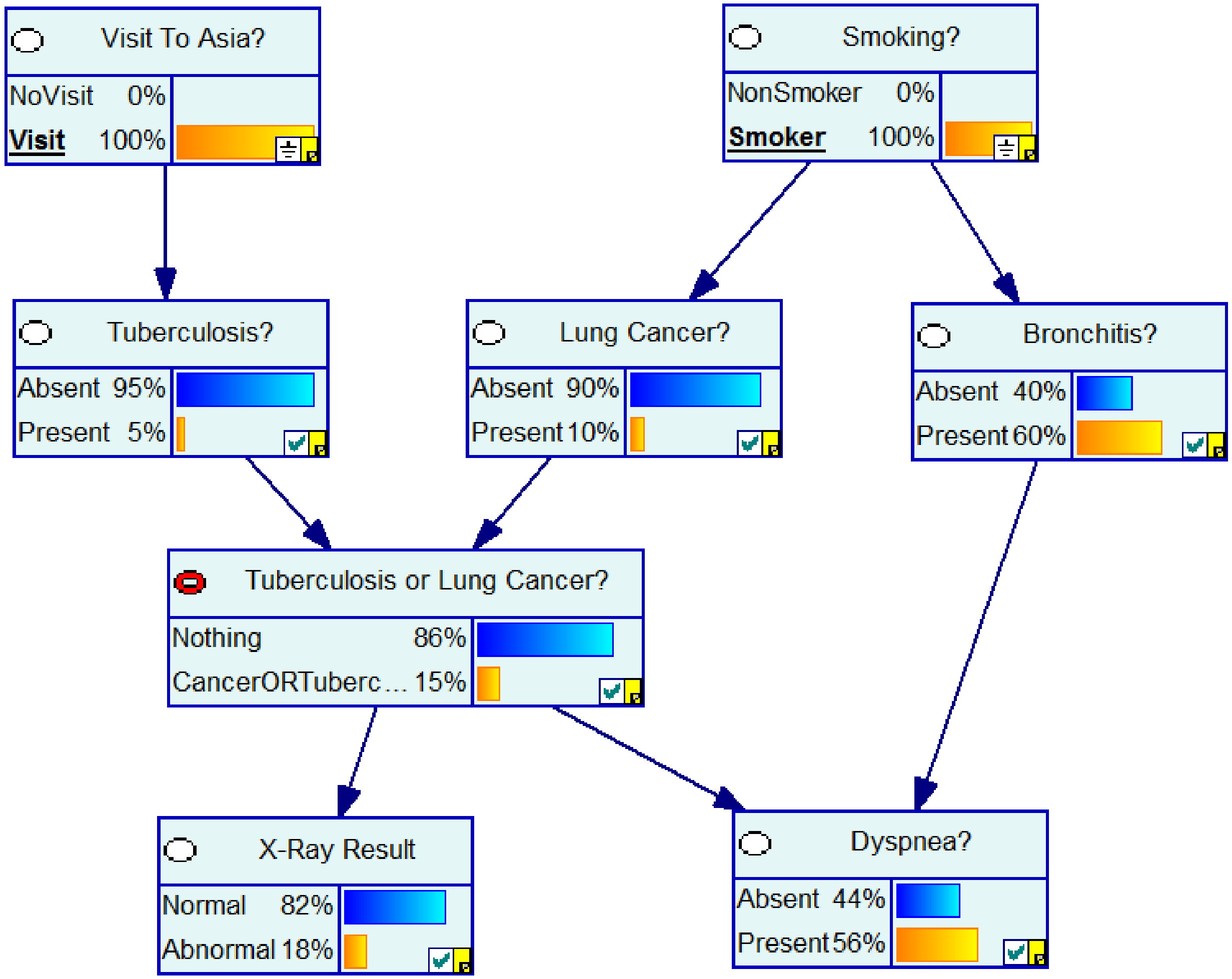
P(X)?



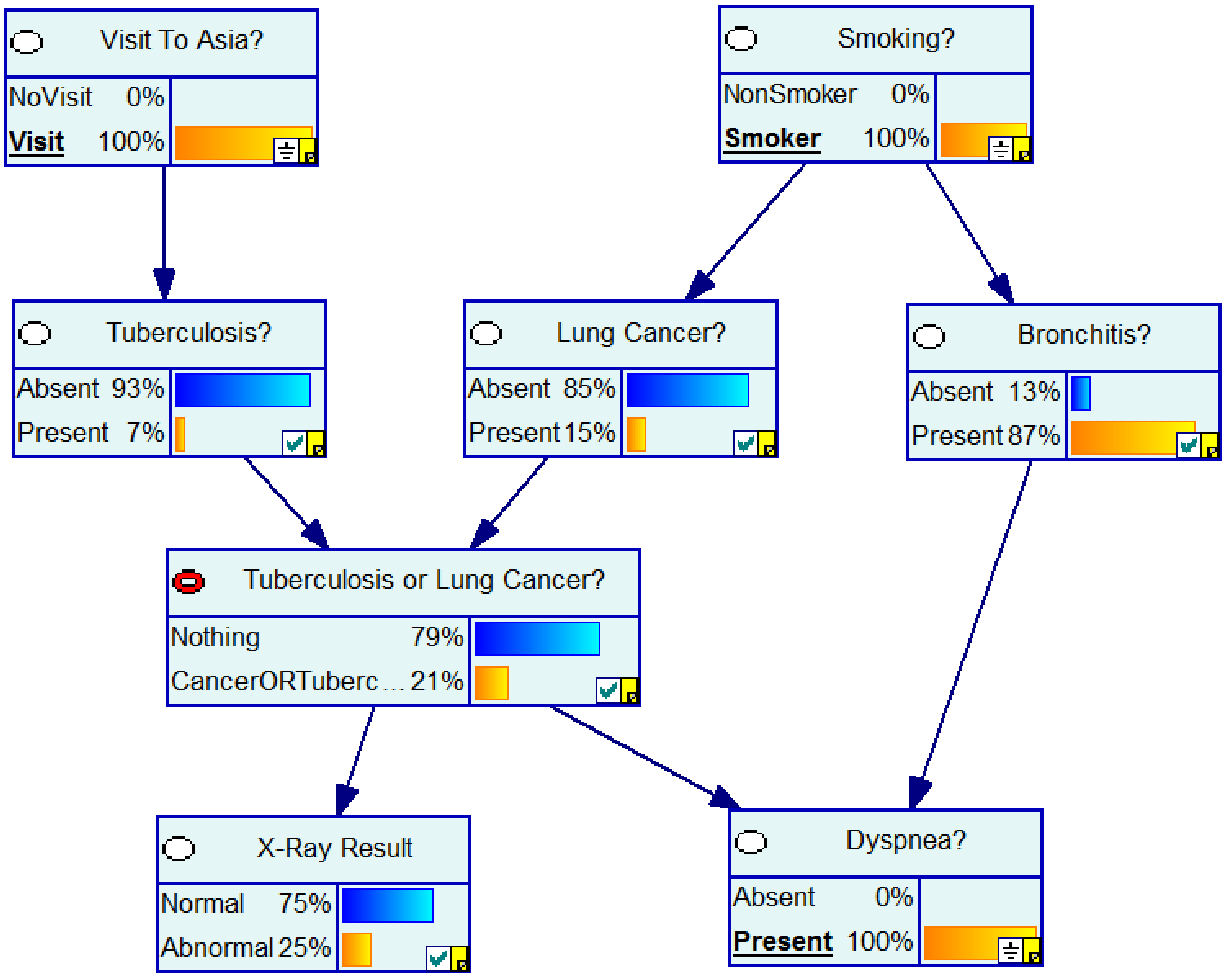
P(X | Smoker=yes)?



P(X|Asia=yes,Smoker=yes)?



P(X|Asia=yes,Smoker=yes,Dyspnea=yes)?



Types of queries

Queries: posterior probabilities

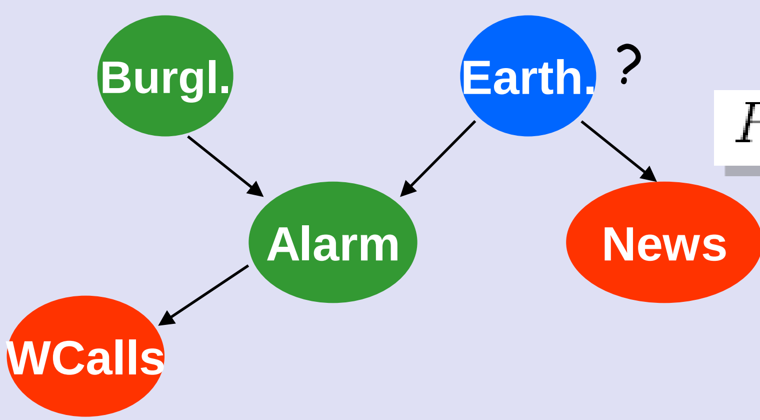
- Given some evidence e (observations),
  - Posterior probability of a target variable(s) X :

answer queries about P

$$P(X|e)$$

Vector

Other names: probability propagation, belief updating or revision...



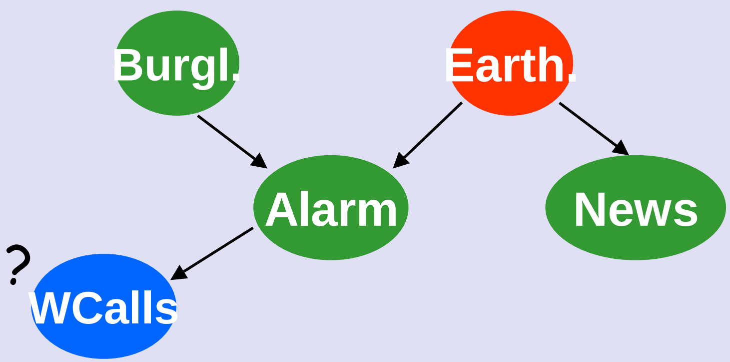
$$P(\text{Earth.}|\text{WCalls=Yes,News=Yes})?$$

Types of queries

Semantically, for any kind of reasoning

- Predictive reasoning or deductive (causal inference): predict effects

Symptoms|Disease

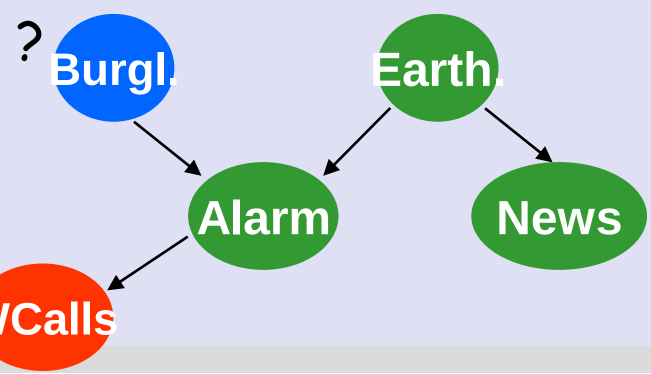


$P(\text{WCalls}|\text{Earth.=Yes})?$

Target variable is usually a descendant of the evidence

- Diagnostic reasoning (diagnostic inference): diagnose the causes

Disease|Symptoms



$P(\text{Burglary}|\text{WCalls=Yes})?$

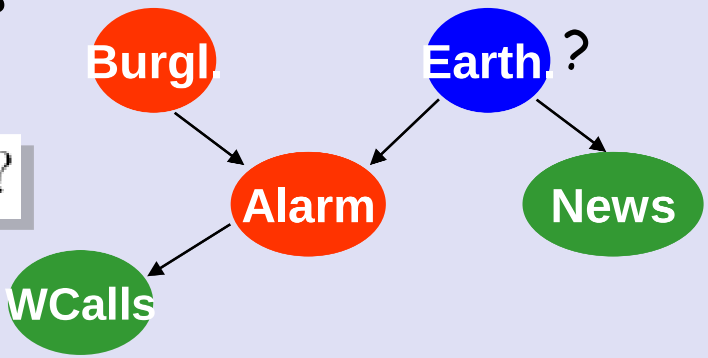
Target variable is usually an ancestor of the evidence

Types of queries

... for any kind of reasoning

- Intercausal reasoning: between causes of a common effect

$$P(\text{Earth.}|\text{Burglary=Yes,Alarm=Yes})?$$



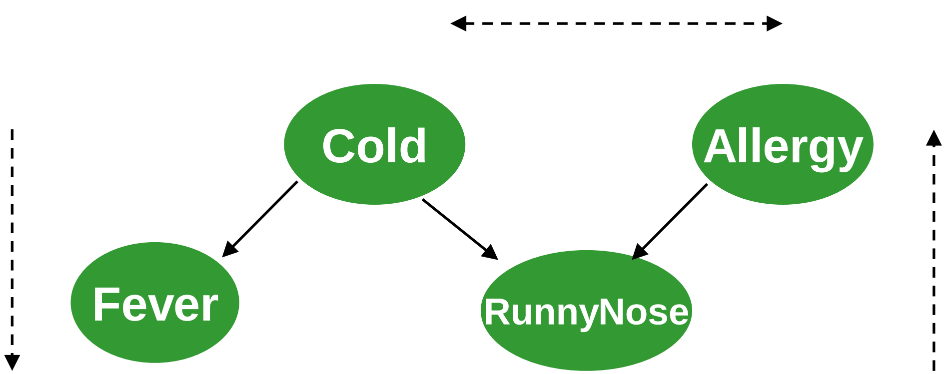
- B and E are independent of each other
- Suppose that A=Yes → It raises the Prob. for both possible causes B and E
- Suppose then that B=Yes → This explains the observed A, which in turn lowers the Prob. that E=Yes

- Two causes initially independent. If the effect is known, the presence of one explanatory cause renders the alternative cause less likely (it is explained away)

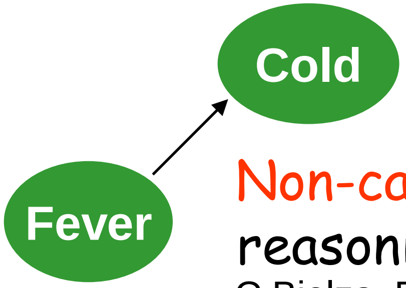
Reasoning

Intercausal reasoning

Causal reasoning



Diagnostic reasoning



Non-causal reasoning

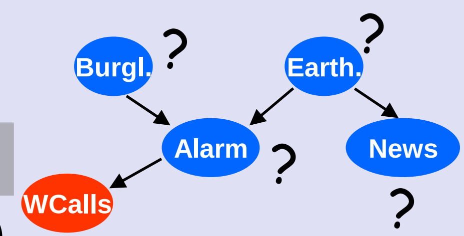
C.Bielza, P.Larrañaga -UPM-

Types of queries

More queries: maximum a posteriori (MAP)

- Most likely configurations (abductive inference): event that best explains the evidence
  - Total abduction (or MPE): search for

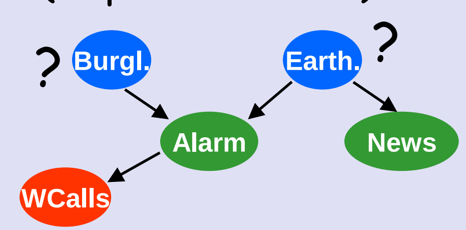
$$(x_1, \dots, x_n) \text{ such that } \max P(\overbrace{x_1, \dots, x_n}^{\text{all the unobserved}}|\mathrm{e})$$



In general, cannot be computed component-wise, with max $P(x_i|\mathbf{e})$

  - Partial abduction: search for

$$(x_1, \dots, x_l) \text{ such that } \max P(\overbrace{x_1, \dots, x_l}^{\text{subset. of unobserved (explanation set)}}|\mathrm{e})$$



  - K most likely explanations

Types of queries

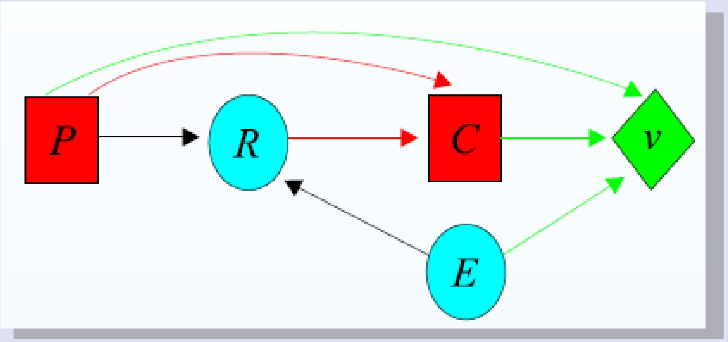
Supervised classification

- Use MPE for:
  - Find most likely label, given the evidence

$$\max_c P(c \mid x_1, \ldots, x_n)$$

Decision-making

- Optimal decisions (of maximum expected utility), with influence diagrams



$$\max_{P,C} \sum_{R,E} u(P,E,C)P(R|E,P)P(E)$$

Exact inference [Pearl'88; Lauritzen & Spiegelhalter'88]

Brute-force computation of P(X|e)

- First, consider $P(X_i)$, without observed evidence e. Conceptually simple but computationally complex
- For a BN with n variables, each with its $P(X_j|Pa(X_j))$:

$$P(X_i) = \sum_{X_1,\ldots,X_{i-1},X_{i+1},\ldots,X_n} \prod_{j=1}^{n} P(X_j|Pa(X_j))$$

Brute-force approach

- But this amounts to computing the JPD, often very inefficient and even intractable computationally
- CHALLENGE: Without computing the JDP, exploit the factorization encoded by the BN and the distributive law (local computations)

Exact inference

Improving brute-force

- Use the JPD factorization and the distributive law

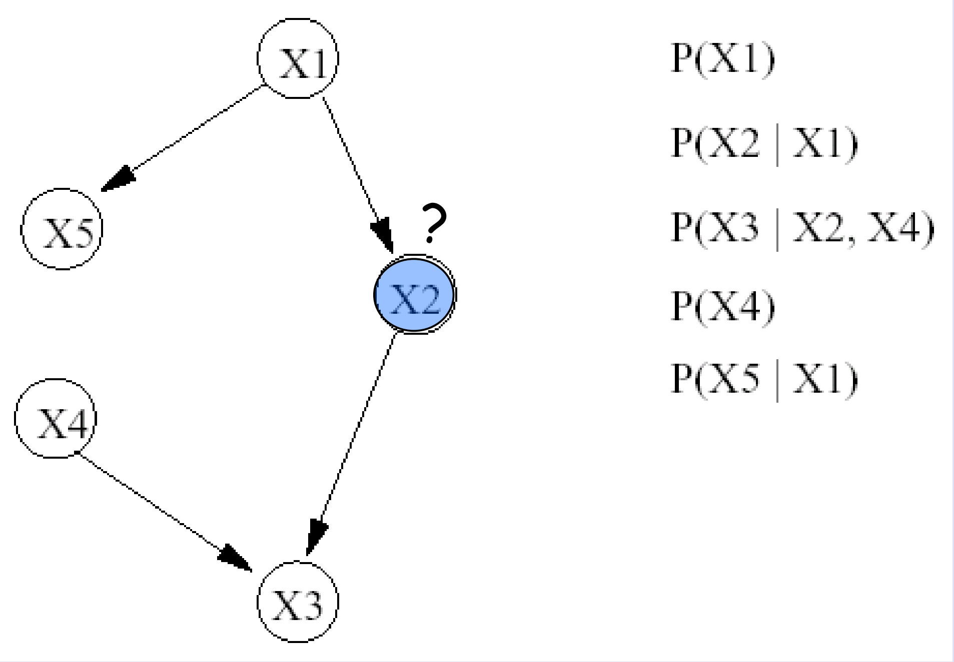


Table with 32 inputs (JPD)
(if binary variables)

$$P(X_2) = \sum_{X_1, X_3, X_4, X_5} \{P(X_1) \cdot P(X_2|X_1) \cdot P(X_3|X_2, X_4) \cdot P(X_4) \cdot P(X_5|X_1)\}$$

Exact inference

Improving brute-force

- Arrange computations effectively, moving some additions
  - over X_5 and X_3:

$$= \sum_{X_1,X_4} \left\{ \left(\underbrace{\sum_{X_5} P(X_5|X_1)}_{f_1(X_1)} \right) \cdot P(X_1) \cdot P(X_2|X_1) \cdot \left(\underbrace{\sum_{X_3} P(X_3|X_2,X_4)}_{f_2(X_2,X_4)} \right) \cdot P(X_4) \right\}$$

  - over X_4:

$$= \sum_{X_1} \left\{ \left(\underbrace{\sum_{X_5} P(X_5|X_1)}_{f_1(X_1)} \right) \cdot P(X_1) \cdot P(X_2|X_1) \cdot \left[\underbrace{\sum_{X_4} \left(\underbrace{\sum_{X_3} P(X_3|X_2,X_4)}_{f_2(X_2,X_4)} \right) \cdot P(X_4)}_{f_3(X_2)} \right] \right\}$$

Biggest table with 8 (like the BN)

Exact inference

Improving brute-force

- I.e., comparing both:

1. $$\sum_{X_1,X_3,X_4,X_5} \{P(X_1)\cdot P(X_2|X_1)\cdot P(X_3|X_2,X_4)\cdot P(X_4)\cdot P(X_5|X_1)\}$$ *Brute-force approach*

   - Table with 32 entries. 52 multiplications (tables in a suitable way) and 30 additions (marginalizations: 16, 8, 4, 2)

2. $$\sum_{X_1}\left\{\left(\sum_{X_5}P(X_5|X_1)\right)\cdot P(X_1)\cdot P(X_2|X_1)\cdot\left[\sum_{X_4}\left(\sum_{X_3}P(X_3|X_2,X_4)\right)\cdot P(X_4)\right]\right\}$$

   Factoriz. & distributive

   - 1 table with 8 and 3 with 4 entries. 14 multiplications and 14 additions (marginalizations)

Exact inference

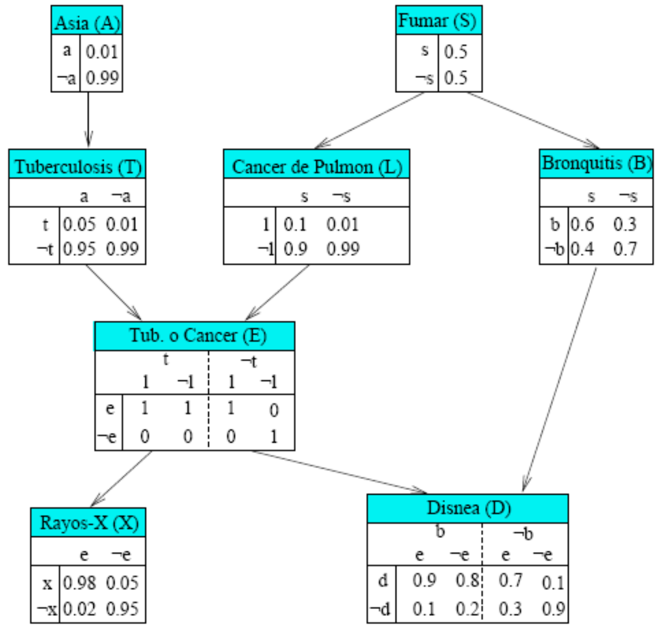
Variable elimination algorithm

- Wanted: $P(X_i|\mathrm{e})$ ONE variable
- A list with all functions of the problem $\{f_1, \ldots, f_n\}$
- Select an elimination order σ of all variables (except i)
- For each X_k from σ, if F is the set of functions that involve X_k:
  - Delete F from the list
  - Compute $f' = \sum_{X_k} (\prod_{f \in F} f)$
  - Add f' to the list
- Output: combination (multiplication) of all functions in the current list

Eliminate X_k= combine all the functions that contain this variable and marginalize out X_k

Repeat the algorithm for each target variable

Example with Asia network; P(D)?



Brute-force approach

- Compute P(D) by brute-force:

$$P(d) = \sum_x \sum_b \sum_e \sum_l \sum_t \sum_s \sum_a P(a,s,t,l,e,b,x,d)$$

- Complexity is exponential in the size of the graph (number of variables *number of states for each variable)

Example with Asia network: VE

$\sigma_1 = T, S, E, A, L, B, X.$

1 $\mathcal{L} = \{f_A(A), \underbrace{f_T(T,A)}, f_S(S), f_L(L,S), f_B(B,S), \underbrace{f_E(E,T,L)}, f_X(X,E),\ f_D(D,E,B)\}$. **Delete T.**

$$g_1(A,E,L) = \sum_T (f_T(A,T) \times f_E(E,T,L))$$

size = 16

not necessarily a probability term

2 $\mathcal{L} = \{f_A(A), \underbrace{f_S(S), f_L(L,S), f_B(B,S)}, f_X(X,E), f_D(D,E,B), g_1(A,E,L)\}$. **Delete S.**

$$g_2(L,B) = \sum_S (f_S(S) \times f_L(L,S) \times f_B(B,S))$$

size = 8

3 $\mathcal{L} = \{f_A(A), \underbrace{f_X(X,E), f_D(D,E,B), g_1(A,E,L)}, g_2(L,B)\}$. **Del. E**

$$g_3(X,D,B,A,L) = \sum_E (f_X(X,E) \times f_D(D,E,B) \times g_1(A,E,L))$$

size = 64

Example with Asia network: VE

4 $\mathcal{L} = \{\underbrace{f_A(A)}, g_2(L,B), \underbrace{g_3(X,D,B,A,L)}\}$. **Delete A** size = 32

$$g_4(X,D,B,L) = \sum_A (f_A(A) \times g_3(X,D,B,A,L))$$

5 $\mathcal{L} = \{\underbrace{g_2(L,B), g_4(X,D,B,L)}\}$. **Delete L.** size = 16

$$g_5(X,D,B) = \sum_L g_2(L,B) \times g_4(X,D,B,L)$$

6 $\mathcal{L} = \{\underbrace{g_5(X,D,B)}\}$. **Delete B.** size = 8

$$g_6(X,D) = \sum_B g_5(X,D,B)$$

7 $\mathcal{L} = \{\underbrace{g_6(X,D)}\}$. **Delete X.** size = 4

$$g_7(D) = \sum_X g_6(X,D)$$

8 return normalize($g_7(D)$)

Exact inference

elimination order $\sigma_1 = A, X, T, S, L, E, B$ → Size = 8

Variable elimination algorithm

- Local computations (due to moving the additions)
- Importance of the elimination ordering, but finding an optimal (minimum cost) is NP-hard [Arnborg et al.'87] (heuristics for good sequences)
- Complexity is exponential in the max No. of var. in the factors of the summation

Exact inference

Message passing algorithm

- It operates passing messages among the nodes of the network. Nodes act as processors that receive, calculate and send information. Called propagation algorithms
- Clique tree propagation, based on the same principle as VE but with a sophisticated caching strategy that:
  - Enables to compute the posterior prob. distr. of all variables in twice the time it takes to compute that of **one** single variable

Message passing algorithm

Basic operations for a node i

- Ask info(i,j): Target node i asks info to node j. Does it for all neighbors j. They do the same until there are no nodes to ask
- Send-message(i,j): Each node sends a message $M^{i \to j}$ to the node that asked him the info... until reaching the target node
- A message is defined over the intersection of domains F_i and F_j of f_i and f_j:

$$M^{i \to j} = \sum_{X \notin F_i \cap F_j} f_i \cdot \left(\prod_{k \neq j} M^{k \to i} \right)$$

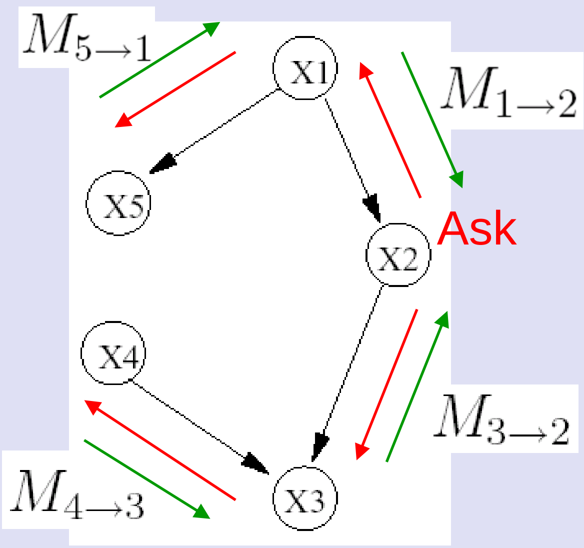
- And finally, we calculate locally at each node i:

Target combines all received info with his info and marginalize over the target variable

$$P(X_i|\mathbf{e}) = \text{normalize}\left[\sum_{X_j \neq X_i} \left(f_i \cdot \prod_{k \in \text{neighbours}(X_i)} M^{k \to i} \right) \right]$$

Message passing algorithm

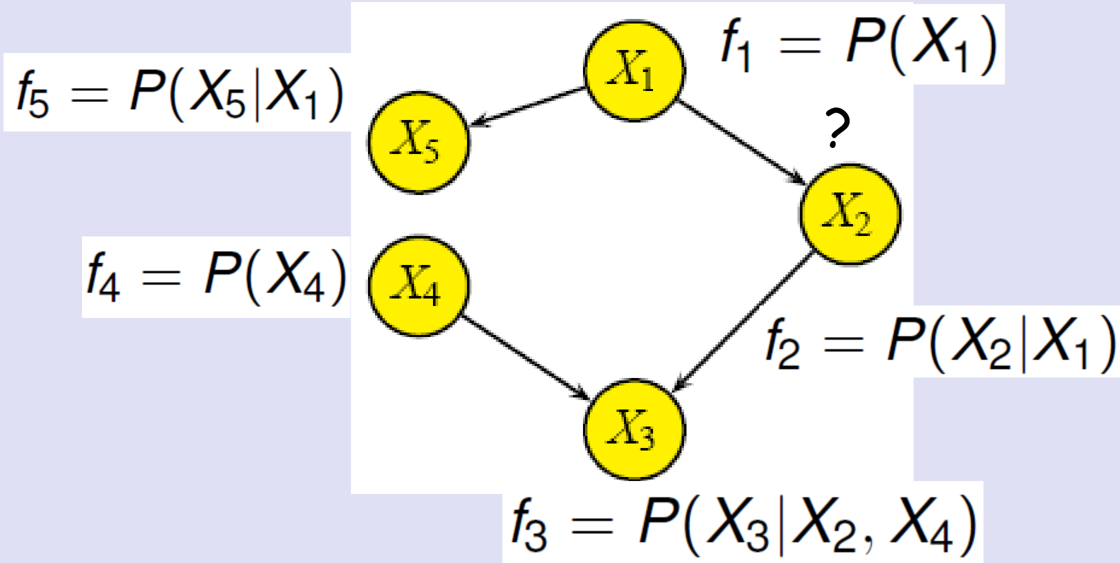
Procedure for X_2



CollectEvidence

Exact inference

$P(X_2)$ as a message passing algorithm



1. Ask(2, 1) and Ask(2, 3). Ask(3, 4) and Ask(1, 5)
2. $M^{5\to1} = \sum_{X_5} f_5(X_5, X_1)$. $M^{4\to3} = f_4(X_4)$
3. $M^{1\to2} = f_1(X_1)M^{5\to1}$. $M^{3\to2} = \sum_{X_3,X_4} f_3(X_3, X_2, X_4)M^{4\to3}$
4. $P(X_2) = \mathrm{normalize}(\sum_{X_1} f_2(X_2, X_1)M^{1\to2}M^{3\to2})$

Approximate inference

Why?

- Because exact inference is intractable (NP-complete) with large (+40) and densely connected BNs
  - the associated cliques for the junction tree algorithm or the intermediate factors in the VE algorithm will grow in size, generating an exponential blowup in the number of computations performed

➡ Stochastic simulation to find approximate answers

Approximate inference

Stochastic simulation

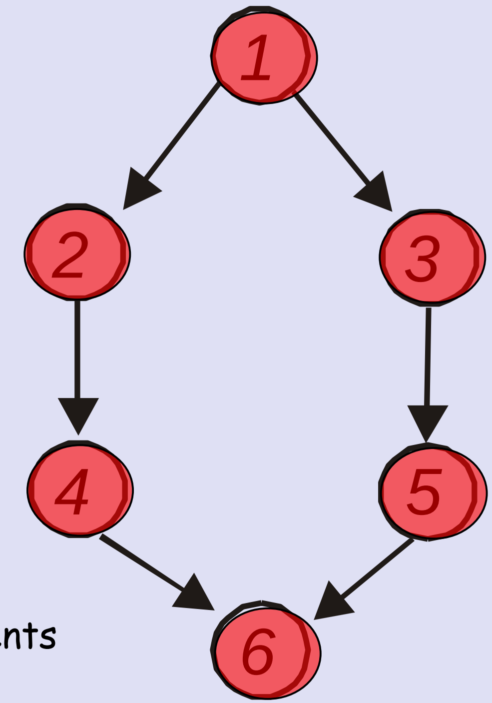
- Uses the network to generate a large number of cases (full instantiations) from the network distribution
- $P(X_i|\mathbf{e})$ is estimated using these cases by counting observed frequencies in the samples. By the Law of Large Numbers, estimate converges to the exact probability as more cases are generated

Approximate inference

Probabilistic logic sampling [Henrion'88]

- Given an ancestral ordering of the nodes (parents before children), generate from X once we have generated from its parents (i.e. from the root nodes down to the leaves)

When all the nodes have been visited, we have a <u>case</u>, an instantiation of all the nodes in the BN



Use conditional prob. given the known values of the parents

- Repeat and use the observed frequencies to estimate $P(X_i|\mathbf{e})$

Approximate inference

Probabilistic logic sampling

- Suppose we obtain the following samples:

(0,1,1,1,1,1), (0,1,0,1,1,1), (1,0,0,1,1,1), (0,0,1,1,1,0), (1,1,1,1,0,0)

- Then: $\hat{p}(X_1 = 0) = \frac{3}{5}$

- With evidence, e.g. X_2=1, we discard the third and fourth samples and we would repeat until having a sample of size 5 as desired

(0,1,1,1,1,1), (0,1,0,1,1,1), (1,1,0,0,1,1), (1,1,1,1,1,0), (1,1,1,1,0,0)

$$\hat{p}(X_1 = 0 | X_2 = 1) = \frac{2}{5}$$

Approximate inference

Likelihood weighting

Likelihood weighting:

- Don't generate from **E**; fix its values **E=e**
- Generate from the rest as in PLS
- Instead of adding "1" to the run count, the CPTs for the evidence nodes are used to determine how likely that evidence combination is:
  - For a sample i, assign a weight w_i given by the likelihood of the evidence given its parents

$$w_i = \prod_{X_i \in \mathbf{E}} P(X_i = e_i | \text{Pa values})$$

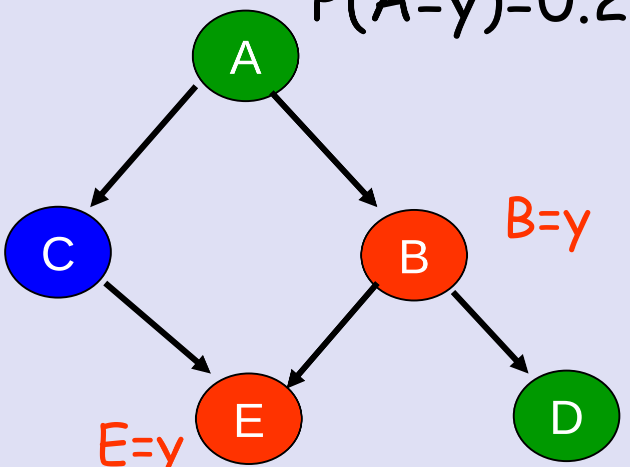
In PLS, w_i=1 for samples consistent with **e** and w_i=0 otherwise

Approximate inference

Likelihood weighting: example

- $P(C=n|\underbrace{B=y,E=y}_{e})$??

$P(A=y)=0.2$ →A=n



C=n← $P(C=y|A=n)=0.7$

B=y $P(B=y|A=n)=0.4$

E=y

$P(D=y|B=y)=0.7$

$P(E=y|C=n,B=y)=0.8$

→D=y

(A=n,**B=y**,C=n,D=y,**E=y**) with $w_1=0.4*0.8=0.32$

(A=n,**B=y**,C=y,D=n,**E=y**) with $w_2=0.88$

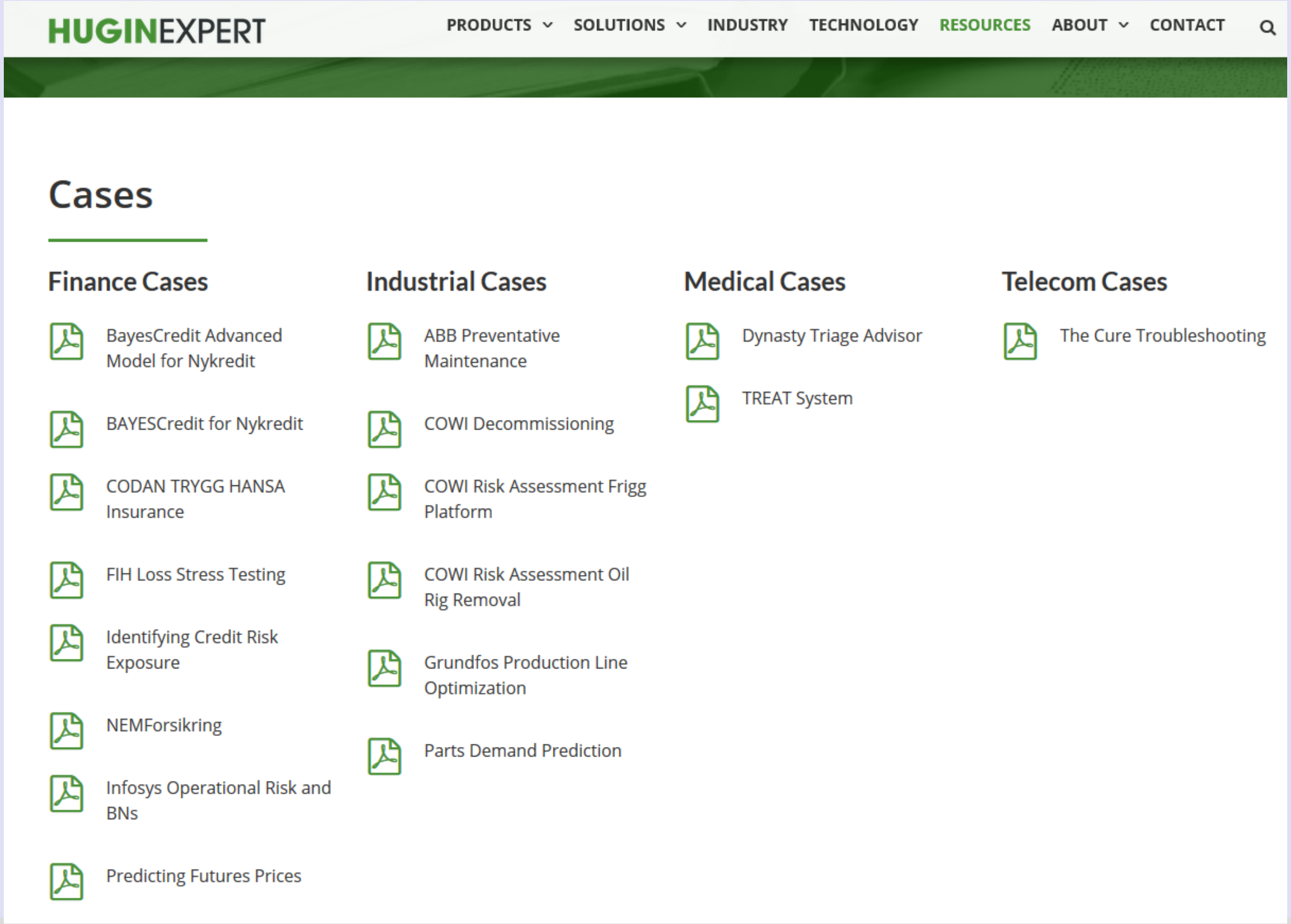
(A=y,**B=y**,C=y,D=y,**E=y**) with $w_3=0.80$

$$P(C=n|\mathbf{e})=\frac{0.32}{0.32+0.88+0.80}=0.16$$

Software

http://www.cs.ubc.ca/~murphyk/Software/bnsoft.html
http://www.cs.iit.edu/~mbilgic/classes/fall10/cs595/tools.html

www.hugin.com/



Software

www.bayesia.com

BAYESIALAB

Bayesia S.A.S. | BayesiaLab Software | BayesiaLab Documentation | Bayesian Networks in Theory & Practice | Courses & Events | BayesiaLab Store | Forum | News

Bayesia S.A.S. is a French software development company, founded in 2001 by Dr. Lionel Jouffe and Dr. Paul Munteanu, which specializes in artificial intelligence technology.

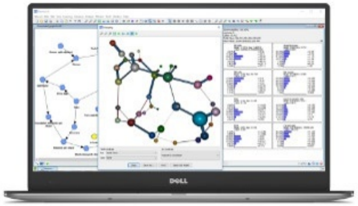
Bayesia's software portfolio focuses on all aspects of decision support with Bayesian networks and includes **BayesiaLab**, **BEST**, and **BRICKS**. Their spectrum of applications ranges from **individual decision support** to **large-scale policy analysis** and **risk assessment** of industrial systems.

The Bayesia Product Portfolio >

- BayesiaLab—Your Desktop Analytics and Research Laboratory
- BEST—Bayesian Expert System for Troubleshooting
- BRICKS—Bayesian Representation and Inference for Complex Knowledge Structuring

The Bayesia Story

BayesiaLab—Your Desktop Analytics and Research Laboratory



BayesiaLab is a powerful Artificial Intelligence software that provides scientists a comprehensive "lab" environment for machine learning, knowledge modeling, analytics, simulation, and optimization — all based on the Bayesian network paradigm.

BayesiaLab is a desktop application (Win/Mac/Unix), which features a highly intuitive graphical user interface. It is ideally suited for individual researchers or subject-matter experts who are exploring complex problem domains. **Learn More...**

BEST—Bayesian Expert System for Troubleshooting

The BEST suite consists of a set of software modules for the design, administration and operational deployment of intelligent troubleshooting applications for complex technical domains. It provides powerful and easy-to-use tools for modeling and disseminating expert knowledge. BEST features machine learning capabilities that can automatically improve models using in-service experience. Additionally, BEST can generate troubleshooting strategies that minimize the cost of

Software

www.bayesfusion.com

BAYESFUSION, LLC
Data Analytics, Mathematical Modeling, Decision Support

Phone: +1-412-444-5476

Home | What We Offer | Downloads | Academia | Forum | Resources | News | FAQ | Contact Us

Scientific Research

BayesFusion, LLC, provides decision modeling software based on decision-theoretic principles. Our key product is **GeNIe Modeler**, a tool for modeling and learning with **Bayesian networks**, **dynamic Bayesian networks**, and **influence diagrams**. Our software library, **SMILE Engine**, allows for including our methodology in customers' applications, which can be written in a variety of programming languages (e.g., C++, Python, Java, .NET). **SMILE.COM** allows for easy integration with MS Excel. We also offer training, scientific consulting, and custom software development.

The areas of application of our software include:

Decision Modeling | Strategic Planning | Data Science

Software

https://code.google.com/archive/p/bnt/

Project

Source

Issues

Wikis

Downloads

bnt

Bayes Net Toolbox for Matlab

Bayes Net Toolbox for Matlab

Written by Kevin Murphy, 1997--2002. Last updated: 19 October 2007.

As on January 2014, a copy of this is available at https://github.com/bayesnet/bnt

http://bnt.googlecode.com/svn/trunk/docs/mathbymatlab.gif

- Major Features
- Examples of supported Models
- Download zip file
- Installation
- How to use the toolbox
- Subscribe to the BNT Email List
- Invited Paper on BNT published in Computing Science and Statistics, 2001.
- Other Bayes net software
- A brief introduction to Bayesian Networks
- Terms and conditions of use (GNU Library GPL)
- Why do I give the code away?
- Changelog
- Why MATLAB?
- Acknowledgements
- How do I contribute changes to the code?

Software

www.openmarkov.org/ (UNED)

Software

reasoning.cs.ucla.edu/samiam/

Software

www.r-project.org/

`bnlearn, deal, pcalg,`
`catnet, mugnet, bnclassify` → learning

`gRbase, gRain` → inference

Textbooks

- E. Castillo, J.M. Gutiérrez, A.S. Hadi (1997) *Expert Systems and Probabilistic Network Models*. Springer
- R.G. Cowell, A.P. Dawid, S.L. Lauritzen, D.J. Spiegelhalter (1999) *Probabilistic Networks and Expert Systems*. Springer
- F.V. Jensen, T. Nielsen (2007) *Bayesian Networks and Decision Graphs*. Springer
- K.B. Korb, A. Nicholson (2004) *Bayesian Artificial Intelligence*. Chapman and Hall
- R. Neapolitan (2004) *Learning Bayesian Networks*. Prentice Hall
- J. Pearl (1988) *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. Morgan Kaufmann
- Proceedings of the most important related conference: *Uncertainty in Artificial Intelligence*. http://www.auai.org

- A. Darwiche (2014) *Modeling and Reasoning with BNs*, Cambridge U.P.
- U. Kjaerulff, A. Madsen (2008) *Bayesian Networks and Influence Diagrams -A Guide to Construction and Analysis*. Springer
- D. Koller, N. Friedman (2009) *Probabilistic Graphical Models*, The MIT Press
- T. Koski, J. Noble (2009) *Bayesian Networks: An Introduction*. Wiley