Topology learning for bayesian networks with the K2 algorithm

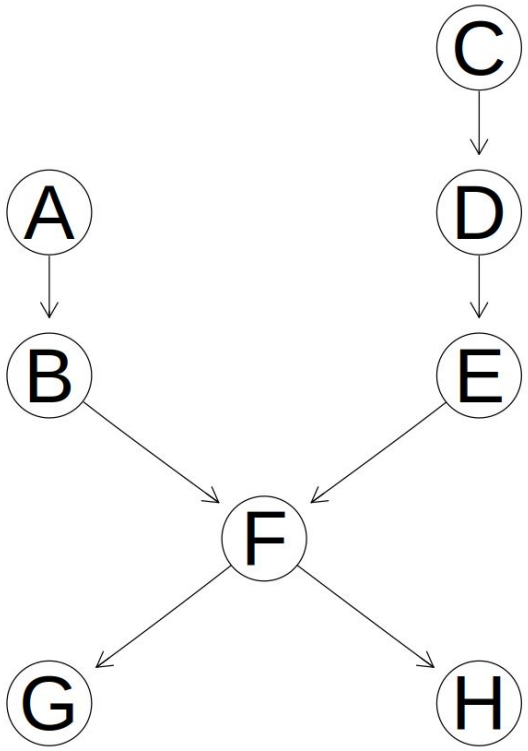
1.

Bayesian Networks

Essential theory review

Bayesian Networks

- Directed acyclic graph (**DAG**)
- Each node represents a random variable
- Connections represent relations of dependency



Bayesian Networks

- Graphical representation of the conditional dependencies between n variables
- Joint probability of n variables written as a factorization of n conditional probabilities

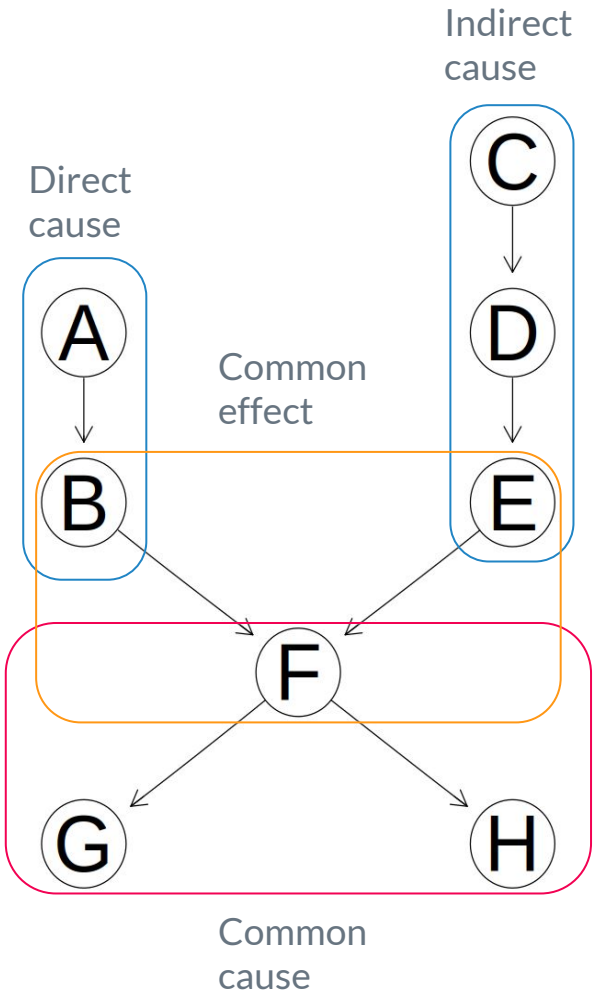
$$p(X_1, \ldots, X_n) = \prod_{i=1}^{n} p(X_i | Parents(X_i))$$

Where parents are just the nodes with a outgoing connection to X_i.

Bayesian Networks

Local probabilities example

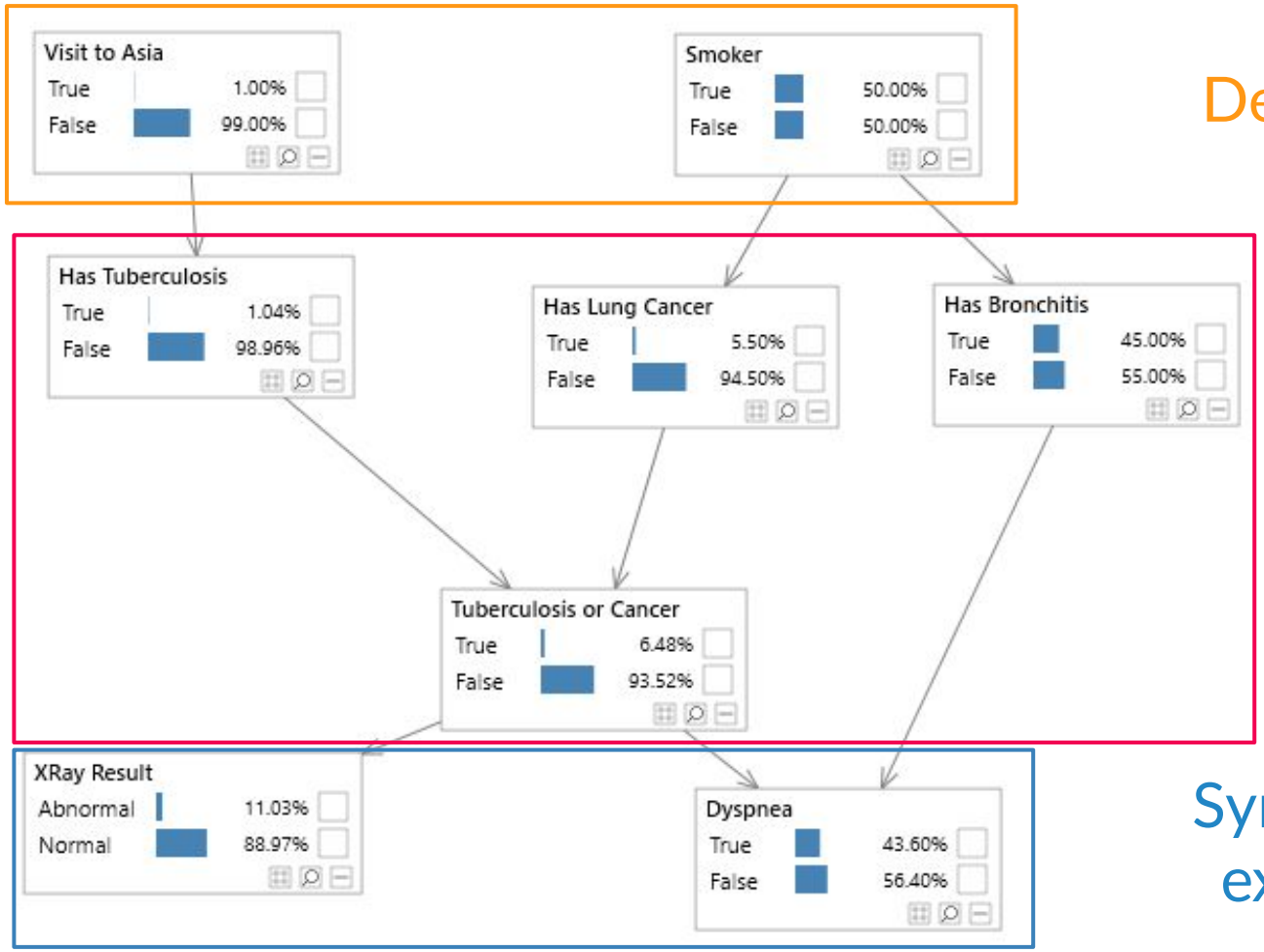
- $P(A,B) = P(B|A)P(A)$
- $P(C,D,E) = P(E|D)P(D|C)P(C)$
- $P(B,E,F) = P(F|B,E)P(B)P(E)$
- $P(F,G,H) = P(G|F)P(H|F)P(F)$



Applications

- Causality maps
- Partial information inference
- Gene regulatory networks
- Medical diagnosis systems

ASIA Medical Diagnosis System



Determinants

Diseases

Symptoms and exam results

Learning Bayesian Networks

Bayesian network

$$B = (G, X) \rightarrow B = (G, \theta)$$

Estimating network probability

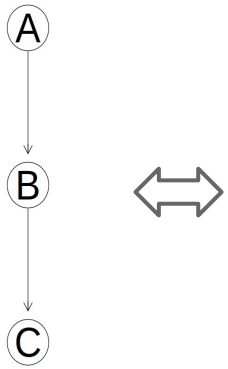
$$Pr(B|D) = Pr(G, \theta|D) = \underbrace{Pr(G|D)}_{\text{Structure Learning}} * \underbrace{Pr(\theta|G, D)}_{\text{Parameters Learning}}$$

Kinds of connections

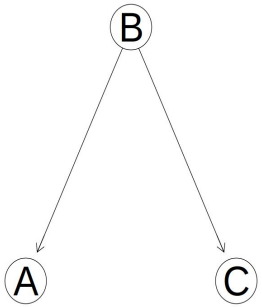
Variable connections

$$Pr(A,B,C) = Pr(C|B)Pr(B|A)Pr(A)$$
$$= Pr(A,B)Pr(C|B)$$
$$= Pr(A|B)Pr(C|B)Pr(B)$$

Serial Connection

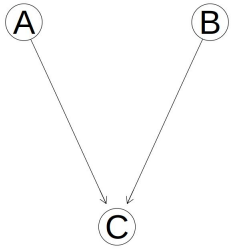


Divergent Connection



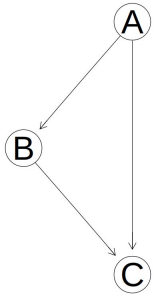
Fixed connections (**V-structures**)

$$Pr(A,B,C) = Pr(C|A,B)Pr(A)Pr(B)$$



V-structure

Not a V-structure



Equivalence classes

Two DAGs are equivalent (they share the same equivalence class) if they have:

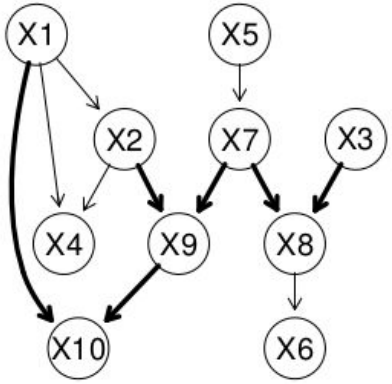
1. Same skeleton
2. Same V-structures

Skeleton := undirected graph associated to the DAG

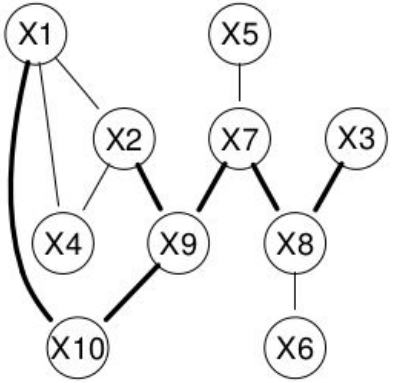
Example

- V-structures highlighted
- DAG 1 and 2 are equivalent
- X1⟶ X4 ⟵X2 is not a V-structure because X1⟶X2

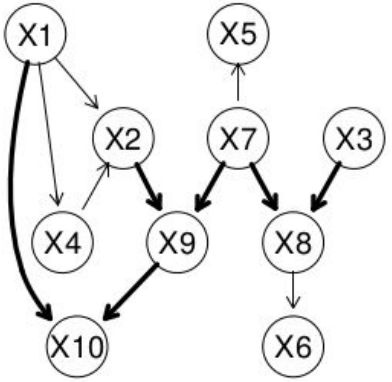
DAG 1



Skeleton



DAG 2

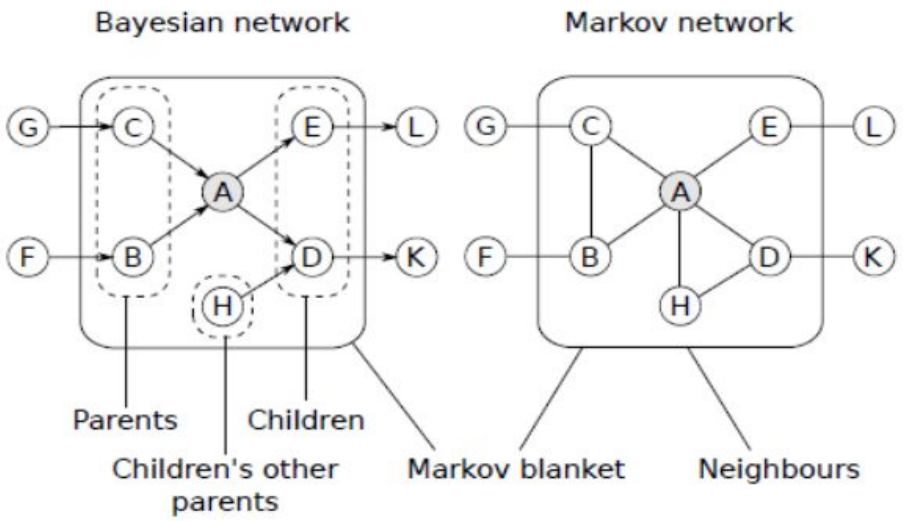


Local Markov Property

Markov blanket

The Markov blanket of a node A is the set consisting of the parents of A, the children of A and all the other nodes sharing a child with A.

It is all we need to know to make inference on A

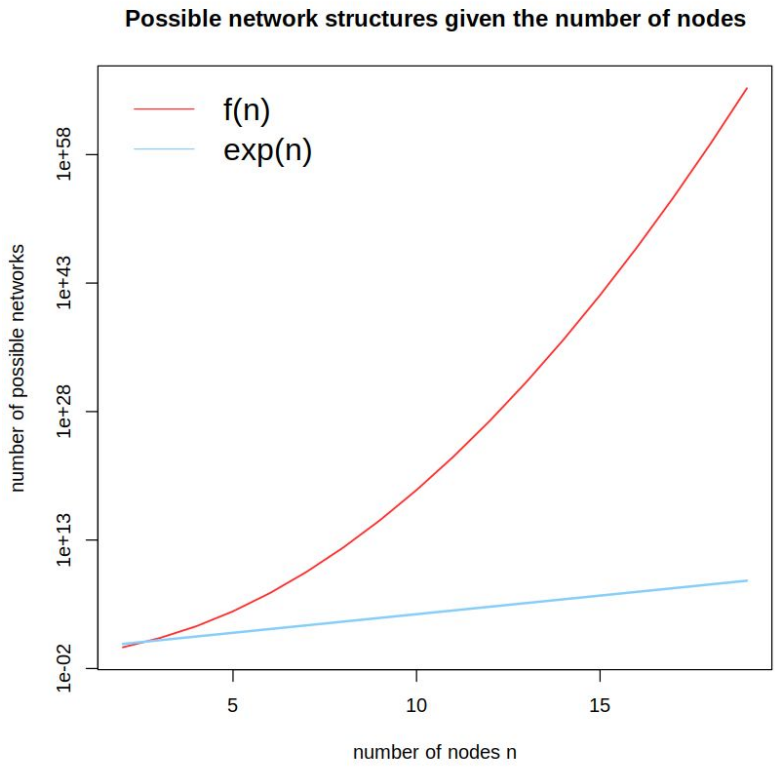


Learning Network Structure

GOAL: find the most probable structure.

ISSUE: number of possible structures f grows more than exponentially in the number of nodes n.

NP-hard problem!



$$f(n) = \sum_{i=1}^{n} (-1)^{i+1} 2^{i(n-i)} f(n-i)$$

2.
K2 Algorithm

Efficient structure learning

Basic ideas

Learning the network structure is an NP-hard problem.

IDEAS:

1. Provide prior nodes ordering to reduce the problem to polynomial complexity.
2. The most probable structure B^*, given a dataset D, is also the one maximizing the joint probability $Pr(B,D)$, since

$$Pr(B_i|D) = \frac{Pr(B_i,D)}{Pr(D)} \propto Pr(B_i, D) \quad \forall i$$

Overview

Task: search the most probable network structure given a database of cases, i.e. $B^* = \underset{B}{\operatorname{argmax}}(Pr(B, D))$

Inputs:

- Database D, containing m cases
- Set of n nodes, one for each variable x_i
- Ordering on the nodes
- Upper bound u on the number of parents

Output:

- Set of parents for each node.

Notation and terminology

- π_i : Set of parents of node x_i
- φ_i : List of all possible configurations of the parents of x_i
- V_i : List of all possible values of the node x_i

$$q_i = |\phi_i| \qquad r_i = |V_i|$$

- α_{ijk} : Number of cases in D in which x_i assumes its k^{th} value, and its parents π_i are instantiated with the j^{th} configuration in φ_i
- N_{ij} : Number of cases in D in which the parents π_i are instantiated with the j^{th} configuration in φ_i

$$N_{ij} = \sum_{k=1}^{r_i} \alpha_{ijk}$$

Main result

- $f(i,\pi_i)$ can be interpreted as the probability of the database D given that the parents of x_i are π_i

$$f(i,\pi_i) = \prod_{j=1}^{q_i} \left(\frac{(r_i-1)!}{(N_{ij}+r_i-1)!} \prod_{k=1}^{r_i} \alpha_{ijk}! \right)$$

- When $\pi_i = \emptyset$, we have:
  - $q_i = 0$, so j loses meaning and the first productory is ignored
  - Without j, we obtain $N_{ij} \rightarrow N_{i\_} = m$
  - $f(i,\emptyset)$ can be written as

$$f(i,\emptyset) = \frac{(r_i-1)!}{(m+r_i-1)!} \prod_{k=1}^{r_i} \alpha_{i\_k}!$$

Implementation of K2

for i:= 1 to n do
    $\pi_i := \emptyset$;
    $P_{old} := f(i, \pi_i)$; {This function is computed using Equation 20.}
    OKToProceed := **true**;
    While OKToProceed and $|\pi_i| < u$ do
        let z be the node in Pred(x_i) - π_i that maximizes $f(i, \pi_i \cup \{z\})$;
        $P_{new} := f(i, \pi_i \cup \{z\})$;
        if $P_{new} > P_{old}$ **then**
            $P_{old} := P_{new}$;
            $\pi_i := \pi_i \cup \{z\}$;
        else OKToProceed := **false**;
    end {while};
    write('Node: ', x_i, ' Parent of x_i: ',π_i);
end {for};
end {K2};

Factorial problem

- Most datasets contain lots of samples ⇨ Computing the factorials in $f(i,\pi_i)$ could be not possible
- We don't need $f(i,\pi_i)$ values directly (we only compare them) ⇨ Use $\ln(f(i,\pi_i))$ instead

$$\ln(f(i,\pi_i)) = \sum_{j=1}^{q_i} \ln\left(\frac{(r_i-1)!}{(N_{ij}+r_i-1)!}\prod_{k=1}^{r_i}\alpha_{ijk}!\right)$$

- Now each term of the sum can be rewritten without factorials. In this contest we will use $\alpha_{ijk} \rightarrow \alpha_k$

Factorial problem - Sum term

$$\ln\left(\frac{(r_i-1)!}{(N_{ij}+r_i-1)!}\prod_{k=1}^{r_i}\alpha_k!\right) =$$

$$\ln((r_i-1)!) + \ln\left(\prod_{k=1}^{r_i}\alpha_k!\right) - \ln\left((N_{ij}+r_i-1)!\right) =$$

$$\ln((r_i-1)!) + \sum_{k=1}^{r_i}\ln(\alpha_k!) - \sum_{x=1}^{N_{ij}+r_i-1}\ln(x) =$$

$$\ln((r_i-1)!) + \sum_{n=1}^{max_k(\alpha_k)} C_n \ln(n) - \sum_{x=1}^{N_{ij}+r_i-1}\ln(x) =$$

$$\ln((r_i-1)!) + \sum_{n=1}^{max_k(\alpha_k)} (C_n-1)\ln(n) - \sum_{n=max_k(\alpha_k)+1}^{N_{ij}+r_i-1}\ln(n) =$$

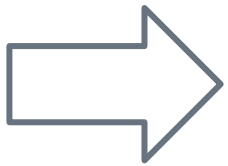
$$\ln((r_i-1)!) + \sum_{n=1}^{N_{ij}+r_i-1}(C_n-1)\ln(n)$$

$$with\ C_n = 0\ \forall n > \max_k(\alpha_k)$$

- Where C_n represents the number of how many α_k are greater than n

Ordering methods

- K2 algorithm needs an ordering over the set of variables to evaluate
- Nodes search their parents only between the ones that come before them



Choosing a good ordering method is fundamental

MI: Mutual information

- **MI** measures how much a variable tells about another one (in other words, the mutual dependence)

$$MI(X;Y) = D_{KL}(P_{(X,Y)} || P_X \otimes P_Y)$$
$$= \sum_{y \in \mathcal{Y}} \sum_{x \in \mathcal{X}} p_{(X,Y)}(x,y) \log\left(\frac{p_{(X,Y)}(x,y)}{p_X(x) p_Y(y)}\right)$$

- **MI** can also be expressed as a function of conditional and joint entropies:

$$MI(X;Y) \equiv H(X) - H(X|Y)$$
$$\equiv H(Y) - H(Y|X)$$
$$\equiv H(X) + H(Y) - H(X,Y)$$
$$\equiv H(X,Y) - H(X|Y) - H(Y|X)$$

MI: Implementation

Procedure:

1. First, select arbitrarily a first node (classification target);
2. Compute **MI** values between it and every other non-ordered node;
3. Find max and add the corresponding node at the end of the ‘order’ array;
4. Repeat cycle from point 2 using this last node until every node is in the ‘order’ array.

MI: Pros and Cons

Pros

- Entropy-based approach ⇨ node ordering is built according to the dependency among variables
- Efficient algorithm ⇨ short running times and does not need a lot of memory

Cons

- We may not have a classification target
- MI is symmetric ⇨ $MI(X;Y) = MI(Y;X)$

Factor Analysis (FA)

Describe observed variables through a lower number of hidden variables called **factors**, plus some errors.

The fractions of the variance in the observed variables that is accounted for by the factors are called **communalities**.

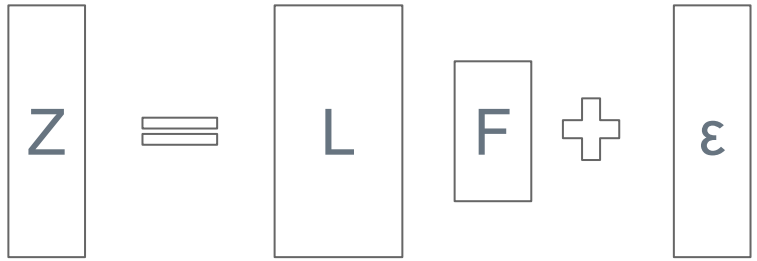
The variables with the highest communality are the most “fundamental”, because they contain the shared factors in a degree higher than the other variables.

Ordering is given by the ranking of the communalities.

FA theory

Assumptions:

1. F and ε independent
2. $E(F) = 0$
3. $Cov(F) = I$



Zero-mean standardized variables | Loadings matrix | Factors | Errors

$$Z = (z_1, \ldots, z_n)$$

$$F = (F_1, \ldots, F_m)\ ,\ m < n$$

$$\begin{aligned} Cov(Z) &= Cov(LF + \epsilon) \\ &= LCov(F)L^T + Cov(\epsilon) \\ &= LL^T + Cov(\epsilon) \end{aligned}$$

FA theory

Matrix notation: $Cov(Z) = LL^T + Cov(\epsilon)$

Indices notation: $\sum_{i=1}^{N} z_{ai} z_{bi} = \sum_{p=1}^{m} l_{ap} l_{bp} + \sum_{i=1}^{N} \epsilon_{ai} \epsilon_{bi}$

On the diagonal holds

$$1 = \underbrace{(LL^T)_{aa}}_{\text{Communalities } C_a} + Cov(\epsilon)_{aa} \Longrightarrow \quad C_a = 1 - Cov(\epsilon)_{aa}$$

Best fit: minimize off-diagonal terms of *Cov(ε)*

$$min \sum_{ab, a \neq b} \left[\sum_i z_{ai} z_{bi} - \sum_p l_{ap} l_{bp} \right]^2$$

FA: Pros and Cons

Pros

- No prior knowledge needed to order the nodes
- Can be used together with other ordering techniques that require knowing the first node when this is unknown

Cons

- Number of factors needed as input ➛ algorithms already implemented to find the best number of factors to use (e.g. parallel analysis)
- Computationally intensive

K2 considerations

K2 algorithm trades off computational complexity for prior knowledge:

- The algorithm uses the ordering to reduce the complexity of an NP-hard problem to a Polynomial one

- K2 output depends heavily on the initial ordering:

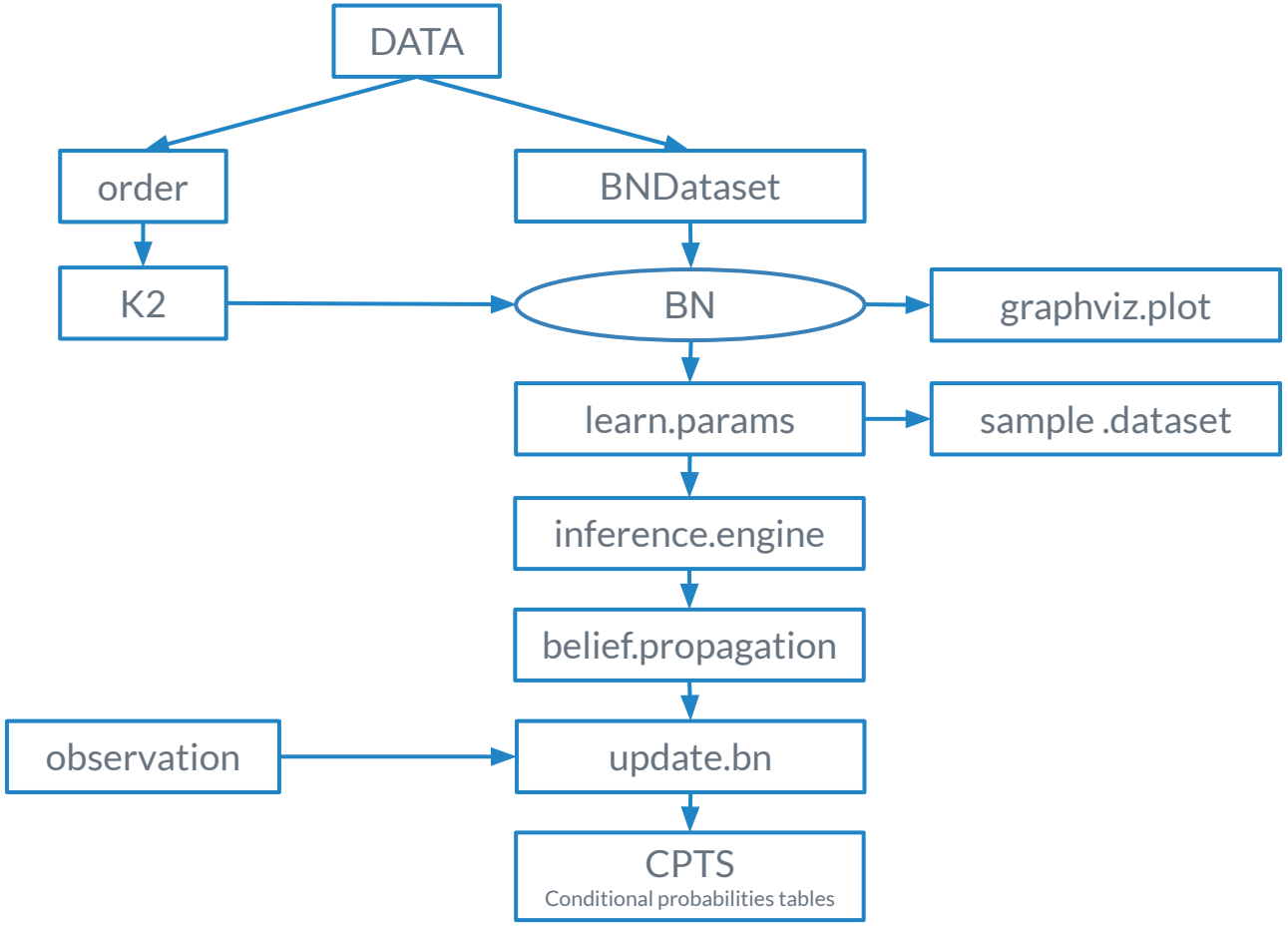
  Small changes in the ordering can reshape significantly the topology of the network.

3.

From Structure to Inference

Concrete applications of Bayesian Networks

Workflow

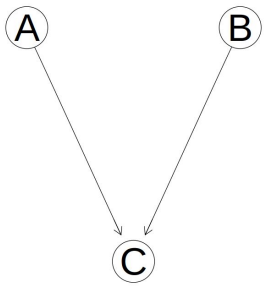

DATA
order
BNDataset
K2
BN
graphviz.plot
learn.params
sample .dataset
inference.engine
belief.propagation
observation
update.bn
CPTS
Conditional probabilities tables

Learning parameters

Estimate conditional probabilities tables (CPTS)

Two possible methods:

- Maximum Likelihood Estimation (MLE)
- Bayesian Inference



CPTS	A=0 B=0	A=0 B=1	A=1 B=0	A=1 B=1
C=0	α_{C00}	α_{C10}	α_{C20}	α_{C30}
C=1	α_{C01}	α_{C11}	α_{C21}	α_{C31}

← parents

Maximum Likelihood Estimation

Frequentist approach

$$p_{ijk} \equiv Pr(X_i = x_{ik} | \pi(X_i) = \phi_{ij}) = \frac{\alpha_{ijk}}{N_{ij}}$$

Where

- α_{ijk} : Number of cases in D in which x_i assumes its k^{th} value, and its parents π_i are instantiated with the j^{th} configuration in φ_i
- N_{ij} : Number of cases in D in which the parents π_i are instantiated with the j^{th} configuration in φ_i

Bayesian Inference

1. Assume prior Π (e.g. uniform)
2. Compute $\mathrm{p_{ijk}}$ with MLE from n samples
3. Choose imaginary sample size (iss)
4. $\forall$ j,k the posterior probability is given by

$$\hat{p}_{ijk} = \frac{iss\ \Pi_{ijk} + n\ p_{ijk}}{n + iss}$$

Basically the iss is the weight in terms of samples that is given to the prior.

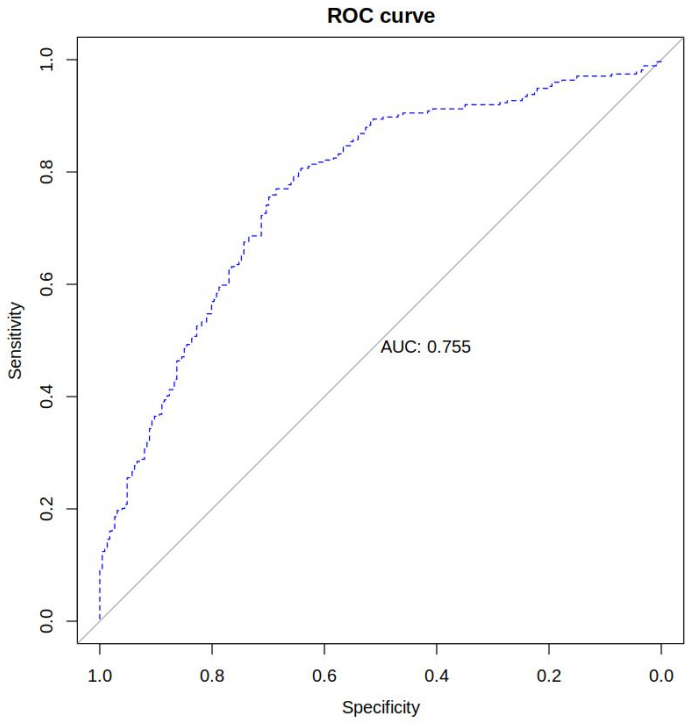
4.
Datasets and Results

ROC curve

Used to evaluate the performance of a binary classifier

Metrics

- Sensitivity: true positive ratio
- Specificity: true negative ratio
- AUC: the area under the ROC curve, it's 1 for a perfect classifier



Naive Dataset

- x_1 is the classification target ⇨ First in the node ordering.
- Output of K2 algorithm

$$\text{Parent of } x_1 : \pi_1 = \emptyset$$
$$\text{Parent of } x_2 : \pi_2 = \{x_1\}$$
$$\text{Parent of } x_3 : \pi_3 = \{x_2\}$$

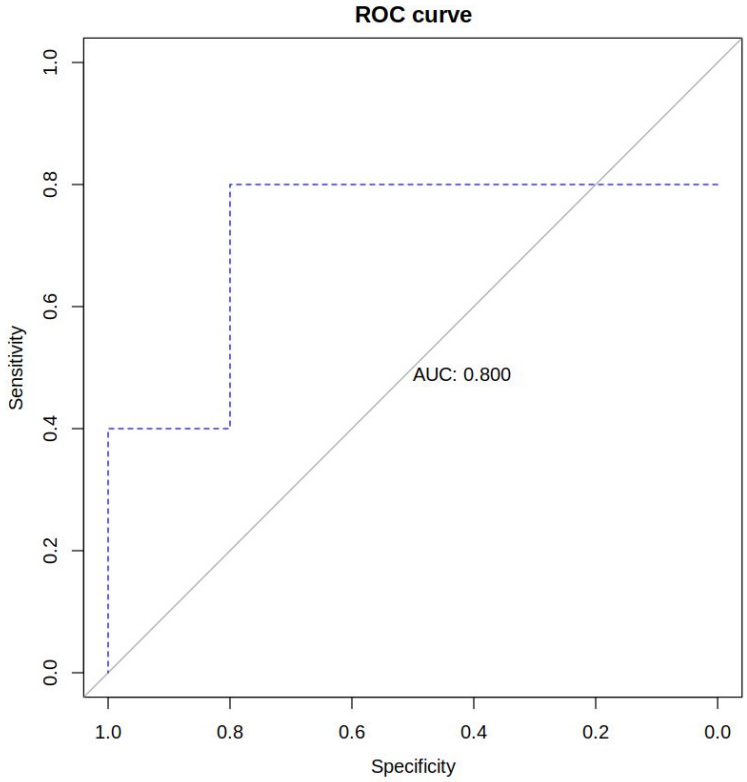
- The learned topology is

$$x_1 \longrightarrow x_2 \longrightarrow x_3$$

case	x_1	x_2	x_3
1	1	0	0
2	1	1	1
3	0	0	1
4	1	1	1
5	0	0	0
6	0	1	1
7	1	1	1
8	0	0	0
9	1	1	1
10	0	0	0

Naive ROC curve

Classifier performances

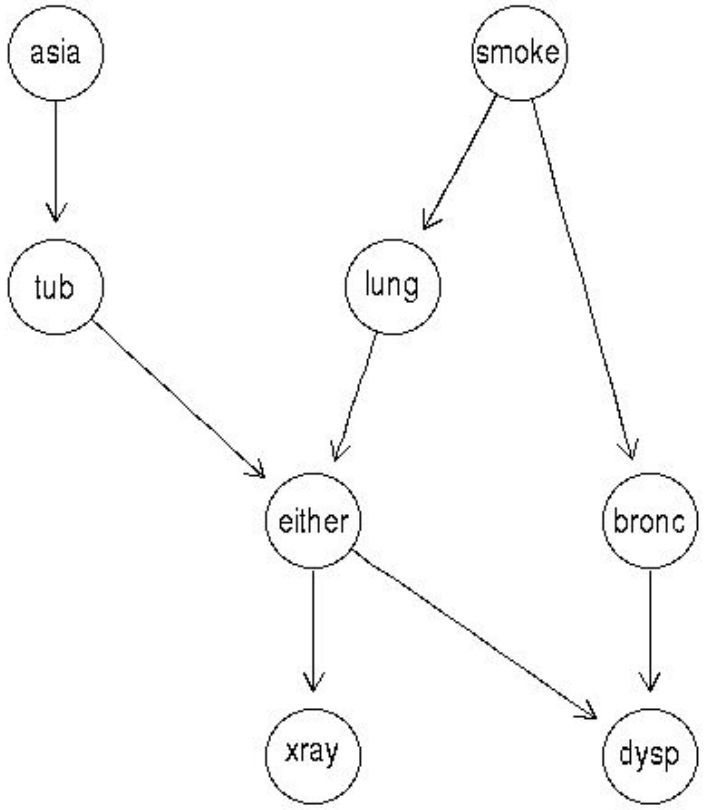


ASIA Dataset

Synthetic dataset about lung diseases (tuberculosis, lung cancer or bronchitis) and visits to Asia.

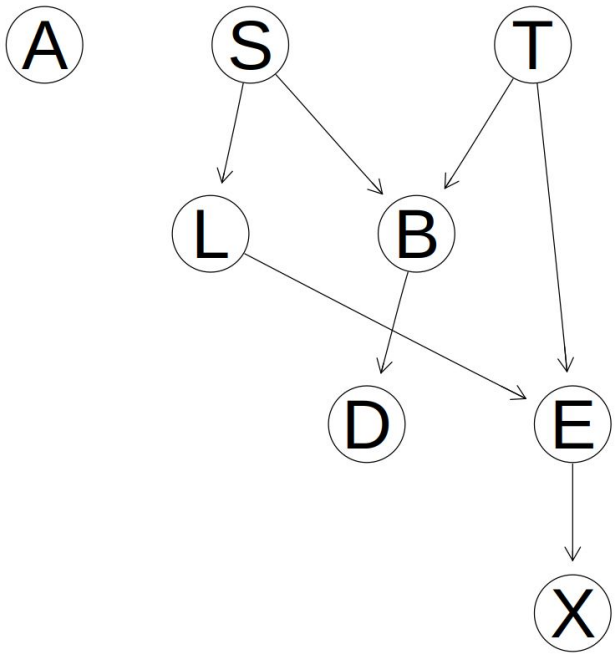
Contains 8 binary variables

- D: dyspnoea
- T: tuberculosis
- L: lung cancer
- B: bronchitis
- A: visit to Asia
- S: smoking
- X: chest X-ray
- E: tuberculosis versus lung cancer/bronchitis

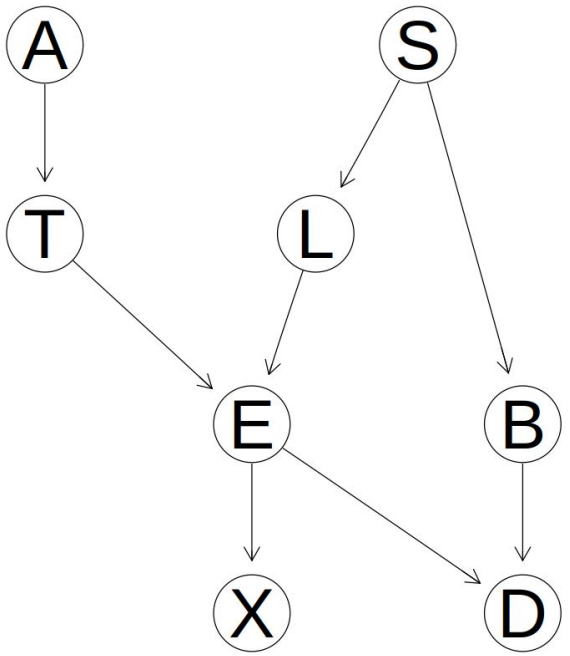


K2 over ASIA

Ordering method: Given order

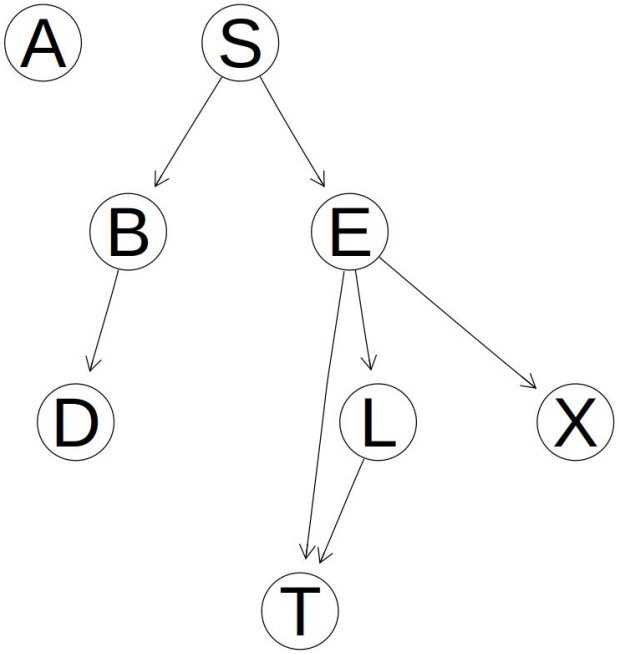


True network

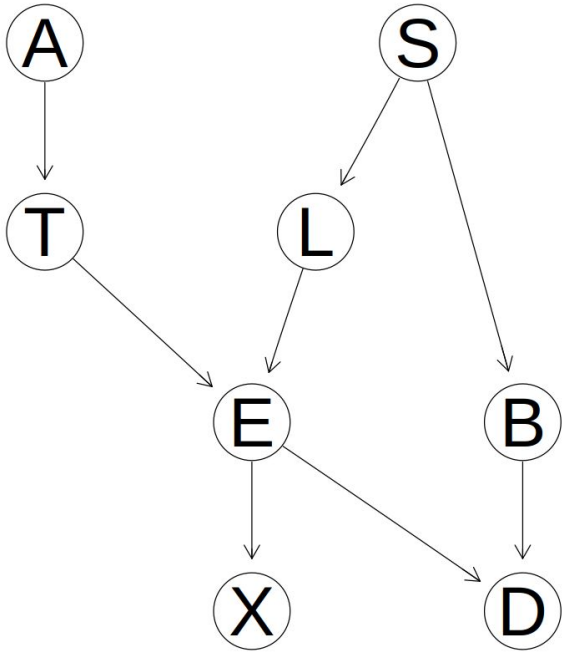


K2 over ASIA

Ordering method: Mutual Information

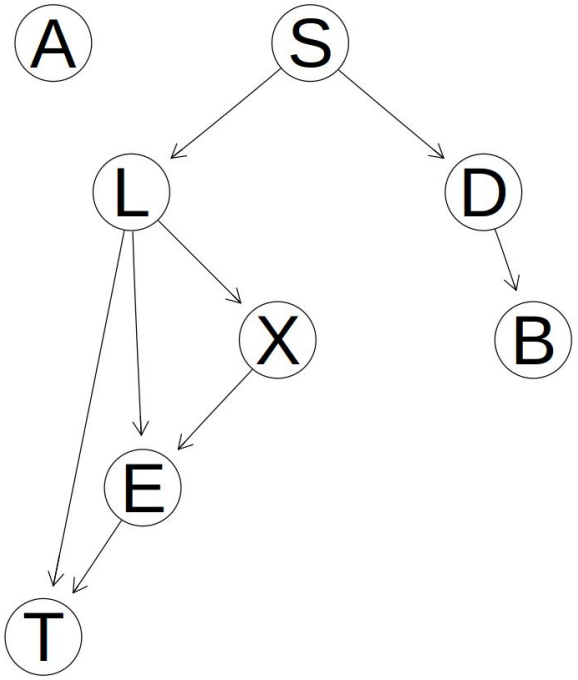


True network

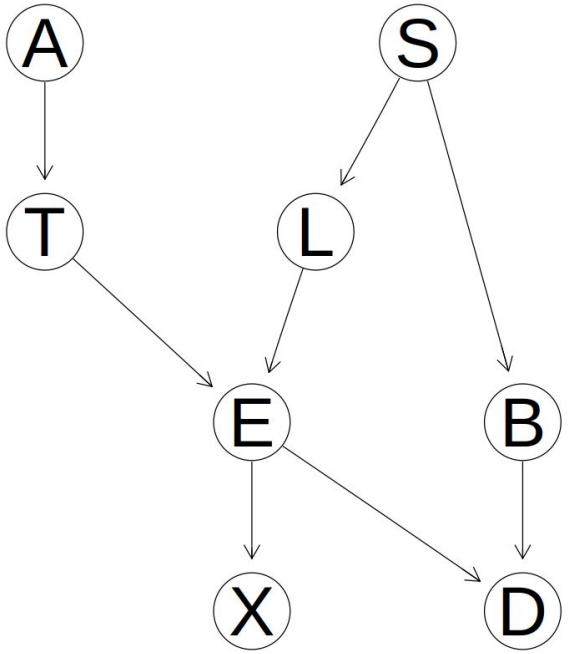


K2 over ASIA

Ordering method: FA, 3 factors

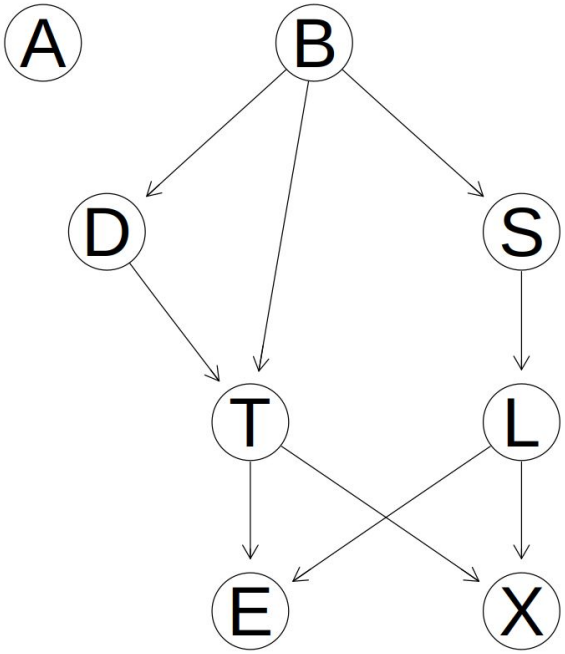


True network

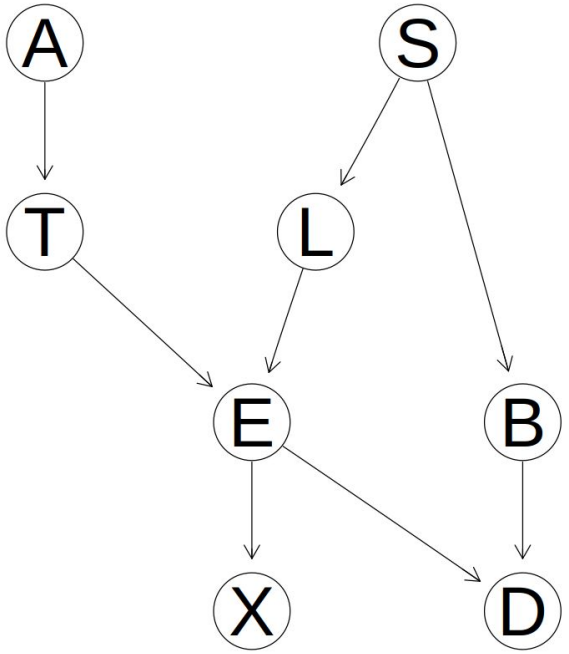


K2 over ASIA

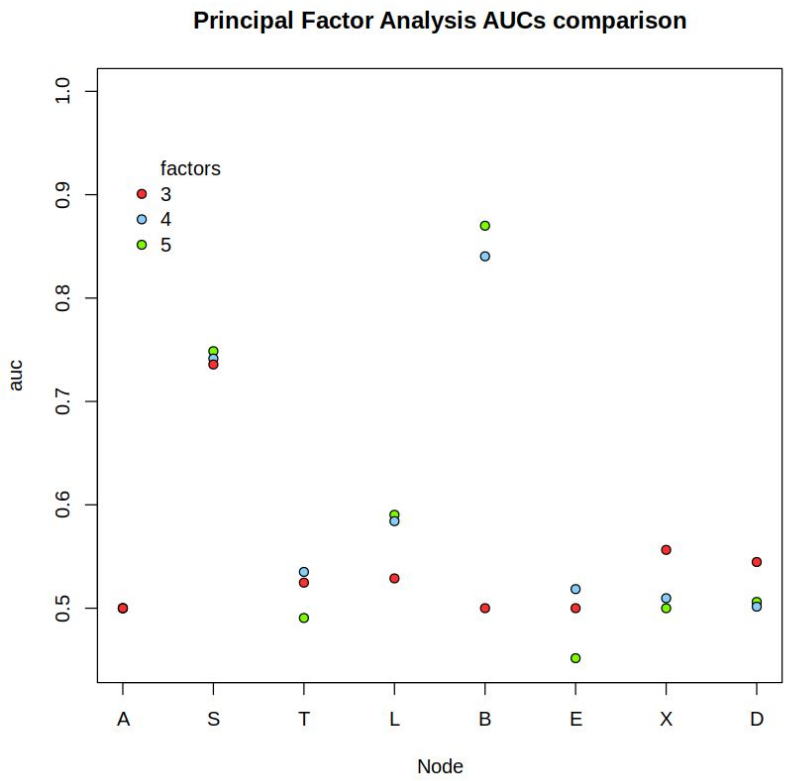
Ordering method: FA, 5 factors

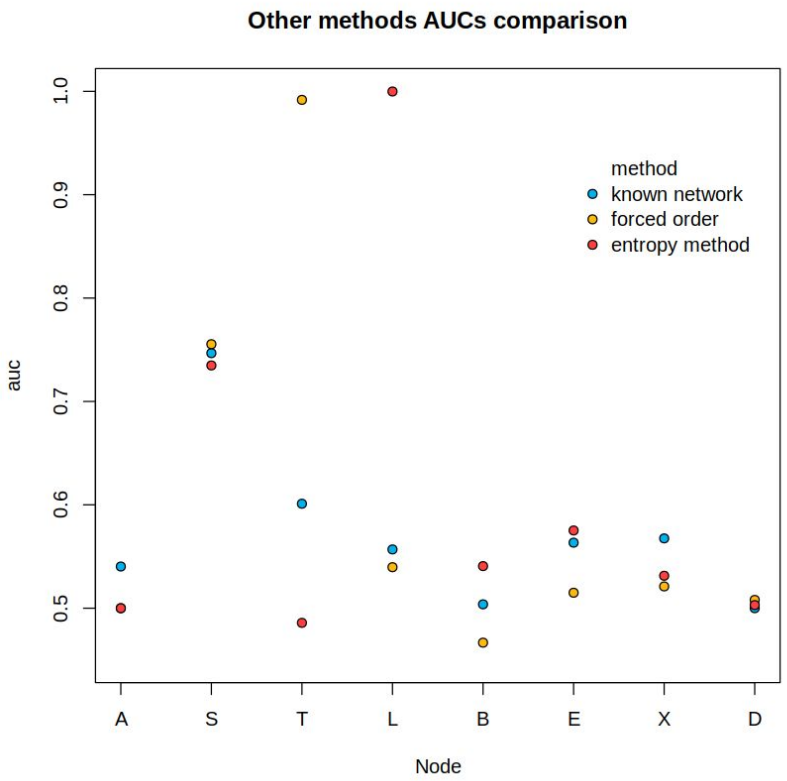


True network



AUCs comparison

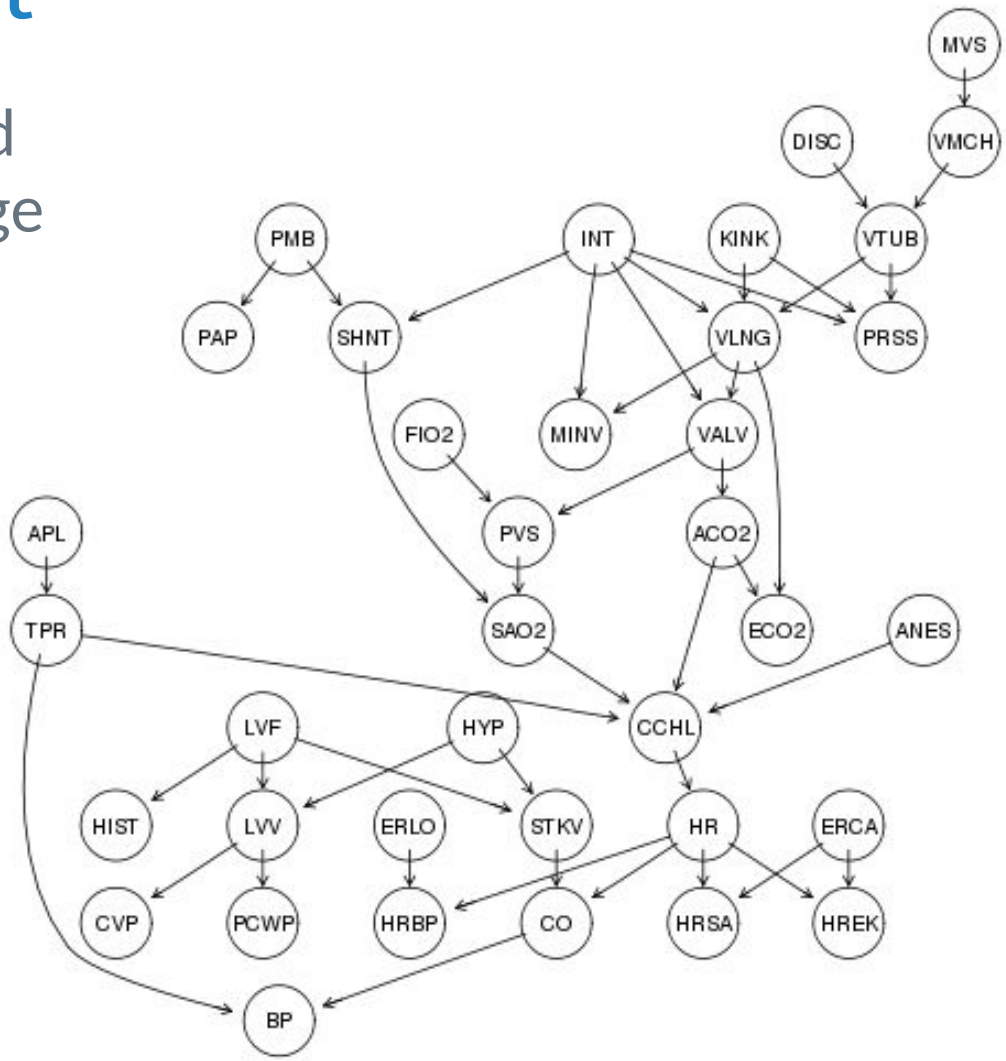




ALARM Dataset

Bayesian network designed to provide an alarm message system for patient monitoring

Contains 37 categorical variables (with 2 or more states)

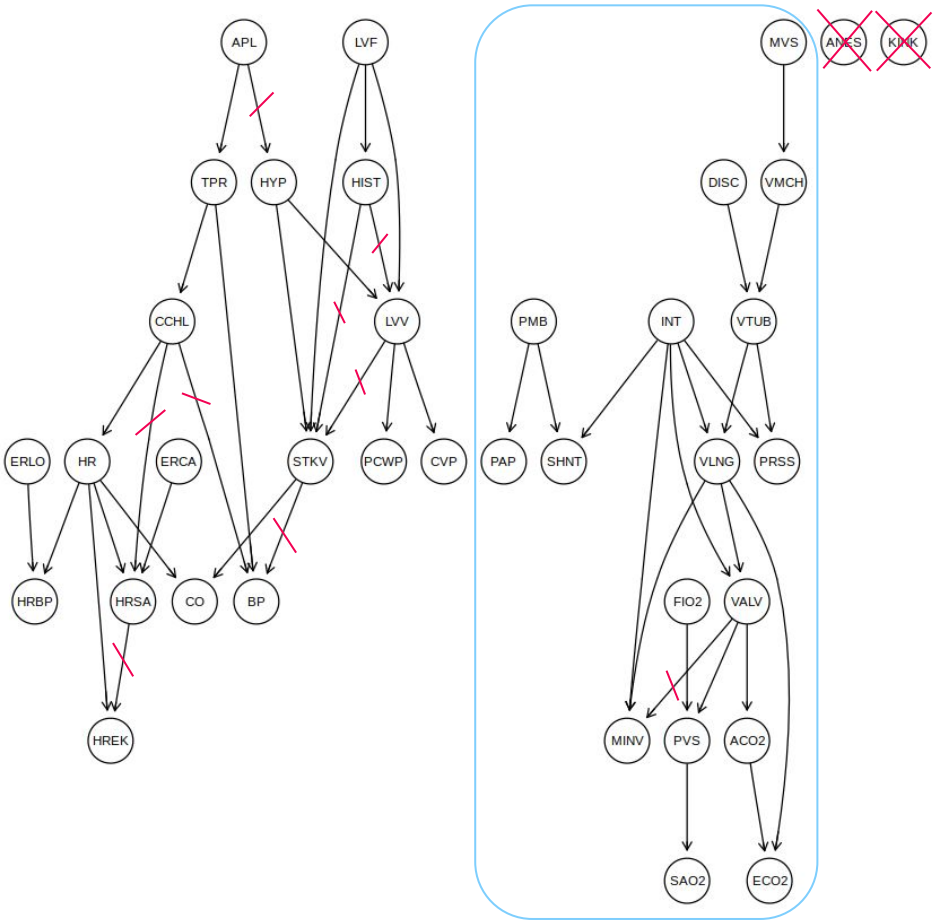


Variables Description

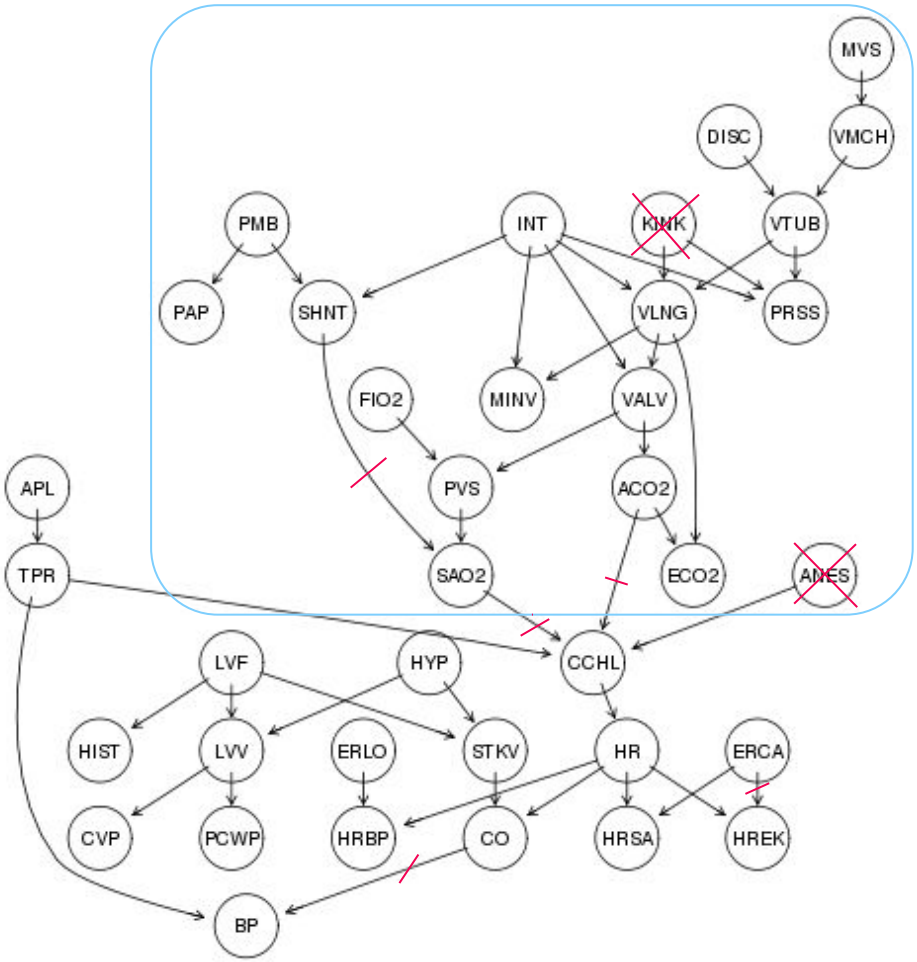
- CVP: central venous pressure.
- PCWP: pulmonary capillary wedge pressure.
- HIST: history.
- TPR: total peripheral resistance.
- BP: blood pressure.
- CO: cardiac output.
- HRBP: heart rate / blood pressure.
- HREK: heart rate measured by an EKG monitor.
- HRSA: heart rate / oxygen saturation.
- PAP: pulmonary artery pressure.
- SAO2: arterial oxygen saturation.
- FIO2: fraction of inspired oxygen.
- PRSS: breathing pressure.
- ECO2: expelled CO2.
- MINV: minimum volume.
- MVS: minimum volume set.
- HYP: hypovolemia.
- LVF: left ventricular failure.
- APL: anaphylaxis.
- ANES: insufficient anesthesia/analgesia.
- PMB: pulmonary embolus.
- INT: intubation.
- KINK: kinked tube.
- DISC: disconnection.
- LVV: left ventricular end-diastolic volume.
- STKV: stroke volume.
- CCHL: catecholamine.
- ERLO: error low output.
- HR: heart rate.
- ERA: electrocauter.
- SHNT: shunt.
- PVS: pulmonary venous oxygen saturation.
- ACO2: arterial CO2.
- VALV: pulmonary alveoli ventilation.
- VLNG: lung ventilation.
- VTUB: ventilation tube.
- VMCH: ventilation machine.

K2 over ALARM

Ordering method: Given order

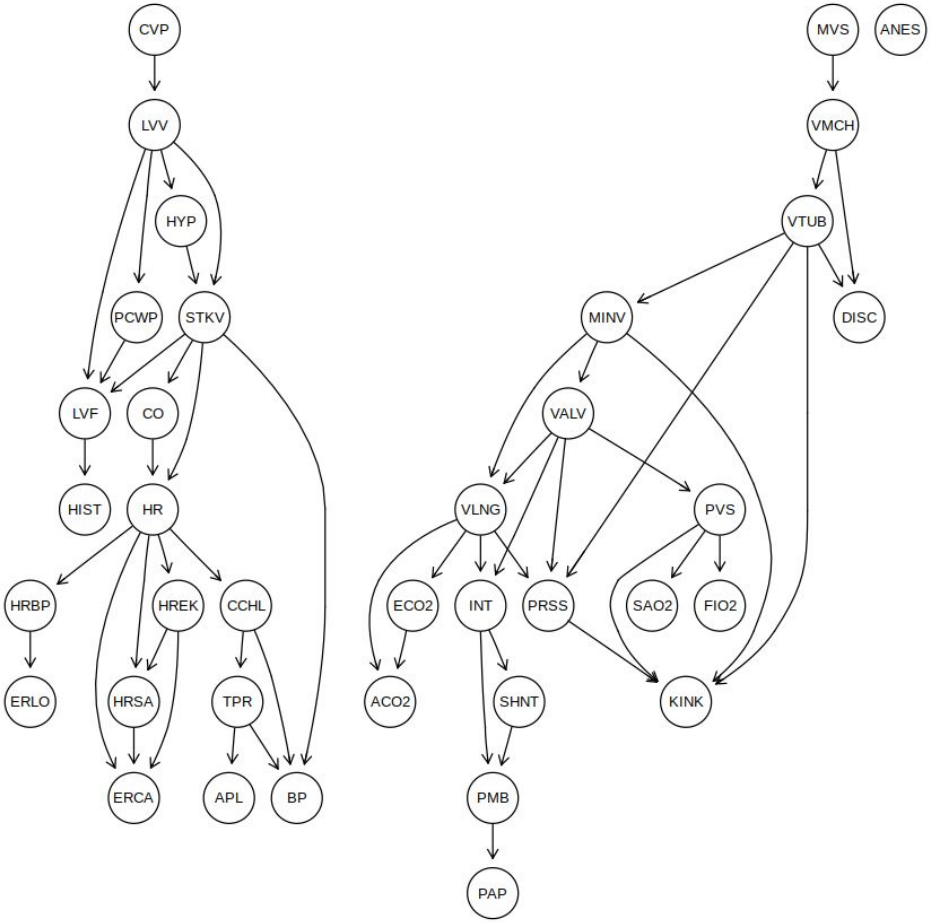


True network

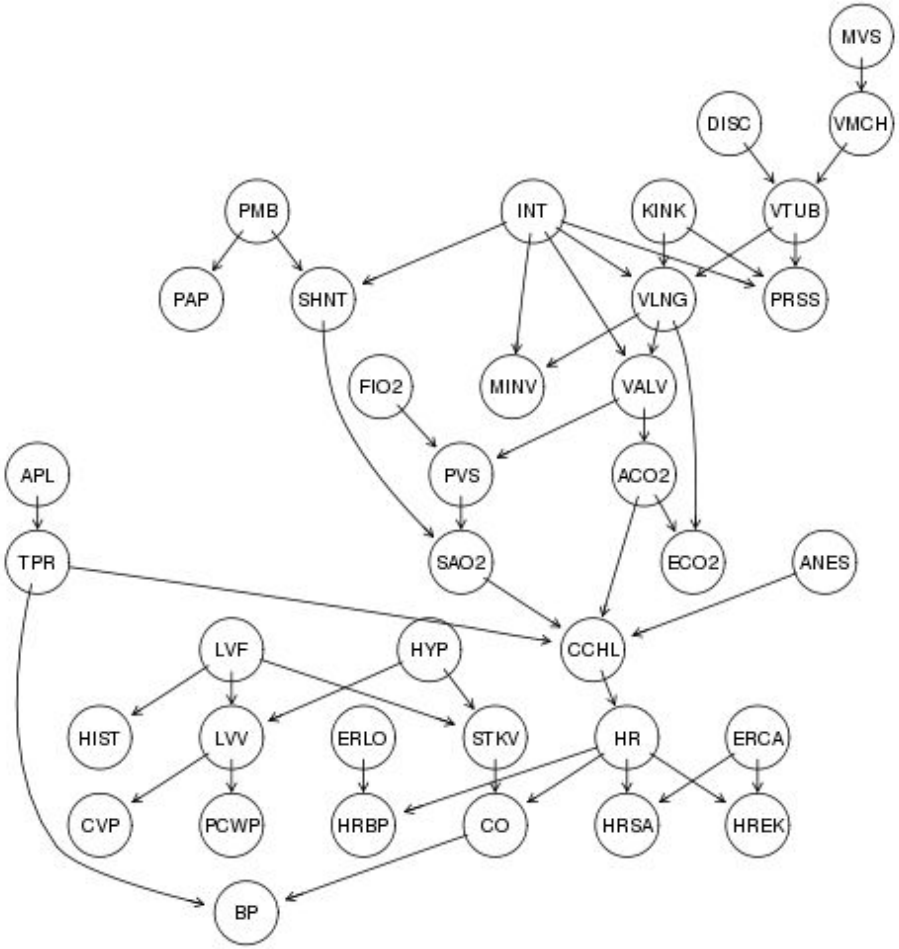


K2 over ALARM

Ordering method: Mutual Information

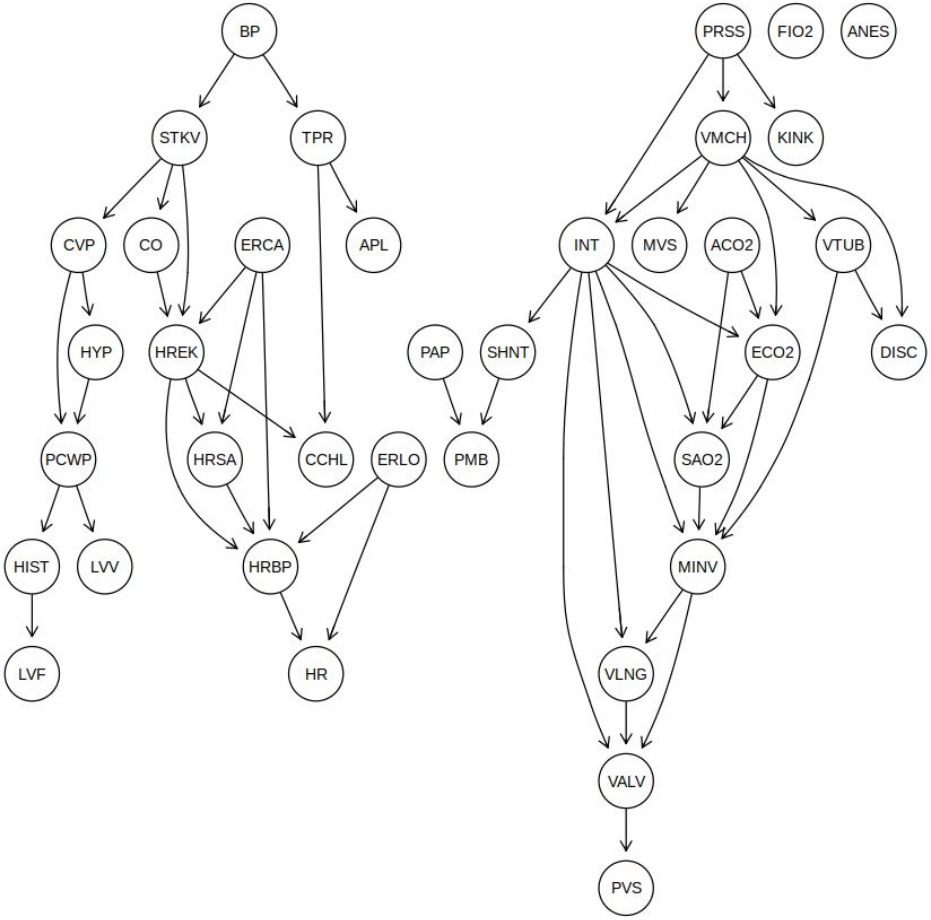


True network

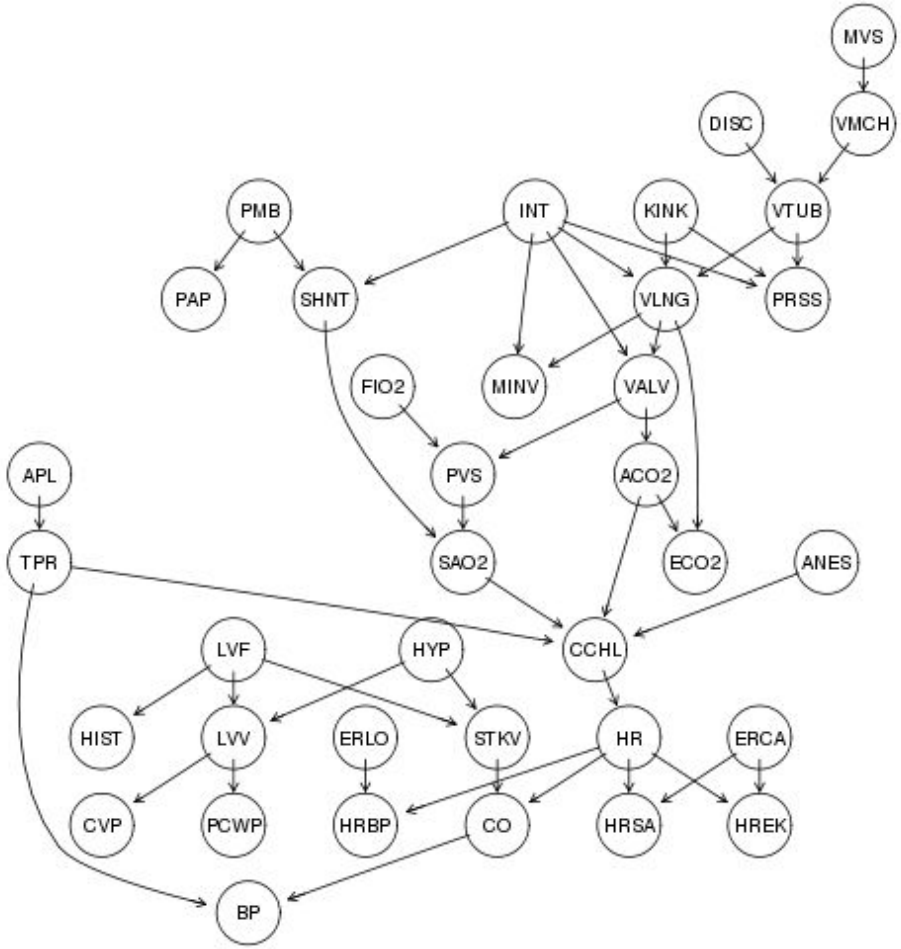


K2 over ALARM

Ordering method: FA, 12 factors



True network



Conclusions

- NP-hardness of the problem forces us to choose K2 algorithm, that solves the issue but in turn leads to suboptimal results
- To find the right ordering is not a trivial problem
- In most practical cases it's not possible to replicate the exact original network, even if a compatible order is given
- BNStruct is a valid choice to work with
- AUCs scores are an indirect measure of performance that does not evaluate the topology per se
- AUCs results are not consistent among nodes

Thanks for the attention