

# Bayesian Networks - II : Parameter and structure learning

Philippe LERAY

`philippe.leray@univ-nantes.fr`

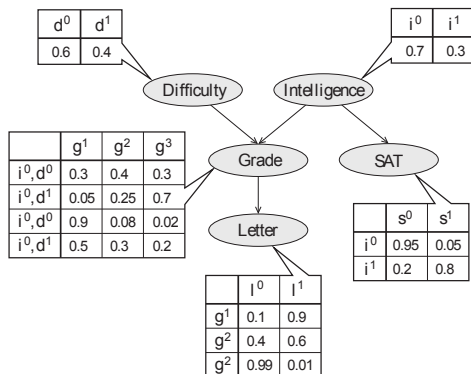
DUKe (Data User Knowledge) Research group  
Laboratoire des Sciences du Numérique de Nantes – UMR 6004  
Site de l'Ecole Polytechnique de l'université de Nantes



# One model... but two learning tasks

**BN = graph  $G$  and set of CPDs  $\Theta$**

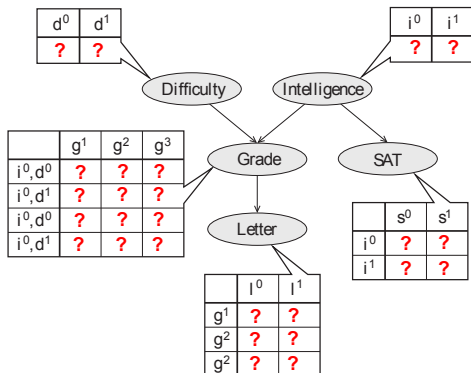
- parameter learning /  $G$  given
- structure learning



## One model... but two learning tasks

**BN = graph  $G$  and set of CPDs  $\Theta$**

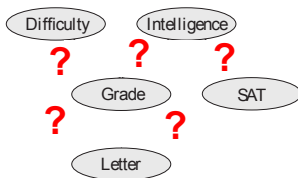
- parameter learning /  $G$  given
- structure learning



## One model... but two learning tasks

**BN = graph  $G$  and set of CPDs  $\Theta$**

- parameter learning /  $G$  given
- structure learning



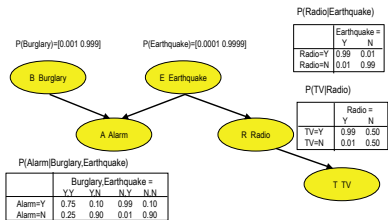
## BN dimension

### Definition

number of independent parameters needed to describe all the CPDs related to one BN.

### Examples

- $Dim(B) = 1+1+4+2+2$
- empty graph :  
 $Dim(B_0) = ?$
- completely connected graph :  $Dim(B_c) = ?$



## Knowledge acquisition

### How to determine the structure by interacting with experts ?

- Identifying variables of interest
- Controlling the model dimension : variable cardinality + intermediate variables

### How to determine the conditional distributions without data ?

- First find one cooperative expert
- Use some specific tools for determining numerical values without asking for it [Drusdel, 2001]

|            |                                                                                     |     |
|------------|-------------------------------------------------------------------------------------|-----|
| certain    |  | 100 |
| probable   |  | 85  |
| expected   |  | 75  |
| 50-50      |  | 50  |
| uncertain  |  | 25  |
| improbable |  | 15  |
| impossible |  | 0   |

## Example

## Simplify some CPDs

### Noisy OR model

[Pearl, 1986]

- $P(Y|X_1 \dots X_n)$  (boolean variables) =  $2^n$  values !
- Let suppose that
  - we can estimate each  $p_i = P(y|\bar{x}_1, \bar{x}_2, \dots, x_i, \dots, \bar{x}_n)$
  - there is no mutual effect between  $X_i$  variables and  $Y$ .
- so
  - if one  $X_i$  is true, then  $Y$  is almost true (with probability  $p_i$ )
  - is several  $X_i$  are true, then

$$P(y|\mathcal{X}) = 1 - \prod_{i/X_i \in \mathcal{X}_p} (1 - p_i)$$

where  $\mathcal{X}_p$  is the set of  $X_i$  equal to true.

## Simplify some CPDs

### Noisy OR model

[Pearl, 1986]

- extension when  $Y$  is true when no cause is present (*leaky noisy-OR gate*) [Henrion, 1989]
- extension to multi-valued variables (*generalized noisy-OR gate*)
- this simplification can also be used for learning less parameters from data...

## Example

## Parameter learning with complete data

### Parameter learning

### complete data $\mathcal{D}$

- estimation of CPD parameters from  $\mathcal{D}$
- usual statistical approach = *max. of likelihood (ML)*

$$\hat{\theta}^{ML} = \operatorname{argmax} P(\mathcal{D}|\theta)$$

- probability of an event = frequency of this event in data

### Maximum of likelihood (ML)

$$\hat{P}(X_i = x_k | Pa(X_i) = x_j) = \hat{\theta}_{i,j,k}^{ML} = \frac{N_{i,j,k}}{\sum_k N_{i,j,k}}$$

$N_{i,j,k}$  = nb of occurrences of  $\{X_i = x_k \text{ and } Pa(X_i) = x_j\}$

## Parameter learning with complete data

### Demonstration

## Parameter learning with complete data

### Other approach

- one Bayesian approach = *max. a posteriori (MAP)*

$$\hat{\theta}^{MAP} = \operatorname{argmax} P(\theta|\mathcal{D}) = \operatorname{argmax} P(\mathcal{D}|\theta)P(\theta)$$

- need to define an a priori distribution  $P(\theta)$
- *conjugated* distribution associated to  $X$  distribution
- if  $P(X)$  multinomial, associated  $P(\theta) = \text{Dirichlet}$  :

$$P(\theta) \propto \prod_{i=1}^n \prod_{j=1}^{q_i} \prod_{k=1}^{r_i} (\theta_{i,j,k})^{\alpha_{i,j,k}-1}$$

where  $\alpha_{i,j,k}$  are the coefficients of Dirichlet distribution associated to parameter  $\theta_{i,j,k}$

## Parameter learning with complete data

### Maximum a Posteriori (MAP)

$$\hat{P}(X_i = x_k | Pa(X_i) = x_j) = \hat{\theta}_{i,j,k}^{MAP} = \frac{N_{i,j,k} + \alpha_{i,j,k} - 1}{\sum_k (N_{i,j,k} + \alpha_{i,j,k} - 1)}$$

### Another bayesian approach

- *expectation a posteriori (EAP)* : Computation of the expectation of  $\theta_{i,j,k}$  instead of the max. value

$$\hat{P}(X_i = x_k | Pa(X_i) = x_j) = \hat{\theta}_{i,j,k}^{EAP} = \frac{N_{i,j,k} + \alpha_{i,j,k}}{\sum_k (N_{i,j,k} + \alpha_{i,j,k})}$$

## ML Example

$$\hat{P}(M = m_0) = 6/15 = 0.4$$

$$\hat{P}(M = m_1) = 8/15 = 0.53$$

$$\hat{P}(M = m_2) = 1/15 = 0.07$$

$$\hat{P}(F = OK|M = m_0) = 1/6 = 0.17$$

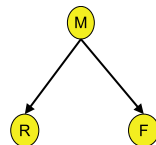
$$\hat{P}(F = BAD|M = m_0) = 5/6 = 0.83$$

etc ...

### Problem

$$\hat{P}(F = BAD|M = m_2) = 0/1$$

because this configuration is not present in the (small) dataset



| M     | F   | R |
|-------|-----|---|
| $m_0$ | BAD | O |
| $m_0$ | BAD | O |
| $m_0$ | BAD | O |
| $m_0$ | BAD | O |
| $m_0$ | BAD | N |
| $m_0$ | OK  | O |
| $m_1$ | BAD | O |
| $m_1$ | BAD | N |
| $m_1$ | OK  | O |
| $m_1$ | OK  | N |
| $m_1$ | OK  | O |
| $m_1$ | OK  | N |
| $m_1$ | OK  | O |
| $m_1$ | OK  | N |
| $m_2$ | OK  | N |

## EAP Example

- we have to give some a priori coefficients for each  $\theta_{i,j,k}$
- $\approx$  pseudo counts corresponding to  $N^*$  virtual measurements
- examples

- Dirichlet coefficients associated to  $M$   
 $= [50 \ 50 \ 0]$

$$\hat{P}(M = m_0) = (6 + 50)/(15 + 100) = 0.487$$

$$\hat{P}(M = m_1) = (8 + 50)/(15 + 100) = 0.5043$$

$$\hat{P}(M = m_2) = (1 + 0)/(15 + 100) = 0.0087$$

- Dirichlet coefficients associated to  
 $(F|M = m_i) = [9 \ 1]$

$$\hat{P}(F = BAD|M = m_2) = (0 + 1)/(1 + 10) = 0.09$$

| M     | F   | R |
|-------|-----|---|
| $m_0$ | BAD | O |
| $m_0$ | BAD | O |
| $m_0$ | BAD | O |
| $m_0$ | BAD | O |
| $m_0$ | BAD | N |
| $m_0$ | OK  | O |
| $m_1$ | BAD | O |
| $m_1$ | BAD | N |
| $m_1$ | OK  | O |
| $m_1$ | OK  | N |
| $m_1$ | OK  | O |
| $m_1$ | OK  | N |
| $m_1$ | OK  | O |
| $m_1$ | OK  | N |
| $m_2$ | OK  | N |

## Parameter learning with incomplete data

### several types of incomplete data

[Rubin, 1976]

- MCAR : *Missing Completely At Random*

- data loss = completely random phenomenon
- how to estimate ML or MAP ?
  - Complete / Available Case Analysis ...

- MAR : *Missing At Random*

- probability one data is lost depends on some known variables
- how to estimate ML or MAP ?
  - Expectation Maximisation ...

- NMAR : *Not Missing At Random*

- probability one data is lost depends on some external unknown variables
- need to improve the model by identifying and adding these variables

## Complete / Available Case Analysis

### Complete Case Analysis

- extract samples completely observed from the incomplete dataset
- advantage : we come back to complete dataset situation
- inconvenient : important missing rate  $\Rightarrow$  few complete samples

### Available Case Analysis

- principle : we don't need to observe  $C$  to estimate the CPD  $P(A|B)$  parameters
- for  $P(A|B)$  estimation, extract samples for which  $A$  and  $B$  are observed

## Complete / Available Case Analysis

### Complete Case Analysis

### Available Case Analysis

- principle : we don't need to observe  $C$  to estimate the CPD  $P(A|B)$  parameters
- for  $P(A|B)$  estimation, extract samples for which  $A$  and  $B$  are observed
- advantage : we come back to complete dataset situation

## Expectation Maximisation Algorithm

Very general algorithm

[Dempster, 1977]

- parameter estimation with incomplete data

### Principle

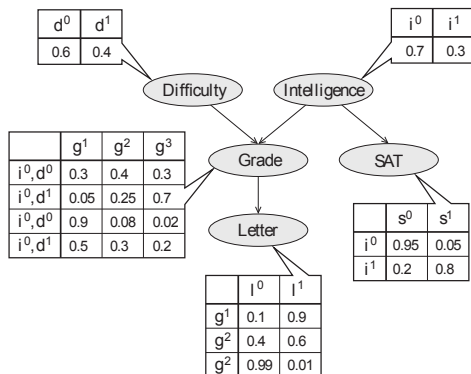
- iterative algorithm
  - parameter initialization  $\theta^{(0)}$
  - **E** estimate missing data distribution from current model  $\theta^{(t)}$ 
    - = compute  $P(X_{\text{missing}}|X_{\text{observed}})$  in current BN
    - = use probabilistic inference
  - **M** estimate again  $\theta^{(t+1)}$  from this new "completed" dataset
    - by using ML, MAP, or EAP approach

## Example

# One model... but two learning tasks

**BN = graph  $G$  and set of CPDs  $\Theta$**

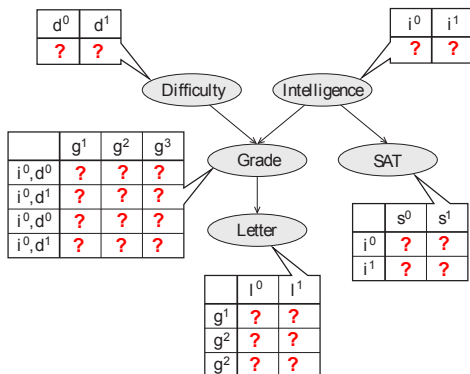
- parameter learning /  $G$  given
- structure learning



# One model... but two learning tasks

**BN = graph  $G$  and set of CPDs  $\Theta$**

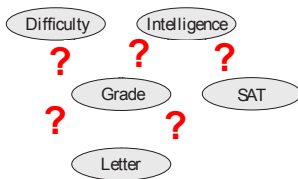
- parameter learning /  $G$  given
- structure learning



## One model... but two learning tasks

**BN = graph  $G$  and set of CPDs  $\Theta$**

- parameter learning /  $G$  given
- structure learning



## Structure learning is a complex task

**What is the number of DAGs with 3 nodes ?**

## Structure learning is a complex task

### Size of the "solution" space

- the number of possible DAGs with  $n$  variables is super-exponential w.r.t  $n$  [Robinson, 1977]

$$NS(n) = \begin{cases} 1 & n = 0 \text{ or } 1 \\ \sum_{i=1}^n (-1)^{i+1} \binom{n}{i} 2^{i(n-1)} NS(n-i), & n > 1 \end{cases}$$

$$NS(4) = 543$$

$$NS(5) = 29281$$

$$NS(10) = 4.2 \times 10^{18}$$

- an exhaustive search is impossible !

One thousand millenniums =  $3.2 \times 10^{13}$  seconds

## Score-based methods

### How to search a good BN ?

- constraint-based methods  
BN = independence model  
⇒ find CI in data in order to build the DAG
- score-based methods  
BN = probabilistic model that must fit data as well as possible  
⇒ search the DAG space in order to maximize a scoring/fitness function
- hybrid methods

## Notion of score

### General principle : Occam razor

- *Pluralitas non est ponenda sine neccesitate*  
plurality should not be posited without necessity
- *Frustra fit per plura quod potest fieri per pauciora*  
It is pointless to do with more what can be done with fewer

### = Parcimony principle : find a model

- fitting the data  $\mathcal{D}$  : likelihood :  $L(\mathcal{D}|\theta, B)$
- the simplest possible : dimension of  $B$  :  $Dim(B)$

## Score examples

### AIC and BIC

- compromise between likelihood and complexity
- application of AIC [Akaike, 1970] and BIC [Schwartz, 1978]

$$S_{AIC}(B, \mathcal{D}) = \log L(\mathcal{D} | \theta^{MV}, B) - \text{Dim}(B)$$

$$S_{BIC}(B, \mathcal{D}) = \log L(\mathcal{D} | \theta^{MV}, B) - \frac{1}{2} \text{Dim}(B) \log N$$

### Bayesian scores : BD, BDe, BDeu

- $S_{BD}(B, \mathcal{D}) = P(B, \mathcal{D})$  [Cooper et Herskovits, 1992]
- $BDe = BD + \text{score equivalence}$  [Heckerman, 1994]

$$S_{BD}(B, \mathcal{D}) = P(B) \prod_{i=1}^n \prod_{j=1}^{q_i} \frac{\Gamma(\alpha_{ij})}{\Gamma(N_{ij} + \alpha_{ij})} \prod_{k=1}^{r_i} \frac{\Gamma(N_{ijk} + \alpha_{ijk})}{\Gamma(\alpha_{ijk})}$$

## Score properties

Two important properties

### Decomposability

$$(Global)Score(B, \mathcal{D}) = \sum_{i=1}^n (local)score(X_i, pa_i)$$

### Score equivalence

If two BN  $B_1$  and  $B_2$  are Markov equivalent then  
 $S(B_1, \mathcal{D}) = S(B_2, \mathcal{D})$

## Example

### Empty graph $G_0$

- estimate CPDs  $P(X_1)$  and  $P(X_2)$  (with ML approach)
- log-likelihood of  $G_0$  ? AIC score ?

### Complete graph $G_c$

- estimate CPDs  $P(X_1)$  and  $P(X_2|X_1)$  (with ML approach)
- log-likelihood of  $G_c$  ? AIC score ?

| $X_1$ | $X_2$ |
|-------|-------|
| Y     | N     |
| Y     | Y     |
| Y     | Y     |
| Y     | Y     |
| N     | N     |
| N     | Y     |
| N     | Y     |
| N     | Y     |

What is the best graph according to likelihood ?  
according to AIC ?

## Heuristic exploration of search space

### Search space and heuristics

- space  $\mathcal{B}$ 
  - restriction to tree space : Chow&Liu, MWST
  - DAG with node ordering : K2 algorithm
  - greedy search
  - genetic algorithms, ...
- space  $\mathcal{E}$ 
  - greedy equivalence search

## Restriction to tree space

### Principle

- what is the best tree connecting all the nodes,  
i.e. maximizing a weight defined for each possible edge ?

### Answer : maximal weighted spanning tree (MWST)

- [Chow and Liu, 1968] : weight = mutual information :

$$W(X_A, X_B) = \sum_{a,b} \frac{N_{ab}}{N} \log \frac{N_{ab}N}{N_a \cdot N_{\cdot b}}$$

- [Heckerman, 1994] : any local score local :

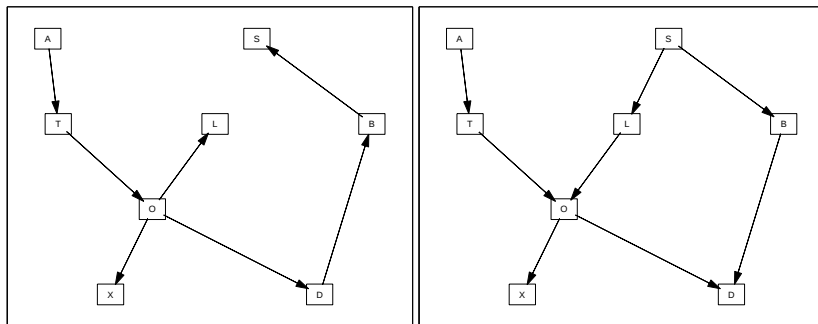
$$W(X_A, X_B) = \text{score}(X_A, Pa(X_A) = X_B) - \text{score}(X_A, \emptyset)$$

## Restriction to tree space

### Remarks

- MWST returns an undirected tree
- this undirected tree = CPDAG of all the directed tree with this skeleton
- obtain a directed tree by (randomly) choosing one root and orienting the edges with a depth first search over this tree

## Example : obtained DAG vs. target one



MWST can not discover cycles neither V-structures (tree space !)

## Example

local score  $s(X_i; Pa_i = X_j)$

| $X_i \setminus Pa_i$ | $X_1$ | $X_2$ | $X_3$ | $X_4$ | $X_5$ |
|----------------------|-------|-------|-------|-------|-------|
| $X_1$                | 5     | 15    | 13    | 7     | 10    |
| $X_2$                | 16    | 6     | 13    | 7     | 10    |
| $X_3$                | 11    | 10    | 3     | 6     | 9     |
| $X_4$                | 6     | 5     | 7     | 4     | 4.5   |
| $X_5$                | 15    | 14    | 16    | 10.5  | 10    |

diagonal contains  $s(X_i; Pa_i = \emptyset)$ .

What is the corresponding MWST and its score ?

## Heuristic exploration of search space

### Search space and heuristics

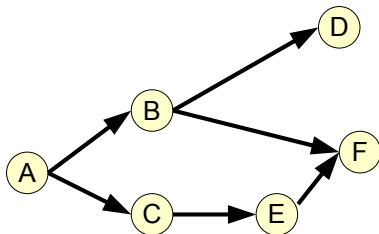
- space  $\mathcal{B}$ 
  - restriction to tree space : Chow&Liu, MWST
  - DAG with node ordering : K2 algorithm
  - greedy search
  - genetic algorithms, ...
- space  $\mathcal{E}$ 
  - greedy equivalence search

## Greedy search

### Principle

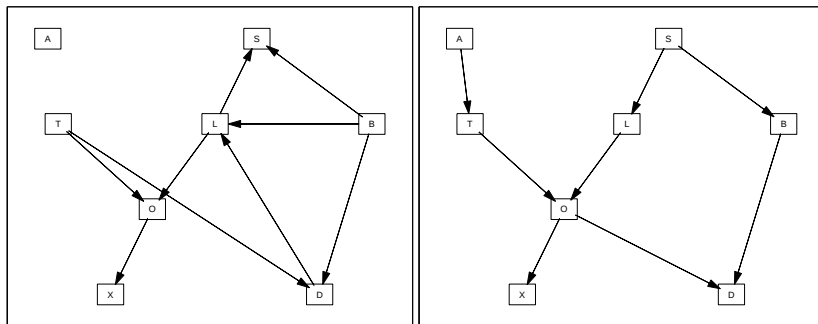
- exploration of the search space with traversal operators
  - add edge
  - invert edge
  - delete edge
- and respect the DAG definition (no cycle)
- exploration can begin from any given DAG

## Example



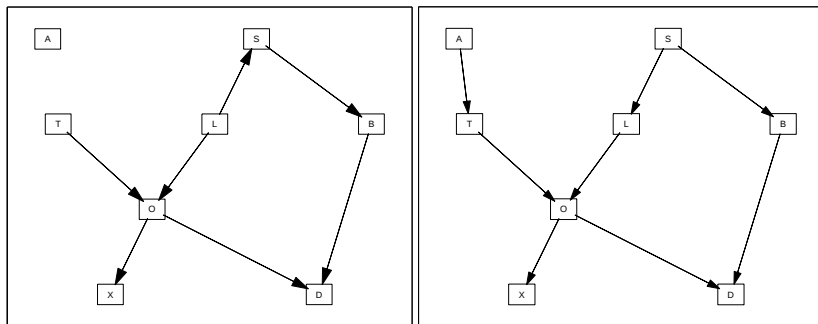
- global score decomposition of this graph ?
- neighborhood ?
- local score computed to evaluate each neighbor ?

## Example : obtained DAG vs. target one



start = empty graph. GS result = local optimum :-)

## Example : obtained DAG vs. target one



start = MWST result. GS result is better

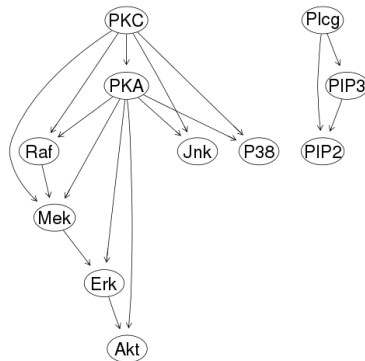
## Exercise (Genie)

### Benchmark model

- [Sachs et al., 2005]
- Causal Protein-signaling network
- 11 nodes, 17 arcs

### Data generation

- generate several datasets of 500 samples (resp. 1500, 2500, ...) (Network / Generate Data File)



## Exercise (Genie)

### Structure Learning - score based method

- open one of the previous datasets
- run Greedy Thick Thinning (like a Greedy search) algorithm
- what is the difference with PC algorithm ?
- compare the resulting BN to the initial one
- repeat these steps for all your datasets

### How to evaluate the learning algorithm ?

- editing distance between graphs ?
- how to take into account Markov equivalence ?
- ... ?