Learning the Network Structure from Data

Learning the Network Structure from Data

Part 1 Relational Networks

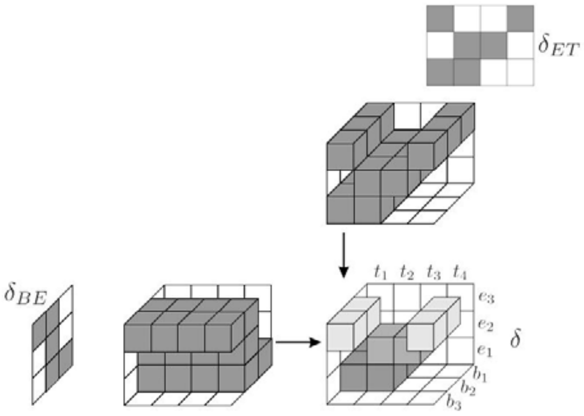
Part 1 Structure Learning for the Relational Case

Given a data set D. The elements of the data set define a subset R of the (unknown) relation S consisting of all possible tuples.

The learning task is to choose a graph G which represents a (perfect or approximate) decomposition of R.

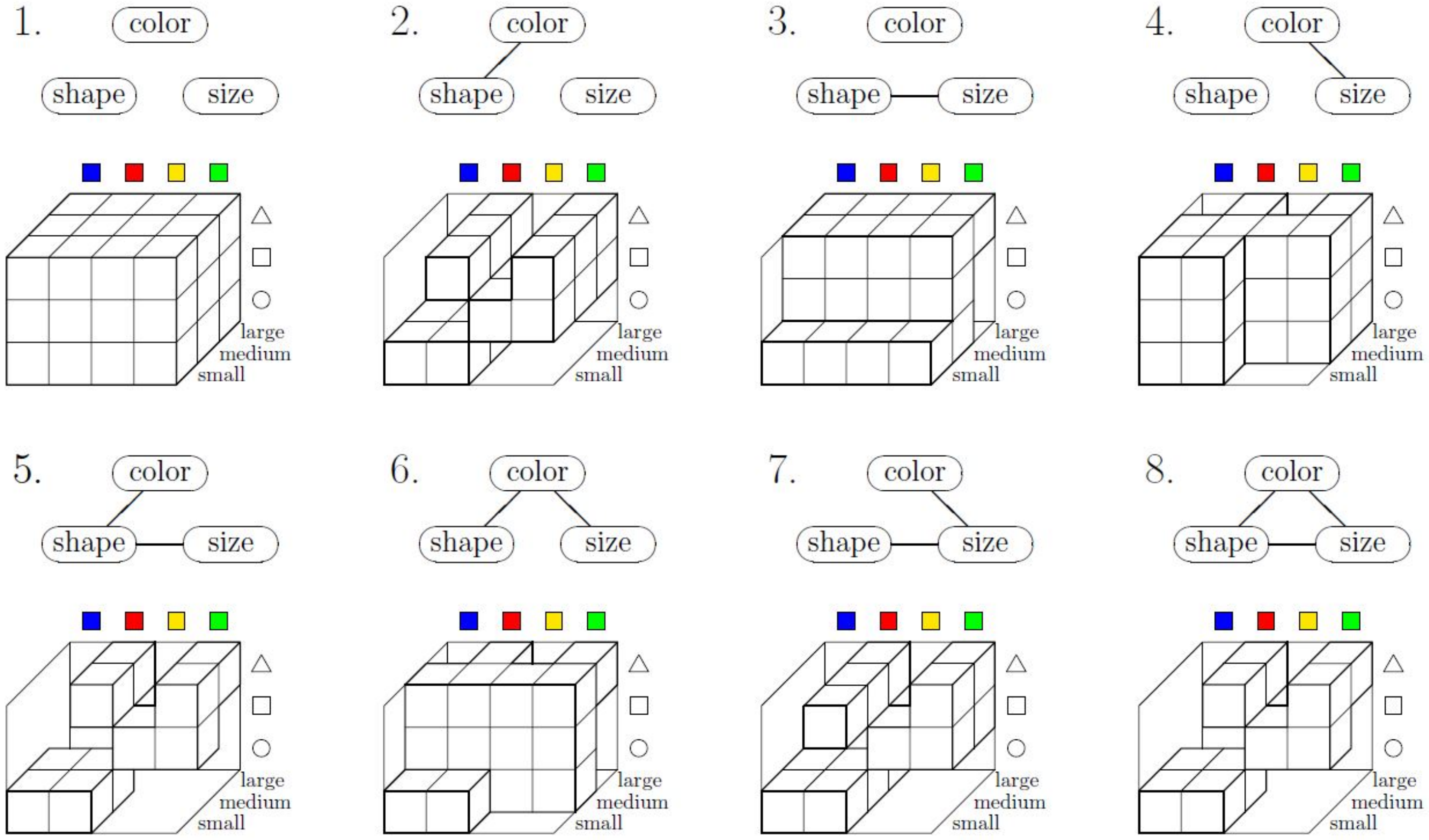
Each graph G defines conditional or marginal relations of R. In the perfect case the relation R can be represented as the intersection of the cylindrical extension of the projections of R to the relevant subspaces defined by the graph.

Example Let δ be a relation in BxExT and
G the undirected graph B - E – T .
There is a (perfect) decomposition.



How to find a „good“ decomposition?

Example (Test all Decompositions)



Graph No 5 gives a perfect decomposition of R. In most cases there is no perfect decomposition, for large numbers of attributes the task is extremely complex. That’s why (depending on the application) finding an approximate solution is the aim.

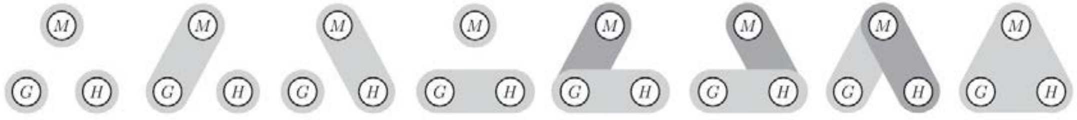
Structure Learning for the Relational Case

Most **learning algorithm** for relational (possibilistic) graphical models consists of an **evaluation measure**, by which a candidate model is assessed, and a (heuristic) **search method**, which determines the candidate models to be inspected.

1. Test whether a distribution is decomposable w. r. t. a given graph:
The most direct approach, very complex for higher dimensions.

An exhaustive search over all graphs is too expensive:

- ○ $2^{\binom{n}{2}}$ possible undirected graphs for n attributes.
- ○ $f(n) = \sum_{i=1}^{n} (-1)^{i+1} \binom{n}{i} 2^{i(n-i)} f(n-i)$ possible directed acyclic graphs.



8 possible undirected graphs with 3 nodes

2. Find a suitable graph by measuring the strength of dependences

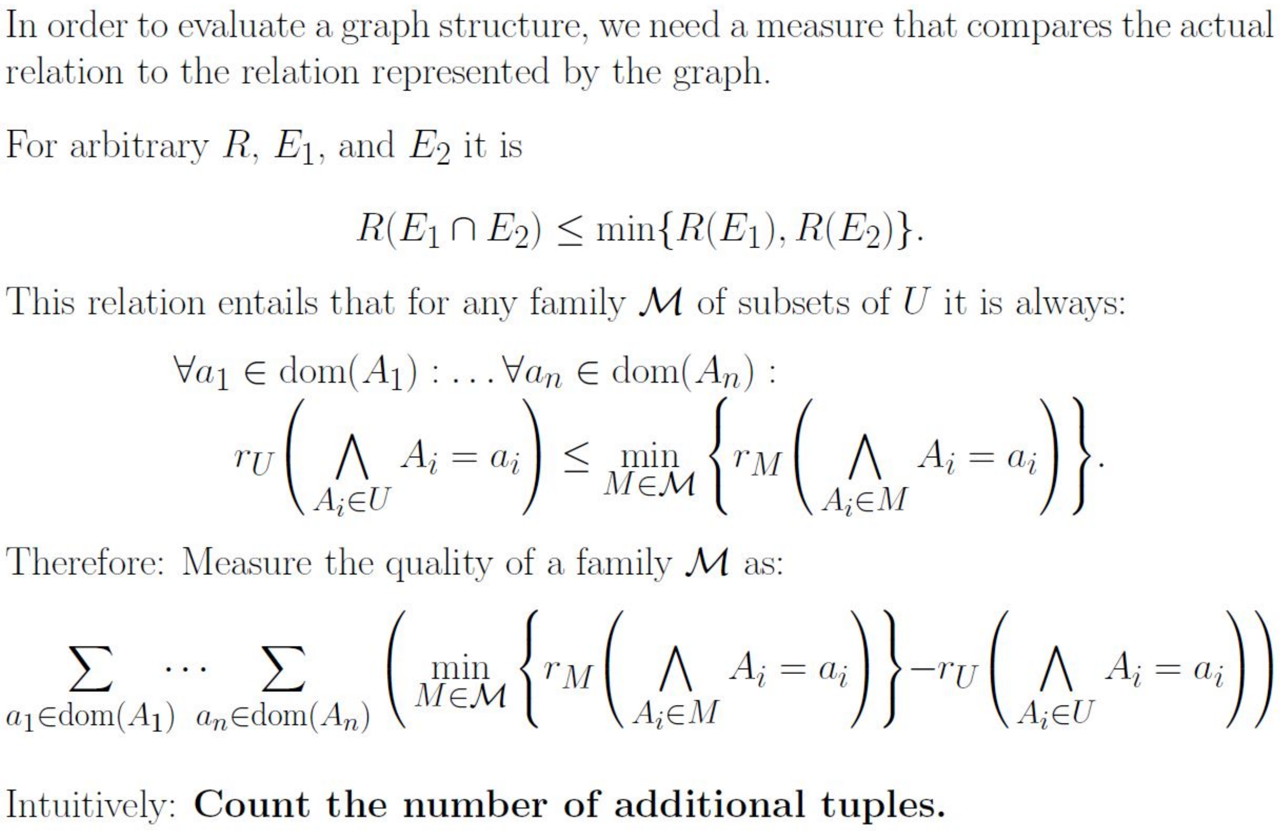
A heuristic, but often highly successful

3. Find an independence map by conditional independence tests

Based on the connection of conditional independence graphs and graphs that represent decompositions.

Method 1 Testing for Decomposability by Comparing Relations

Evaluation of Candidates



In order to evaluate a graph structure, we need a measure that compares the actual relation to the relation represented by the graph.

For arbitrary R, E_1, and E_2 it is

$$R(E_1 \cap E_2) \leq \min\{R(E_1), R(E_2)\}.$$

This relation entails that for any family $\mathcal{M}$ of subsets of U it is always:

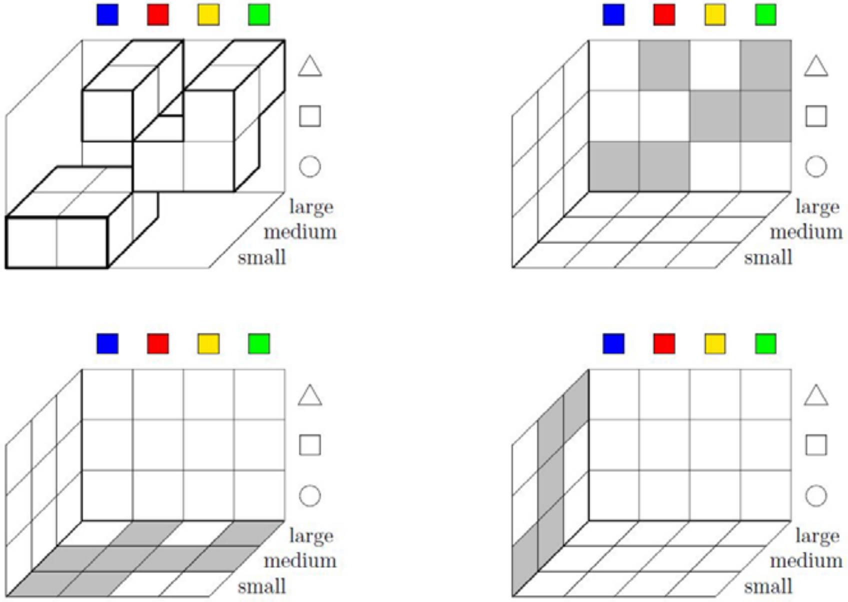
$$\forall a_1 \in \mathrm{dom}(A_1) : \ldots \forall a_n \in \mathrm{dom}(A_n) :$$
$$r_U\left(\bigwedge_{A_i \in U} A_i = a_i\right) \leq \min_{M \in \mathcal{M}} \left\{ r_M\left(\bigwedge_{A_i \in M} A_i = a_i\right)\right\}.$$

Therefore: Measure the quality of a family $\mathcal{M}$ as:

$$\sum_{a_1 \in \mathrm{dom}(A_1)} \cdots \sum_{a_n \in \mathrm{dom}(A_n)} \left(\min_{M \in \mathcal{M}} \left\{ r_M\left(\bigwedge_{A_i \in M} A_i = a_i\right)\right\} - r_U\left(\bigwedge_{A_i \in U} A_i = a_i\right)\right)$$

Intuitively: **Count the number of additional tuples.**

Method 2 Strength of Marginal Dependences



subspace	color × shape	shape × size	size × color
possible combinations	12	9	12
occurring combinations	6	5	8
relative number	50%	56%	67%

The relational network can be obtained by interpreting the relative numbers as edge weights and constructing the minimum weight spanning tree.

Method 2 Strength of Marginal Dependences: Hartley Information

Definition: Let A be an attribute and R a discrete possibility measure with $\exists a \in \mathrm{dom}(A) : R(A = a) = 1$. Then

$$(\mathrm{Hartley})(A) \quad = \quad \log_2 \left(\textstyle\sum_{a \in \mathrm{dom}(A)} R(A = a) \right)$$

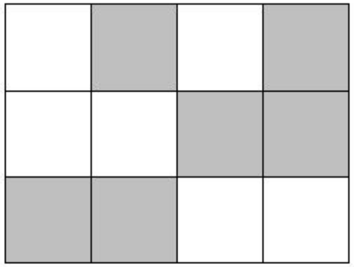
is called the **Hartley information** of A w.r.t. R.

Method 2 Strength of Marginal Dependences: Comparing Local Relations

Definition: Let A and B be two attributes and R a discrete possibility measure with $\exists a \in \mathrm{dom}(A) : \exists b \in \mathrm{dom}(B) : R(A = a, B = b) = 1$. Then

$$
\begin{aligned}
I_{\text{gain}}^{(\text{Hartley})}(A, B) &= \log_2\left(\sum_{a\in\mathrm{dom}(A)} R(A = a)\right) + \log_2\left(\sum_{b\in\mathrm{dom}(B)} R(B = b)\right) \\
&\quad - \log_2\left(\sum_{a\in\mathrm{dom}(A)}\ \sum_{b\in\mathrm{dom}(B)} R(A = a, B = b)\right) \\
&= \log_2 \frac{\left(\sum_{a\in\mathrm{dom}(A)} R(A = a)\right) \cdot \left(\sum_{b\in\mathrm{dom}(B)} R(B = b)\right)}{\sum_{a\in\mathrm{dom}(A)} \sum_{b\in\mathrm{dom}(B)} R(A = a, B = b)},
\end{aligned}
$$

is called the **Hartley information gain** of A and B w.r.t. R.



Hartley information needed to determine

coordinates:	$\log_2 4 + \log_2 3 = \log_2 12$	≈ 3.58
coordinate pair:	$\log_2 6$	≈ 2.58
gain:	$\log_2 12 - \log_2 6 = \log_2 2 = 1$	

Method 2 Strength of Marginal Dependences: Comparing Local Relations

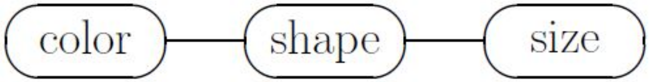
The graph can be obtained by interpreting the relative numbers as edge weights and constructing the minimum weight spanning tree.

Intuitive interpretation of Hartley information gain:
The binary logarithm measures the number of questions to find the obtaining value with a scheme like a binary search. Thus Hartley information gain measures the reduction in the number of necessary questions.

Results for the simple example:

$$\begin{aligned} I_{\text{gain}}^{(\text{Hartley})}(\text{color}, \text{shape}) &= 1.00 \text{ bit} \\ I_{\text{gain}}^{(\text{Hartley})}(\text{shape}, \text{size}) &\approx 0.86 \text{ bit} \\ I_{\text{gain}}^{(\text{Hartley})}(\text{color}, \text{size}) &\approx 0.58 \text{ bit} \end{aligned}$$

Applying the Kruskal algorithm yields as a learning result:



As we know, this graph describes indeed a decomposition of the relation.

Method 2 Strength of Marginal Dependences: Comparing Local Relations

Kruskal’s algorithm is a greedy algorithm in graph theory, which finds a minimum spanning tree for a connected weighted graph.

A minimum spanning tree consists of a subset of edges that forms a tree, which includes every vertex, where the total weight of all the edges in the tree is minimized.

Algorithm:

1. Create a forest (a set of trees) F, where each vertex in the graph is a separate tree
2. Create a set S containing all the edges in the graph
3. While S is nonempty and F is not yet spanning:
   (a) remove an edge with minimum weight from S
   (b) if the removed edge connects two different trees then add it to the forest F, combining two trees into a single tree

At the termination of the algorithm, the forest forms a minimum spanning tree of the graph. If the graph is connected, the forest has a single component and forms a minimum spanning tree.

https://de.wikipedia.org/wiki/Algorithmus_von_Kruskal#/media/Datei:KruskalDemo.gif

Method 3 Find an Independence Map using Conditional Independence Tests

General Idea

Exploit the theorems that connect conditional independence graphs and graphs that represent decompositions: We want a graph describing a decomposition, but we search for a conditional independence graph.

This approach has the advantage that a single conditional independence test, if it fails, can exclude several candidate graphs.

Assumptions

Faithfulness: The domain under consideration can be accurately described with a graphical model (more precisely: there exists a perfect map).
Reliability of Tests: The result of all conditional independence tests coincides with the actual situation in the underlying distribution.

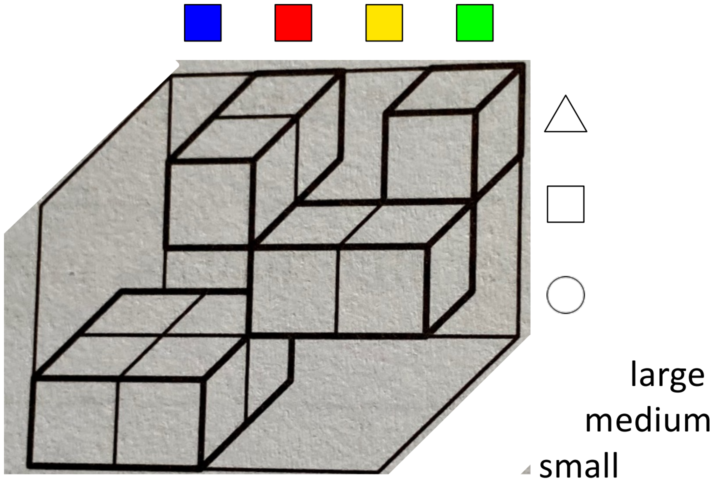
Algorithm

In order to test for (approximate conditional independence)

- Compute the Hartley Information Gain for each possible instantiation of the conditioning attributes
- Aggregate the results over all possible instantiations , e.g. bei averaging them

Then use Kruskal‘s algorithm

Method 3 Conditional Independence Tests: Simple Example

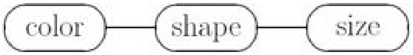


color	Hartley information gain	
🟦	$\log_2 1 + \log_2 2 - \log_2 2 = 0$	
🟥	$\log_2 2 + \log_2 3 - \log_2 4 \approx 0.58$	
🟨	$\log_2 1 + \log_2 1 - \log_2 1 = 0$	
🟩	$\log_2 2 + \log_2 2 - \log_2 2 = 1$	
	average:	≈ 0.40

shape	Hartley information gain	
○	$\log_2 2 + \log_2 2 - \log_2 4 = 0$	
□	$\log_2 2 + \log_2 1 - \log_2 2 = 0$	
△	$\log_2 2 + \log_2 2 - \log_2 3 \approx 0.42$	
	average:	$\approx$ 0.14

size	Hartley information gain	
large	$\log_2 2 + \log_2 1 - \log_2 2 = 0$	
medium	$\log_2 4 + \log_2 3 - \log_2 5 \approx 1.26$	
small	$\log_2 2 + \log_2 1 - \log_2 2 = 0$	
	average:	≈ 0.42

None of the averages is zero, neither $B \perp\!\!\!\perp C \mid A$, $A \perp\!\!\!\perp C \mid B$, $A \perp\!\!\!\perp B \mid C$ hold. A and C show a „weak“ conditional dependence given B, so we may decide to treat them as conditional independent:



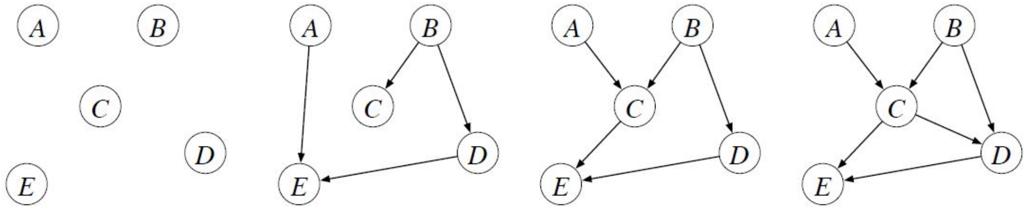
Note: In the above relation (in comparison to our standard example) one tuple is removed

Learning the Network Structure from Data

Part 2 Bayesian Networks

Part 2: Structure Learning in Bayesian Nets

Given a data set, the learning task is to choose a DAG G that indicates, which conditional or marginal distributions constitute the represented decomposition.



There are several families of methods, most commonly used are

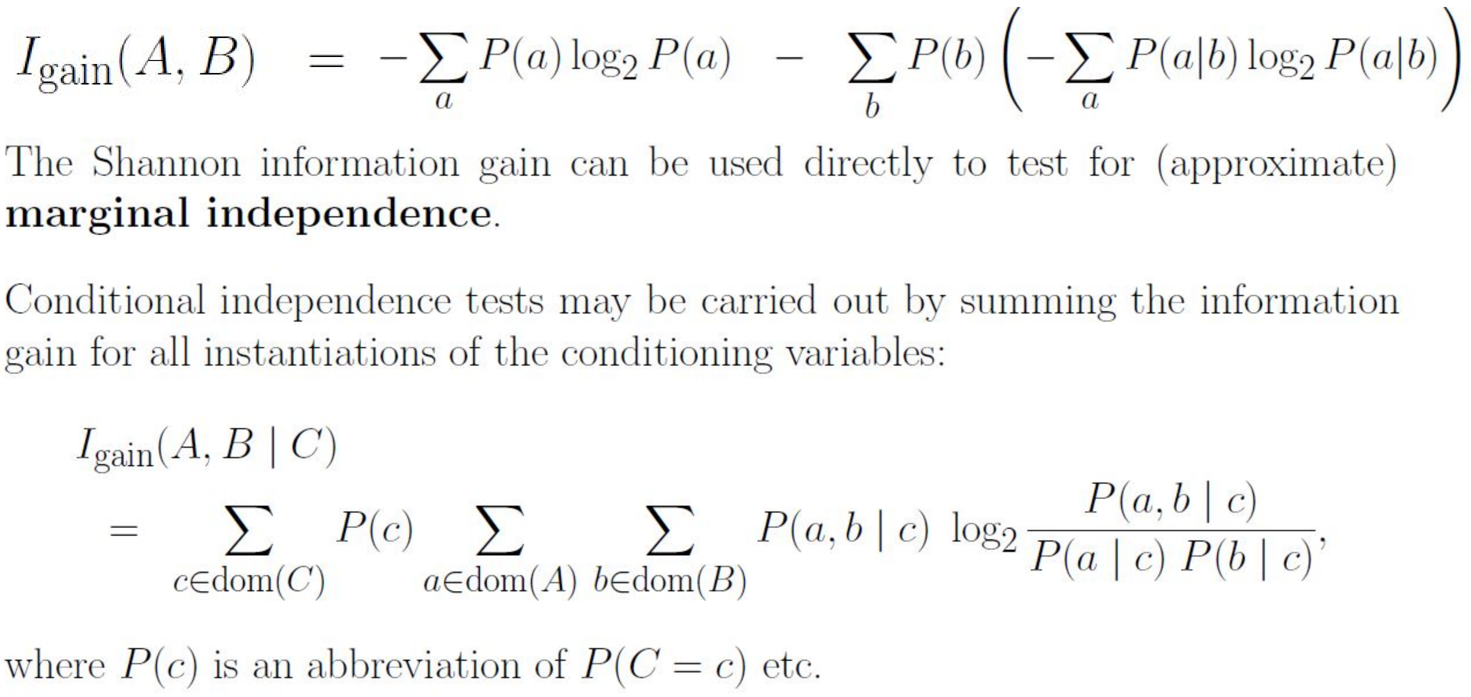
- **Constraint-based Methods**
  Find independences and conditional independences in given data via statistical tests and choose a DAG with the same (or similar) independences.

- **Score-based Method**
  Search over all possible DAGs and choose the one with a maximum score.

Constraint Based Methods: (Conditional) Independence Tests

How to decide A $\perp\!\!\!\perp$ B , A $\perp\!\!\!\perp$ B I C from a given dataset?

Let P be the (unknown) probability distribution that „induced“ the given data set D.



$$I_{\text{gain}}(A,B) \quad = -\sum_a P(a)\log_2 P(a) \quad - \quad \sum_b P(b)\left(-\sum_a P(a|b)\log_2 P(a|b)\right)$$

The Shannon information gain can be used directly to test for (approximate) **marginal independence**.

Conditional independence tests may be carried out by summing the information gain for all instantiations of the conditioning variables:

$$I_{\text{gain}}(A,B \mid C) = \sum_{c\in\text{dom}(C)} P(c) \sum_{a\in\text{dom}(A)} \sum_{b\in\text{dom}(B)} P(a,b \mid c)\ \log_2 \frac{P(a,b \mid c)}{P(a \mid c)\ P(b \mid c)},$$

where $P(c)$ is an abbreviation of $P(C=c)$ etc.

Note: P is unknown, but we can estimate the information gain by using the corresponding frequences in the data set instead. Large deviance of P(AIC) x P(BIC) from P(A,B I C) rejects the null hypothesis of conditional independence. Often an analyst assumes conditional independence if the information gain is below a given treshhold Ɛ.

Constrained Based Methods: A Simple Example

Suppose that the following conditional independence statements hold:

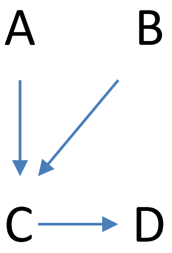
$$
\begin{array}{ll}
A \perp\!\!\!\perp_{\hat{P}} B \mid \emptyset & \qquad B \perp\!\!\!\perp_{\hat{P}} A \mid \emptyset \\
A \perp\!\!\!\perp_{\hat{P}} D \mid C & \qquad D \perp\!\!\!\perp_{\hat{P}} A \mid C \\
B \perp\!\!\!\perp_{\hat{P}} D \mid C & \qquad D \perp\!\!\!\perp_{\hat{P}} B \mid C
\end{array}
$$

All other possible conditional independence statements that can be formed with the attributes A, B, C, and D (with single attributes on the left) do not hold.

Step 1: Since there is no set rendering A and C, B and C and C and D independent, the edges $A - C$, $B - C$, and $C - D$ are inserted.

Step 2: Since C is a common neighbor of A and B and we have $A \perp\!\!\!\perp_{\hat{P}} B \mid \emptyset$, but $A \not\!\perp\!\!\!\perp_{\hat{P}} B \mid C$, the first two edges must be directed $A \rightarrow C \leftarrow B$.

Step 3: Since A and D are not adjacent, $C - D$ and $A \rightarrow C$, the edge $C - D$ must be directed $C \rightarrow D$.
(Otherwise step 2 would have already fixed the orientation $C \leftarrow D$.)



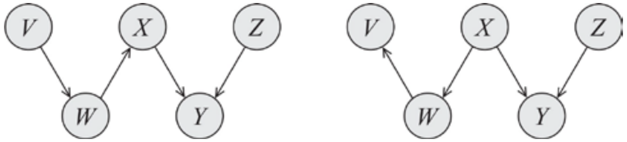
The resulting directed graph is a (minimal) perfect I-map.

Constraint Based Methods: Find a DAG with a same set of conditional independences

Given: Data Set with good Conditional Independence Tests (CI-Tests)

Goal : Find a **Perfect Independance Map** (P-Map) , i.e. a DAG with a same set of conditional independences as the underlying probability distribution P that generated the data set (the iid sample assumption)

Restrictions: Not every distribution has P-Map. If the distribution P has a P-Map (its faithfulness), then it is not unique. We can only determine a class of faithful DAGs. If two DAGs are in the same class, they are called I-equivalent. Two I-equivalent DAGs share the same undirected skeleton and the same immoral set of v-structures



PC Algorithm

1. find the **undirected skeleton** using CI tests
2. identify **immoralities** in the undirected graph
3. **add directions** that follow from immoralities and DAG structure

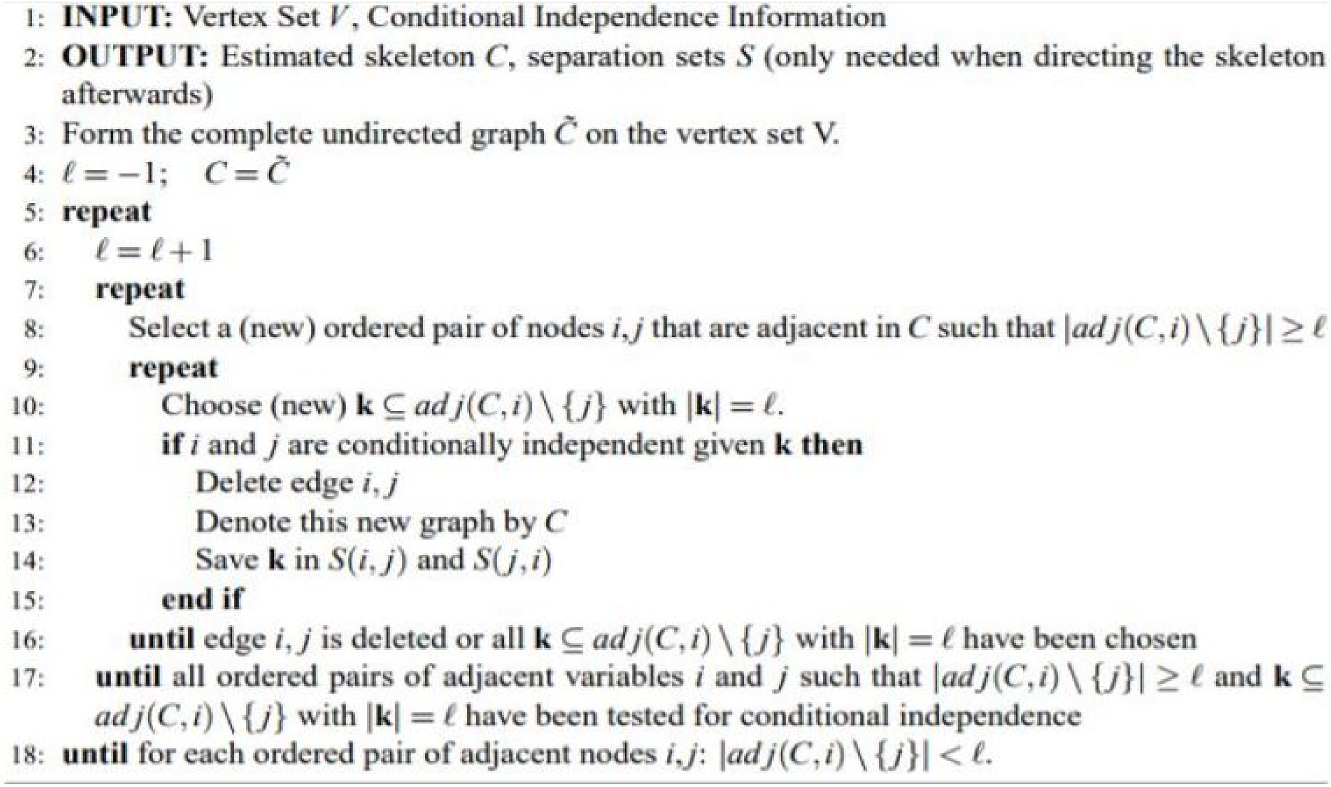
In case of exact CI tests, faithfullness the so called **PC-algorithm** guarantees to find the exact I-equivalence family.

Constraint Bases Methods: Perfect map from CI Oracles

Idea: Find appropriate separation sets S(X,Y) for all variables X and Y in order to remove edges
The check $X \perp\!\!\!\perp Y \mid S(X, Y)$ for all possible separation sets $S(X,Y) \subseteq V \setminus \{X,Y\}$ is infeasible for large spaces.

Step 1 Construction of an (undirected) skeleton of the graph by iteration over the size of the separation sets.
A separation is verified by a CI-test (a so called CI Oracle)

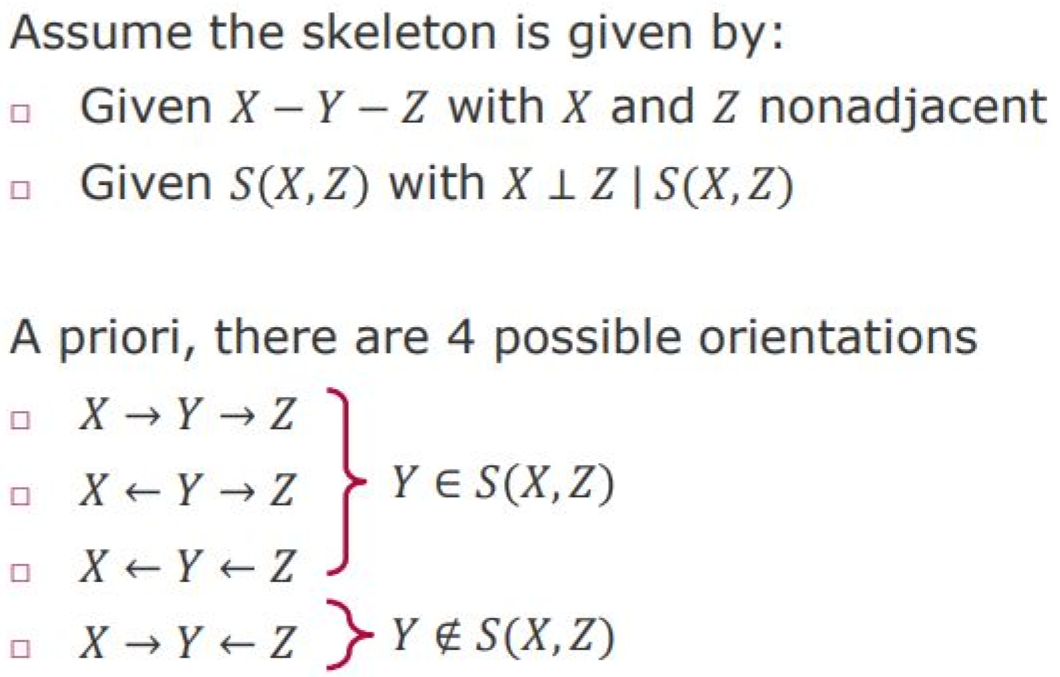
Pseudocode PC Algorithm for step 1



```
INPUT: Vertex Set $V$, Conditional Independence Information
OUTPUT: Estimated skeleton $C$, separation sets $S$ (only needed when directing the skeleton afterwards)
Form the complete undirected graph $\tilde{C}$ on the vertex set V.
$\ell = -1; \quad C = \tilde{C}$
repeat
    $\ell = \ell + 1$
    repeat
        Select a (new) ordered pair of nodes $i, j$ that are adjacent in $C$ such that $|adj(C,i) \setminus \{j\}| \geq \ell$
        repeat
            Choose (new) $\mathbf{k} \subseteq adj(C,i) \setminus \{j\}$ with $|\mathbf{k}| = \ell$.
            if $i$ and $j$ are conditionally independent given $\mathbf{k}$ then
                Delete edge $i, j$
                Denote this new graph by $C$
                Save $\mathbf{k}$ in $S(i,j)$ and $S(j,i)$
            end if
        until edge $i, j$ is deleted or all $\mathbf{k} \subseteq adj(C,i) \setminus \{j\}$ with $|\mathbf{k}| = \ell$ have been chosen
    until all ordered pairs of adjacent variables $i$ and $j$ such that $|adj(C,i) \setminus \{j\}| \geq \ell$ and $\mathbf{k} \subseteq adj(C,i) \setminus \{j\}$ with $|\mathbf{k}| = \ell$ have been tested for conditional independence
until for each ordered pair of adjacent nodes $i,j$: $|adj(C,i) \setminus \{j\}| < \ell$.
```

Constraint Bases Methods: Perfect map from CI Oracles

Idea Find V-Structures (so called immoralities) in the skeleton



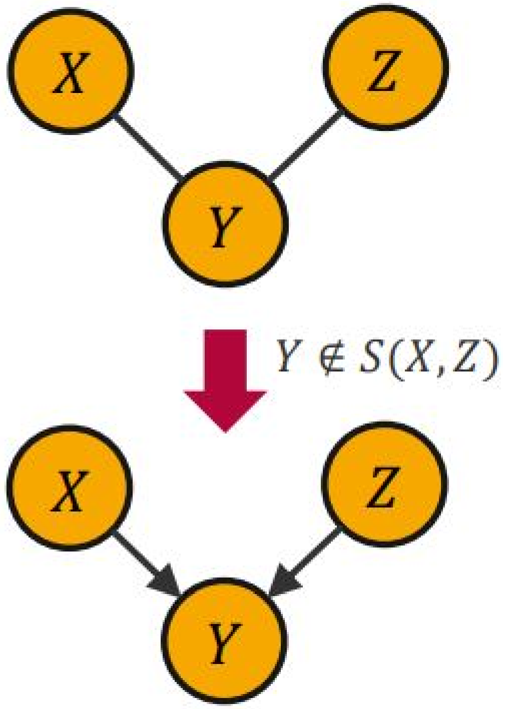
Assume the skeleton is given by:

- □ Given $X - Y - Z$ with X and Z nonadjacent
- □ Given $S(X,Z)$ with $X \perp Z \mid S(X,Z)$

A priori, there are 4 possible orientations

$$\left.\begin{array}{l} \square \quad X \rightarrow Y \rightarrow Z \\ \square \quad X \leftarrow Y \rightarrow Z \\ \square \quad X \leftarrow Y \leftarrow Z \end{array}\right\} Y \in S(X,Z)$$

$$\left.\begin{array}{l} \square \quad X \rightarrow Y \leftarrow Z \end{array}\right\} Y \notin S(X,Z)$$

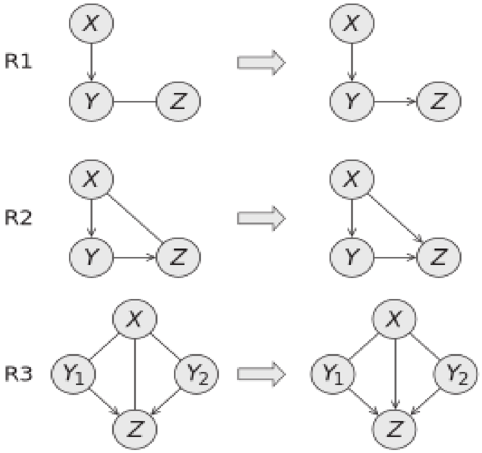


Step 2 If Y $\notin$ S(X,Z) then replace X – Y – Z by X → Y ← Z

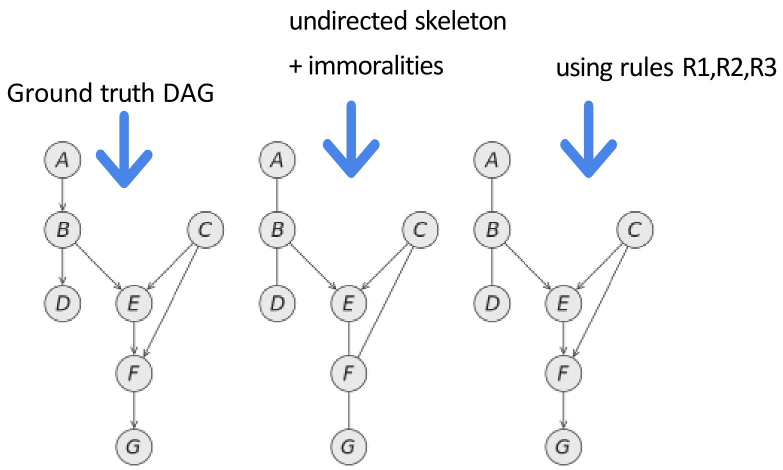
Constraint Bases Methods: Perfect map from CI Oracles

Idea: At this point we have a mix of directed and undirected edges. Add directions using constraints , that are needed to preserve immoralities or follow from the DAG structure.

Step 3 Propagate Contraints



Example



Ground truth DAG

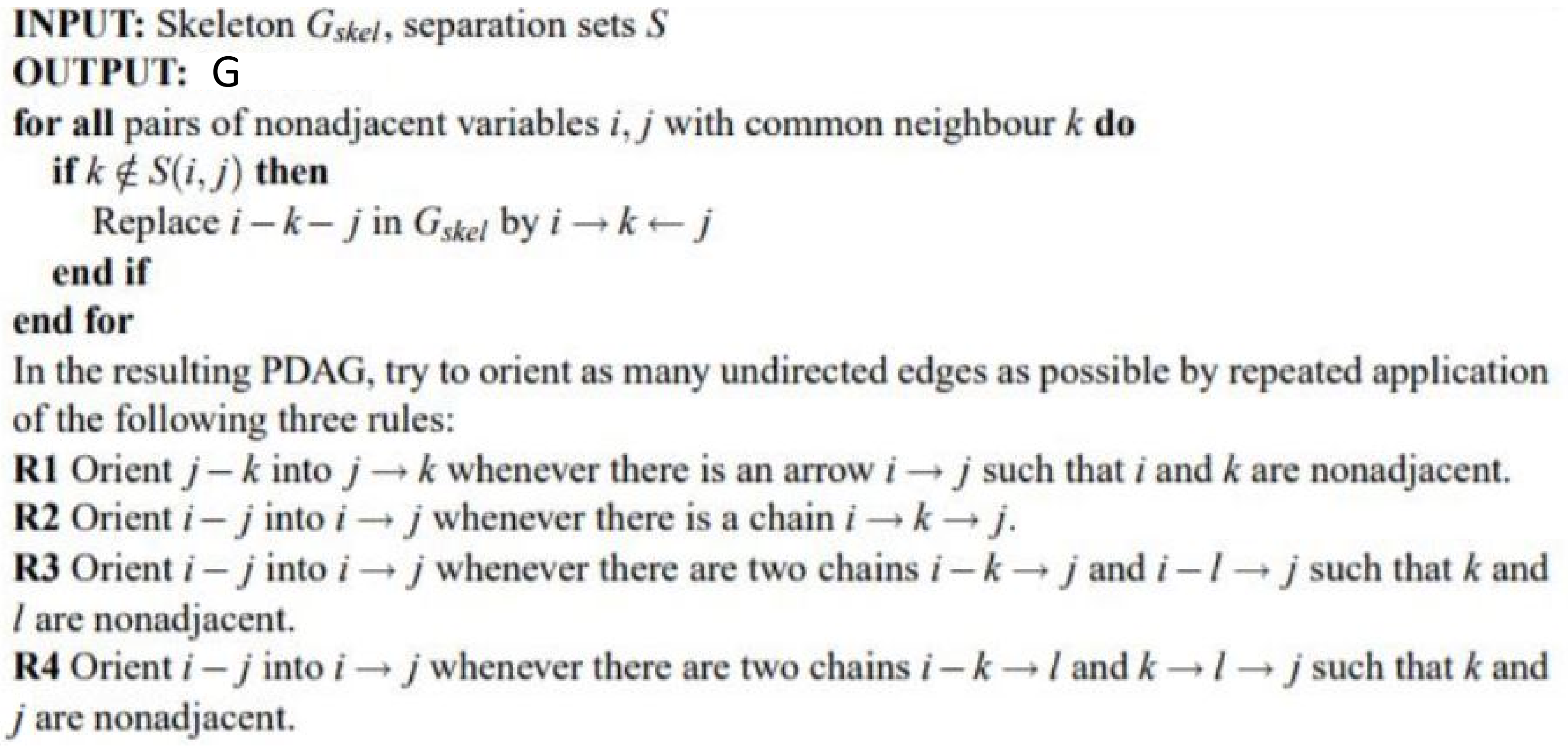
undirected skeleton

+ immoralities

using rules R1,R2,R3

Constraint Bases Methods: Perfect map from CI tests

Pseudocode PC Algorithm for steps 2 and 3



```
INPUT: Skeleton $G_{skel}$, separation sets $S$
OUTPUT: G
for all pairs of nonadjacent variables $i, j$ with common neighbour $k$ do
  if $k \notin S(i, j)$ then
    Replace $i - k - j$ in $G_{skel}$ by $i \rightarrow k \leftarrow j$
  end if
end for
```

In the resulting PDAG, try to orient as many undirected edges as possible by repeated application of the following three rules:

R1 Orient $j - k$ into $j \rightarrow k$ whenever there is an arrow $i \rightarrow j$ such that i and k are nonadjacent.

R2 Orient $i - j$ into $i \rightarrow j$ whenever there is a chain $i \rightarrow k \rightarrow j$.

R3 Orient $i - j$ into $i \rightarrow j$ whenever there are two chains $i - k \rightarrow j$ and $i - l \rightarrow j$ such that k and l are nonadjacent.

R4 Orient $i - j$ into $i \rightarrow j$ whenever there are two chains $i - k \rightarrow l$ and $k \rightarrow l \rightarrow j$ such that k and j are nonadjacent.

Evaluation of the PC algorithm

- Works under the (strong) assumptions of causal sufficiency, faithfulness and Global Markov Condition.
- Testing all sets $S(X,Y)$ containing the adjacencies of X is sufficient
- Polynomial complexity for graph of N vertices of bounded degree, but in the worst case exponential complexity to N

Score Based Methods: Different scores available

Score-based approaches cast the learning problem as an optimization problem. Given a scoring criterium S and a data set D, the optimal graph is the one with the highest score:

$$G^* = \arg\max_{G} S(G,D)$$

There are **lots of different scores** from the area of model selection such Bayesian Information Criterium (BIC), Akaike Information Criterion (AIC), Minimum Description lenght (MDL), or Log Likelihood.

In this course we study mainly the discrete case, so we present the **Bayesian Dirichlet Score**, which conjugates with multinomial probability distributions.

The process of solving can be seen as a search over the space of all possible graphs. For trees the Chow-Liu algorithm gives good results, for more general DAGs (several parents) the problem of finding an optimal structure is NP-hard. So one uses **heuristic search methods** or **Monte Carlo Methods** (e.g. Markov Chain Monte Carlo).

Score-based methods: Kullback-Leibler Information Divergence

Definition: Let P_1 and P_2 be two strictly positive probability distributions on the same set $\mathcal{E}$ of events. Then

$$I_{\text{KLdiv}}(P_1, P_2) = \sum_{F \in \mathcal{E}} P_1(F) \log_2 \frac{P_1(F)}{P_2(F)}$$

is called the **Kullback-Leibler information divergence** of P_1 and P_2.

The Kullback-Leibler information divergence is non-negative.

It is zero if and only if $P_1 \equiv P_2$.

Therefore it is plausible that this measure can be used to assess the quality of the approximation of a given multi-dimensional distribution P_1 by the distribution P_2 that is represented by a given graph:

The smaller the value of this measure, the better the approximation.

Score-based methods: Kullback-Leibler Information Divergence

Mutual Information / Cross Entropy / Information Gain

Based on Shannon Entropy $H = -\sum_{i=1}^{n} p_i \log_2 p_i$ (Shannon 1948)

$$\begin{aligned} I_{\text{gain}}(A, B) &= \overbrace{\qquad H(A) \qquad}^{} - \overbrace{\qquad H(A \mid B) \qquad}^{} \\ &= -\sum_a P(a) \log_2 P(a) - \sum_b P(b) \left(-\sum_a P(a|b) \log_2 P(a|b) \right) \end{aligned}$$

$H(A)$ Entropy of the distribution on attribute A

$H(A|B)$ *Expected entropy* of the distribution on attribute A if the value of attribute B becomes known

$H(A) - H(A|B)$ Expected reduction in entropy or *information gain*

Score-based methods: Kullback-Leibler Information Divergence

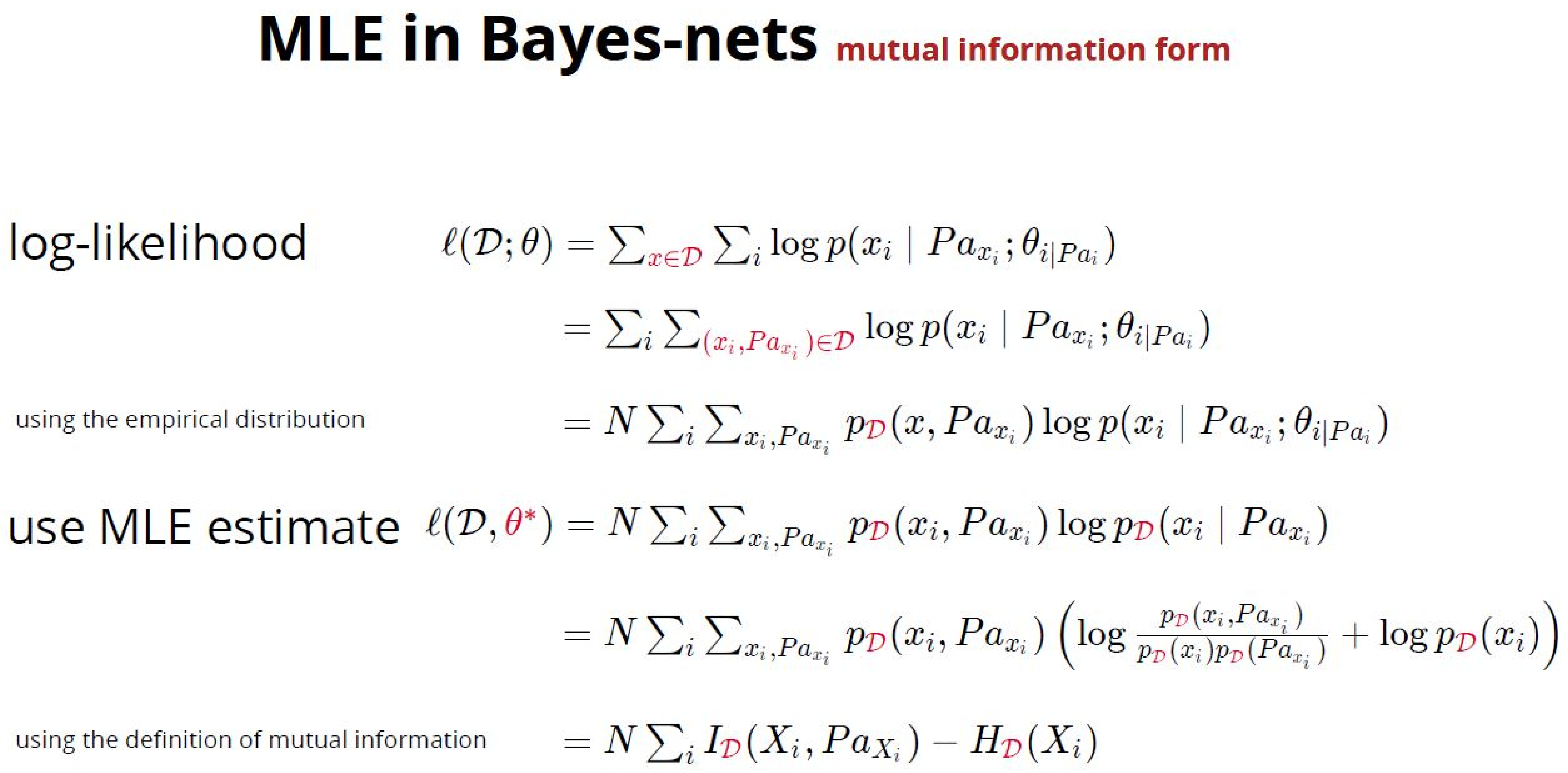
$$\begin{aligned}
I_{\text{gain}}(A,B) &= -\sum_a P(a)\log_2 P(a) - \sum_b P(b)\left(-\sum_a P(a|b)\log_2 P(a|b)\right) \\
&= -\sum_a \sum_b P(a,b)\log_2 P(a) + \sum_b \sum_a P(a|b)P(b)\log_2 P(a|b) \\
&= \quad \sum_a \sum_b P(a,b)\left(\log_2 \frac{P(a,b)}{P(b)} - \log_2 P(a)\right) \\
&= \quad \sum_a \sum_b P(a,b)\log_2 \frac{P(a,b)}{P(a)P(b)}
\end{aligned}$$

The information gain equals the Kullback-Leibler information divergence between the actual distribution $P(A,B)$ and a hypothetical distribution P^* in which A and B are marginal independent:

$$P^*(A,B) = P(A)\cdot P(B)$$

$$I_{\text{gain}}(A,B) = I_{\text{KLdiv}}(P,P^*)$$

Score Based Methods: Likelihood scores



MLE in Bayes-nets mutual information form

log-likelihood $\ell(\mathcal{D};\theta) = \sum_{x\in\mathcal{D}} \sum_i \log p(x_i \mid Pa_{x_i}; \theta_{i|Pa_i})$

$$= \sum_i \sum_{(x_i, Pa_{x_i})\in\mathcal{D}} \log p(x_i \mid Pa_{x_i}; \theta_{i|Pa_i})$$

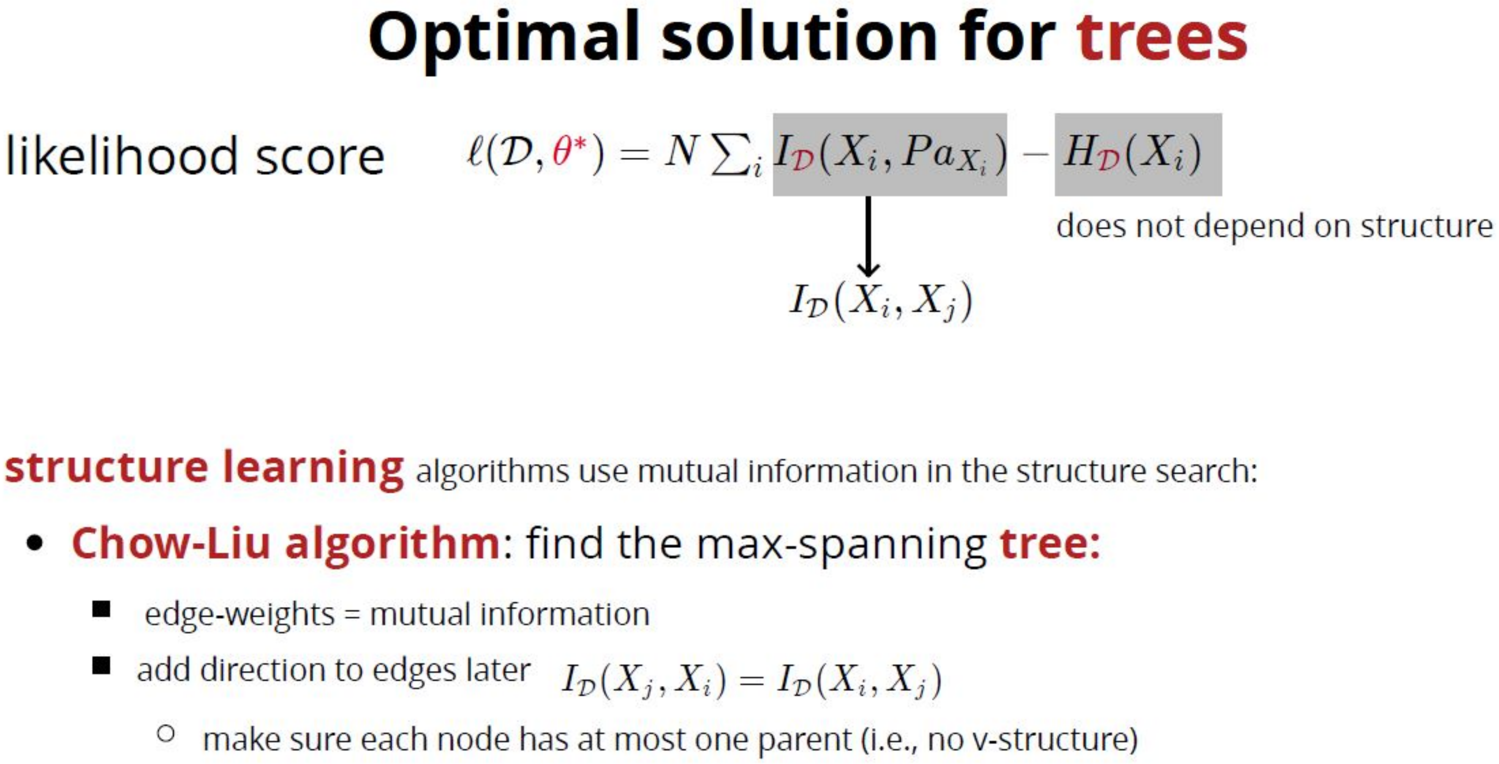
using the empirical distribution $= N \sum_i \sum_{x_i, Pa_{x_i}} p_{\mathcal{D}}(x, Pa_{x_i}) \log p(x_i \mid Pa_{x_i}; \theta_{i|Pa_i})$

use MLE estimate $\ell(\mathcal{D},\theta^*) = N \sum_i \sum_{x_i, Pa_{x_i}} p_{\mathcal{D}}(x_i, Pa_{x_i}) \log p_{\mathcal{D}}(x_i \mid Pa_{x_i})$

$$= N \sum_i \sum_{x_i, Pa_{x_i}} p_{\mathcal{D}}(x_i, Pa_{x_i}) \left(\log \frac{p_{\mathcal{D}}(x_i, Pa_{x_i})}{p_{\mathcal{D}}(x_i) p_{\mathcal{D}}(Pa_{x_i})} + \log p_{\mathcal{D}}(x_i)\right)$$

using the definition of mutual information $= N \sum_i I_{\mathcal{D}}(X_i, Pa_{X_i}) - H_{\mathcal{D}}(X_i)$

Score Based Methods: Likelihood scores for trees



Optimal solution for **trees**

likelihood score $\ell(\mathcal{D},\theta^*) = N\sum_i \boxed{I_{\mathcal{D}}(X_i, Pa_{X_i})} - \boxed{H_{\mathcal{D}}(X_i)}$

$\downarrow$

$I_{\mathcal{D}}(X_i, X_j)$

does not depend on structure

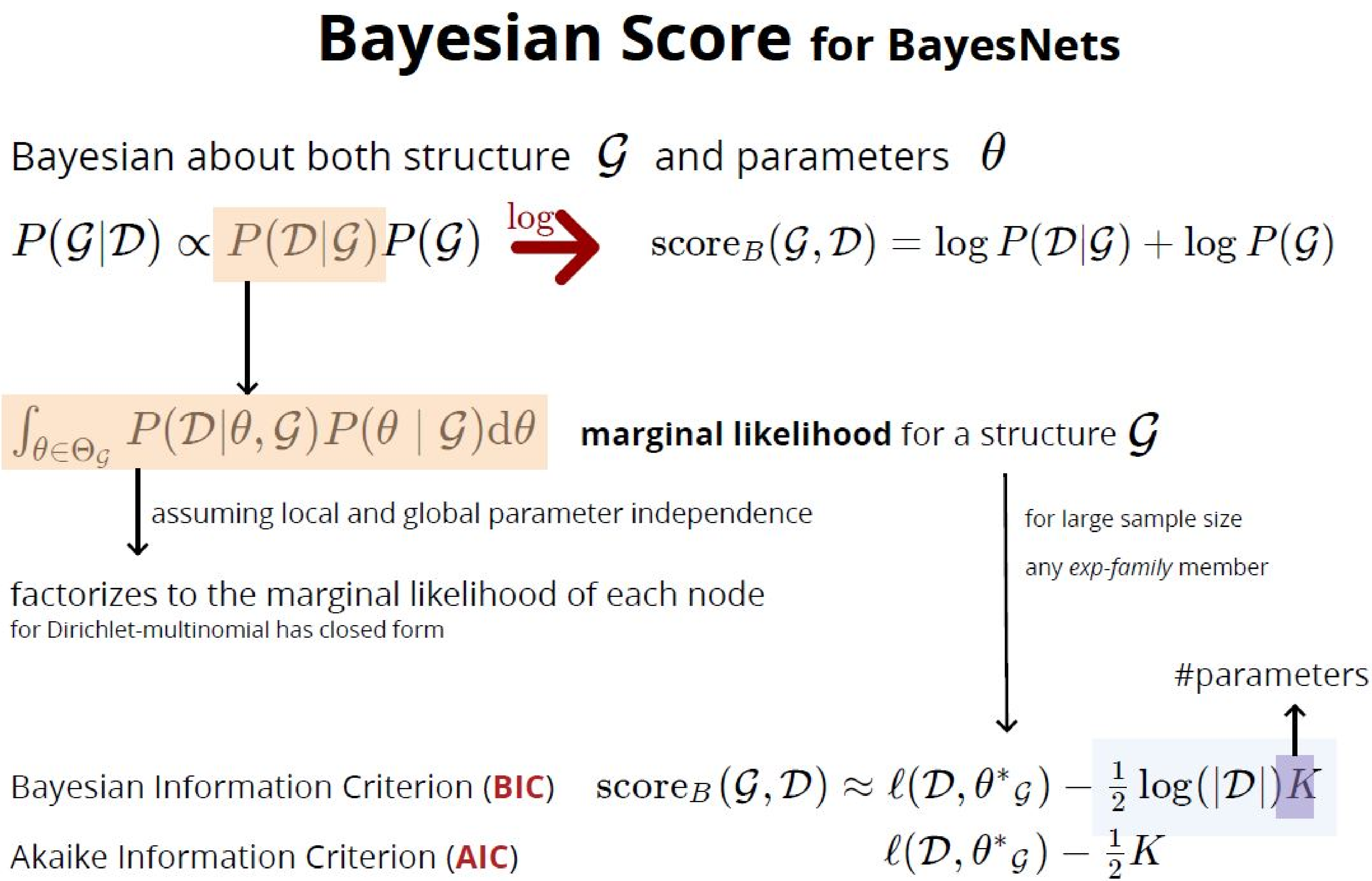
structure learning algorithms use mutual information in the structure search:

- **Chow-Liu algorithm**: find the max-spanning **tree:**
  - edge-weights = mutual information
  - add direction to edges later $I_{\mathcal{D}}(X_j, X_i) = I_{\mathcal{D}}(X_i, X_j)$
    - make sure each node has at most one parent (i.e., no v-structure)

Note: The distribution associated to the tree constructed by Chow-Liu algorithm is the one that is closest to the distribution associated to the data as measured by the Kullbach-Leibler divergence - in case of no missing data , discrete variables, and iid data.

https://de.slideshare.net/vangjee/a-quick-introduction-to-the-chow-liu-algorithm

Score Based Methods: Bayesian scores



Bayesian Score for BayesNets

Bayesian about both structure $\mathcal{G}$ and parameters θ

$$P(\mathcal{G}|\mathcal{D}) \propto P(\mathcal{D}|\mathcal{G})P(\mathcal{G}) \xrightarrow{\log} \text{score}_B(\mathcal{G},\mathcal{D}) = \log P(\mathcal{D}|\mathcal{G}) + \log P(\mathcal{G})$$

$$\int_{\theta \in \Theta_{\mathcal{G}}} P(\mathcal{D}|\theta,\mathcal{G})P(\theta \mid \mathcal{G})\mathrm{d}\theta$$

marginal likelihood for a structure $\mathcal{G}$

assuming local and global parameter independence

factorizes to the marginal likelihood of each node

for Dirichlet-multinomial has closed form

for large sample size

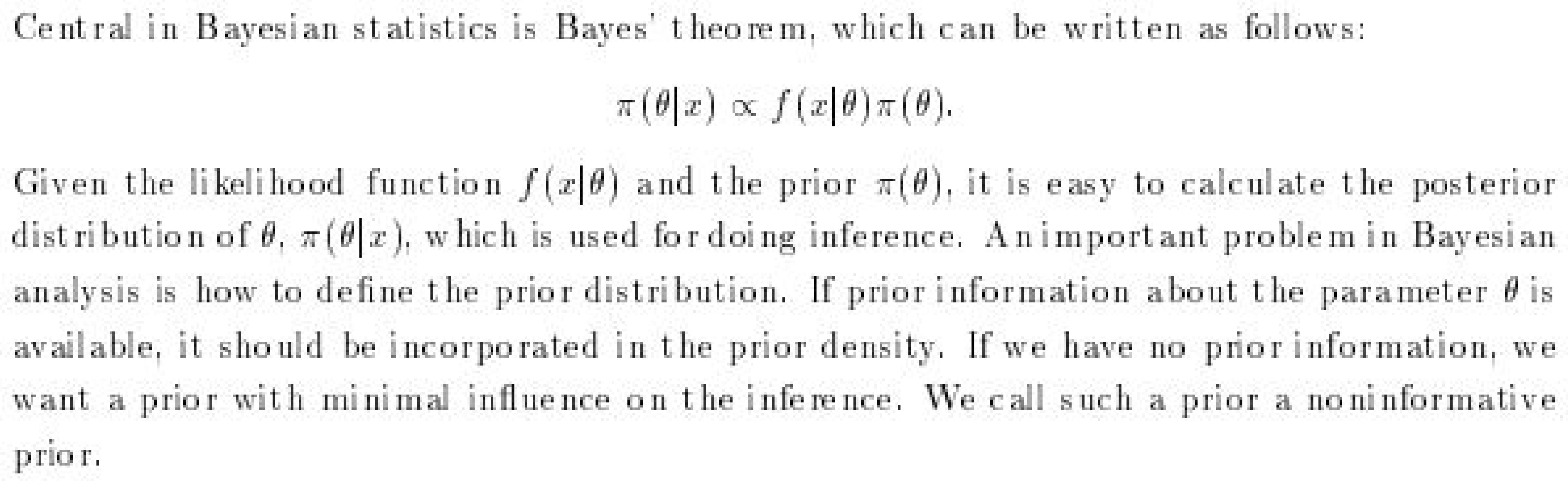
any *exp-family* member

#parameters

Bayesian Information Criterion (**BIC**) $\text{score}_B(\mathcal{G},\mathcal{D}) \approx \ell(\mathcal{D},\theta^*{}_{\mathcal{G}}) - \frac{1}{2}\log(|\mathcal{D}|)K$

Akaike Information Criterion (**AIC**) $\ell(\mathcal{D},\theta^*{}_{\mathcal{G}}) - \frac{1}{2}K$

Bayesian Statistics



Central in Bayesian statistics is Bayes' theorem, which can be written as follows:

$$\pi(\theta|x) \propto f(x|\theta)\pi(\theta).$$

Given the likelihood function $f(x|\theta)$ and the prior $\pi(\theta)$, it is easy to calculate the posterior distribution of θ, $\pi(\theta|x)$, which is used for doing inference. An important problem in Bayesian analysis is how to define the prior distribution. If prior information about the parameter θ is available, it should be incorporated in the prior density. If we have no prior information, we want a prior with minimal influence on the inference. We call such a prior a noninformative prior.

The Dirichlet distribution is a conjugate prior for the multinomial distribution. This means that if the prior distribution of the multinomial parameters is Dirichlet then the posterior distribution is also a Dirichlet distribution (with parameters different from those of the prior). The benefit of this is that (a) the posterior distribution is easy to compute and (b) it in some sense is possible to quantify how much our beliefs have changed after collecting the data.

https://en.wikipedia.org/wiki/Dirichlet_distribution

Example: Dirichlet Bayesian Equivalent Uniform Score

Model Averaging

We first consider $P(B_S, D)$ to be the marginalization of $P(B_S, B_P, D)$ over all possible parameters B_P.

$$\begin{aligned}
P(B_S, D) &= \int_{B_P} P(B_S, B_P, D)\,\mathrm{d}B_P \\
&= \int_{B_P} P(D \mid B_S, B_P)\, P(B_S, B_P)\,\mathrm{d}B_P \\
&= \int_{B_P} P(D \mid B_S, B_P)\, f(B_P \mid B_S) P(B_S)\,\mathrm{d}B_P \\
&= \underbrace{P(B_S)}_{\text{A priori prob.}} \int_{B_P} \underbrace{P(D \mid B_S, B_P)}_{\text{Likelihood of } D} \underbrace{f(B_P \mid B_S)}_{\text{Parameter densities}} \,\mathrm{d}B_P
\end{aligned}$$

Score Based Algorithms: Bayesian Dirichlet Score

The a priori distribution $P(B_S)$ can be used to bias the evaluation measure towards user-specific network structures.

Substitute the likelihood $P(D \mid B_S, B_P)$ for its specific form:

$$P(B_S, D) \;=\; P(B_S) \int_{B_P} \underbrace{\left[\prod_{i=1}^{n} \prod_{j=1}^{q_i} \prod_{k=1}^{r_i} \theta_{ijk}^{\alpha_{ijk}} \right]}_{P(D|B_S,B_P)} f(B_P \mid B_S)\, \mathrm{d}B_P$$

Score Based Algorithms: Bayesian Dirichlet Score

The parameter densities $f(B_P \mid B_S)$ describe the probabilities of the parameters given a network structure.

They are densities of second order (distribution over distributions)

For fixed i and j, a vector $(\theta_{ij1}, \ldots, \theta_{ijr_i})$ represents a probability distribution, namely the j-th column of the i-th potential table.

Assuming mutual independence between the potential tables, we arrive for $f(B_P \mid B_S)$ at the following:

$$f(B_P \mid B_S) \quad = \quad \prod_{i=1}^{n} \prod_{j=1}^{q_i} f(\theta_{ij1}, \ldots, \theta_{ijr_i})$$

Score Based Algorithms: Bayesian Dirichlet Score

Thus, we can further concretize the equation for $P(B_S, D)$:

$$\begin{aligned} P(B_S, D) &= P(B_S) \underset{\theta_{ijk}}{\int \cdots \int} \left[\prod_{i=1}^{n} \prod_{j=1}^{q_i} \prod_{k=1}^{r_i} \theta_{ijk}^{\alpha_{ijk}} \right] \cdot \left[\prod_{i=1}^{n} \prod_{j=1}^{q_i} f(\theta_{ij1}, \ldots, \theta_{ijr_i}) \right] \mathrm{d}\theta_{111}, \ldots, \mathrm{d}\theta_{nq_nr_n} \\ &= P(B_S) \prod_{i=1}^{n} \prod_{j=1}^{q_i} \underset{\theta_{ijk}}{\int \cdots \int} \left[\prod_{k=1}^{r_i} \theta_{ijk}^{\alpha_{ijk}} \right] \cdot f(\theta_{ij1}, \ldots, \theta_{ijr_i}) \, \mathrm{d}\theta_{ij1}, \ldots, \mathrm{d}\theta_{ijr_i} \end{aligned}$$

Score Based Algorithms: Bayesian Dirichlet Score

A last assumption: For fixed i and j the density $f(\theta_{ij1}, \ldots, \theta_{ijr_i})$ is uniform:

$$f(\theta_{ij1}, \ldots, \theta_{ijr_i}) = (r_i - 1)!$$

It simplifies $P(B_S, D)$ further:

$$\begin{aligned}
P(B_S, D) &= P(B_S) \prod_{i=1}^{n} \prod_{j=1}^{q_i} \int_{\theta_{ijk}} \cdots \int \left[\prod_{k=1}^{r_i} \theta_{ijk}^{\alpha_{ijk}} \right] \cdot (r_i - 1)! \, \mathrm{d}\theta_{ij1}, \ldots, \mathrm{d}\theta_{ijr_i} \\
&= P(B_S) \prod_{i=1}^{n} \prod_{j=1}^{q_i} (r_i - 1)! \qquad \underbrace{\int_{\theta_{ijk}} \cdots \int \prod_{k=1}^{r_i} \theta_{ijk}^{\alpha_{ijk}} \, \mathrm{d}\theta_{ij1}, \ldots, \mathrm{d}\theta_{ijr_i}}_{\text{Dirichlet's integral} = \frac{\prod_{k=1}^{r_i} \alpha_{ijk}!}{(\sum_{k=1}^{r_i} \alpha_{ijk} + r_i - 1)!}}
\end{aligned}$$

Score Based Algorithms: Bayesian Dirichlet Score

We finally arrive at an expression for $P(B_S, D)$:

$$P(B_S, D) \quad = \quad \mathrm{K2}(B_S \mid D) \quad = \quad P(B_S) \prod_{i=1}^{n} \prod_{j=1}^{q_i} \left[\frac{(r_i - 1)!}{(N_{ij} + r_i - 1)!} \prod_{k=1}^{r_i} \alpha_{ijk}! \right]$$

n number of attributes describing the domain under consideration

r_i number of values of the i-th attribute A_i, i. e., $r_i = |\mathrm{dom}(A_i)|$

q_i number of instantiations of the parents of the i-th attribute in $\vec{G}$, i. e., $q_i = \prod_{A_j \in \mathrm{parents}(A_i)} r_i = \prod_{A_j \in \mathrm{parents}(A_i)} |\mathrm{dom}(A_i)|$

α_{ijk} number of sample cases in which the i-th attribute has its k-th value and its parents in $\vec{G}$ have their j-th instantiation

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

Score Based Algorithms: Bayesian Dirichlet Score

Global — Refers to the outer product: The total value of the K2 metric is the product over all K2 values of attribute families.

Local — The likelihood equation assumes that given a parents instantiation, the probabilities for the respective child attribute values are mutual independent. This is reflected in the product over all q_i different parent attributes' value combinations of attribute A_i.

We exploit the global property to write the K2 metric as follows:

$$\mathrm{K2}(B_S \mid D) \quad = \quad P(B_S) \prod_{i=1}^{n} \mathrm{K2}_{\text{local}}(A_i \mid D)$$

with

$$\mathrm{K2}_{\text{local}}(A_i \mid D) \quad = \quad \prod_{j=1}^{q_i} \left[\frac{(r_i - 1)!}{(N_{ij} + r_i - 1)!} \prod_{k=1}^{r_i} \alpha_{ijk}! \right]$$

Score Based Algorithms: Bayesian Dirichlet Score

Prerequisites:

Choose a topological order on the attributes $(A_1, \dots, A_n)$

Start out with a network that consists of n isolated nodes.

Let ζ_i be the quality of the i-th attribute given the (tentative) set of parent attributes M:

$$\zeta_i(M) = \mathrm{K2}_{\mathrm{local}}(A_i \mid D) \quad \text{with} \quad \mathrm{parents}(A_i) = M$$

Score Based Algorithms: Bayesian Dirichlet Score

Execution:

1. Determine for the parentless node A_i the quality measure $\zeta_i(\emptyset)$
2. Evaluate for every predecessor $\{A_1, \ldots, A_{i-1}\}$ whether inserted as parent of A_i, the quality measure would increase. Let Y be the node that yields the highest quality (increase):

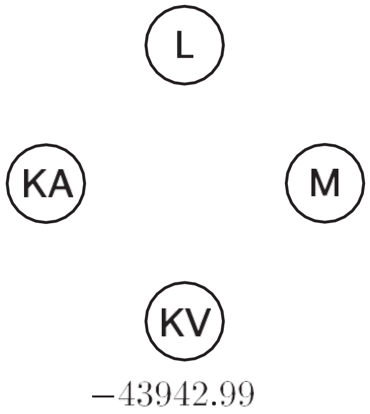
   $$Y = \arg\max_{1 \leq l \leq i-1} \zeta_i(\{A_l\})$$

   This best quality measure be $\zeta = \zeta_i(\{Y\})$.
3. If ζ is better than $\zeta_i(\emptyset)$, Y is inserted permanently as a parent node: $\text{parents}(A_i) = \text{parents}(A_i) \cup \{Y\}$
4. Repeat steps 2 and 3 to increase the parent set until no quality increase can be achieved or no nodes are left or a predefined maximum number of parent nodes per node is reached.

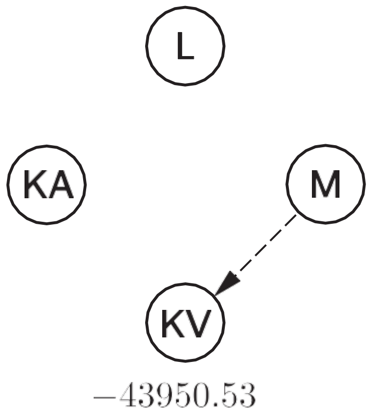
Score Based Algorithms: Bayesian Dirichlet Score

```
for $i \leftarrow 1 \ldots n$ do // Initialization
    $\text{parents}(A_i) \leftarrow \emptyset$
end for
for $i \leftarrow n, \ldots, 1$ do // Iteration
    repeat
        Select $Y \in \{A_1, \ldots, A_{i-1}\} \setminus \text{parents}(A_i)$,
            which maximizes $\zeta = \zeta_i(\text{parents}(A_i) \cup \{Y\})$
        $\delta \leftarrow \zeta - \zeta_i(\text{parents}(A_i))$
        if $\delta > 0$ then
            $\text{parents}(A_i) \leftarrow \text{parents}(A_i) \cup \{Y\}$
        end if
    until $\delta \le 0$ or $\text{parents}(A_i) = \{A_1, \ldots, A_{i-1}\}$ or $|\text{parents}(A_i)| = n_{\text{max}}$
end for
```

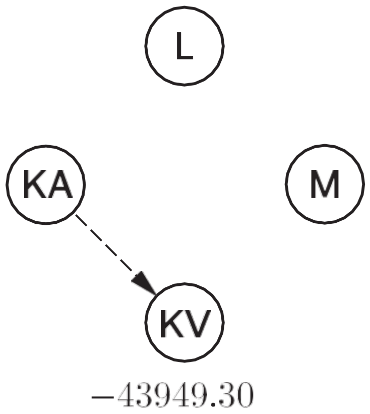
Demo of K2 Algorithm



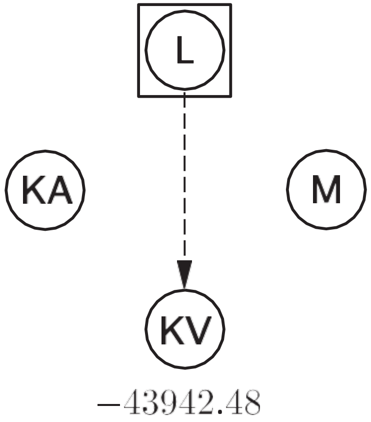
Step 1 – Edgeless graph



Step 2 – Insert M temporarily.

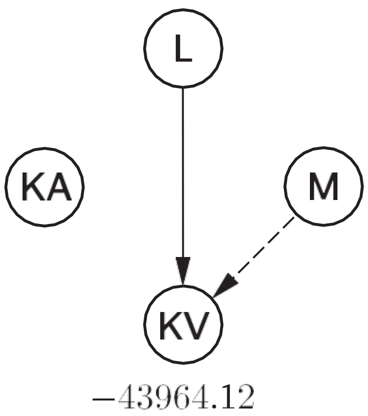


Step 3 – Insert KA temporarily.

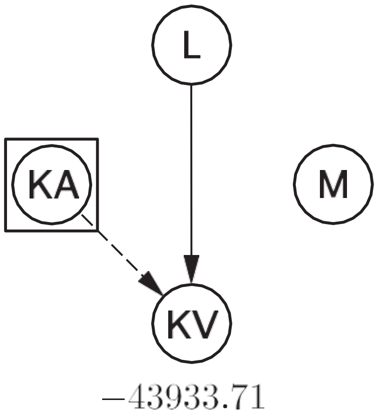


Step 4 – Node L maximizes K2 value and thus is added permantently.

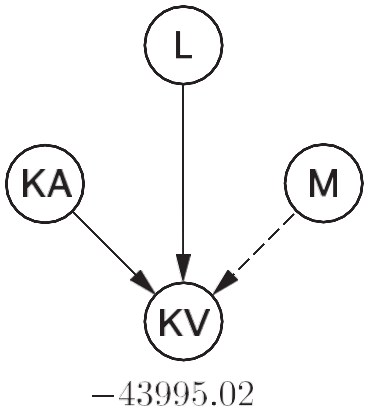
Demo of K2 Algorithm



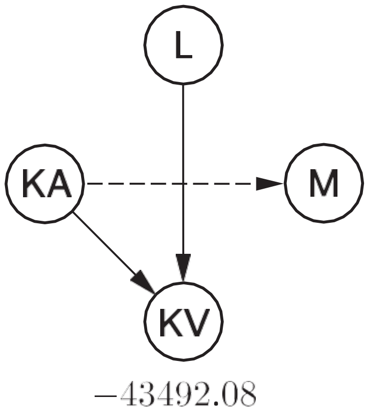
Step 5 – Insert M temporarily.



Step 6 – KA is added as second parent node of KV.

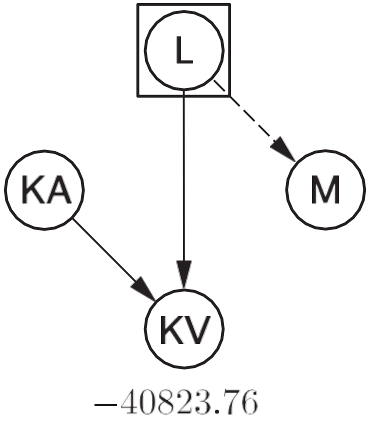


Step 7 – M does not increase the quality of the network if insertes as third parent node.

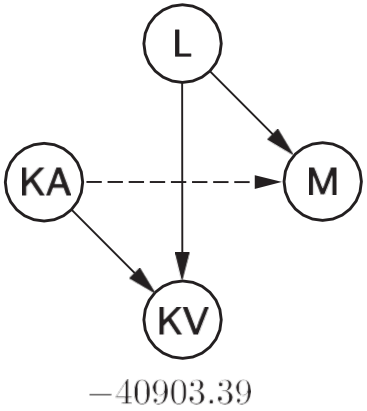


Step 8 – Insert KA temporarily.

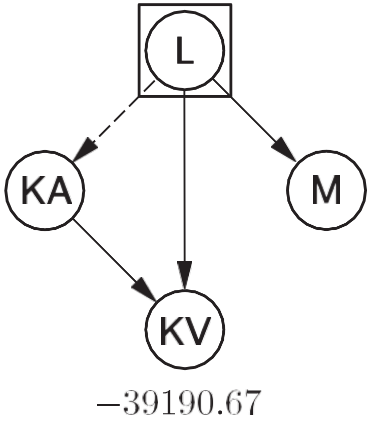
Demo of K2 Algorithm



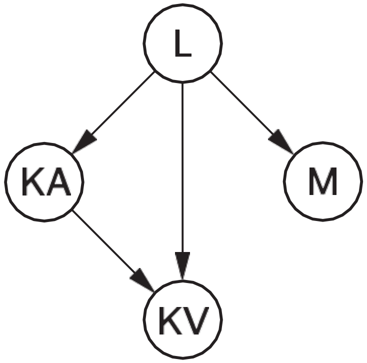
Step 9 – Node L becomes parent node of M.



Step 10 – Adding KA does not increase overall Network quality.



Step 11 – Node L becomes parent node of KA.



Result

State of the Art Procedure for Structure Learning

Step 1 Feature Selection

To support feature selection, the p-value of the test for marginal independence is computed.
The p-value is the tail probability under the independence assumption.

The higher the value the more likely the nodes are to be independent.

Step 2 Structure constraints

Specification of any known dependences or independences in the data set

Step 3 Structure Learning

NPC, PC, Greedy search and score, Chow-Liu Tree, Rebane Pearl Polytree,
Tree augmented Naive Bayes mit Scores AIC, BIC

Step 4 Structure Uncertainty

The structure learning algorithms (e.g. NPC algorithm) contains ambiguous regions (i.e., groups of inter-dependent uncertain links) and/or other undirected links. There are intuitive graphical interfaces for resolving these structural uncertainties.

Step 5 Data Dependences

Detect relative strenght of the dependences found in the data, use these information