## A Brief Introduction to Graphical Models and How to Learn Them from Data

Christian Borgelt

Dept. of Knowledge Processing and Language Engineering
Otto-von-Guericke-University of Magdeburg
Universitätsplatz 2, D-39106 Magdeburg, Germany

E-mail: `borgelt@iws.cs.uni-magdeburg.de`
WWW: `http://fuzzy.cs.uni-magdeburg.de/~borgelt`

## Overview

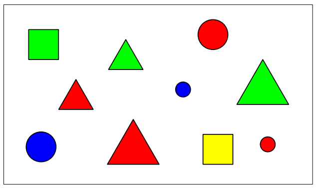
- **Graphical Models: Core Ideas and Notions**
- **A Simple Example: How does it work in principle?**
- **Conditional Independence Graphs**
  - conditional independence and the graphoid axioms
  - separation in (directed and undirected) graphs
  - decomposition/factorization of distributions
- **Evidence Propagation in Graphical Models**
- **Building Graphical Models**
- **Learning Graphical Models from Data**
  - quantitative (parameter) and qualitative (structure) learning
  - evaluation measures and search methods
  - learning by conditional independence tests
  - learning by measuring the strength of marginal dependences
- **Summary**

## Graphical Models: Core Ideas and Notions

- **Decomposition:** Under certain conditions a distribution $\delta$ (e.g. a probability distribution) on a multi-dimensional domain, which encodes *prior* or *generic knowledge* about this domain, can be decomposed into a set $\{\delta_1, \ldots, \delta_s\}$ of (overlapping) distributions on lower-dimensional subspaces.
- **Simplified Reasoning:** If such a decomposition is possible, it is sufficient to know the distributions on the subspaces to draw all inferences in the domain under consideration that can be drawn using the original distribution $\delta$.
- Such a decomposition can nicely be represented as a **graph** (in the sense of graph theory), and therefore it is called a **Graphical Model**.
- The graphical representation
  - encodes **conditional independences** that hold in the distribution,
  - describes a **factorization** of the probability distribution,
  - indicates how **evidence propagation** has to be carried out.

## A Simple Example

**Example World**

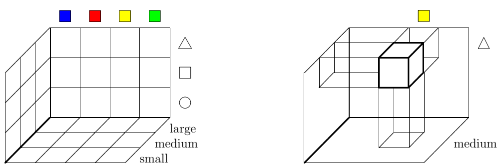


**Relation**

| color | shape | size |
|---|---|---|
| ■ | ○ | small |
| ■ | ○ | medium |
| ■ | ○ | small |
| ■ | ○ | medium |
| ■ | △ | medium |
| ■ | △ | large |
| ■ | □ | medium |
| ■ | □ | medium |
| ■ | △ | medium |
| ■ | △ | large |

- 10 simple geometrical objects, 3 attributes.
- One object is chosen at random and examined.
- Inferences are drawn about the unobserved attributes.

## The Reasoning Space



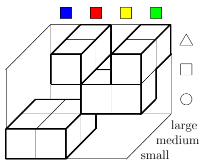

- The reasoning space consists of a finite set $\Omega$ of world states.
- The world states are described by a set of attributes $A_i$, whose domains $\{a_1^{(i)}, \ldots, a_{k_i}^{(i)}\}$ can be seen as sets of propositions or events.
- The events in a domain are mutually exclusive and exhaustive.
- The reasoning space is assumed to contain the true, but unknown state $\omega_0$.

## The Relation in the Reasoning Space

**Relation**

| color | shape | size |
|---|---|---|
| ■ | ○ | small |
| ■ | ○ | medium |
| ■ | ○ | small |
| ■ | ○ | medium |
| ■ | △ | medium |
| ■ | △ | large |
| ■ | □ | medium |
| ■ | □ | medium |
| ■ | △ | medium |
| ■ | △ | large |

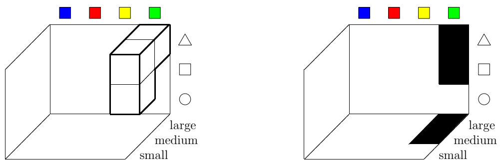
**Relation in the Reasoning Space**



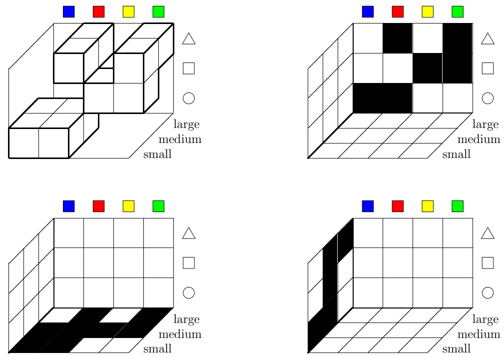

Each cube represents one tuple.

## Reasoning

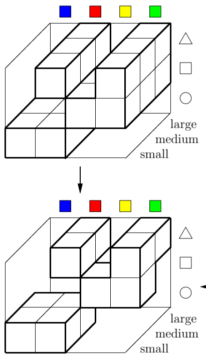
- Let it be known (e.g. from an observation) that the given object is green. This information considerably reduces the space of possible value combinations.
- From the prior knowledge it follows that the given object must be
  - either a triangle or a square and
  - either medium or large.



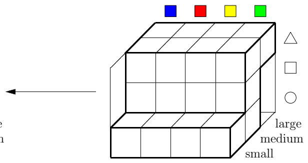

## Prior Knowledge and Its Projections




## Cylindrical Extensions and Their Intersection

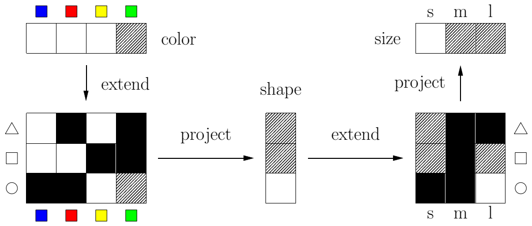



Intersecting the cylindrical extensions of the projection to the subspace formed by color and shape and of the projection to the subspace formed by shape and size yields the original three-dimensional relation.

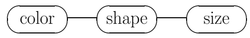



## Reasoning with Projections

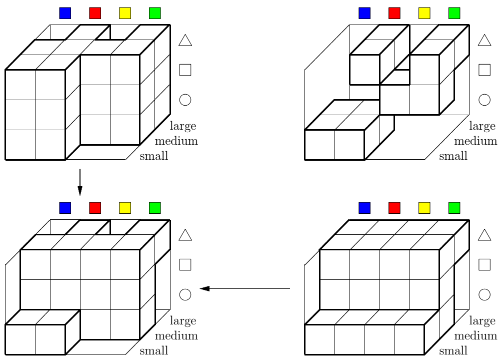
The same result can be obtained using only the projections to the subspaces without reconstructing the original three-dimensional space:



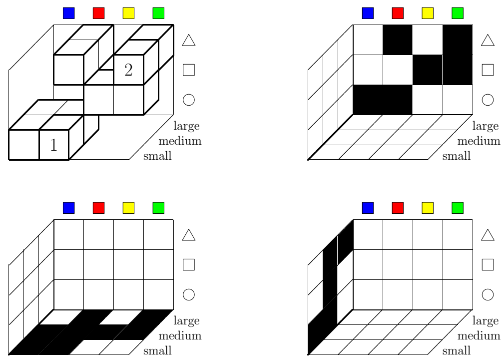

This justifies a network representation:



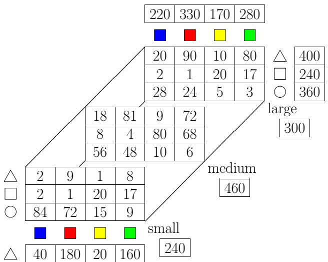

## Using other Projections




## Is Decomposition Always Possible?



## A Probability Distribution




all numbers in parts per 1000

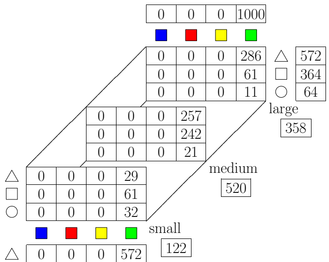
| | | | | |
|---|---|---|---|---|
| △ | 40 | 180 | 20 | 160 |
| □ | 12 | 6 | 120 | 102 |
| ○ | 168 | 144 | 30 | 18 |

| | s | m | l |
|---|---|---|---|
| △ | 20 | 180 | 200 |
| □ | 40 | 160 | 40 |
| ○ | 180 | 120 | 60 |

| | | | | |
|---|---|---|---|---|
| large | 50 | 115 | 35 | 100 |
| medium | 82 | 133 | 99 | 146 |
| small | 88 | 82 | 36 | 34 |

- The numbers state the probability of the corresponding value combination.

## Reasoning: Computing Conditional Probabilities




all numbers in parts per 1000

| | | | | |
|---|---|---|---|---|
| △ | 0 | 0 | 0 | 572 |
| □ | 0 | 0 | 0 | 364 |
| ○ | 0 | 0 | 0 | 64 |

| | s | m | l |
|---|---|---|---|
| △ | 29 | 257 | 286 |
| □ | 61 | 242 | 61 |
| ○ | 32 | 21 | 11 |

| | | | | |
|---|---|---|---|---|
| large | 0 | 0 | 0 | 358 |
| medium | 0 | 0 | 0 | 531 |
| small | 0 | 0 | 0 | 111 |

- Using the information that the given object is green.

## Probabilistic Decomposition

- As for relational networks, the three-dimensional probability distribution can be decomposed into projections to subspaces, namely the marginal distribution on the subspace formed by color and shape and the marginal distribution on the subspace formed by shape and size.
- The original probability distribution can be reconstructed from the marginal distributions using the following formulae $\forall i, j, k$ :

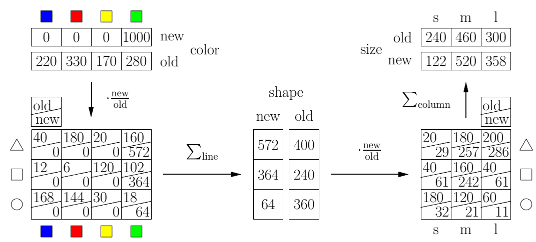
$$\begin{aligned} P\Big(a_i^{(\text{color})}, a_j^{(\text{shape})}, a_k^{(\text{size})}\Big) &= P\Big(a_i^{(\text{color})}, a_j^{(\text{shape})}\Big) \cdot P\Big(a_k^{(\text{size})} \,\Big|\, a_j^{(\text{shape})}\Big) \\ &= P\Big(a_i^{(\text{color})}, a_j^{(\text{shape})}\Big) \cdot \frac{P\Big(a_j^{(\text{shape})}, a_k^{(\text{size})}\Big)}{P\Big(a_j^{(\text{shape})}\Big)} \end{aligned}$$

- These equations express the *conditional independence* of attributes *color* and *size* given the attribute *shape*, since they only hold if $\forall i, j, k$ :

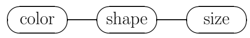
$$P\Big(a_k^{(\text{size})} \,\Big|\, a_j^{(\text{shape})}\Big) = P\Big(a_k^{(\text{size})} \,\Big|\, a_i^{(\text{color})}, a_j^{(\text{shape})}\Big)$$

## Reasoning with Projections

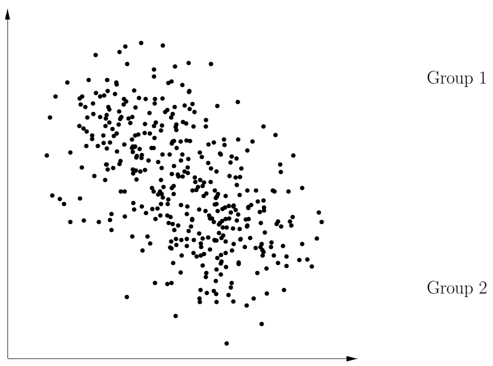
Again the same result can be obtained using only projections to subspaces (marginal distributions):



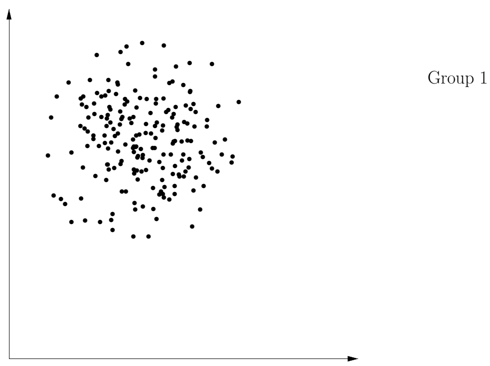

This justifies a network representation:



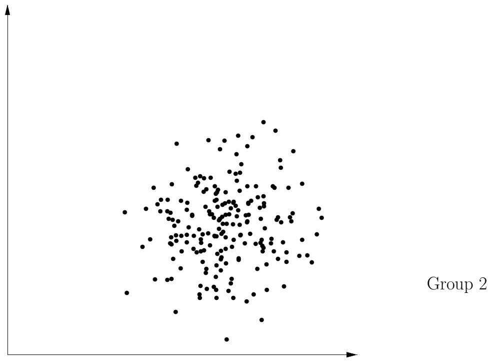

## Conditional Independence: An Example




## Conditional Independence: An Example



## Conditional Independence: An Example




## Conditional Independence

**Definition:** Let $\Omega$ be a (finite) sample space, $P$ a probability measure on $\Omega$, and $A$, $B$, and $C$ attributes with respective domains $\mathrm{dom}(A)$, $\mathrm{dom}(B)$, and $\mathrm{dom}(C)$. $A$ and $B$ are called **conditionally probabilistically independent** given $C$, written $A \perp\!\!\!\perp_P B \mid C$, iff

$$\forall a \in \mathrm{dom}(A) : \forall b \in \mathrm{dom}(B) : \forall c \in \mathrm{dom}(C) :$$
$$P(A = a, B = b \mid C = c) = P(A = a \mid C = c) \cdot P(B = b \mid C = c)$$

Equivalent formula:

$$\forall a \in \mathrm{dom}(A) : \forall b \in \mathrm{dom}(B) : \forall c \in \mathrm{dom}(C) :$$
$$P(A = a \mid B = b, C = c) = P(A = a \mid C = c)$$

- Conditional independences make it possible to consider parts of a probability distribution independent of others.
- Therefore it is plausible that a set of conditional independences may enable a decomposition of a joint probability distribution.

## (Semi-)Graphoid Axioms

**Definition:** Let $V$ be a set of (mathematical) objects and $(\cdot \perp\!\!\!\perp \cdot \mid \cdot)$ a three-place relation of subsets of $V$. Furthermore, let $W$, $X$, $Y$, and $Z$ be four disjoint subsets of $V$. The four statements

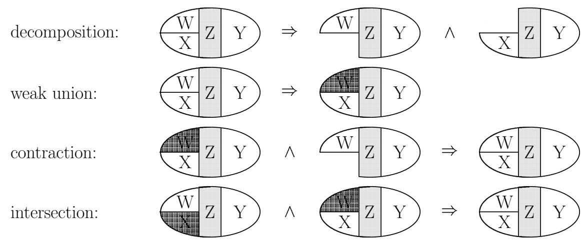
symmetry: $(X \perp\!\!\!\perp Y \mid Z) \Rightarrow (Y \perp\!\!\!\perp X \mid Z)$
decomposition: $(W \cup X \perp\!\!\!\perp Y \mid Z) \Rightarrow (W \perp\!\!\!\perp Y \mid Z) \wedge (X \perp\!\!\!\perp Y \mid Z)$
weak union: $(W \cup X \perp\!\!\!\perp Y \mid Z) \Rightarrow (X \perp\!\!\!\perp Y \mid Z \cup W)$
contraction: $(X \perp\!\!\!\perp Y \mid Z \cup W) \wedge (W \perp\!\!\!\perp Y \mid Z) \Rightarrow (W \cup X \perp\!\!\!\perp Y \mid Z)$

are called the **semi-graphoid axioms**. A three-place relation $(\cdot \perp\!\!\!\perp \cdot \mid \cdot)$ that satisfies the semi-graphoid axioms for all $W$, $X$, $Y$, and $Z$ is called a **semi-graphoid**. The above four statements together with

intersection: $(W \perp\!\!\!\perp Y \mid Z \cup X) \wedge (X \perp\!\!\!\perp Y \mid Z \cup W) \Rightarrow (W \cup X \perp\!\!\!\perp Y \mid Z)$

are called the **graphoid axioms**. A three-place relation $(\cdot \perp\!\!\!\perp \cdot \mid \cdot)$ that satisfies the graphoid axioms for all $W$, $X$, $Y$, and $Z$ is called a **graphoid**.

## Illustration of the (Semi-)Graphoid Axioms




- Similar to the properties of **separation in graphs**.
- **Idea:** Represent conditional independence by separation in graphs.

## Separation in Graphs

**Definition:** Let $G = (V, E)$ be an undirected graph and $X$, $Y$, and $Z$ three disjoint subsets of nodes. $Z$ **u-separates** $X$ and $Y$ in $G$, written $\langle X \mid Z \mid Y\rangle_G$, iff all paths from a node in $X$ to a node in $Y$ contain a node in $Z$. A path that contains a node in $Z$ is called **blocked** (by $Z$), otherwise it is called **active**.

**Definition:** Let $\vec{G} = (V, \vec{E})$ be a directed acyclic graph and $X, Y$, and $Z$ three disjoint subsets of nodes. $Z$ **d-separates** $X$ and $Y$ in $\vec{G}$, written $\langle X \mid Z \mid Y\rangle_{\vec{G}}$, iff there is *no* path from a node in $X$ to a node in $Y$ along which the following two conditions hold:

1. every node with converging edges either is in $Z$ or has a descendant in $Z$,
2. every other node is not in $Z$.

A path satisfying the two conditions above is said to be **active**, otherwise it is said to be **blocked** (by $Z$).

## Conditional (In)Dependence Graphs

**Definition:** Let $(\cdot \perp\!\!\!\perp_\delta \cdot \mid \cdot)$ be a three-place relation representing the set of conditional independence statements that hold in a given distribution $\delta$ over a set $U$ of attributes. An undirected graph $G = (U, E)$ over $U$ is called a **conditional dependence graph** or a **dependence map** w.r.t. $\delta$, iff for all disjoint subsets $X, Y, Z \subseteq U$ of attributes

$$X \perp\!\!\!\perp_\delta Y \mid Z \Rightarrow \langle X \mid Z \mid Y\rangle_G,$$

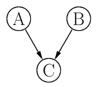
i.e., if $G$ captures by $u$-separation all (conditional) independences that hold in $\delta$ and thus represents only valid (conditional) dependences. Similarly, $G$ is called a **conditional independence graph** or an **independence map** w.r.t. $\delta$, iff for all disjoint subsets $X, Y, Z \subseteq U$ of attributes

$$\langle X \mid Z \mid Y\rangle_G \Rightarrow X \perp\!\!\!\perp_\delta Y \mid Z,$$

i.e., if $G$ captures by $u$-separation only (conditional) independences that are valid in $\delta$. $G$ is said to be a **perfect map** of the conditional (in)dependences in $\delta$, if it is both a dependence map and an independence map.

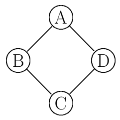
## Limitations of Graph Representations

**Perfect directed map, no perfect undirected map:**




| $p_{ABC}$ | $A=a_1$ | | $A=a_2$ | |
|---|---|---|---|---|
| | $B=b_1$ | $B=b_2$ | $B=b_1$ | $B=b_2$ |
| $C=c_1$ | $^4/_{24}$ | $^3/_{24}$ | $^3/_{24}$ | $^2/_{24}$ |
| $C=c_2$ | $^2/_{24}$ | $^3/_{24}$ | $^3/_{24}$ | $^4/_{24}$ |

**Perfect undirected map, no perfect directed map:**

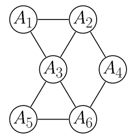



| $p_{ABCD}$ | | $A=a_1$ | | $A=a_2$ | |
|---|---|---|---|---|---|
| | | $B=b_1$ | $B=b_2$ | $B=b_1$ | $B=b_2$ |
| $C=c_1$ | $D=d_1$ | $^1/_{47}$ | $^1/_{47}$ | $^1/_{47}$ | $^2/_{47}$ |
| | $D=d_2$ | $^1/_{47}$ | $^1/_{47}$ | $^2/_{47}$ | $^4/_{47}$ |
| $C=c_2$ | $D=d_1$ | $^1/_{47}$ | $^2/_{47}$ | $^1/_{47}$ | $^4/_{47}$ |
| | $D=d_2$ | $^2/_{47}$ | $^4/_{47}$ | $^4/_{47}$ | $^{16}/_{47}$ |

## Undirected Graphs and Decompositions

**Definition:** A probability distribution $p_V$ over a set $V$ of variables is called **decomposable** or **factorizable w.r.t. an undirected graph** $G = (V, E)$ over $V$ iff it can be written as a product of nonnegative functions on the maximal cliques of $G$. That is, let $\mathcal{M}$ be a family of subsets of variable, such that the subgraphs of $G$ induced by the sets $M \in \mathcal{M}$ are the maximal cliques of $G$. Then there exist functions $\phi_M : \mathcal{E}_M \to \mathbb{R}_0^+$, $M \in \mathcal{M}$, $\forall a_1 \in \mathrm{dom}(A_1) : \ldots \forall a_n \in \mathrm{dom}(A_n) :$

$$p_V\Big(\bigwedge_{A_i \in V} A_i = a_i\Big) = \prod_{M \in \mathcal{M}} \phi_M\Big(\bigwedge_{A_i \in M} A_i = a_i\Big).$$

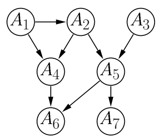



$$\begin{aligned} & p_V(A_1 = a_1, \ldots, A_6 = a_6) \\ = \; & \phi_{A_1A_2A_3}(A_1 = a_1, A_2 = a_2, A_3 = a_3) \\ \cdot \; & \phi_{A_3A_5A_6}(A_3 = a_3, A_5 = a_5, A_6 = a_6) \\ \cdot \; & \phi_{A_2A_4}(A_2 = a_2, A_4 = a_4) \\ \cdot \; & \phi_{A_4A_6}(A_4 = a_4, A_6 = a_6). \end{aligned}$$

## Directed Acyclic Graphs and Decompositions

**Definition:** A probability distribution $p_U$ over a set $U$ of attributes is called **decomposable** or **factorizable w.r.t. a directed acyclic graph** $\vec{G} = (U, \vec{E})$ over $U$, iff it can be written as a product of the conditional probabilities of the attributes given their parents in $\vec{G}$, i.e., iff

$$\forall a_1 \in \mathrm{dom}(A_1) : \ldots \forall a_n \in \mathrm{dom}(A_n) :$$
$$p_U\Big(\bigwedge_{A_i \in U} A_i = a_i\Big) = \prod_{A_i \in U} P\Big(A_i = a_i \,\Big|\, \bigwedge_{A_j \in \mathrm{parents}_{\vec{G}}(A_i)} A_j = a_j\Big).$$




$$\begin{aligned} & P(A_1 = a_1, \ldots, A_7 = a_7) \\ = \; & P(A_1 = a_1) \cdot P(A_2 = a_2 \mid A_1 = a_1) \cdot P(A_3 = a_3) \\ \cdot \; & P(A_4 = a_4 \mid A_1 = a_1, A_2 = a_2) \\ \cdot \; & P(A_5 = a_5 \mid A_2 = a_2, A_3 = a_3) \\ \cdot \; & P(A_6 = a_6 \mid A_4 = a_4, A_5 = a_5) \\ \cdot \; & P(A_7 = a_7 \mid A_5 = a_5). \end{aligned}$$

## Conditional Independence Graphs and Decompositions

**Core Theorem of Graphical Models:**
Let $p_V$ be a strictly positive probability distribution on a set $V$ of (discrete) variables. A directed or undirected graph $G = (V, E)$ is a conditional independence graph w.r.t. $p_V$ if and only if $p_V$ is factorizable w.r.t. $G$.

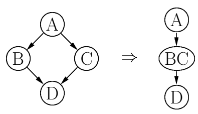
**Definition:** A **Markov network** is an undirected conditional independence graph of a probability distribution $p_V$ together with the family of positive functions $\phi_M$ of the factorization induced by the graph.

**Definition:** A **Bayesian network** is a directed conditional independence graph of a probability distribution $p_U$ together with the family of conditional probabilities of the factorization induced by the graph.

- Sometimes the conditional independence graph is required to be minimal.
- For correct evidence propagation it is not required that the graph is minimal. Evidence propagation may just be less efficient than possible.

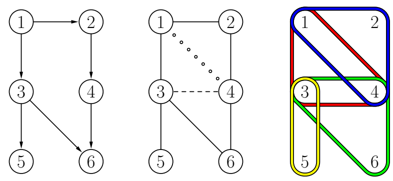
## Evidence Propagation in Graphical Models

- It is fairly easy to derive evidence propagation formulae for singly connected networks (undirected trees, directed polytrees).
- However, in practice, there are often be multiple paths connecting two variables, all of which may be needed for proper evidence propagation.
- Propagating evidence along all paths can lead to wrong results (multiple incorporation of the same evidence).
- **Solution** (one out of many):
  Turn the graph into a **singly connected structure**.

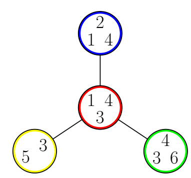



Merging attributes can make the polytree algorithm applicable in multiply connected networks.

## Triangulation and Join Tree Construction



original graph | triangulated moral graph | maximal cliques



join tree

- A singly connected structure is obtained by triangulating the graph and then forming a tree of maximal cliques, the so-called **join tree**.
- For evidence propagation a join tree is enhanced by so-called **separators** on the edges, which are intersection of the connected nodes → **junction tree**.

## Graph Triangulation

**Algorithm:** (graph triangulation)

**Input:** An undirected graph $G = (V, E)$.
**Output:** A triangulated undirected graph $G' = (V, E')$ with $E' \supseteq E$.

1. Compute an ordering of the nodes of the graph using *maximum cardinality search*, i.e., number the nodes from 1 to $n = |V|$, in increasing order, always assigning the next number to the node having the largest set of previously numbered neighbors (breaking ties arbitrarily).
2. From $i = n$ to $i = 1$ recursively fill in edges between any nonadjacent neighbors of the node numbered $i$ having lower ranks than $i$ (including neighbors linked to the node numbered $i$ in previous steps). If no edges are added, then the original graph is chordal; otherwise the new graph is chordal.

## Join Tree Construction

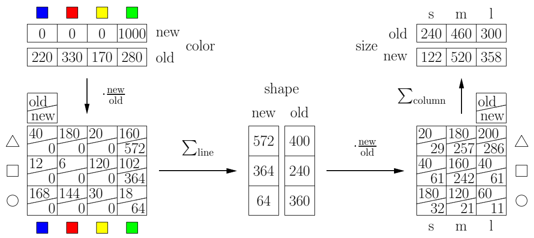
**Algorithm:** (join tree construction)

**Input:** A triangulated undirected graph $G = (V, E)$.
**Output:** A join tree $G' = (V', E')$ for $G$.

1. Determine a numbering of the nodes of $G$ using maximum cardinality search.
2. Assign to each clique the maximum of the ranks of its nodes.
3. Sort the cliques in ascending order w.r.t. the numbers assigned to them.
4. Traverse the cliques in ascending order and for each clique $C_i$ choose from the cliques $C_1, \ldots, C_{i-1}$ preceding it the clique with which it has the largest number of nodes in common (breaking ties arbitrarily).

## Reasoning in Join/Junction Trees

- Reasoning in join trees follows the same lines as shown in the simple example.
- Multiple pieces of evidence from different branches may be incorporated into a distribution before continuing by summing/marginalizing.




## Building Graphical Models: Causal Modeling

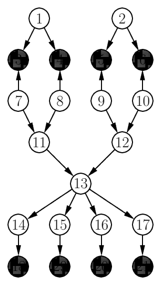
**Manual creation of a reasoning system based on a graphical model:**

causal model of given domain
↓ **heuristics!**
conditional independence graph
↓ formally provable
decomposition of the distribution
↓ formally provable
evidence propagation scheme

- Problem: strong assumptions about the statistical effects of causal relations.

## Probabilistic Graphical Models: An Example

**Danish Jersey Cattle Blood Type Determination**




21 attributes:
1 – dam correct?
2 – sire correct?
3 – stated dam ph.gr. 1
4 – stated dam ph.gr. 2
5 – stated sire ph.gr. 1
6 – stated sire ph.gr. 2
7 – true dam ph.gr. 1
8 – true dam ph.gr. 2
9 – true sire ph.gr. 1
10 – true sire ph.gr. 2
11 – offspring ph.gr. 1
12 – offspring ph.gr. 2
13 – offspring genotype
14 – factor 40
15 – factor 41
16 – factor 42
17 – factor 43
18 – lysis 40
19 – lysis 41
20 – lysis 42
21 – lysis 43

The grey nodes correspond to observable attributes.

## Danish Jersey Cattle Blood Type Determination

- Full 21-dimensional domain has $2^6 \cdot 3^{10} \cdot 6 \cdot 8^4 = 92\,876\,046\,336$ possible states.
- Bayesian network requires only 306 conditional probabilities.
- Example of a conditional probability table (attributes 2, 9, and 5):

| sire correct | true sire phenogroup 1 | stated sire phenogroup 1 F1 | V1 | V2 |
|---|---|---|---|---|
| yes | F1 | 1 | 0 | 0 |
| yes | V1 | 0 | 1 | 0 |
| yes | V2 | 0 | 0 | 1 |
| no | F1 | 0.58 | 0.10 | 0.32 |
| no | V1 | 0.58 | 0.10 | 0.32 |
| no | V2 | 0.58 | 0.10 | 0.32 |

## Learning Graphical Models from Data

Given: A database of sample cases from a domain of interest.
Desired: A (good) graphical model of the domain of interest.

- **Quantitative or Parameter Learning**
  - The structure of the conditional independence graph is known.
  - Conditional or marginal distributions have to be estimated by standard statistical methods. (*parameter estimation*)
- **Qualitative or Structural Learning**
  - The structure of the conditional independence graph is not known.
  - A good graph has to be selected from the set of all possible graphs. (*model selection*)
  - Tradeoff between model complexity and model accuracy.

## Danish Jersey Cattle Blood Type Determination

A fraction of the database of sample cases:

```
y y f1 v2 f1 v2 f1 v2 f1 v2 v2 v2 v2v2 n y n y 0 6 0 6
y y f1 v2 ** ** f1 v2 ** ** ** ** f1v2 y y n y 7 6 0 7
y y f1 v2 f1 f1 f1 v2 f1 f1 f1 f1 f1f1 y y n n 7 7 0 0
y y f1 v2 f1 f1 f1 v2 f1 f1 f1 f1 f1f1 y y n n 7 7 0 0
y y f1 v2 f1 v1 f1 v2 f1 v1 v2 f1 f1v2 y y n y 7 7 0 7
y y f1 f1 ** ** f1 f1 ** ** f1 f1 f1f1 y y n n 6 6 0 0
y y f1 v1 ** ** f1 v1 ** ** v1 v2 v1v2 n y y y 0 5 4 5
y y f1 v2 f1 v1 f1 v2 f1 v1 f1 v1 f1v1 y y y y 7 7 6 7
                 ⋮                            ⋮
```

- 21 attributes
- 500 real world sample cases
- A lot of missing values (indicated by `**`)

## Naive Bayes Classifiers: Star-like Networks

- A naive Bayes classifier is a Bayesian network with a star-like structure.
- The class attribute is the only unconditioned attribute.
- All other attributes are conditioned on the class only.
- The classifier may be augmented by additional edges between the attributes.


$$
\begin{aligned}
P(C = c_i, A_1 = a_1, \ldots, A_n = a_n) &= P(C = c_i \mid A_1 = a_1, \ldots) \cdot P(A_1 = a_1, \ldots) \\
&= P(C = c_i) \cdot \prod_{j=1}^{n} P(A_j = a_j \mid C = c_i)
\end{aligned}
$$

## Naive Bayes Classifiers

- Consequence: Manageable amount of data to store.
  Store distributions $P(C = c_i)$ and $\forall 1 \leq j \leq m : P(A_j = a_j \mid C = c_i)$.
- Classification: Compute $P(C = c_i) \cdot \prod_{j=1}^{n} P(A_j = a_j \mid C = c_i)$ for all $c_i$ and predict the class $c_i$ for which this value is largest.

**Estimation of Probabilities:**

- Here: restriction to symbolic attributes.

$$\hat{P}(A_j = a_j \mid C = c_i) = \frac{\#(A_j = a_j, C = c_i) + \gamma}{\#(C = c_i) + n_{A_j}\gamma}$$

$\gamma$ is called **Laplace correction**.

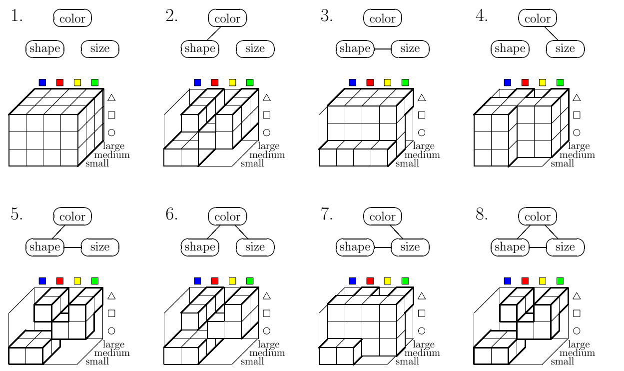
$\gamma = 0$: Maximum likelihood estimation.

Common choices: $\gamma = 1$ or $\gamma = \frac{1}{2}$.

## Learning the Structure of Graphical Models from Data

- **Test whether a distribution is decomposable w.r.t. a given graph.**
  This is the most direct approach. It is not bound to a graphical representation, but can also be carried out w.r.t. other representations of the set of subspaces to be used to compute the (candidate) decomposition of the given distribution.
- **Find an independence map by conditional independence tests.**
  This approach exploits the theorems that connect conditional independence graphs and graphs that represent decompositions. It has the advantage that a single conditional independence test, if it fails, can exclude several candidate graphs.
- **Find a suitable graph by measuring the strength of dependences.**
  This is a heuristic, but often highly successful approach, which is based on the frequently valid assumption that in a conditional independence graph an attribute is more strongly dependent on adjacent attributes than on attributes that are not directly connected to them.

## Direct Test for Decomposability: Relational



## Comparing Probability Distributions

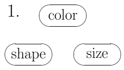
**Definition:** Let $p_1$ and $p_2$ be two strictly positive probability distributions on the same set $\mathcal{E}$ of events. Then

$$I_{\mathrm{KLdiv}}(p_1, p_2) = \sum_{E \in \mathcal{E}} p_1(E) \log_2 \frac{p_1(E)}{p_2(E)}$$

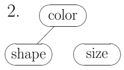
is called the **Kullback-Leibler information divergence** of $p_1$ and $p_2$.

- 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.

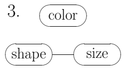
## Direct Test for Decomposability: Probabilistic

1. 

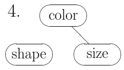
0.640
−5041

2. 

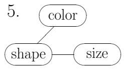
0.211
−4612

3. 

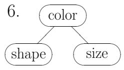
0.429
−4830

4. 

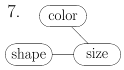
0.590
−4991

5. 

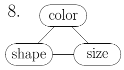
0
−4401

6. 

0.161
−4563

7. 

0.379
−4780

8. 

0
−4401

Upper numbers: The Kullback-Leibler information divergence of the original distribution and its approximation.

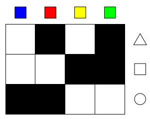
Lower numbers: The binary logarithms of the probability of an example database (log-likelihood of data).

## Evaluation Measures and Search Methods

- 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.
- Therefore all learning algorithms consist of
  an **evaluation measure** (scoring function), e.g.
  - Hartley information gain (relational networks)
  - Shannon information gain, K2 metric (probabilistic networks)

  and a (heuristic) **search method**, e.g.
  - conditional independence search
  - greedy search (spanning tree or K2 algorithm)
  - guided random search (simulated annealing, genetic algorithms)

## Measuring Dependence Strength: Relational




Hartley information needed to determine

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

**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

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

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

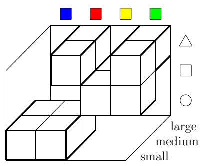
## Marginal and Conditional Independence Tests

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

| attributes | relative number of possible value combinations | Hartley information gain |
|---|---|---|
| color, shape | $\frac{6}{3 \cdot 4} = \frac{1}{2} = 50\%$ | $\log_2 3 + \log_2 4 - \log_2 6 = 1$ |
| color, size | $\frac{8}{3 \cdot 4} = \frac{2}{3} \approx 67\%$ | $\log_2 3 + \log_2 4 - \log_2 8 \approx 0.58$ |
| shape, size | $\frac{5}{3 \cdot 3} = \frac{5}{9} \approx 56\%$ | $\log_2 3 + \log_2 3 - \log_2 5 \approx 0.85$ |

- In order to test for (approximate) **conditional independence**:
  - Compute the Hartley information gain for each possible instantiation of the conditioning attributes.
  - Aggregate the result over all possible instantiations, for instance, by simply averaging them.

## Conditional Independence Tests: Relational




| color | Hartley information gain |
|---|---|
| blue ■ | $\log_2 1 + \log_2 2 - \log_2 2 = 0$ |
| red ■ | $\log_2 2 + \log_2 3 - \log_2 4 \approx 0.58$ |
| yellow ■ | $\log_2 1 + \log_2 1 - \log_2 1 = 0$ |
| green ■ | $\log_2 2 + \log_2 2 - \log_2 2 = 1$ |
| | average: $\approx 0.40$ |

| shape | Hartley information gain |
|---|---|
| △ | $\log_2 2 + \log_2 2 - \log_2 4 = 0$ |
| □ | $\log_2 2 + \log_2 1 - \log_2 2 = 0$ |
| ○ | $\log_2 2 + \log_2 2 - \log_2 4 = 0$ |
| | average: $= 0$ |

| size | Hartley information gain |
|---|---|
| large | $\log_2 2 + \log_2 1 - \log_2 2 = 0$ |
| medium | $\log_2 4 + \log_2 3 - \log_2 6 = 1$ |
| small | $\log_2 2 + \log_2 1 - \log_2 2 = 0$ |
| | average: $\approx 0.33$ |

## Measuring Dependence Strength: Probabilistic

**Mutual Information / Cross Entropy / Information Gain**

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

$$I_{\text{gain}}(A,B) = \overbrace{H(A)}^{} - \overbrace{H(A \mid B)}^{}$$

$$= \underbrace{-\sum_{i=1}^{n_A} p_{i.} \log_2 p_{i.}}_{H(A)} - \underbrace{\sum_{j=1}^{n_B} p_{.j} \left( -\sum_{i=1}^{n_A} p_{i|j} \log_2 p_{i|j} \right)}_{H(A \mid B)}$$

| | |
|---|---|
| $H(A)$ | 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* |

## Interpretation of Shannon Entropy

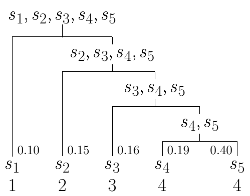
- Let $S = \{s_1, \ldots, s_n\}$ be a finite set of alternatives having positive probabilities $P(s_i)$, $i = 1, \ldots, n$, satisfying $\sum_{i=1}^{n} P(s_i) = 1$.
- **Shannon Entropy:**
$$H(S) = -\sum_{i=1}^{n} P(s_i) \log_2 P(s_i)$$
- Intuitively: **Expected number of yes/no questions that have to be asked in order to determine the obtaining alternative.**
  - Suppose there is an oracle, which knows the obtaining alternative, but responds only if the question can be answered with "yes" or "no".
  - A better question scheme than asking for one alternative after the other can easily be found: Divide the set into two subsets of about equal size.
  - Ask for containment in an arbitrarily chosen subset.
  - Apply this scheme recursively → number of questions bounded by $\lceil \log_2 n \rceil$.

## Question/Coding Schemes

$P(s_1) = 0.10$, $P(s_2) = 0.15$, $P(s_3) = 0.16$, $P(s_4) = 0.19$, $P(s_5) = 0.40$

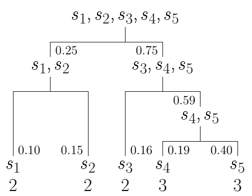
Shannon entropy: $-\sum_i P(s_i) \log_2 P(s_i) = 2.15$ bit/symbol

**Linear Traversal**




Code length: 3.24 bit/symbol
Code efficiency: 0.664

**Equal Size Subsets**




Code length: 2.59 bit/symbol
Code efficiency: 0.830

## Question/Coding Schemes

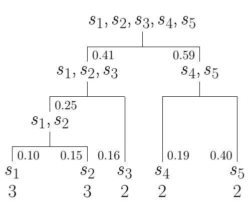
- Splitting into subsets of about equal size can lead to a bad arrangement of the alternatives into subsets → high expected number of questions.
- Good question schemes take the probability of the alternatives into account.
- **Shannon-Fano Coding** (1948)
  - Build the question/coding scheme top-down.
  - Sort the alternatives w.r.t. their probabilities.
  - Split the set so that the subsets have about equal *probability* (splits must respect the probability order of the alternatives).
- **Huffman Coding** (1952)
  - Build the question/coding scheme bottom-up.
  - Start with one element sets.
  - Always combine those two sets that have the smallest probabilities.

## Question/Coding Schemes

$P(s_1) = 0.10$, $P(s_2) = 0.15$, $P(s_3) = 0.16$, $P(s_4) = 0.19$, $P(s_5) = 0.40$

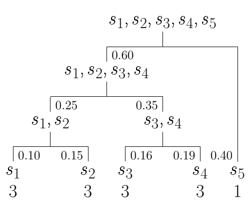
Shannon entropy: $-\sum_i P(s_i) \log_2 P(s_i) = 2.15$ bit/symbol

**Shannon–Fano Coding** (1948)




Code length: 2.25 bit/symbol
Code efficiency: 0.955

**Huffman Coding** (1952)




Code length: 2.20 bit/symbol
Code efficiency: 0.977

## Question/Coding Schemes

- It can be shown that Huffman coding is optimal if we have to determine the obtaining alternative in a single instance.
(No question/coding scheme has a smaller expected number of questions.)
- Only if the obtaining alternative has to be determined in a sequence of (independent) situations, this scheme can be improved upon.
- Idea: Process the sequence not instance by instance, but combine two, three or more consecutive instances and ask directly for the obtaining combination of alternatives.
- Although this enlarges the question/coding scheme, the expected number of questions per identification is reduced (because each interrogation identifies the obtaining alternative for several situations).
- However, the expected number of questions per identification cannot be made arbitrarily small. Shannon showed that there is a lower bound, namely the Shannon entropy.

## Interpretation of Shannon Entropy

$$P(s_1) = \tfrac{1}{2}, \quad P(s_2) = \tfrac{1}{4}, \quad P(s_3) = \tfrac{1}{8}, \quad P(s_4) = \tfrac{1}{16}, \quad P(s_5) = \tfrac{1}{16}$$

Shannon entropy: $\quad -\sum_i P(s_i) \log_2 P(s_i) = 1.875$ bit/symbol

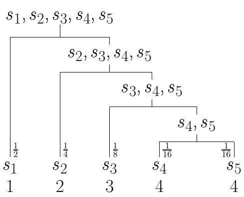
If the probability distribution allows for a perfect Huffman code (code efficiency 1), the Shannon entropy can easily be interpreted as follows:

$$-\sum_i P(s_i) \log_2 P(s_i)$$

$$= \sum_i \underbrace{P(s_i)}_{\substack{\text{occurrence}\\ \text{probability}}} \cdot \underbrace{\log_2 \frac{1}{P(s_i)}}_{\substack{\text{path length}\\ \text{in tree}}} .$$

In other words, it is the expected number of needed yes/no questions.

**Perfect Question Scheme**




Code length: 1.875 bit/symbol
Code efficiency: 1

## Mutual Information for the Example

| projection to subspace | | | | | product of marginals | | | | | mutual information |
|---|---|---|---|---|---|---|---|---|---|---|
| | ■ | ■ | ■ | ■ | | ■ | ■ | ■ | ■ | |
| △ | 40 | 180 | 20 | 160 | △ | 88 | 132 | 68 | 112 | |
| □ | 12 | 6 | 120 | 102 | □ | 53 | 79 | 41 | 67 | 0.429 bit |
| ○ | 168 | 144 | 30 | 18 | ○ | 79 | 119 | 61 | 101 | |

| | s | m | l | | s | m | l | |
|---|---|---|---|---|---|---|---|---|
| △ | 20 | 180 | 200 | △ | 96 | 184 | 120 | |
| □ | 40 | 160 | 40 | □ | 58 | 110 | 72 | 0.211 bit |
| ○ | 180 | 120 | 60 | ○ | 86 | 166 | 108 | |

| | ■ | ■ | ■ | ■ | | ■ | ■ | ■ | ■ | |
|---|---|---|---|---|---|---|---|---|---|---|
| large | 50 | 115 | 35 | 100 | large | 66 | 99 | 51 | 84 | |
| medium | 82 | 133 | 99 | 146 | medium | 101 | 152 | 78 | 129 | 0.050 bit |
| small | 88 | 82 | 36 | 34 | small | 53 | 79 | 41 | 67 | |

## Conditional Independence Tests: Probabilistic

- There are no marginal independences, although the dependence of color and size is rather weak.
- Conditional independence tests may be carried out by summing the mutual information for all instantiations of the conditioning variables:

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

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

- Since $I_{\text{mut}}(\text{color}, \text{size} \mid \text{shape}) = 0$ indicates the only conditional independence, we get the following learning result:

color — shape — size

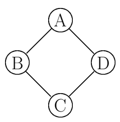
## Conditional Independence Tests: General Algorithm

**Algorithm:** (conditional independence graph construction)

1. For each pair of attributes $A$ and $B$, search for a set $S_{AB} \subseteq U \backslash \{A, B\}$ such that $A \perp\!\!\!\perp B \mid S_{AB}$ holds in $\widehat{P}$, i.e., $A$ and $B$ are independent in $\widehat{P}$ conditioned on $S_{AB}$. If there is no such $S_{AB}$, connect the attributes by an undirected edge.
2. For each pair of non-adjacent variables $A$ and $B$ with a common neighbour $C$ (i.e., $C$ is adjacent to $A$ as well as to $B$), check whether $C \in S_{AB}$.
   - If it is, continue.
   - If it is not, add arrowheads pointing to $C$, i.e., $A \rightarrow C \leftarrow B$.
3. Recursively direct all undirected edges according to the rules:
   - If for two adjacent variables $A$ and $B$ there is a strictly directed path from $A$ to $B$ not including $A \rightarrow B$, then direct the edge towards $B$.
   - If there are three variables $A$, $B$, and $C$ with $A$ and $B$ not adjacent, $B - C$, and $A \rightarrow C$, then direct the edge $C \rightarrow B$.

## Conditional Independence Tests: Drawbacks

- The conditional independence graph construction algorithm presupposes that there is a **perfect map**. If there is no perfect map, the result may be invalid.




| $p_{ABCD}$ | | $A = a_1$ | | $A = a_2$ | |
|---|---|---|---|---|---|
| | | $B = b_1$ | $B = b_2$ | $B = b_1$ | $B = b_2$ |
| $C = c_1$ | $D = d_1$ | $^1/_{47}$ | $^1/_{47}$ | $^1/_{47}$ | $^2/_{47}$ |
| | $D = d_2$ | $^1/_{47}$ | $^1/_{47}$ | $^2/_{47}$ | $^4/_{47}$ |
| $C = c_2$ | $D = d_1$ | $^1/_{47}$ | $^2/_{47}$ | $^1/_{47}$ | $^4/_{47}$ |
| | $D = d_2$ | $^2/_{47}$ | $^4/_{47}$ | $^4/_{47}$ | $^{16}/_{47}$ |

- **Independence tests of high order**, i.e., with a large number of conditions, may be necessary.
- There are approaches to mitigate these drawbacks.
  (For example, the order is restricted and all tests of higher order are assumed to fail, if all tests of lower order failed.)

## Strength of Marginal Dependences: Relational

- Learning a relational network consists in finding those subspace, for which the intersection of the cylindrical extensions of the projections to these subspaces approximates best the set of possible world states, i.e. contains as few additional states as possible.
- Since computing explicitly the intersection of the cylindrical extensions of the projections and comparing it to the original relation is too expensive, local evaluation functions are used, for instance:

| 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 minimal weight spanning tree.

## Strength of Marginal Dependences: Probabilistic

- Results for the simple example:

$$\begin{aligned} I_{\text{mut}}(\text{color}, \text{shape}) &= 0.429 \text{ bit} \\ I_{\text{mut}}(\text{shape}, \text{size}) &= 0.211 \text{ bit} \\ I_{\text{mut}}(\text{color}, \text{size}) &= 0.050 \text{ bit} \end{aligned}$$

- Applying the Kruskal algorithm yields as a learning result:

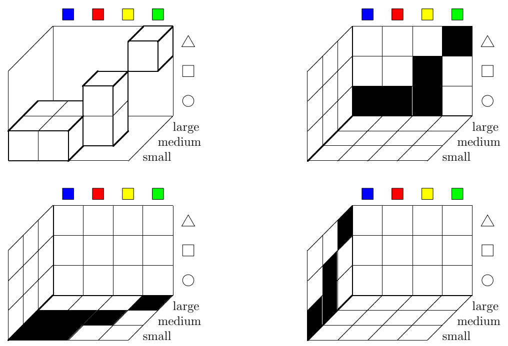
color — shape — size

- It can be shown that this approach always yields the best possible spanning tree w.r.t. Kullback-Leibler information divergence (Chow and Liu 1968).
- In an extended form this also holds for certain classes of graphs (for example, tree-augmented naive Bayes classifiers).
- For more complex graphs, the best graph need not be found (there are counterexamples, see below).

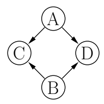
## Strength of Marginal Dependences: General Algorithms

- **Optimum Weight Spanning Tree Construction**
  - Compute an evaluation measure on all possible edges (two-dimensional subspaces).
  - Use the Kruskal algorithm to determine an optimum weight spanning tree.
- **Greedy Parent Selection** (for directed graphs)
  - Define a topological order of the attributes (to restrict the search space).
  - Compute an evaluation measure on all single attribute hyperedges.
  - For each preceding attribute (w.r.t. the topological order): add it as a candidate parent to the hyperedge and compute the evaluation measure again.
  - Greedily select a parent according to the evaluation measure.
  - Repeat the previous two steps until no improvement results from them.

## Strength of Marginal Dependences: Drawbacks




## Strength of Marginal Dependences: Drawbacks




| $p_A$ | $a_1$ | $a_2$ |
|---|---|---|
| | 0.5 | 0.5 |

| $p_B$ | $b_1$ | $b_2$ |
|---|---|---|
| | 0.5 | 0.5 |

| $p_{C\mid AB}$ | $a_1b_1$ | $a_1b_2$ | $a_2b_1$ | $a_2b_2$ |
|---|---|---|---|---|
| $c_1$ | 0.9 | 0.3 | 0.3 | 0.5 |
| $c_2$ | 0.1 | 0.7 | 0.7 | 0.5 |

| $p_{D\mid AB}$ | $a_1b_1$ | $a_1b_2$ | $a_2b_1$ | $a_2b_2$ |
|---|---|---|---|---|
| $d_1$ | 0.9 | 0.3 | 0.3 | 0.5 |
| $d_2$ | 0.1 | 0.7 | 0.7 | 0.5 |

| $p_{AD}$ | $a_1$ | $a_2$ |
|---|---|---|
| $d_1$ | 0.3 | 0.2 |
| $d_2$ | 0.2 | 0.3 |

| $p_{BD}$ | $b_1$ | $b_2$ |
|---|---|---|
| $d_1$ | 0.3 | 0.2 |
| $d_2$ | 0.2 | 0.3 |

| $p_{CD}$ | $c_1$ | $c_2$ |
|---|---|---|
| $d_1$ | 0.31 | 0.19 |
| $d_2$ | 0.19 | 0.31 |

- Greedy parent selection can lead to suboptimal results if there is more than one path connecting two attributes.
- Here: the edge $C \to D$ is selected first.

## Danish Jersey Cattle Blood Type Determination

| network | edges | params. | train | test |
|---|---|---|---|---|
| indep. | 0 | 59 | −19921.2 | −20087.2 |
| orig. | 22 | 219 | −11391.0 | −11506.1 |

**Optimum Weight Spanning Tree Construction**

| measure | edges | params. | train | test |
|---|---|---|---|---|
| $I_{\text{gain}}^{(\text{Shannon})}$ | 20.0 | 285.9 | −12122.6 | −12339.6 |
| $\chi^2$ | 20.0 | 282.9 | −12122.6 | −12336.2 |

**Greedy Parent Selection w.r.t. a Topological Order**

| measure | edges | add. | miss. | params. | train | test |
|---|---|---|---|---|---|---|
| $I_{\text{gain}}^{(\text{Shannon})}$ | 35.0 | 17.1 | 4.1 | 1342.2 | −11229.3 | −11817.6 |
| $\chi^2$ | 35.0 | 17.3 | 4.3 | 1300.8 | −11234.9 | −11805.2 |
| K2 | 23.3 | 1.4 | 0.1 | 229.9 | −11385.4 | −11511.5 |
| $L_{\text{red}}^{(\text{rel})}$ | 22.5 | 0.6 | 0.1 | 219.9 | −11389.5 | −11508.2 |

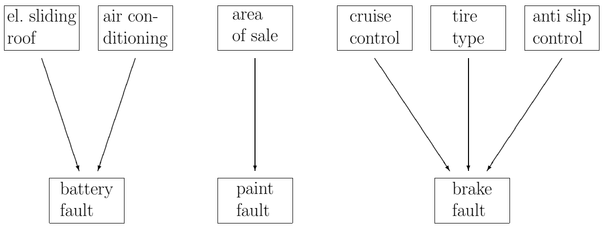
## Fields of Application (DaimlerChrysler AG)

- **Improvement of Product Quality by Finding Weaknesses**
  - Learn decision trees or inference network for vehicle properties and faults.
  - Look for unusual conditional fault frequencies.
  - Find causes for these unusual frequencies.
  - Improve construction of vehicle.
- **Improvement of Error Diagnosis in Garages**
  - Learn decision trees or inference network for vehicle properties and faults.
  - Record properties of new faulty vehicle.
  - Test for the most probable faults.

## A Simple Approach to Fault Analysis

- Check subnets consisting of an attribute and its parent attributes.
- Select subnets with highest deviation from independent distribution.

**Vehicle Properties**




**Fault Data**

## Example Subnet

**Influence of special equipment on battery faults:**

| (fictitious) frequency of battery faults | | air conditioning with | air conditioning without |
|---|---|---|---|
| electrical sliding roof | with | 8 % | 3 % |
| | without | 3 % | 2 % |

- Significant deviation from independent distribution.
- Hints to possible causes and improvements.
- Here: Larger battery may be required, if an air conditioning system. *and* an electrical sliding roof are built in.

(The dependencies and frequencies of this example are fictitious, true numbers are confidential.)

## Summary

- **Decomposition:** Under certain conditions a distribution $\delta$ (e.g. a probability distribution) on a multi-dimensional domain, which encodes *prior* or *generic knowledge* about this domain, can be decomposed into a set $\{\delta_1, \ldots, \delta_s\}$ of (overlapping) distributions on lower-dimensional subspaces.
- **Simplified Reasoning:** If such a decomposition is possible, it is sufficient to know the distributions on the subspaces to draw all inferences in the domain under consideration that can be drawn using the original distribution $\delta$.
- **Graphical Model:** The decomposition is represented by a graph (in the sense of graph theory). The edges of the graph indicate the paths along which evidence has to be propagated. Efficient and correct evidence propagation algorithms can be derived, which exploit the graph structure.
- **Learning from Data:** There are several highly successful approaches to learn graphical models from data, although all of them are based on heuristics. Exact learning methods are usually too costly.