

Non-Standard-Datenbanken

Bayesian Networks, Inference, and Learning

Prof. Dr. Ralf Möller

Universität zu Lübeck

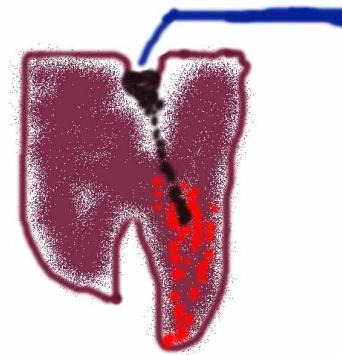
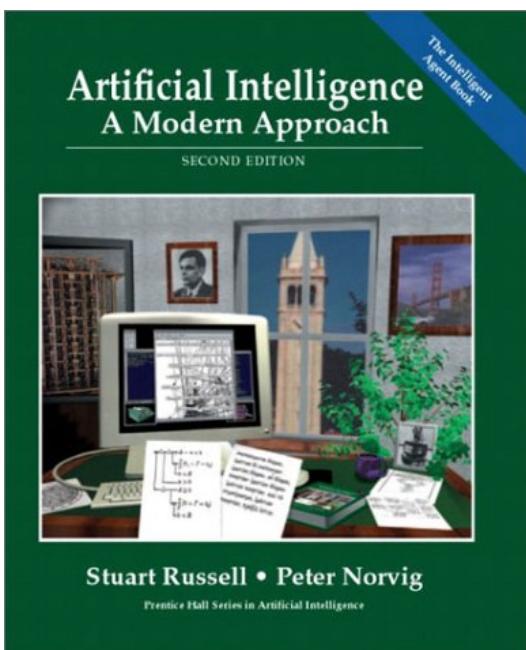
Institut für Informationssysteme



Beispiel

Zahnarzt-Problem mit vier Variablen:

- **Toothache** (Sind besagte Schmerzen wirklich Zahnschmerzen?)
- **Cavity** (Es könnte ein Loch sein?)
- **Catch** (Stahlinstrument erzeugt Testschmerz?)
- **Weather** (Wetter: sunny,rainy,cloudy,snow)



Nachfolgende
Präsentationen
enthalten Material aus
Kapitel 14
(Sektion 1 and 2)

Prior probability

- Prior or unconditional probabilities of propositions
e.g., $P(Cavity = \text{true}) = 0.1$ and $P(Weather = \text{sunny}) = 0.72$ correspond to belief prior to arrival of any (new) evidence
- Probability distribution gives values for all possible assignments:
 $P(Weather) = <0.72, 0.1, 0.08, 0.1>$
(normalized, i.e., sums to 1 because one condition must be the case)

Full joint probability distribution

- Joint probability distribution for a set of random variables gives the probability of every atomic event on those random variables
 $P(\text{Weather}, \text{Cavity})$ is a 4×2 matrix of values:

Weather =	sunny	rainy	cloudy	snow
Cavity = true	0.144	0.02	0.016	0.02
Cavity = false	0.576	0.08	0.064	0.08

- Full joint probability distribution: all random variables involved
 - $P(\text{Toothache}, \text{Catch}, \text{Cavity}, \text{Weather})$
- Every query about a domain can be answered by the full joint distribution

Discrete random variables: Notation

- $\text{Dom}(\text{Weather}) = \{\text{sunny}, \text{rainy}, \text{cloudy}, \text{snow}\}$ and $\text{Dom}(\text{Weather})$ disjoint from domain of other random variables:
 - Atomic event $\text{Weather}=\text{rainy}$ often written as rainy
 - Example: $P(\text{rainy})$, the random variable Weather is implicitly defined by the value rainy
- Boolean variable Cavity
 - Atomic event $\text{Cavity}=\text{true}$ written as cavity
 - Atomic event $\text{Cavity}=\text{false}$ written as $\neg\text{cavity}$
 - Examples: $P(\text{cavity})$ or $P(\neg\text{cavity})$

Conditional probability

- Conditional or posterior probabilities
e.g., $P(cavity | toothache) = 0.8$
or: $\langle 0.8 \rangle$
i.e., given that *toothache* is all I know
- (Notation for conditional distributions:
 $P(Cavity | Toothache)$ is a 2-element vector of 2-element vectors
- If we know more, e.g., *cavity* is also given, then we have
 $P(cavity | toothache, cavity) = 1$
- New evidence may be irrelevant, allowing simplification, e.g.,
 $P(cavity | toothache, sunny) = P(cavity | toothache) = 0.8$
- This kind of inference, sanctioned by domain knowledge, is crucial

Conditional probability

- Definition of conditional probability (in terms of uncond. prob.):
 $P(a | b) = P(a \wedge b) / P(b)$ if $P(b) > 0$
- Product rule gives an alternative formulation (\wedge is commutative):
 $P(a \wedge b) = P(a | b) P(b) = P(b | a) P(a)$
- A general version holds for whole distributions, e.g.,
 $P(Weather, Cavity) = P(Weather | Cavity) P(Cavity)$

View as a set of 4×2 equations, **not** matrix mult.

$$(1,1) P(Weather=sunny | Cavity=true) P(Cavity=true)$$

$$(1,2) P(Weather=sunny | Cavity=false) P(Cavity=false), \dots$$

- Chain rule is derived by successive application of product rule:

$$\begin{aligned} P(X_1, \dots, X_n) &= P(X_1, \dots, X_{n-1}) P(X_n | X_1, \dots, X_{n-1}) \\ &= P(X_1, \dots, X_{n-2}) P(X_{n-1} | X_1, \dots, X_{n-2}) P(X_n | X_1, \dots, X_{n-1}) \\ &= \dots \\ &= \prod_{i=1}^n P(X_i | X_1, \dots, X_{i-1}) \end{aligned}$$

Inference by enumeration

- Start with the joint probability distribution:

	toothache		\neg toothache	
	catch	\neg catch	catch	\neg catch
cavity	.108	.012	.072	.008
\neg cavity	.016	.064	.144	.576

- For any proposition ϕ , sum the probability where it is true: $P(\phi) = \sum_{\omega: \omega \models \phi} P(\omega)$
- $P(\text{toothache}) = 0.108 + 0.012 + 0.016 + 0.064 = 0.2$
- Unconditional or **marginal probability** of toothache
- Process is called marginalization or summing out

$$\text{Ax}_3: P(A \cup B) = P(A) + P(B)$$

for disjoint events $A, B \subseteq \Omega$

Marginalization and conditioning

- Let \mathbb{Y}, \mathbb{Z} be sequences of random variables s.th.
 $\mathbb{Y} \cup \mathbb{Z}$ denotes all random variables describing the world
- Marginalization
 - $P(\mathbb{Y}) = \sum_{z \in Z} P(\mathbb{Y}, z)$
- Conditioning
 - $P(\mathbb{Y}) = \sum_{z \in Z} P(\mathbb{Y} | z)P(z)$

Inference by enumeration

- Start with the joint probability distribution:

		toothache		\neg toothache	
		catch	\neg catch	catch	\neg catch
cavity		.108	.012	.072	.008
\neg cavity		.016	.064	.144	.576

For any proposition ϕ , sum the atomic events where it is true:

$$P(\phi) = \sum_{\omega: \omega \models \phi} P(\omega)$$

- $P(\text{cavity} \vee \text{toothache}) = 0.108 + 0.012 + 0.072 + 0.008 + 0.016 + 0.064 = 0.28$

$$(P(\text{cavity} \vee \text{toothache}) = P(\text{cavity}) + P(\text{toothache}) - P(\text{cavity} \wedge \text{toothache}))$$

Inference by enumeration

- Start with the joint probability distribution:

		<i>toothache</i>	\neg <i>toothache</i>		
		<i>catch</i>	\neg <i>catch</i>	<i>catch</i>	\neg <i>catch</i>
<i>cavity</i>	.108	.012	.072	.008	
\neg <i>cavity</i>	.016	.064	.144	.576	

- Can also compute conditional probabilities:

$$\begin{aligned} P(\neg \text{cavity} \mid \text{toothache}) &= \frac{P(\neg \text{cavity} \wedge \text{toothache})}{P(\text{toothache})} && \text{Product rule} \\ &= \frac{0.016 + 0.064}{0.108 + 0.012 + 0.016 + 0.064} \\ &= 0.08 / 0.2 = 0.4 \end{aligned}$$

$$P(\text{cavity} \mid \text{toothache}) = (0.108 + 0.012) / 0.2 = 0.6$$

Normalization

	toothache		\neg toothache	
	catch	\neg catch	catch	\neg catch
cavity	.108	.012	.072	.008
\neg cavity	.016	.064	.144	.576

- Denominator $P(z)$ (or $P(\text{toothache})$ in the example before) can be viewed as a **normalization constant α**

$$\begin{aligned} P(\text{Cavity} \mid \text{toothache}) &= \alpha P(\text{Cavity,toothache}) \\ &= \alpha [P(\text{Cavity,toothache,catch}) + P(\text{Cavity,toothache},\neg \text{catch})] \\ &= \alpha [<0.108,0.016> + <0.012,0.064>] \\ &= \alpha <0.12,0.08> = <0.6,0.4> \end{aligned}$$

General idea: compute distribution on query variable by fixing evidence variables (Toothache) and summing over **hidden variables** (Catch)

Inference by enumeration, contd.

Typically, we are interested in

the posterior joint distribution of the **query variables Y**
given specific values **e** for the **evidence variables E**
(X are all variables of the modeled world)

Let the **hidden variables** be $H = X - Y - E$ then the required summation of joint entries is done by summing out the hidden variables:

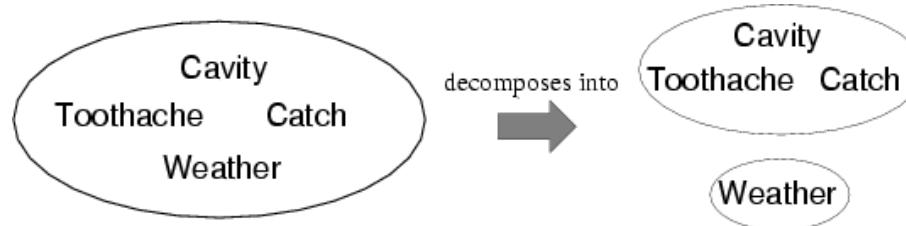
$$P(Y | E = e) = \alpha P(Y, E = e) = \alpha \sum_h P(Y, E = e, H = h)$$

- The terms in the summation are joint entries because Y , E and H together exhaust the set of random variables (X)
- Obvious problems:
 1. Worst-case time complexity $O(d^n)$ where d is the largest arity and n denotes the number of random variables
 2. Space complexity $O(d^n)$ to store the joint distribution
 3. How to find the numbers for $O(d^n)$ entries?

Independence

- A and B are independent iff

$$P(A|B) = P(A) \quad \text{or} \quad P(B|A) = P(B) \quad \text{or} \quad P(A, B) = P(A) P(B)$$



$$\begin{aligned} P(\text{Toothache, Catch, Cavity, Weather}) \\ = P(\text{Toothache, Catch, Cavity}) P(\text{Weather}) \end{aligned}$$

- 32 entries reduced to 12;
- *Absolute* independence powerful but rare
- Dentistry is a large field with hundreds of variables, none of which are independent. What to do?

Conditional independence

- $P(\text{Toothache}, \text{Cavity}, \text{Catch})$ has $2^3 - 1 = 7$ independent entries
- If I have a cavity, the probability that the probe catches it doesn't depend on whether I have a toothache:
(1) $P(\text{catch} \mid \text{toothache}, \text{cavity}) = P(\text{catch} \mid \text{cavity})$
- The same independence holds if I haven't got a cavity:
(2) $P(\text{catch} \mid \text{toothache}, \neg\text{cavity}) = P(\text{catch} \mid \neg\text{cavity})$
- Catch is **conditionally independent** of Toothache given Cavity:
 $P(\text{Catch} \mid \text{Toothache}, \text{Cavity}) = P(\text{Catch} \mid \text{Cavity})$
- Equivalent statements:
 $P(\text{Toothache} \mid \text{Catch}, \text{Cavity}) = P(\text{Toothache} \mid \text{Cavity})$
 $P(\text{Toothache}, \text{Catch} \mid \text{Cavity}) = P(\text{Toothache} \mid \text{Cavity}) P(\text{Catch} \mid \text{Cavity})$

Conditional independence contd.

- Write out full joint distribution using chain rule:

$$P(\text{Toothache}, \text{Catch}, \text{Cavity})$$

$$= P(\text{Toothache} | \text{Catch}, \text{Cavity}) P(\text{Catch}, \text{Cavity})$$

$$= P(\text{Toothache} | \text{Catch}, \text{Cavity}) P(\text{Catch} | \text{Cavity}) P(\text{Cavity})$$

conditional independence

$$= P(\text{Toothache} | \text{Cavity}) P(\text{Catch} | \text{Cavity}) P(\text{Cavity})$$

i.e., $2 + 2 + 1 = 5$ independent numbers

- In most cases, the use of conditional independence reduces the size of the representation of the joint distribution from exponential in n to linear in n
- Conditional independence is our most basic and robust form of knowledge about uncertain environments

Naïve Bayes Model

$$P(Cavity|toothache \wedge catch)$$

$$= \alpha P(toothache \wedge catch|Cavity)P(Cavity)$$

$$= \alpha P(toothache|Cavity)P(catch|Cavity)P(Cavity)$$

Usually, the assumption that effects are independent is wrong,
but works well in practice

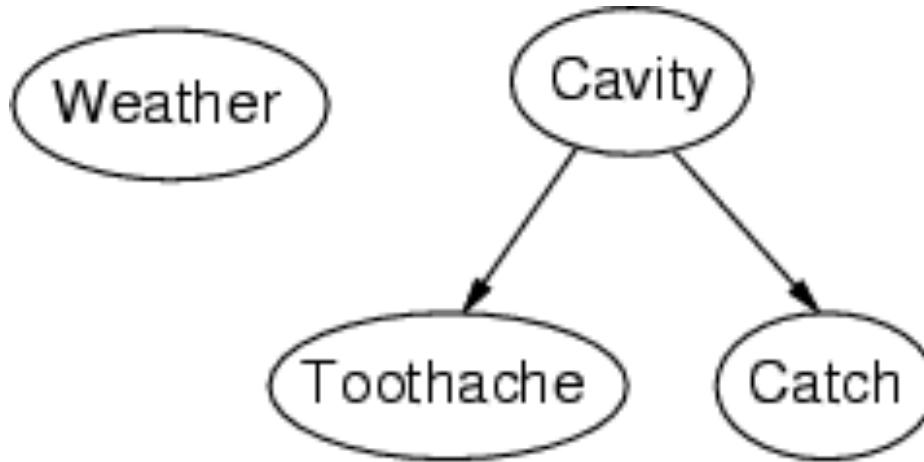


Bayesian networks

- A simple, graphical notation for conditional independence assertions and hence for compact specification of full joint distributions
- Syntax:
 - a set of nodes, one per variable
 - a directed, acyclic graph (link \approx "directly influences")
 - a conditional distribution for each node given its parents:
$$P(X_i | \text{Parents}(X_i))$$
- In the simplest case, conditional distribution represented as a **conditional probability table** (CPT) giving the distribution over X_i for each combination of parent values

Example

- Topology of network encodes conditional independence assertions:

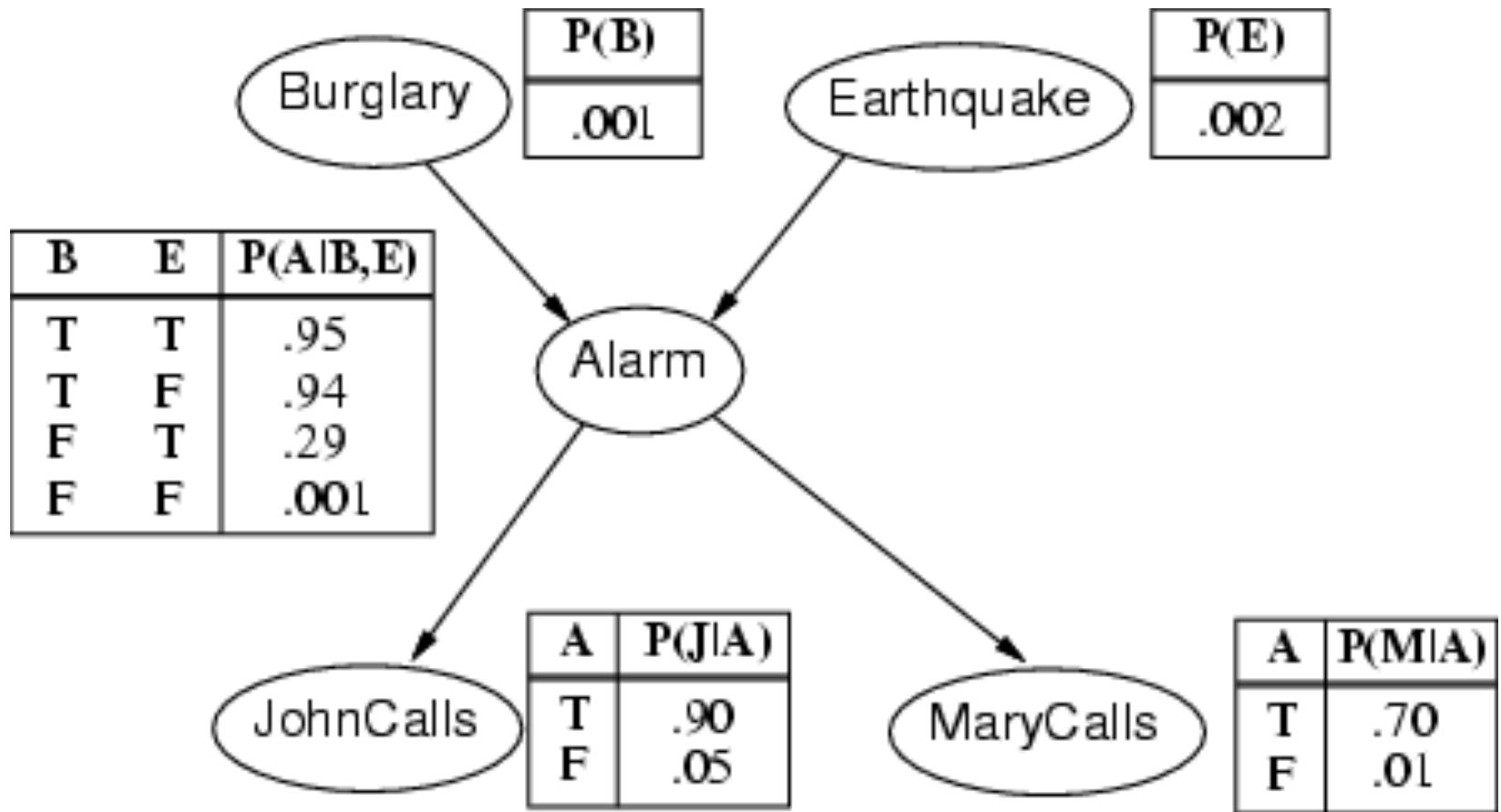


- Weather* is independent of the other variables
- Toothache* and *Catch* are conditionally independent given *Cavity*

Example

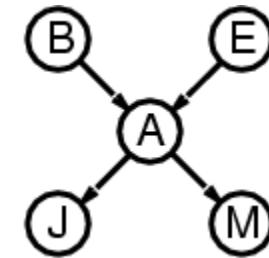
- I'm at work, neighbor John calls to say my alarm is ringing, but neighbor Mary doesn't call. Sometimes it's set off by minor earthquakes. Is there a burglary?
- Variables: *Burglary*, *Earthquake*, *Alarm*, *JohnCalls*, *MaryCalls*
- Network topology reflects "causal" knowledge:
 - A burglar can set the alarm off
 - An earthquake can set the alarm off
 - The alarm can cause Mary to call
 - The alarm can cause John to call

Example contd.



Compactness of Full Joint Encoding Due to Network Sparseness

- A CPT for Boolean X_i with k Boolean parents has 2^k rows for the combinations of parent values
- Each row requires one number p for $X_i = \text{true}$ (the number for $X_i = \text{false}$ is just $1-p$)
- If each variable has no more than k parents, the complete network requires $n \cdot 2^k$ numbers
- i.e., grows linearly with n , vs. 2^n for the full joint distribution
- For burglary net, $1 + 1 + 4 + 2 + 2 = 10$ numbers (vs. $2^5 - 1 = 31$)



Semantics

The full joint distribution is defined as the product of the local conditional distributions:

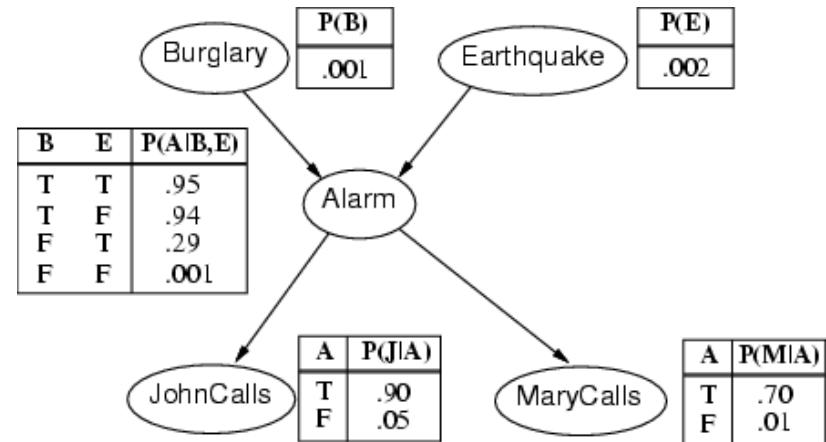
$$P(X_1, \dots, X_n) = \prod_{i=1}^n P(X_i | \text{Parents}(X_i))$$

e.g., $P(j \wedge m \wedge a \wedge \neg b \wedge \neg e)$

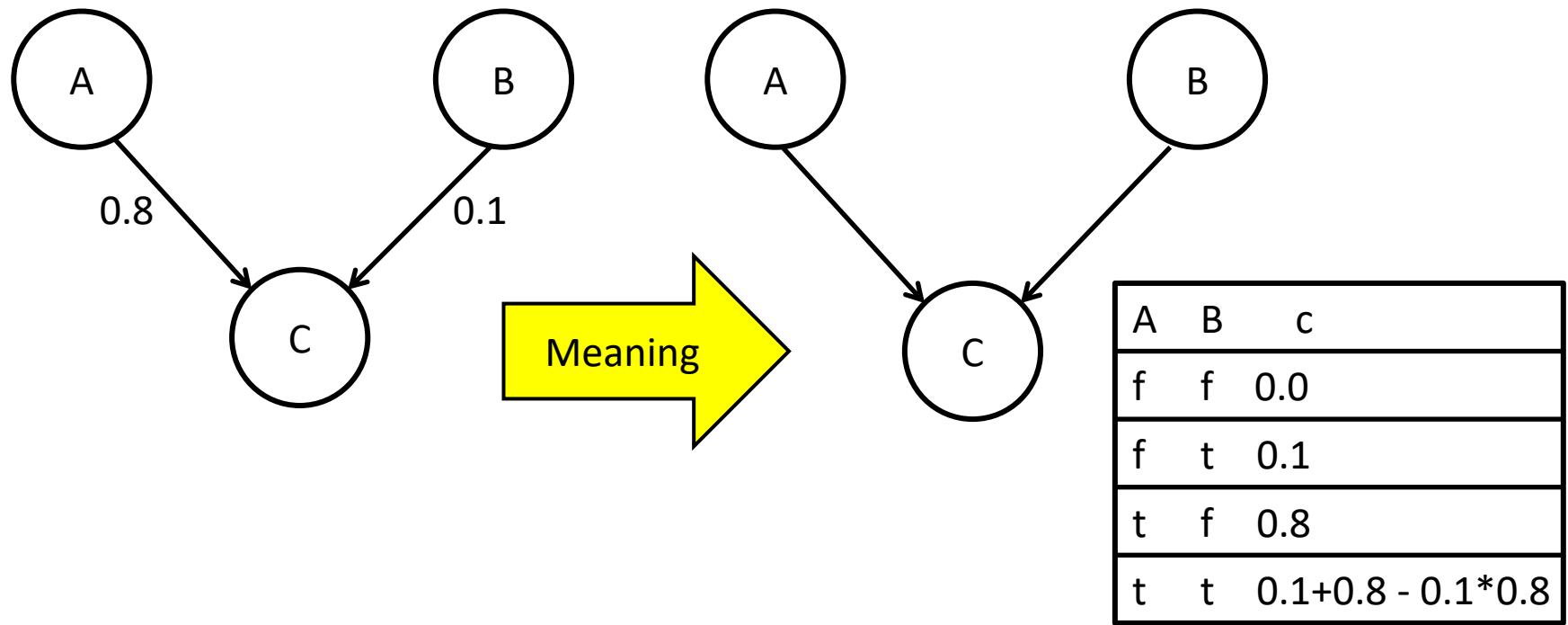
$$= P(j | a) P(m | a) P(a | \neg b, \neg e) P(\neg b) P(\neg e)$$

$$= 0.90 \times 0.7 \times 0.001 \times 0.999 \times 0.998$$

$$\approx 0.00063$$



Noisy-OR-Representation



- Substantial reduction of the probabilities to be stored

Inference tasks

Simple queries: compute posterior marginal $\mathbf{P}(X_i|\mathbf{E}=\mathbf{e})$

e.g., $P(\text{NoGas}|\text{Gauge}=\text{empty}, \text{Lights}=\text{on}, \text{Starts}=\text{false})$

Conjunctive queries: $\mathbf{P}(X_i, X_j|\mathbf{E}=\mathbf{e}) = \mathbf{P}(X_i|\mathbf{E}=\mathbf{e})\mathbf{P}(X_j|X_i, \mathbf{E}=\mathbf{e})$

Optimal decisions: decision networks include utility information;
probabilistic inference required for $P(\text{outcome}|\text{action}, \text{evidence})$

Value of information: which evidence to seek next?

Sensitivity analysis: which probability values are most critical?

Explanation: what must be true to allow for a certain (desired) conclusion?

Inference by enumeration

Slightly intelligent way to sum out variables from the joint without actually constructing its explicit representation

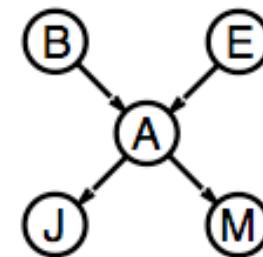
Simple query on the burglary network:

$$\mathbf{P}(B|j, m)$$

$$= \mathbf{P}(B, j, m) / P(j, m)$$

$$= \alpha \mathbf{P}(B, j, m)$$

$$= \alpha \sum_e \sum_a \mathbf{P}(B, e, a, j, m)$$



Rewrite full joint entries using product of CPT entries:

$$\mathbf{P}(B|j, m)$$

$$= \alpha \sum_e \sum_a \mathbf{P}(B) P(e) \mathbf{P}(a|B, e) P(j|a) P(m|a)$$

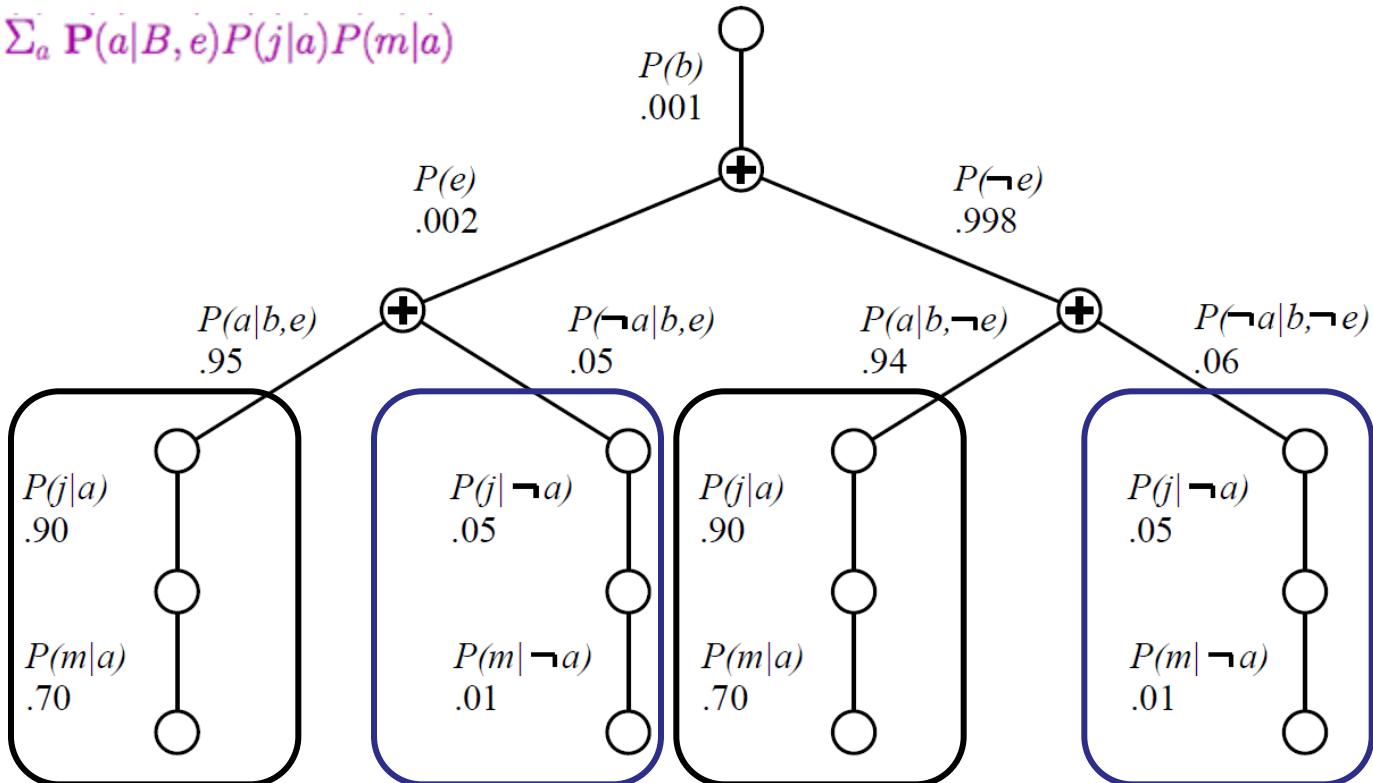
$$= \alpha \mathbf{P}(B) \sum_e P(e) \sum_a \mathbf{P}(a|B, e) P(j|a) P(m|a)$$

Recursive depth-first enumeration: $O(n)$ space, $O(d^n)$ time

(Calculation over tree sum-product evaluation tree
with path length n (=number of RVs) and degree d
(=maximal number domain elements))

Evaluation Tree

$$\mathbf{P}(B) \sum_e P(e) \sum_a \mathbf{P}(a|B, e) P(j|a) P(m|a)$$



Enumeration is inefficient: repeated computation

e.g., computes $P(j|a)P(m|a)$ for each value of e

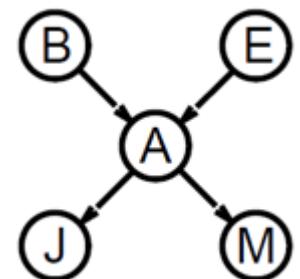
Can do better with **Variable Elimination (VE)**

Basic Objects of VE

$$\mathbf{P}(B) \sum_e P(e) \sum_a \mathbf{P}(a|B, e)P(j|a)P(m|a)$$

- Track objects called **factors**
(right-to-left, in tree: bottom up)
- Initial factors are local CPTs

$$\underbrace{P(B)}_{f_B(B)} \quad \underbrace{P(J|A)}_{f_J(A, J)} \quad \underbrace{P(A|B, E)}_{f_A(A, B, E)} \quad \underbrace{P(M|A)}_{f_M(A)}$$



- During elimination create new factors
- Form of **Dynamic Programming**

Basic Operations: Pointwise Product

- Pointwise product of factors f_1 and f_2
 - for example: $f_1(A,B) * f_2(B,C) = f(A,B,C)$
 - in general:
$$f_1(X_1, \dots, X_j, Y_1, \dots, Y_k) * f_2(Y_1, \dots, Y_k, Z_1, \dots, Z_l) =$$
$$f(X_1, \dots, X_j, Y_1, \dots, Y_k, Z_1, \dots, Z_l)$$
 - has 2^{j+k+l} entries (if all variables are binary)



Join by pointwise product

A	$f_{JM}(A)$
T	.9 * .7
F	.05 * .01

A	$f_J(A)$
T	.9
F	.05

A	$f_M(A)$
T	.7
F	.01

$$\mathbf{P}(B) \sum_e P(e) \sum_a \mathbf{P}(a|B, e) P(j|a) P(m|a)$$

A	B	E	$f_{AJM}(A, B, E)$
T	T	T	.95 * .63
T	T	F	.94 * .63
T	F	T	.29 * .63
T	F	F	.001 * .63
F	T	T	.05 * .0005
F	T	F	.06 * .0005
F	F	T	.71 * .0005
F	F	F	.999 * .0005

A	B	E	$f_{AJM}(A, B, E)$
T	T	T	.95
T	T	F	.94
T	F	T	.29
T	F	F	.001
F	T	T	.05
F	T	F	.06
F	F	T	.71
F	F	F	.999

A	$f_{JM}(A)$
T	.63
F	.0005



Basic Operations: Summing out

- Summing out a variable from a product of factors
 - Move any constant factors outside the summation
 - Add up submatrices in pointwise product of remaining factors

$$\begin{aligned}\sum_X f_1^* \dots * f_k &= f_1^* \dots * f_i^* \sum_X f_{i+1}^* \dots * f_k \\ &= f_1^* \dots * f_i^* f_X\end{aligned}$$

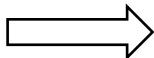
assuming f_1, \dots, f_i do not depend on X

Summing out

$$\mathbf{P}(B) \sum_e P(e) \sum_a \mathbf{P}(a|B, e)P(j|a)P(m|a)$$

A	B	E	$f_{AJM}(A, B, E)$
T	T	T	.95 * .63
T	T	F	.94 * .63
T	F	T	.29 * .63
T	F	F	.001 * .63
F	T	T	.05 * .0005
F	T	F	.06 * .0005
F	F	T	.71 * .0005
F	F	F	.999 * .0005

Summing out a



B	E	$f_{\bar{A}JM}(B, E)$
T	T	.95 * .63 + .05 * .0005 = .5985
T	F	.94 * .63 + .06 * .0005 = .5922
F	T	.29 * .63 + .71 * .0005 = .1830
F	F	.001 * .63 + .999 * .0005 = .001129

VE result for Burglary Example

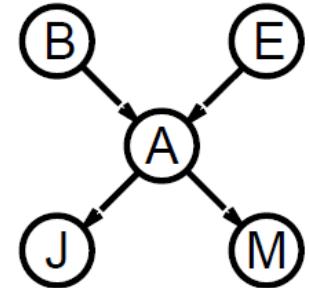
$$\begin{aligned}\mathbf{P}(B|j, m) &= \alpha \underbrace{\mathbf{P}(B)}_B \underbrace{\sum_e P(e)}_E \underbrace{\sum_a \mathbf{P}(a|B, e)}_A \underbrace{P(j|a)}_J \underbrace{P(m|a)}_M \\ &= \alpha \mathbf{P}(B) \sum_e P(e) \sum_a \mathbf{P}(a|B, e) P(j|a) f_M(a) \\ &= \alpha \mathbf{P}(B) \sum_e P(e) \sum_a \mathbf{P}(a|B, e) f_J(a) f_M(a) \\ &= \alpha \mathbf{P}(B) \sum_e P(e) \sum_a f_A(a, b, e) f_J(a) f_M(a) \\ &= \alpha \mathbf{P}(B) \sum_e P(e) f_{\bar{A}JM}(b, e) \text{ (sum out } A\text{)} \\ &= \alpha \mathbf{P}(B) f_{\bar{E}\bar{A}JM}(b) \text{ (sum out } E\text{)} \\ &= \alpha f_B(b) \times f_{\bar{E}\bar{A}JM}(b)\end{aligned}$$

Further Optimization by Finding Irrelevant Variables

Consider the query $P(JohnCalls|Burglary=true)$

$$P(J|b) = \alpha P(b) \sum_e P(e) \sum_a P(a|b, e) P(J|a) \sum_m P(m|a)$$

Sum over m is identically 1; M is **irrelevant** to the query



Thm 1: Y is irrelevant unless $Y \in Ancestors(\{X\} \cup \mathbf{E})$

Here, $X = JohnCalls$, $\mathbf{E} = \{Burglary\}$, and
 $Ancestors(\{X\} \cup \mathbf{E}) = \{\text{Alarm, Earthquake, Burglary}\}$
so $MaryCalls$ is irrelevant

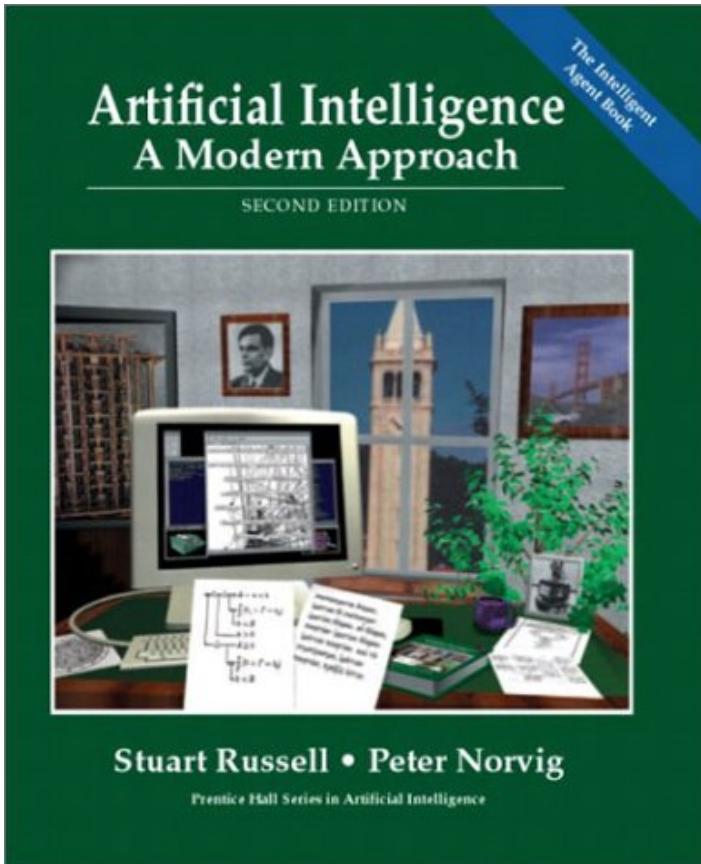
Irrelevant variables can be identified by a graph theoretical criterion on the associated moral graph by m-separation

Many other inference algorithms and optimizations known

→ Discussed in detail in course on Bayesian Knowledge Representation

Acknowledgements

- Slides from AIMA book provided by Cristina Conati, UBC



Data Mining Bayesian Networks

- Full Bayesian Learning
- MAP learning
- Maximum Likelihood Learning
- Learning Bayesian Networks
 - Fully observable
 - With hidden (unobservable) variables

Full Bayesian Learning

- In the learning methods we have seen so far, the idea was always to find the best model that could explain some observations
- In contrast, **full Bayesian learning** sees learning as Bayesian **updating of a probability distribution** over the hypothesis space, given data
 - H is the hypothesis variable
 - Possible hypotheses (values of H) h_1, \dots, h_n
 - $P(H)$ = prior probability distribution over hypothesis space
- j_{th} observation d_j gives the outcome of random variable D_j
 - training data $\mathbf{d} = d_1, \dots, d_k$

Non-Standard-Datenbanken

Bayesian Networks, Inference, and Learning

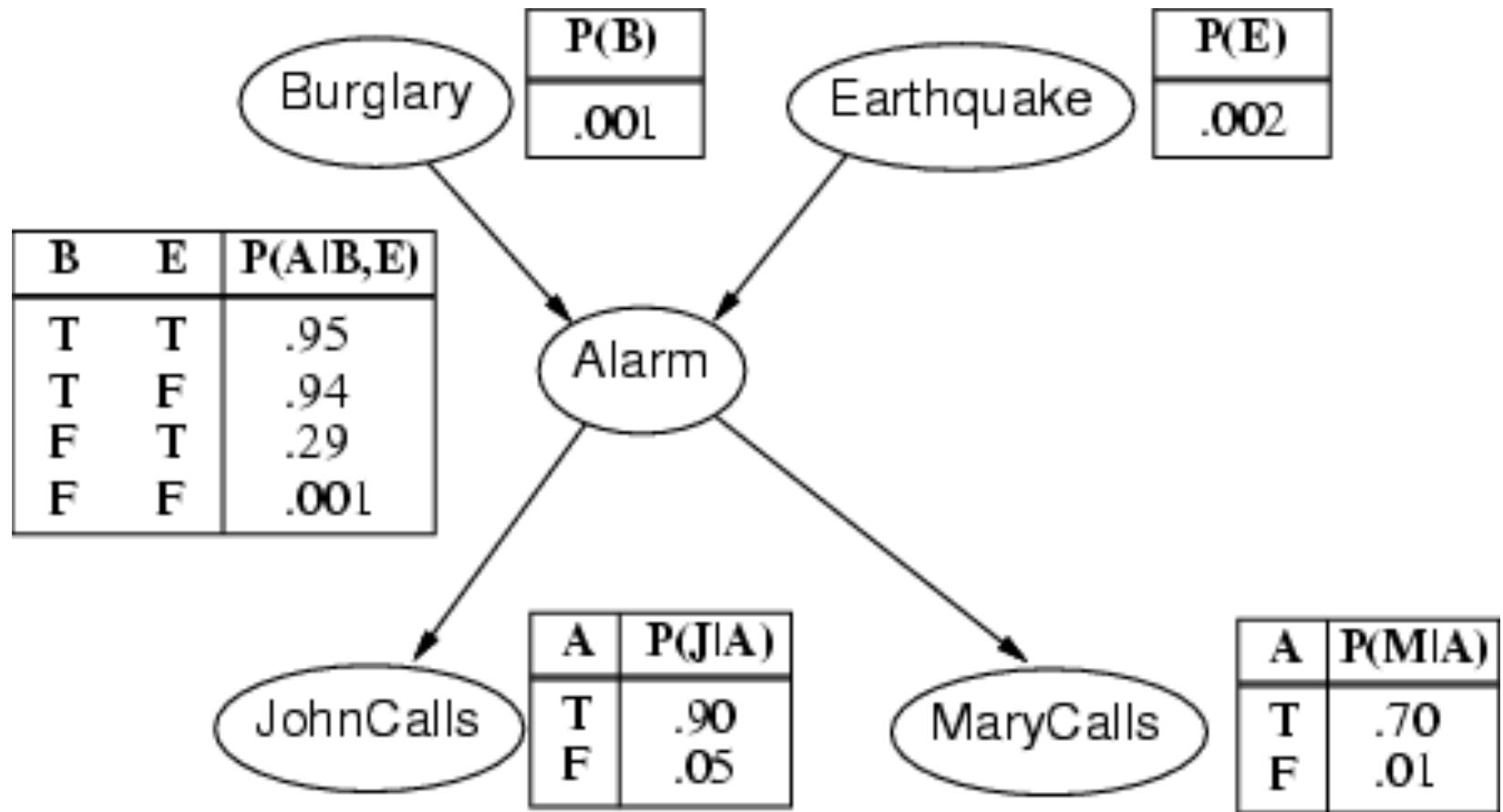
Prof. Dr. Ralf Möller

Universität zu Lübeck

Institut für Informationssysteme

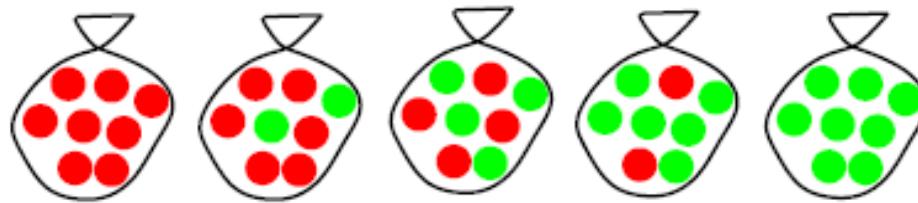


Example contd.



Example

- Suppose we have 5 types of candy bags
 - 10% are 100% cherry candies (h_{100})
 - 20% are 75% cherry + 25% lime candies (h_{75})
 - 40% are 50% cherry + 50% lime candies (h_{50})
 - 20% are 25% cherry + 75% lime candies (h_{25})
 - 10% are 100% lime candies (h_0)



- Then we observe candies drawn from some bag 
- Let's call θ the parameter that defines the fraction of cherry candy in a bag, and h_θ the corresponding hypothesis
- Which of the five kinds of bag has generated my 10 observations? $P(h_\theta | d)$.
- What flavour will the next candy be? Prediction $P(X | d)$

Full Bayesian Learning

- Given the data so far, each hypothesis h_i has a posterior probability:
 - $P(h_i | d) = \alpha P(d | h_i) P(h_i)$ (**Bayes theorem**)
 - where $P(d | h_i)$ is called the likelihood of the data under each hypothesis
- Predictions over a new entity X are a weighted average over the prediction of each hypothesis:
 - $$\begin{aligned} P(X | d) &= \sum_i P(X, h_i | d) \\ &= \sum_i P(X | h_i, d) P(h_i | d) \\ &= \sum_i P(X | h_i) P(h_i | d) \\ &\sim \sum_i P(X | h_i) P(d | h_i) P(h_i) \end{aligned}$$
 - The data does not add anything to a prediction given a hypothesis
 - The weights are given by the data likelihood and prior of each h

- + No need to pick one best-guess hypothesis!
- Need to iterate over all hypotheses

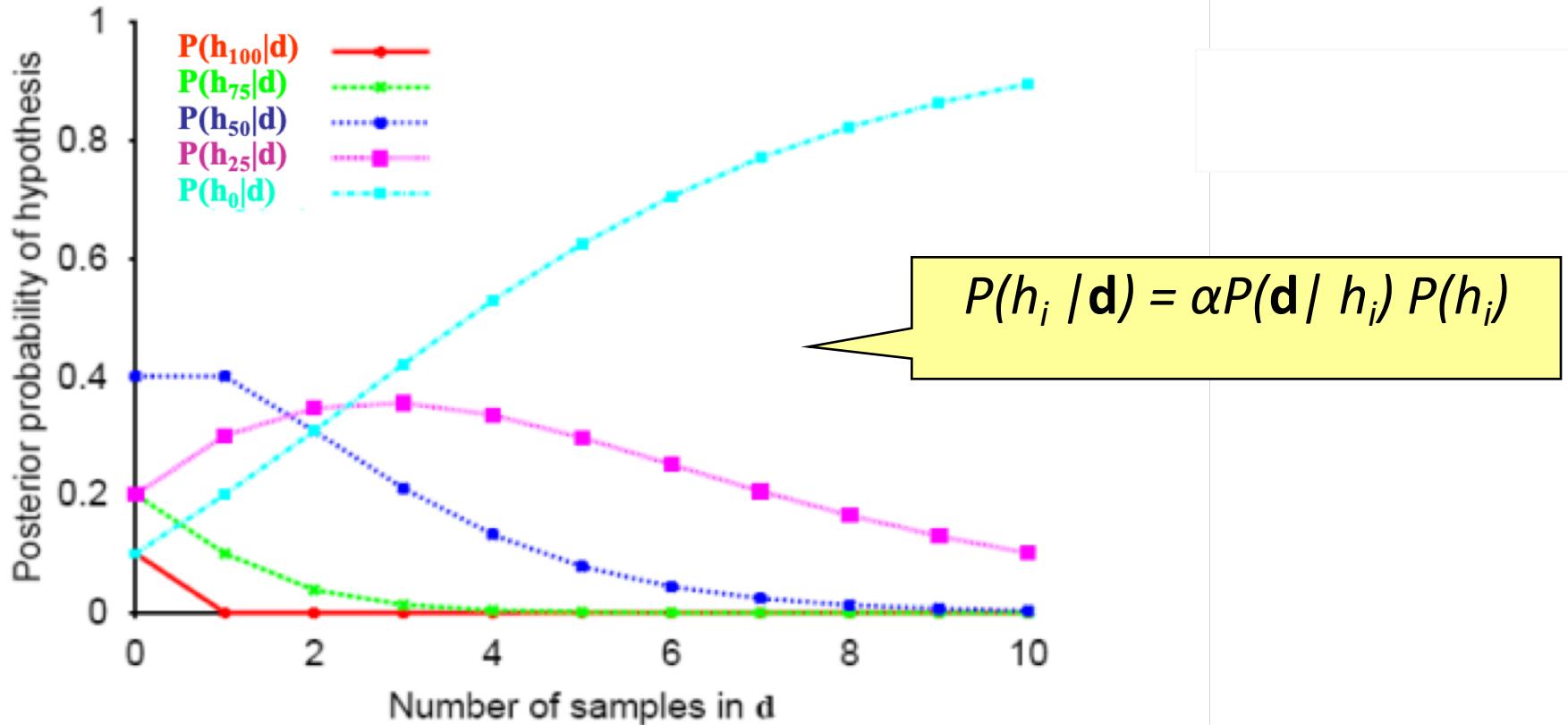
Example

- If we re-wrap each candy and return it to the bag, our 10 observations are independent and identically distributed, i.i.d, so
 - $P(\mathbf{d} | h_\theta) = \prod_j P(d_j | h_\theta)$ for $j=1,\dots,10$
- For a given h_θ , the value of $P(d_j | h_\theta)$ is
 - $P(d_j = \text{cherry} | h_\theta) = \theta; P(d_j = \text{lime} | h_\theta) = (1-\theta)$
- And given N observations, of which c are cherry and $\ell = N-c$ lime

$$P(\mathbf{d} | h_\theta) = \prod_{j=1}^c \theta \prod_{j=1}^\ell (1-\theta) = \theta^c (1-\theta)^\ell$$

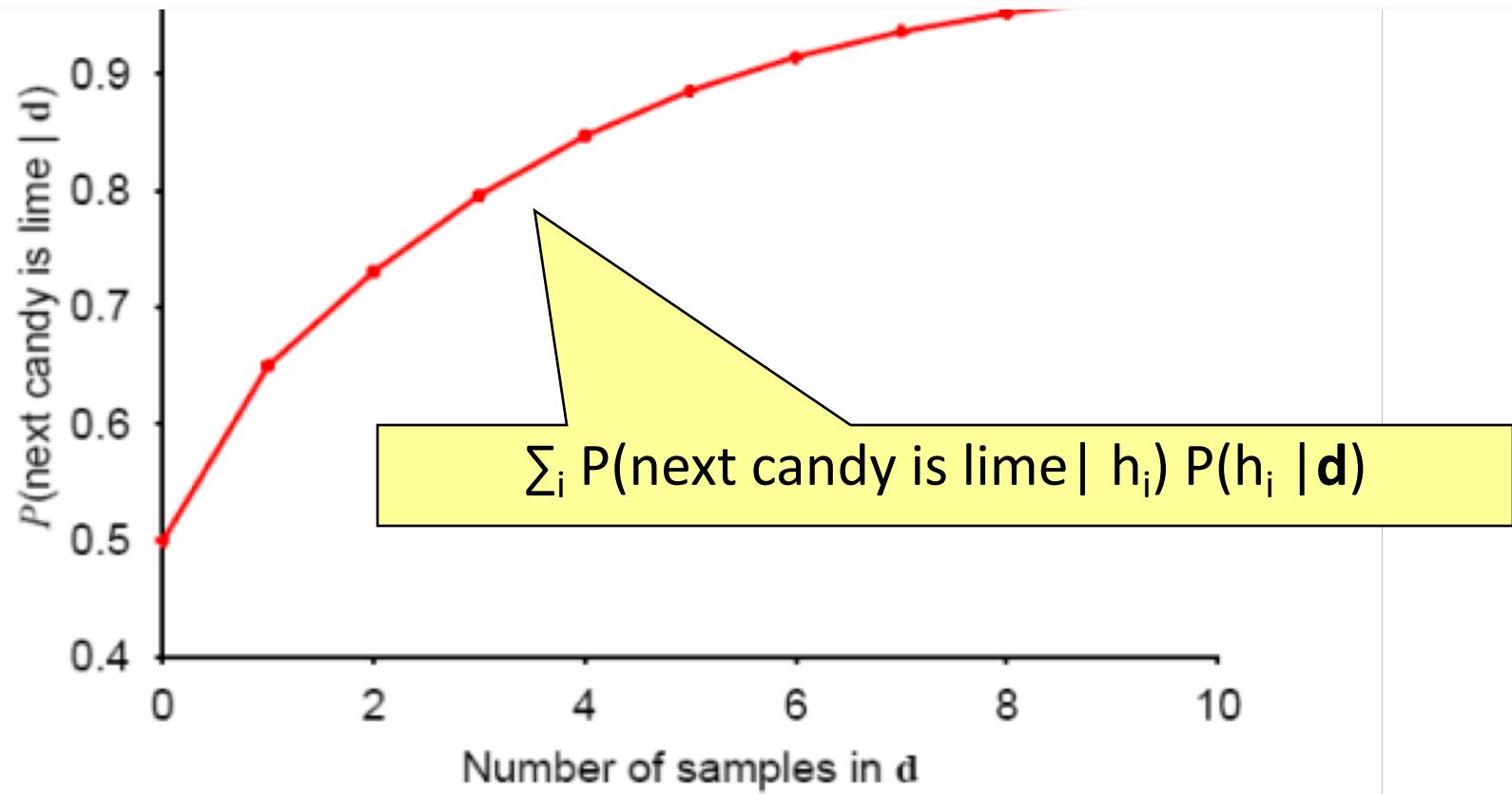
- **Binomial distribution:** probability of # of successes in a sequence of N independent trials with binary outcome, each of which yields success with probability θ .
- For instance, after observing 3 lime candies in a row:
 - $P([\text{lime}, \text{lime}, \text{lime}] | h_{50}) = 0.5^3$ because the probability of seeing lime for each observation is 0.5 under this hypotheses

All-limes: Posterior Probability of H



- Initially, the h_p with higher priors dominate (h_{50} with prior = 0.4)
- As data comes in, the finally best hypothesis (h_0) starts dominating, as the probability of seeing this data given the other hypotheses gets increasingly smaller
 - After seeing three lime candies in a row, the probability that the bag is the all-lime one starts taking off

Prediction Probability



- The probability that the next candy is lime increases with the probability that the bag is an all-lime one

Overview

- Full Bayesian Learning
- MAP learning
- Maximum Likelihood Learning
- Learning Bayesian Networks
 - Fully observable
 - With hidden (unobservable) variables



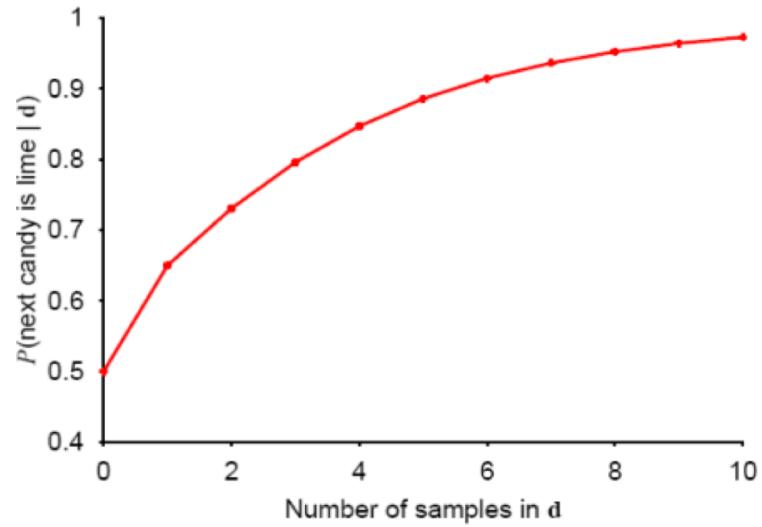
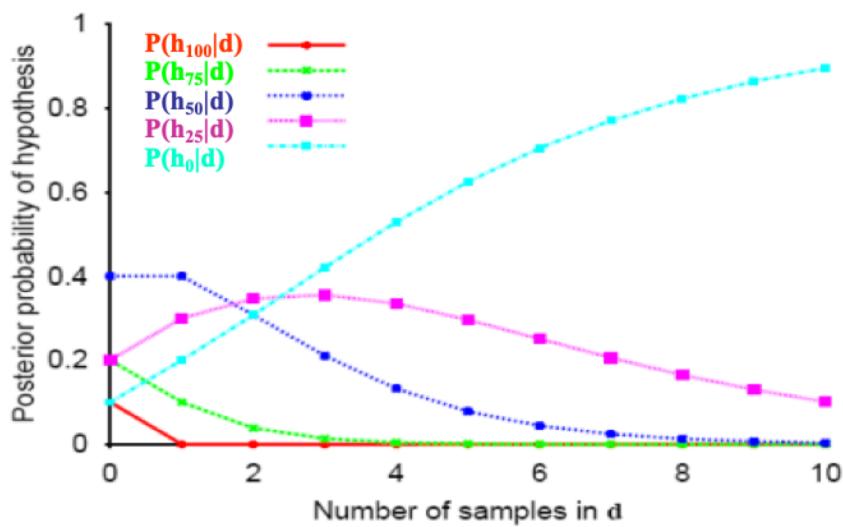
MAP approximation

- Full Bayesian learning seems like a very safe bet, but unfortunately it does not work well in practice
 - Summing over the hypothesis space is often intractable (e.g., 18,446,744,073,709,551,616 Boolean functions of 6 attributes)
- Very common approximation: Maximum a posterior (MAP) learning:
 - Instead of doing prediction by considering all possible hypotheses , as in
 - $P(X|d) = \sum_i P(X| h_i) P(h_i | d)$
 - Make predictions based on h_{MAP} that maximises $P(h_i | d)$
 - I.e., maximize $P(d | h_i) P(h_i)$
 - $P(X|d) \approx P(X| h_{MAP})$

MAP approximation

➤ MAP is a good approximation when $P(X | d) \approx P(X | h_{MAP})$

- In our example, h_{MAP} is the all-lime bag after only 3 candies, predicting that the next candy will be lime with $p = 1$
- The Bayesian learner gave a prediction of 0.8, safer after seeing only 3 candies



Bias

- As more data arrive, MAP and Bayesian prediction become closer, as MAP's competing hypotheses become less likely
- Often easier to find MAP (optimization problem) than deal with a large summation problem
- $P(H)$ plays an important role in both MAP and Full Bayesian Learning (defines learning bias)
- Used to define a tradeoff between model complexity and its ability to fit the data
 - More complex models can explain the data better => higher $P(\mathbf{d} | h_i)$
danger of overfitting
 - But they are less likely a priory because there are more of them than simpler model => lower $P(h_i)$
 - I.e. common learning bias is to penalize complexity

Overview

- Full Bayesian Learning
- MAP learning
- Maximum Likelihood Learning
- Learning Bayesian Networks
 - Fully observable
 - With hidden (unobservable) variables

Maximum Likelihood (ML)Learning

- Further simplification over full Bayesian and MAP learning
 - Assume uniform priors over the space of hypotheses
 - MAP learning (maximize $P(\mathbf{d} | h_i) P(h_i)$) reduces to maximize $P(\mathbf{d} | h_i)$
- When is ML appropriate?

Maximum Likelihood (ML) Learning

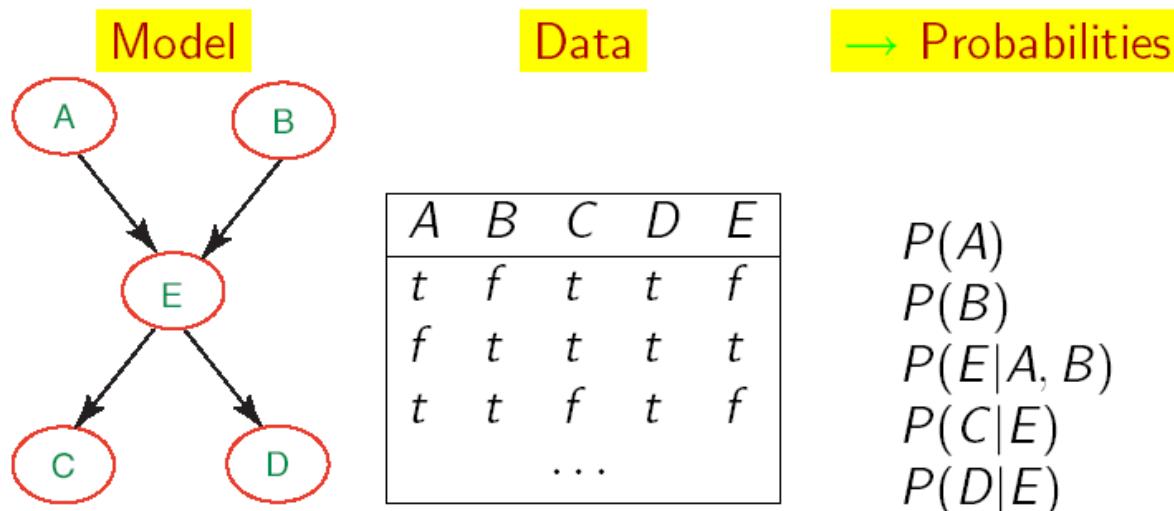
- Further simplification over Full Bayesian and MAP learning
 - Assume uniform prior over the space of hypotheses
 - MAP learning (maximize $P(\mathbf{d} | h_i) P(h_i)$) reduces to maximize $P(\mathbf{d} | h_i)$
- When is ML appropriate?
 - Used in statistics as the standard (non-bayesian) statistical learning method by those who distrust subjective nature of hypotheses priors
 - When the competing hypotheses are indeed equally likely (e.g. have same complexity)
 - With very large datasets, for which $P(\mathbf{d} | h_i)$ tends to overcome the influence of $P(h_i)$

Overview

- Full Bayesian Learning
- MAP learning
- Maximum Likelihood Learning
- Learning Bayesian Networks
 - Fully observable (complete data)
 - With hidden (unobservable) variables

Learning BNets: Complete Data

- We will start by applying ML to the simplest type of BNets learning:
 - known structure
 - Data containing observations for all variables
 - ✓ All variables are observable, no missing data
- The only thing that we need to learn are the network's parameters



Maximum-Likelihood-Parameterschätzung

- Nehme an, die Struktur eines BNs sei bekannt
- Ziel: Schätze BN-Parameter θ
 - Einträge in CPTs, $P(X | \text{Parents}(X))$
- Eine Parametrierung θ ist gut, falls hierdurch die beobachteten Daten wahrscheinlich generiert werden:

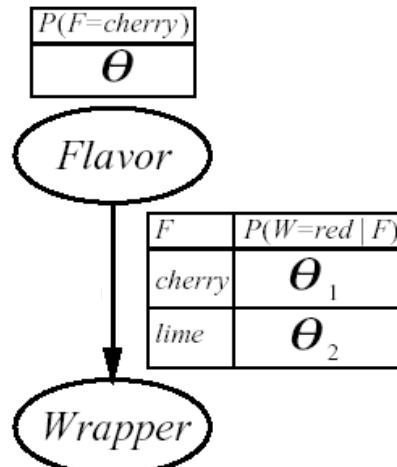
$$P(D | \theta) = \prod_m P(x[m] | \theta)$$

- Maximum Likelihood Estimation (MLE) Prinzip: Wähle θ^* so, dass $P(D | \theta^*)$ maximiert wird

Gleichverteilte,
unabhängige
Stichprobem
(i.i.d. samples)

Anwendungsbeispiel Bonbonfabrik

- Ein Hersteller wählt die Farbe des Bonbonpapiers mit einer bestimmten Wahrscheinlichkeit je nach Geschmack, wobei die entsprechende Verteilung nicht bekannt sei
 - Wenn Geschmack=cherry, wähle rotes Papier mit W'keit θ_1
 - Wenn Geschmack=lime, wähle rotes Papier mit W'keit θ_2
- Das Bayessche Netzwerk enthält drei zu lernende Parameter
 - $\theta \theta_1 \theta_2$



Anwendungsbeispiel Bonbonfabrik

➤ $P(W=\text{green}, F = \text{cherry} | h_{\theta\theta_1\theta_2}) = (*)$

$$= P(W=\text{green} | F = \text{cherry}, h_{\theta\theta_1\theta_2}) P(F = \text{cherry} | h_{\theta\theta_1\theta_2})$$

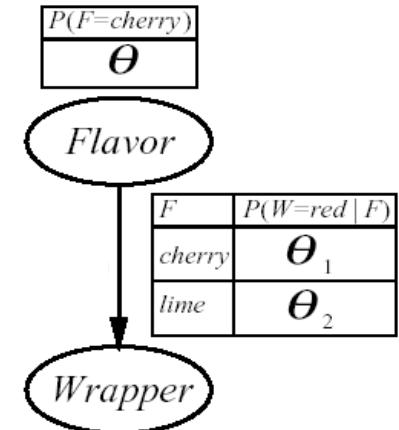
$$= \theta (1-\theta_1)$$

➤ Wir packen N Bonbons aus

- c sind cherry und ℓ sind lime
- r^c cherry mit rotem Papier, g^c cherry mit grünem Papier
- r^ℓ lime mit rotem Papier, g^ℓ lime mit grünem Papier
- Jeder Versuch liefert eine Kombination aus Papier und Geschmack wie bei (*)

➤ $P(\mathbf{d} | h_{\theta\theta_1\theta_2})$

$$= \prod_j P(d_j | h_{\theta\theta_1\theta_2}) = \theta^c (1-\theta)^\ell (\theta_1)^{r^c} (1-\theta_1)^{g^c} (\theta_2)^{r^\ell} (1-\theta_2)^{g^\ell}$$



Anwendungsbeispiel Bonbonfabrik

➤ Maximierung des Logarithmus der Zielfunktion

- $c \log \theta + \ell \log(1 - \theta) + r^c \log \theta_1 + g^c \log(1 - \theta_1) + r^\ell \log \theta_2 + g^\ell \log(1 - \theta_2)$

➤ Bestimmung der Ableitungen bzgl. $\theta, \theta_1, \theta_2$

- Ausdrücke ohne Term, nach dem abgeleitet wird, verschwinden

$$\frac{\partial L}{\partial \theta} = \frac{c}{\theta} - \frac{\ell}{1 - \theta} = 0 \quad \Rightarrow \quad \theta = \frac{c}{c + \ell}$$

$$\frac{\partial L}{\partial \theta_1} = \frac{r_c}{\theta_1} - \frac{g_c}{1 - \theta_1} = 0 \quad \Rightarrow \quad \theta_1 = \frac{r_c}{r_c + g_c}$$

$$\frac{\partial L}{\partial \theta_2} = \frac{r_\ell}{\theta_2} - \frac{g_\ell}{1 - \theta_2} = 0 \quad \Rightarrow \quad \theta_2 = \frac{r_\ell}{r_\ell + g_\ell}$$

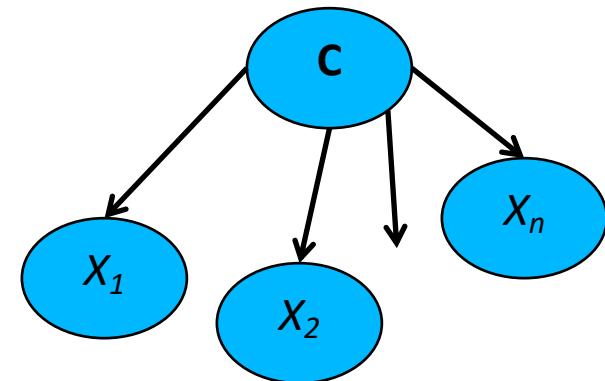


Maximum-Likelihood-Parameterschätzung

- Schätzung durch Bildung relativer Häufigkeiten
- Dieser Prozess ist auf jedes voll beobachtbare BN anwendbar
- Mit vollständigen Daten und Maximum-Likelihood-Parameterschätzung:
 - Parameterlernen zerfällt in separate Lernprobleme für jeden Parameter (CPT) durch Logarithmierung
 - Jeder Parameter wird durch die relative Häufigkeit eines Knotenwertes bei gegebenen Werten der Elternknoten bestimmt

Beliebte Anwendung: Naives Bayes-Modell

- Naïve Bayes-Modell: Sehr einfaches Bayessches Netzwerk zur Klassifikation
 - *Klassenvariable C* (vorherzusagen) bildet Wurzel
 - Attributvariablen X_i (Beobachtungen) sind Blätter



- Naiv, weil angenommen wird, dass die Attributwerte bedingt unabhängig sind, wenn die Klasse gegeben ist

$$P(C|x_1, x_2, \dots, x_n) = \frac{P(C, x_1, x_2, \dots, x_n)}{P(x_1, x_2, \dots, x_n)} = \alpha P(C) \prod_i P(x_i | C)$$

- Deterministische Vorhersagen können durch Wahl der wahrscheinlichsten Klasse erreicht werden
- Skalierung auf realen Daten sehr gut:
 - $2n + 1$ Parameter benötigt

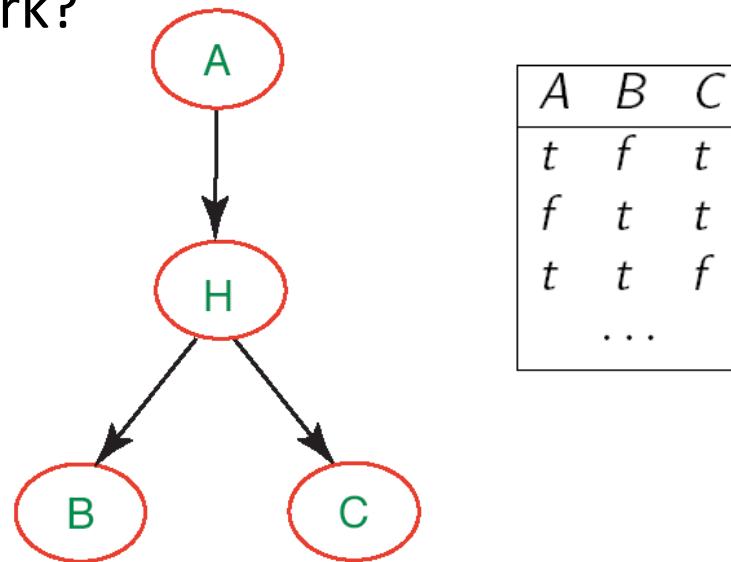
Überblick

- Volles Bayessches Lernen (BMA)
- MAP-Lernen
- Maximum-Likelihood-Lernen
 - Vollständig beobachtbar (vollständige Daten)
 - Mit versteckten (unbeobachtbaren) Variablen



Parameterlernen mit versteckten Variablen

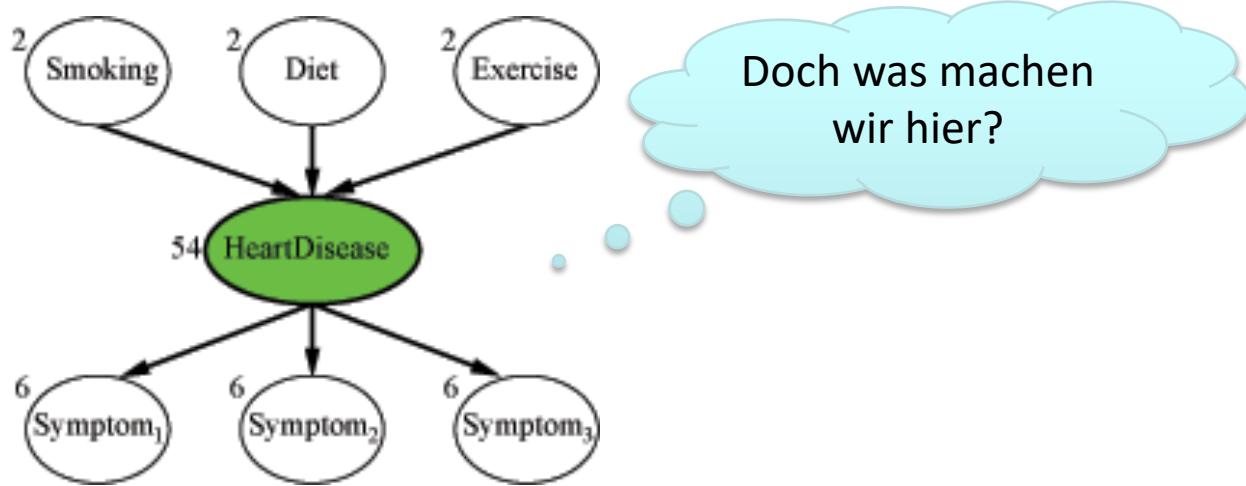
- Bisher haben wir angenommen, Daten für jede Variable des BNs stehen zur Verfügung
- Was machen wir, wenn das nicht der Fall ist, d.h. es gibt versteckte (hidden) Variablen im Netzwerk?



- Den Ansatz mit relativen Häufigkeiten können wir nicht so einfach übernehmen (keine Zähler für Variable H bekannt)

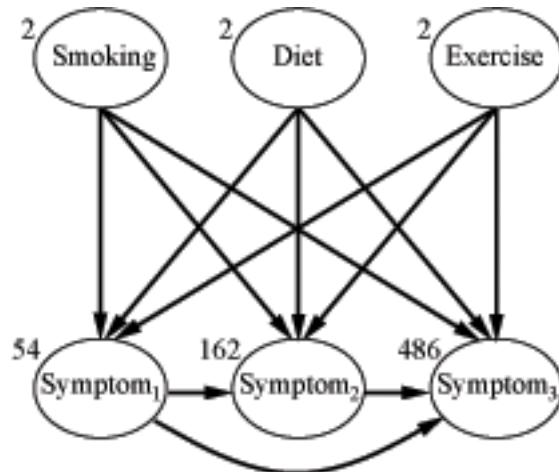
Vermeintliche Lösung

- Vermeide versteckte Variablen
- Könnte klappen bei kleinen Netzwerken



- Jede Variable habe 3 Werte (low, moderate, high)
- Die Zahlen an den Knoten repräsentieren, wie viele Parameter für die CPT des betreffenden Knotens bestimmt werden müssen
- 78 Wahrscheinlichkeitswerte insgesamt

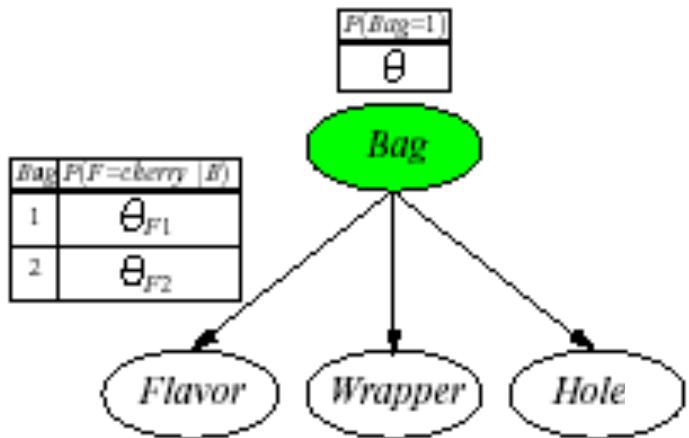
HeartDisease wegzulassen ist gar keine gute Lösung!



- Die Symptome sind nicht länger bedingt unabhängig gegeben die Elternknotenwerte
 - Sehr viel mehr Kanten, sehr viel mehr Wahrscheinlichkeiten zu spezifizieren: 708 insgesamt
 - Wir brauchen sehr viel mehr Daten, um diese ganzen Werte vernünftig zu lernen

Beispiel: Bonbonfabrik wieder einmal

- Zwei Bonbontüten (1 and 2) wurden vermischt
- Bonbons werden durch drei Eigenschaften beschrieben: Flavor und Wrapper wie vorher, plus Hole (ob ein Loch in der Mitte ist)
- Die Merkmale von Bonbon hängen mit bestimmten Wahrscheinlichkeiten von der Tüte ab, aus der sie kommen
- Wir wollen für jedes Bonbon vorhersagen, aus welcher Tüte es kam, je nach vorgefundenen Eigenschaften: Naïve-Bayes-Modell



$$\theta = P(\text{Bag} = 1)$$

$$\theta_{Fj} = P(\text{Flavor} = \text{cherry} \mid \text{Bag} = j)$$

$$\theta_{Wj} = P(\text{Wrapper} = \text{red} \mid \text{Bag} = j)$$

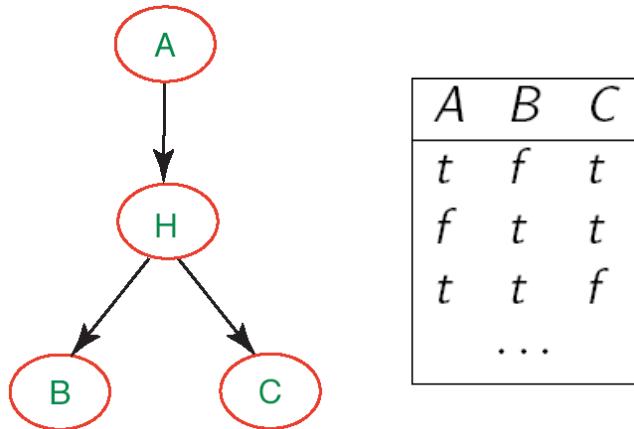
$$\theta_{Hj} = P(\text{Hole} = \text{yes} \mid \text{Bag} = j)$$

$$j = 1, 2$$

Expectation-Maximization (EM)

- Wenn wir versteckte Variablen beibehalten und die Netzwerkparameter aus Daten lernen wollen, haben wir eine Form des **unüberwachten Lernens** (*unsupervised learning*)
 - Die Daten geben nicht über alle Aspekte von Datenpunkten Auskunft
- Expectation-Maximization
 - Genereller Algorithmus zum Lernen von Modellparametern bei unvollständigen Daten
 - Hier für diskrete Datenwerte im BN-Kontext gezeigt

EM: Generelle Idee



- Falls wir Daten für alle Variablen im BN hätten, könnten wir die Parameter mit ML- (oder MAP-) Ansätzen lernen
 - Relative Häufigkeiten bestimmen, wie vorher besprochen
- Falls wir die Parameter hätten, könnten wir die Posterior-Wahrscheinlichkeiten jedes Ereignisses schätzen:

$$P(H|A,B,C)$$

Dempster, A.P., Laird. N.M., Rubin, D.B.: *Maximum-Likelihood from incomplete data via the EM algorithm*. Journal of the Royal Statistical Society, 1977

EM: Generelle Idee

- Der Algorithmus startet mit "erfundenen" (zufällig generierten) Informationen, um das Lernproblem zu lösen:
 - Erfundene Netzwerkparameterwerte (virtuell notiert in den CPTs)
- Dann werden die Initialwerte in zwei Schritten verfeinert:
 - Expectation (E): Aktualisiere die Daten (virtuell) mit aus dem aktuellen Modell hergeleiteten Erwartungen
 - Maximization (M): Gegeben die aktualisierten Daten, aktualisieren die Parameter des BNs mitteln Maximum Likelihood (ML) Ansatz
 - ✓ Das ist der gleiche Schritt, wie im voll beobachtbaren Fall

EM: Wie funktioniert das mit Naive Bayes-Modellen

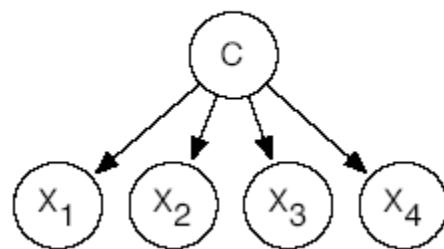
➤ Betrachtet die folgenden Daten, ...

- N Beispiele mit Booleschen Attributen X_1, X_2, X_3, X_4

Data			
X_1	X_2	X_3	X_4
t	f	t	t
f	t	t	f
f	f	t	t
...			

- ... die wir kategorisieren wollen mit möglichen Werten einer Klasse $C = \{1,2,3\}$
- Wir verwenden einen naiven Bayes-Klassifikator mit versteckter Variable C

Model

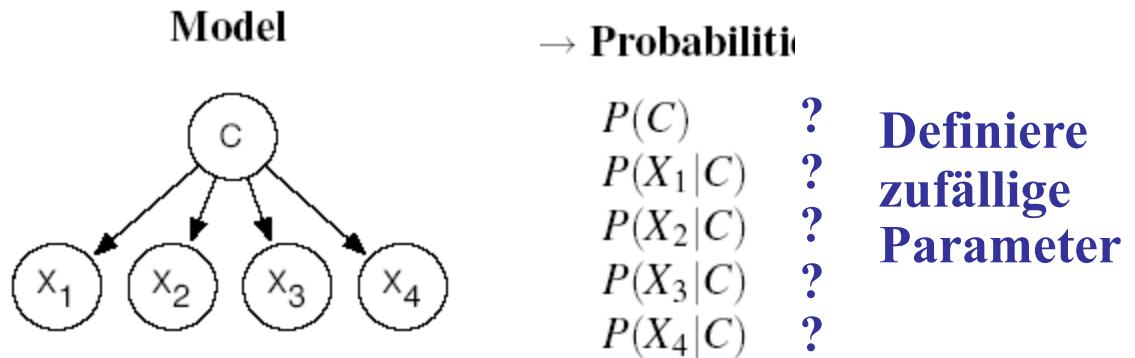


→ Probabilitäten

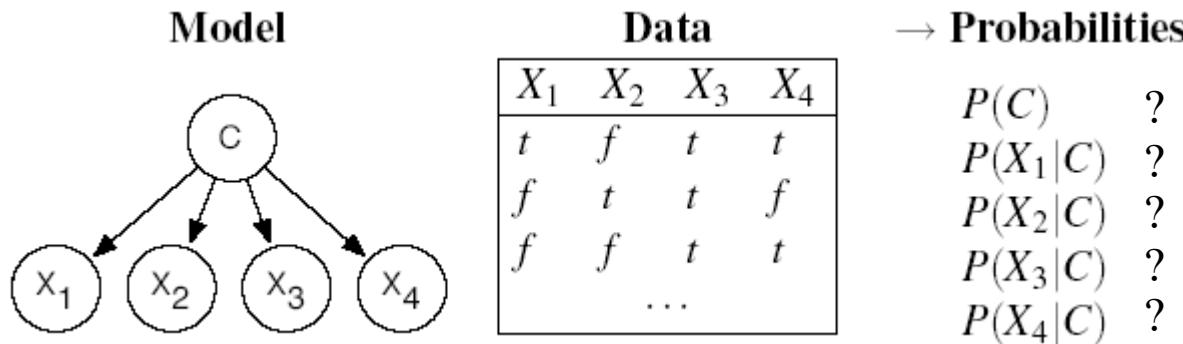
$$\begin{array}{ll} P(C) & ? \\ P(X_1|C) & ? \\ P(X_2|C) & ? \\ P(X_3|C) & ? \\ P(X_4|C) & ? \end{array}$$

EM: Initialisierung

- Der Algorithmus startet mit "erfundenen" (zufällig generierten) Informationen, um das Lernproblem zu lösen:
- Erfundene Netzwerkparameterwerte (virtuell notiert in den CPTs)



EM: Expectation-Schritt (Bestimmung der erwarteter Zahlen)



➤ Was benötigen wir, um die Netzwerkparameter mit dem Maximum-Likelihood-Ansatz zu lernen?

- Für $P(C) = \text{Anzahl}(\text{Datenpunkte mit } C=i) / \text{Anzahl} (\text{alle Datenpunkte}) \quad i=1,2,3$
- Für $P(X_h|C) = \text{Anzahl}(\text{Daten mit } X_h = \text{val}_k \text{ und } C=i) / \text{Anzahl}(\text{Daten mit } C=i)$
für alle Werte val_k von X_h und $i=1,2,3$

EM: Expectation-Schritt (Bestimmung der erwarteter Zahlen)

- Bislang haben wir nur: $N = \text{Anzahl}(\text{Datenpunkte})$
- Wir approximieren alle weiteren Zähler mit erwarteten Zählerwerten, die aus dem Modell mit den "erfundenen" Parametern abgeleitet werden
- Der erwartete Zähler $\hat{N}(C = i)$ ist die Summe, über alle N Beispieldaten, der Wahrscheinlichkeiten, dass jedes Beispiel in Kategorie i fällt

$$\begin{aligned}\hat{N}(C = i) &= \sum_{j=1}^N P(C = i \mid \text{Attribute von Beispiel } e_j) \\ &= \sum_{j=1}^N P(C = i \mid x_{1j}, x_{2j}, x_{3j}, x_{4j})\end{aligned}$$

EM: Expectation-Schritt (Bestimmung der erwarteter Zahlen)

- Wie erhalten wir die notwendigen Werte aus dem Modell?

$$\begin{aligned}\hat{N}(C = i) &= \sum_{j=1}^N P(C = i \mid \text{Attribute von Beispiel } e_j) \\ &= \sum_{j=1}^N P(C = i \mid x_{1j}, x_{2j}, x_{3j}, x_{4j})\end{aligned}$$

- Leicht mit naivem Bayesschen Netzwerk

$$\begin{aligned}P(C = i \mid x_{1j}, x_{2j}, x_{3j}, x_{4j}) &= \frac{P(C = i, x_{1j}, x_{2j}, x_{3j}, x_{4j})}{P(x_{1j}, x_{2j}, x_{3j}, x_{4j})} \\ &= \frac{P(x_{1j} \mid C = i) \dots P(x_{4j} \mid C = i) P(C = i)}{P(x_{1j}, x_{2j}, x_{3j}, x_{4j})}\end{aligned}$$

Auch direkt aus Netzwerk bestimmbar.
Übungsaufgabe

Erfundene Parameter
für das Netzwerk

EM: Expectation-Schritt (Bestimmung der erwarteter Zahlen)

- Durch einen ähnlichen Schritt erhalten wir die erwarteten Zähler für Beispiele mit Attribut $X_h = val_k$ und Kategorie i
- Diese werden später benötigt zur Schätzung von $P(X_h | C)$:

$$P(X_h | C) = \frac{\text{Exp. Counts(examples with } X_h = val_k \text{ and } C = i)}{\text{Exp. Counts(examples with } C = i)} = \frac{\hat{N}(X_h = val_k, C = i)}{\hat{N}(C = i)}$$

- für alle Werte val_k von X_h und $i=1,2,3$
- Zum Beispiel

$$\hat{N}(X_1 = t, C = 1) = \sum_{e_j \text{ with } X_1 = t} P(C = 1 | x_{1j} = t, x_{2j}, x_{3j}, x_{4j})$$

Wiederum erhalten wir diese W'keiten aus dem aktuellen Modell

EM: Generelle Idee

- Der Algorithmus startet mit "erfundenen" (zufällig generierten) Informationen, um das Lernproblem zu lösen:
 - Erfundene Netzwerkparameterwerte (virtuell notiert in den CPTs)
- Dann werden die Initialwerte in zwei Schritten verfeinert:
 - Expectation (E): Aktualisiere die Daten (virtuell) mit aus dem aktuellen Modell hergeleiteten Erwartungen
 - Maximization (M): Gegeben die aktualisierten Daten, aktualisieren die Parameter des BNs mitteln Maximum Likelihood (ML) Ansatz
 - ✓ Das ist der gleiche Schritt, wie im voll beobachtbaren Fall

Maximization-Schritt: (Verfeinerung der Parameter)

- Nun verfeinern wir die Netzwerkparameter mittels Maximum-Likelihood-Lernen auf den erwarteten Zählern

$$P(C = i) = \frac{\hat{N}(C = i)}{N}$$

$$P(X_j = val_k | C = i) = \frac{\hat{N}(X_j = val_k | C = i)}{\hat{N}(C = i)}$$

- für alle Werte val_k von X_j und $i=1,2,3$

EM-Zyklus

- Nun kann der E-Schritt wiederholt werden

Erwartete Zähler
("Augmented data")

X_1	X_2	X_3	X_4	C	count
:	:	:	:	:	:
t	f	t	t	1	
t	f	t	t	2	
t	f	t	t	3	
:	:	:	:	:	:

M-step

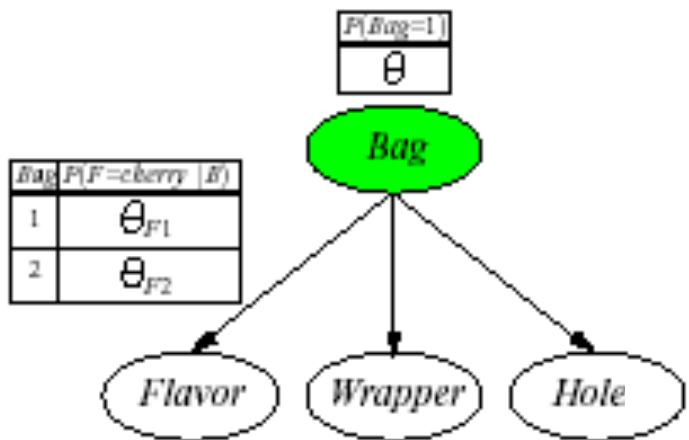
Wahrscheinlichkeiten

$$\begin{aligned}P(C) \\ P(X_1|C) \\ P(X_2|C) \\ P(X_3|C) \\ P(X_4|C)\end{aligned}$$

E-step

Beispiel: Bonbonfabrik wieder einmal

- Zwei Bonbontüten (1 and 2) wurden vermischt
- Bonbons werden durch drei Eigenschaften beschrieben: Flavor und Wrapper wie vorher, plus Hole (ob ein Loch in der Mitte ist)
- Die Merkmale von Bonbon hängen mit bestimmten Wahrscheinlichkeiten von der Tüte ab, aus der sie kommen
- Wir wollen für jedes Bonbon vorhersagen, aus welcher Tüte es kam, je nach vorgefundenen Eigenschaften: Naïve-Bayes-Modell



$$\theta = P(\text{Bag} = 1)$$

$$\theta_{Fj} = P(\text{Flavor} = \text{cherry} \mid \text{Bag} = j)$$

$$\theta_{Wj} = P(\text{Wrapper} = \text{red} \mid \text{Bag} = j)$$

$$\theta_{Hj} = P(\text{Hole} = \text{yes} \mid \text{Bag} = j)$$

$$j = 1, 2$$

Daten

- Nehmen wir an, die wahren Parameter seien:
 - $\theta = 0.5$;
 - $\theta_{F1} = \theta_{W1} = \theta_{H1} = 0.8$;
 - $\theta_{F2} = \theta_{W2} = \theta_{H2} = 0.3$;
- Annahme: Die folgenden Zähler werden "gesampelt" aus $P(C, F, W, H)$ ($N = 1000$): Wir haben einen Datensatz

	W=red		W=green	
	H=1	H=0	H=1	H=0
F=cherry	273	93	104	90
F=lime	79	100	94	167

- Wir wollen nun die wahren Parameter aus diesen Daten mittels EM rekonstruieren

EM: Initialisierung

- Weise Parametern zufällige Initialwerte zu
 - Zu Vereinfachung der Darstellung verwenden wir hier:

$$\theta^{(0)} = 0.6;$$

$$\theta_{F1}^{(0)} = \theta_{W1}^{(0)} = \theta_{H1}^{(0)} = 0.6;$$

$$\theta_{F2}^{(0)} = \theta_{W2}^{(0)} = \theta_{H2}^{(0)} = 0.4$$

- Nun durchlaufen wir einen EM-Zyklus zur Berechnung von $\theta^{(1)}$.

E-Schritt

- Zuerst brauchen wir die erwarteten Zähler für Bonbon von Tüte 1:
- Addiere die Wahrscheinlichkeiten, dass jedes der N Datenpunkte aus Tüte 1 kommt
 - Seien $flavor_j$, $wrapper_j$, $hole_j$ die Werte der entsprechenden Attribute für den j-ten Datenpunkt

$$\begin{aligned}\hat{N}(\text{Bag} = 1) &= \sum_{j=1}^N P(\text{Bag} = 1 | flavor_j, wrapper_j, hole_j) = \\ &= \sum_{j=1}^N \frac{P(flavor_j, wrapper_j, hole_j | \text{Bag} = 1)P(\text{Bag} = 1)}{P(flavor_j, wrapper_j, hole_j)} \\ &= \sum_{j=1}^N \frac{P(flavor_j | \text{Bag} = 1)P(wrapper_j | \text{Bag} = 1)P(hole_j | \text{Bag} = 1)P(\text{Bag} = 1)}{\sum_i P(flavor_j | \text{Bag} = i)P(wrapper_j | \text{Bag} = i)P(hole_j | \text{Bag} = i)P(\text{Bag} = i)}\end{aligned}$$

E-step

$$\sum_{j=1}^N \frac{P(\text{flavor}_j | \text{Bag} = 1) P(\text{wrapper}_j | \text{Bag} = 1) P(\text{hole}_j | \text{Bag} = 1) P(\text{Bag} = 1)}{\sum_i P(\text{flavor}_j | \text{Bag} = i) P(\text{wrapper}_j | \text{Bag} = i) P(\text{hole}_j | \text{Bag} = i) P(\text{Bag} = i)}$$

- Die Summation kann in die 8 Bonbongruppen aus der Tabelle aufgebrochen werden.
 - Zum Beispiel ergibt die Summe über 273 cherry-Bonbons mit rotem Papier und einem Loch (erster Eintrag in der Tabelle)

	W=red		W=green	
	H=1	H=0	H=1	H=0
F=cherry	273	93	104	90
F=lime	79	100	94	167

$$= 273 \frac{\theta_{F1}^{(0)} \theta_{W1}^{(0)} \theta_{H1}^{(0)} \theta^{(0)}}{\theta_{F1}^{(0)} \theta_{W1}^{(0)} \theta_{H1}^{(0)} \theta^{(0)} + \theta_{F2}^{(0)} \theta_{W2}^{(0)} \theta_{H2}^{(0)} (1 - \theta^{(0)})} = \\ 273 \frac{0.6^4}{0.6^4 + 0.4^4} = 273 \frac{0.1296}{0.1552} = 227.97$$

$$\theta^{(0)} = 0.6; \\ \theta_{F1}^{(0)} = \theta_{W1}^{(0)} = \theta_{H1}^{(0)} = 0.6; \\ \theta_{F2}^{(0)} = \theta_{W2}^{(0)} = \theta_{H2}^{(0)} = 0.4$$

M-Schritt

- Mit der Berechnung der anderen 7 Bonbongruppe erhalten wir

$$\hat{N}(\text{Bag} = 1) = 612.4$$

- Nun führen wir den M-Schritt aus, um θ zu verfeinern. Hierzu nehmen wir den erwarteten Zählerwert der Datenpunkte aus Tüte 1

$$\theta^{(1)} = \frac{\hat{N}(\text{Bag} = 1)}{N} = 0.6124$$

Noch ein Parameter...

- Wir machen das gleiche für den Parameter θ_{F1}
- E-Schritt: Bestimme erwarteten Zählerwert von cherry-Bonbons aus Tüte 1

$$\hat{N}(Bag = 1 \wedge Flavor = cherry) = \sum_{j:Flavor_j = cherry} P(Bag = 1 / Flavor_j = cherry, wrapper_j, hole_j)$$

- Bestimmbar aus naivem Bayesschen Modell wie vorher besprochen
- Als Übung...
- M-Schritt: Verfeinere θ_{F1} durch Bestimmung der relativen Häufigkeiten

$$\theta_{F1}^{(1)} = \frac{\hat{N}(Bag = 1 \wedge Flavor = cherry)}{\hat{N}(Bag = 1)}$$

Lernergebnis

- Nach einem vollen Zyklus über alle Parameter haben wir

$$\theta^{(1)} = 0.6124;$$

$$\theta_{F1}^{(1)} = 0.6684; \quad \theta_{W1}^{(1)} = 0.6483; \quad \theta_{H1}^{(1)} = 0.658;$$

$$\theta_{F2}^{(1)} = 0.3887; \quad \theta_{W2}^{(1)} = 0.3817; \quad \theta_{H2}^{(1)} = 0.3827;$$

- Für jeden Satz von Parametern können wir die Log-Likelihood bestimmen wie auch schon früher
- Man kann zeigen, dass dieser Wert mit jeder EM-Iteration steigt (Konvergenz)
- EM mit Maximum-Likelihood bleibt aber bei lokalen Maxima hängen bleiben (ggf. mehrere Startwerte nehmen, Priors berücksichtigen oder MAP nehmen)

Lernergebnis

- Nach einem vollen Zyklus über alle Parameter haben wir

$$\theta^{(1)} = 0.6124;$$

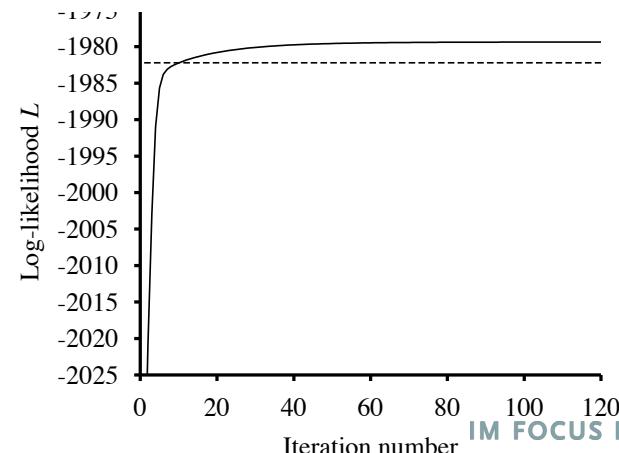
$$\theta_{F1}^{(1)} = 0.6684; \quad \theta_{W1}^{(1)} = 0.6483; \quad \theta_{H1}^{(1)} = 0.658;$$

$$\theta_{F2}^{(1)} = 0.3887; \quad \theta_{W2}^{(1)} = 0.3817; \quad \theta_{H2}^{(1)} = 0.3827;$$

- Für jeden Satz von Parametern können wir die Log-Likelihood bestimmen wie auch schon früher

$$P(\mathbf{d} | h_{\theta^{(i)} \theta_{F1}^{(i)} \theta_{W1}^{(i)} \theta_{H1}^{(i)} \theta_{F2}^{(i)} \theta_{W2}^{(i)} \theta_{H2}^{(i)}}) = \prod_{j=1}^{1000} P(d_j | h_{\theta^{(i)} \theta_{F1}^{(i)} \theta_{W1}^{(i)} \theta_{H1}^{(i)} \theta_{F2}^{(i)} \theta_{W2}^{(i)} \theta_{H2}^{(i)}})$$

- Wir können zeigen, dass die Log-Likelihood in jeder Iteration ansteigt, so dass die initiale Likelihood schon nach drei Iterationen überflügelt wird



EM: Diskussion

➤ Für komplexere BNs ist der Algorithmus der gleiche

- Im allgemeinen müssen wir die CPT-Einträge für jede Variable X_i gegeben die Elternknotenwerte Pa_i berechnen
- $\theta_{ijk} = P(X_i = x_{ij} | Pa_i = pa_{ik})$

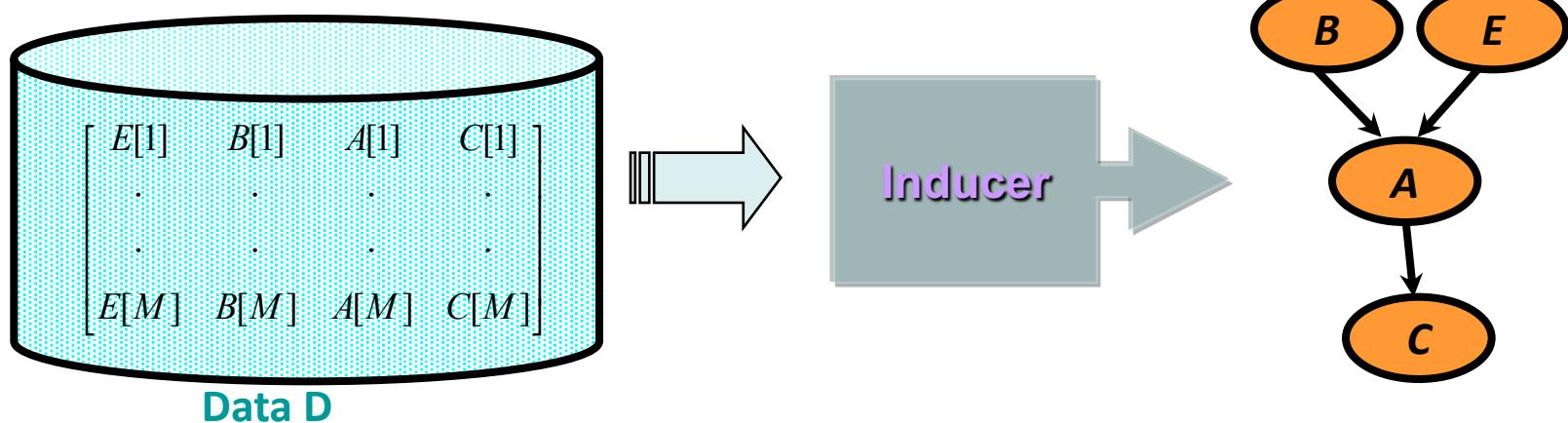
$$\theta_{ijk} = \frac{\hat{N}(X_i = x_{ij}; Pa_i = pa_{ik})}{\hat{N}(Pa_i = pa_{ik})}$$

- Die erwarteten Zählerwerte werden durch Summierung über die Beispiele berechnet, nachdem all notwendigen $P(X_i = x_{ij}, Pa_i = pa_{ik})$ mittels BN-Algorithmen bestimmt sind
- Das Verfahren kann in eine kombinatorische Explosion laufen (hierfür gibt es dann Approximationstechniken)



Learning Bayesian network structures

- Given training set $D = \{x[1], \dots, x[M]\}$
- Find model that best matches D
 - model selection
 - parameter estimation



Some of the following slides from an AI course „Graphical models“
by Burgard/De Raedt/Kersting/Nebel

Model selection

Goal: Select the best network structure, given the data

Input:

- Training data
- Scoring function

Output:

- A network that maximizes the score

Structure selection: Scoring

- Bayesian: prior over parameters and structure
 - get balance between model complexity and fit to data as a byproduct

Can we learn G's params from D?

Does G explain D with ML?

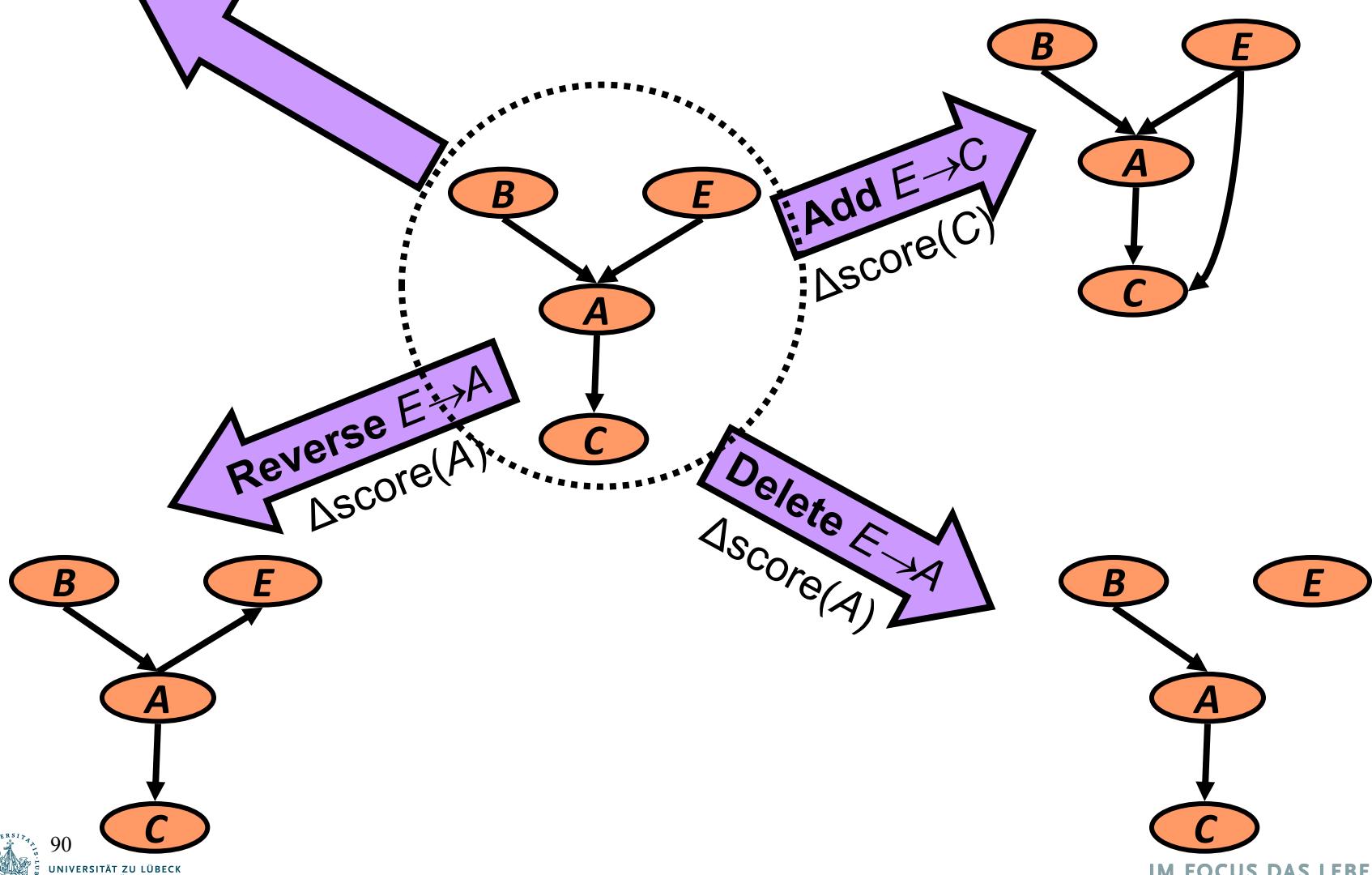
Prior w.r.t. MDL

- $\text{Score}_D(G) = \log P(G|D) = \alpha \log [P(D|G) P(G)]$
- Marginal likelihood just comes from our parameter estimates
- Prior on structure can be any measure we want; typically a function of the network complexity (MDL principle)

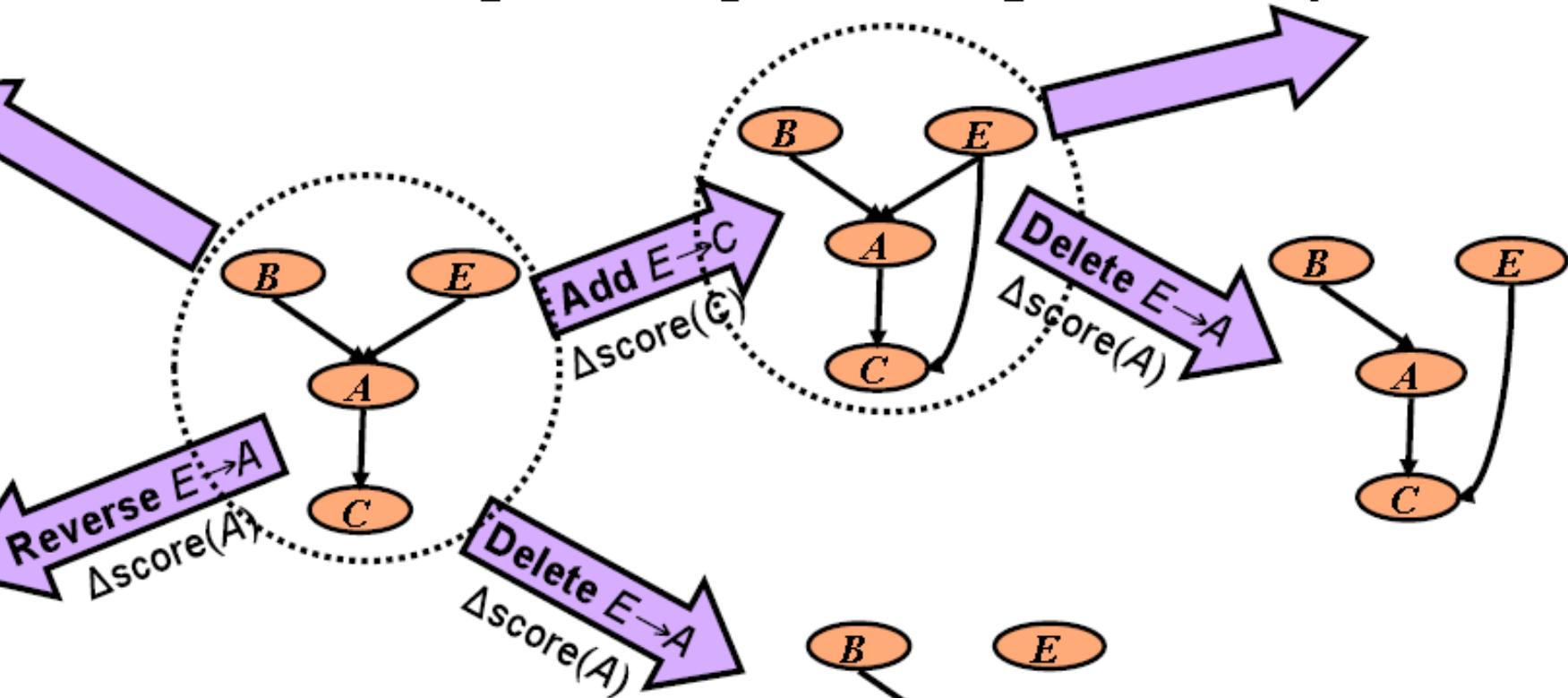
Same key property: Decomposability

$$\text{Score}(\text{structure}) = \sum_i \text{Score}(\text{substructure of } X_i)$$

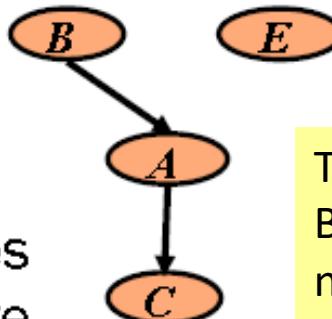
Heuristic search



Exploiting decomposability



To recompute scores,
only need to re-score families
that changed in the last move

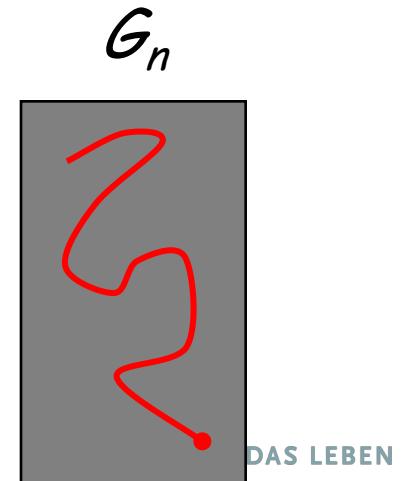
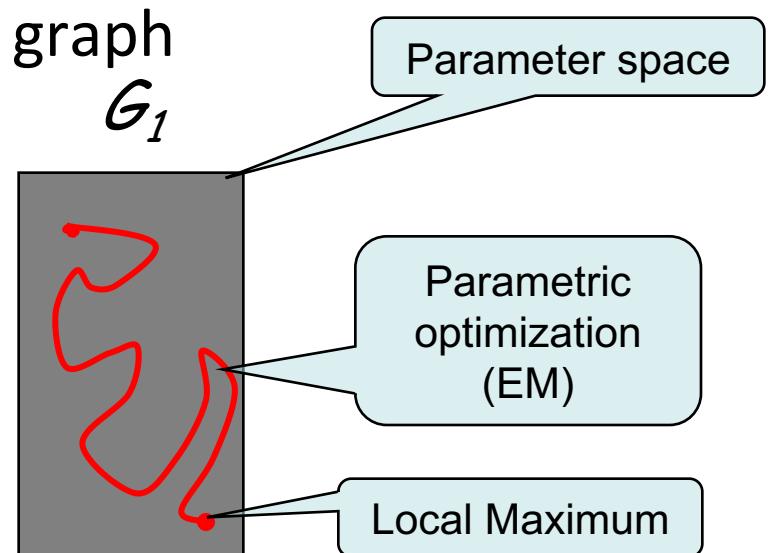
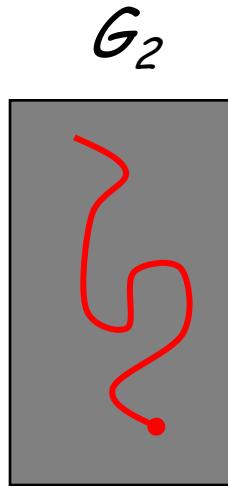
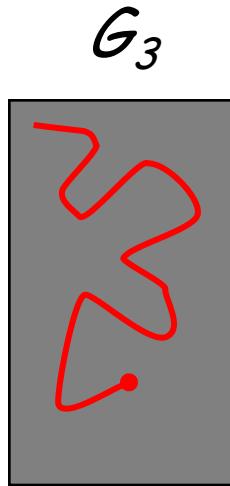


That's fine.
But what to do if we have
non-complete data??

Cannot use decomposability

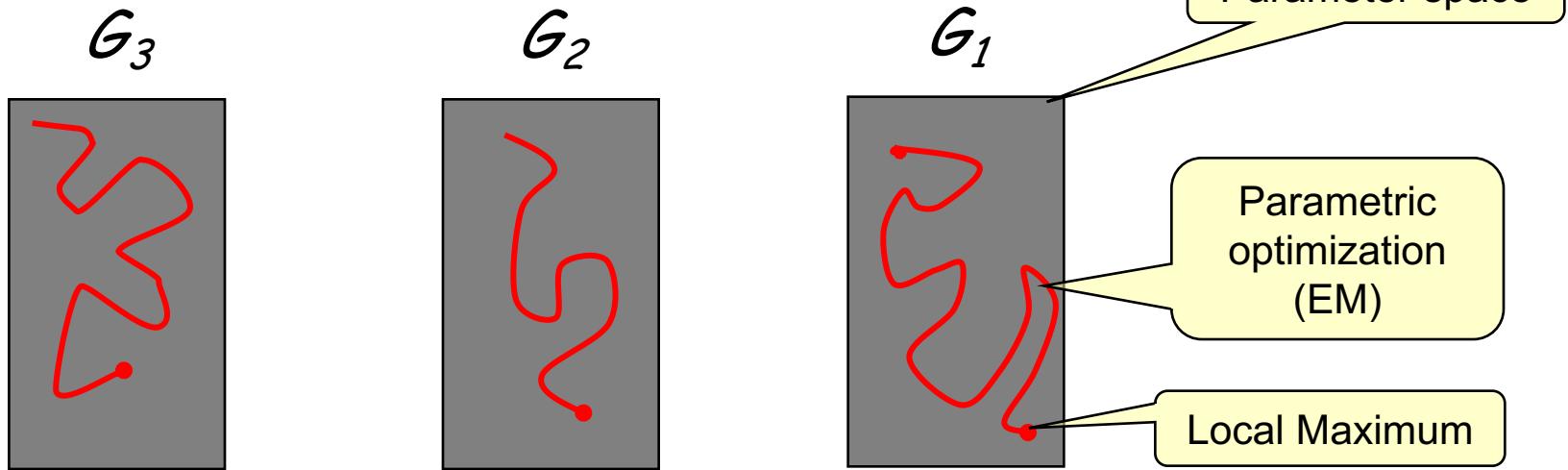
Local Search in Practice

- Perform EM for each candidate graph



Local Search in Practice

- Perform EM for each candidate graph



- ◆ Computationally expensive:
 - Parameter optimization via EM — non-trivial
 - Need to perform EM for all candidate structures
 - Spend time even on poor candidates
- ⇒ In practice, considers only a few candidates

Structural EM [Friedman et al. 98]

Recall, in complete data we had

- Decomposition \Rightarrow efficient search

Idea:

- Instead of optimizing the real score...
- Find **decomposable** alternative score
- Such that maximizing new score
 \Rightarrow improvement in real score

Structural EM

Idea:

- Use current model to help evaluate new structures

Outline:

- Perform search in (Structure, Parameters) space
- At each iteration, use current model for finding either:
 - Better scoring parameters: “parametric” EM step or
 - Better scoring structure: “structural” EM step

- Score for structure G and parameterization θ given data over observed RVs O

$$\text{Score}_O(G, \theta) = \log P(O|G, \theta) - \text{Pen}(G, \theta, O)$$

- Handle hiddens H (non-observed) by conditionalizing (like in full bayesian learning) using current structure G^* and parameterization θ^*

$$Q(G, \theta : G^*, \theta^*) = E_{h \sim P(h|O:G^*, \theta^*)} [\log P(O, h|G, \theta)] - \text{Pen}(G, \theta, O)$$

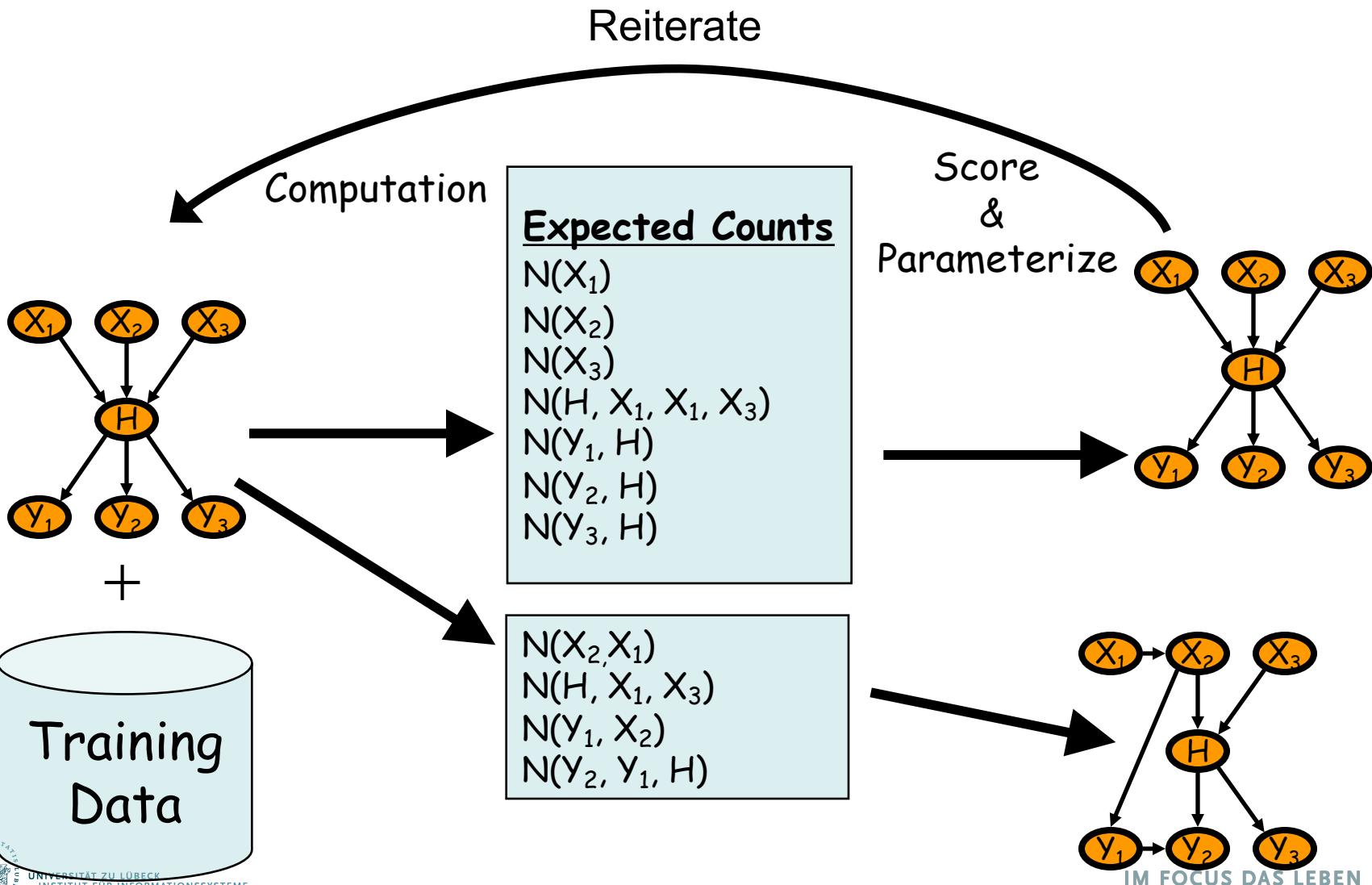
Alternating model selection EM

Choose G^0 and θ^0 randomly

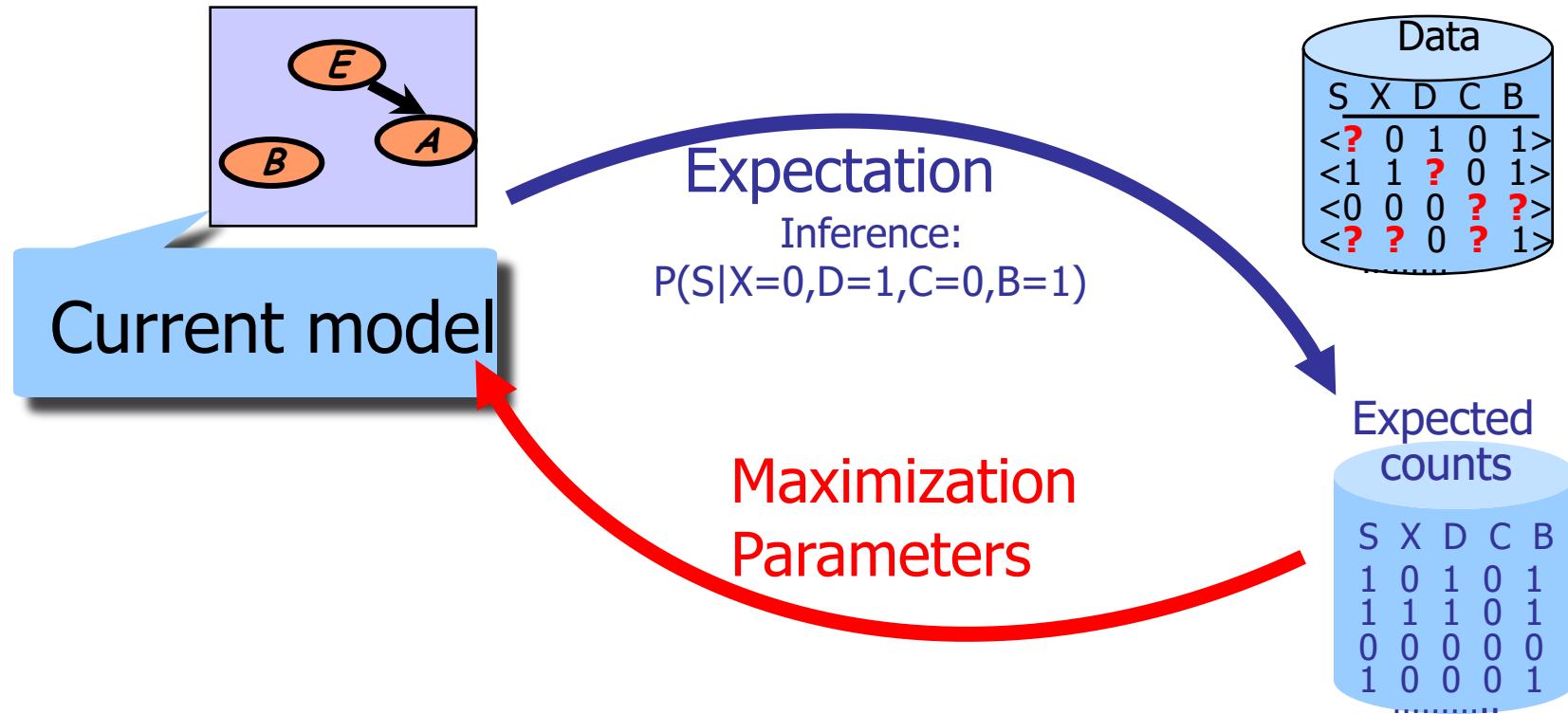
Loop for $n=0, 1, \dots$ until convergence

Find G^{n+1}, θ^{n+1} s.t. $Q(G^{n+1}, \theta^{n+1}: G^n, \theta^n) > Q(G^n, \theta^n: G^n, \theta^n)$

Structural EM



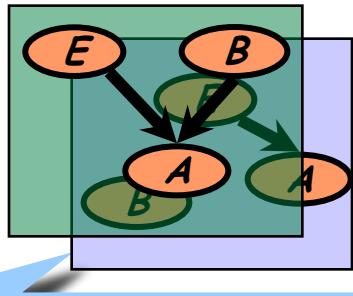
Structure Learning: incomplete data



EM-algorithm:
iterate until convergence



Structure Learning: incomplete data



Current model

Expectation
Inference:
 $P(S|X=0,D=1,C=0,B=1)$

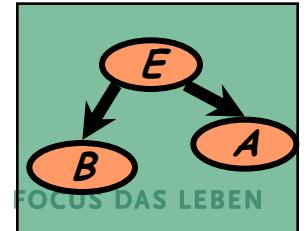
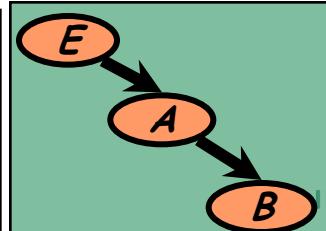
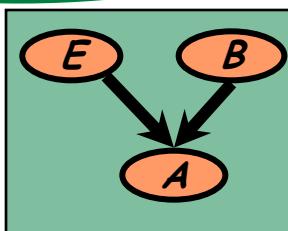
Data				
S	X	D	C	B
<?	0	1	0	1
<1	1	?	0	1
<0	0	0	?	?
<?	?	0	?	1

Expected counts				
S	X	D	C	B
1	0	1	0	1
1	1	1	0	1
0	0	0	0	0
1	0	0	0	1

Maximization
Parameters

Maximization
Structure

SEM-algorithm:
iterate until convergence



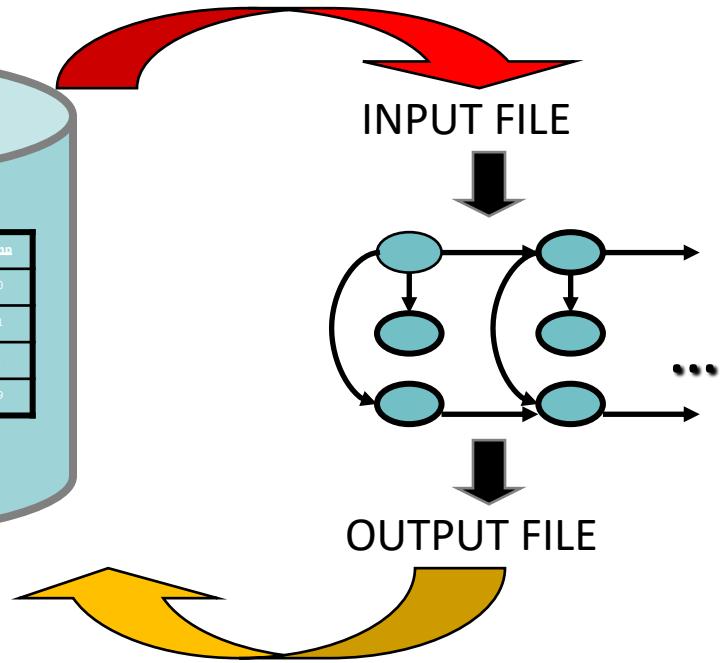
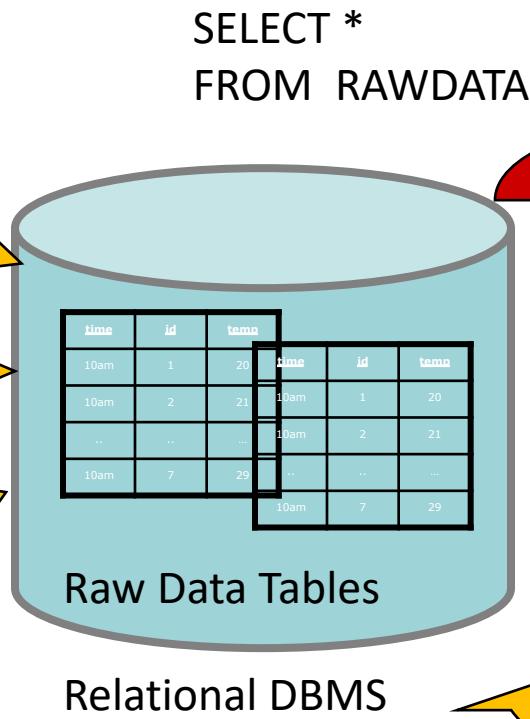
Variations on a theme

- **Known structure, fully observable:** only need to do parameter estimation
- **Known structure, hidden variables:** use expectation maximization (EM) to estimate parameters
- **Unknown structure, fully observable:** do heuristic search through structure space, then parameter estimation
- **Unknown structure, hidden variables:** structural EM

Big Datasets Destroy Simple Abstractions



Sensor/RFID streams
(+ metadata, floor plans, ...)



HeisenData -- Towards a Next-Generation Uncertain Data Management System

Project Funding: FP7 Marie-Curie International Reintegration Grants

(FP7-PEOPLE-2009-RG, Reference No. 249217)

Project Duration: **1/3/2010 - 28/2/2014**

Fellow & Scientist-in-Charge: Prof. Minos Garofalakis

Managing Large, Uncertain Data

Repositories with Probabilistic Graphical Models

Daisy Zhe Wang+, Eirinaios Michelakis+, Minos Garofalakis*+, Joseph M.

Hellerstein+ University of California Berkeley+, Yahoo! Research*

25th August **2008**, VLDB

BayesStore Data Model

1. Incomplete Relation -- R^p
2. Distribution over Possible Worlds – F

Sensor1(Time(T), Room(R), Sid, Temperature(Tp)^p, Light(L)^p)

***Incomplete Relation of
Sensor1^p***

	T	R	Sid	Tp ^p	L ^p
t1	1	1	1	Hot	X1
t2	1	1	2	Cold	Drk
t3	1	1	3	X2	X3
t4	1	2	1	X4	Brt
t5	1	2	2	Hot	X5
t6	1	2	3	X6	X7

***Probabilistic Distribution of
Sensor1^p***

$$F = \Pr [X_1, \dots, X_7]$$

N: number of missing values
 $|X|$: size of the domain

$$|F| = \Theta(|X|^N)$$

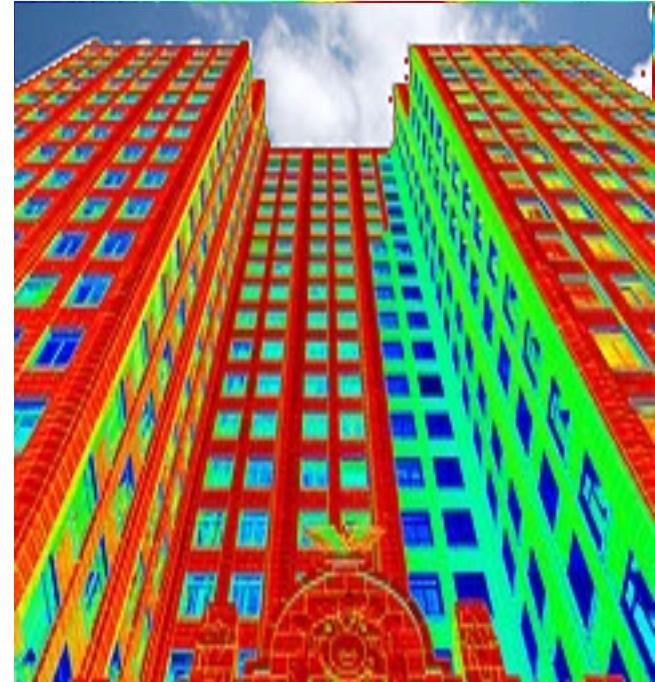
The Skyscrapers Example

For all sensor in all rooms at all timestamp, Light and Temperature readings are correlated.

Light



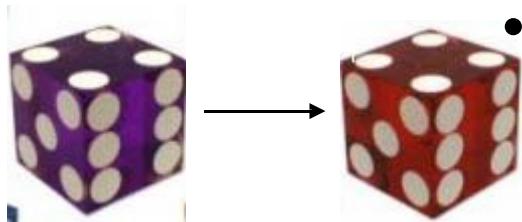
Temperature



Definitions



- **Stripe:** A family of random variables from the same probabilistic attribute.



- **First-order Factor:** A family of local models, which share the same structure and conditional probability table(CPT).



- **BayesStore Data Type:** The input and output abstract data type of queries in BayesStore, which consists of data and model.
- **Possible Worlds**

F as a First-order Bayesian Network (I)

Sensor1^p

	T	R	Sid	Tp ^p	L ^p
t1	1	1	1	H 	X1
t2	1	1	2	Cold	Drk
t3	1	1	3	X2	X3
t4	1	2	1	X 	Brt
t5	1	2	2	Hot	X5
t6	1	2	3	X6	X7
t7	2	1	1	H 	X8
t8	2	1	2	Cold	Drk
t9	2	1	3	X9	X10
t10	2	2	1	X 	Brt
t11	2	2	2	Hot	X12
t12	2	2	3	X13	X14

Stripe (FO Variable) Definitions



All Tp values in Sensor1^p with Sid=1

F as a First-order Bayesian Network (I)

Sensor1^p

T	R	Sid	Tp ^p	L ^p
t1	1	1	H	X1
t2	1	2	Cd	Dr
t3	1	3	>	X3
t4	1	2	>	Bi
t5	1	2	H	X5
t6	1	2	>	X7
t7	2	1	H	X8
t8	2	1	Cd	Dr
t9	2	1	>	X1
t10	2	2	>	Bi
t11	2	2	H	X1
t12	2	2	>	X1

Stripe (FO Variable) Definitions



All Tp values in Sensor1^p with Sid=1



All Tp values in Sensor1^p with Sid=2



All Tp values in Sensor1^p with Sid !=2



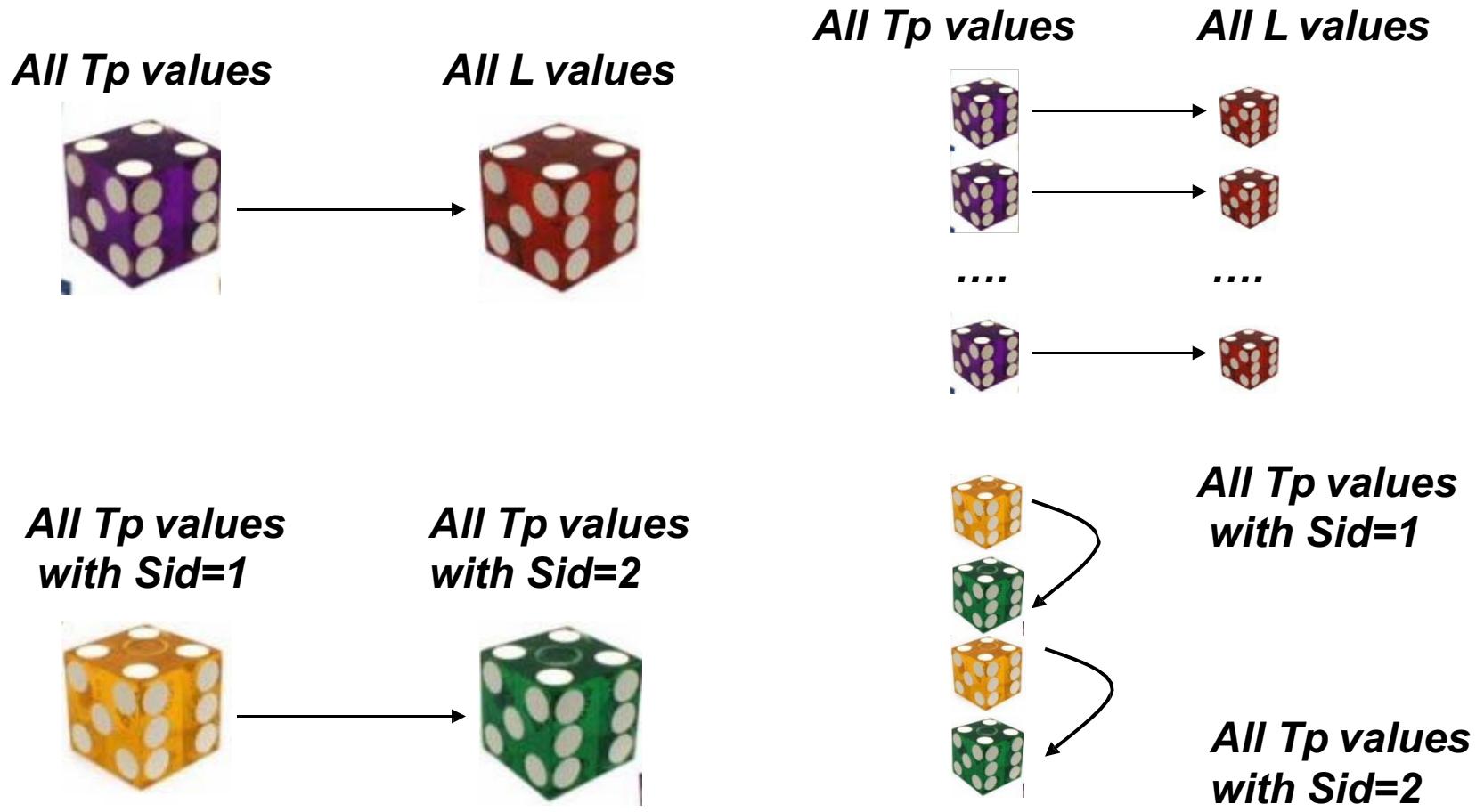
All Tp values in Sensor1^p



All L values in Sensor1^p

F as a First-order Bayesian Model

Mapping between Stripes



F as a First-order Bayesian Model

First-order Factor Definitions

All Tp values



All L values



*All Tp values
with Sid=1*



*All Tp values
with Sid=2*



*All Tp values
with Sid !=2*

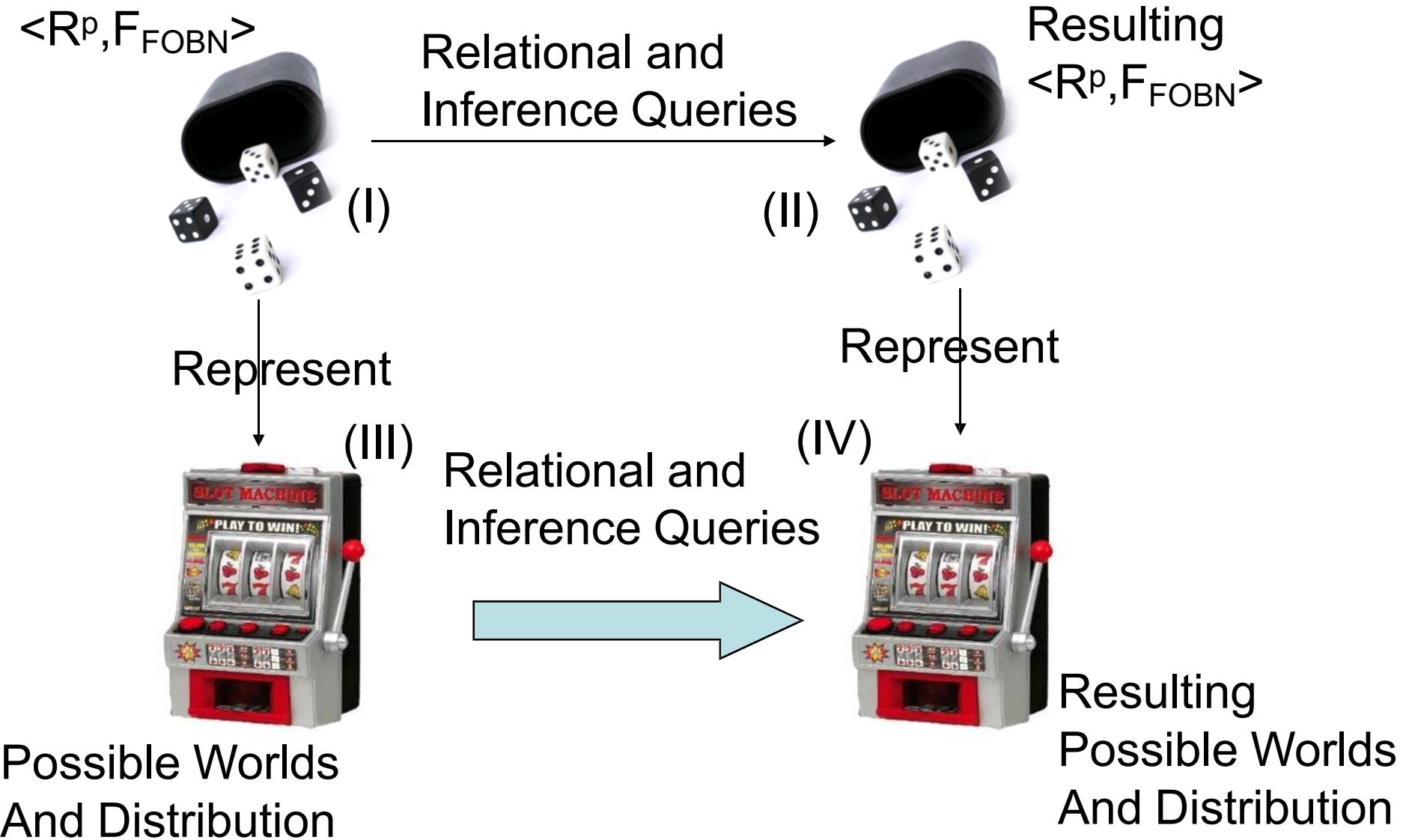


Tp	L	p
Cold	Brt	0.1
Hot	Brt	0.9
Hot	Drk	0.1
Cold	Drk	0.9

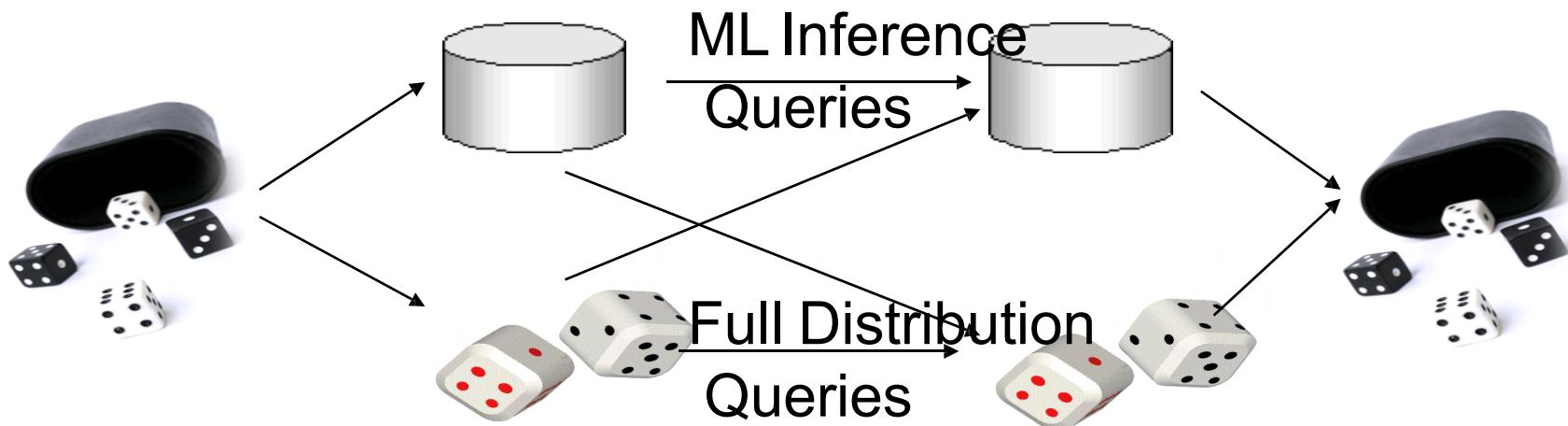
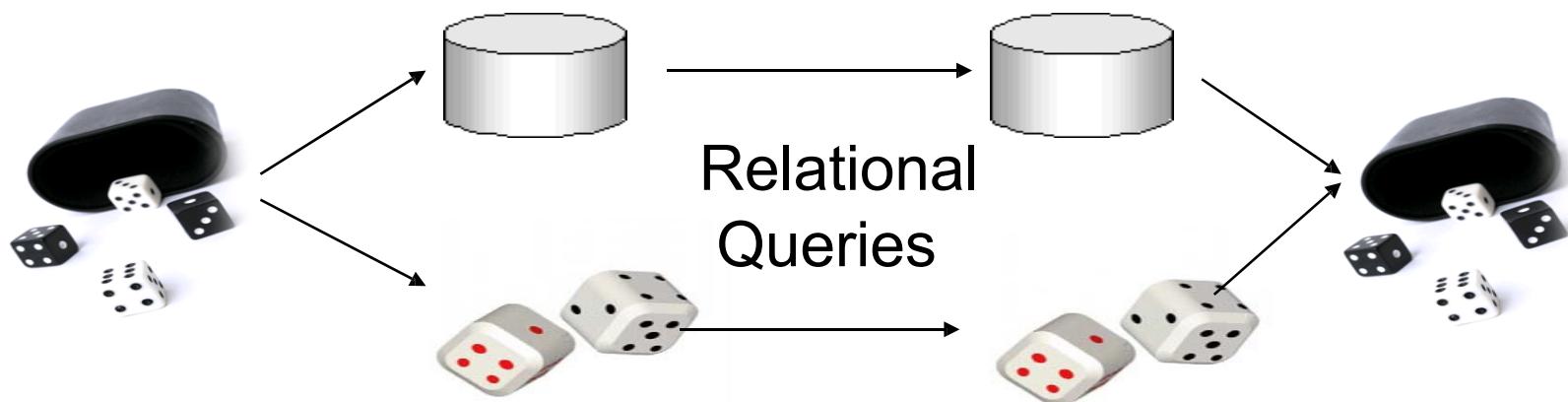
Tp1	Tp2	p
Cold	Cold	0.1
Cold	Hot	0.9
Hot	Hot	0.1
Hot	Cold	0.9

Tp	p
Cold	0.6
Hot	0.4

Query Semantics



Query Algebra



Selection

- Selection over Incomplete Relation R^p
- Selection over Model M_{FOBN}

Sensor1^p

	T	R	Sid	Tp ^p	Lp
t1	1	1	1	Hot	X1
t2	1	1	2	Cold	Drk
t3	1	1	3	X2	X3
t4	1	2	1	X4	Brt
t5	1	2	2	Hot	X5
t6	1	2	3	X6	X7

$\sigma_{Tp=Cold}$



Sensor1^p

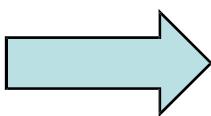
Selection

- Selection over Incomplete Relation R^p
- Selection over Model M_{FOBN}

Sensor1^p

	T	R	Sid	Tp ^p	L ^p
t1	1	1	1	Hot	X1
t2	1	1	2	Cold	Drk
t3	1	1	3	X2	X3
t4	1	2	1	X4	Brt
t5	1	2	2	Hot	X5
t6	1	2	3	X6	X7

$\sigma_{Tp=Cold}$



Sensor1^p

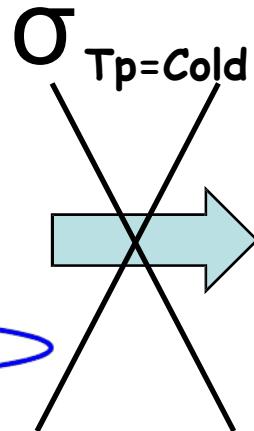
	T	R	Sid	Tp ^p	L ^p
t2	1	1	2	Cold	Drk
t3	1	1	3	X2	X3
t4	1	2	1	X4	Brt
t6	1	2	3	X6	X7

Selection

- Selection over Incomplete Relation R^p
- Selection over Model M_{FOBN}

Sensor1^p

	T	R	Sid	Tp ^p	L ^p
t1	1	1	1	Hot	X1
t2	1	1	2	Cold	Drk
t3	1	1	3	X2	X3
t4	1	2	1	X4	Brt
t5	1	2	2	Hot	X5
t6	1	2	3	X6	X7



Sensor1^p

	T	R	Sid	Tp ^p	L ^p
t2	1	1	2	Cold	Drk
t3	1	1	3	X2	X3
t4	1	2	1	X4	Brt
t6	1	2	3	X6	X7

Selection

- Selection over Incomplete Relation R^p
- Selection over Model M_{FOBN}

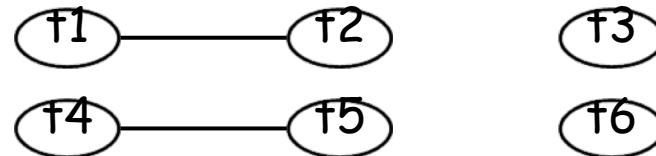
Sensor1 ^p				
T	R	Sid	Tp ^p	L ^p
t1	1	1	Hot	X1
t2	1	1	Cold	Drk
t3	1	1	X2	X3
t4	1	2	X4	Brt
t5	1	2	Hot	X5
t6	1	2	X6	X7

$\sigma_{Tp^p = \text{Cold} | \text{Null}}$

T	R	Sid	Tp ^p	L ^p
t2	1	1	Cold	Drk
t3	1	1	X2	X3
t4	1	2	X4	Brt
t6	1	2	X6	X7

Compute Transitive Closure over TCG

Tuple Correlation Graph (TCG) for $F_{FOBN}(\text{Sensor1})$



T	R	Sid	Tp ^p	L ^p
1	1	1	Hot	X1
1	1	2	Cold	Drk
1	1	3	X2	X3
1	2	1	X4	Brt
1	2	2	Hot	X5
1	2	3	X6	X7

t1
t2
t3
t4
t5
t6

Selection

- Selection over Incomplete Relation R^p
- Selection over Model M_{FOBN}

Probabilistic Distribution F_{FOBN} of Sensor1^p

All Tp values



All L values



Tp	L	p
Cold	Brt	0.1
Hot	Brt	0.9
Hot	Drk	0.1
Cold	Drk	0.9

All Tp values
with Sid=1



All Tp values
with Sid=2

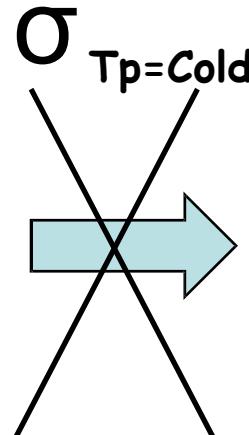


All Tp values
with Sid != 2



Tp1	Tp2	p
Cold	Cold	0.9
Cold	Hot	0.1
Hot	Hot	0.9
Hot	Cold	0.1

Tp	p
Cold	0.6
Hot	0.4



$F_{FOBN} \mid Tp=Cold$

Selection

- Selection over Incomplete Relation R^p

- Selection over Model M_{FOBN}

$Sensor1(T, R, Sid, Tp^p, L^p, \text{Exist}(E)^p)$

F_{FOBN} of $Sensor1^p$

All Tp values



All L values



$\sigma_{Tp=\text{Cold}}$

All Tp values
with $Sid=1$



All Tp values
with $Sid=2$



All Tp values
with $Sid \neq 2$



Tp	L	p
Cold	Brt	0.1
Hot	Brt	0.9
...

All Tp values

Tp1		
Cold		
Cold	Hot	0.1
Hot		
Hot		

$pr(E=1)=1$ iff $Tp=\text{Cold}$
 $pr(E=0)$ iff $Tp=\text{Hot}$

Tp	p
Cold	0.6
Hot	0.4

$\sigma_{Tp=\text{Cold}}$

All Exist values



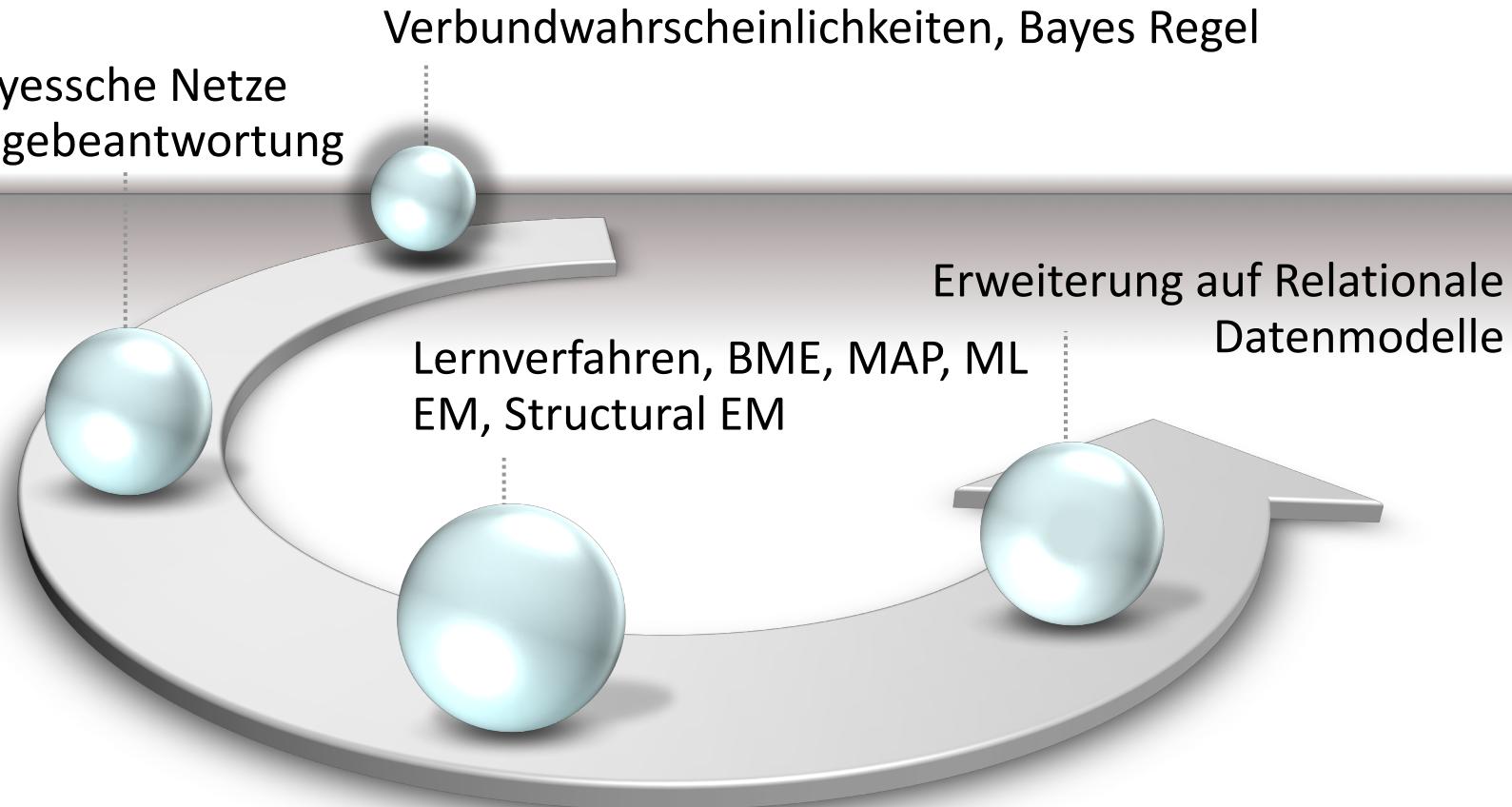
IM FOCUS DAS LEBEN

Project & Join

- Project
 - Project over Incomplete Relation – projected attributes and correlated attributes
 - Project over Model – retrieve only part of the model relevant to the projected attributes
- Join
 - Join over Incomplete Relations with deterministic join condition (e.g. Sensor1.Sid = Sensor2.Sid)
 - Join over Models by merging the local models for Exist^p attribute
 - Probabilistic selection with probabilistic join condition (e.g. Sensor1.Light^p = Sensor2.Light^p)

Non-Standard-Datenbanken

Bayessche Netze, Inferenz, Lernen, Relationale Erweiterung



Ausblick

- Vertiefung zum Thema
Probabilistische Relationale Modellierung
im Master-Programm
- Vorlesung **Web-Mining-Agents**

