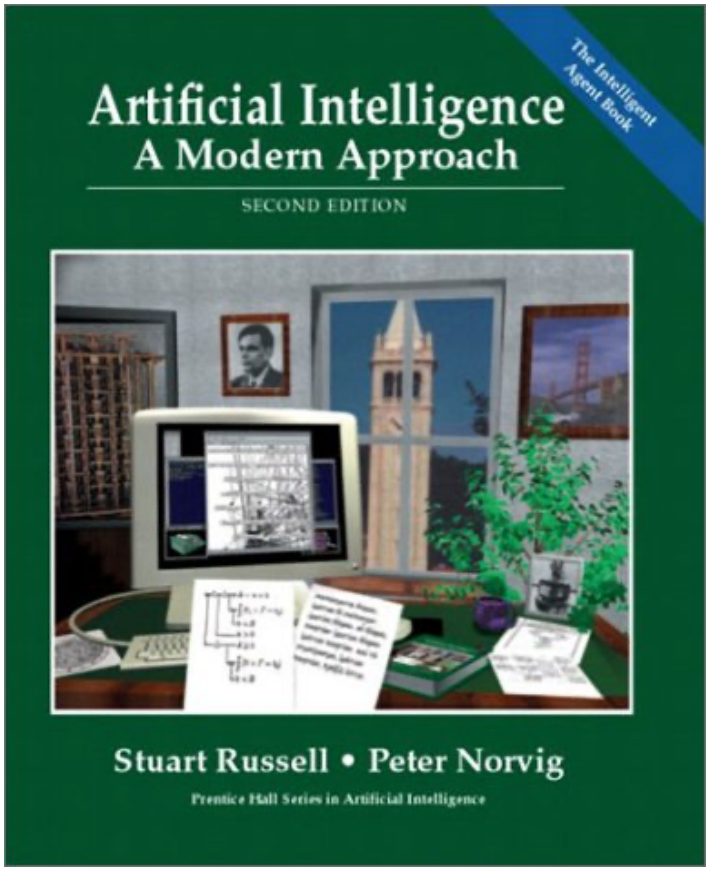
Foundations of Learning Bayesian Networks

Prof. Dr. Ralf Möller
Universität zu Lübeck
Institut für Informationssysteme

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) $h_1 \ldots, h_n$
  - $P(H)$ = prior probability distribution over hypothesis space
- j_{th} observation d_j gives the outcome of random variable D_j
  - training data $\mathbf{d} = d_1,..,d_k$

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? $P(h_\theta | \mathbf{d})$.
- What flavour will the next candy be? Prediction $\mathbf{P}(X|\mathbf{d})$

Full Bayesian Learning

- Given the data so far, each hypothesis h_i has a posterior probability:
  - $P(h_i \mid \mathbf{d}) = \alpha P(\mathbf{d} \mid h_i)\, P(h_i)$ (**Bayes theorem**)
  - where $P(\mathbf{d} \mid h_i)$ 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:
  - $\mathbf{P}(X \mid \mathbf{d}) =$
    $= \sum_i \mathbf{P}(X, h_i \mid \mathbf{d})$
    $= \sum_i \mathbf{P}(X \mid h_i, \mathbf{d})\, P(h_i \mid \mathbf{d})$
    $= \sum_i \mathbf{P}(X \mid h_i)\, P(h_i \mid \mathbf{d})$
    $\sim \sum_i \mathbf{P}(X \mid h_i)\, P(\mathbf{d} \mid h_i)\, P(h_i)$
  - The weights are given by the data likelihood and prior of each h

The data does not add anything to a prediction given a hypothesis

+ No need to pick one best-guess hypothesis!

– Need to iterate over all hypotheses

Man findet $\propto$ oder $\sim$ oder $= \alpha \cdot \ldots$

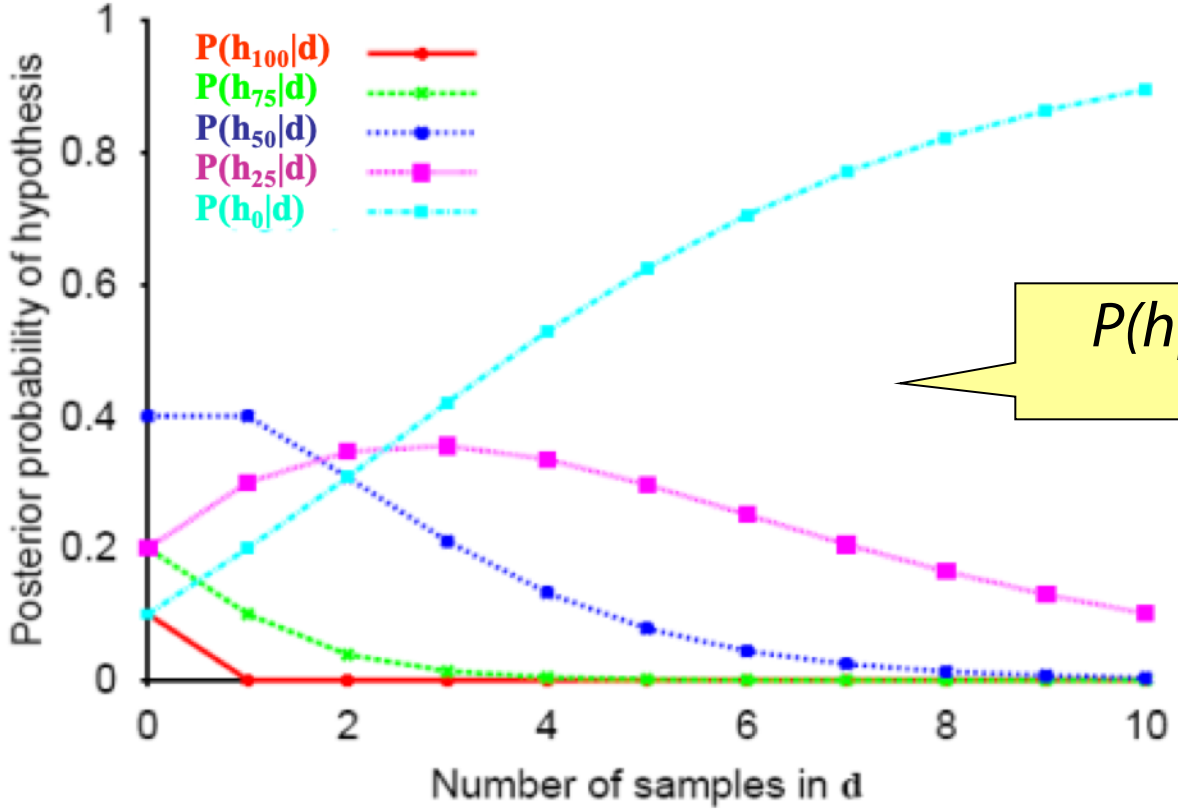
Example

- If we re-wrap each candy and return it to the bag, our 10 observations are independent and identically distributed, i.i.d, so
  - $P(\mathbf{d}|\, h_\theta) = \Pi_j P(d_j|\, h_\theta)$ for $j=1,..,10$
- 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 l = N-c lime

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

  - **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:
  - $P([\text{lime, lime, lime}] \mid h_{50}) = 0.5^3$ because the probability of seeing lime for each observation is 0.5 under this hypotheses

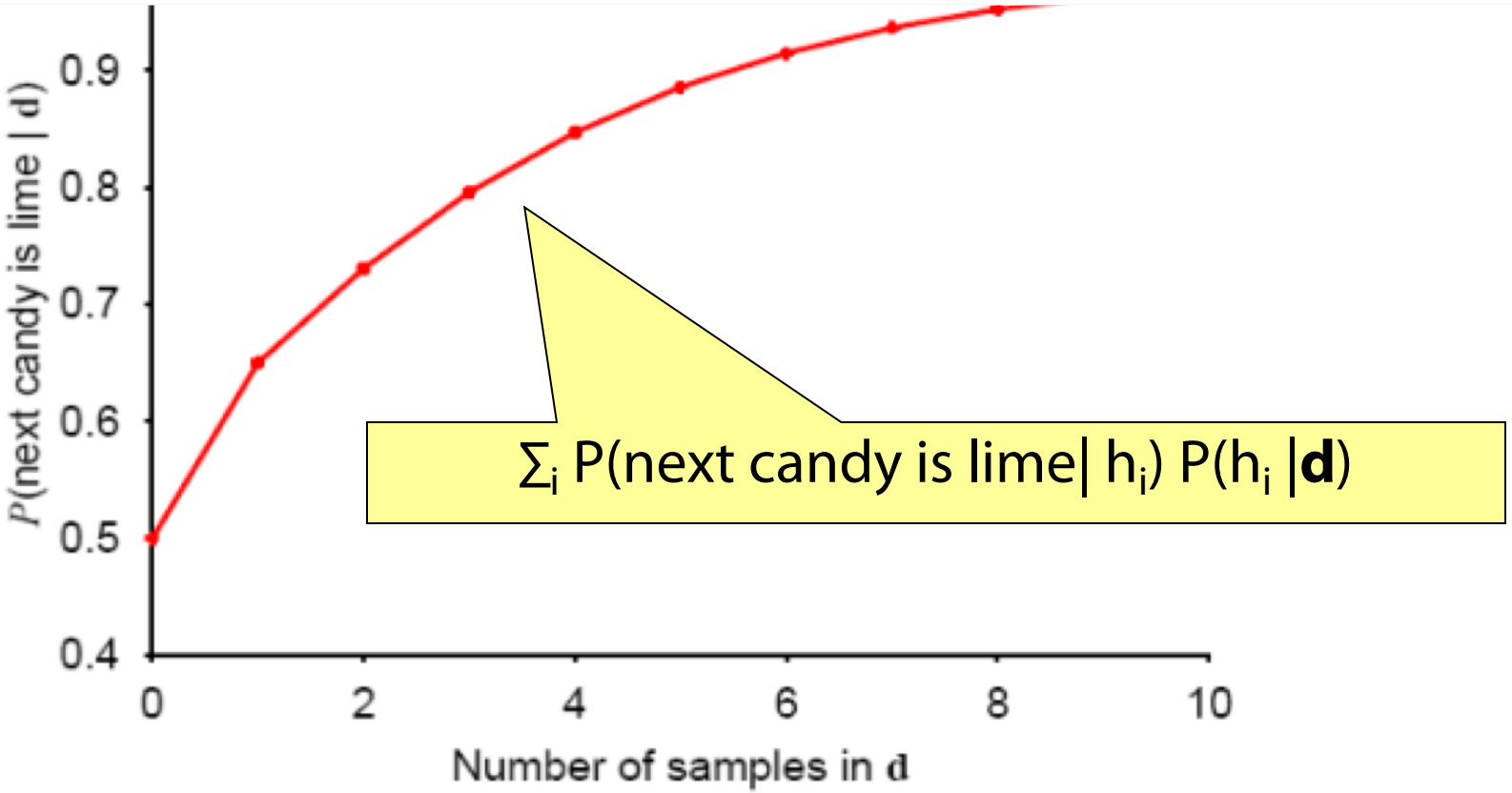
All-limes: Posterior Probability of H



$$P(h_i \,|\mathbf{d}) = \alpha P(\mathbf{d}|\, h_i)\, P(h_i)$$

- 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

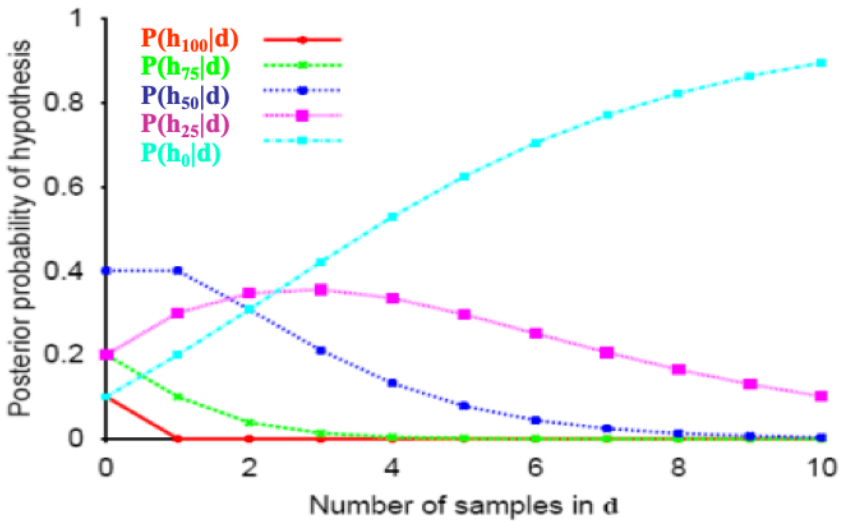
MAP approximation

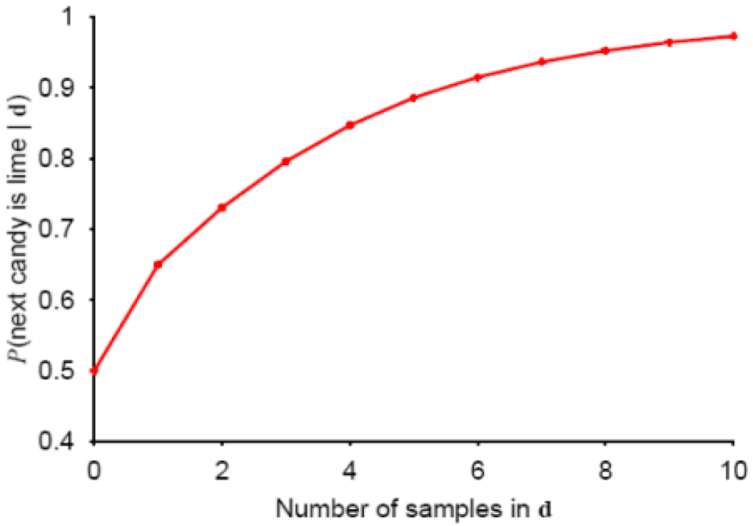
- Full Bayesian learning seems like a very safe bet, but unfortunately it does not work well in practice
  - Summing over the hypothesis space is often intractable (e.g., 18,446,744,073,709,551,616 Boolean functions of 6 attributes)
- Very common approximation: Maximum a posterior (MAP) learning:
  -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

Man findet $\propto$ oder $\sim$ oder $= \alpha \cdot \ldots$

MAP approximation

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

Howard, R.: Decision analysis: Perspectives on inference, decision, and experimentation. Proceedings of the IEEE 58(5), 632-643, **1970**

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 $P(h_i)$

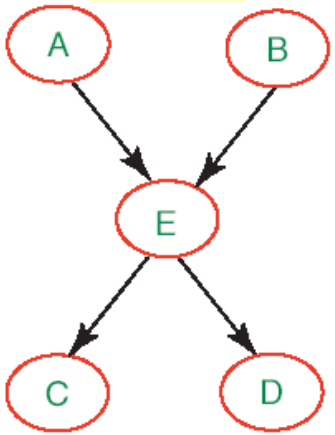
Overview

- 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

Model → Data → Probabilities



A	B	C	D	E
t	f	t	t	f
f	t	t	t	t
t	t	f	t	f
		...		

$P(A)$
$P(B)$
$P(E|A, B)$
$P(C|E)$
$P(D|E)$

Maximum-Likelihood-Parameterschätzung

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

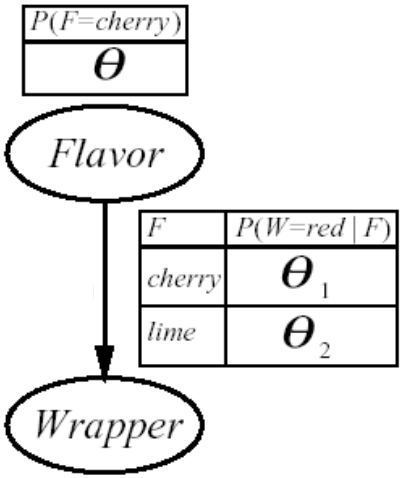
$$P(D \mid \theta) = \prod_m P(x[m] \mid \theta)$$

- Maximum Likelihood Estimation (MLE) Prinzip: Wähle θ^* so, dass $P(D \mid \theta^*)$ 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
  - $\theta\,\theta_1\,\theta_2$

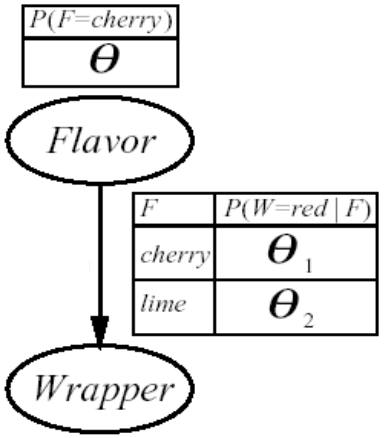


Anwendungsbeispiel Bonbonfabrik

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

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

  $= \theta (1-\theta_1)$



- 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
  - $L = c\log\theta + \ell \log(1-\theta) + r^c \log\theta_1 + g^c \log(1-\theta_1) + r^l \log\theta_2 + g^\ell \log(1-\theta_2)$
- Bestimmung der Ableitungen bzgl. $\theta, \theta_1, \theta_2$
  - Ausdrücke ohne Term, nach dem abgeleitet wird, verschwinden

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

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

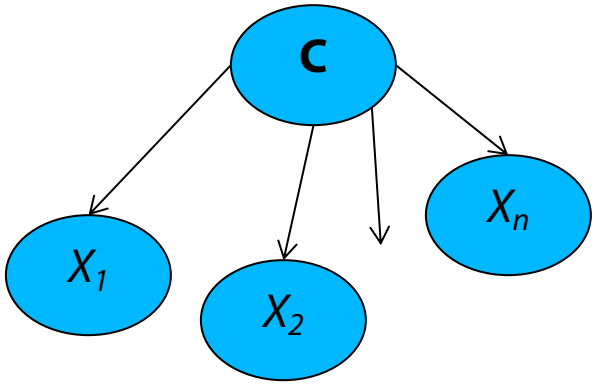
$$\frac{\partial L}{\partial \theta_2} = \frac{r_\ell}{\theta_2} - \frac{g_\ell}{1-\theta_2} = 0 \quad \Rightarrow \quad \theta_2 = \frac{r_\ell}{r_\ell+g_\ell}$$

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

$$\boldsymbol{P}(C|x_1,x_2,..,x_n) = \frac{\boldsymbol{P}(C,x_1,x_2,..,x_n)}{\boldsymbol{P}(x_1,x_2,..,x_n)} = \alpha \boldsymbol{P}(C)\prod_i \boldsymbol{P}(x_n \mid C)$$

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

Extension to MAP

If we replace the likelihood in the MLE formula above with the posterior, we get:

$$\theta_{MAP} = \arg\max_{\theta} P(X|\theta)P(\theta)$$

$$= \arg\max_{\theta} \log P(X|\theta) + \log P(\theta)$$

$$= \arg\max_{\theta} \log \prod_{i} P(x_i|\theta) + \log P(\theta)$$

$$= \arg\max_{\theta} \sum_{i} \log P(x_i|\theta) + \log P(\theta)$$

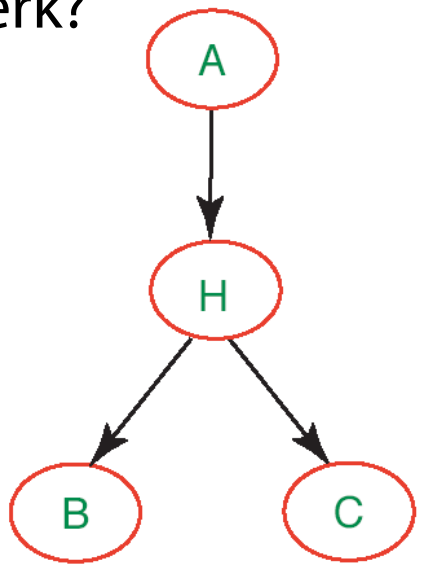
Comparing both MLE and MAP equation, the only thing differs is the inclusion of prior $P(\theta)$ in MAP, otherwise they are identical. What it means is that, the likelihood is now weighted with some weight coming from the prior.

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?

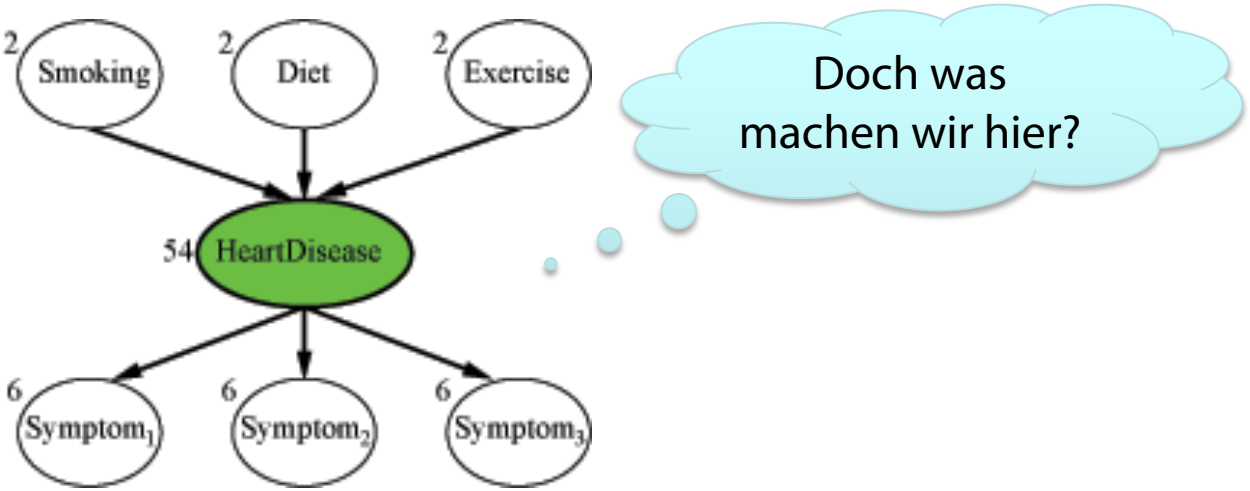


A	B	C
t	f	t
f	t	t
t	t	f
	...	

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

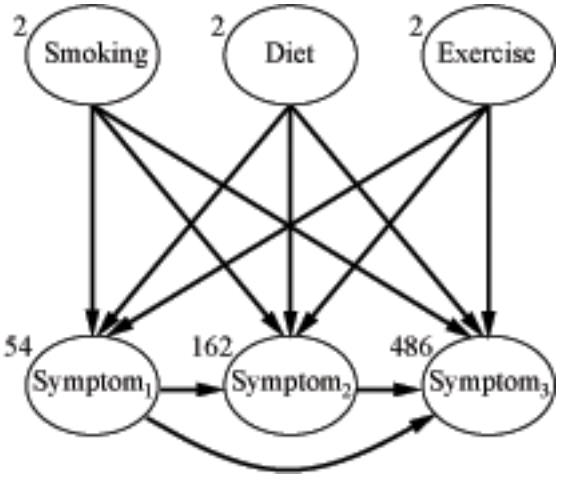
Vermeintliche Lösung

- 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

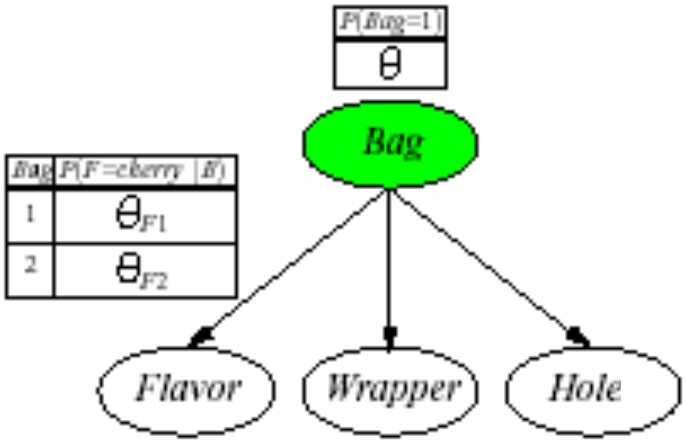
HeartDisease wegzulassen ist gar keine gute Lösung!



- 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 Bonbons 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



$\theta = P(Bag = 1)$

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

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

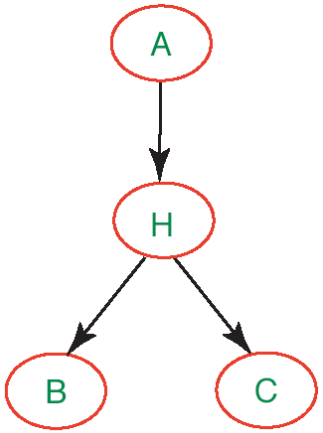
$\theta_{Hj} = P(Hole = yes | Bag = j)$

$j = 1,2$

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



A	B	C
t	f	t
f	t	t
t	t	f
	...	

- 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

CPT: Conditional Probability Table

EM: Wie funktioniert das mit Naive Bayes-Modelen

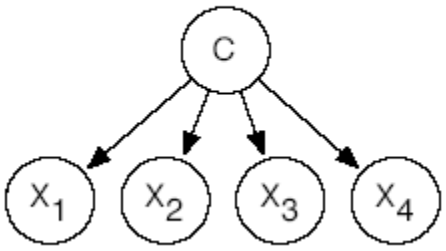
- Betrachtet die folgenden Daten, …
  - N Beispiele mit Booleschen Attributen X_1, X_2, X_3, X_4

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

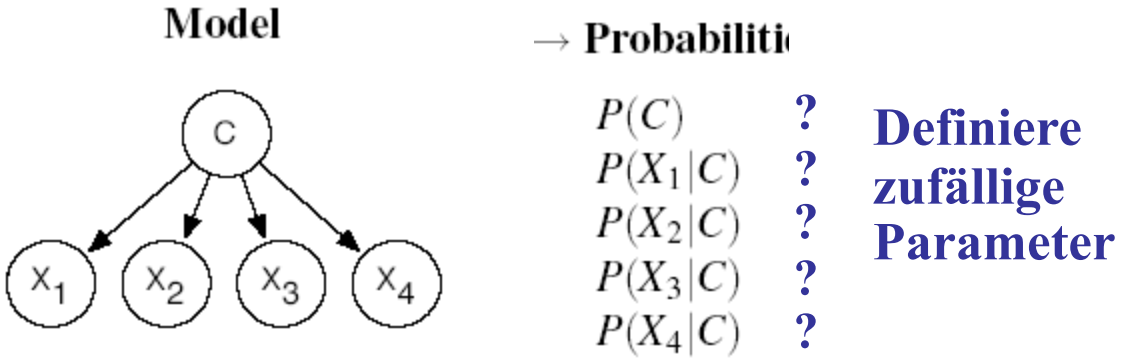
Model → **Probabiliti**



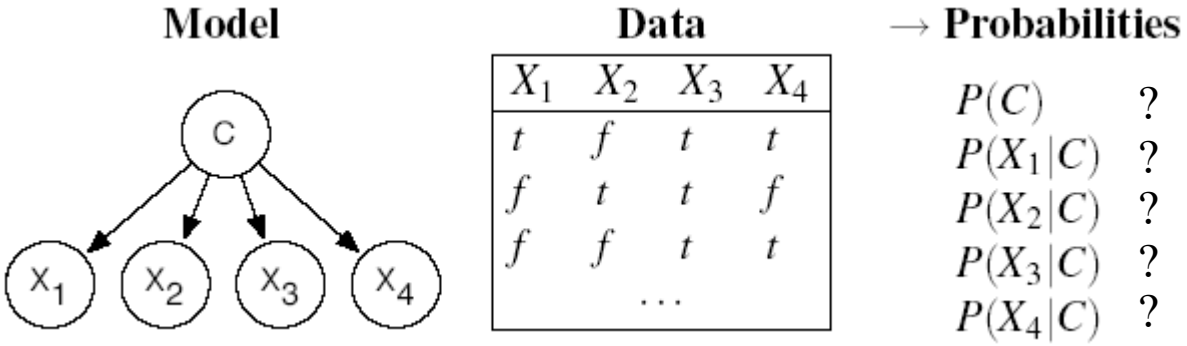
$$P(C) \quad ?$$
$$P(X_1|C) \quad ?$$
$$P(X_2|C) \quad ?$$
$$P(X_3|C) \quad ?$$
$$P(X_4|C) \quad ?$$

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) = Anzahl(Datenpunkte mit C=i) / Anzahl (alle Datenpunkte) i=1,2,3
  - Für $P(X_h|C)$ = Anzahl(Daten mit $X_h = val_k$ und C=i) / Anzahl(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 = Anzahl(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

$$\hat{N}(C = i) = \sum_{j=1}^{N} P(C = i \mid \text{Attribute von Beispiel } e_j)$$

$$= \sum_{j=1}^{N} P(C = i \mid x1_j, x2_j, x3_j, x4_j)$$

EM: Expectation-Schritt (Bestimmung der erwarteter Zahlen)

- Wie erhalten wir die notwendigen Werte aus dem Modell?

$$\hat{N}(C=i) = \sum_{j=1}^{N} P(C=i \mid \text{ Attribute von Beispiel } e_j)$$

$$= \sum_{j=1}^{N} P(C=i \mid x1_j, x2_j, x3_j, x4_j)$$

- Leicht mit naivem Bayesschen Netzwerk

$$P(C=i \mid x1_j, x2_j, x3_j, x4_j) = \frac{P(C=i, x1_j, x2_j, x3_j, x4_j)}{P(x1_j, x2_j, x3_j, x4_j)}$$

$$= \frac{P(x1_j \mid C=i)..,P(x4_j \mid C=i)P(C=i)}{P(x1_j, x2_j, x3_j, x4_j)}$$

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 $\mathbf{P}(X_h \mid C)$:

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

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

- Zum Beispiel

$$\hat{N}(X_1 = t, C = 1) = \sum_{e_j \text{ with } X_1 = t} P(C = i \mid x1_j = t, x2_j, x3_j, x4_j)$$

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

CPT: Conditional Probability Table

Maximization-Schritt: (Verfeinerung der Parameter)

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

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

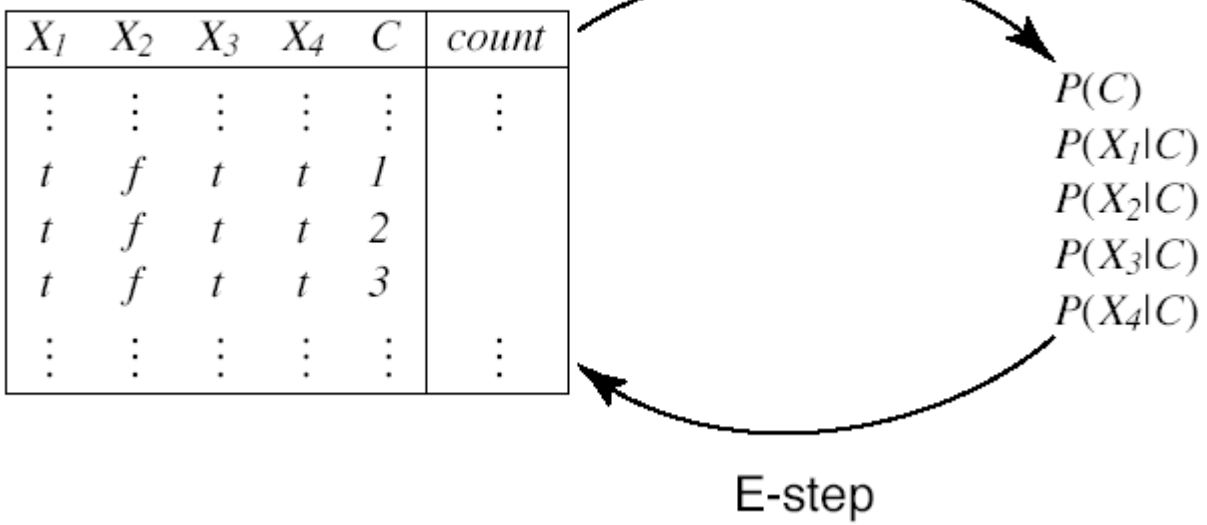
$$P(X_j = val_k \mid C = i) = \frac{\hat{N}(X_j = val_k C = i)}{\hat{N}(C = i)}$$

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

EM-Zyklus

- Nun kann der E-Schritt wiederholt werden

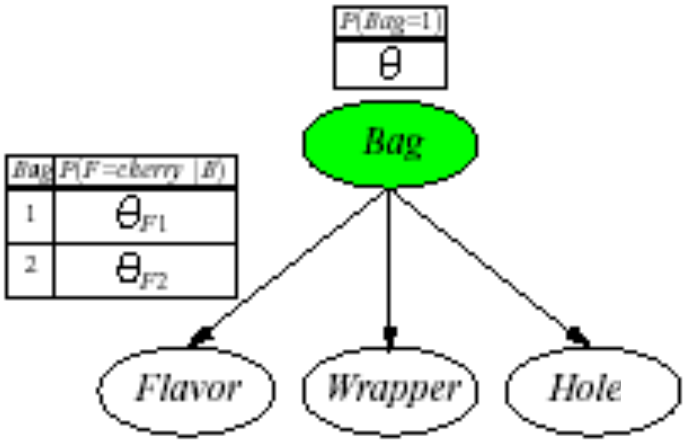
Erwartete Zähler
("Augmented data")



Wahrscheinlichkeiten

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



$\theta = P(Bag = 1)$

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

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

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

$j = 1,2$

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 Bonbons 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

$$\hat{N}(\text{Bag} = 1) = \sum_{j=1}^{N} P(Bag = 1 \mid flavor_j, wrapper_j, hole_j) =$$

$$= \sum_{j=1}^{N} \frac{P(flavor_j, wrapper_j, hole_j | Bag = 1) P(Bag = 1)}{P(flavor_j, wrapper_j, hole_j)}$$

$$= \sum_{j=1}^{N} \frac{P(flavor_j | Bag = 1) P(wrapper_j | Bag = 1) P(hole_j | Bag = 1) P(Bag = 1)}{\sum_i P(flavor_j | Bag = i) P(wrapper_j | Bag = i) P(hole_j | Bag = i) P(Bag = i)}$$

E-step

$$\sum_{j=1}^{N} \frac{P(flavor_j|Bag=1)P(wrapper_j|Bag=1)P(hole_j|Bag=1)P(Bag=1)}{\sum_i P(flavor_j|Bag=i)P(wrapper_j|Bag=i)P(hole_j|Bag=i)P(Bag=i)}$$

- 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

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

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

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

Noch ein Parameter…

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

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

  - 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

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

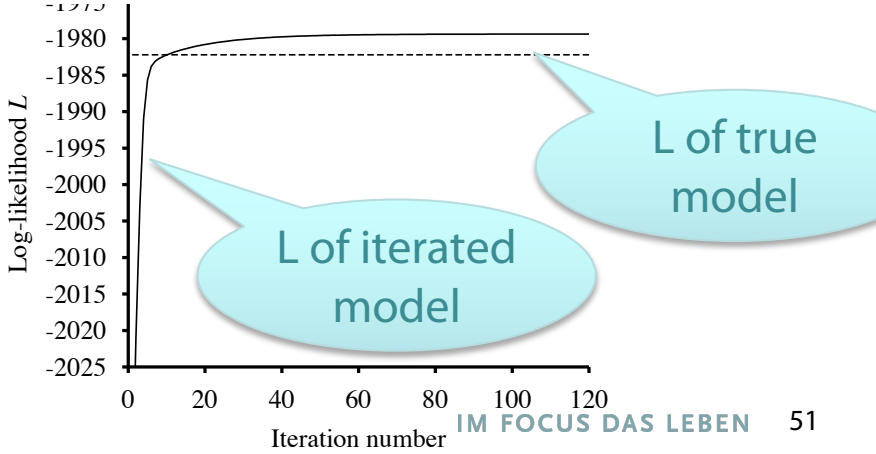
$$\theta_{F1}^{(1)} = 0.6684; \quad \theta_{W1}^{(1)} = 0.6483; \quad \theta_{H1}^{(1)} = 0.658;$$

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

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

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

- Wir können zeigen, das die Log-Likelihood $L = \log P(d, h_\theta)$ in jeder Iteration ansteigt, so dass die initiale Likelihood schon nach drei Iterationen überflügelt wird



EM: Diskussion

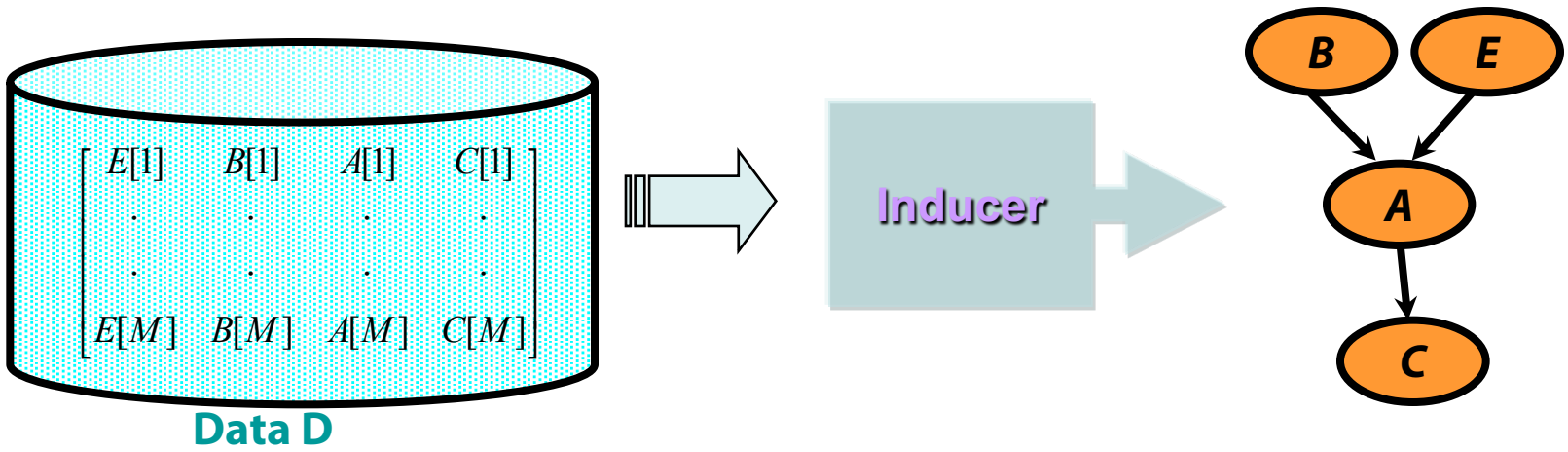
- Für komplexere BNs ist der Algorithmus der gleiche
  - Im allgemeinen müssen wir die CPT-Einträge für jede Variable X_i gegeben die Elternknotenwerte Pa_i berechnen
  - $\theta_{ijk} = P(X_i = x_{ij} | Pa_{i=} pa_{ik})$

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

- 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 (ggf. Approximationstechniken angewendet)

Learning Bayesian network structures

- Given training set $\boldsymbol{D} = \{\boldsymbol{x}[1],\ldots,\boldsymbol{x}[\boldsymbol{M}]\}$
- 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

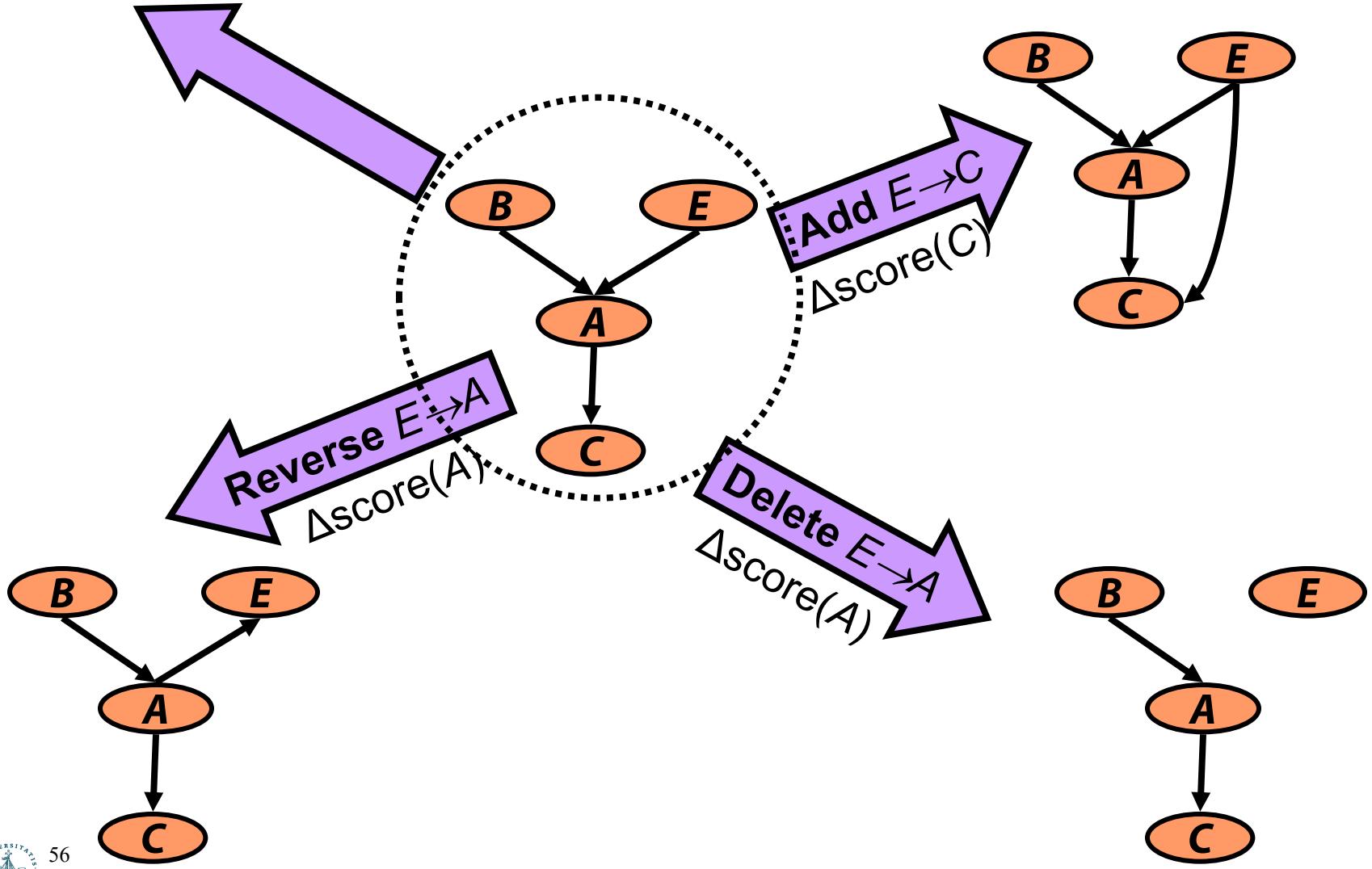
Can we learn G's params from D? | Does G explain D with ML? | Prior w.r.t. MDL

- $\text{Score}_D(G) = \log P(G|D) = \alpha \log [P(D|G)\, P(G)]$
- 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)

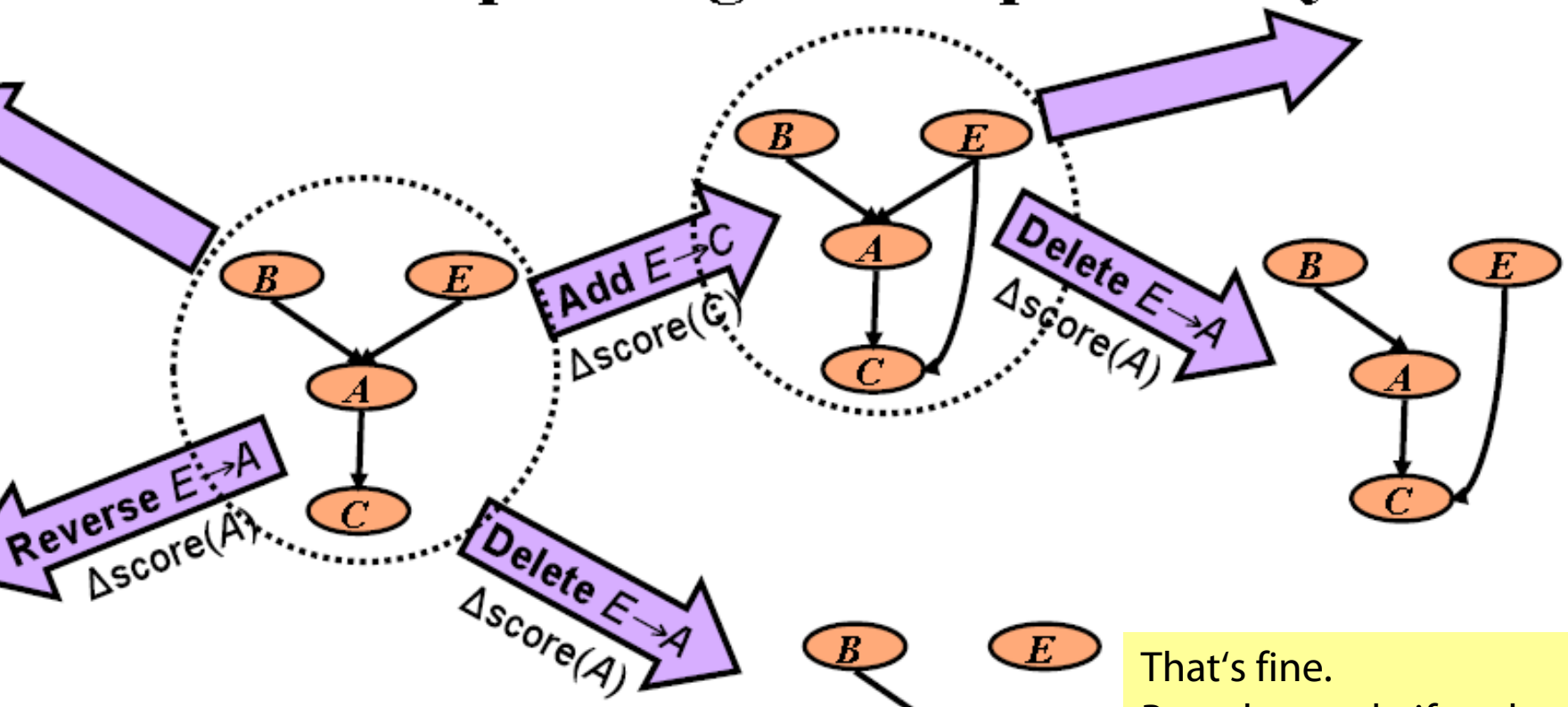
Same key property: Decomposability

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

Heuristic search



Exploiting decomposability



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

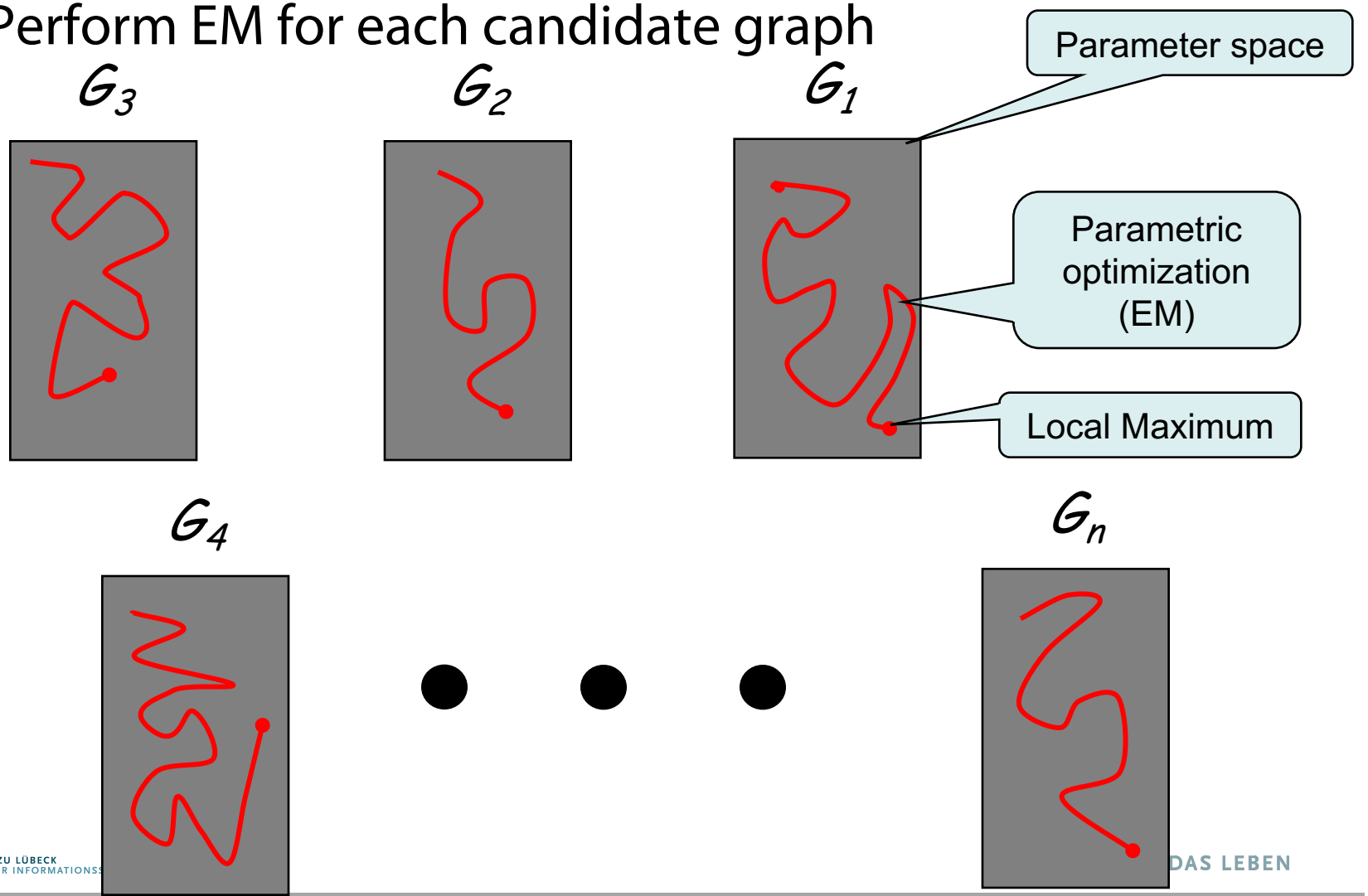
That's fine.
But what to do if we have non-complete data??

Cannot use decomposability

=> Rerun parameter estimation

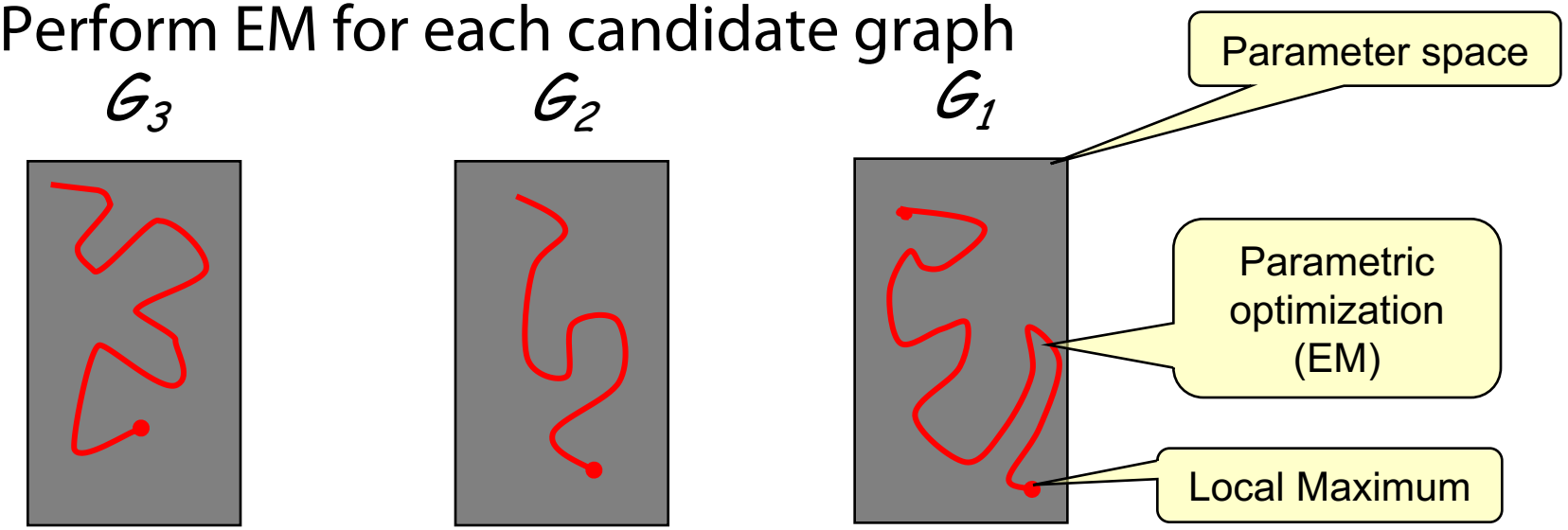
Local Search in Practice

- 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

$\Rightarrow$ 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

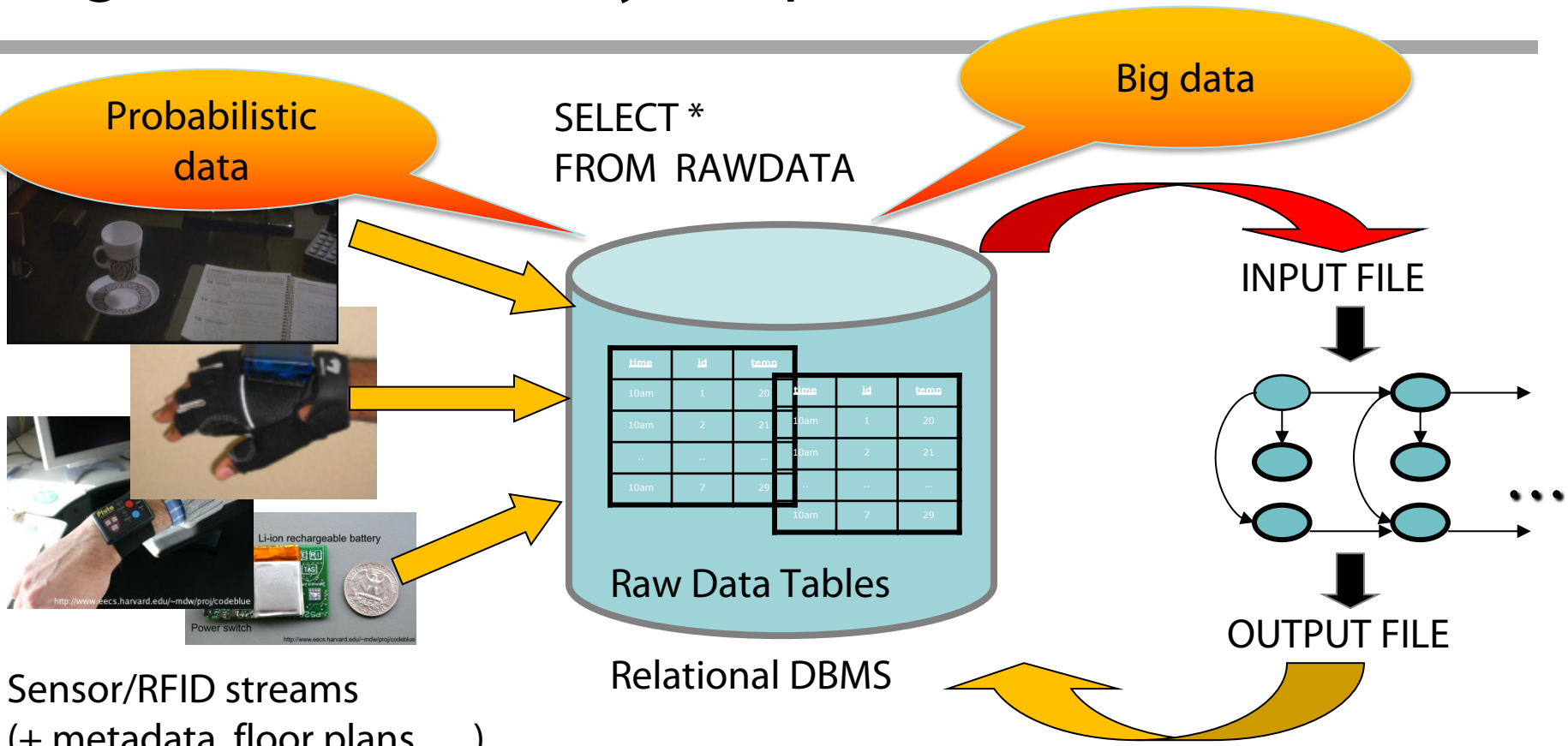
  or
  - Better scoring structure: "structural" EM step

Nir Friedman. The Bayesian structural EM algorithm. In Proceedings of the 14th conference on Uncertainty in artificial intelligence (UAI'98), 129-138. **1998**

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



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

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

Daisy Zhe Wang, Eirinaios Michelakis, Minos Garofalakis, and Joseph M. Hellerstein. "**BAYESSTORE**: Managing Large, Uncertain Data Repositories with Probabilistic Graphical Models", Proceedings of VLDB'2008 (PVLDB, Vol. 1), **2008**.