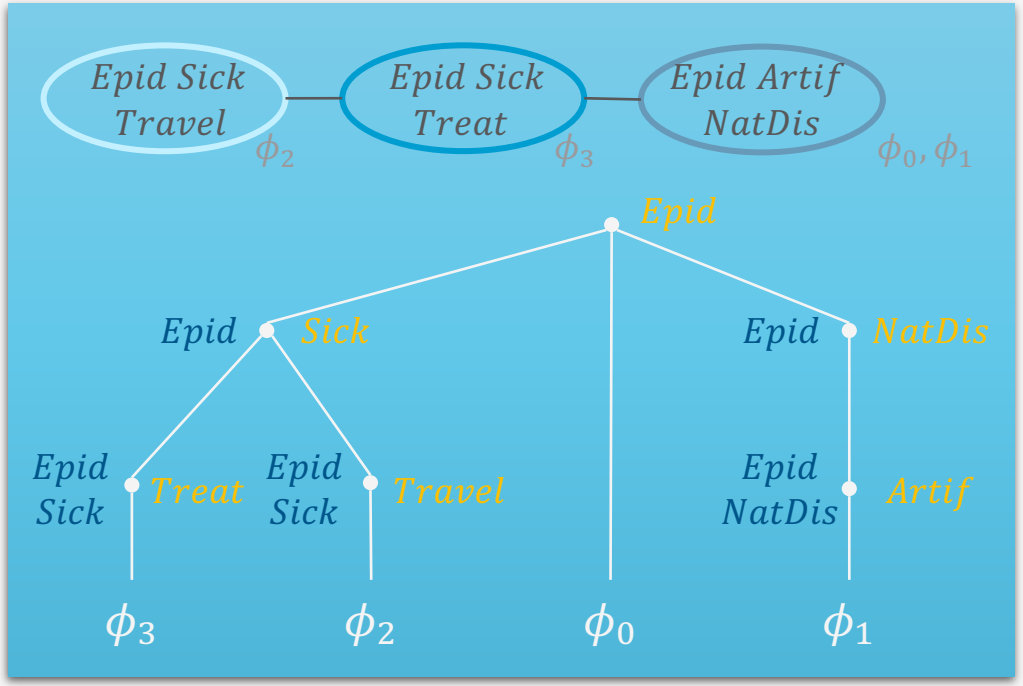
WWU
MÜNSTER

Exakte Inferenz in Episodischen PGMs

(c) Inferenzproblem Zustandsanfragen

Einführung in die
Künstliche Intelligenz



wissen.leben

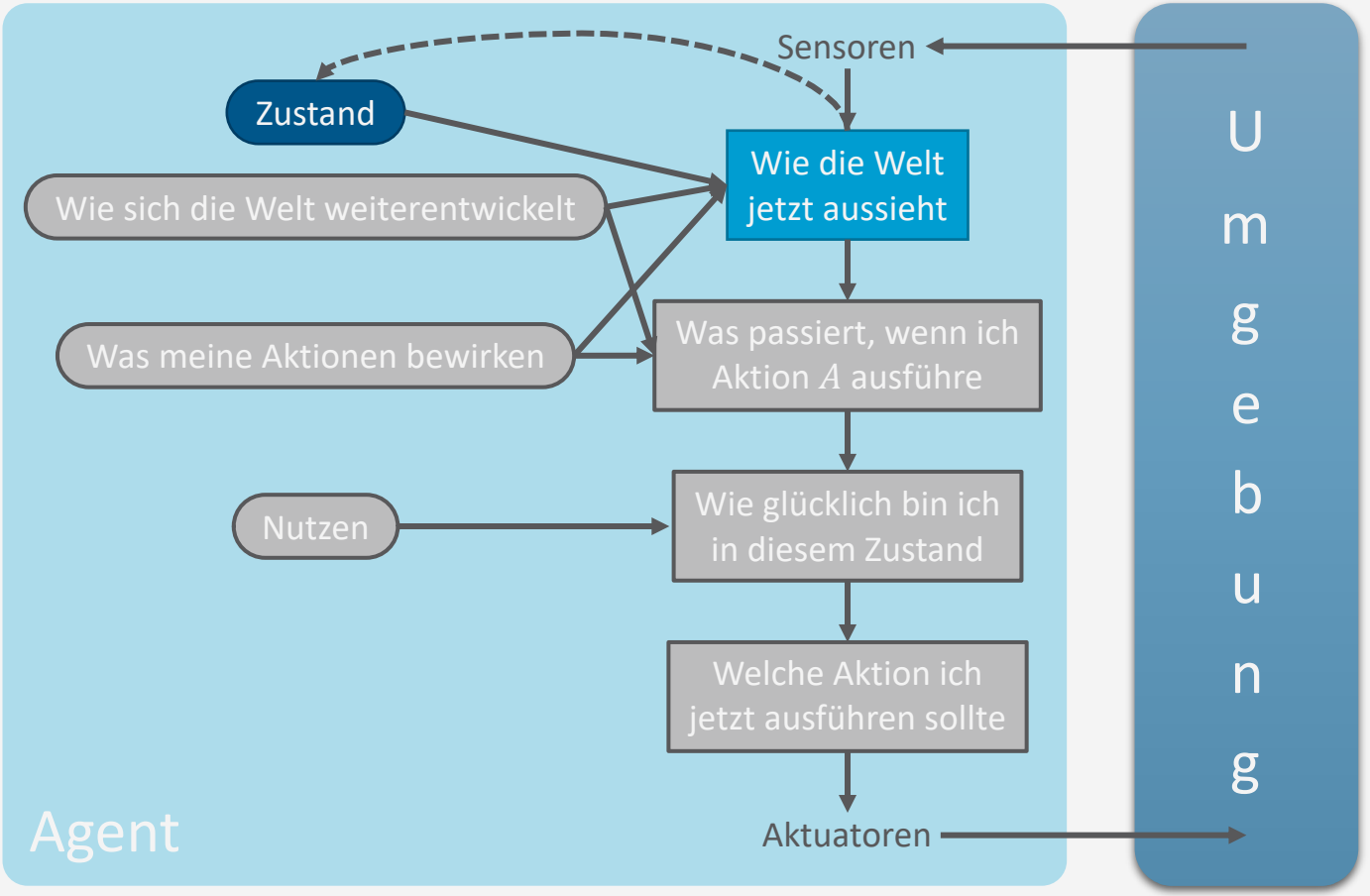
Tanya Braun
Arbeitsgruppe Data Science, Institut für Informatik

Inhalte

1. **Künstliche Intelligenz & Agenten**
   - Agentenabstraktion, Rationalität
   - Aufgabenumgebung
2. **Episodische PGMs**
   - Gerichtetes Modell: Bayes Netze (BNs)
   - Ungerichtete Modelle
3. **Exakte Inferenz in episodischen PGMs**
   - Wahrscheinlichkeits- und Zustandsanfragen
   - Direkt auf den Modellen, mittels Hilfsstrukturen
4. **Approximative Inferenz in episodischen PGMs**
   - Wahrscheinlichkeitsanfragen
   - Deterministische, stochastische Algorithmen
5. **Lernalgorithmen für episodische PGMs**
   - Bei (nicht) vollständigen Daten, (un)bekannter Struktur
6. **Sequentielle PGMs und Inferenz**
   - Dynamische BNs, Hidden-Markov-Modelle
   - filtering / prediction / hindsight Anfragen, wahrscheinlichste Zustandssequenz
   - Exakter, approximativer Algorithmus
7. **Entscheidungstheoretische PGMs**
   - Präferenzen, Nutzenprinzip
   - PGMs mit Entscheidungs- und Nutzenknoten
   - Berechnung der besten Aktion (Aktionssequenz)
8. **Abschlussbetrachtungen**

Einordnung der Vorlesung: *Modell- und nutzenbasierter Agent*

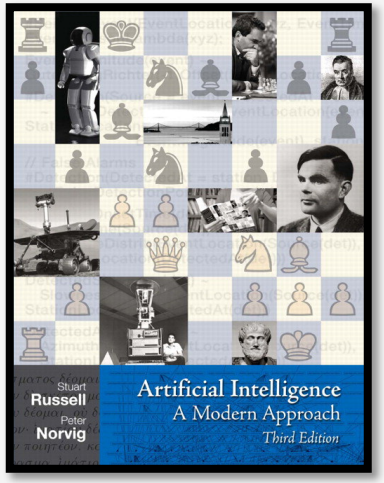
- Nachfolgende Themen der Vorlesung
  2. Episodische PGMs
  3. Exakte Inferenz in episodischen PGMs
  4. Approximative Inferenz in episodischen PGMs
  5. Lernalgorithmen für episodische PGMs
  6. Sequentielle PGMs und Inferenz
  7. Entscheidungstheoretische PGMs

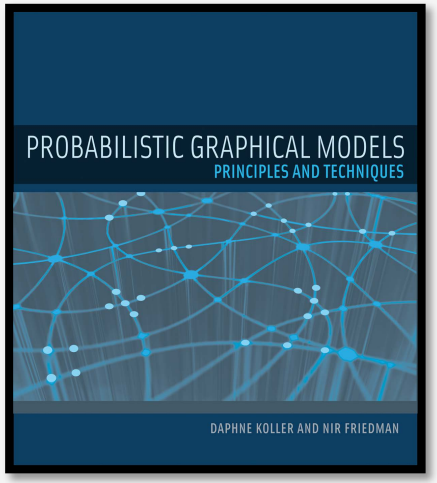


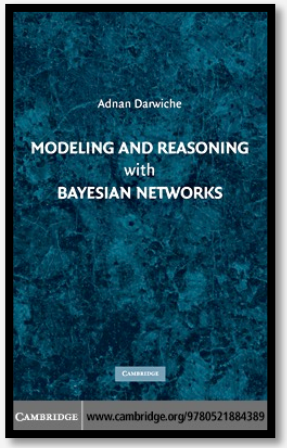
Literaturhinweise

Inhalte dieses Themenblocks werden in den folgenden Kapiteln der Vorlesungsbücher behandelt

- AIMA(de)
  - Kap. 14.4: Exakte Inferenz in Bayes Netzen
- PGM
  - Kap. 9: Variableneliminierung
  - Kap. 10: Exakte Inferenz: Cliquen-Bäume
  - Kap. 13: MAP Inferenz







- Wer gerne ein anderes Buch ausprobieren möchte (Fokus auf BNs):
  - Adnan Darwiche, *Modelling and Reasoning with Bayesian Networks*, 2009.

Überblick: 3. Exakte Inferenz in episodischen PGMs

A. *Einzelanfragen: Variableneliminierung (VE)*
   - Algorithmus, Operatoren für Wahrscheinlichkeitsanfragen
   - Dekompositionsbäume, Komplexität

B. *Multi-Anfragen: Junction Tree (Cliquen-Bäume) Algorithmus (JT)*
   - Cliquen, Junction Tree als Hilfsstruktur, Vorverarbeitung und Anfragebeantwortung
   - Zusammenhang mit VE, Komplexität
   - *Geschichtsstunde: Pearl's Probability Propagation (PP)* auf Polytree BNs

C. *Inferenzproblem Zustandsanfragen*
   - Ausprägungen: Most probable explanation (MPE) / maximum a posteriori Anfragen (MAP)
   - Semantik: Ausmaximieren statt aussummieren; max-out Operator in VE
   - Auswirkungen auf die Komplexität

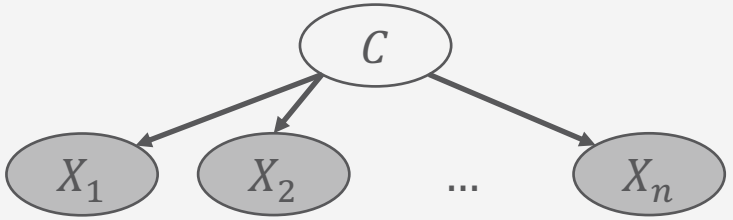
Most Probable Explanation (MPE)

Zustandsanfragen*

* Im Kontrast zu *Wahrscheinlichkeitsanfragen* (Anfragen zu marginalen und bedingten Wahrscheinlichkeitsverteilungen)

Anwendungen

- Naive Bayes Klassifizierer: $P(C \mid \boldsymbol{x})$
  - Nichts auszusummieren: $\boldsymbol{S} = \{C\}, \boldsymbol{T} = \mathrm{rv}(\boldsymbol{x})$ und damit
    $$\boldsymbol{U} = \boldsymbol{R} \setminus \boldsymbol{S} \setminus \boldsymbol{T} = \emptyset$$



  - Eigentlich will man $\arg\max_{c \in \mathrm{Val}(C)} P(C \mid \boldsymbol{x})$ haben
    - Zustandsanfrage gegeben Evidenz:
      „Was ist der wahrscheinlichste Zustand der Zufallsvariablen, der die Evidenz hervorgerufen haben könnte?“
    - Variablen ausmaximieren anstatt aussummieren

Problemformulierung: Most Probable Explanation (MPE)

- Gegeben Evidenz, was ist der wahrscheinlichste Zustand der verbleibenden Zufallsvariablen?
  - *Zustandsanfragen* an den wahrscheinlichsten Zustand aller Zufallsvariablen ohne Evidenz
    - Wahrscheinlichste Erklärung (*most probable explanation*, MPE)
  - Formal betrachtet, gegeben ein Modell F mit der vollständigen gemeinsamen Wahrscheinlichkeitsverteilung P_F und Evidenz $\boldsymbol{e}$:

$$MPE_F(\boldsymbol{e}) = \arg\max_{\boldsymbol{v} \in \mathrm{Val}(\boldsymbol{V})} P(\boldsymbol{v} \mid \boldsymbol{e}) = \arg\max_{\boldsymbol{v} \in \mathrm{Val}(\boldsymbol{V})} P(\boldsymbol{v}, \boldsymbol{e})$$

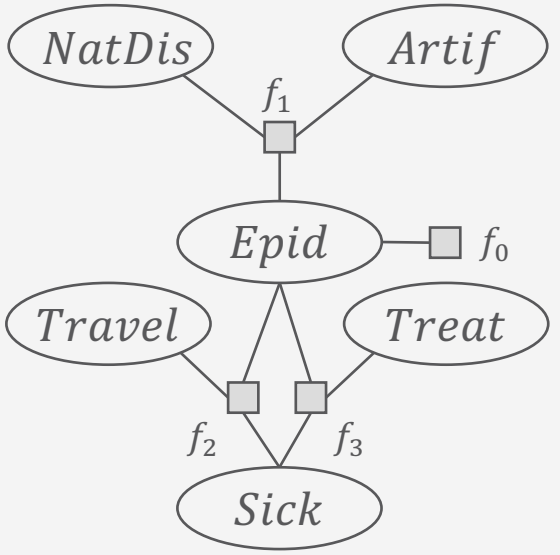
    - $\boldsymbol{V} = \mathrm{rv}(F) \setminus \mathrm{rv}(\boldsymbol{e})$ die Zufallsvariablen in F ohne Evidenz
    - Im Vergleich zu „Wahrscheinlichkeitsanfragen“, wird $\sum$ mit argmax als Eliminierungsoperation ersetzt
    - $MPE_F(\boldsymbol{e}) = \arg\max_{\boldsymbol{v} \in \mathrm{Val}(\boldsymbol{V})} P(\boldsymbol{v} \mid \boldsymbol{e}) = \arg\max_{\boldsymbol{v} \in \mathrm{Val}(\boldsymbol{V})} \frac{P(\boldsymbol{v}, \boldsymbol{e})}{P(\boldsymbol{e})} = \frac{1}{P(\boldsymbol{e})} \arg\max_{\boldsymbol{v} \in \mathrm{Val}(\boldsymbol{V})} P(\boldsymbol{v}, \boldsymbol{e}) = \arg\max_{\boldsymbol{v} \in \mathrm{Val}(\boldsymbol{V})} P(\boldsymbol{v}, \boldsymbol{e})$
      - Maximierung invariant bezüglich positiver linearer Transformation ($\frac{1}{P(\boldsymbol{e})}$ lineare Transformation)

Alexander Philip Dawid. Applications of a General Propagation Algorithm for Probabilistic Expert Systems. *Statistics and Computing*, 2(1):25–36, 1992.
Rina Dechter. Bucket Elimination: A Unifying Framework for Probabilistic Inference. In *Learning and Inference in Graphical Models*, pages 75–104. MIT Press, 1999.

MPE: Semantik

- Gegeben die Semantik könnte man die vollständige gemeinsame Verteilung bauen, Evidenz absorbieren lassen und dann die Welt mit der höchsten Wahrscheinlichkeit wählen
  - Beispiel: Gegeben $Sick = true$, was ist der wahrscheinlichste Zustand von $Epid, Travel, Treat, NatDis, Artif$?

$$MPE(sick) = \arg\max_{e,tr,tt,n,a} P(Epid = e, Travel = tl, Treat = tt, NatDis = n, Artif = a \mid sick)$$



$$MPE(sick) = \arg\max_{e,tr,tt,n,a} P(Epid = e, Travel = tl, Treat = tt, NatDis = n, Artif = a \mid sick)$$

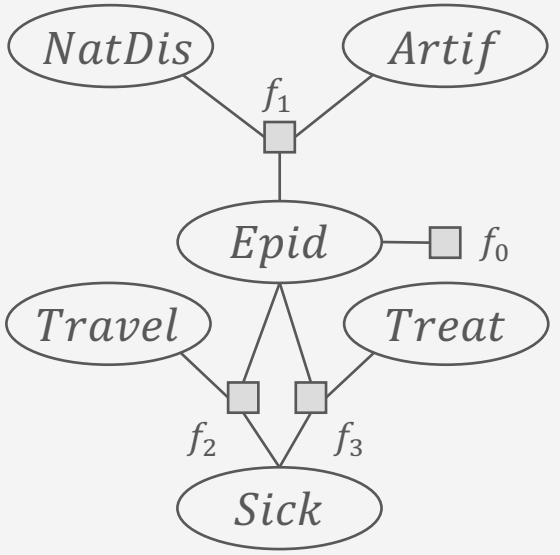
$Epid$	ϕ_0
$false$	50
$true$	1

$Travel$	$Epid$	$Sick$	ϕ_2
$false$	$false$	$false$	20
$false$	$false$	$true$	24
$false$	$true$	$false$	5
$false$	$true$	$true$	6
$true$	$false$	$false$	28
$true$	$false$	$true$	8
$true$	$true$	$false$	7
$true$	$true$	$true$	2

$Epid$	$Sick$	$Treat$	ϕ_3
$false$	$false$	$false$	5
$false$	$false$	$true$	1
$false$	$true$	$false$	3
$false$	$true$	$true$	2
$true$	$false$	$false$	5
$true$	$false$	$true$	4
$true$	$true$	$false$	1
$true$	$true$	$true$	7

$Epid$	$NatDis$	$Artif$	ϕ_1
$false$	$false$	$false$	12
$false$	$false$	$true$	2
$false$	$true$	$false$	3
$false$	$true$	$true$	1
$true$	$false$	$false$	7
$true$	$false$	$true$	4
$true$	$true$	$false$	5
$true$	$true$	$true$	1

$Epid$	$Travel$	$Treat$	$Artif$	$NatDis$	$Sick$	P
~~$false$~~	~~$false$~~	~~$false$~~	~~$false$~~	~~$false$~~	~~$false$~~	~~...~~
$false$	$false$	$false$	$false$	$false$	$true$	0.298
~~$false$~~	~~$false$~~	~~$false$~~	~~$false$~~	~~$true$~~	~~$false$~~	~~...~~
$false$	$false$	$false$	$false$	$true$	$true$	0.050
⋮	⋮	⋮	⋮	⋮		⋮



$$MPE(sick) = \arg\max_{e,tr,tt,n,a} P(Epid = e, Travel = tl, Treat = tt, NatDis = n, Artif = a \mid sick)$$

$Epid$	$Travel$	$Treat$	$Artif$	$NatDis$	P
$false$	$false$	$false$	$false$	$false$	0.298
$false$	$false$	$false$	$false$	$true$	0.050
$false$	$false$	$false$	$true$	$false$	0.074
$false$	$false$	$false$	$true$	$true$	0.025
$false$	$false$	$true$	$false$	$false$	0.199
$false$	$false$	$true$	$false$	$true$	0.033
$false$	$false$	$true$	$true$	$false$	0.050
$false$	$false$	$true$	$true$	$true$	0.017
$false$	$true$	$false$	$false$	$false$	0.099
$false$	$true$	$false$	$false$	$true$	0.017
$false$	$true$	$false$	$true$	$false$	0.025
$false$	$true$	$false$	$true$	$true$	0.008
$false$	$true$	$true$	$false$	$false$	0.066
$false$	$true$	$true$	$false$	$true$	0.011
$false$	$true$	$true$	$true$	$false$	0.017
$false$	$true$	$true$	$true$	$true$	0.006
⋮	⋮	⋮	⋮	⋮	⋮

max Wert in P

⇓

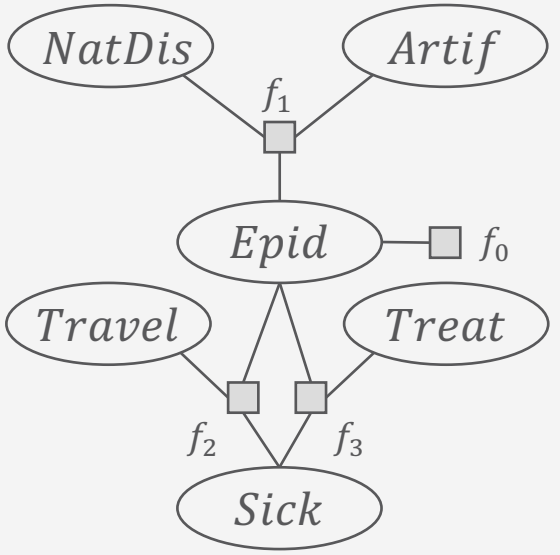
arg max Zuweisung zum max Wert

Obere Hälfte der verbliebenen Verteilung nach Absorption von $sick$
(Untere Hälfte noch sehr viel kleinere Werte)

Kombinatorische Explosion!

- Gleicher Aufwand wie bei Wahrscheinlichkeitsanfragen: $O(r^n)$
- Ignoriert Wissen über Faktorisierung

Können wir nicht für MPE das gleiche machen, wie wir es für Wahrscheinlichkeitsanfragen in VE gemacht haben?



Maximierung und Summierung

Verhalten von Maximierung zu Summierung ähnlich:

- Kommutativ
  - Summierung:

$$\sum_{s \in \mathrm{Val}(S)} \sum_{r \in \mathrm{Val}(R)} \phi(r,s,\boldsymbol{t}) = \sum_{r \in \mathrm{Val}(R)} \sum_{s \in \mathrm{Val}(S)} \phi(r,s,\boldsymbol{t})$$

  - Maximierung:

$$\max_{s \in \mathrm{Val}(S)} \max_{r \in \mathrm{Val}(R)} \phi(r,s,\boldsymbol{t}) = \max_{r \in \mathrm{Val}(R)} \max_{s \in \mathrm{Val}(S)} \phi(r,s,\boldsymbol{t})$$

  - Beispiel: $S = NatDis, R = Artif, \boldsymbol{t} = \{Epid\}$

Epid	*NatDis*	*Artif*	ϕ_1	
false	*false*	*false*	12	+ 14
false	*false*	*true*	2	+ 15 → + 18
false	*true*	*false*	3	+ 4
false	*true*	*true*	1	+ 3
true	*false*	*false*	7	+ 11
true	*false*	*true*	4	+ 12 → + 17
true	*true*	*false*	5	+ 6
true	*true*	*true*	1	+ 5

Epid	*NatDis*	*Artif*	ϕ_1	
false	*false*	*false*	12	m 12
false	*false*	*true*	2	m 12 → m 12
false	*true*	*false*	3	m 3
false	*true*	*true*	1	m 2
true	*false*	*false*	7	m 7
true	*false*	*true*	4	m 7 → m 7
true	*true*	*false*	5	m 5
true	*true*	*true*	1	m 4

Maximierung und Summierung

Verhalten von Maximierung zu Summierung ähnlich:

- „Ausklammern“ möglich ($\boldsymbol{r} \cap \boldsymbol{s} = \emptyset$)
  - Summierung:

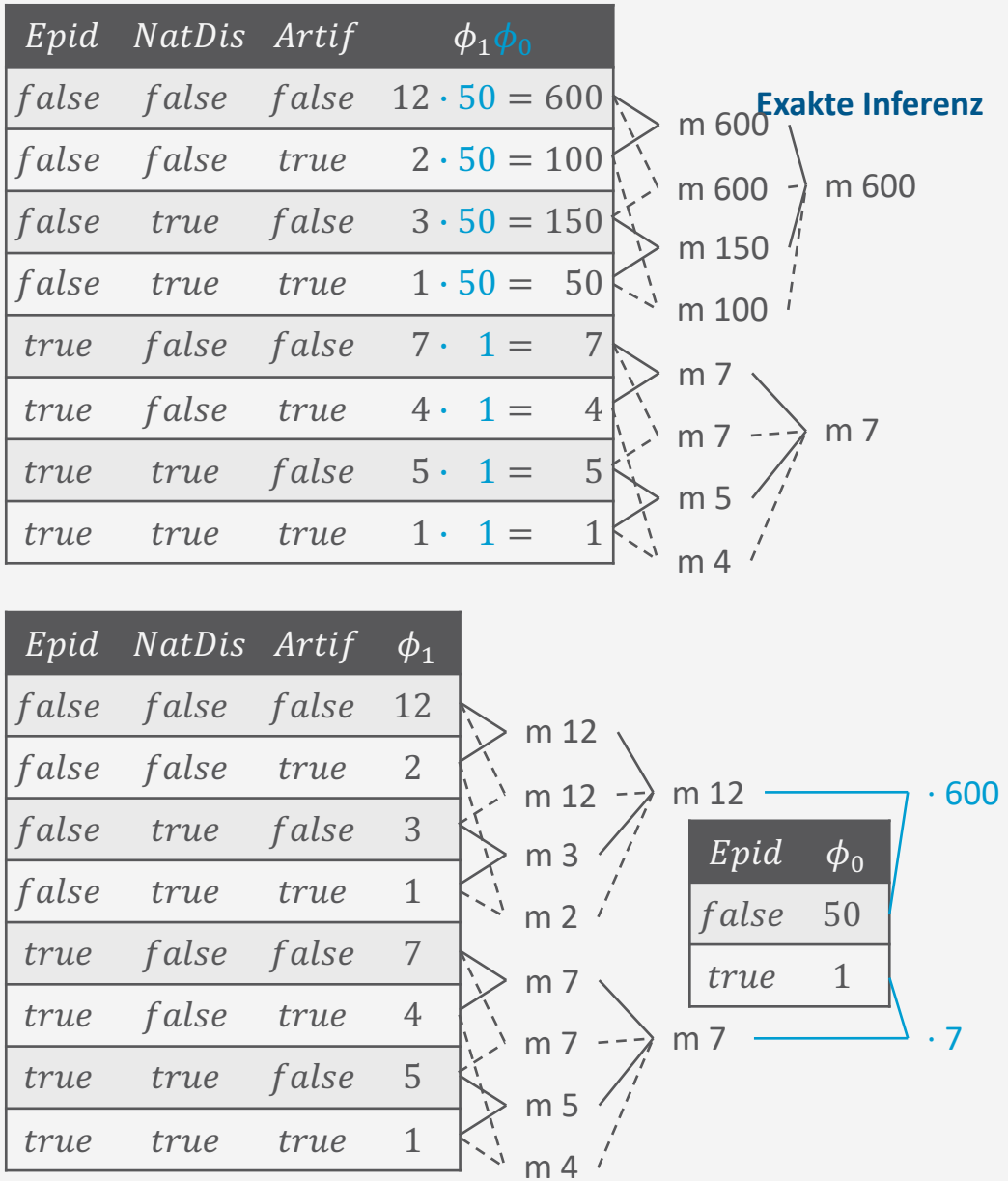
$$\sum_{\boldsymbol{r} \in \mathrm{Val}(\boldsymbol{R})} \phi(\boldsymbol{r}, \boldsymbol{s}) \phi'(\boldsymbol{s}) = \phi'(\boldsymbol{s}) \sum_{\boldsymbol{r} \in \mathrm{Val}(\boldsymbol{R})} \phi(\boldsymbol{r}, \boldsymbol{s})$$

  - Maximierung:

$$\max_{\boldsymbol{r} \in \mathrm{Val}(\boldsymbol{R})} \phi(\boldsymbol{r}, \boldsymbol{s}) \phi'(\boldsymbol{s}) = \phi'(\boldsymbol{s}) \max_{\boldsymbol{r} \in \mathrm{Val}(\boldsymbol{R})} \phi(\boldsymbol{r}, \boldsymbol{s})$$

  - Beispiel: $\boldsymbol{r} = \{NatDis, Artif\}, \boldsymbol{s} = \{Epid\}$

Beides für effiziente Inferenz im Stile von VE nutzen



VE für MPE Anfragen: Beispiel

$$MPE(sick)$$

$$= \arg\max_{e \in \mathrm{Val}(E)} \arg\max_{n \in \mathrm{Val}(N)} \arg\max_{a \in \mathrm{Val}(A)} \arg\max_{tl \in \mathrm{Val}(Tl)} \arg\max_{tt \in \mathrm{Val}(Tt)} P_{\boldsymbol{R}}(E = e, N = n, A = a, sick, Tl = tl, Tt = tt)$$

$$= \arg\max_{e \in \mathrm{Val}(E)} \arg\max_{n \in \mathrm{Val}(N)} \arg\max_{a \in \mathrm{Val}(A)} \arg\max_{tl \in \mathrm{Val}(Tl)} \arg\max_{tt \in \mathrm{Val}(Tt)} \frac{1}{Z} \prod_{i=0}^{3} \phi_i(\boldsymbol{R}_i = \boldsymbol{r}_i)$$

Lineare Transformation → weglassen

$$= \arg\max_{e \in \mathrm{Val}(E)} \arg\max_{n \in \mathrm{Val}(N)} \arg\max_{a \in \mathrm{Val}(A)} \arg\max_{tl \in \mathrm{Val}(Tl)} \arg\max_{tt \in \mathrm{Val}(Tt)} \phi_0(e)\phi_1(e,n,a)\phi_2(tl,e,sick)\phi_3(e,sick,tt)$$

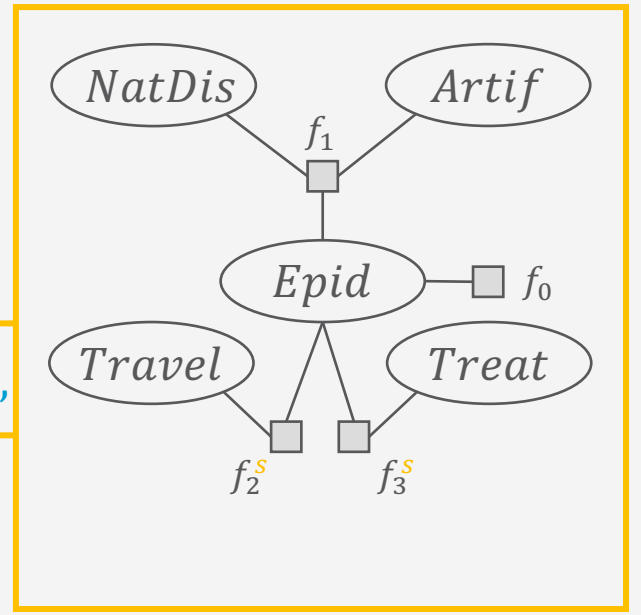
Absorption wie vorher

$$= \arg\max_{e \in \mathrm{Val}(E)} \arg\max_{n \in \mathrm{Val}(N)} \arg\max_{a \in \mathrm{Val}(A)} \arg\max_{tl \in \mathrm{Val}(Tl)} \arg\max_{tt \in \mathrm{Val}(Tt)} \phi_0(e)\phi_1(e,n,a)\phi_2^s(tl,e)\phi_3^s(e,tt)$$

$$= \arg\max_{e \in \mathrm{Val}(E)} \phi_0(e) \arg\max_{n \in \mathrm{Val}(N)} \arg\max_{a \in \mathrm{Val}(A)} \phi_1(e,n,a) \arg\max_{tl \in \mathrm{Val}(Tl)} \phi_2^s(tl,e) \arg\max_{tt \in \mathrm{Val}(Tt)} \phi_3^s(e,tt)$$

VE für MPE Anfragen: Beispiel

$$MPE(sick) = \arg\max_{e\in Val(E)} \arg\max_{n\in Val(N)} \arg\max_{a\in Val(A)} \arg\max_{tl\in Val(Tl)} \arg\max_{tt\in Val(Tt)} \phi_0(e)\phi_1(e,n,a)\phi_2(tl,e,sick)\phi_3(e,sick,$$



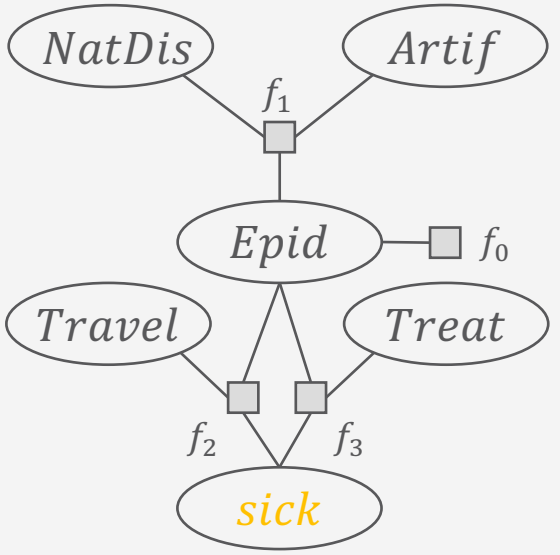
- Absorbieren von $sick$ in f_2, f_3 wie vorher

Epid	Sick	Treat	ϕ_3
~~false~~	~~false~~	~~false~~	~~5~~
~~false~~	~~false~~	~~true~~	~~1~~
false	true	false	3
false	true	true	2
~~true~~	~~false~~	~~false~~	~~5~~
~~true~~	~~false~~	~~true~~	~~4~~
true	true	false	1
true	true	true	7

Epid	Treat	ϕ_3^s
false	false	3
false	true	2
true	false	1
true	true	7

Travel	Epid	Sick	ϕ_2
~~false~~	~~false~~	~~false~~	~~20~~
false	false	true	24
~~false~~	~~true~~	~~false~~	~~5~~
false	true	true	6
~~true~~	~~false~~	~~false~~	~~28~~
true	false	true	8
~~true~~	~~true~~	~~false~~	~~7~~
true	true	true	2

Travel	Epid	ϕ_2^s
false	false	24
false	true	6
true	false	8
true	true	2



VE für MPE Anfragen: Beispiel

$$MPE(sick)$$

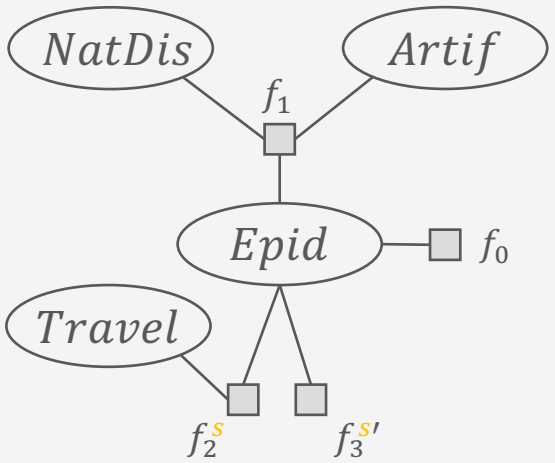
$$= \arg\max_{e \in \mathrm{Val}(E)} \phi_0(e) \arg\max_{n \in \mathrm{Val}(N)} \arg\max_{a \in \mathrm{Val}(A)} \phi_1(e,n,a) \arg\max_{tl \in \mathrm{Val}(Tl)} \phi_2^s(tl,e) \arg\max_{tt \in \mathrm{Val}(Tt)} \phi_3^s(e,tt)$$

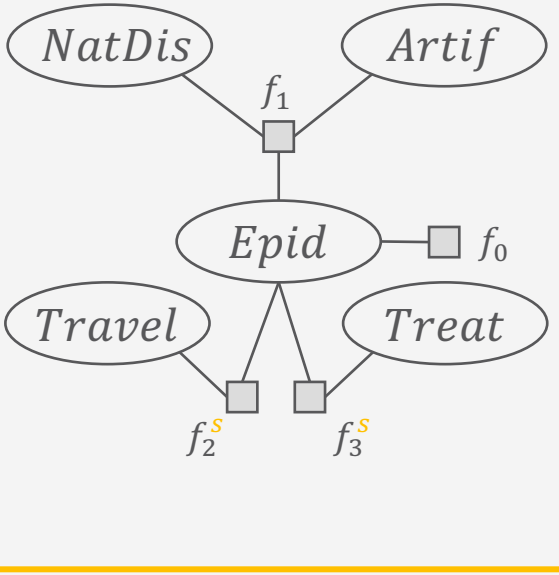
$$= \arg\max_{e \in \mathrm{Val}(E)} \phi_0(e)\phi_3^{s\prime}(e) \arg\max_{n \in \mathrm{Val}(N)} \arg\max_{a \in \mathrm{Val}(A)} \phi_1(e,n,a) \arg\max_{tl \in \mathrm{Val}(Tl)} \phi_2^s(tl,e)$$

- $Treat$ ausmaximieren:
  - Wähle die argmax Zuweisung von $Treat$ für jede mögliche Welt von $Epid$
    - $Epid = false$
      - Maximaler Wert: 3
      - Zuweisung merken: $Treat = false$
    - $Epid = true$
      - Maximaler Wert: 7
      - Zuweisung merken: $Treat = true$
  - Verbleibender Faktor: $f_3^{s\prime} = \phi_3^{s\prime}(Epid)$

$Epid$	$Treat$	ϕ_3^s
$false$	$false$	3
$false$	$true$	2
$true$	$false$	1
$true$	$true$	7

$Epid$	$\phi_3^{s\prime}$
$false$	3
$true$	7





VE für MPE Anfragen: Beispiel

$$MPE(sick)$$

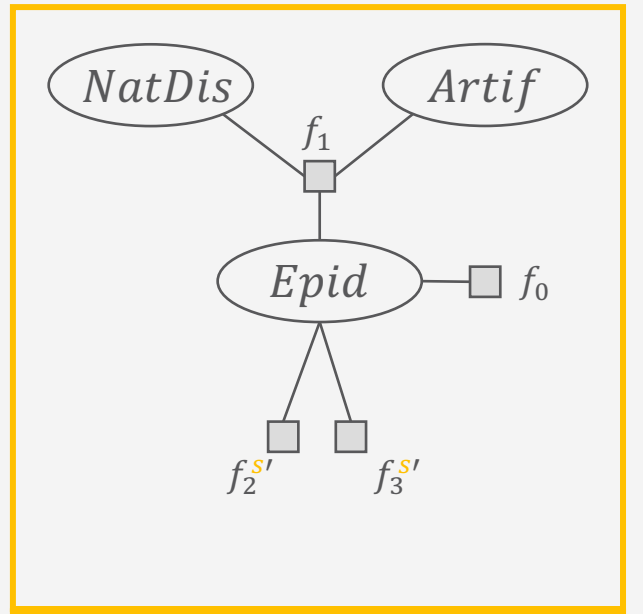
$$= \arg\max_{e \in \mathrm{Val}(E)} \phi_0(e)\phi_3^s(e) \arg\max_{n \in \mathrm{Val}(N)} \arg\max_{a \in \mathrm{Val}(A)} \phi_1(e,n,a) \boxed{\arg\max_{tl \in \mathrm{Val}(Tl)} \phi_2^{s\prime}(tl,e)}$$

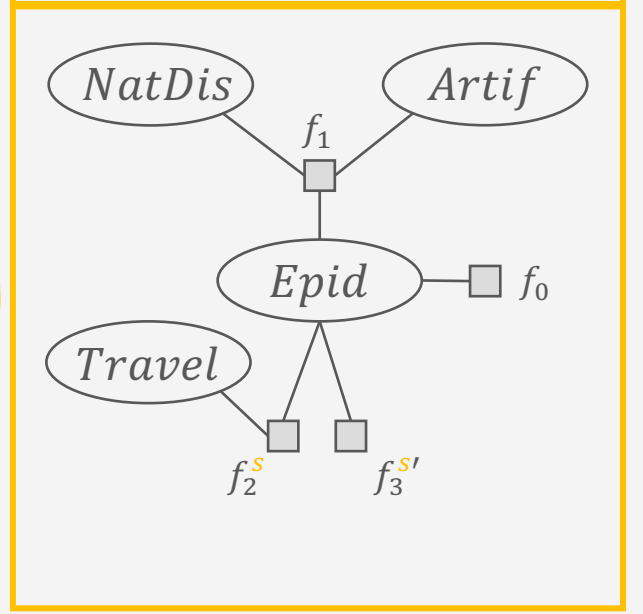
$$= \arg\max_{e \in \mathrm{Val}(E)} \phi_0(e)\phi_3^{s\prime}(e)\phi_2^{s\prime}(e) \arg\max_{n \in \mathrm{Val}(N)} \arg\max_{a \in \mathrm{Val}(A)} \phi_1(e,n,a)$$

- $Travel$ ausmaximieren:
  - Wähle die argmax Zuweisung von $Travel$ für jede mögliche Welt von $Epid$
    - $Epid = false$
      - Maximaler Wert: 24
      - Zuweisung merken: $Travel = false$
    - $Epid = true$
      - Maximaler Wert: 6
      - Zuweisung merken: $Travel = false$
  - Verbleibender Faktor: $f_2^{s\prime} = \phi_2^{s\prime}(Epid)$

$Epid$	$Travel$	ϕ_2^s
$false$	$false$	24
$false$	$true$	8
$true$	$false$	6
$true$	$true$	2

$Epid$	$\phi_2^{s\prime}$
$false$	24
$true$	6





VE für MPE Anfragen: Beispiel

$$MPE(sick)$$

$$= \arg\max_{e \in \mathrm{Val}(E)} \phi_0(e)\phi_3^{s\prime}(e)\phi_2^{s\prime}(e) \arg\max_{n \in \mathrm{Val}(N)} \arg\max_{a \in \mathrm{Val}(A)} \phi_1(e,n,a)$$

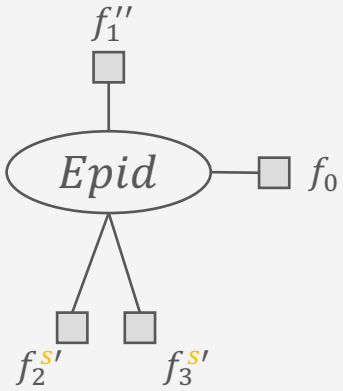
$$= \arg\max_{e \in \mathrm{Val}(E)} \phi_0(e)\phi_3^{s\prime}(e)\phi_2^{s\prime}(e)\phi_1''(e)$$

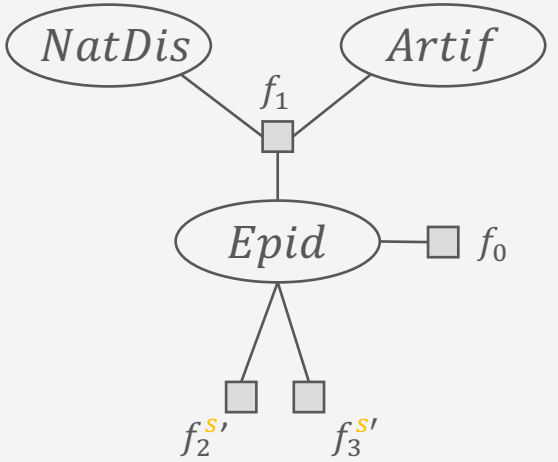
- $Artif, NatDis$ (in zwei Schritten) ausmaximieren:
  - Wähle die argmax Zuweisung von $Artif, NatDis$ nacheinander; es bleibt für jede mögliche Welt von $Epid$
    - $Epid = false$
      - Maximaler Wert: 12
      - Zuweisungen merken: $Artif = false, NatDis = false$
    - $Epid = true$
      - Maximaler Wert: 7
      - Zuweisungen merken: $Artif = false, NatDis = false$
  - Verbleibender Faktor: $f_1'' = \phi_1''(Epid)$

$Epid$	$NatDis$	$Artif$	ϕ_1
$false$	$false$	$false$	12
$false$	$false$	$true$	2
$false$	$true$	$false$	3
$false$	$true$	$true$	1
$true$	$false$	$false$	7
$true$	$false$	$true$	4
$true$	$true$	$false$	5
$true$	$true$	$true$	1

$Epid$	$NatDis$	ϕ_1'
$false$	$false$	12
$false$	$true$	3
$true$	$false$	7
$true$	$true$	5

$Epid$	ϕ_1''
$false$	12
$true$	7





VE für MPE Anfragen: Beispiel

$$MPE(sick)$$
$$= \arg\max_{e \in \mathrm{Val}(E)} \phi_0(e)\phi_3^{s'}(e)\phi_2^{s'}(e)\phi_1''(e)$$
$$= \arg\max_{e \in \mathrm{Val}(E)} \phi(e)$$
$$= \phi'()$$

- Faktormultiplikation
  - Wie vorher
- $Epid$ ausmaximieren:
  - Wähle die argmax Zuweisung von $Epid$
    - Maximaler Wert: 43200
    - Zuweisung merken: $Epid = false$
  - Verbleibender Faktor: $f' = \phi'()$

$Epid$	ϕ_0
$false$	50
$true$	1

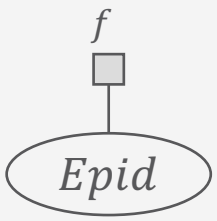
$Epid$	$\phi_3^{s'}$
$false$	3
$true$	7

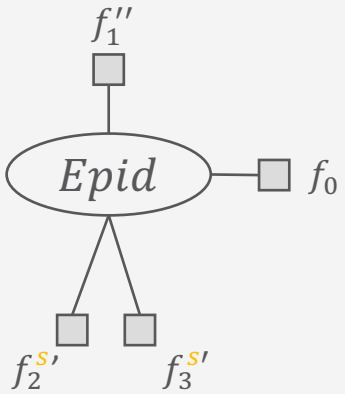
$Epid$	$\phi_2^{s'}$
$false$	24
$true$	6

$Epid$	ϕ_1''
$false$	12
$true$	7

$Epid$	ϕ
$false$	$50 \cdot 3 \cdot 24 \cdot 12 = 43200$
$true$	$1 \cdot 7 \cdot 6 \cdot 7 = 294$

ϕ'
43200





$E \triangleq Epid, N \triangleq NatDis, A \triangleq Artif, S \triangleq Sick, Tr \triangleq Travel, Tt \triangleq Treat$

Zuweisungen der Zufallsvariablen merken

- Im Faktor mitführen, bei Multiplikation vereinigen

Epid	Treat	Travel	NatDis	Artif	$\phi_0 \cdot \phi_1'' \cdot \phi_2^{s'} \cdot \phi_3^{s'} = \phi$
false	false	false	false	false	$50 \cdot 3 \cdot 24 \cdot 12 = 43200$
true	true	true	false	false	$1 \cdot 7 \cdot 6 \cdot 7 = 294$

Epid	Treat	Travel	NatDis	Artif	ϕ'
false	false	false	false	false	$50 \cdot 3 \cdot 24 \cdot 12 = 43200$

Epid	ϕ_0
false	50
true	1

Epid	Treat	ϕ_3^s
false	false	3
false	true	2
true	false	1
true	true	7

Epid	Treat	$\phi_3^{s'}$
false	false	3
true	true	7

Epid	Travel	ϕ_2^s
false	false	24
false	true	8
true	false	6
true	true	2

Epid	Travel	$\phi_2^{s'}$
false	false	24
true	false	6

Epid	NatDis	Artif	ϕ_1'
false	false	false	12
false	true	false	3
true	false	false	7
true	true	false	5

Epid	NatDis	Artif	ϕ_1''
false	false	false	12
true	false	false	7

Epid	NatDis	Artif	ϕ_1
false	false	false	12
false	false	true	2
false	true	false	3
false	true	true	1
true	false	false	7
true	false	true	4
true	true	false	5
true	true	true	1

ϕ

Zuweisungen der Zufallsvariablen merken

- Im Faktor mitführen, bei Multiplikation vereinigen

$Epid$	$Treat$	$Travel$	$NatDis$	$Artif$	$\phi_0 \cdot \phi_1'' \cdot \phi_2^{s'} \cdot \phi_3^{s'} = \phi$
$false$	$false$	$false$	$false$	$false$	$50 \cdot 3 \cdot 24 \cdot 12 = 43200$
$true$	$true$	$true$	$false$	$false$	$1 \cdot 7 \cdot 6 \cdot 7 = 293$

$Epid$	$Treat$	$Travel$	$NatDis$	$Artif$	ϕ
$false$	$false$	$false$	$false$	$false$	$50 \cdot 3 \cdot 24 \cdot 12 = 43200$

- MPE Zuweisungen $MPE_F(\boldsymbol{e})$ stehen am Ende der Inferenz direkt im Faktor
  - Zuweisungen, die man nicht mehr braucht, weil eine andere Zuweisung arg max ist, werden fallengelassen
- Aber: Faktorgröße bleibt abhängig von allen Zufallsvariablen $\mathrm{rv}(F) \setminus \mathrm{rv}(\boldsymbol{e})$
  - Immerhin nur noch *linear* in der Anzahl der ausmaximierten Variablen
  - Exponentiell in der Anzahl der noch auszumaximierenden Variablen
  - Aufwand nimmt also im Laufe der Zeit ab

VE für MPE Anfragen

- MPE: Zwei Aufgaben in einem:
  Maximierung durchführen (arg max) und Zuweisung bzw. Zustand speichern (arg max)
- arg max: In VE, ersetze Aussummierung mit Ausmaximierung
  - Rest bleibt gleich (inklusive Suche nach Eliminierungsreihenfolge bzw. Heuristik)
  - In VE Operatormenge, ersetze Operator $\text{sum}-\text{out}$ mit Operator $\max-\text{out}$
    - Gleiche Eingaben
    - Gleiche Vorbedingungen
    - Äquivalente Nachbedingung
    - Vergleichbare Spezifizierung der Ausgabe
- arg max: Erweiterte Faktordefinition um Zuweisungen zu merken
  - Anpassung der Operatoren an die erweiterte Faktordefinition

Zuweisungen der Zufallsvariablen merken

- Erweiterter Faktor $f := \phi(\boldsymbol{R})$
  - Potentialfunktion $\phi : \text{Val}(\boldsymbol{R}) \rightarrow (\boldsymbol{U}, \mathbb{R}^{0,+})$
    - $\text{rv}(f) = \boldsymbol{R}$, $\text{mrv}(f) = \boldsymbol{U}$
    - Abbildung von der Domäne von $\boldsymbol{R}$ in die Menge der positiven reellen Zahlen und die Domäne von $\boldsymbol{U}$
    - Im Rahmen von VE
      - $\boldsymbol{R}$: Variablen, die noch zu eliminieren sind
      - $\boldsymbol{U}$: Variablen, die schon eliminiert wurden
    - Zur einfacheren Referenzierung der Elemente des abgebildeten Tupels: bei Abbildung $\boldsymbol{r} \longmapsto (\boldsymbol{u}, x)$:
      - $\phi^A(\boldsymbol{r})$ gibt Zuweisung (*Assignment*) $\boldsymbol{u}$ zurück
      - $\phi^P(\boldsymbol{r})$ gibt Potential x zurück
    - Beispiel: $\boldsymbol{R} = \{Epid\}, \boldsymbol{U} = \{Travel\}$
      - $\text{rv}(f) = \{Epid\}$
      - $\text{mrv}(f) = \{Travel\}$

$Epid$	$Travel$	ϕ
$false$	$false$	24
$true$	$false$	6

$\phi^A(\neg epid) = false$ $\quad$ $\phi^A(epid) = false$

$\phi^P(\neg epid) = 24$ $\quad$ $\phi^P(epid) = 6$

Zuweisungen der Zufallsvariablen merken

- Operatordefinitionen anpassen
  - ABSORB: Unberührt von den Änderungen
    - In VE, $\boldsymbol{U} = \emptyset$, wenn ABSORB genutzt wird
  - MULTIPLY: Änderungen nötig
    - f_1, f_2 bilden auf $\boldsymbol{U}_1, \boldsymbol{U}_2$ ab → Ausgabefaktor erweitern um Vereinigung von $\boldsymbol{U}_1, \boldsymbol{U}_2$
    - $\boldsymbol{U}_1, \boldsymbol{U}_2$ dürfen keine Variablen teilen → neue Vorbedingung
      - Ansonsten wäre es möglich, dass f_1, f_2 einer Variable $U, U \in \boldsymbol{U}_1 \wedge U \in \boldsymbol{U}_2$ zwei unterschiedliche Werte zuweisen: $\phi_1(\boldsymbol{r}_1) = (x, u), \phi_2(\boldsymbol{r}_2) = (y, u'), u \neq u'$
      - In VE, $\boldsymbol{U}_1, \boldsymbol{U}_2$ schon ausmaximiert unter Vorbedingung, dass $\boldsymbol{U}_1, \boldsymbol{U}_2$ jeweils nur in einem Faktor vorkamen (Vorläufer von f_1, respektive f_2)
        → $\boldsymbol{U}_1 \cap \boldsymbol{U}_2 = \emptyset$ und Vorbedingung im Rahmen von VE erfüllt
  - SUM-OUT: Für MPE nicht genutzt

$Epid$	$Travel$	ϕ_1
$false$	$false$	24
$true$	$false$	6

$Epid$	$Travel$	ϕ_2
$false$	$true$	4
$true$	$false$	13

Multiplikation von Faktoren: Formale Definition

- Operator: MULTIPLY
  - Inputs:
    - Faktor $f_1 = \phi_1(R_1, \dots, R_n) \in F$ mit Zufallsvariablen $\boldsymbol{U}_1$ in der Zielmenge
    - Faktor $f_2 = \phi_2(S_1, \dots, S_m) \in F$ mit Zufallsvariablen $\boldsymbol{U}_2$ in der Zielmenge
  - Vorbedingung: $\boldsymbol{U}_1 \cap \boldsymbol{U}_2 = \emptyset$
  - Output: Faktor $\phi(T_1, \dots, T_k)$
    - $\{(T_1, \dots, T_k)\} = \{(R_1, \dots, R_n)\} \bowtie \{(S_1, \dots, S_m)\}$ (geordnete Vereinigung)
    - Für alle möglichen Werte $t_1, \dots, t_k$ von $T_1, \dots, T_k$, i.e., $t_1, \dots, t_k \in \mathrm{Val}(T_1, \dots, T_k)$, mit
      - $r_1, \dots, r_n = \pi_{R_1,\dots,R_n}(t_1, \dots, t_k)$, $s_1, \dots, s_m = \pi_{S_1,\dots,S_m}(t_1, \dots, t_k)$ (Auswahl passend der Argumente)

$$\phi(t_1, \dots, t_k) = \left(\phi_1^A(r_1, \dots, r_n) \cup \phi_2^A(s_1, \dots, s_m), \phi_1^P(r_1, \dots, r_n) \cdot \phi_2^P(s_1, \dots, s_m)\right)$$

  - Nachbedingung: $F \sim F \setminus \{f_1, f_2\} \cup \mathrm{MULTIPLY}(f_1, f_2)$

Ausmaximieren von Zufallsvariablen: Formale Definition

- Operator: MAX-OUT
  - Inputs:
    - Faktor $f = \phi(R_1, \dots, R_n) \in F$
    - Variable $R \in \{R_1, \dots, R_n\}$ an Position i zum Ausmaximieren
  - Vorbedingung: $\forall f' \in F \setminus \{f\}: R \notin \mathrm{rv}(f')$
  - Output: Faktor $\phi'(R_1, \dots, R_{i-1}, R_{i+1}, \dots, R_n)$
    - Für alle möglichen Werte $r_1, \dots, r_{i-1}, r_{i+1}, \dots, r_n$ von $R_1, \dots, R_{i-1}, R_{i+1}, \dots, R_n$
      - I.e., $r_1, \dots, r_{i-1}, r_{i+1}, \dots, r_n \in \mathrm{Val}(R_1, \dots, R_{i-1}, R_{i+1}, \dots, R_n)$
      - mit $r^* = \arg\max_{r \in \mathrm{Val}(R)} \phi^P(r_1, \dots, r_{i-1}, r, r_{i+1}, \dots, r_n)$

$$\phi'(r_1, \dots, r_{i-1}, r_{i+1}, \dots, r_n) = \left(\phi^A(r_1, \dots, r_{i-1}, r^*, r_{i+1}, \dots, r_n) \cup r^*, \phi^P(r_1, \dots, r_{i-1}, r^*, r_{i+1}, \dots, r_n)\right)$$

  - Nachbedingung: $\max_{r \in \mathrm{Val}(R)} P_F \equiv P_{F \setminus \{f\} \cup \mathrm{MAX\text{-}OUT}(f,R)}$

VE Algorithmus für MPE-Anfragen mit Eliminationsreihenfolge

MPE–VE$(F, \{\phi_t(T_t)\}_{t=1}^{m}, \mathcal{U})$

for $t = 1, \dots, m$ **do** — Evidenz behandeln

  while $\exists f \in F : T_t \in \mathrm{rv}(f)$ **do**

    $F \leftarrow F \setminus \{f\} \cup \{\text{ABSORB}(f, T, \phi_t(T))\}$ ▷ Absorbiere $\phi_t(T_t)$ in F

for $i = 1, \dots, \mathrm{len}(\mathcal{U})$ **do** — Nichtanfrage-Variablen eliminieren

  while $\exists f_1, f_2 \in F : U_i \in \mathrm{rv}(f_1) \wedge \mathrm{rv}(f_2)$ **do**

    $F \leftarrow F \setminus \{f_1, f_2\} \cup \{\text{MULTIPLY}(f_1, f_2)\}$ ▷ Multipliziere f_1, f_2 in F

  $F \leftarrow F \setminus \{f\} \cup \{\text{MAX–OUT}(f, U_i)\}$ ▷ Maximiere U_i aus F aus

while $\exists f_1, f_2 \in F$ **do** ▷ Multipliziere f_1, f_2 in F, bis $|F| = 1$

  $F \leftarrow F \setminus \{f_1, f_2\} \cup \{\text{MULTIPLY}(f_1, f_2)\}$ — Kombinieren

return f

MPE-VE

MPE-VE Algorithmus mit Online-Heuristik h

MPE–VE$(F, \{\phi_t(T_t)\}_{t=1}^m, h)$

for $t = 1, \dots, m$ **do** — Evidenz behandeln

  while $\exists f \in F : T_t \in \mathrm{rv}(f)$ **do**

    $F \leftarrow F \setminus \{f\} \cup \{\text{ABSORB}(f, T, \phi_t(T))\}$ ▷ Absorbiere $\phi_t(T_t)$ in F

while $\mathrm{rv}(F) \neq \emptyset$ **do** — Nichtanfrage-Variablen eliminieren

  $U \leftarrow \arg\min_R h(F)$ ▷ Wähle nächstes U zur Eliminierung

  while $\exists f_1, f_2 \in F : U \in \mathrm{rv}(f_1) \wedge \mathrm{rv}(f_2)$ **do**

    $F \leftarrow F \setminus \{f_1, f_2\} \cup \{\text{MULTIPLY}(f_1, f_2)\}$ ▷ Multipliziere f_1, f_2 in F

  $F \leftarrow F \setminus \{f\} \cup \{\text{MAX–OUT}(f, U)\}$ ▷ Maximiere U aus F aus

while $\exists f_1, f_2 \in F$ **do** ▷ Multipliziere f_1, f_2 in F, bis $|F| = 1$

  $F \leftarrow F \setminus \{f_1, f_2\} \cup \{\text{MULTIPLY}(f_1, f_2)\}$ — Kombinieren

return f — MPE-VE

Normalisierung und MPE

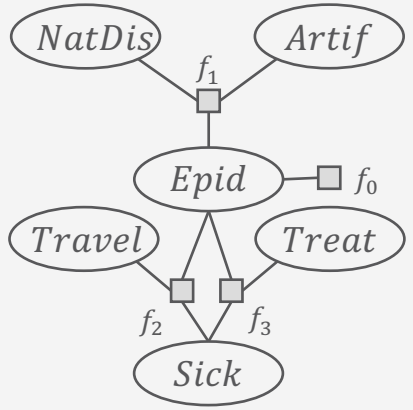
- Bei Faktorgraphen steht am Ende ein Potential im leeren Faktor, $\phi^P( ) = x$
  - Im Beispiel:

$Epid$	$Treat$	$Travel$	$NatDis$	$Artif$	ϕ
$false$	$false$	$false$	$false$	$false$	$50 \cdot 3 \cdot 24 \cdot 12 = 43200$

- Um die MPE Wahrscheinlichkeit zum MPE zu erhalten, müsste man x normalisieren
  - Dafür Berechnung der Normalisierungskonstante notwendig → eigener VE-Aufruf dafür
  - Je nach Definition vom MPE:
    - $\arg\max_{v \in \mathrm{Val}(\boldsymbol{V})} P(\boldsymbol{v} \mid \boldsymbol{e})$: $P(\boldsymbol{e})$ berechnen
    - $\arg\max_{v \in \mathrm{Val}(\boldsymbol{V})} P(\boldsymbol{v}, \boldsymbol{e})$: $P( )$ berechnen
    - Beispiel: $P(\boldsymbol{e}) = 145088$
      - MPE Wahrscheinlichkeit: $P(v \mid \boldsymbol{e}) = \frac{43200}{145088} = 0.298$

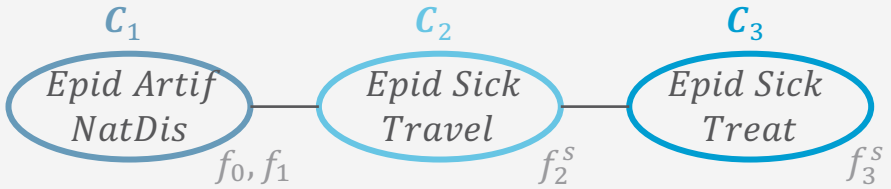
JT für MPE Anfragen

- Dtrees für MPE Anfragen sind gleich den Dtrees für Wahrscheinlichkeitsanfragen → Jtree unabhängig vom Anfragetyp
- Evidenzbehandlung in VE unverändert
- Damit Schritte 1 + 2 wie vorher:
  - Jtree z.B. mittels Dtrees bauen
  - Evidenz eingeben



Evidenz: $sick$

- Beispiel: Modell F wie abgebildet, Evidenz $sick$
  - Jtree für F bauen
  - Evidenz $sick$ behandeln

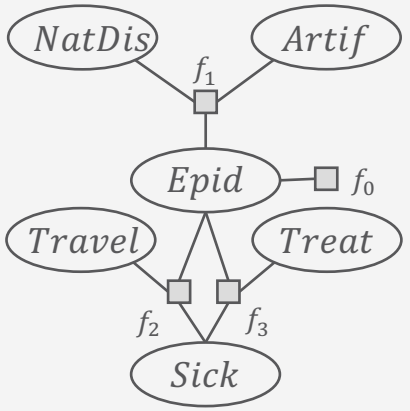


Alexander Philip Dawid. Applications of a General Propagation Algorithm for Probabilistic Expert Systems. *Statistics and Computing*, 2(1):25–36, 1992.

JT für MPE Anfragen

- Nachrichtenversand
  - Nur **eine** Phase (von der Peripherie zum Zentrum)
    - Nur gegeben Bedingung 1 Nachrichten versenden:
      Wenn Cluster $\boldsymbol{C_i}$ alle Nachrichten bis auf die von Nachbar $\boldsymbol{C_j}$ erhalten hat, sendet $\boldsymbol{C_i}$ die Nachricht m_{ij} an $\boldsymbol{C_j}$.

Sieht jetzt schon jemand, warum nur eine Phase?

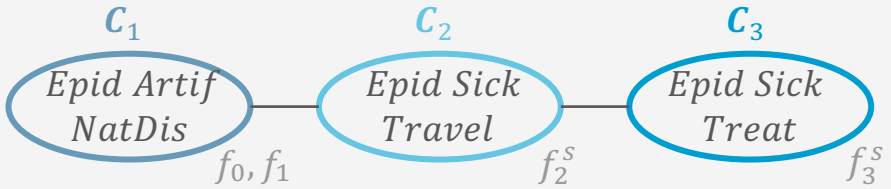


- Berechnung der Nachrichten mittels MPE-VE
  - Aber: Cluster nicht vollständig ausmaximieren, sondern nur bis runter auf $\boldsymbol{S_{ij}}$, und keine Kombination am Ende

$$m_{ij} = \text{MPE–VE–JT}\left(F_i \cup \bigcup_{\boldsymbol{C_k} \in \text{Nb}(\boldsymbol{C_i}), k \neq j} m_{ki}, \boldsymbol{S_{ij}}, h\right)$$

    - $\boldsymbol{S_{ij}}$ als Eingabe nötig (nicht ∅ wie sonst beim MPE)

Evidenz: $sick$



Alexander Philip Dawid. Applications of a General Propagation Algorithm for Probabilistic Expert Systems. *Statistics and Computing*, 2(1):25–36, 1992.

MPE-VE für Nachrichtenversand in JT: Mit Separator als Eingabe, ohne Evidenz, ohne Kombination

MPE–VE–$\mathrm{JT}(F, \boldsymbol{S}, h)$

while $\mathrm{rv}(F) \setminus \boldsymbol{S} \neq \emptyset$ **do** — Nichtseparator-Variablen eliminieren

  $U \leftarrow \arg\min_R h(F)$ ▷ Wähle nächstes U zur Eliminierung

  while $\exists f_1, f_2 \in F : U \in \mathrm{rv}(f_1) \wedge \mathrm{rv}(f_2)$ **do**

    $F \leftarrow F \setminus \{f_1, f_2\} \cup \{\mathrm{MULTIPLY}(f_1, f_2)\}$ ▷ Multipliziere f_1, f_2 in F

  $F \leftarrow F \setminus \{f\} \cup \{\mathrm{MAX\text{–}OUT}(f, U)\}$ ▷ Summiere U aus F aus

return F

JT für MPE-Anfragen: Beispiel (Forts.)

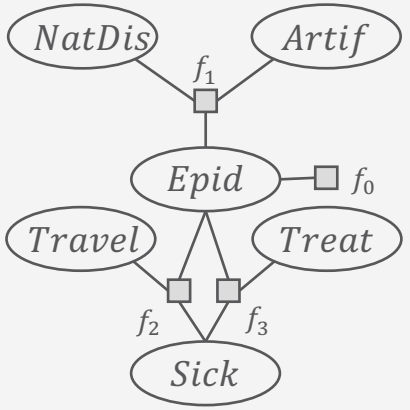
- Nachrichten senden
  - C_1: Bedingung 1 gilt mit Nachbar C_2
    - Berechnung: $m_{12} = \text{MPE}-\text{VE}-\text{JT}(\{f_0, f_1\}, \{Epid\}, .)$
      - $m_{12} = \{\phi_0(Epid), \phi_1''(Epid)\}$

Epid	NatDis	Artif	ϕ_1
false	false	false	12
false	false	true	2
false	true	false	3
false	true	true	1
true	false	false	7
true	false	true	4
true	true	false	5
true	true	true	1

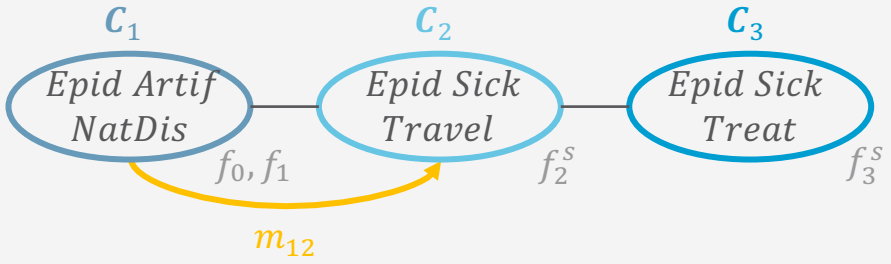
Epid	ϕ_0
false	50
true	1

Epid	NatDis	Artif	ϕ_1'
false	false	false	12
false	true	false	3
true	false	false	7
true	true	false	5

Epid	NatDis	Artif	ϕ_1''
false	false	false	12
true	false	false	7



Evidenz: *sick*

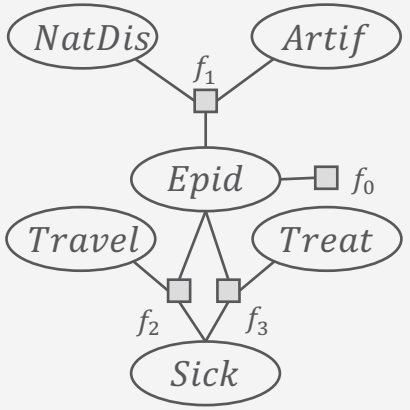


JT für MPE-Anfragen: Beispiel (Forts.)

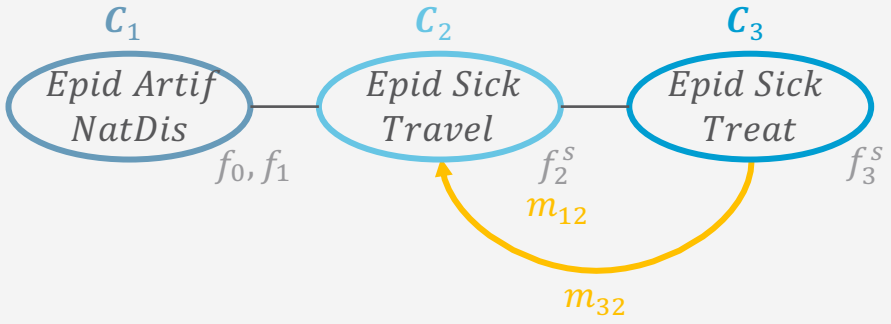
- Nachrichten senden
  - C_1: Bedingung 1 gilt mit Nachbar C_2
    - Berechnung: $m_{12} = \text{MPE–VE–JT}(\{f_0, f_1\}, \{Epid\}, .)$
      - $m_{12} = \{\phi_0(Epid), \phi_1''(Epid)\}$
  - C_3: Bedingung 1 gilt mit Nachbar C_2
    - Berechnung: $m_{32} = \text{MPE–VE–JT}(\{f_3^s\}, \{Epid\}, .)$
      - $m_{32} = \{\phi_3^{s\prime}(Epid)\}$

$Epid$	$Treat$	ϕ_3^s
$false$	$false$	3
$false$	$true$	2
$true$	$false$	1
$true$	$true$	7

$Epid$	$Treat$	$\phi_3^{s\prime}$
$false$	$false$	3
$true$	$true$	7



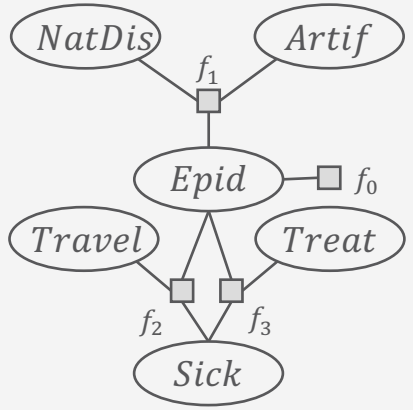
Evidenz: $sick$



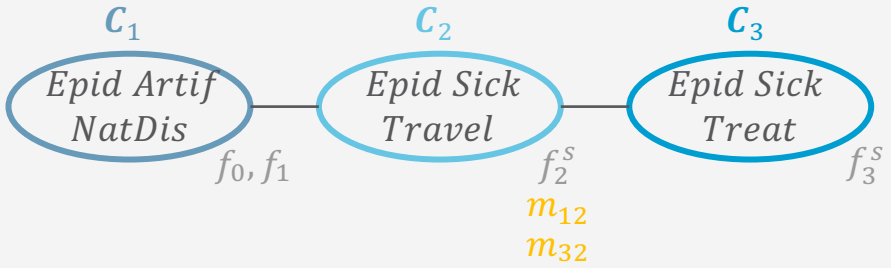
JT für MPE-Anfragen

- Anfrage in Schritt 4: MPE fertig berechnen
  - Maximiere die verbleibenden Variablen an dem Knoten aus, bei dem Bedingung 2 des Nachrichtenversands zuerst getriggert wird:
    - Wenn Cluster $\boldsymbol{C}_i$ die letzte fehlende Nachricht von Nachbar $\boldsymbol{C}_j$ erhält, sendet $\boldsymbol{C}_i$ Nachrichten m_{ik} an alle anderen Nachbarn $\boldsymbol{C}_k$
    - Anschließend MPE ausgeben
    - Aufruf an MPE-VE (nicht MPE-VE-JT, da wir ja das vollständige MPE am Ende ausgeben wollen):

$$\text{MPE}-\text{VE}\left(F_i \cup \bigcup_{\boldsymbol{C}_k \in \text{Nb}(\boldsymbol{C}_i)} m_{ki}, \emptyset, .\right)$$

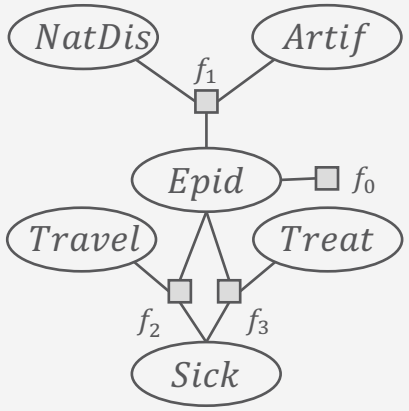


Evidenz: $sick$

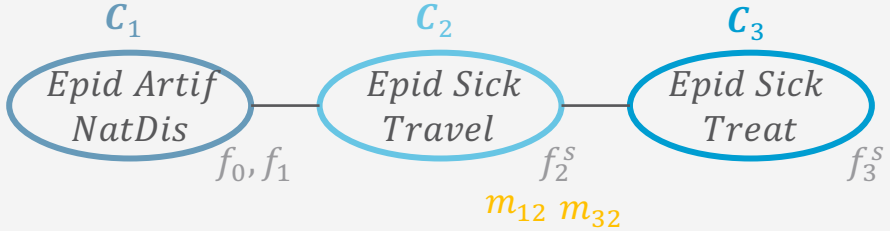


JT für MPE-Anfragen: Beispiel (Forts.)

- Anfrage in Schritt 4: MPE fertig berechnen
  - C_2: Bedingung 2 wird getriggert → Maximiere $Travel, Epid$ aus F_2, m_{12}, m_{32} aus
    - Berechnung: $f = \text{MPE–VE}(\{f_2^s, f_0, f_1'', f_3^{s'}\}, \emptyset, .)$



Evidenz: $sick$



Epid	Travel	ϕ_2^s
false	false	24
false	true	8
true	false	6
true	true	2

Epid	ϕ_0
false	50
true	1

Epid	NatDis	Artif	ϕ_1''
false	false	false	12
true	false	false	7

Epid	Treat	$\phi_3^{s'}$
false	false	3
true	true	7

Epid	Travel	$\phi_2^{s'}$
false	false	24
true	false	6

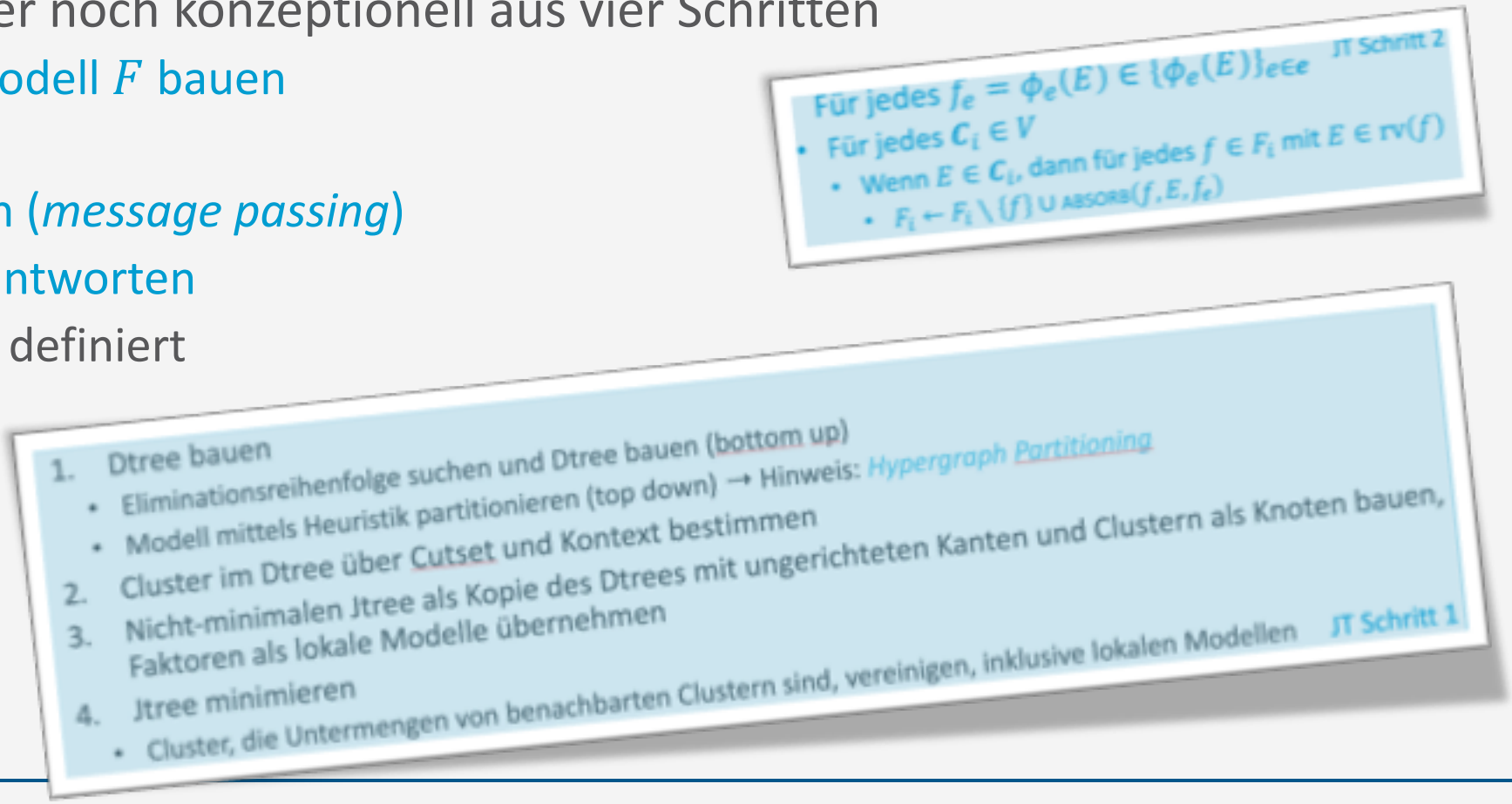
Epid	Treat	Travel	NatDis	Artif	$\phi_0 \cdot \phi_1'' \cdot \phi_2^{s'} \cdot \phi_3^{s'} = \phi$
false	false	false	false	false	$50 \cdot 3 \cdot 24 \cdot 12 = 43200$
true	true	true	false	false	$1 \cdot 7 \cdot 6 \cdot 7 = 294$

Epid	Treat	Travel	NatDis	Artif	ϕ
false	false	false	false	false	$50 \cdot 3 \cdot 24 \cdot 12 = 43200$

Tanya B and Ralf Möller. Lifted Most Probable Explanation. In *Proceedings of the International Conference on Conceptual Structures*, 2018.

MPE-JT

- JT besteht eigentlich immer noch konzeptionell aus vier Schritten
  1. Jtree J für das Eingabemodell F bauen
  2. Evidenz $\boldsymbol{e}$ in J eingeben
  3. Nachrichten in J schicken (*message passing*)
  4. MPE-Anfragen fertig beantworten
  - Schritt 1 und 2 wie vorher definiert
  - Schritt 3 und 4 angepasst
    - Algorithmisch einfacher in einem Schritt aufzuschreiben



Algorithmische Übersicht: Schritt 3 + 4 für MPE

- Gegeben eine Heuristik h
- Während es ein Cluster $\boldsymbol{C_i}$ gibt, welches noch nicht Nachrichten an alle Nachbarn verschickt hat, i.e., $\sum_{j \in \mathrm{Nb}(\boldsymbol{C_i})} 1 \mid m_{ij}\ gesendet \leq |\mathrm{Nb}(\boldsymbol{C_i})|$,
  - Wenn Bedingung 1 mit Nachbar $\boldsymbol{C_j}$ gilt
    - Berechne Nachricht m_{ij}: $m_{ij} \leftarrow \mathrm{MPE{-}VE{-}JT}\left(F_i \cup_{\boldsymbol{C_k} \in \mathrm{Nb}(C_i)} m_{ki}, \boldsymbol{S_{ij}}, h\right)$
    - Sende m_{ij} an $\boldsymbol{C_j}$
  - Wenn Bedingung 2 gilt
    - Berechne MPE fertig: $f \leftarrow \mathrm{MPE{-}VE}\left(F_i \cup_{\boldsymbol{C_k} \in \mathrm{Nb}(C_i)} m_{ki}, \emptyset, h\right)$
    - **Return** f (oder abspeichern)

MPE-JT Schritt 3 + 4

Letztlich rechnet MPE-JT genau das gleich wie MPE-VE, nur das durch den Jtree eine gewisse Eliminierungsreihenfolge schon vorgegeben ist, so dass der Suchraum der Eliminierungen für die Nachrichten kleiner ist.

Laufzeitkomplexität

- Dtree-Repräsentation einer MPE-VE Rechnung gleich der Dtree-Repräsentation einer VE-Rechnung
  - Innere Knoten repräsentieren immer noch Eliminierung, nur jetzt mittels Ausmaximierung
  - Worst-case Größe der Zwischenergebnisse gegeben durch Baumweite w
  - Anzahl der Eliminierungen gedeckelt durch Anzahl der Zufallsvariablen n, die gleich der Anzahl an inneren Knoten n_T im Dtree sind

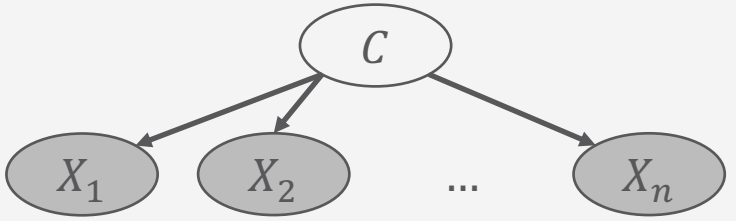
→ Ergebnisse zur Laufzeitkomplexität für Wahrscheinlichkeitsanfragen mittels VE, JT gelten auch für MPE Anfragen mittels MPE-VE, MPE-JT

$$T^{VE} \in O(n_T \cdot r^w)$$
$$T^{JT} \in O\left(n_J \cdot r^w\right)$$

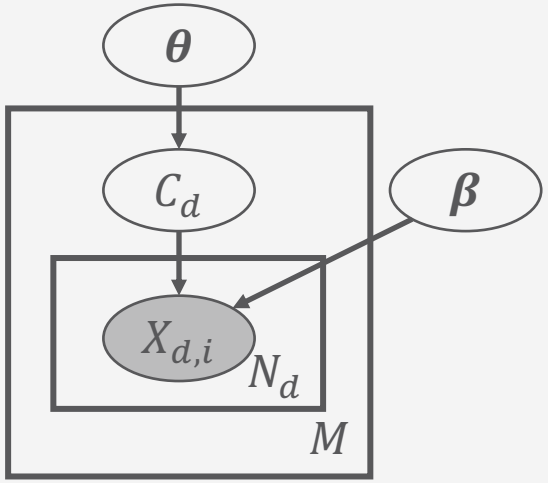
  - Nur eine MPE Anfrage pro Evidenzmenge möglich, daher $m = 1$
  - Nachrichtenverstand: nur eine Nachricht pro Kante → keine Auswirkung auf Komplexität

Anwendungen

- Naive Bayes Klassifizierer: $P(C \mid \boldsymbol{x})$
  - Nichts auszusummieren: $\boldsymbol{S} = \{C\}, \boldsymbol{T} = \mathrm{rv}(\boldsymbol{x})$ und damit
    $$\boldsymbol{U} = \boldsymbol{R} \setminus \boldsymbol{S} \setminus \boldsymbol{T} = \emptyset$$

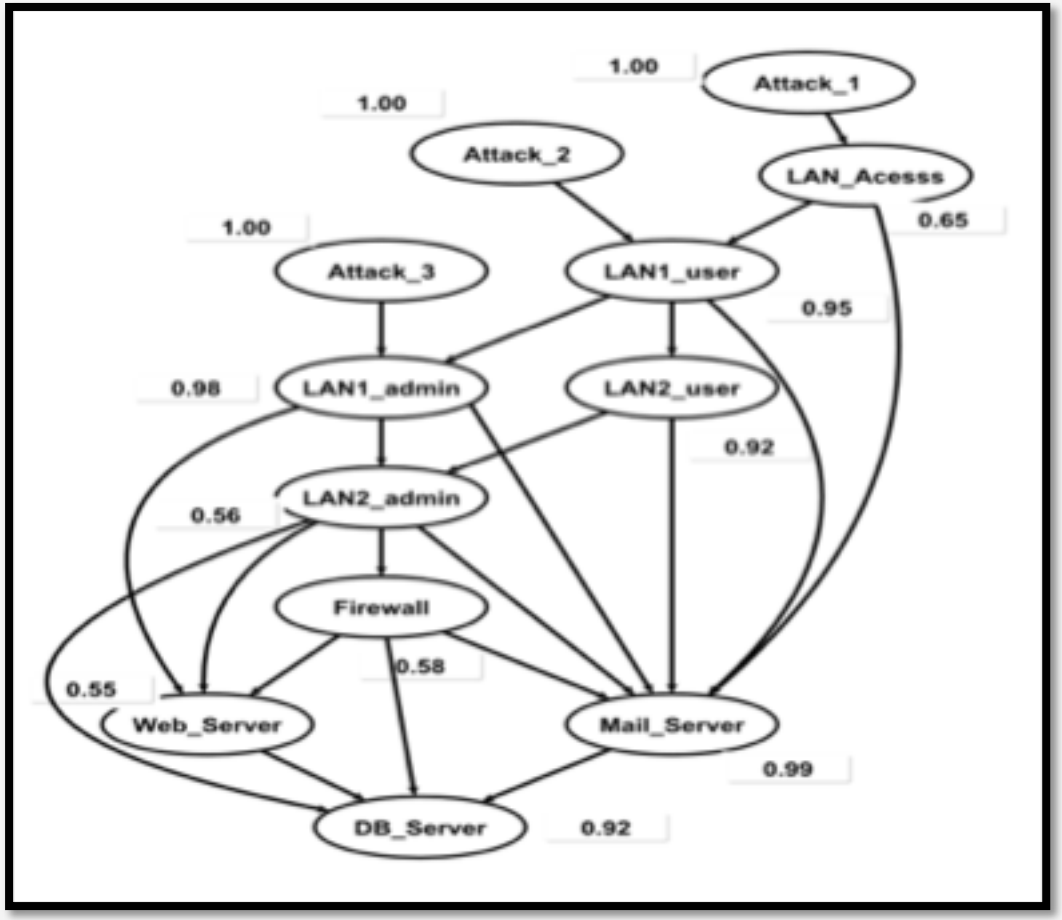


  - Eigentlich will man $\arg\max_{c \in \mathrm{Val}(C)} P(C \mid \boldsymbol{x})$ haben
    - Zustandsanfrage gegeben Evidenz:
      „Was ist der wahrscheinlichste Zustand der Zufallsvariablen, der die Evidenz hervorgerufen haben könnte?“
  - Diagnosesystem: Welche Diagnose / Krankheit hat die Symptome X_i verursacht?
  - Mixture of Unigrams: Welches Topic hat die Worte $X_{d,i}, i \in \{1, \dots, N_d\}$ hervorgerufen?



Anwendungen

- Bayesian Attack Graphs:
  Weitere Anfragemöglichkeiten
  - Neben Wahrscheinlichkeitsanfragen:
    $P(A_1), P(A_2), P(A_3), P(L_{ac}), P(L_{1u}), P(L_{2u}),$
    $P(L_{1a}), P(L_{2a}), P(F), P(S_w), P(S_m), P(S_{db})$
  - MPE Anfragen:
    - Evidenz: Server S_w, S_m, S_{db} kompromittiert, $\boldsymbol{s}$
    - $MPE(\boldsymbol{s})$ um den wahrscheinlichsten Zustand des restlichen Systems zu erhalten
      - Zustand der restlichen Komponenten
      - Ursachenforschung



Luis Muñoz-González, Daniele Sgandurra, Martín Barrère, and Emil C. Lupu: Exact Inference Techniques for the Analysis of Bayesian Attack Graphs. In: *IEEE Transactions on dependable and Secure Computing*, 2017.
Abbildungen ebenda.

Zwischenzusammenfassung

- Zustandsanfrage: Most probable explanation (MPE)
  - Gegeben Evidenz, wahrscheinlichster Zustand der verbleibenden Zufallsvariablen
  - Eliminierung mittels Maximierung
- Änderungen zu bisheriger Inferenz
  - MPE-VE
    - Erweiterter Faktor um Zuweisungen zu speichern
    - MULTIPY: vereinigt zusätzlich existierende Zuweisungen (keine Überschneidung in diesen)
    - MAX-OUT: maximiert eine gegebene Variable aus einem Faktor aus (wie SUM-OUT)
  - MPE-JT
    - Schritt 3 + 4 fusionieren zu einer Nachrichten-Phase, in der Nicht-Separatoren ausmaximiert werden und am Ende am zentralen Cluster die verbliebenen Variablen ausmaximiert werden
- Keine Auswirkung auf Komplexität in Bezug auf Baumweite und Anzahl Eliminierungen

Maximum Aposteriori Zuweisung (MAP)

Allgemeine Zustandsanfragen

Wahrscheinlichste Zuweisung

- *Zustandsanfrage* fragt nach der wahrscheinlichsten Zuweisung an eine Untermenge von Zufallsvariablen ohne Evidenz: Maximum a posteriori Zuweisung (MAP)
  - Verallgemeinerung von MPE
  - Formal betrachtet, gegeben ein Modell F mit der vollständigen gemeinsamen Wahrscheinlichkeitsverteilung P_F, Evidenz $\boldsymbol{e}$ und eine Menge von Anfragevariablen $\boldsymbol{S}$:

$$MAP_F(\boldsymbol{U} \mid \boldsymbol{e}) = \arg\max_{\boldsymbol{u} \in \mathrm{Val}(\boldsymbol{U})} \sum_{\boldsymbol{v} \in Val(\boldsymbol{V})} P(\boldsymbol{u}, \boldsymbol{v} \mid \boldsymbol{e}) = \arg\max_{\boldsymbol{u} \in \mathrm{Val}(\boldsymbol{U})} \sum_{\boldsymbol{v} \in Val(\boldsymbol{V})} P(\boldsymbol{u}, \boldsymbol{v}, \boldsymbol{e})$$

    - $\boldsymbol{V} = \mathrm{rv}(F) \setminus \mathrm{rv}(\boldsymbol{e}) \setminus \boldsymbol{U}$ die verbleibenden Zufallsvariablen
  - Wenn $\boldsymbol{U} = \mathrm{rv}(F) \setminus \mathrm{rv}(\boldsymbol{e})$, i.e., $\boldsymbol{V} = \emptyset$, dann $MAP_F(\boldsymbol{U} \mid \boldsymbol{e}) = MPE_F(\boldsymbol{e})$

Alexander Philip Dawid. Applications of a General Propagation Algorithm for Probabilistic Expert Systems. *Statistics and Computing*, 2(1):25–36, 1992.
Rina Dechter. Bucket Elimination: A Unifying Framework for Probabilistic Inference. In *Learning and Inference in Graphical Models*, pages 75–104. MIT Press, 1999.

Wahrscheinlichste Zuweisung

- Problem mit der MAP-Anfrage

$$MAP_F(\boldsymbol{U} \mid \boldsymbol{e}) = \arg\max_{\boldsymbol{u} \in \mathrm{Val}(\boldsymbol{U})} \sum_{\boldsymbol{v} \in Val(\boldsymbol{V})} P(\boldsymbol{u}, \boldsymbol{v}, \boldsymbol{e})$$

  - Enthält Summierung und Maximierung, welche nicht kommutativ sind:
    - Beispiel: Maximierung von R, Summierung von S in Faktor $\phi(t, s, r)$:

$$\max_{s \in \mathrm{Val}(S)} \sum_{r \in \mathrm{Val}(R)} \phi(t, s, r) \neq \sum_{r \in \mathrm{Val}(R)} \max_{s \in \mathrm{Val}(S)} \phi(t, s, r)$$

      - Siehe rechts
    - Verhalten zudem möglicherweise nicht eindeutig bei $\sum_{r \in \mathrm{Val}(R)} \max_{s \in \mathrm{Val}(S)} \phi(t, r, s)$ (max vor $\sum$)

Exakte Inferenz

T	S	R	ϕ_{nc}	
$false$	$false$	$false$	12	+ 14
$false$	$false$	$true$	2	
$false$	$true$	$false$	3	+ 4
$false$	$true$	$true$	1	
$true$	$false$	$false$	7	+ 11
$true$	$false$	$true$	4	
$true$	$true$	$false$	8	+ 9
$true$	$true$	$true$	1	

m 14, false

m 11, false

T	S	R	ϕ_{nc}	
$false$	$false$	$false$	12	m 12, true
$false$	$false$	$true$	2	
$false$	$true$	$false$	3	m 2, true
$false$	$true$	$true$	1	
$true$	$false$	$false$	7	m 8, true
$true$	$false$	$true$	4	
$true$	$true$	$false$	8	m 4, false
$true$	$true$	$true$	1	

+ 14, false

+ 12, true/false?

Alexander Philip Dawid. Applications of a General Propagation Algorithm for Probabilistic Expert Systems. *Statistics and Computing*, 2(1):25–36, 1992.
Rina Dechter. Bucket Elimination: A Unifying Framework for Probabilistic Inference. In *Learning and Inference in Graphical Models*, pages 75–104. MIT Press, 1999.

Wahrscheinlichste Zuweisung

- Für ein korrektes Ergebnis muss man
  - ❖ zuerst $\boldsymbol{V}$ aussummieren und
  - ❖ erst dann $\boldsymbol{U}$ ausmaximieren
  - Kann die Baumweite vergrößern im Vergleich zu MPE- und Wahrscheinlichkeitsanfragen
    - Grund: Eliminierungsfolge eingeschränkt durch Vorgabe, dass $\boldsymbol{V}$ vor $\boldsymbol{U}$
    - Bedingte Unabhängigkeiten ermöglichen Verschränkungen, aber nur bis zu einem gewissen Grad

Exakte Inferenz

T	S	R	ϕ_{nc}
$false$	$false$	$false$	12
$false$	$false$	$true$	2
$false$	$true$	$false$	3
$false$	$true$	$true$	1
$true$	$false$	$false$	7
$true$	$false$	$true$	4
$true$	$true$	$false$	8
$true$	$true$	$true$	1

+ 14, + 4 → m 14, false

+ 11, + 9 → m 11, false

T	S	R	ϕ_{nc}
$false$	$false$	$false$	12
$false$	$false$	$true$	2
$false$	$true$	$false$	3
$false$	$true$	$true$	1
$true$	$false$	$false$	7
$true$	$false$	$true$	4
$true$	$true$	$false$	8
$true$	$true$	$true$	1

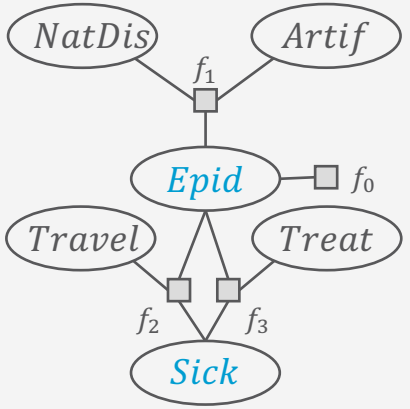
m 12, m 2 → + 14, false

m 8, m 4 → + 12, true/false?

Alexander Philip Dawid. Applications of a General Propagation Algorithm for Probabilistic Expert Systems. *Statistics and Computing*, 2(1):25–36, 1992.
Rina Dechter. Bucket Elimination: A Unifying Framework for Probabilistic Inference. In *Learning and Inference in Graphical Models*, pages 75–104. MIT Press, 1999.

Beispiel

- MAP-Anfrage mit gleicher Baumweite



$$MAP_F(Epid, Sick)$$

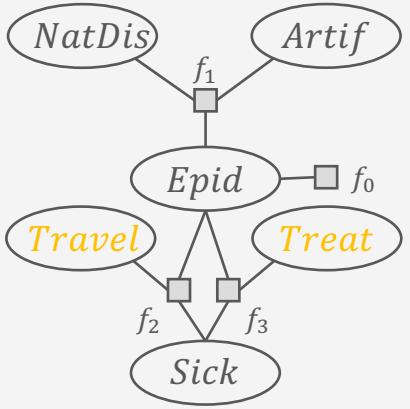
$$= \arg\max_{e,s \in \mathrm{Val}(E,S)} \sum_{n,a,tl,tt \in \mathrm{Val}(N,A,Tl,Tt)} P(e,n,a,s,tl,tt)$$

$$= \arg\max_{e,s \in \mathrm{Val}(E,S)} \sum_{n,a,tl,tt \in \mathrm{Val}(N,A,Tl,Tt)} \phi_0(e)\phi_1(e,n,m)\phi_2(tl,e,s)\phi_3(e,s,tt)$$

$$= \arg\max_{e \in \mathrm{Val}(E)} \phi_0(e) \sum_{n \in \mathrm{Val}(N)} \sum_{a \in \mathrm{Val}(A)} \phi_1(e,n,a) \arg\max_{s \in \mathrm{Val}(S)} \sum_{tl \in \mathrm{Val}(Tl)} \phi_2(tl,e,s) \sum_{tt \in \mathrm{Val}(Tt)} \phi_3(e,s,tt)$$

Beispiel

- MAP-Anfrage mit vergrößerter Baumweite



$$MAP_F(Travel, Treat)$$

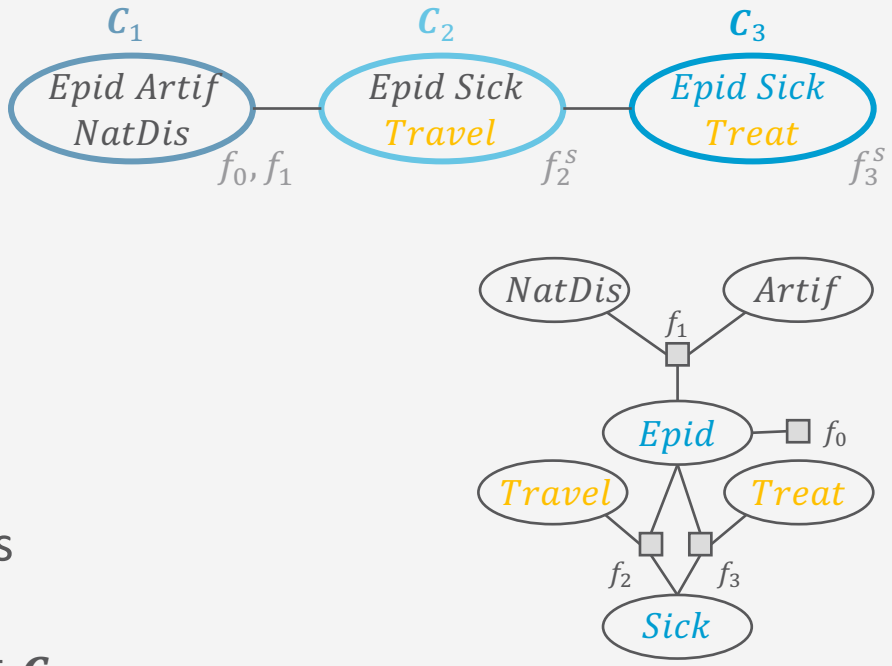
$$= \arg\max_{tl,tt \in \mathrm{Val}(Tl,Tt)} \sum_{e,s,n,a \in \mathrm{Val}(E,S,N,A)} P(e,n,a,s,tl,tt)$$

$$= \arg\max_{tl,tt \in \mathrm{Val}(Tl,Tt)} \sum_{e,s,n,a \in \mathrm{Val}(E,S,N,A)} \phi_0(e)\phi_1(e,n,a)\phi_2(tl,e,s)\phi_3(e,s,tt)$$

$$= \arg\max_{tl \in \mathrm{Val}(Tl)} \arg\max_{tt \in \mathrm{Val}(TT)} \sum_{e \in \mathrm{Val}(E)} \phi_0(e) \sum_{n \in \mathrm{Val}(N)} \sum_{a \in \mathrm{Val}(A)} \phi_1(e,n,a) \sum_{s \in \mathrm{Val}(S)} \phi_2(tl,e,s)\phi_3(e,s,tt)$$

Beschränkte MAP-Anfragen

- Beschränkte MAP-Anfragen $MAP_F(\boldsymbol{U} \mid \boldsymbol{e})$ an ein Modell F mit Jtree J:
  - Baumweite steigt nicht im Vergleich zu MPE- und Wahrscheinlichkeitsanfragen, wenn $\boldsymbol{U}$ mindestens die Separatoren des Subgraphs J' enthält, in dem $\boldsymbol{U}$ in J vorkommt
    - Eliminierungsfolge nicht durcheinander bringen
    - Formal: Es muss für Subgraph $J' \subseteq J : \boldsymbol{U} \subseteq \mathrm{rv}(J')$ gelten, dass
      $$\forall \{i, j\} \in J' : \boldsymbol{S}_{ij} \subseteq \boldsymbol{U}$$
      - *Bemerkung 1*: J' enthält ein Cluster $\boldsymbol{C}_i \rightarrow$ keine Separatoren, $\boldsymbol{U} \subseteq \boldsymbol{C}_i$
      - *Bemerkung 2*: $\mathrm{rv}(J') = \boldsymbol{U} \rightarrow$ Separatoren immer enthalten
      - *Bemerkung 3*: $J' = J \wedge \mathrm{rv}(J') = \boldsymbol{U} \rightarrow$ MPE-Anfrage
- Dann gelten auch die Komplexitätsresultate: $O(n \cdot r^w)$

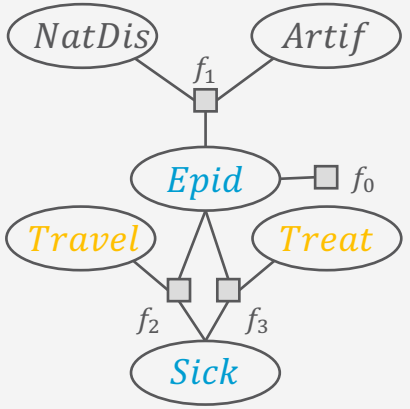


Beschränkte MAP-Anfragen

Baumweite steigt nicht, wenn $\boldsymbol{U}$ mindestens die Separatoren des Subgraphs J' enthält, in dem $\boldsymbol{U}$ in J vorkommt:

$$\forall \{i,j\} \in J' : \boldsymbol{S}_{ij} \subseteq \boldsymbol{U}, J' \subseteq J : \boldsymbol{U} \subseteq \mathrm{rv}(J')$$

- Bemerkung 1: J' enthält ein Cluster $\boldsymbol{C}_i \rightarrow$ keine Separatoren, $\boldsymbol{U} \subseteq \boldsymbol{C}_i$
- Bemerkung 2: $\mathrm{rv}(J') = \boldsymbol{U} \rightarrow$ Separatoren immer enthalten



- Beispiele:
  - $MAP_F(Epid, Sick)$
    - Teil eines Clusters, $J' = (\boldsymbol{C}_2, \emptyset), \{Epid, Sick\} \subseteq \boldsymbol{C}_2$
  - $MAP_F(Treat, Travel)$
    - Enthält nicht alle Separatoren, $J' = (\{\boldsymbol{C}_2, \boldsymbol{C}_3\}, \{\{2,3\}\})$: $\boldsymbol{S}_{23} = \{Epid, Sick\}$ fehlt
  - $MAP_F(Epid, Sick, Treat, Travel)$
    - Alle Variablen des Subgraphs $\{\boldsymbol{C}_2, \boldsymbol{C}_3\}$
  - $MAP_F(Epid, Travel, NatDis, Artif)$
    - Enthält alle Separatoren, $J' = (\{\boldsymbol{C}_1, \boldsymbol{C}_2\}, \{\{1,2\}\})$: $\boldsymbol{S}_{12} = \{Epid\}$ enthalten

Notiz nach Vorlesung: Berücksichtigt nicht, dass Eliminierungen bei Unabhängigkeit verschränkbar
- Nur eine Eliminationsreihenfolge bauen (dazu notieren, ob $\arg\max$, $\sum$)

VE Algorithmus für MAP-Anfragen mit Eliminationsreihenfolgen

MAP–VE$(F, \{\phi_t(T_t)\}_{t=1}^{m}, \mathcal{V}, \mathcal{U})$

for $t = 1, \dots, m$ **do** — Evidenz behandeln
  while $\exists f \in F : T_t \in \mathrm{rv}(f)$ **do**
    $F \leftarrow F \setminus \{f\} \cup \{\text{ABSORB}(f, T, \phi_t(T))\}$ ▹ Absorbiere $\phi_t(T_t)$ in F

for $i = 1, \dots, \mathrm{len}(\mathcal{V})$ **do** — Nichtanfrage-Variablen eliminieren
  while $\exists f_1, f_2 \in F : V_i \in \mathrm{rv}(f_1) \wedge \mathrm{rv}(f_2)$ **do**
    $F \leftarrow F \setminus \{f_1, f_2\} \cup \{\text{MULTIPLY}(f_1, f_2)\}$ ▹ Multipliziere f_1, f_2 in F
  $F \leftarrow F \setminus \{f\} \cup \{\text{SUM–OUT}(f, V_i)\}$ ▹ Summiere V_i aus F aus

for $i = 1, \dots, \mathrm{len}(\mathcal{U})$ **do** — Anfrage-Variablen eliminieren
  while $\exists f_1, f_2 \in F : U_i \in \mathrm{rv}(f_1) \wedge \mathrm{rv}(f_2)$ **do**
    $F \leftarrow F \setminus \{f_1, f_2\} \cup \{\text{MULTIPLY}(f_1, f_2)\}$ ▹ Multipliziere f_1, f_2 in F
  $F \leftarrow F \setminus \{f\} \cup \{\text{MAX–OUT}(f, U_i)\}$ ▹ Maximiere U_i aus F aus

while $\exists f_1, f_2 \in F$ **do** — ▹ Multipliziere f_1, f_2 in F, bis $|F| = 1$
  $F \leftarrow F \setminus \{f_1, f_2\} \cup \{\text{MULTIPLY}(f_1, f_2)\}$ — Kombinieren

return f — MAP-VE

MAP-VE Algorithmus mit Online-Heuristik h

MAP–VE$(F, \boldsymbol{U}, \{\phi_t(T_t)\}_{t=1}^m, h)$

for $t = 1, \dots, m$ **do**
  while $\exists f \in F : T_t \in \mathrm{rv}(f)$ **do**
    $F \leftarrow F \setminus \{f\} \cup \{\text{ABSORB}(f, T, \phi_t(T))\}$

Evidenz behandeln

▷ Absorbiere $\phi_t(T_t)$ in F

while $\mathrm{rv}(F) \setminus \boldsymbol{U} \neq \emptyset$ **do**

Nichtanfrage-Variablen eliminieren

  $V \leftarrow \arg\min_R h(F)$

▷ Wähle nächstes V zur Eliminierung

  while $\exists f_1, f_2 \in F : V \in \mathrm{rv}(f_1) \wedge \mathrm{rv}(f_2)$ **do**
    $F \leftarrow F \setminus \{f_1, f_2\} \cup \{\text{MULTIPLY}(f_1, f_2)\}$

▷ Multipliziere f_1, f_2 in F

  $F \leftarrow F \setminus \{f\} \cup \{\text{SUM–OUT}(f, V)\}$

▷ Maximiere V aus F aus

while $\mathrm{rv}(F) \neq \emptyset$ **do**

Anfrage-Variablen eliminieren

  $U \leftarrow \arg\min_R h(F)$

▷ Wähle nächstes U zur Eliminierung

  while $\exists f_1, f_2 \in F : U \in \mathrm{rv}(f_1) \wedge \mathrm{rv}(f_2)$ **do**
    $F \leftarrow F \setminus \{f_1, f_2\} \cup \{\text{MULTIPLY}(f_1, f_2)\}$

▷ Multipliziere f_1, f_2 in F

  $F \leftarrow F \setminus \{f\} \cup \{\text{MAX–OUT}(f, U)\}$

▷ Maximiere U_i aus F aus

while $\exists f_1, f_2 \in F$ **do**
  $F \leftarrow F \setminus \{f_1, f_2\} \cup \{\text{MULTIPLY}(f_1, f_2)\}$

▷ Multipliziere f_1, f_2 in F, bis $|F| = 1$

Kombinieren

return f

MAP-VE

JT für eine Menge von MAP-Anfragen: Algorithmus

MAP–JT$(F, \{\boldsymbol{U}_i\}_{i=1}^n, \boldsymbol{e})$

Baue Jtree J für F

Behandle Evidenz $\boldsymbol{e}$ in J

Versende Nachrichten in J mittels VE-JT ▷ Nichtseparatoren aussummieren

for jede MAP-Anfrage mit Anfragevariablen $\boldsymbol{U}_i$ **do**

Finde Subgraph J', so dass $\boldsymbol{U}_i \subseteq \mathrm{rv}(J')$

$F' \leftarrow \bigcup_{i \in J'} F_i \cup \bigcup_{k \in J \wedge k \notin J'} m_{ki}$

$f \leftarrow \mathrm{MAP\text{–}VE}(F', \boldsymbol{U}_i, \emptyset)$

f ausgeben oder speichern

if $\boldsymbol{U}_i = \mathrm{rv}(J')$ **then**
Nutze MPE-LJT: Schritt 3 + 4 auf J'

MAP-JT

- Alles außerhalb J' muss aussummiert werden, daher VE-JT für Nachrichten
- Innerhalb J' muss der Rest $\mathrm{rv}(J') \setminus \boldsymbol{U}_i$ aussummiert und dann $\boldsymbol{U}_i$ ausmaximiert werden

Approximative MAP-Anfragenbeantwortung

$$MAP_F(\boldsymbol{U} \mid \boldsymbol{e}) = \arg\max_{\boldsymbol{u} \in \mathrm{Val}(\boldsymbol{U})} \sum_{\boldsymbol{v} \in Val(\boldsymbol{V})} P(\boldsymbol{u}, \boldsymbol{v}, \boldsymbol{e})$$

- MAP Anfrage $MAP_G\left(\boldsymbol{U}_{|C'} | \boldsymbol{e}\right)$ approximativ beantworten
  - Ersetze sum mit max

$$\widetilde{MAP}_F(\boldsymbol{U} \mid \boldsymbol{e}) = \arg\max_{\boldsymbol{u} \in \mathrm{Val}(\boldsymbol{U})} \max_{\boldsymbol{v} \in Val(\boldsymbol{V})} P(\boldsymbol{u}, \boldsymbol{v}, \boldsymbol{e}) = \pi_{\boldsymbol{U}}\left(MPE_F(\boldsymbol{e})\right)$$

    - Operationen für $\boldsymbol{U}$ und $\boldsymbol{V}$ sind wieder kommutativ
    - Berechnet eine MPE-Anfrage und projiziert das Ergebnis auf die Anfragevariablen der MAP-Anfrage
    - Liefert eine untere Schranke für das MAP Potential am Ende

A	B	ϕ	$\sum_B$	argmax_A
$false$	$false$	1	6 4	
$false$	$true$	3 5		
$true$	$false$	3	7	$A = true$
$true$	$true$	4		

$MAP_\phi(A)$

A	B	ϕ	$\max_B$	argmax_A
$false$	$false$	1	5 3	$A = false$
$false$	$true$	3 5		
$true$	$false$	3	4	$A = true$
$true$	$true$	4		

$\widetilde{MAP}_\phi(A)$

Kann dieselbe MAP-Zuweisung liefern, muss aber nicht sein

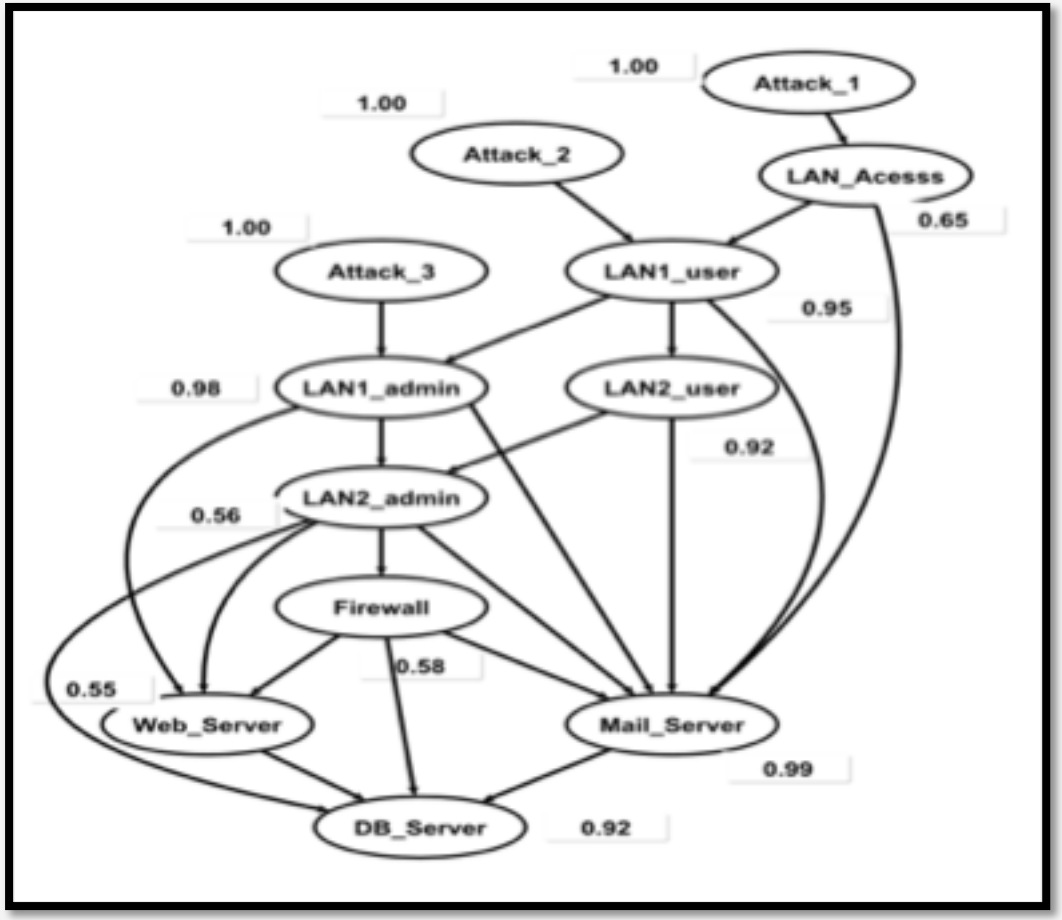
Kombinierter JT für alle Anfragetypen

- Gegeben Anfragen der verschiedenen Typen
  - Typen:
    - MPE
    - MAP
    - Wahrscheinlichkeit
      - Anfrage zu einer (bedingten) Wahrscheinlichkeit (-sverteilung)
- Auf einer Evidenzmenge
- Jtree so viel wie möglich wiederbenutzen

ComJT$(F, \{\boldsymbol{Q}_i\}_{i=1}^n, \boldsymbol{e})$
Baue Jtree J für F
Behandle Evidenz $\boldsymbol{e}$ in J
Versende Nachrichten in J mittels VE-JT
for jede Anfrage mit Anfragevariablen $\boldsymbol{Q}_i$ und Typ τ **do**
if $\tau =$ MPE **then**
Mache Schritt 3+4 aus MPE-JT in J
else
Finde Subgraph J', so dass $\boldsymbol{Q}_i \subseteq \mathrm{rv}(J')$
if $\tau =$ MAP, $\boldsymbol{Q}_i = \mathrm{rv}(J')$ **then**
Mache Schritt 3+4 aus MPE-JT in J'
else
$F' \leftarrow \mathrm{U}_{i \in J'} F_i \cup \mathrm{U}_{k \in J \wedge k \notin J'} m_{ki}$
if $\tau =$ MAP **then**
$\mathrm{MAP{-}VE}(F', \boldsymbol{Q}_i, \emptyset)$
else
$\mathrm{VE}(F', \boldsymbol{Q}_i, \emptyset)$

Anwendungen

- Bayesian Attack Graphs: Anfragemöglichkeiten
  - Wahrscheinlichkeitsanfragen:
    $P(A_1), P(A_2), P(A_3), P(L_{ac}), P(L_{1u}), P(L_{2u}),$
    $P(L_{1a}), P(L_{2a}), P(F), P(S_w), P(S_m), P(S_{db})$
  - MPE Anfragen:
    - Evidenz: Server S_w, S_m, S_{db} kompromittiert, $\boldsymbol{s}$
    - $MPE(\boldsymbol{s})$ um den wahrscheinlichsten Zustand des restlichen Systems zu erhalten
  - MAP Anfragen
    - Evidenz: A_1, L_{ac} (Wissen um einen Angriff)
    - $MAP(S_w, S_m, S_{db})$ um den wahrscheinlichsten Zustand der Server zu erhalten



Luis Muñoz-González, Daniele Sgandurra, Martín Barrère, and Emil C. Lupu: Exact Inference Techniques for the Analysis of Bayesian Attack Graphs. In: *IEEE Transactions on dependable and Secure Computing*, 2017.
Abbildungen ebenda.

Zwischenzusammenfassung

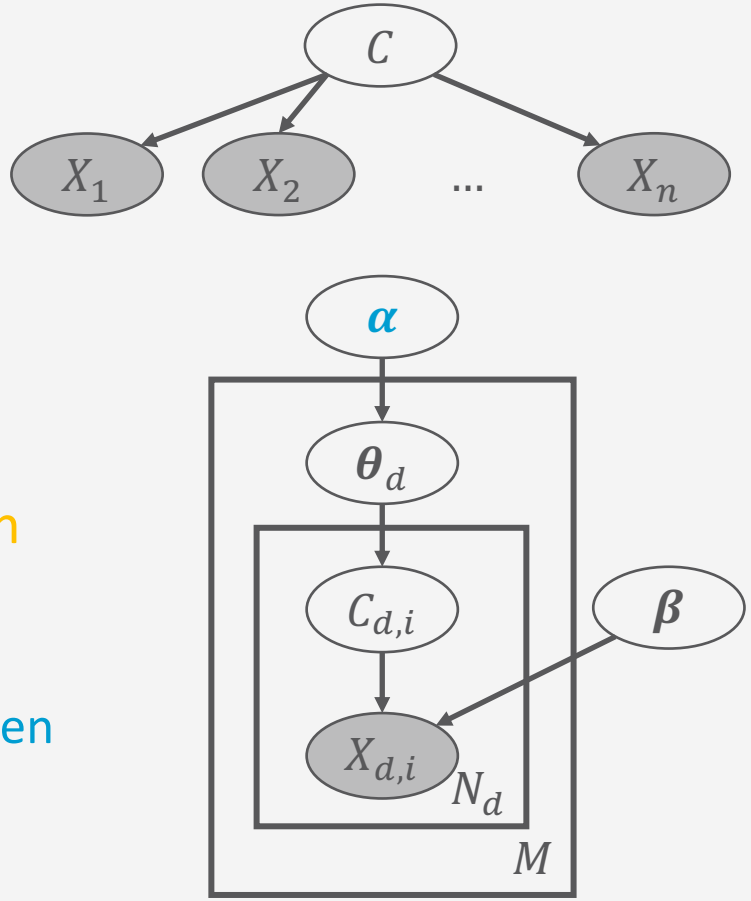
- Generelle Zustandsanfragen: Maximum a posteriori Anfragen (MAP)
  - Generalisierung von MPE
- Maximiere Anfragevariablen aus, nachdem Nichtanfragevariablen aussummiert wurden
  - Wenn Anfragevariablen = alle Variablen ohne Evidenz, dann MAP = MPE
- Problem $\sum$ und argmax nicht kommutativ
  - Kann die Baumweite vergrößern
  - Bei Anfragen an Zufallsvariablen, die mindestens die Separatoren in dem Subgraph aufdecken, der die Anfragevariablen abdeckt, bleibt die Baumweite gleich der von Wahrscheinlichkeits- / MPE-Anfragen
- MAP-VE mit zwei Schleifen, eine für Aussummieren, eine für Ausmaximieren
- MAP-JT mit normalen VE-Nachrichten im Jtree und Aufruf von MAP-VE auf dem Subgraph
- Com-JT für eine Menge von Anfragen unterschiedlichen Typs

Bemerkung zur Nomenklatur

- Mit BNs als Forschungshintergrund, Namen der Anfragetypen wie hier benutzt:
  1. Wahrscheinlichkeitsanfragen
  2. MPE-Anfrage
  3. MAP-Anfrage
- Mit MNs als Forschungshintergrund, Namen häufig eher
  1. Wahrscheinlichkeitsanfragen = *marginale Anfrage*
  2. MPE-Anfrage = *MAP-Anfrage*
  3. MAP-Anfrage = *marginale MAP-Anfrage*
- In der Logik wären die korrespondierenden Inferenzprobleme
  1. Wahrscheinlichkeitsanfragen = *marginale Anfrage* ≙ *Induktion*
  2. MPE-Anfrage = *MAP-Anfrage* ≙ *Abduktion*

Anwendungen

- Mixture of Topics
  - Mögliche Anfrage: Topic eines Wortes $P(C_{d,i} \mid \boldsymbol{x}_d)$
    - $\boldsymbol{S} = \{C_{d,i}\}, \boldsymbol{T} = \mathrm{rv}(\boldsymbol{x}_d)$ und damit
      $\boldsymbol{U} = \boldsymbol{R} \setminus \boldsymbol{S} \setminus \boldsymbol{T} = \{C_{d,j}\}_{i \neq j} \cup \{C_{d',k}\}_{d' \neq d}$
  - Unabhängigkeit zwischen den Dokumenten
    → Vereinfachung: nur d betrachten, damit $\boldsymbol{U} = \{C_{d,j}\}_{i \neq j}$
  - Problem: Immer noch (zu) viele Zufallsvariablen auszusummieren
    → Nächstes Thema:
    Approximative Inferenz durch Sampling (Thema 4)
    - Mögliche Anfrage: Fragmente eines Textes (mit Topic) vervollständigen
      $P(\boldsymbol{x}'_d \mid \boldsymbol{x}''_d)$ (bzw. $P(\boldsymbol{x}'_d \mid \boldsymbol{x}''_d, C_{d,i})$) mit $\boldsymbol{x}'_d \cup \boldsymbol{x}''_d = \boldsymbol{x}_d$
      - Werte (Worte) sample'n



Überblick: 3. Exakte Inferenz in episodischen PGMs

A. *Einzelanfragen: Variableneliminierung (VE)*
   - Algorithmus, Operatoren für Wahrscheinlichkeitsanfragen
   - Dekompositionsbäume, Komplexität

B. *Multi-Anfragen: Junction Tree (Cliquen-Bäume) Algorithmus (JT)*
   - Cliquen, Junction Tree als Hilfsstruktur, Vorverarbeitung und Anfragebeantwortung
   - Zusammenhang mit VE, Komplexität
   - *Geschichtsstunde: Pearl's Probability Propagation (PP)* auf Polytree BNs

C. *Inferenzproblem Zustandsanfragen*
   - Ausprägungen: Most probable explanation (MPE) / maximum a posteriori Anfragen (MAP)
   - Semantik: Ausmaximieren statt aussummieren; max-out Operator in VE
   - Auswirkungen auf die Komplexität

→ Approximative Inferenz in episodischen PGMs

Einordnung der Vorlesung: *Modell- und nutzenbasierter Agent*

- Nachfolgende Themen der Vorlesung
  2. Episodische PGMs
  3. Exakte Inferenz in episodischen PGMs
  4. Approximative Inferenz in episodischen PGMs
  5. Lernalgorithmen für episodische PGMs
  6. Sequentielle PGMs und Inferenz
  7. Entscheidungstheoretische PGMs