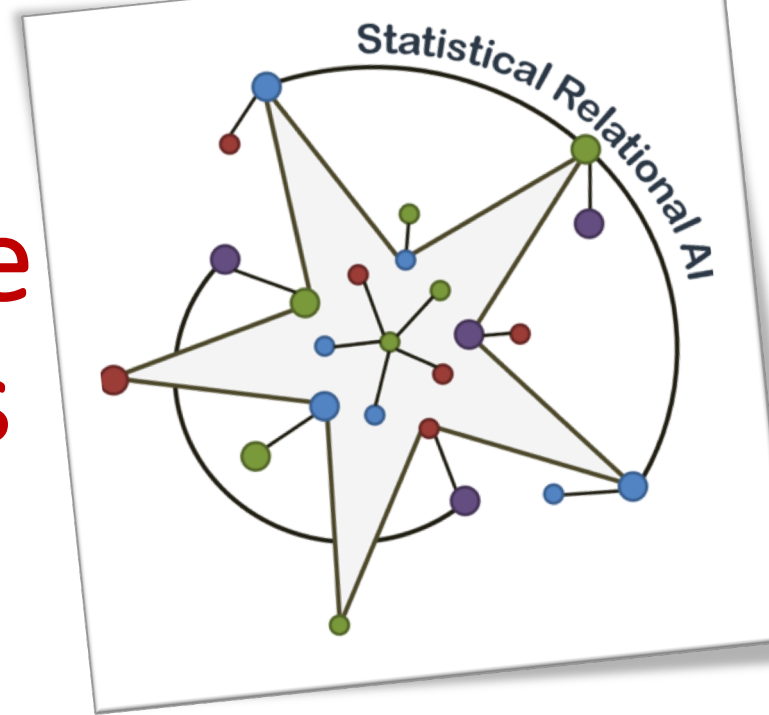


Exact Lifted Inference on Relational Models

Statistical Relational AI

Tutorial at ICCS 2019



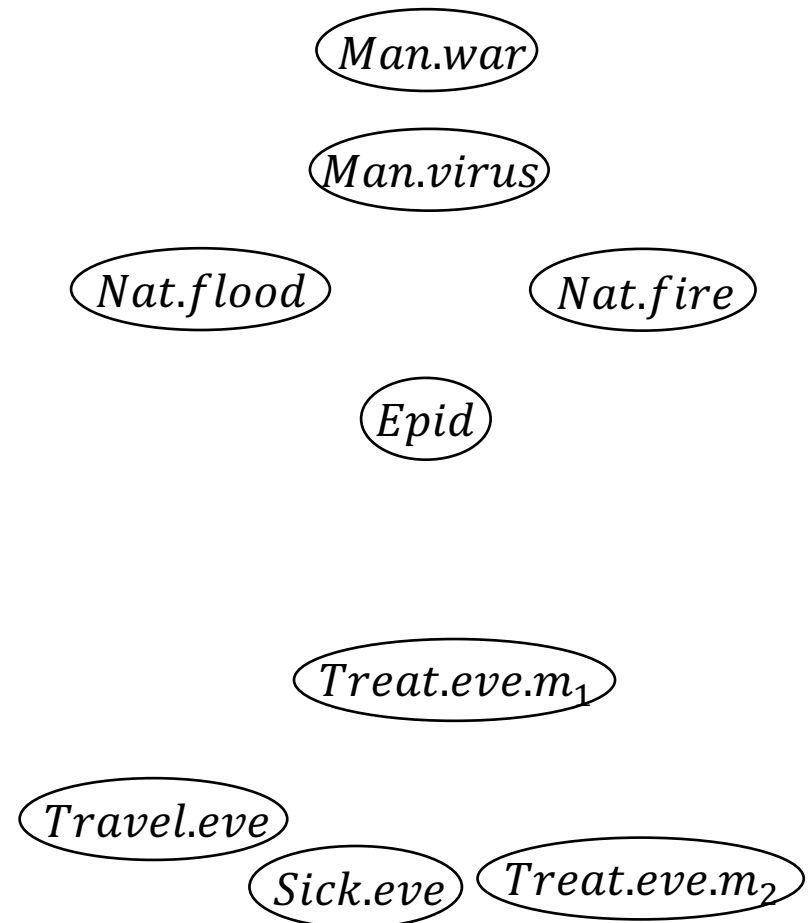
Tanya Braun, University of Lübeck



UNIVERSITÄT ZU LÜBECK

Propositional Models: Worlds

- Characterise world by **random variables**
 - E.g., *Epid*
 - Possible values (range):
 $r(Epid) = \{true, false\}$
- 1 world = specific values for variables
- Probability per world
- **Joint probability distribution** P_F over all possible worlds



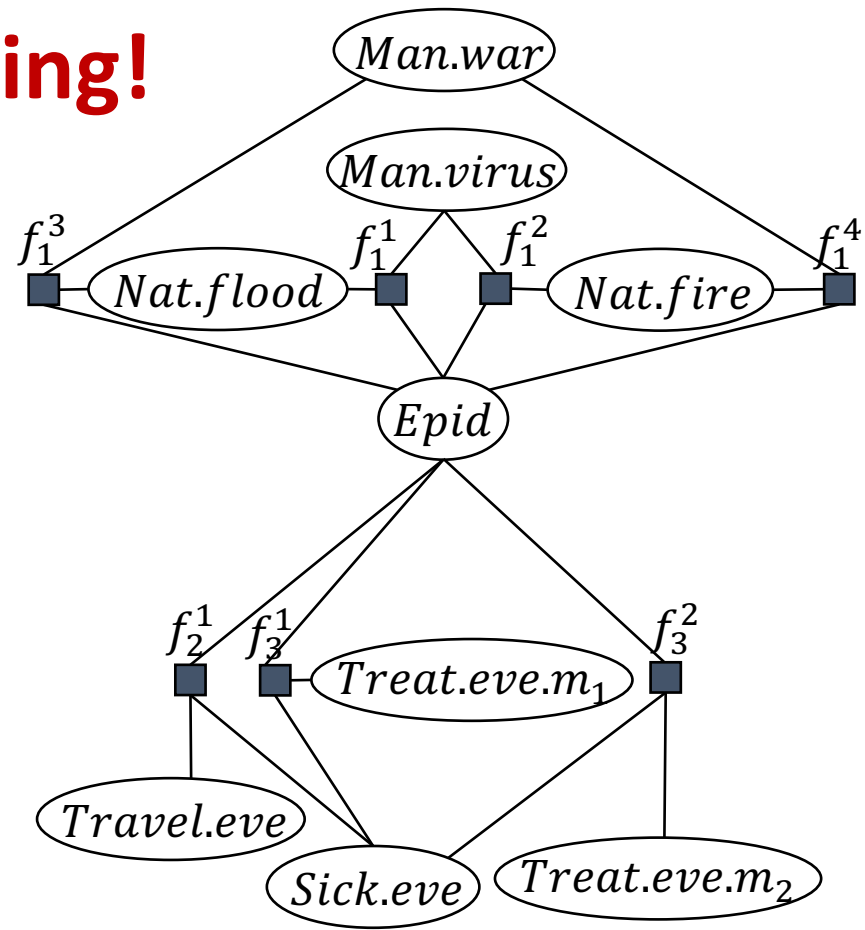
$2^9 = 512$ possible worlds

Propositional Models: Factors

- Full joint P_F as product of factors **Sparse encoding!**
 - Model F
 - $rv(F), rv(f)$
random variables in F, f

$$P_F = \frac{1}{Z} \prod_{i=1}^n f_i$$

$$Z = \sum_{v \in r(rv(F))} \prod_{i=1}^n f_i(\pi_{rv(f_i)}(v))$$



$7 \cdot 2^3 = 56$ entries

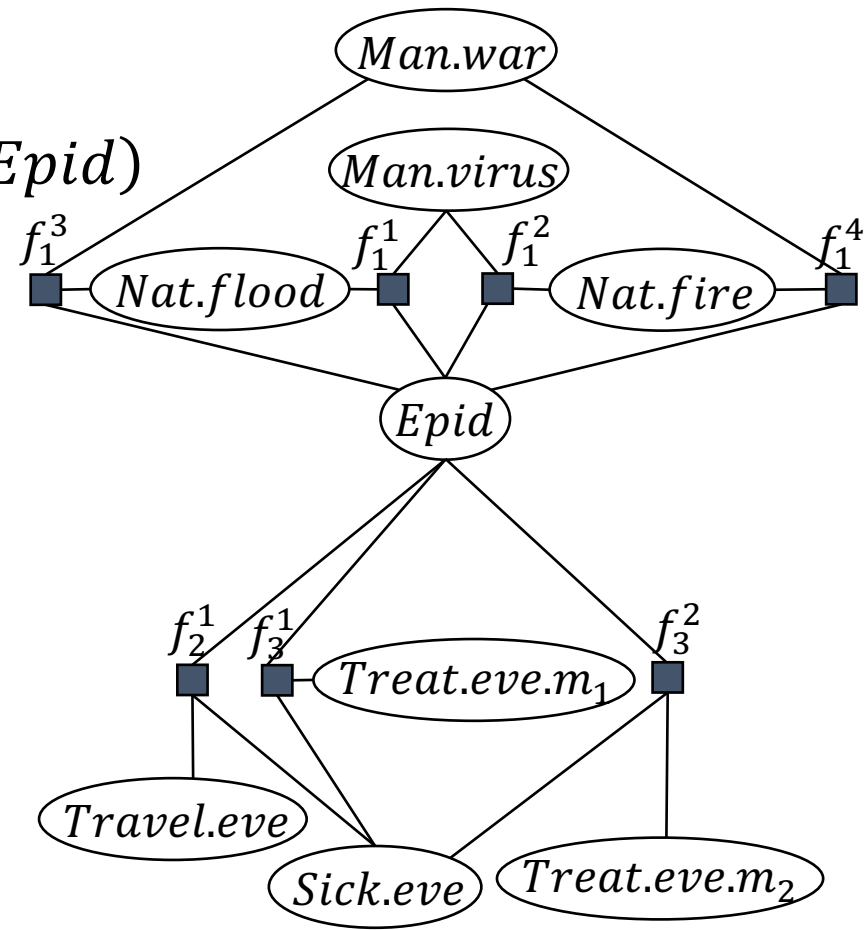
Propositional Models: Factors

- Factors

- Potentials

- $f_2^1(\text{Travel.eve}, \text{Sick.eve}, \text{Epid})$

<i>Travel.eve</i>	<i>Epid</i>	<i>Sick.eve</i>	f_2^1
false	false	false	5
false	false	true	0
false	true	false	4
false	true	true	6
true	false	false	4
true	false	true	6
true	true	false	2
true	true	true	9

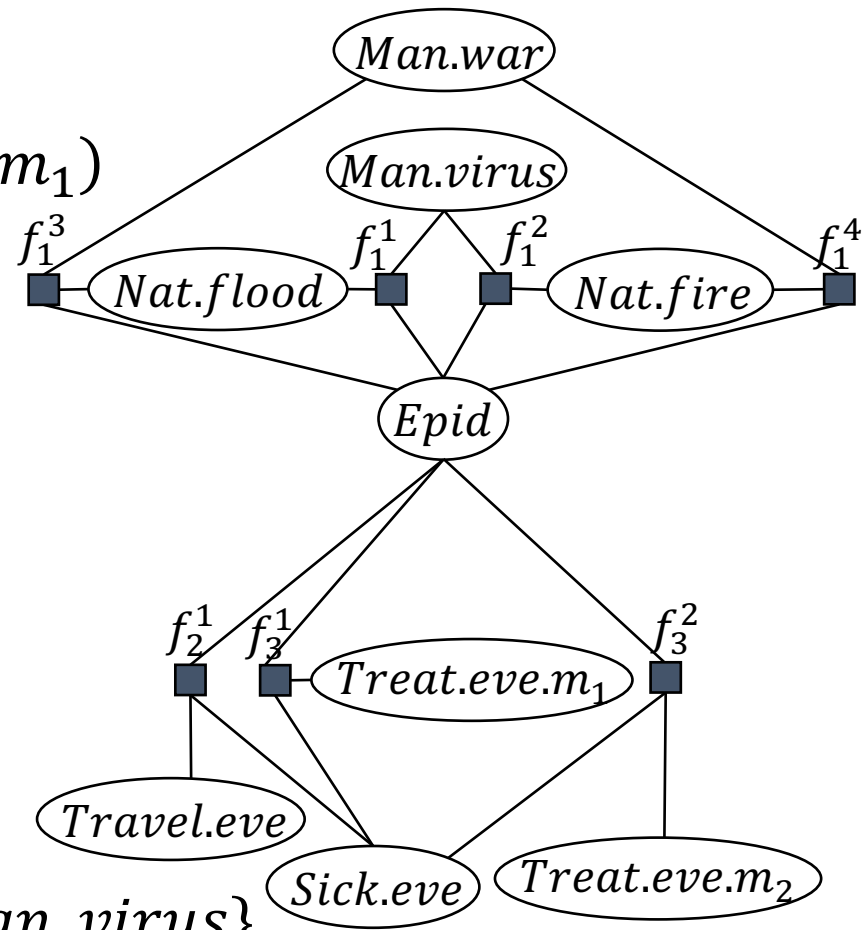


Query Answering (QA): Queries

- **Marginal** distribution
 - $P(\text{Sick.eve})$
 - $P(\text{Travel.eve}, \text{Treat.eve.m}_1)$
- **Conditional** distribution
 - $P(\text{Sick.eve} | \text{Epid})$
 - $P(\text{Epid} | \text{Sick.eve} = \text{true})$
- Most probable **assignment**

$$\operatorname{argmax}_{v \in r(R)} \sum_{v \in r(R)} P_F(v), R \subseteq rv(F)$$

- **MPE**: $R = rv(F)$
- **MAP**: $R = \{\text{Travel.eve}, \text{Man.virus}\}$

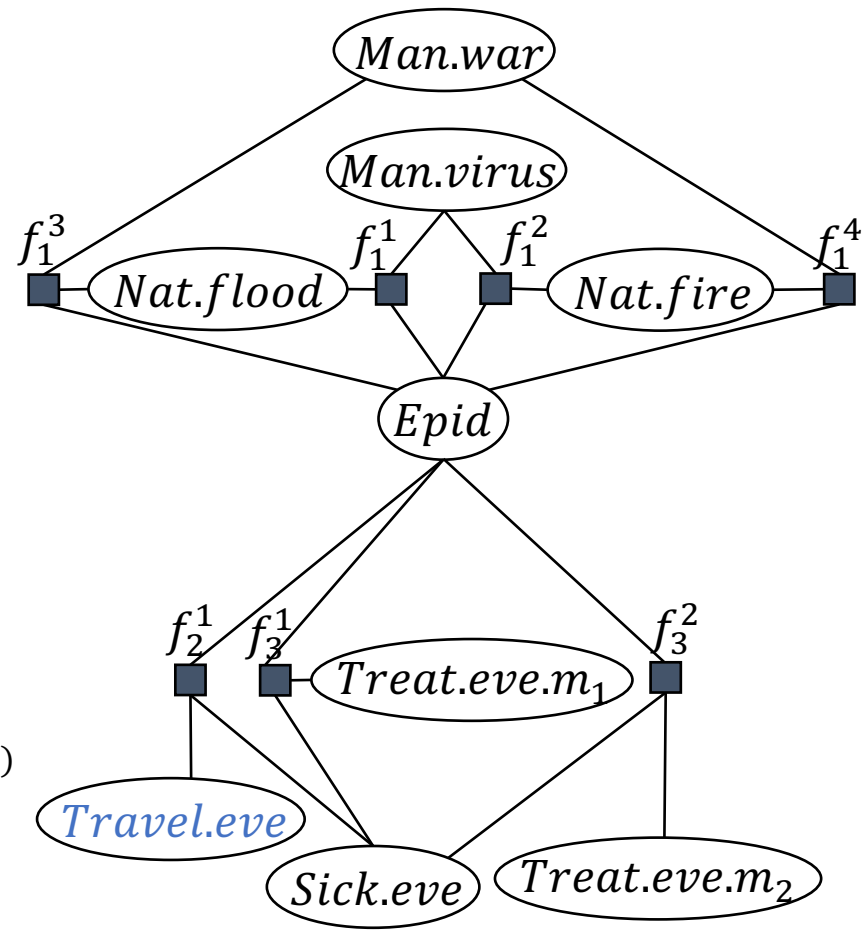


QA: Variable Elimination (VE)

- Eliminate all variables not appearing in query Pearl (1988), Zhang and Poole (1994), Dechter (1999)
 - Through **summing out**
- E.g., marginal
 - $P(\textit{Travel.eve})$

$$P(\textit{Travel.eve}) \propto$$

$$\sum_{e \in r(\textit{Epid})} \sum_{s \in r(\textit{Sick.eve})} \sum_{m_1 \in r(\textit{Treat.eve.m}_1)} \sum_{m_2 \in r(\textit{Treat.eve.m}_2)} \sum_{o \in r(\textit{Nat.flood})} \sum_{i \in r(\textit{Nat.fire})} \sum_{w \in r(\textit{Man.war})} \sum_{v \in r(\textit{Man.virus})} P_F$$



QA: Variable Elimination (VE)

Zhang and Poole (1994)

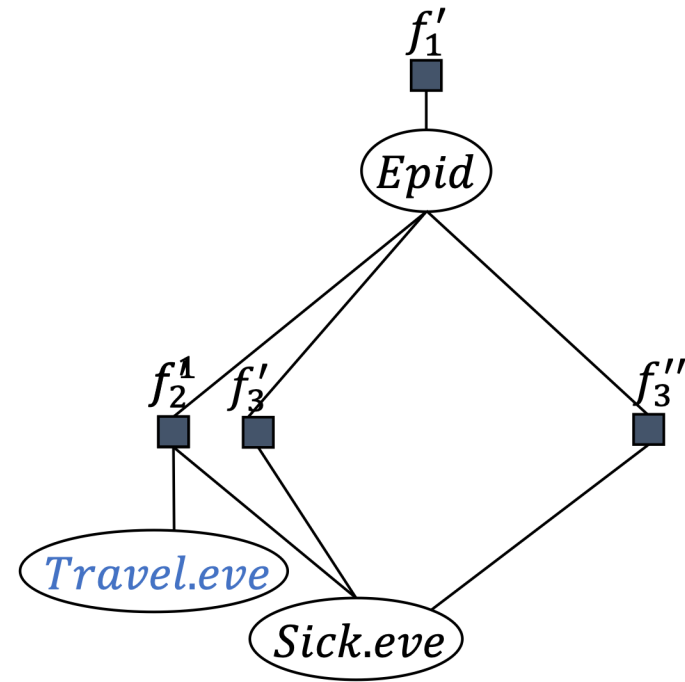
$$\begin{aligned}
 P(\textit{Travel.eve}) \propto & \sum_{e \in r(\textit{Epid})} \sum_{s \in r(\textit{Sick.eve})} f_2^1(\textit{Travel.eve}, e, s) \sum_{m_1 \in r(\textit{Treat.eve.m}_1)} f_3^1(e, s, m_1) \\
 & \sum_{m_2 \in r(\textit{Treat.eve.m}_2)} f_3^2(e, s, m_2) \\
 & \sum_{o \in r(\textit{Nat.flood})} \sum_{i \in r(\textit{Nat.fire})} \sum_{w \in r(\textit{Man.war})} f_1^1(o, w, e) f_1^2(i, w, e) \\
 & \sum_{v \in r(\textit{Man.virus})} f_1^3(o, i, e) f_1^4(i, v, e)
 \end{aligned}$$

QA: Variable Elimination (VE)

Zhang and Poole (1994)

$$\begin{aligned}
 P(\textit{Travel.eve}) &\propto \sum_{e \in r(\textit{Epid})} f'_1(e) \sum_{s \in r(\textit{Sick.eve})} f_2^1(\textit{Travel.eve}, e, s) f'_3(e, s) f_3''(e, s) \\
 &= \sum_{e \in r(\textit{Epid})} f'_1(e) \sum_{s \in r(\textit{Sick.eve})} f'(\textit{Travel.eve}, e, s)
 \end{aligned}$$

<i>Travel.eve</i>	<i>Epid</i>	<i>Sick.eve</i>	f'
<i>false</i>	<i>false</i>	<i>false</i>	50
<i>false</i>	<i>false</i>	<i>true</i>	10
<i>false</i>	<i>true</i>	<i>false</i>	34
<i>false</i>	<i>true</i>	<i>true</i>	16
<i>true</i>	<i>false</i>	<i>false</i>	54
<i>true</i>	<i>false</i>	<i>true</i>	36
<i>true</i>	<i>true</i>	<i>false</i>	12
<i>true</i>	<i>true</i>	<i>true</i>	69

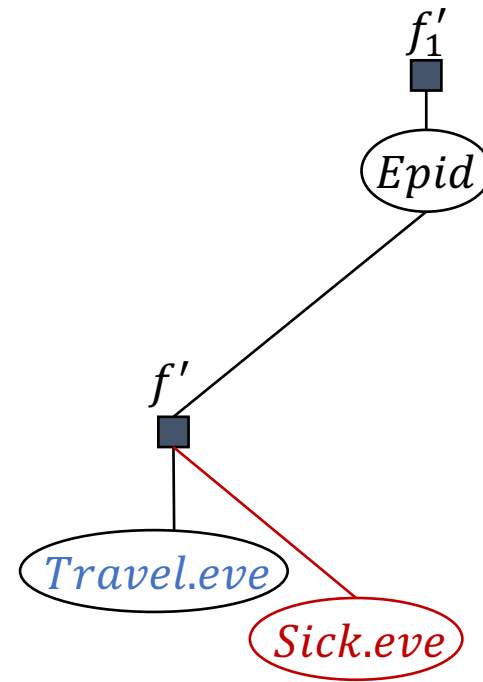


QA: Variable Elimination (VE)

Zhang and Poole (1994)

$$\begin{aligned}
 P(\textit{Travel.eve}) &\propto \sum_{e \in r(\textit{Epid})} f'_1(e) \sum_{s \in r(\textit{Sick.eve})} f_2^1(\textit{Travel.eve}, e, s) f'_3(e, s) f_3''(e, s) \\
 &= \sum_{e \in r(\textit{Epid})} f'_1(e) \sum_{s \in r(\textit{Sick.eve})} f'(\textit{Travel.eve}, e, s)
 \end{aligned}$$

<i>Travel.eve</i>	<i>Epid</i>	<i>Sick.eve</i>	f'	Σ
false	false	false	50	60
false	false	true	10	
false	true	false	34	50
false	true	true	16	
true	false	false	54	90
true	false	true	36	
true	true	false	12	81
true	true	true	69	

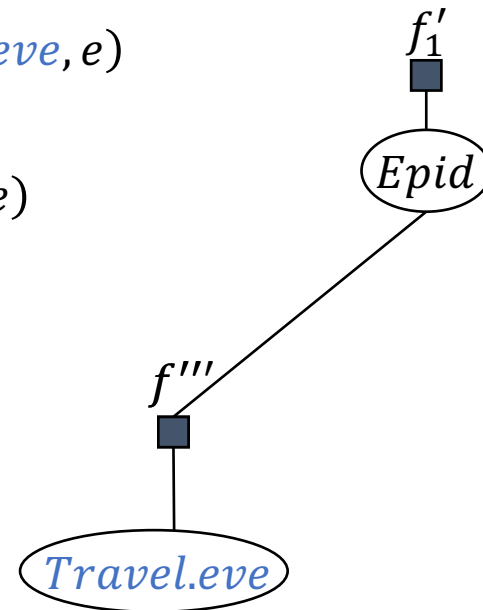


QA: Variable Elimination (VE)

Zhang and Poole (1994)

$$\begin{aligned}
 P(\textit{Travel.eve}) &\propto \sum_{e \in r(\textit{Epid})} f'_1(e) \sum_{s \in r(\textit{Sick.eve})} f_2^1(\textit{Travel.eve}, e, s) f'_3(e, s) f_3''(e, s) \\
 &= \sum_{e \in r(\textit{Epid})} f'_1(e) \sum_{s \in r(\textit{Sick.eve})} f'(\textit{Travel.eve}, e, s) \\
 &= \sum_{e \in r(\textit{Epid})} f'_1(e) f''(\textit{Travel.eve}, e) \\
 &= \sum_{e \in r(\textit{Epid})} f'''(\textit{Travel.eve}, e)
 \end{aligned}$$

<i>Travel.eve</i>	<i>Epid</i>	f'''
<i>false</i>	<i>false</i>	90
<i>false</i>	<i>true</i>	100
<i>true</i>	<i>false</i>	135
<i>true</i>	<i>true</i>	162

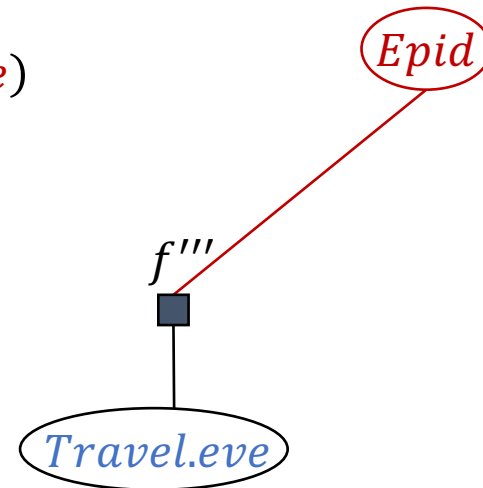


QA: Variable Elimination (VE)

Zhang and Poole (1994)

$$\begin{aligned}
 P(\textit{Travel.eve}) &\propto \sum_{e \in r(\textit{Epid})} f'_1(e) \sum_{s \in r(\textit{Sick.eve})} f_2^1(\textit{Travel.eve}, e, s) f'_3(e, s) f''_3(e, s) \\
 &= \sum_{e \in r(\textit{Epid})} f'_1(e) \sum_{s \in r(\textit{Sick.eve})} f'(\textit{Travel.eve}, e, s) \\
 &= \sum_{e \in r(\textit{Epid})} f'_1(e) f''(\textit{Travel.eve}, e) \\
 &= \sum_{e \in r(\textit{Epid})} f'''(\textit{Travel.eve}, e)
 \end{aligned}$$

<i>Travel.eve</i>	<i>Epid</i>	f'''	Σ
<i>false</i>	<i>false</i>	90	190
<i>false</i>	<i>true</i>	100	
<i>true</i>	<i>false</i>	135	297
<i>true</i>	<i>true</i>	162	

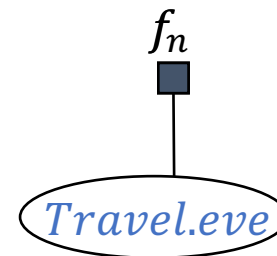


QA: Variable Elimination (VE)

Zhang and Poole (1994)

$$\begin{aligned}
 P(\textit{Travel.eve}) &\propto \sum_{e \in r(\textit{Epid})} f'_1(e) \sum_{s \in r(\textit{Sick.eve})} f_2^1(\textit{Travel.eve}, e, s) f'_3(e, s) f''_3(e, s) \\
 &= \sum_{e \in r(\textit{Epid})} f'_1(e) \sum_{s \in r(\textit{Sick.eve})} f'(\textit{Travel.eve}, e, s) \\
 &= \sum_{e \in r(\textit{Epid})} f'_1(e) f''(\textit{Travel.eve}, e) \\
 &= \sum_{e \in r(\textit{Epid})} f'''(\textit{Travel.eve}, e) = f(\textit{Travel.eve}) = f_n(\textit{Travel.eve})
 \end{aligned}$$

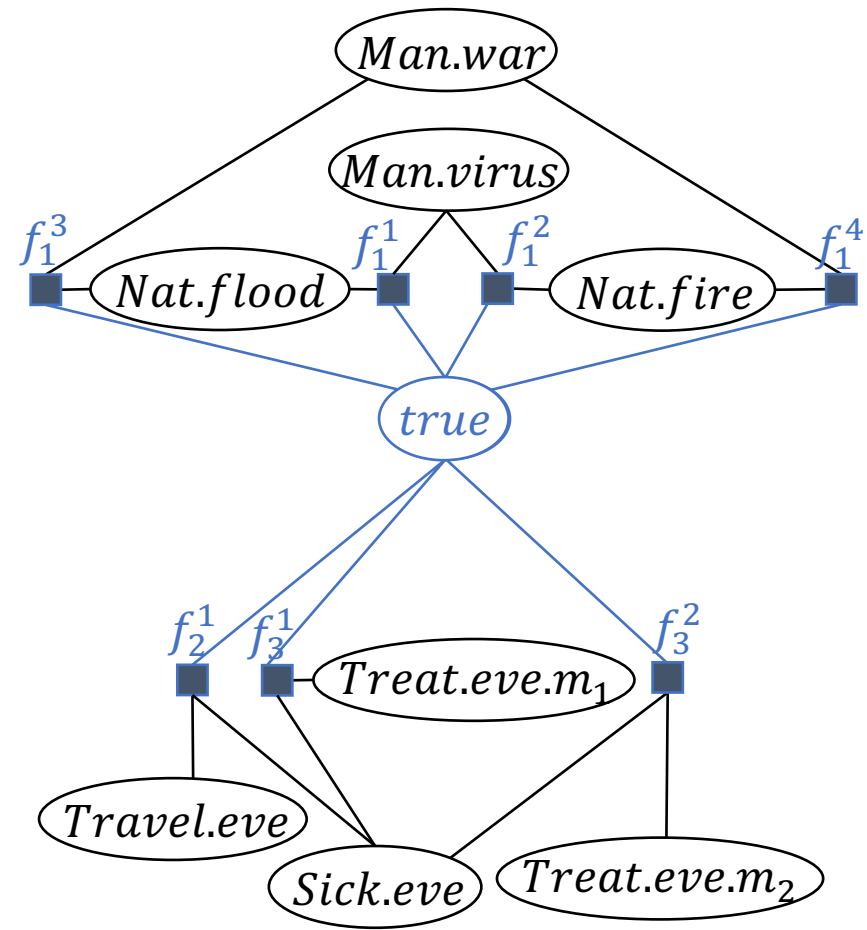
<i>Travel.eve</i>	<i>f</i>	<i>Travel.eve</i>	<i>f_n</i>
<i>false</i>	190	<i>false</i>	0.39
<i>true</i>	297	<i>true</i>	0.61



QA: Variable Elimination (VE)

Zhang and Poole (1994)

- Eliminate all variables not appearing in query
 - Through **summing out**
- E.g., conditional
 - $P(\text{Travel.eve} | \text{Epid} = \text{true})$
 - **Absorb** $\text{Epid} = \text{true}$ in all factors
 - Sum out remaining variables

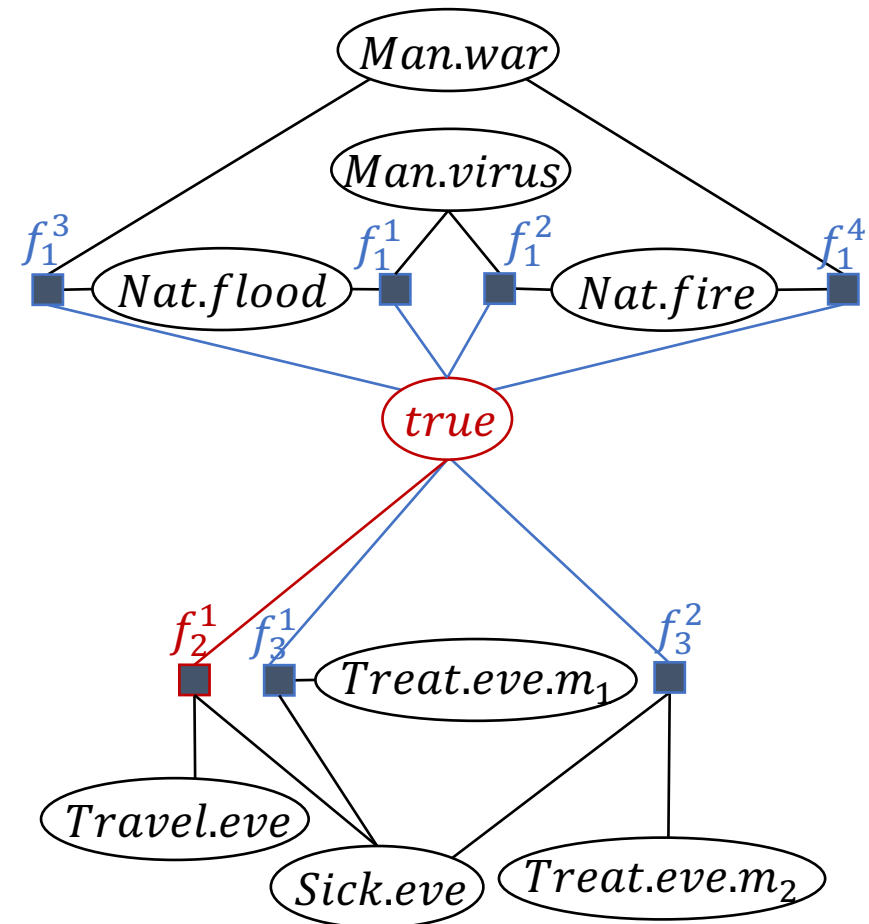


QA: Evidence Absorption

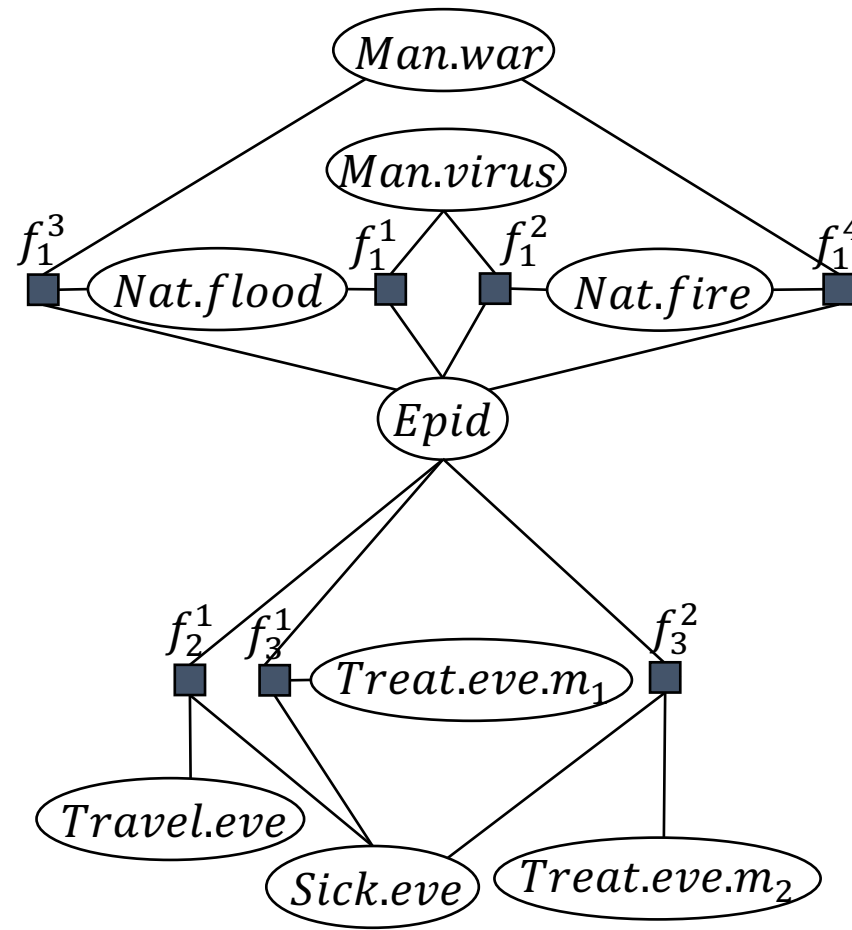
Zhang and Poole (1994)

- Absorb *Epid* = *true* in all factors, e.g., f_2^1
- Possibly eliminate variable

<i>Travel.eve</i>	<i>Sick.eve</i>	f_2^1	<i>e</i>	f_2^1
<i>false</i>	<i>false</i>	4		4
<i>false</i>	<i>true</i>	6		6
<i>true</i>	<i>false</i>	2		2
<i>true</i>	<i>true</i>	9		9
<i>true</i>	<i>false</i>	<i>false</i>		4 0
<i>true</i>	<i>false</i>	<i>true</i>		6 0
<i>true</i>	<i>true</i>	<i>false</i>		2
<i>true</i>	<i>true</i>	<i>true</i>		9

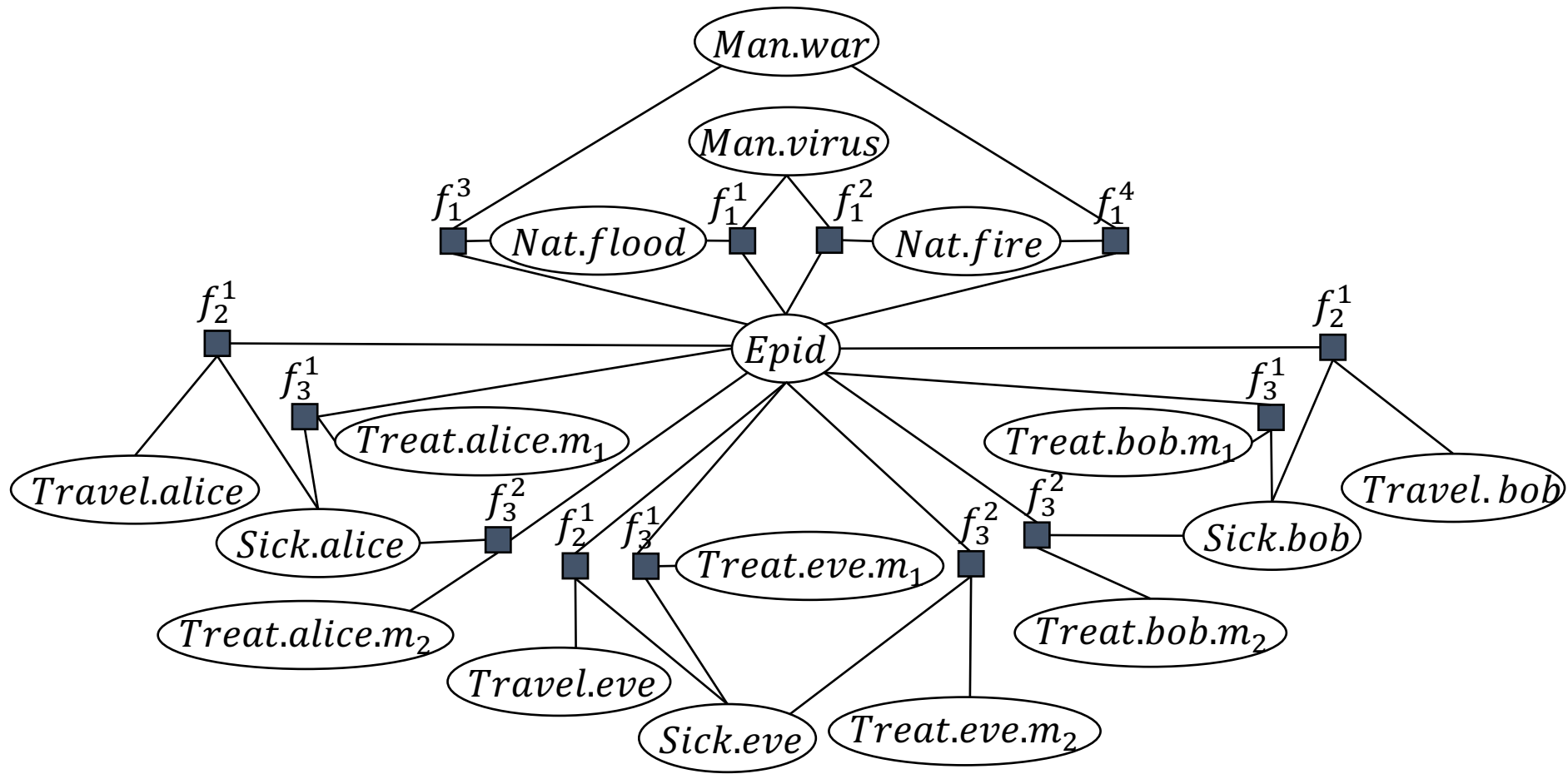


Problem: Models Explode



$7 \cdot 2^3 = 56$ entries in 7 factors, 9 variables

Problem: Models Explode

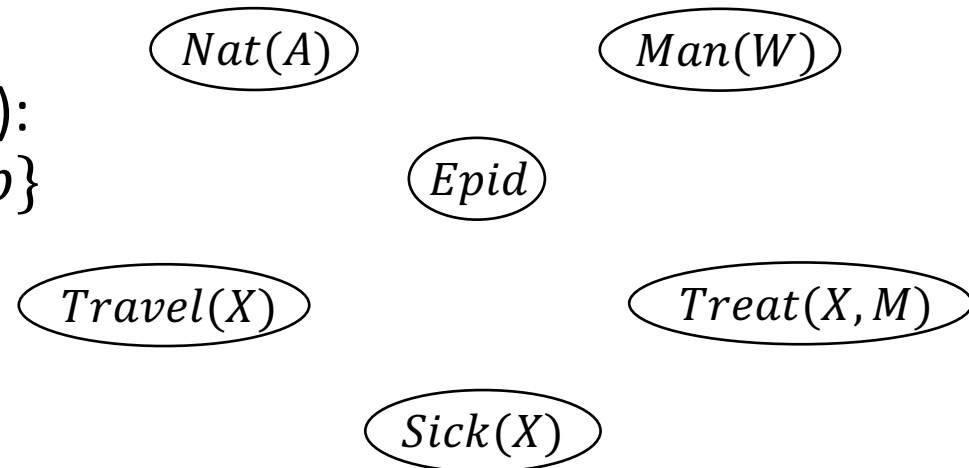


$13 \cdot 2^3 = 104$ entries in 13 factors, 17 variables

Solution: Lifting

Poole (2003)

- Parameterised random variables = PRV
 - With logical variables
 - E.g., X
 - Possible values (domain):
 $\mathcal{D}(X) = \{alice, eve, bob\}$
- Inputs to factors!



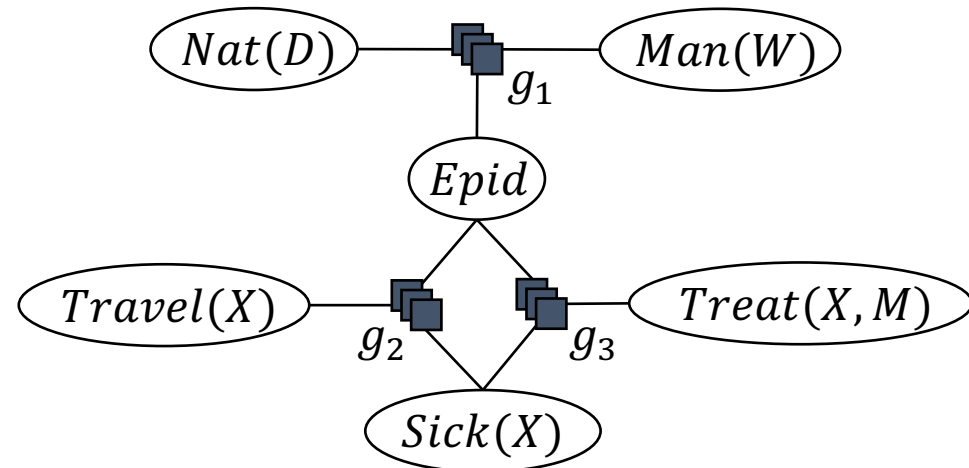
Solution: Lifting

Poole (2003)

- Factors with PRVs = **parfactors**
 - (Graphical) Model G
 - E.g., g_2

<i>Travel(X)</i>	<i>Epid</i>	<i>Sick(X)</i>	g_2
<i>false</i>	<i>false</i>	<i>false</i>	5
<i>false</i>	<i>false</i>	<i>true</i>	0
<i>false</i>	<i>true</i>	<i>false</i>	4
<i>false</i>	<i>true</i>	<i>true</i>	6
<i>true</i>	<i>false</i>	<i>false</i>	4
<i>true</i>	<i>false</i>	<i>true</i>	6
<i>true</i>	<i>true</i>	<i>false</i>	2
<i>true</i>	<i>true</i>	<i>true</i>	9

Sparse encoding!



$3 \cdot 2^3 = 24$ entries in 3 parfactors, 6 PRVs

Solution: Lifting

Poole (2003)

- **Grounding**

- E.g., $gr(g_2) = \{f_2^1, f_2^2, f_2^3\}$

		<i>Travel(eve)</i>	<i>Epid</i>	<i>Sick(eve)</i>	f_2^1				
<i>Travel(X)</i>	<i>Epid</i>					<i>Travel(bob)</i>	<i>Epid</i>	<i>Sick(bob)</i>	f_2^3
<i>Travel(alice)</i>	<i>Epid</i>	false	false	false	5	<i>Travel(bob)</i>	<i>Epid</i>	<i>Sick(bob)</i>	f_2^3
<i>false</i>	<i>false</i>	false	false	true	0	<i>false</i>	<i>false</i>	<i>false</i>	5
<i>false</i>	<i>false</i>	false	true	false	4	<i>false</i>	<i>false</i>	<i>true</i>	0
<i>false</i>	<i>false</i>	false	true	true	6	<i>false</i>	<i>true</i>	<i>false</i>	4
<i>false</i>	<i>true</i>	true	false	false	4	<i>false</i>	<i>true</i>	<i>true</i>	6
<i>false</i>	<i>true</i>	true	false	true	6	<i>false</i>	<i>false</i>	<i>false</i>	4
<i>true</i>	<i>false</i>	true	true	false	2	<i>false</i>	<i>false</i>	<i>true</i>	6
<i>true</i>	<i>false</i>	true	true	true	9	<i>false</i>	<i>true</i>	<i>false</i>	2
<i>true</i>	<i>true</i>	<i>false</i>	2			<i>true</i>	<i>true</i>	<i>true</i>	9
<i>true</i>	<i>true</i>	<i>true</i>	9						

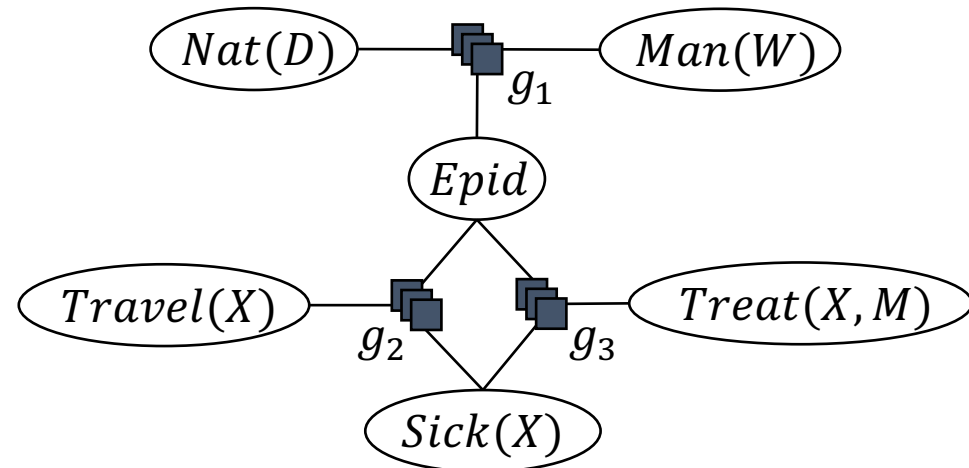
Solution: Lifting

Poole (2003)

- Joint probability distribution P_G by grounding

$$P_G = \frac{1}{Z} \prod_{f \in \text{gr}(G)} f$$

$$Z = \sum_{v \in r(\text{rv}(\text{gr}(G)))} \prod_{f \in \text{gr}(G)} f_i(\pi_{rv(f_i)}(v))$$



QA: Queries

- Marginal distribution

- $P(\text{Sick}(\text{eve}))$
- $P(\text{Travel}(\text{eve},) \text{ Treat}(\text{eve}, m_1))$

Avoid groundings!

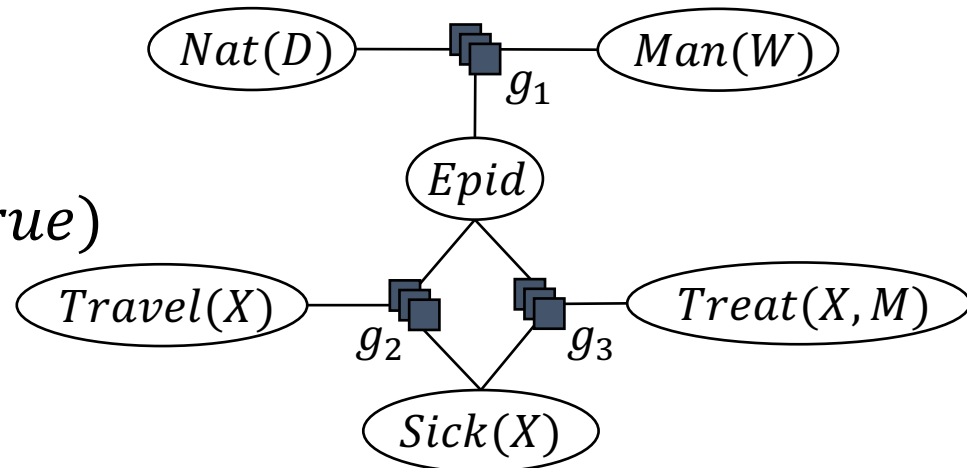
- Conditional distribution

- $P(\text{Sick}(\text{eve}) | \text{Epid})$
- $P(\text{Epid} | \text{Sick}(\text{eve}) = \text{true})$

- Most probable assignment

$$\operatorname{argmax}_{v \in r(R)} \sum_{v \in r(R)} P_F(v), R \subseteq rv(gr(G))$$

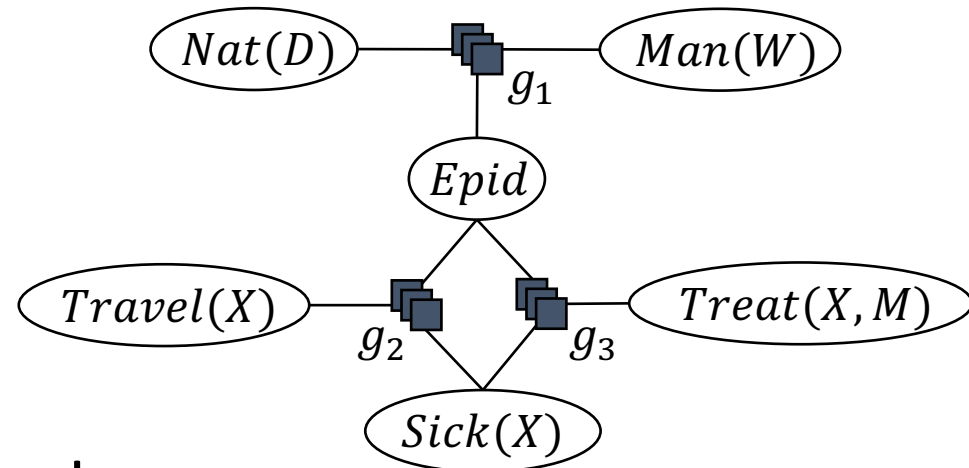
- MPE: $R = rv(gr(G))$
- MAP: $R = \{\text{Travel}(\text{eve}), \text{Man}(\text{virus})\}$



QA: Lifted VE (LVE)

Poole (2003), de Salvo Braz et al. (2005, 2006),
Milch et al. (2008), Taghipour et al. (2012, 2013)

- Eliminate all variables not appearing in query
- Lifted summing out
 - Sum out **representative** instance
 - Exponentiate result for **isomorphic** instances
- Preconditions exist!
- Count conversion (not part of this tutorial)

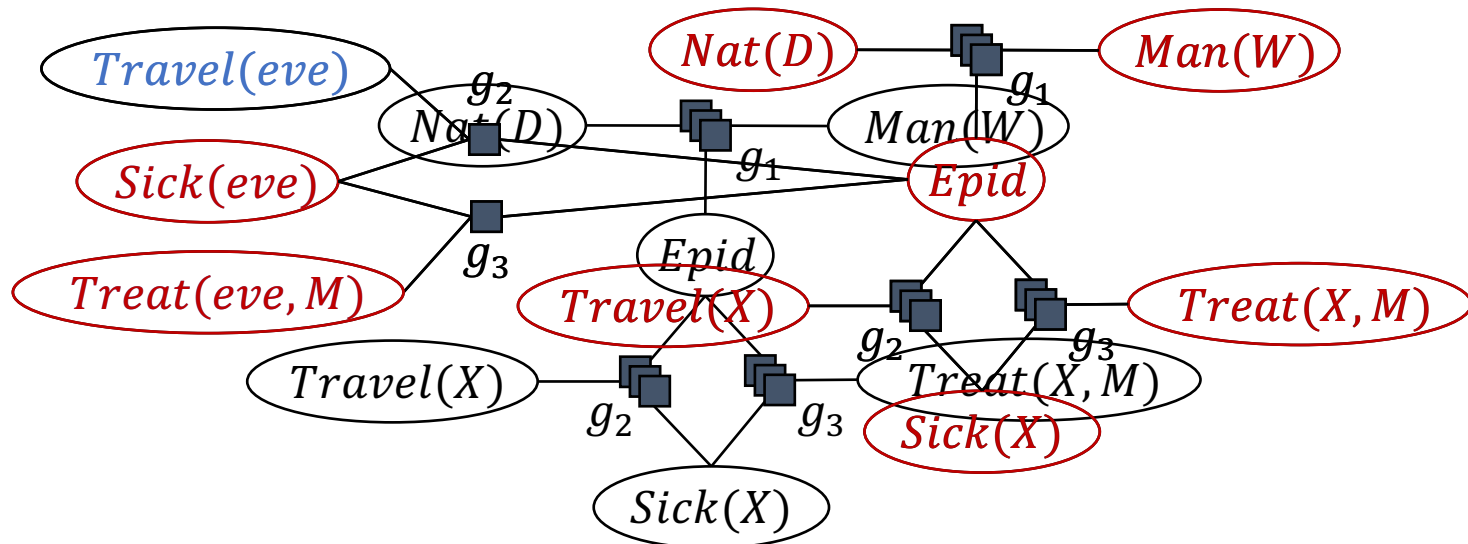


Avoid groundings!

QA: LVE in Detail

Poole (2003), de Salvo Braz et al. (2005, 2006),
Milch et al. (2008), Taghipour et al. (2013, 2013a)

- E.g., marginal
 - $P(\textit{Travel}(\textit{eve}))$
 - Split w.r.t. $\textit{Travel}(\textit{eve})$
 - Eliminate all **non-query variables**
 - Normalise



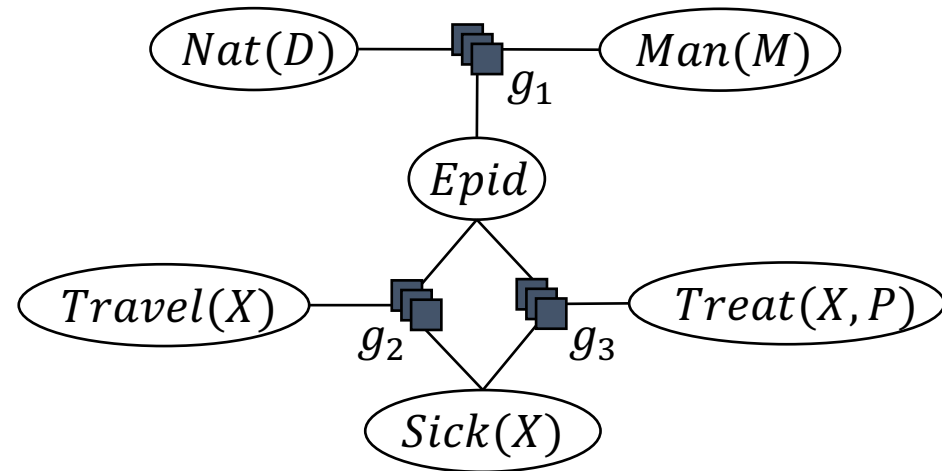
Problem: Many Queries

- Set of queries

- $P(\text{Travel}(\text{eve}))$
- $P(\text{Sick}(\text{bob}))$
- $P(\text{Treat}(\text{eve}, m_1))$
- $P(\text{Epid})$
- $P(\text{Nat}(\text{flood}))$
- $P(\text{Man}(\text{virus}))$
- Combinations of variables

- Under evidence

- $\text{Sick}(X') = \text{true}$
- $X' \in \{\text{alice}, \text{eve}\}$

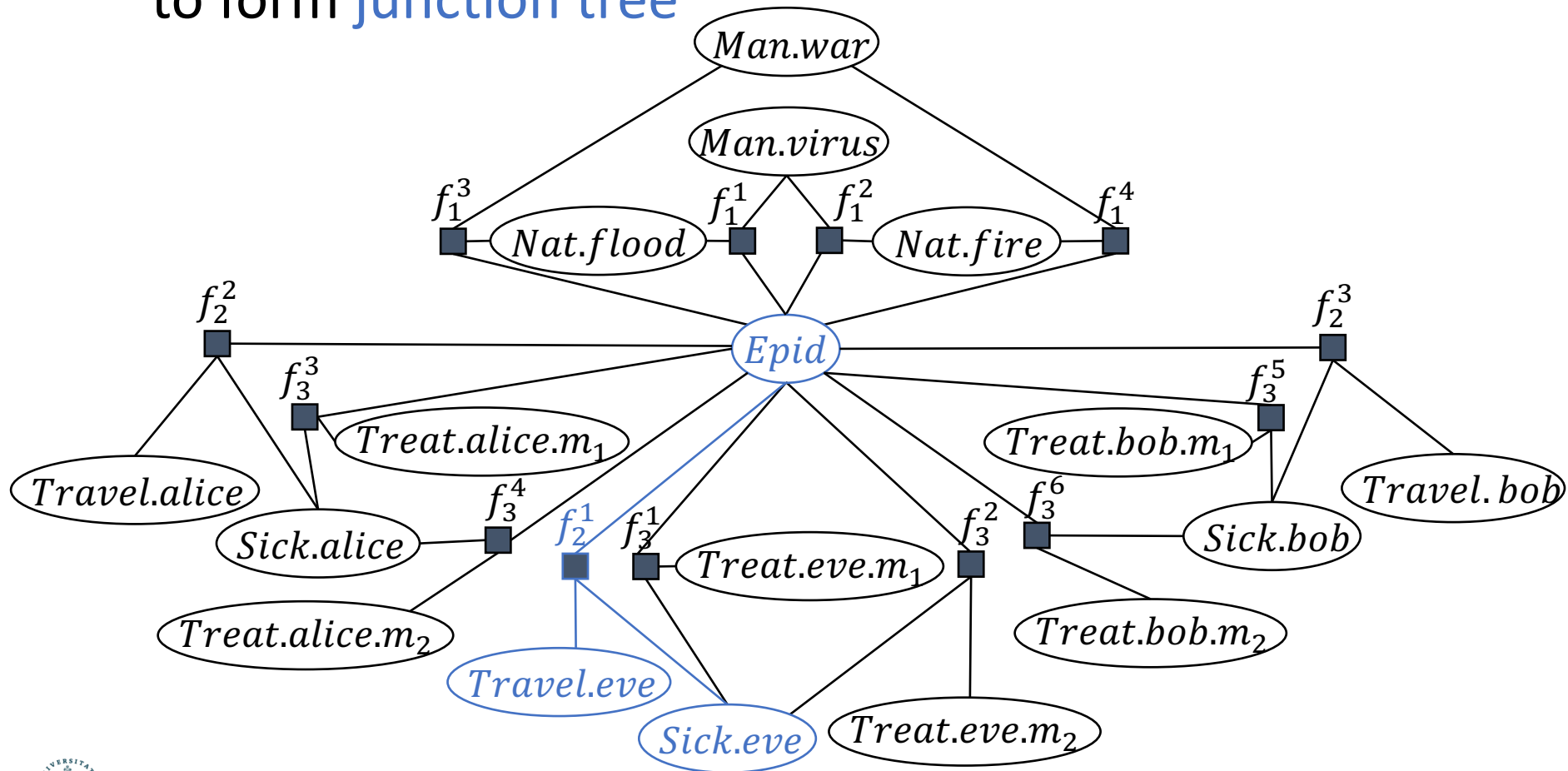


- (L)VE starts with complete model for QA

Solution: Junction Tree Algorithm

Lauritzen and Spiegelhalter (1988), Shafer and Shenoy (1989), Jensen et al. (1990)

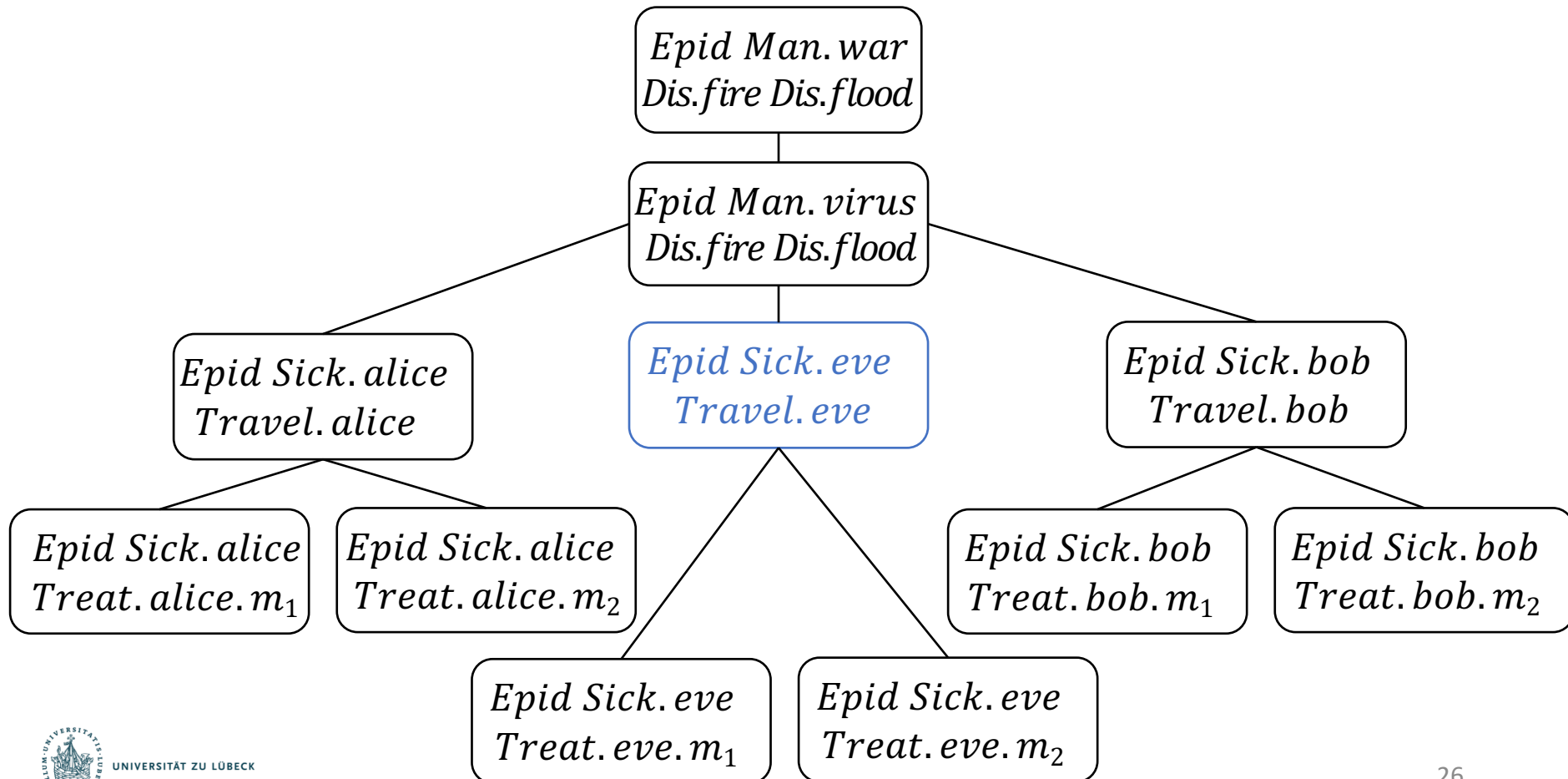
- Identify clusters in model to form **junction tree**



Junction Tree: Data Structure

Lauritzen and Spiegelhalter (1988), Shafer and Shenoy (1989), Jensen et al. (1990)

- **Clusters**: sets of variables from underlying model



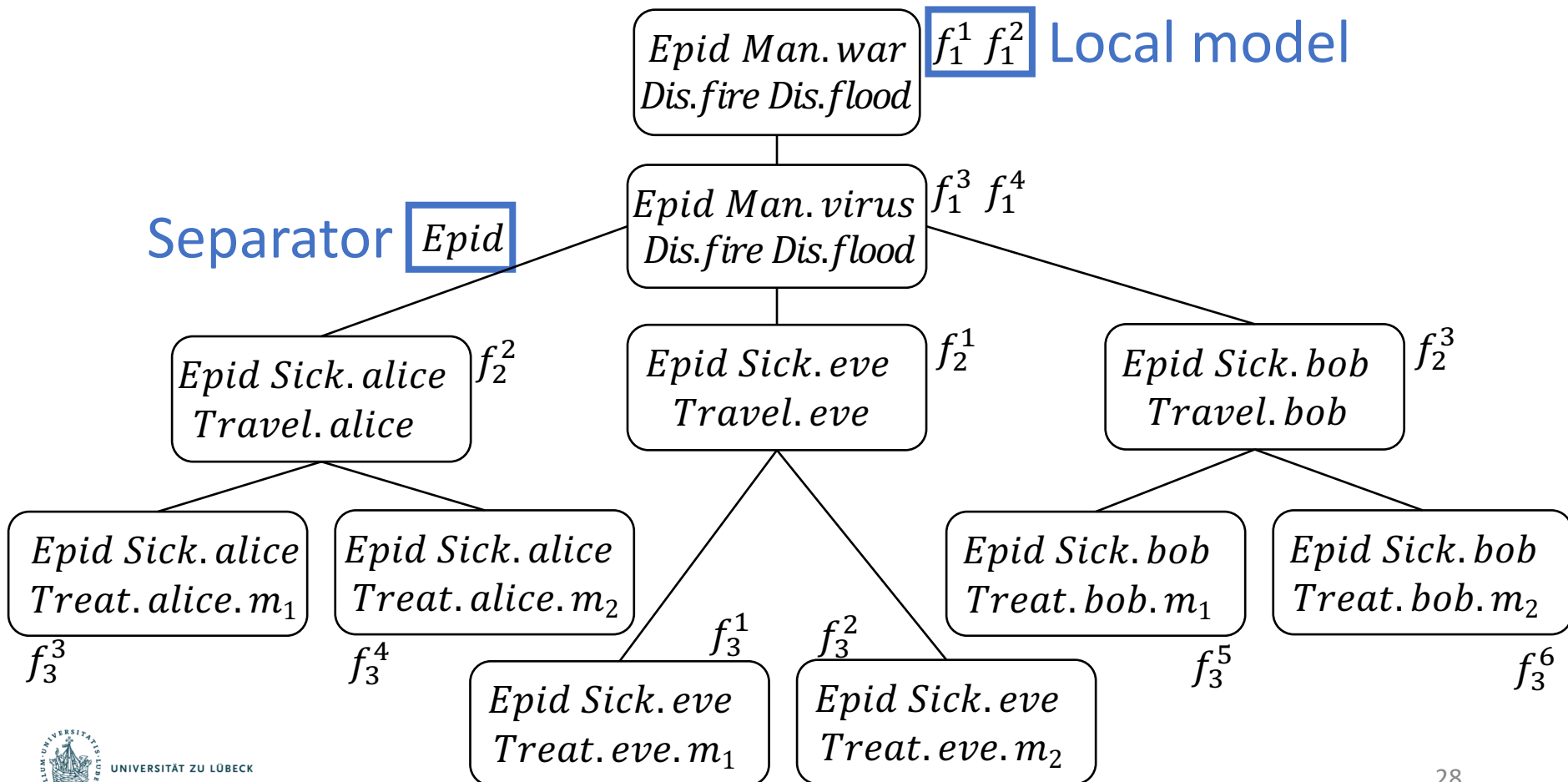
Junction Tree: Definition

Lauritzen and Spiegelhalter (1988)

- Junction tree $J = (V, E)$ for model F
 - V set of nodes, a node = cluster C
 - E set of edges
 1. A cluster C_i is a set of variables from F
 2. For every factor f in F , its arguments appear in some cluster C_i
 3. If a variable from F appears in clusters C_i and C_j , it must appear in every cluster C_k on the path between nodes i and j .
- Each cluster has a **local model** F_i : set of factors
 - factor arguments subset of cluster
 - $\bigcup_{i \in V} F_i = F$ (F_i partition F)
- **Separator**: shared variables of edges $(i, j) \in E$

Junction Tree: Components

- Clusters sufficient for QA after some preprocessing



Junction Tree: Message Passing

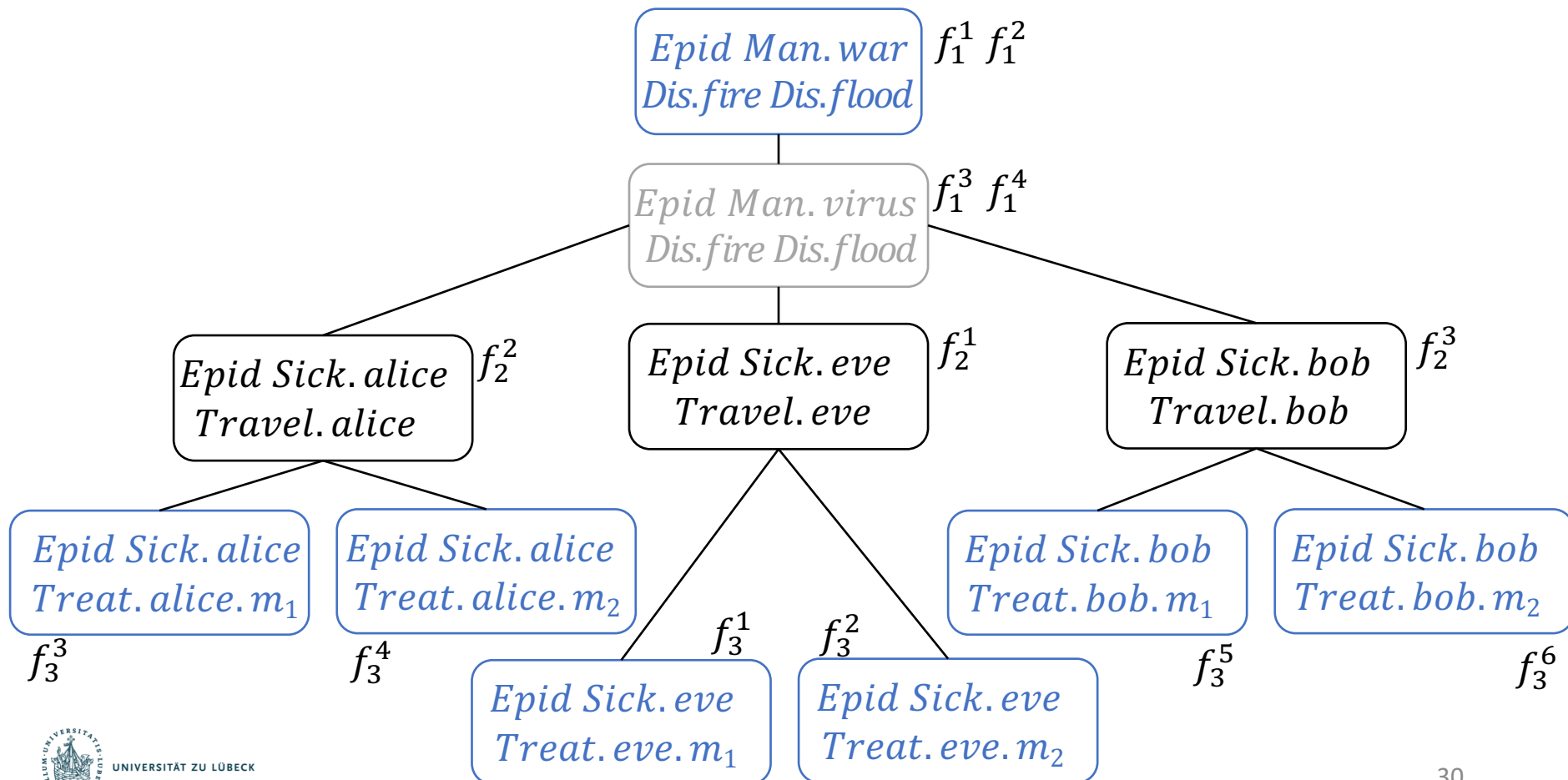
Lauritzen and Spiegelhalter (1988), Shenoy and Shafer (1990)

- Distribute local information by messages
 1. If a cluster received messages from all neighbours but one, it sends message to remaining neighbour
 2. If a cluster received all messages, it send messages to all neighbours that have not received a message yet
 - **Message:** VE on local model and other messages with query over separator
- Local computations correctness:
 - Valid junction tree (w.r.t. properties)
 - **Combination & marginalisation**
(in the form of multiplication & summing out)
- QA on local models and messages
 - Use cluster that contains query variable

Junction Tree: Messages

Shafer and Shenoy (1989)

- From **periphery** to **centre** and back



Junction Tree: Messages

Shafer and Shenoy (1989)

- Message: eliminate all **non-separator variables**
 - E.g., cluster with f_3^3 in local model

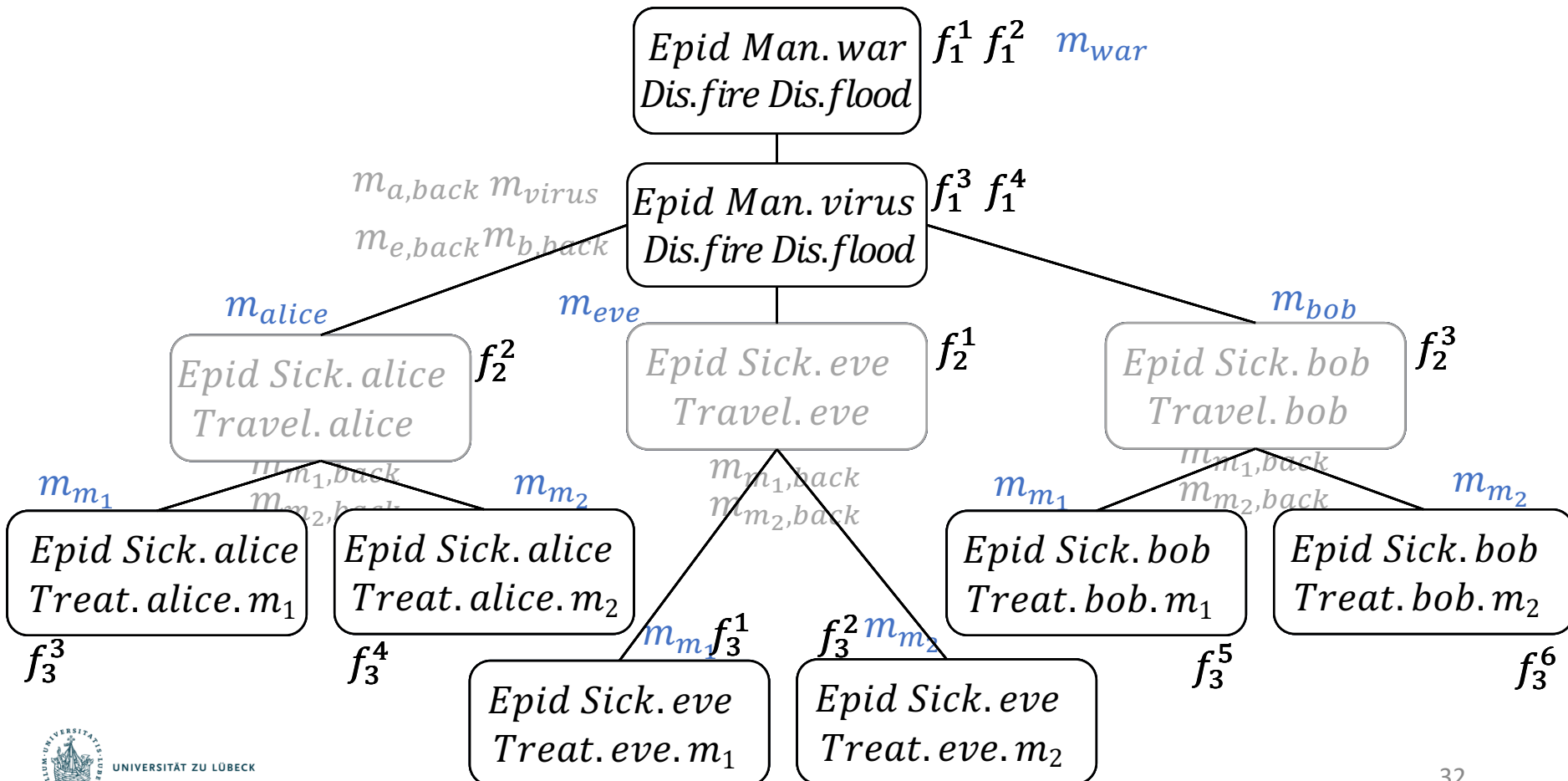
<i>Sick.alice</i>	<i>Epid</i>	Σ	<i>Treat.alice.m₁</i>	f_3^3	Σ
<i>false</i>	<i>false</i>	11	<i>false</i>	9	11
<i>false</i>	<i>true</i>	7	<i>true</i>	2	
<i>true</i>	<i>false</i>	8	<i>false</i>	3	7
<i>true</i>	<i>true</i>	15	<i>true</i>	4	
<i>true</i>	<i>false</i>		<i>false</i>	1	8
<i>true</i>	<i>false</i>		<i>true</i>	7	
<i>true</i>	<i>true</i>		<i>false</i>	6	15
<i>true</i>	<i>true</i>		<i>true</i>	9	



Junction Tree: Messages

Shafer and Shenoy (1989)

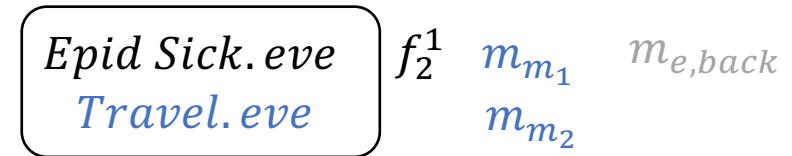
- From **periphery** to centre and back



QA: Query Answering with JT

Shafer and Shenoy (1989)

- Eliminate all variables not appearing in query
 - Through summing out
- E.g., marginal
 - $P(\textit{Travel.eve})$

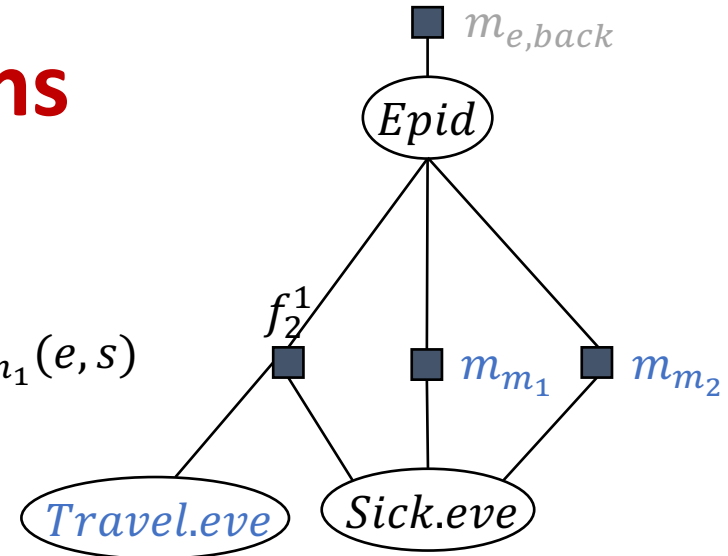


$$P(\textit{Travel.eve}) \propto$$

$$\sum_{e \in r(\textit{Epid})} m_{e,back}(e)$$

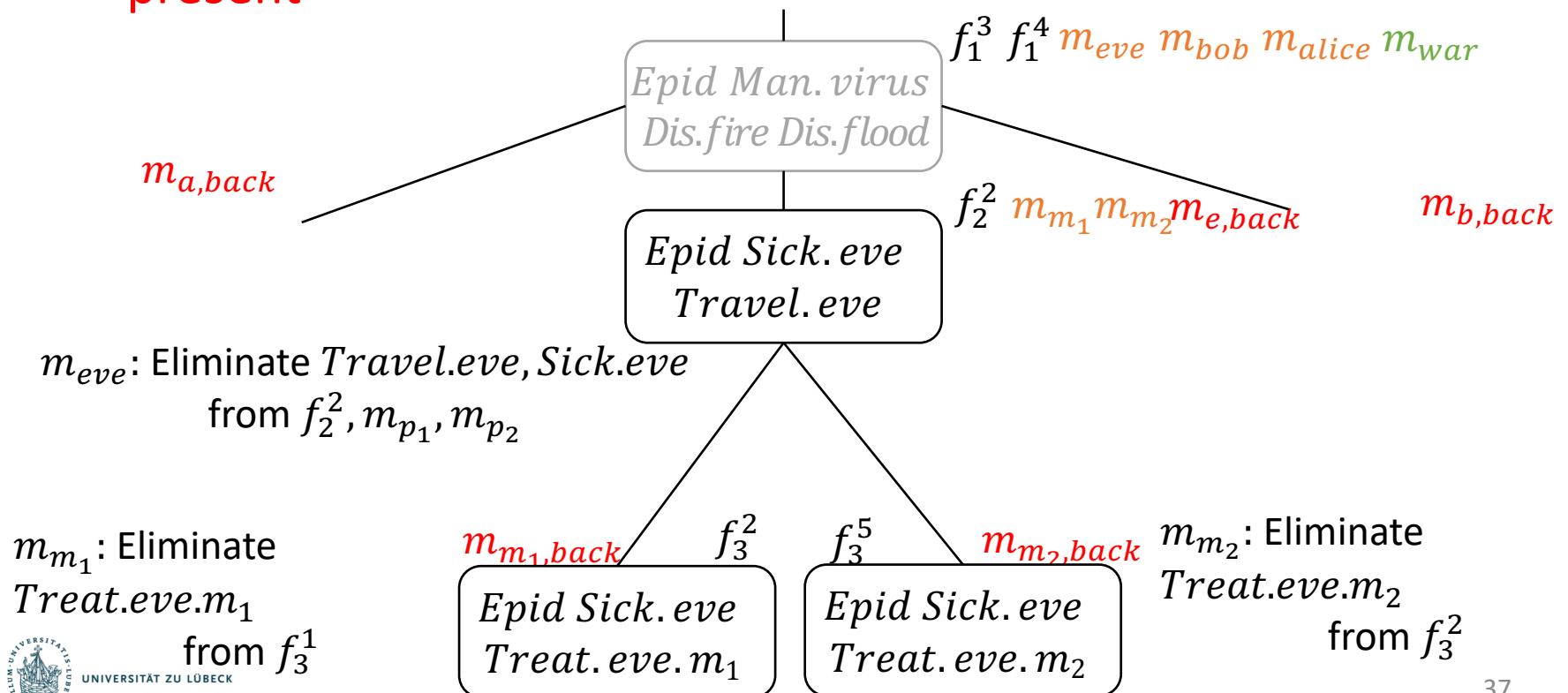
$$\sum_{s \in r(\textit{Sick.eve})} f_2^1(\textit{Travel.eve}, e, s) m_{m_1}(e, s) m_{m_1}(e, s)$$

2 sums



Junction Tree: Symmetry → Inefficiency

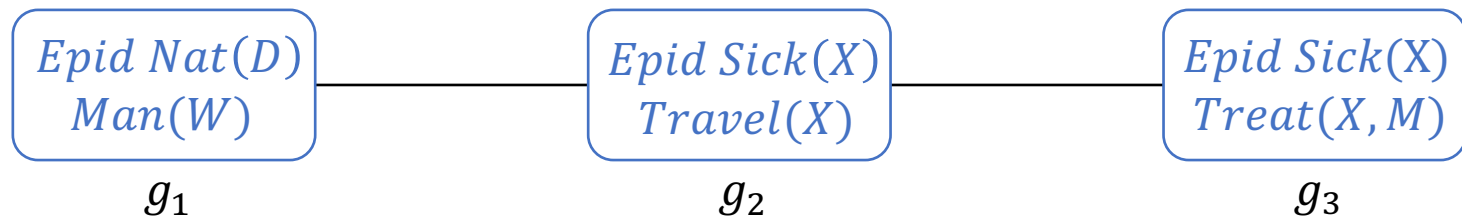
- Identical messages incoming
- Information already present
- Calculating identical messages + sending information partially present



First-order Junction Tree: FO jtree

Braun and Möller (2016)

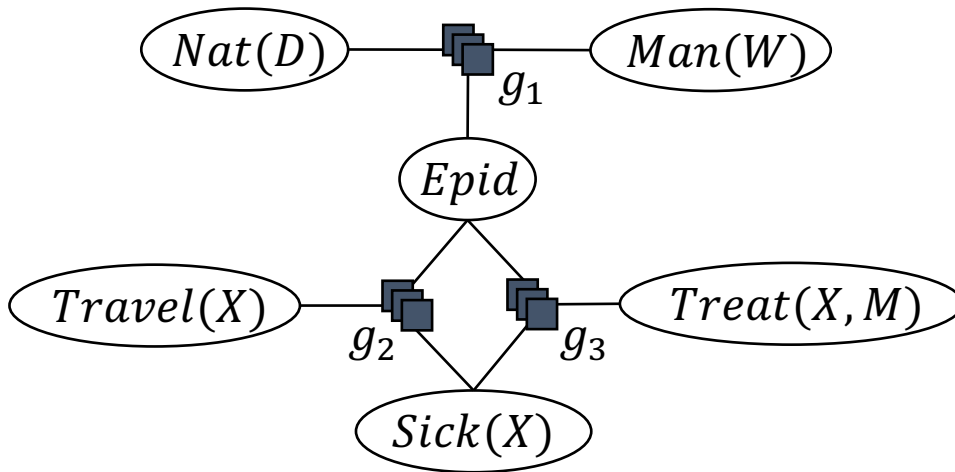
- Parameterised clusters = **parclusters**
 - Nodes of FO jtree
 - Shared PRVs between neighbours = separators
 - Parclusters have local models of parfactors
- FO jtree for parfactor model G :
 - Analogous definition, junction tree properties apply
 - Message passing, QA analogous



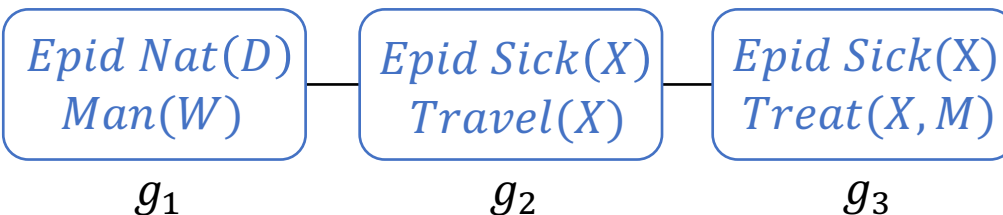
Sparse encoding!

Lifted Junction Tree Algorithm: LJT

Braun and Möller (2017)



Queries on grounded PRVs, e.g.,
Travel(eve), Treat(eve, p₁), Epid



- **Input**

- Model G
- Evidence E
- Queries Q

- **Algorithm**

1. Build FO jtree J for G
2. Enter evidence E into J
3. Pass messages in J
 - Inbound
 - Outbound
4. Answer queries Q

LJT: Example Input

Braun and Möller (2017)

- Model $G = \{g_i\}_{i=1}^3$
 - $g_1(Epid, Nat(D), Man(W))$
 - $g_2(Travel(X), Epid, Sick(X))$
 - $g_3(Epid, Sick(X), Treat(X, M))$→ Including function specification
- Evidence $E = \{Sick(alice) = true, Sick(eve) = true\}$
- Queries $Q = \{Travel(eve), Epid\}$
- Algorithm
 1. Build FO jtree J for G

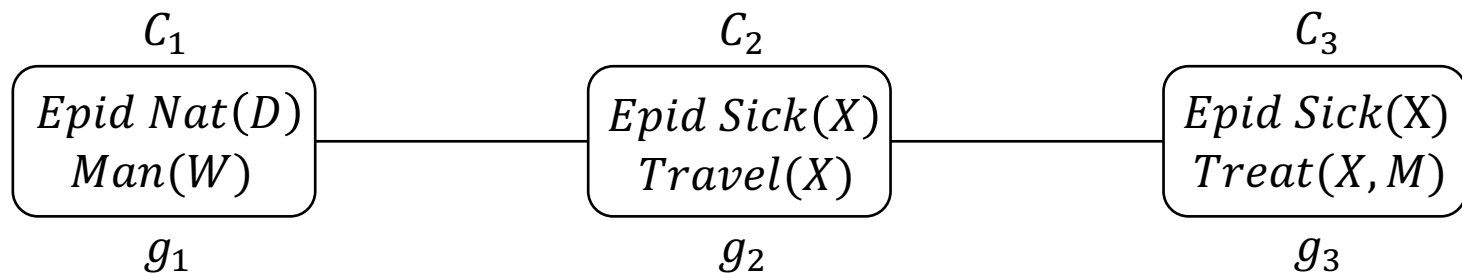
FO Jtree Construction

- Propositional junction tree construction
 - Triangulation, compute maximum spanning tree, ...
 - Hypergraph partitioning
 - Decomposition tree (dtree), clusters, ...
- First-order: logical variables Taghipour et al. (2013b)
 - First-order decomposition trees (**FO dtrees**)
 - FO dtrees have node properties (cutset, context, **cluster**)
 - (FO) dtree + clusters = (FO) jtree
 - Heuristic to build an FO dtree
(logical variables guide the construction)

Lifted Junction Tree Algorithm: LJT

Braun and Möller (2017)

- Input
 - Model G
 - Evidence E
 - Queries Q
- Algorithm
 1. Build FO jtree J for G



2. Enter evidence E into J

LJT: Enter Evidence

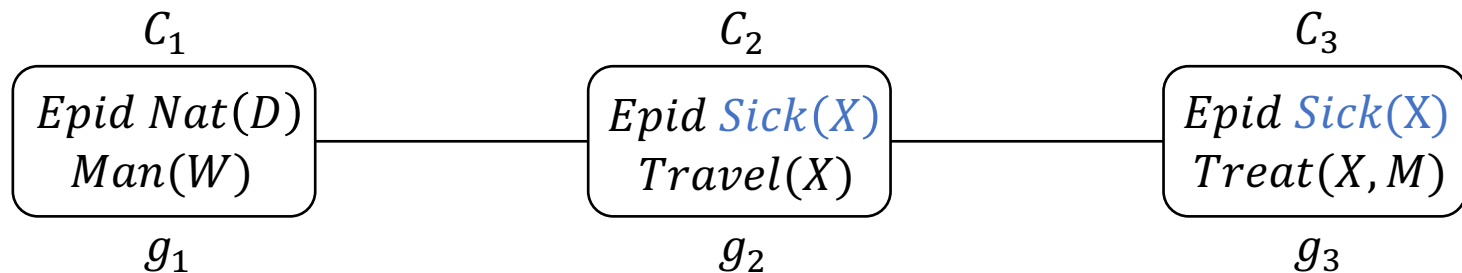
- Evidence as a set of events
 - $E = \{Sick(eve) = true, Sick(alice) = true\}$
- Evidence as a parfactor
 - $g_E(Sick(X'))$
 - $X' \in \{eve, alice\}$
 - Function specification

$Sick(X')$	g_E
<i>false</i>	0
<i>true</i>	1

- At every parcluster that contains evidence variables, enter evidence

LJT: Enter Evidence

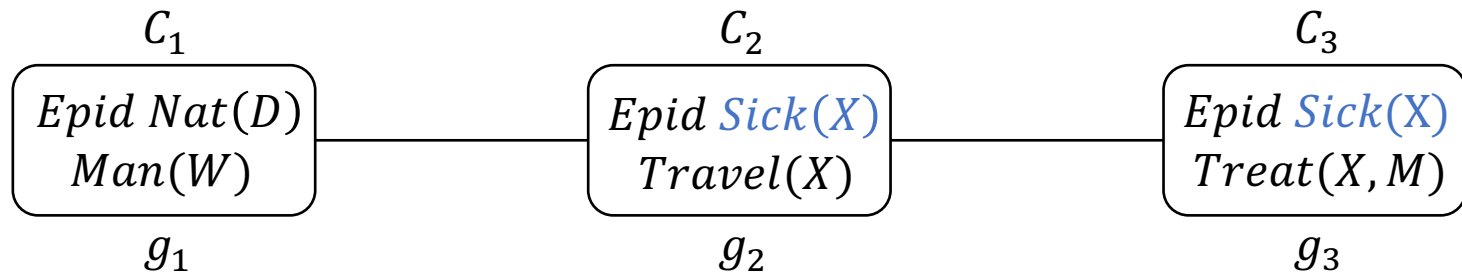
- At every parcluster that contains evidence variables
 - $g_E(\textit{Sick}(X')), X' \in \{\textit{eve}, \textit{alice}\}$
 - Parclusters
 - $\textit{Sick}(X') \notin C_1$
 - $\textit{Sick}(X') \subset C_2$
 - $\textit{Sick}(X') \subset C_3$



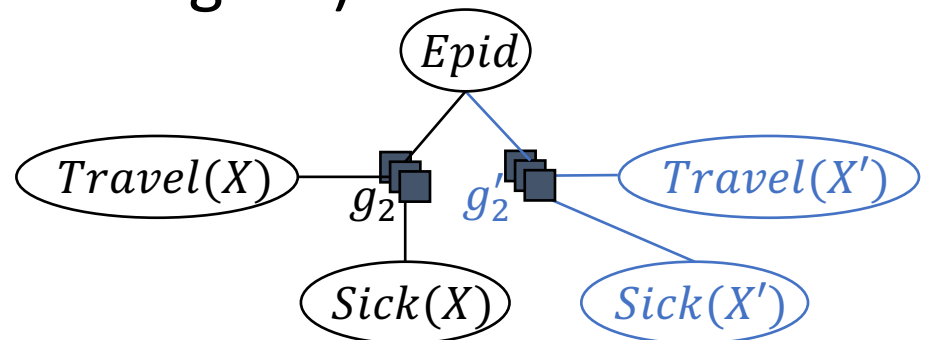
- Enter evidence at C_2 and C_3

LJT: Enter Evidence

- At every parcluster that contains evidence variables
 - $g_E(\textit{Sick}(X')), X' \in \{\textit{eve}, \textit{alice}\}$
 - Parclusters C_2 and C_3



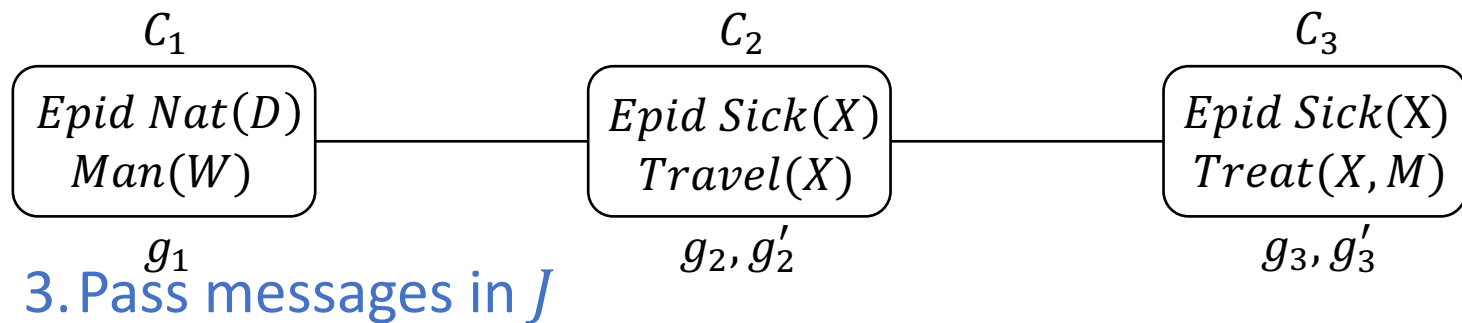
- Enter evidence at C_2 (C_3 analogous)
 - Split** local model
 - $X \in \{\textit{bob}, \dots\}$
 - Absorb** evidence in g'_2



Lifted Junction Tree Algorithm: LJT

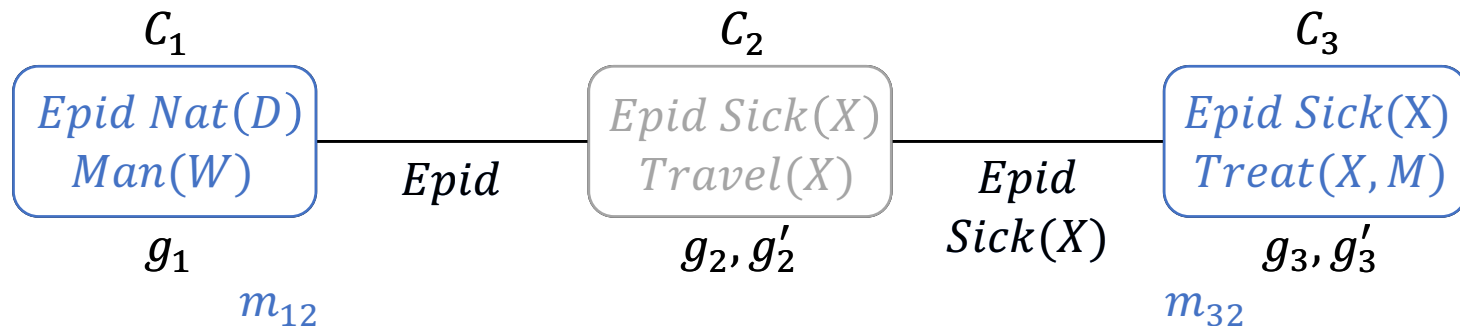
Braun and Möller (2017)

- Input
 - Model G
 - Evidence E
 - Queries Q
- Algorithm
 1. Build FO jtree J for G
 2. Enter evidence E into J



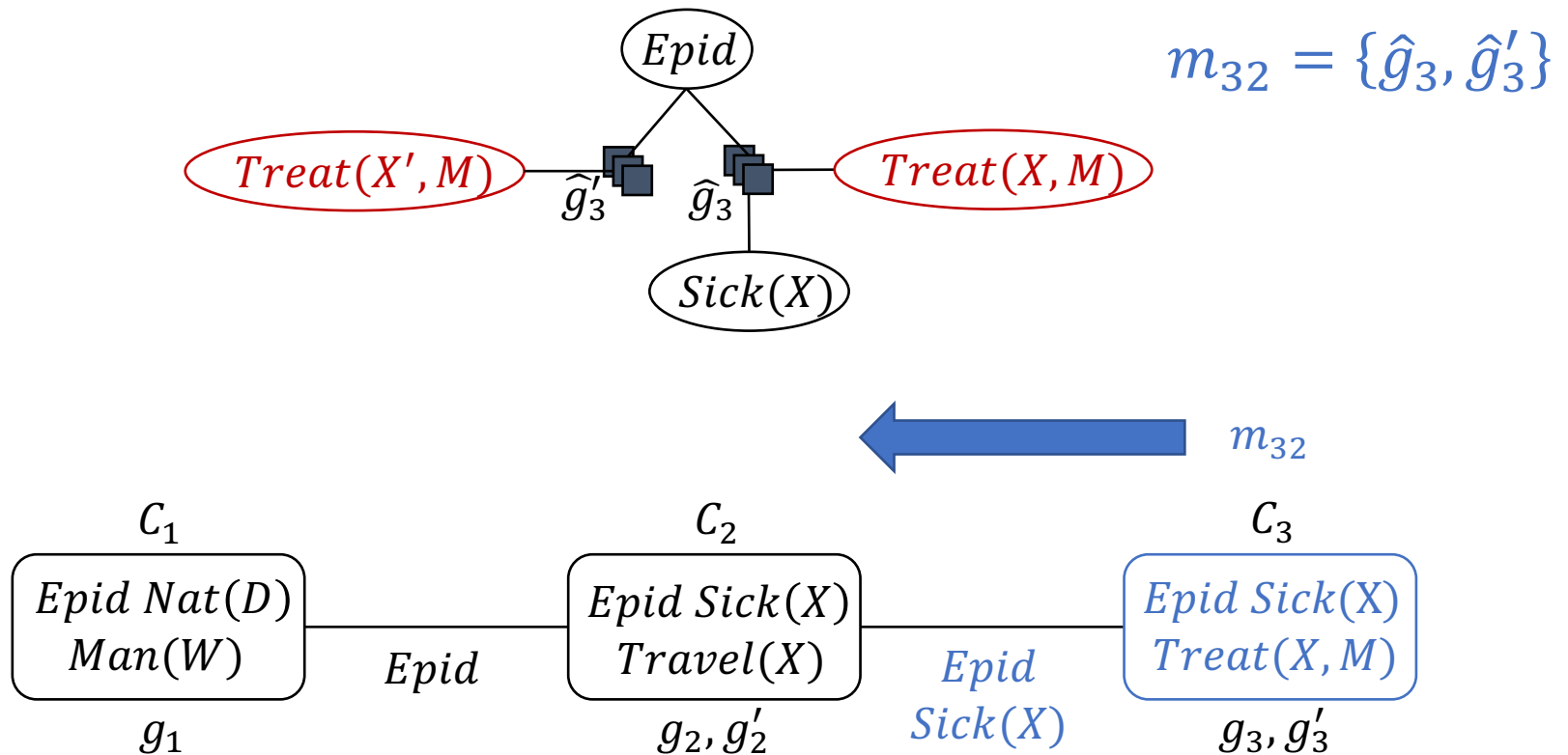
LJT: Pass Messages

- Separators
- Messages
 - Inbound
 - m_{12} from C_1 to C_2 over $Epid$
 - m_{32} from C_3 to C_2 over $Epid, Sick(X)$
 - Outbound
 - m_{21} from C_1 to C_2 over $Epid$
 - m_{23} from C_3 to C_2 over $Epid, Sick(X)$



LJT: Example Message Inbound

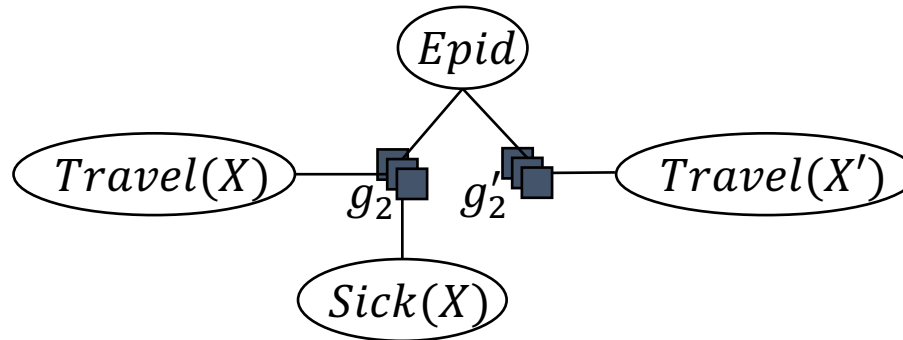
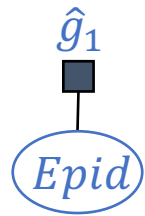
- m_{32} from C_3 to C_2
 - Eliminate $Treat(X, P), Treat(X', P)$



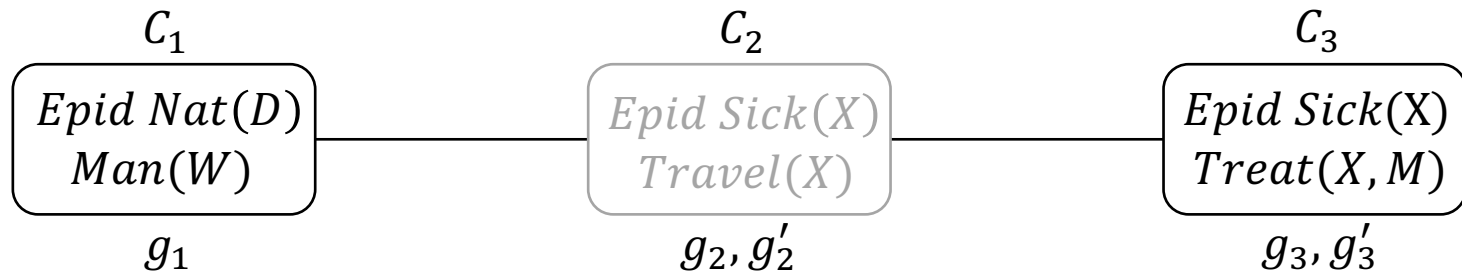
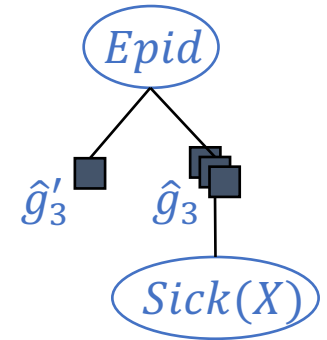
LJT: Messages at C_2

- After m_{12} and m_{32} arrived

$$m_{12} = \{\hat{g}_1\}$$



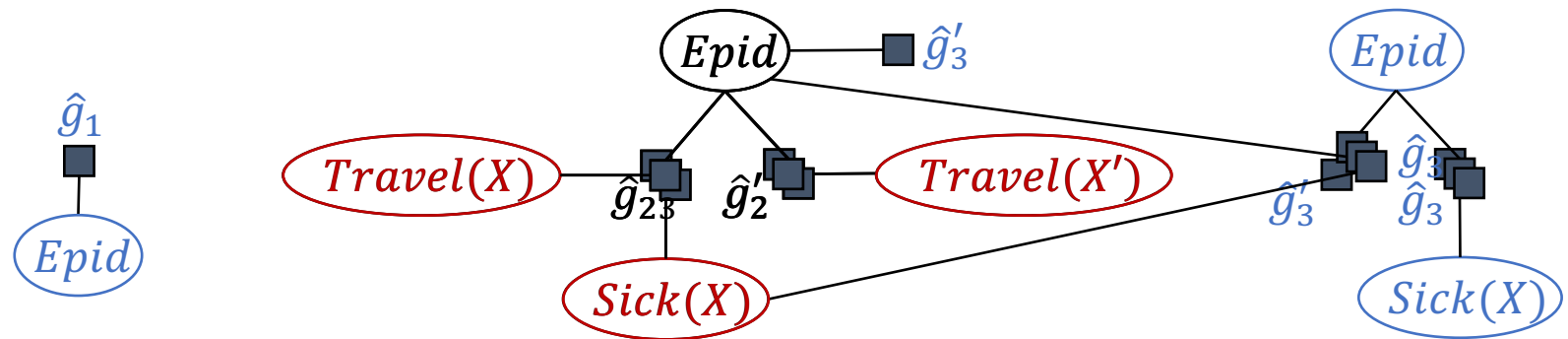
$$m_{32} = \{\hat{g}_3, \hat{g}_3'\}$$



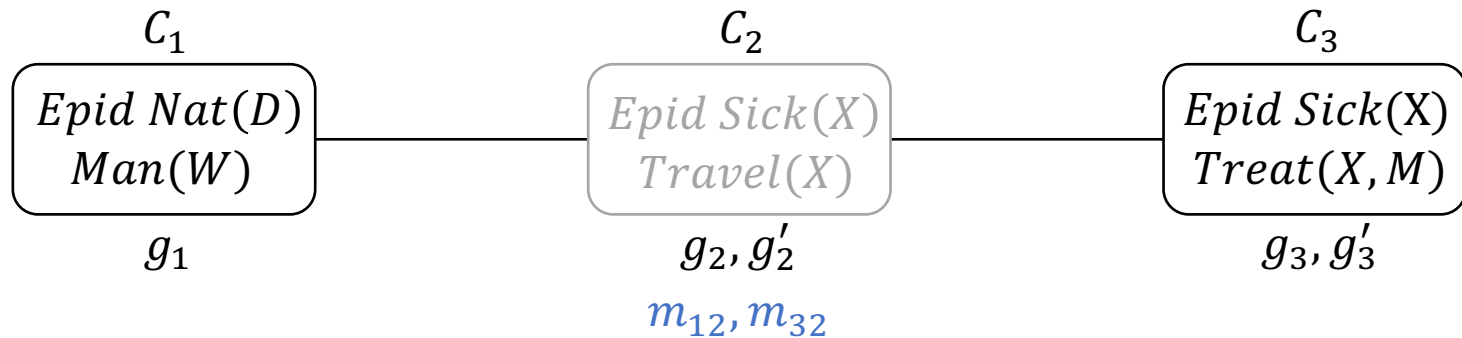
$$m_{12}, m_{32}$$

LJT: Example Message Outbound

- m_{21} from C_2 to C_1
 - Eliminate *Sick(X), Travel(X), Travel(X')* from g_2, g'_2, m_{32}

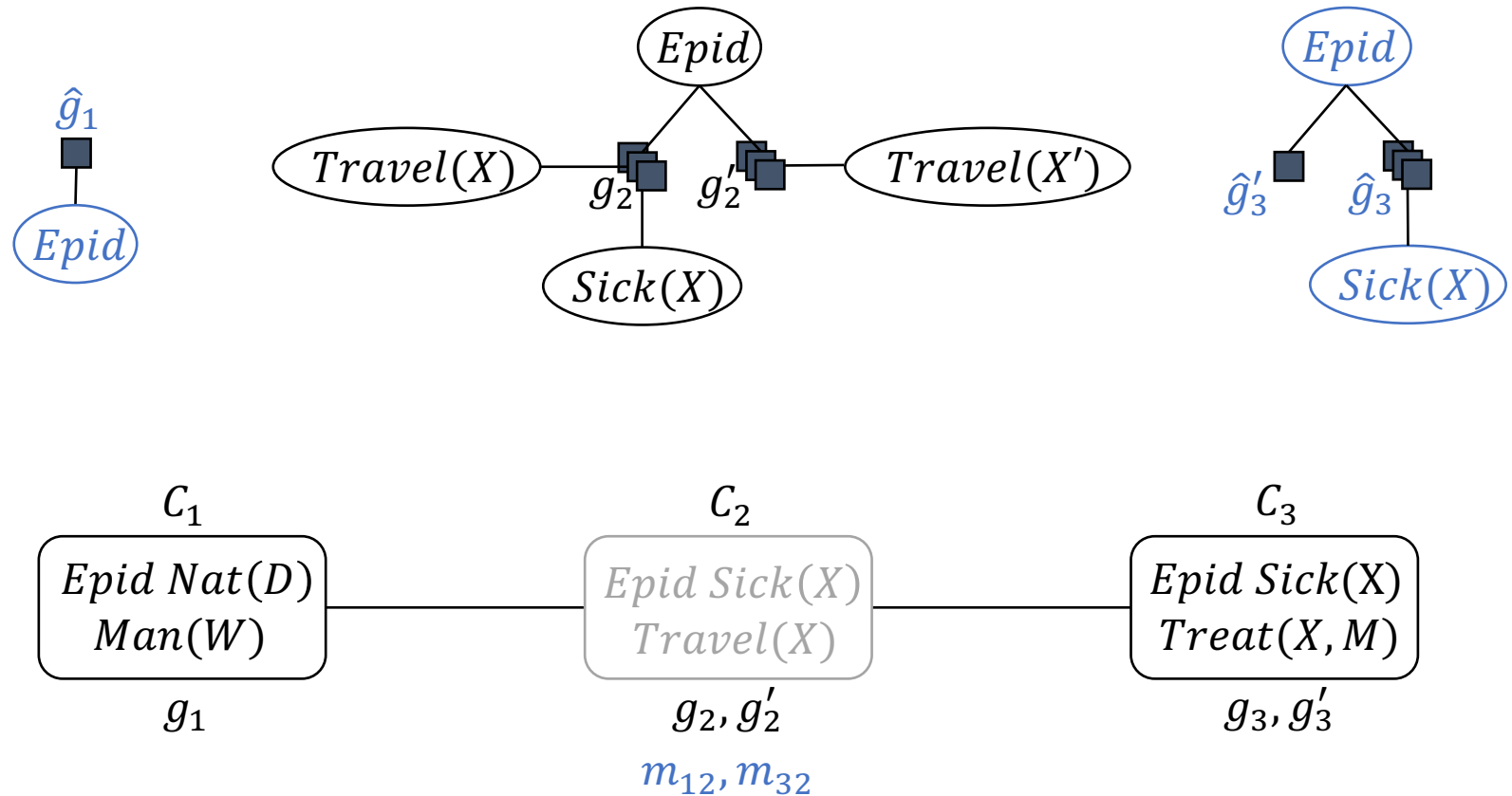


$$m_{21} = \{\hat{g}_{23}, \hat{g}'_2, \hat{g}'_3\}$$



LJT: Example Message Outbound

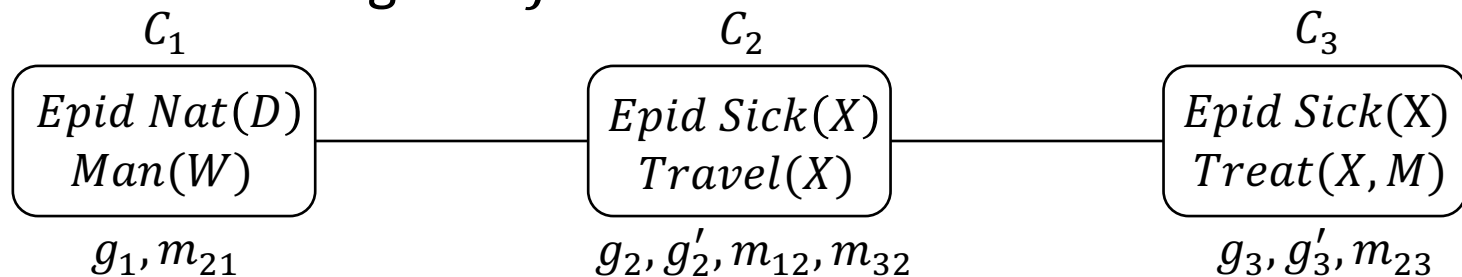
- m_{23} from C_2 to C_3
 - Eliminate $Travel(X), Travel(X')$ from g_2, g'_2, m_{12}



Lifted Junction Tree Algorithm: LJT

Braun and Möller (2017)

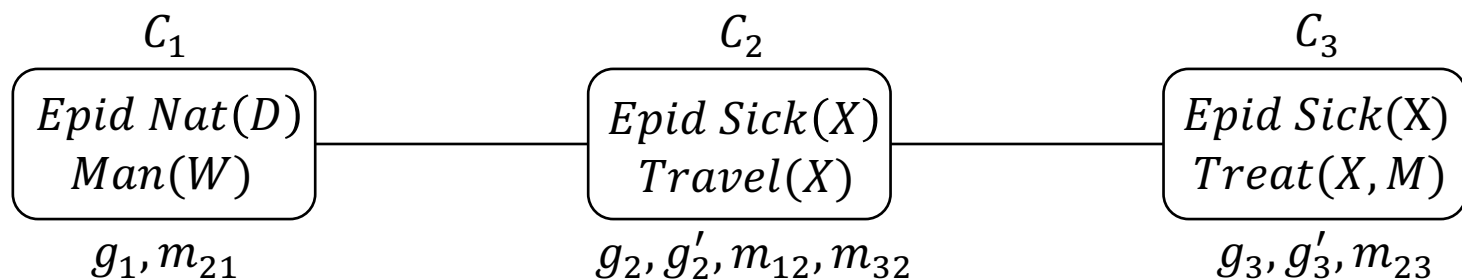
- Input
 - Model G
 - Evidence E
 - Queries Q
- Algorithm
 1. Build FO jtree J for G
 2. Enter evidence E into J
 3. Pass messages in J



4. Answer queries Q

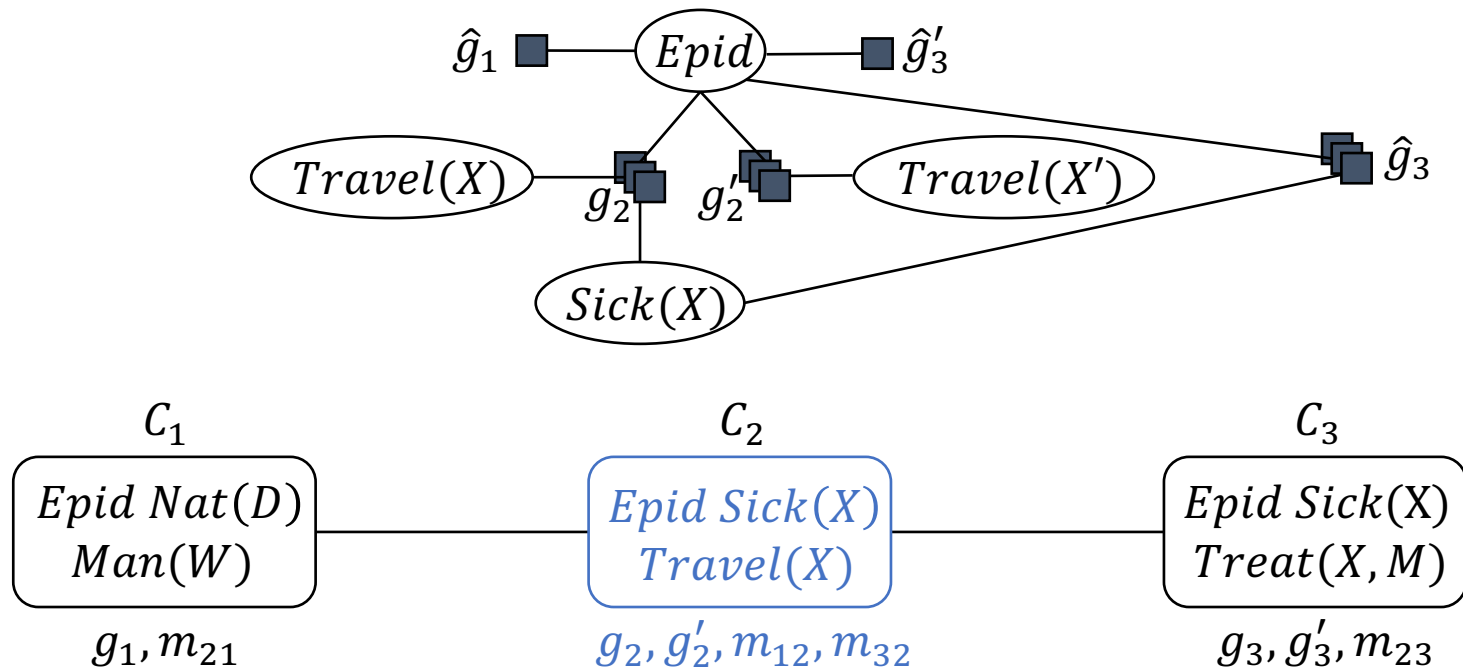
LJT: Answer Queries

- Queries $Q = \{Travel(eve), Epid\}$
- For each query Q
 - Find **parcluster** that contains Q
 - Extract **submodel** of local model and messages
 - Use LVE to answer Q



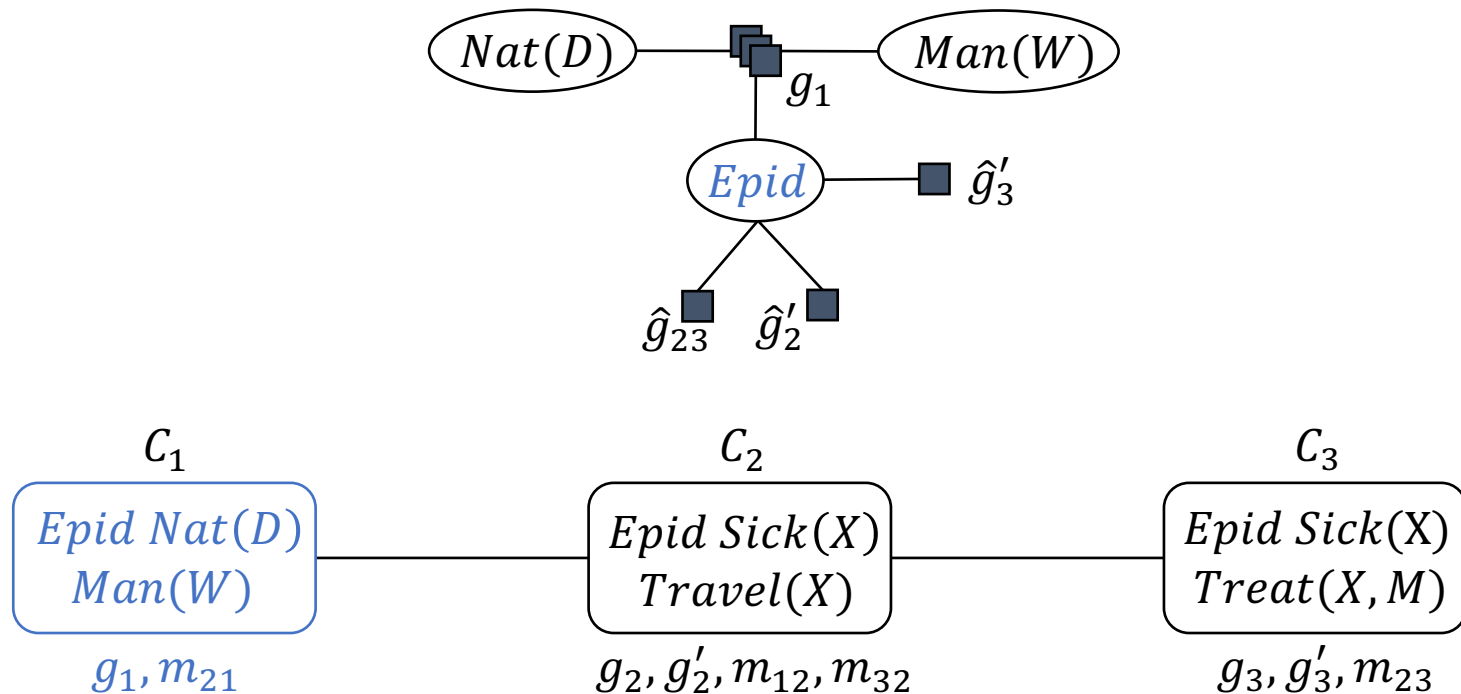
LJT: Answer Queries

- $Q_1 = Travel(eve)$
 - Find parcluster: C_2
 - Extract submodel: $G' = \{g_2, g'_2, m_{12}, m_{32}\}$
 - Answer $Travel(eve)$ with LVE



LJT: Answer Queries

- $Q_2 = \text{Epid}$
 - Find parcluster: C_1 (any of the three parclusters)
 - Extract submodel: $G' = \{g_1, m_{21}\}$
 - Answer *Epid* with LVE



LJT: Analysis

- Static overhead
 - Construction
 - Evidence entering
 - Message passing
- Payoff during QA
 - Multiple queries
 - Complexity of LVE for one query = Complexity of message pass in LJ

Queries all under the same evidence

$$E = \{Sick(eve) = true, \\ Sick(alice) = true\}$$

Extending LVE and LJT

- Liftable models (**fusion**)
- Conjunctive queries
 - $P(\text{Epid}, \text{Travel}(\text{eve}))$
- **Isomorphic query terms** (**parameterised queries**)
 - $P(\text{Sick}(\text{eve}), \text{Sick}(\text{alice}), \text{Sick}(\text{bob})) \triangleq P(\text{Sick}(X))$
- Most probable assignment (**MPE, MAP**)
 - New **argmax** operators
- Adaptive inference (**incremental changes**)
 - Evidence, model structure, parfactors
→ Adaptive steps of LJT

Isomorphic Conjunctive Queries

Braun and Möller (2018a)

- $Q = \text{Sick}(\text{alice}) \wedge \text{Sick}(\text{eve}) \wedge \text{Sick}(\text{bob})$

- Same parcl

- But

- Split mode

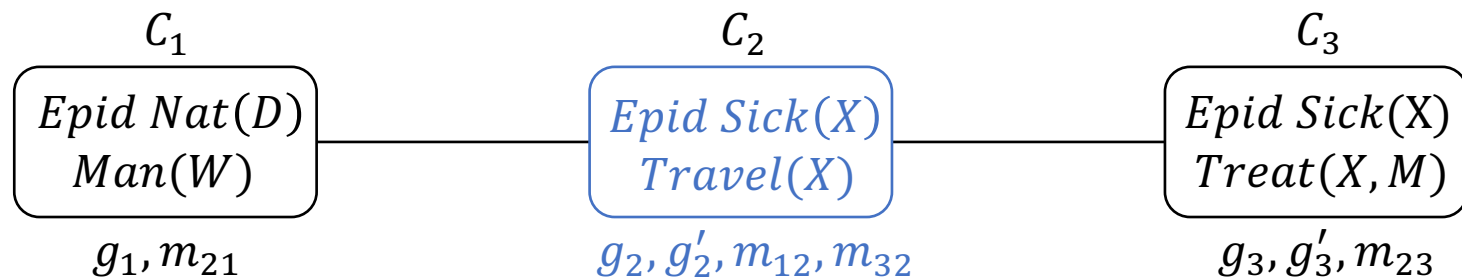
- Eliminate r

large inter

- Normalise

<i>Sick(alice)</i>	<i>Sick(eve)</i>	<i>Sick(bob)</i>	<i>g</i>
<i>false</i>	<i>false</i>	<i>false</i>	1
<i>false</i>	<i>false</i>	<i>true</i>	2
<i>false</i>	<i>true</i>	<i>false</i>	2
<i>false</i>	<i>true</i>	<i>true</i>	3
<i>true</i>	<i>false</i>	<i>false</i>	2
<i>true</i>	<i>false</i>	<i>true</i>	3
<i>true</i>	<i>true</i>	<i>false</i>	3
<i>true</i>	<i>true</i>	<i>true</i>	4

query
tical sums,



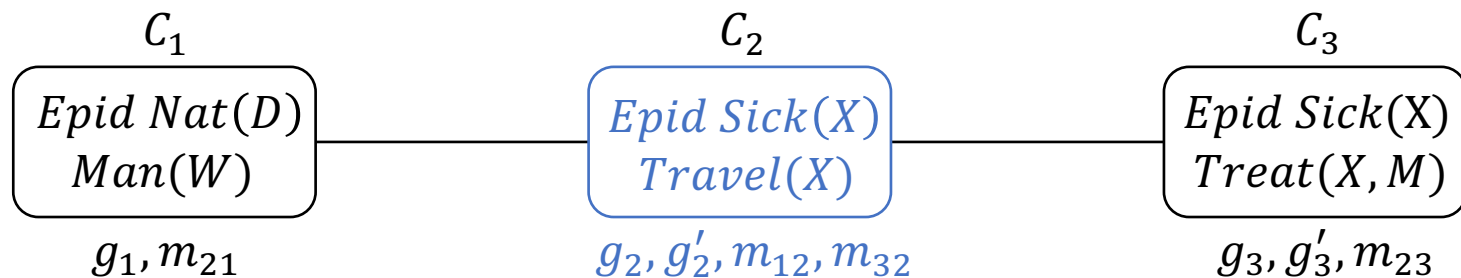
Parameterised Queries

Braun and Möller (2018a)

- Compact query representation
- Lifted computations during LVE
- Compact result representation

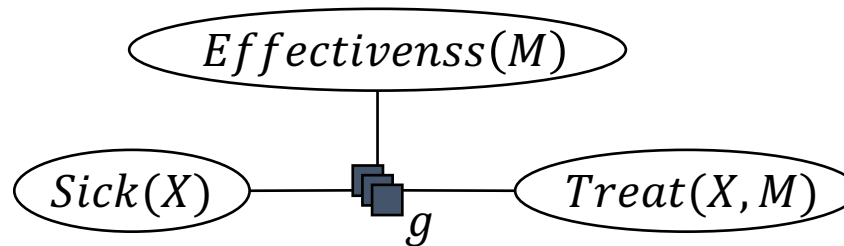
- $Q = \textit{Sick}(X)$
- Result: $\#_X[\textit{Sick}(X)]$

$\#_X[\textit{Sick}(X)]$	g
[0,3]	1
[1,2]	2
[2,1]	3
[3,0]	4



Grounding Parameterised Queries

- $g(Sick(X), Treat(X, P), Effectiveness(P))$



- $Q = Treat(X, M)$
 - LVE grounds one logical variable
- $Q = Sick(X), Treat(X, M), Effectiveness(M)$
 - LVE grounds one logical variable
 - Compare forbidden queries in PDBs (Dalvi & Suciu 12)

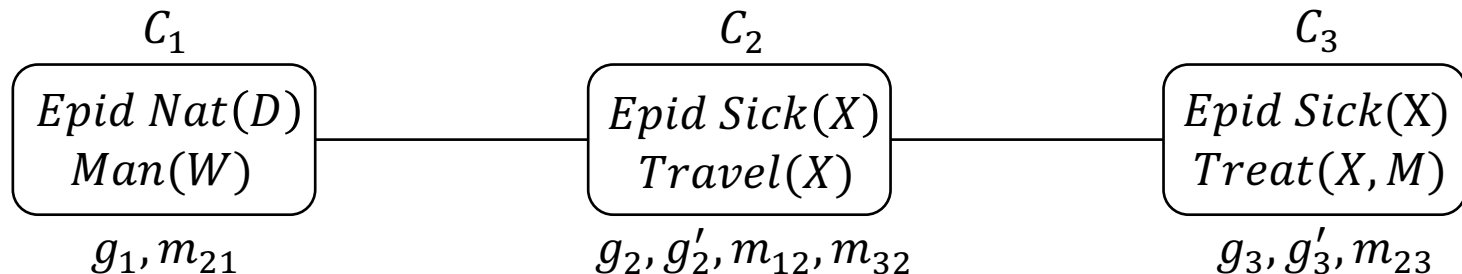
Does it have to be LVE in LJT?

Braun and Möller (2018c)

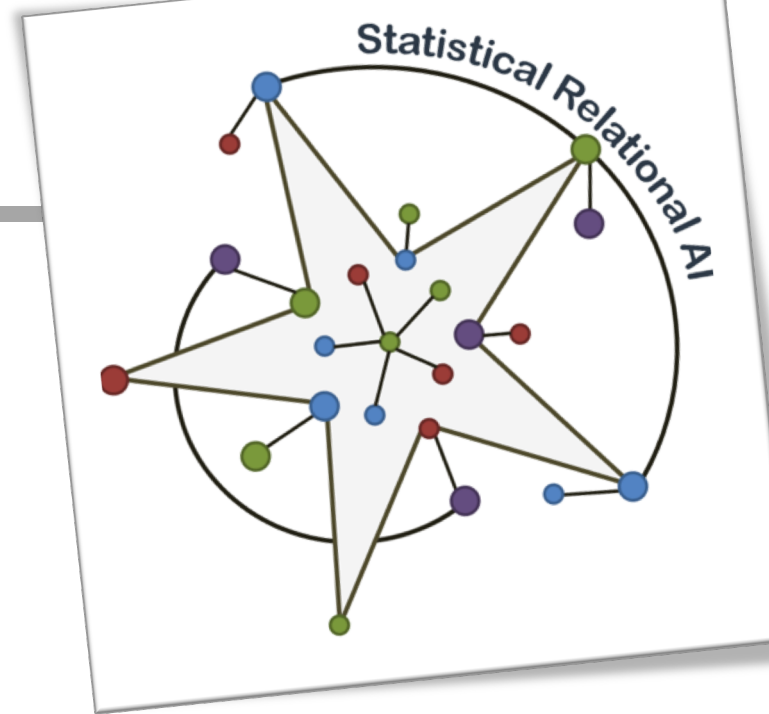
LJT with LVE &

First-order Knowledge Compilation (FOKC)
to solve a WFOMC problem

- LVE for evidence entering and message passing
- FOKC for query answering



- Other lifted algorithms to replace LVE in LJT...

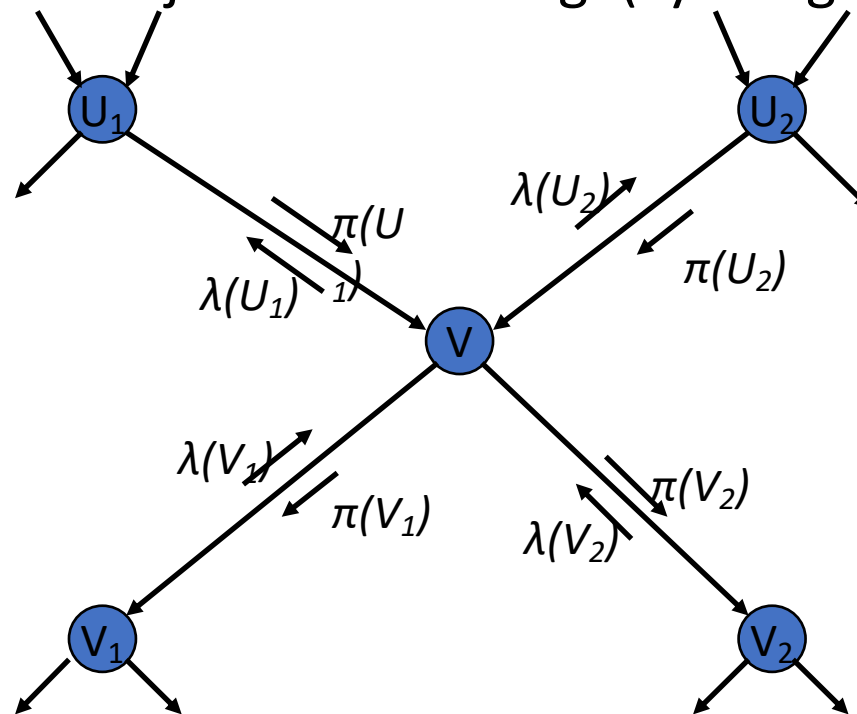


But...

From where comes the lifted model?

A Bit of History...

- Pearl's Belief propagation [Pearl 1982]
 - Messages on Bayes net
 - Exact for polytrees (**no cycles in undirected graph!**)
 - Precursor of junction tree alg. (cycles go into clusters)



J. Pearl, Reverend Bayes on Inference Engines: A Distributed Hierarchical Approach. Proc. AAAI-82. 1982.

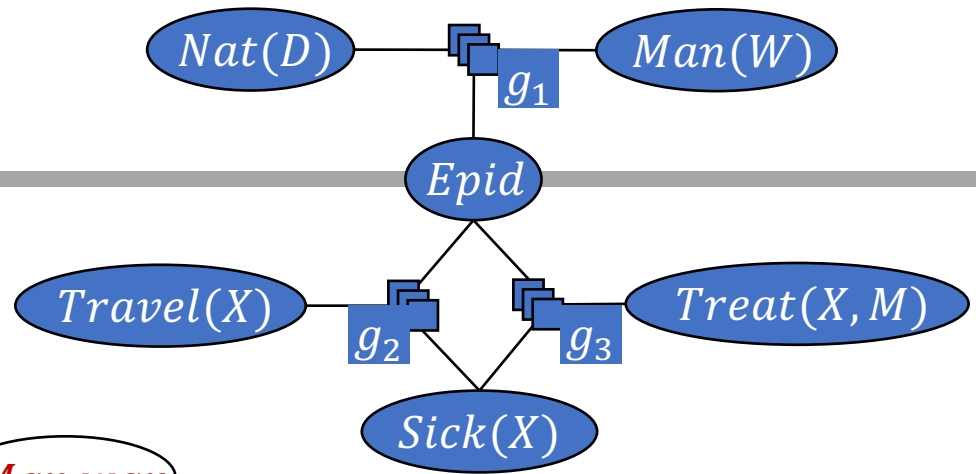
Loopy Belief Propagation

- Pass messages on graph
 - If no cycles: exact
 - Else: approximate
- Lifted (loopy) belief propagation
 - Exploit computational symmetries
 - Compress graph whenever nodes would send identical messages
 - Send messages on compressed graph

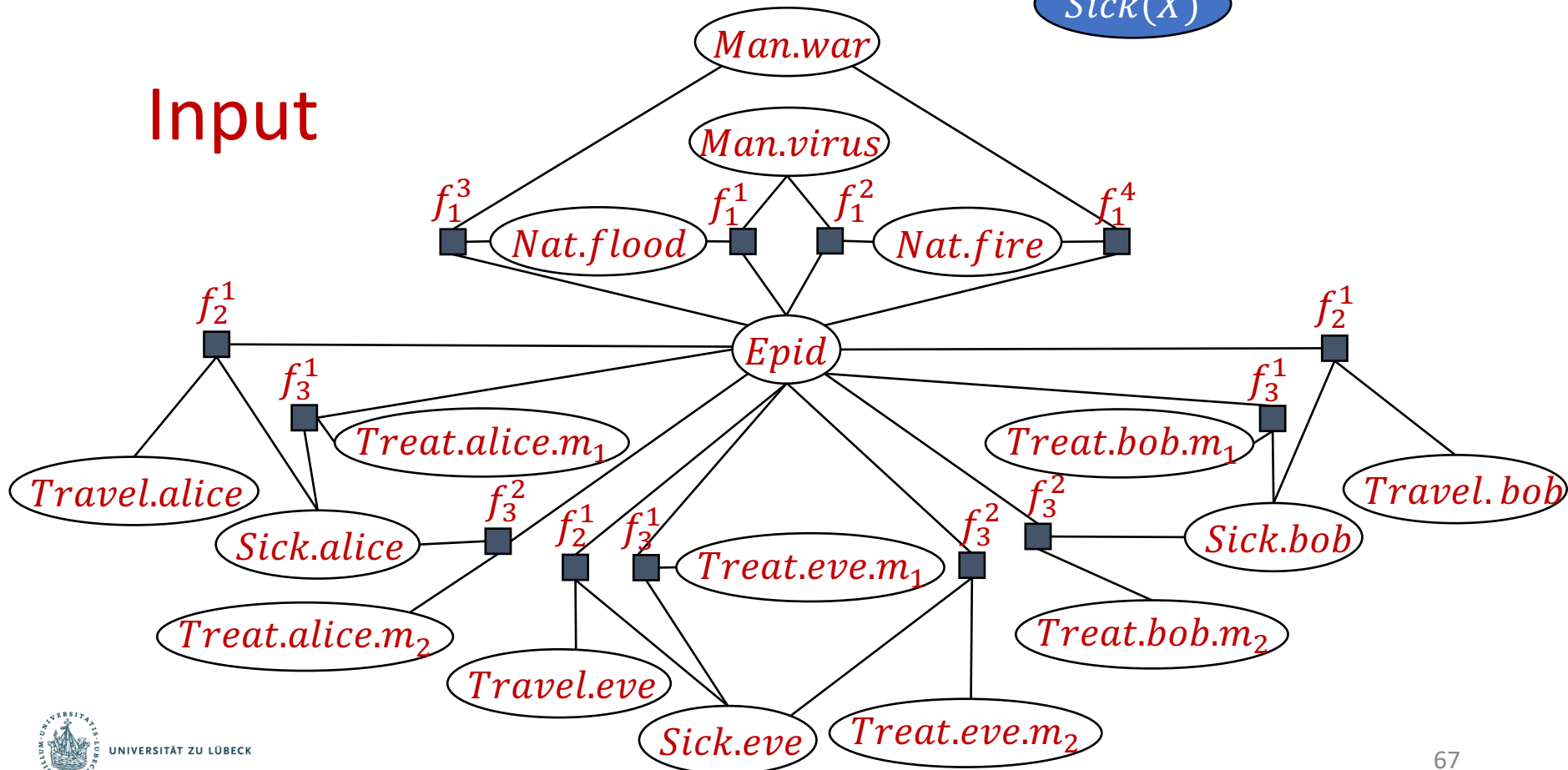
→ Colour passing algorithm for compression

Compression

Goal



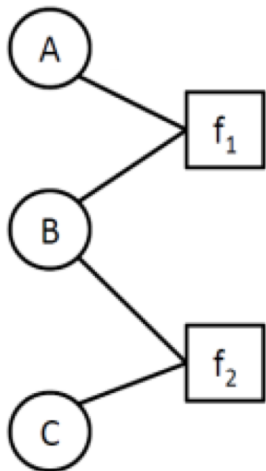
Input



Compression: Pass the colours around*

- **Colour nodes according to the evidence you have**

- No evidence, say **red**
- State „one“, say **brown**
- State „two“, say **orange**
- ...

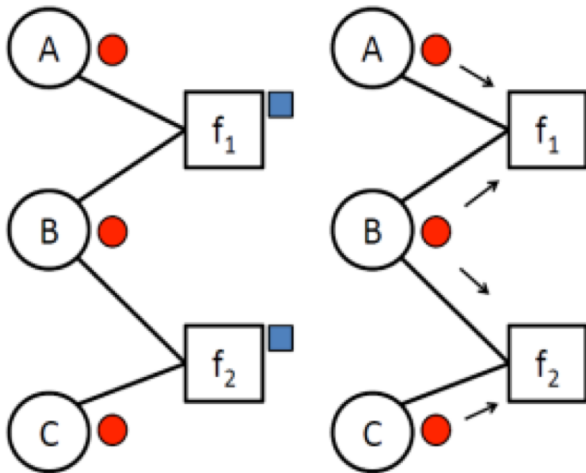


- **Colour factors distinctively according to their equivalences** For instance, assuming f_1 and f_2 to be identical and B appears at the second position within both, say **blue**

*can also be done at the „lifted“, i.e., relational level

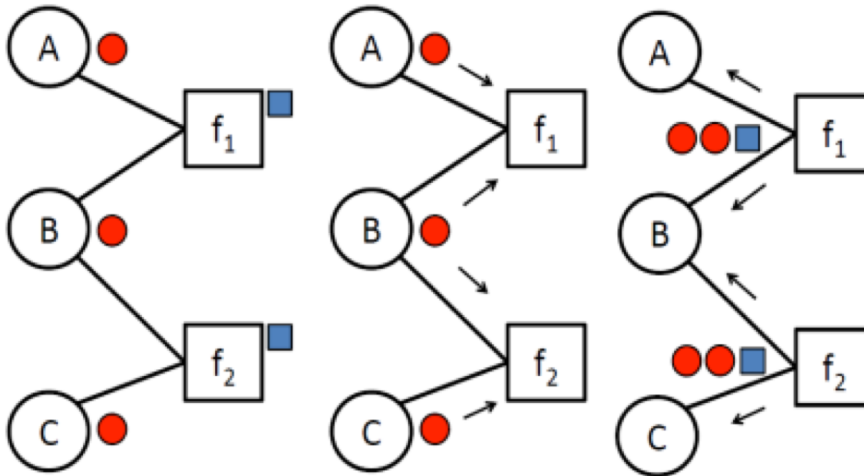
Compression: Pass the colours around

1. Each factor collects the colours of its neighbouring nodes



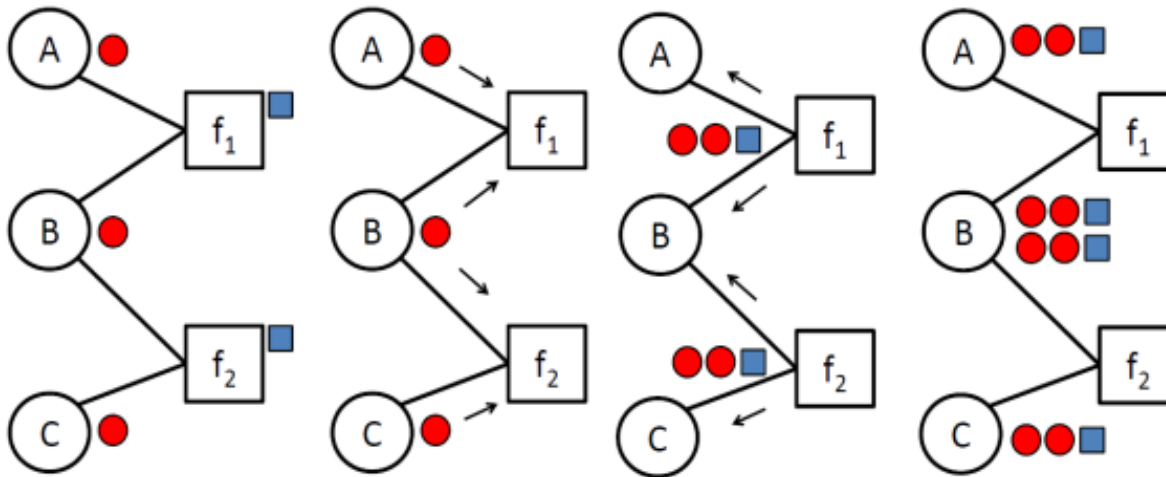
Compression: Pass the colours around

1. Each factor collects the colours of its neighbouring nodes
2. Each factor „signs“ its colour signature with its own colour



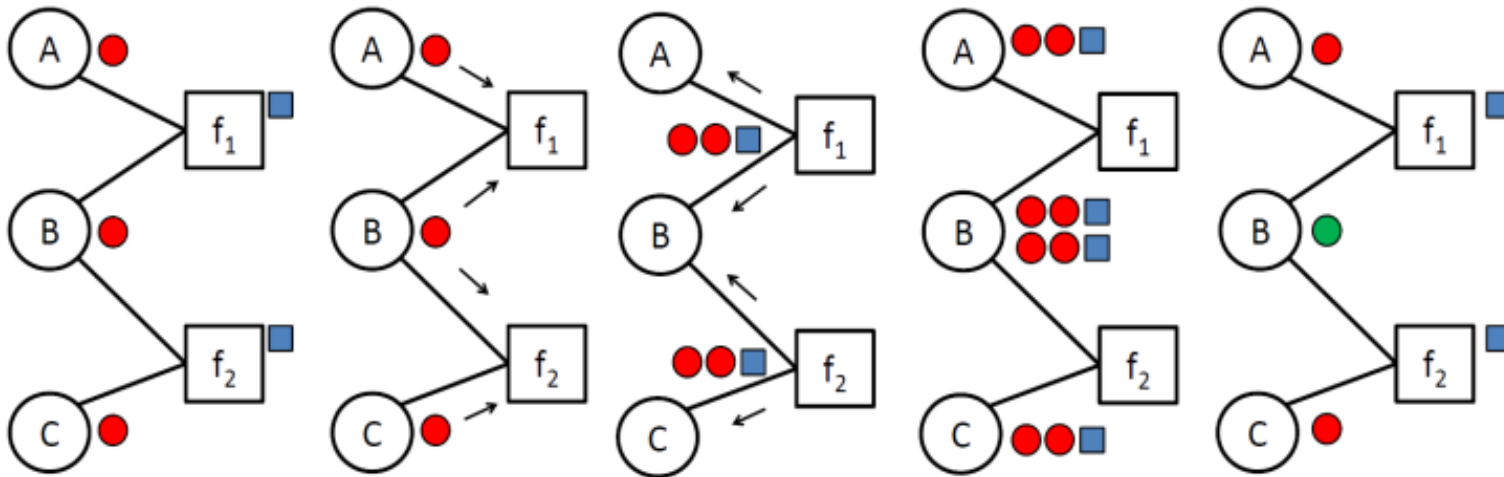
Compression: Pass the colours around

1. Each factor collects the colours of its neighbouring nodes
2. Each factor „signs“ its colour signature with its own colour
3. Each node collects the signatures of its neighbouring factors



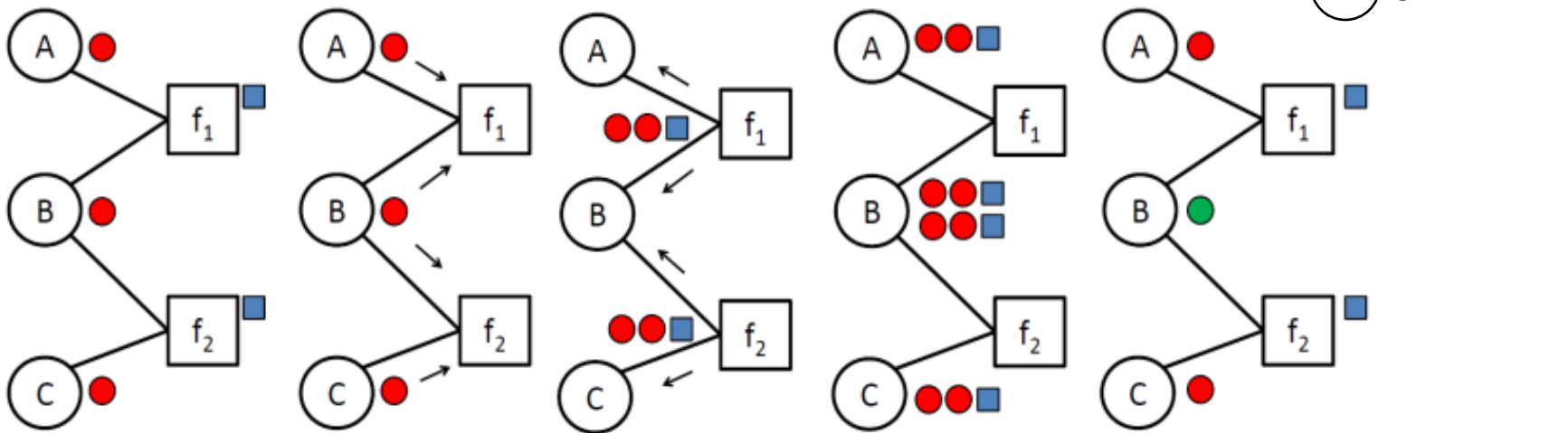
Compression: Pass the colours around

1. Each factor collects the colours of its neighbouring nodes
2. Each factor „signs“ its colour signature with its own colour
3. Each node collects the signatures of its neighbouring factors
4. Nodes are recoloured according to the collected signatures



Compression: Pass the colours around

1. Each factor collects the colours of its neighbouring nodes
2. Each factor „signs“ its colour signature with its own colour
3. Each node collects the signatures of its neighbouring factors
4. Nodes are recoloured according to the collected signatures
5. If no new colour is created stop, otherwise go back to 1



Outlook

- Continue optimising
 - Parallelisation
 - Caching
- From discrete over interval to continuous ranges
- Learning?
 - Structure
 - Potentials
 - Symmetries
 - *Transfer learning*
- Open world?
 - Unknown domains
 - Unknown behaviour

Wrap-up Exact Lifted Inference

- Parfactor models for **sparse encoding**
 - Factorisation of full joint distribution
 - Logical variables to model objects
- Algorithms for exact query answering
 - **LVE** for single inference
 - **LJT** for repeated inference
 - Extensions possible
 - Parameterised, conjunctive queries
 - Temporal models
 - Incremental changes
 - Assignment queries
- Colour passing for compressing models

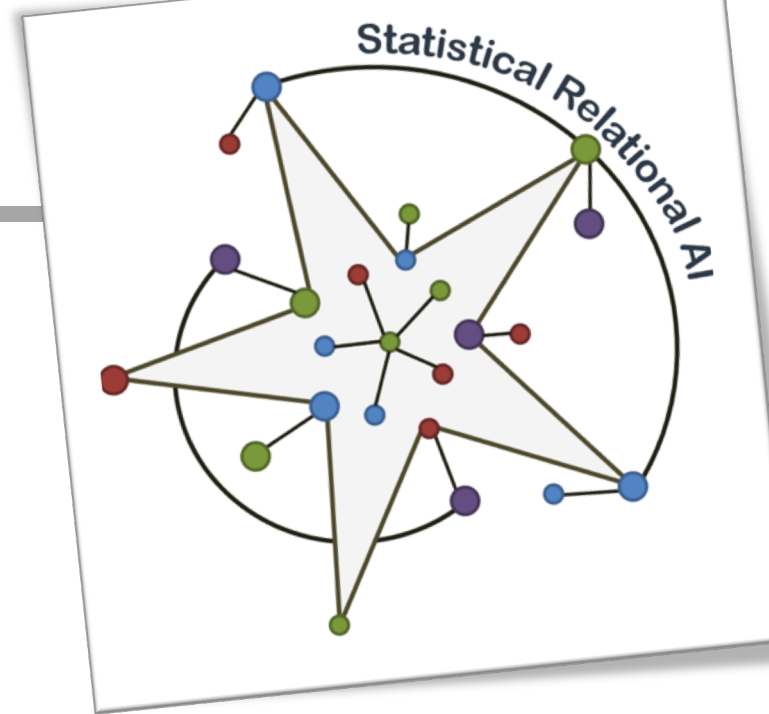
Next: Temporal Lifted Inference

Mission and Schedule of the Tutorial*

Providing an introduction into inference in StaRAI

- Introduction 20 min ✓
 - StaR AI
- Overview: Probabilistic relational modeling 30 min ✓
 - Semantics (grounded-distributional, maximum entropy)
 - Inference problems and their applications
 - Algorithms and systems
- Scalable static inference 40 + 30 min ✓
 - Exact propositional inference
 - Exact lifted inference
- Scalable dynamic inference 50 min
 - Exact propositional inference
 - Exact lifted inference
- Summary 10 min

*Thank you to the SRL/StaRAI crowd for all their exciting contributions! The tutorial is necessarily incomplete. Apologies to anyone whose work is not cited



References

And own work

References

- **Apsel and Brafman (2012)**

Udi Apsel and Ronen I. Brafman. Exploiting Uniform Assignments in First-Order MPE. *Proceedings of the 28th Conference on Uncertainty in Artificial Intelligence*, 2012.

- **Cohen (2016)**

William W. Cohen. TensorLog: A Differentiable Deductive Database, arxiv (<https://arxiv.org/abs/1605.06523>), 2016.

- **Cohen et al. (2017)**

William W. Cohen, Fan Yang, and Kathryn Rivard Mazaitis. TensorLog: Deep Learning Meets Probabilistic Databases, arxiv (<https://arxiv.org/abs/1707.05390>), 2017.

- **Dalvi and Suciu (2012)**

Nilesh Salvi and Dan Suciu. Philip Dawid. The Dichotomy of Probabilistic Inference for Unions of Conjunctive Queries. *Journal of the ACM*, 59(6):1–97, 2012.

- **Dawid (1992)**

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

References

- Dechter (1999)

Rina Dechter. Bucket Elimination: A Unifying Framework for Probabilistic Inference. In *Learning and Inference in Graphical Models*, pages 75–104. MIT Press, 1999.

- De Salvo Braz et al. (2005)

Rodrigo de Salvo Braz, Eyal Amir, and Dan Roth. Lifted First-order Probabilistic Inference. *IJCAI-05 Proceedings of the 19th International Joint Conference on Artificial Intelligence*, 2005.

- De Salvo Braz et al. (2006)

Rodrigo de Salvo Braz, Eyal Amir, and Dan Roth. MPE and Partial Inversion in Lifted Probabilistic Variable Elimination. *AAAI-06 Proceedings of the 21st Conference on Artificial Intelligence*, 2006.

- Jensen et al. (1990)

Finn V. Jensen, Steffen L. Lauritzen, and Kristian G. Olesen. Bayesian Updating in Recursive Graphical Models by Local Computations. *Computational Statistics Quarterly*, 4:269–282, 1990.

References

- Koller and Friedman (2009)

Daphne Koller and Nir Friedman. Chapter 10: Exact Inference: Clique Trees. In: *Probabilistic Graphical Models: Principles and Techniques*, MIT Press, 2009.

- Lauritzen and Spiegelhalter (1988)

Steffen L. Lauritzen and David J. Spiegelhalter. Local Computations with Probabilities on Graphical Structures and Their Application to Expert Systems. *Journal of the Royal Statistical Society. Series B: Methodological*, 50:157–224, 1988.

- Milch et al. (2008)

Brian Milch, Luke S. Zettlemoyer, Kristian Kersting, Michael Haimes, and Leslie Pack Kaelbling. Lifted Probabilistic Inference with Counting Formulas AAAI-08 Proceedings of the 23rd Conference on Artificial Intelligence, 2008.

- Murphy (2002)

Kevin P. Murphy. Dynamic Bayesian Networks: Representation, Inference and Learning. *PhD Thesis University of California, Berkeley*, 2002.

References

- Poole (2003)

David Poole. First-order Probabilistic Inference. In *Proceedings of the 18th International Joint Conference on Artificial Intelligence*, 2003.

- Shafer and Shenoy (1989)

Glenn R. Shafer and Prakash P. Shenoy. Probability Propagation. *Annals of Mathematics and Artificial Intelligence*, 2(1):327–351, 1989.

- Sharma et al. (2018)

Vishal Sharma, Noman Ahmed Sheikh, Happy Mittal, Vibhav Gogate, and Parag Singla. Lifted Marginal MAP Inference. In *Proceedings of the 34th Conference on Uncertainty in Artificial Intelligence 4*, 2018.

- Shenoy and Shafer (1990)

Prakash P. Shenoy and Glenn R. Shafer. Axioms for Probability and Belief-Function Propagation. *Uncertainty in Artificial Intelligence 4*, 9:169–198, 1990.

References

- Taghipour et al. (2013)

Nima Taghipour, Jesse Davis, and Hendrik Blockeel. A Generalized Counting for Lifted Variable Elimination. In *Proceedings of the International Conference on Inductive Logic Programming*, pages 107–122, 2013.

- Taghipour et al. (2013a)

Nima Taghipour, Daan Fierens, Jesse Davis, and Hendrik Blockeel. Lifted Variable Elimination: Decoupling the Operators from the Constraint Language. *Journal of Artificial Intelligence Research*, 47(1):393–439, 2013.

- Taghipour et al. (2013b)

Nima Taghipour, Jesse Davis, and Hendrik Blockeel. First- order decomposition trees. In *Advances in Neural Information Processing Systems 26*, pages 1052–1060. Curran Associates, Inc., 2013.

- Zhang and Poole (1994)

Nevin L. Zhang and David Poole. A Simple Approach to Bayesian Network Computations. In *Proceedings of the 10th Canadian Conference on Artificial Intelligence*, pages 171–178, 1994.

Work @ IFIS

- **Braun and Möller (2016)**

Tanya Braun and Ralf Möller. Lifted Junction Tree Algorithm. In *Proceedings of KI 2016: Advances in Artificial Intelligence*, pages 30–42, 2016.

- **Braun and Möller (2017)**

Tanya Braun and Ralf Möller. Preventing Groundings and Handling Evidence in the Lifted Junction Tree Algorithm. In *Proceedings of KI 2017: Advances in Artificial Intelligence*, pages 85–98, 2017.

- **Braun and Möller (2017a)**

Tanya Braun and Ralf Möller. Counting and Conjunctive Queries in the Lifted Junction Tree Algorithm. In *Postproceedings of the 5th International Workshop on Graph Structures for Knowledge Representation and Reasoning*, 2017.

Work @ IFIS

- **Braun and Möller (2018)**

Tanya Braun and Ralf Möller. Adaptive Inference on Probabilistic Relational Models. In *Proceedings of the 31st Australasian Joint Conference on Artificial Intelligence*, 2018, to appear.

- **Braun and Möller (2018a)**

Tanya Braun and Ralf Möller. Parameterised Queries and Lifted Query Answering. In *IJCAI-18 Proceedings of the 27th International Joint Conference on Artificial Intelligence*, 2018.

- **Braun and Möller (2018b)**

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

- **Braun and Möller (2018c)**

Tanya Braun and Ralf Möller. Fusing First-order Knowledge Compilation and the Lifted Junction Tree Algorithm. In *Proceedings of KI 2018: Advances in Artificial Intelligence*, 2018.

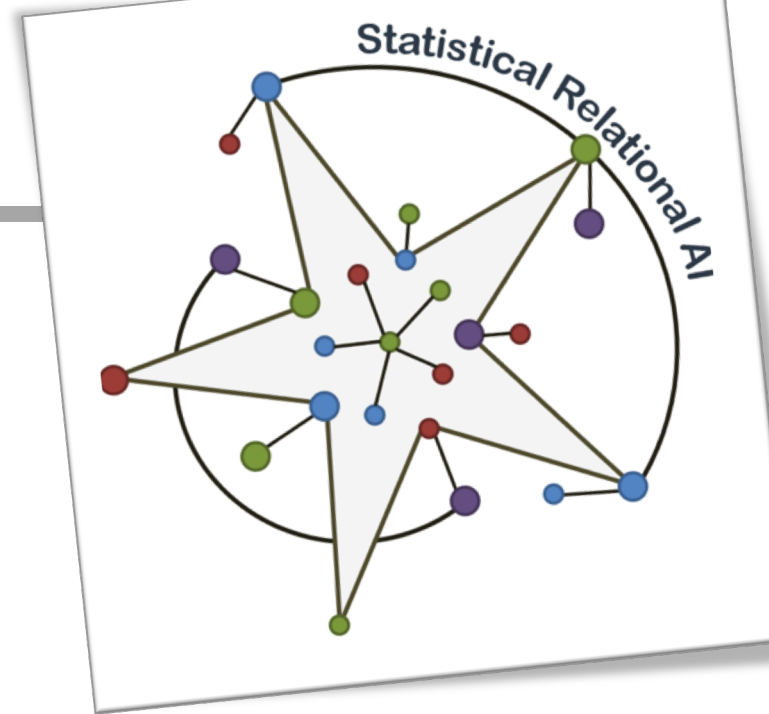
Work @ IFIS

- Gehrke et al. (2018)

Marcel Gehrke, Tanya Braun, and Ralf Möller. Lifted Dynamic Junction Tree Algorithm. In *Proceedings of the International Conference on Conceptual Structures*, 2018.

- Gehrke et al. (2019)

Marcel Gehrke, Tanya Braun, and Ralf Möller. Relational Forward Backward Algorithm for Multiple Queries. In *FLAIRS-32 Proceedings of the 32nd International Florida Artificial Intelligence Research Society Conference*, 2019.

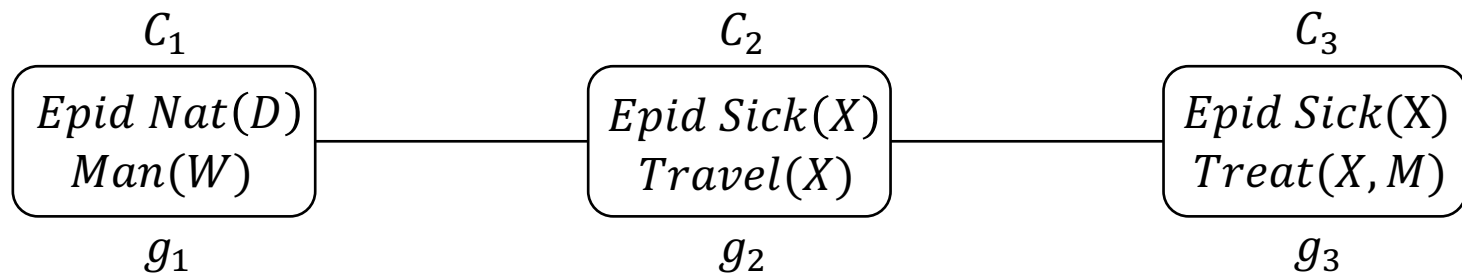


Appendix

Extensions to LVE and LJT

New Evidence? Back to Step 2

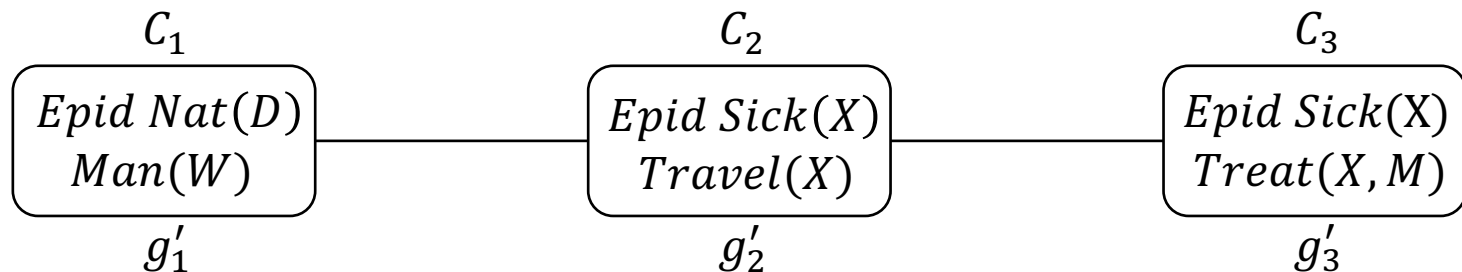
- Input
 - Model G
 - Evidence $E = \{Epid = true\}$
 - Queries Q
- Algorithm
 1. Build FO jtree J for G



2. Enter evidence E into J

And continue...

- Input
 - Model G
 - Evidence $E = \{Epid = true\}$
 - Queries Q
- Algorithm
 1. Build FO jtree J for G
 2. Enter evidence E into J

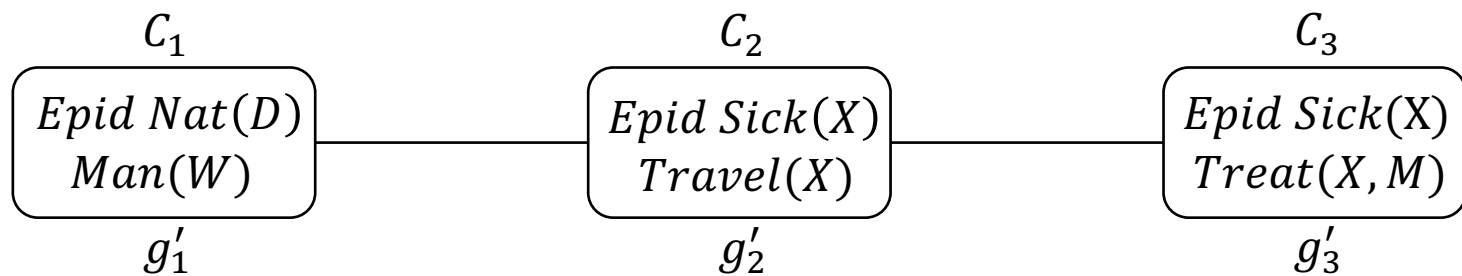


3. Pass messages

What about incremental changes?

Braun and Möller (2018)

- Input
 - Model G
 - Evidence $E = \{Epid = true\} \cup \{Travel(eve) = true\}$
 - Queries Q
- Algorithm
 1. Build FO jtree J for G

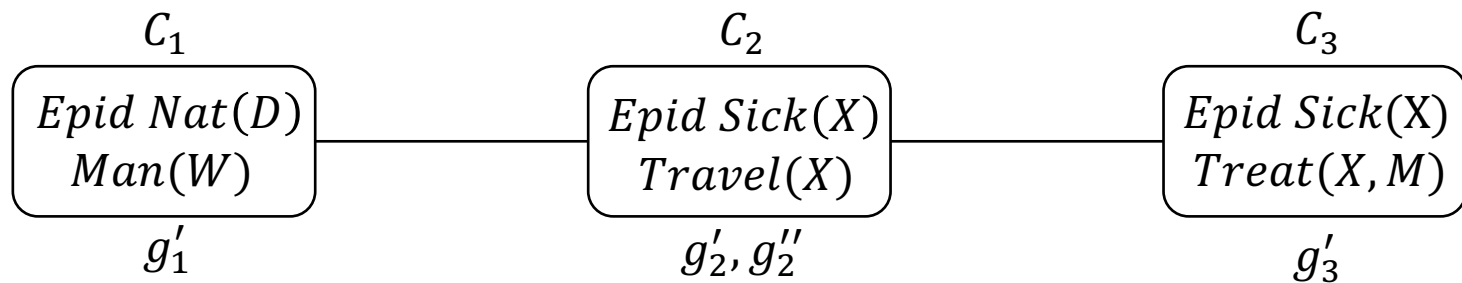


2. Enter evidence E into J incrementally

What about incremental changes?

Braun and Möller (2018)

- Input
 - Model G
 - Evidence $\mathbf{E} = \{Epid = true\} \cup \{Travel(eve) = true\}$
 - Queries $\mathbf{Q}$
- Algorithm
 1. Build FO jtree J for G
 2. Enter evidence $\mathbf{E}$ into J **adaptively**



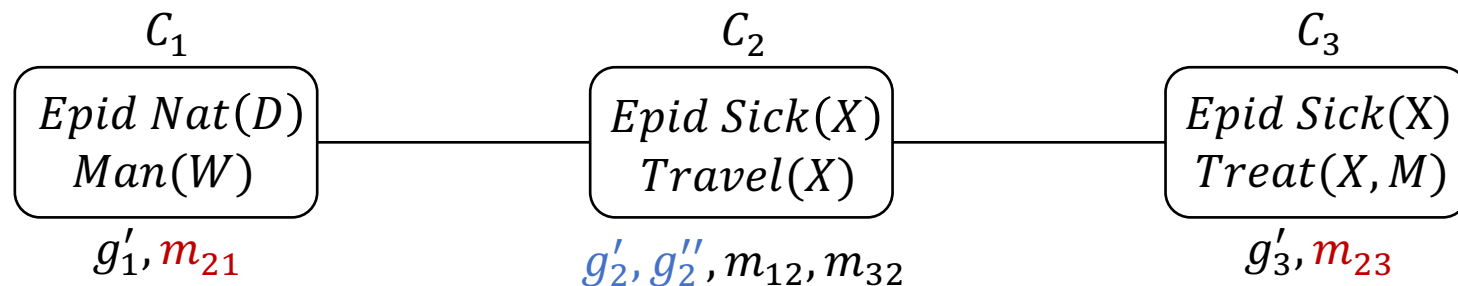
3. Pass messages **adaptively**

Adaptive Messages

Braun and Möller (2018)

- Calculate new message only if
 - Local model changed
 - Incoming message changed
 - Existing messages **invalid**
- Else
 - C_3 calculates new messages m_{21}, m_{23}
 - Send an empty message
 - Existing messages still valid

C_1, C_2 send empty messages



Changes: Adaptive Inference

Braun and Möller (2018)

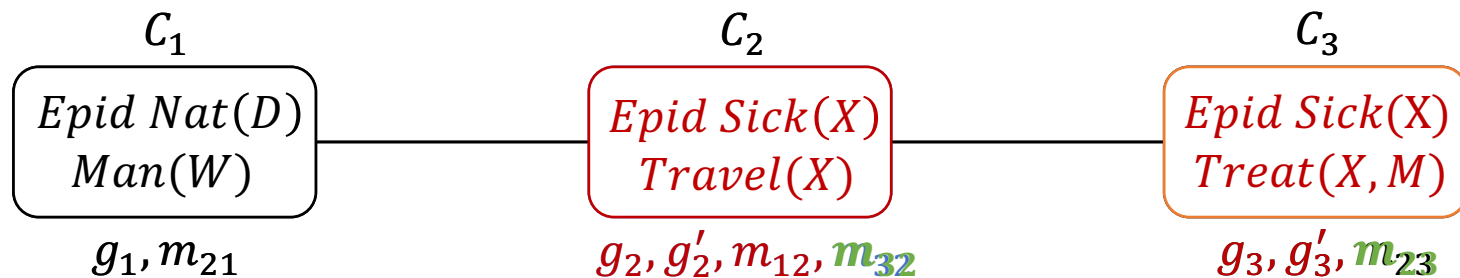
- Changes
 - Domains
 - Parfactors
 - Changing the structure: new/removed argument
 - Without changing the model structure: new range
 - Evidence (additional, retracted, new)
- Adaptive LJT
 - Adapt FO jtree J to changes in G
 - Adaptively enter evidence
 - Changes in evidence E
 - Changes in J
 - Adaptively pass messages
 - Answer queries

J may require
reconstruction
after a while

LJT: Conjunctive queries?

Koller and Friedman (2009), Braun and Möller (2017a)

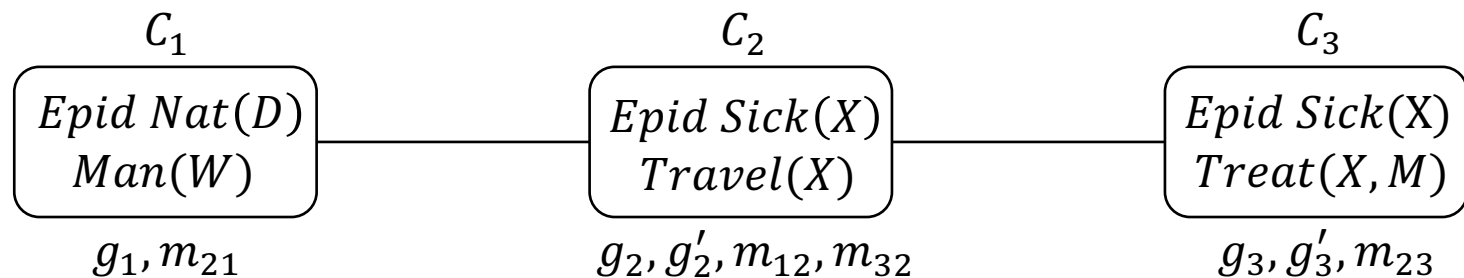
- **Conjunctive** query: $Epid \wedge Travel(eve)$
 - QA as before: Eliminate all non-query variables
- **Conjunctive** query: $Travel(eve) \wedge Treat(eve, m_1)$
 - **Query variables not in one parcluster!**
 - **Duplicate information in messages!**



LJT: Answer Conjunctive Queries

Koller and Friedman (2009), Braun and Möller (2017a)

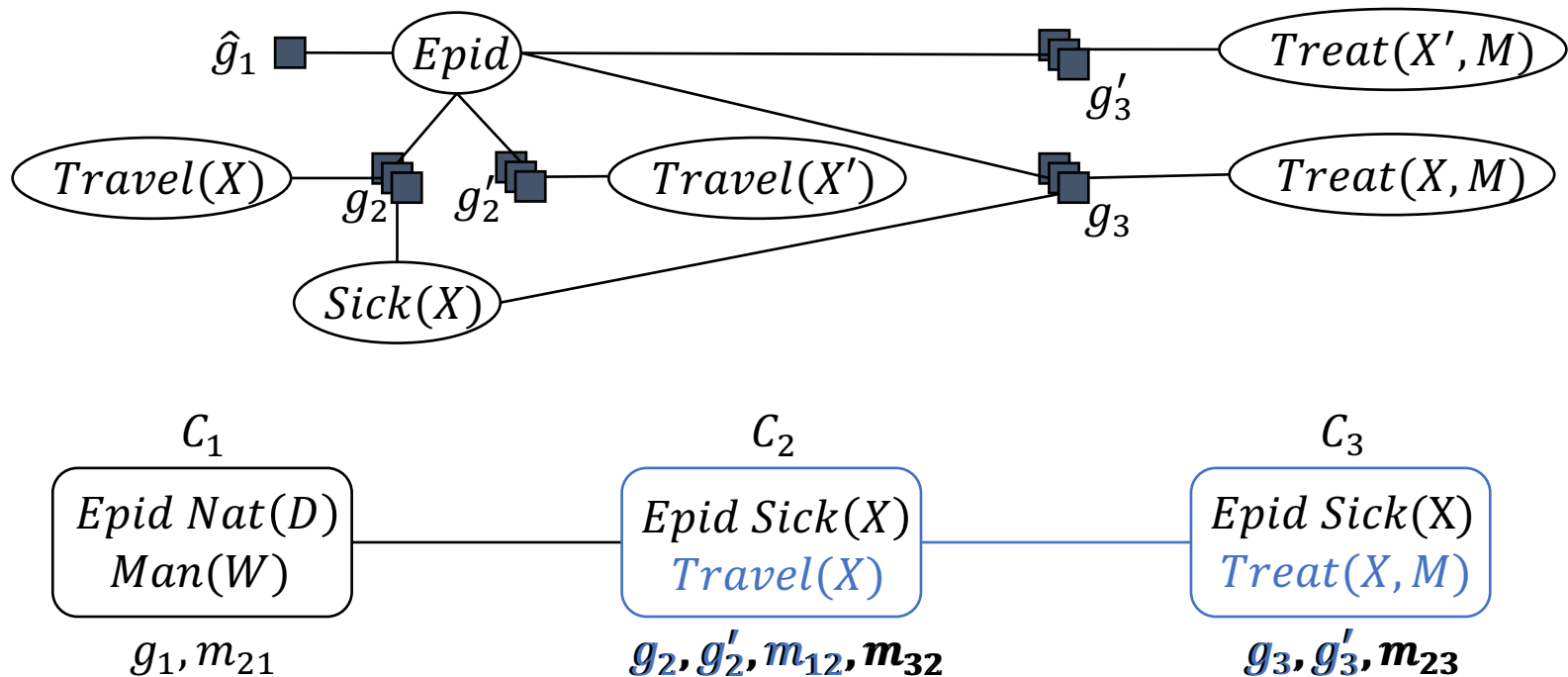
- Queries $Q = \{Q_1, Q_2, Q_3, Q_4\}$
 - $Q_1 = \text{Travel}(\text{eve}) \wedge \text{Treat}(\text{eve}, m_1)$
 - $Q_2 = \text{Epid} \wedge \text{Travel}(\text{eve})$
 - $Q_3 = \text{Travel}(\text{eve})$
 - $Q_4 = \text{Epid}$
- For each query Q
 - Find **subtree** that covers the query variables
 - Extract **submodel** without duplicate information
 - Use LVE to answer query



LJT: Answer Conjunctive Queries

Braun and Möller (2017a)

- $Q_1 = \textit{Travel}(\textit{eve}) \wedge \textit{Treat}(\textit{eve}, m_1)$
 - Find **subtree** that covers the query variables
 - Extract **submodel** without duplicate information
 - Use LVE to answer query

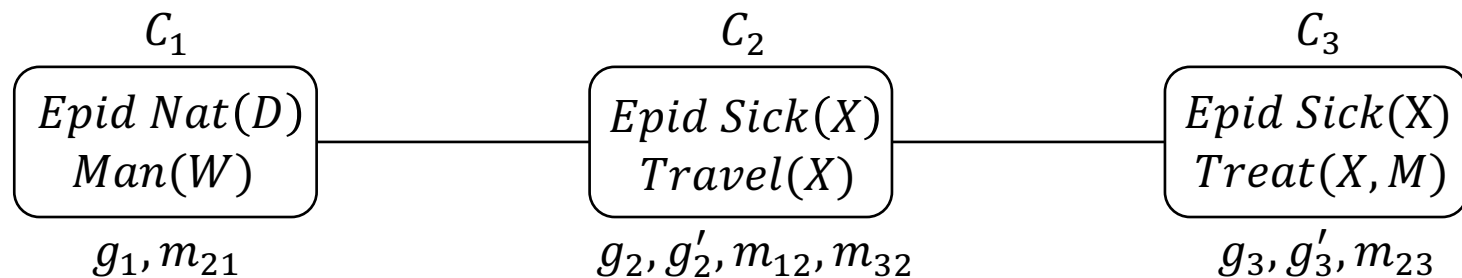


LJT: Answer Conjunctive Queries

Braun and Möller (2017a)

- Queries $Q = \{Q_1, Q_2, Q_3, Q_4\}$
 - $Q_1 = Travel(eve) \wedge Treat(eve, m_1)$
 - $Q_2 = Epid \wedge Travel(eve)$
 - $Q_3 = Travel(eve)$
 - $Q_4 = Epid$

} Subtree = Single parcluster
- For each query Q
 - Find **subtree** that covers the query variables
 - Extract **submodel** without duplicate information
 - Use LVE to answer query



QA: Most probable assignment

Dawid (1992), Dechter (1999), de Salvo Braz et al. (2006),
Apsel and Brafman (2012), Braun and Möller (2018b)

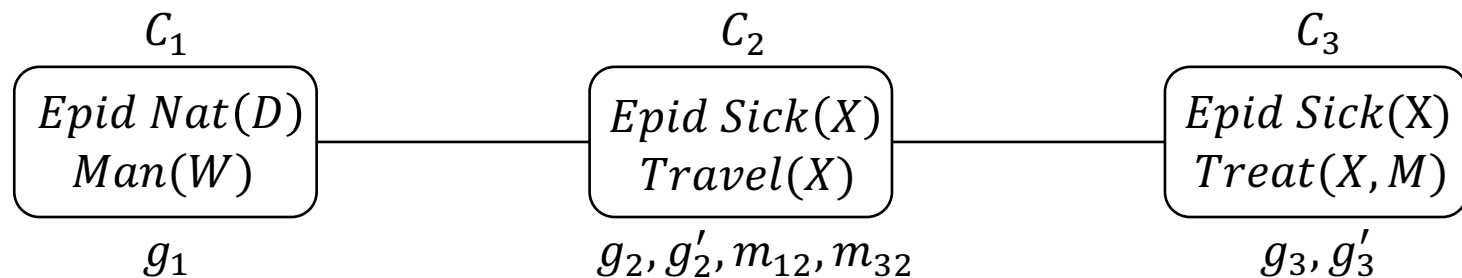
- To all variables: most probable explanation (**MPE**)

$$\operatorname{argmax}_{rv(G)} P_G$$

- (L)VE: **maxing out** (instead of summing out)
- (L)JT: message calculation by maxing out
 - **Isomorphic instances: identical most probable assignment**
- As before: construction, evidence entering
- Message passing
 - Only **one message pass** (from periphery to centre)
 - Max out remaining variables at centre

LJT: MPE

- $Epid = false$
- $\forall D, W : Nat(D) = false, Man(W) = false$
- $\forall X' \in \{alice, eve\}, P : Travel(X') = true, Treat(X', P) = true$
- $\forall X, M : Travel(X) = true, Sick(X) = true, Treat(X, M) = true$



QA: Most probable assignment

Dechter (1999), Sharma et al. (2018)
Braun and Möller (2018b)

- To subset of variables: maximum a posteriori (**MAP**)

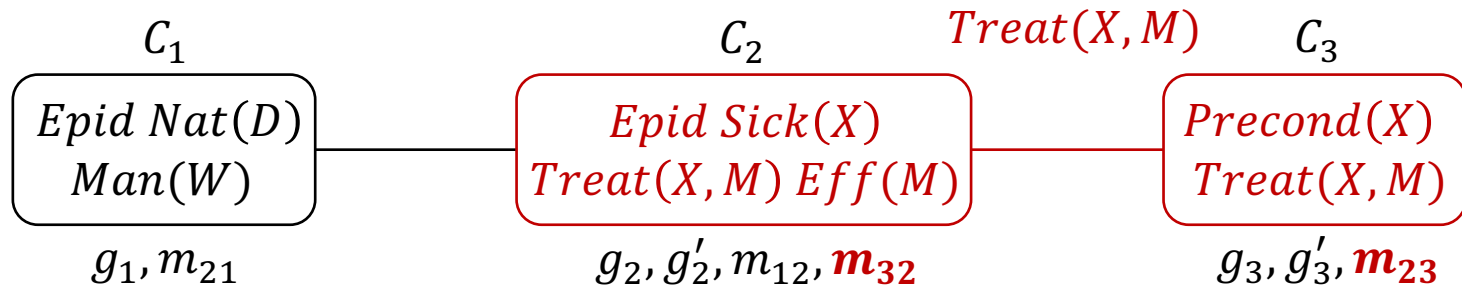
$$\operatorname{argmax}_{\text{Travel.eve,Epid}} \sum_{r(\text{Nat}(D), \text{Man}(W), \text{Sick}(X), \text{Treat}(X,P)), \\ r(\text{Travel}(X)), X \neq \text{eve}} P_G$$

- *MAP a more general case of MPE*
- **$\sum$ and max not commutative!**
- As before: construction, evidence entering, message passing (with summing out)
- Answer MAP query: get submodel (as before)
 - Sum out non-query variables, max out query variables
- Assignment to **complete parclusters** safe

Does LJT lift what is liftable?

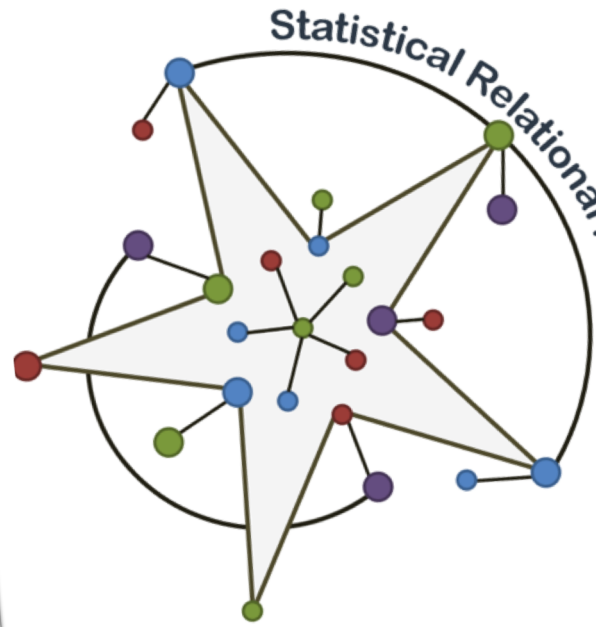
Braun and Möller (2017)

- Original LJT had avoidable groundings



- After **fusion**, answer is **YES!**





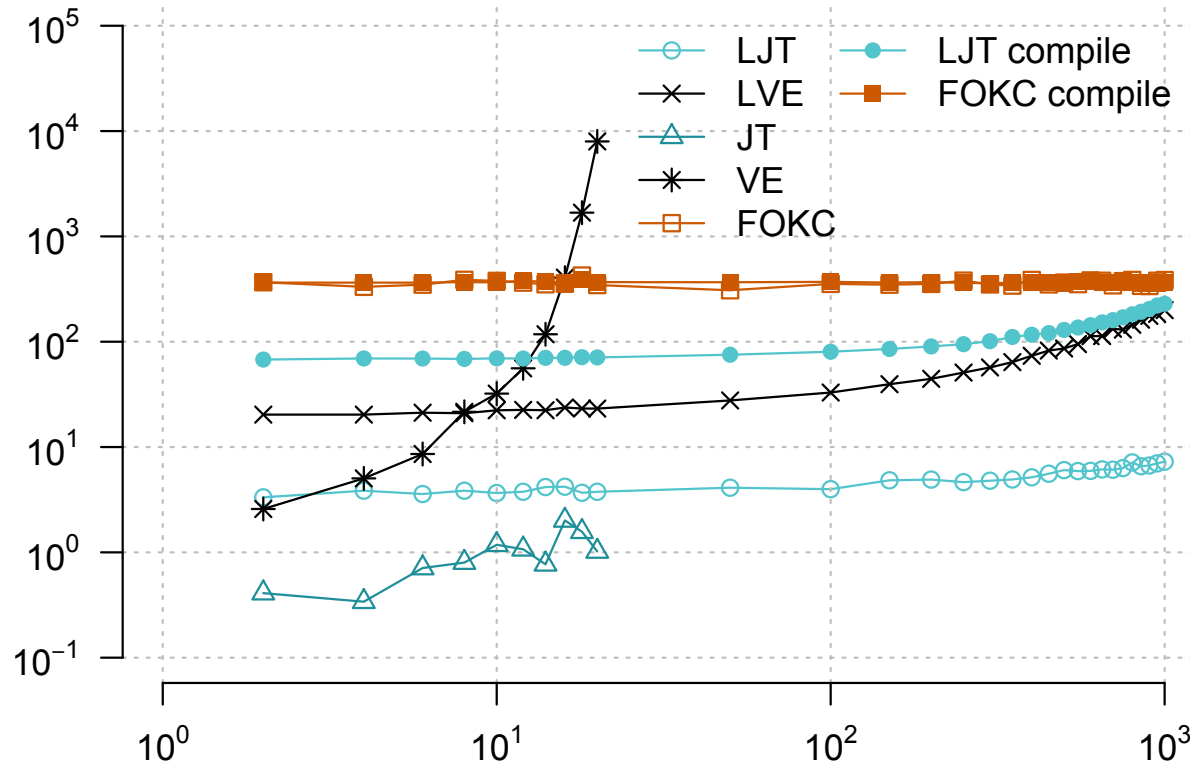
Does it work in practice?

Prototype implementation

LVE by Taghipour <https://dtai.cs.kuleuven.be/software/gcfove>

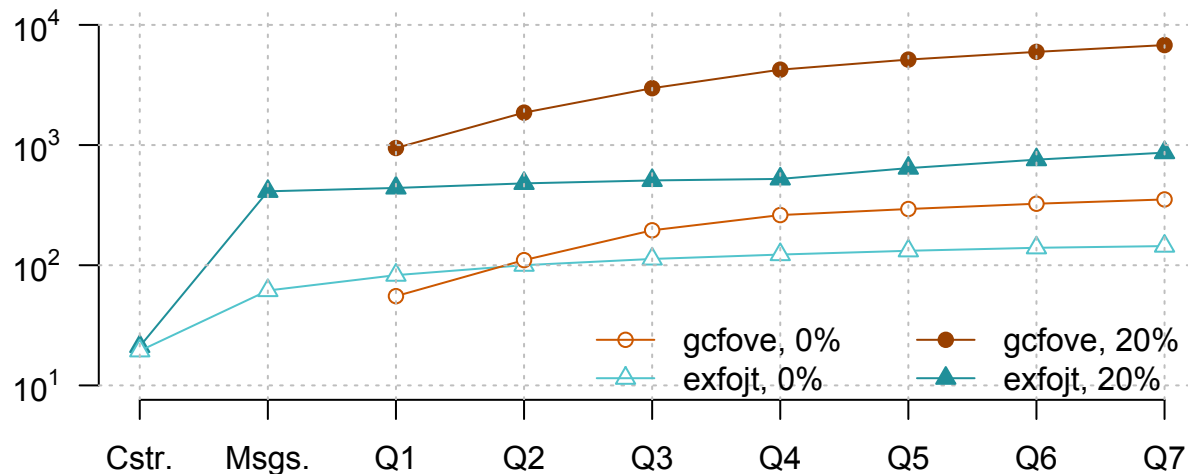
LJT <https://www.ifis.uni-luebeck.de/index.php?id=518>

Grounded vs. Lifted Inference



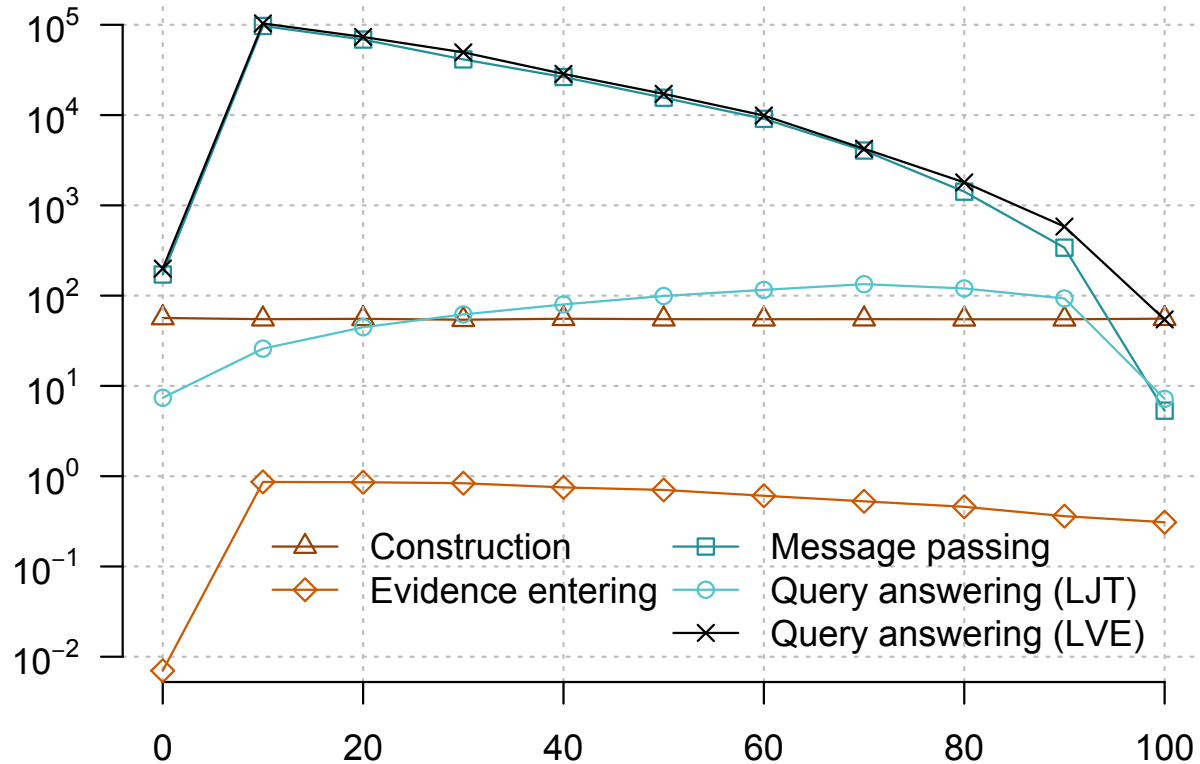
Runtimes in milliseconds, one query,
domain sizes between 2 and 1000,
evidence at 0%

LJT: Payoff versus LVE



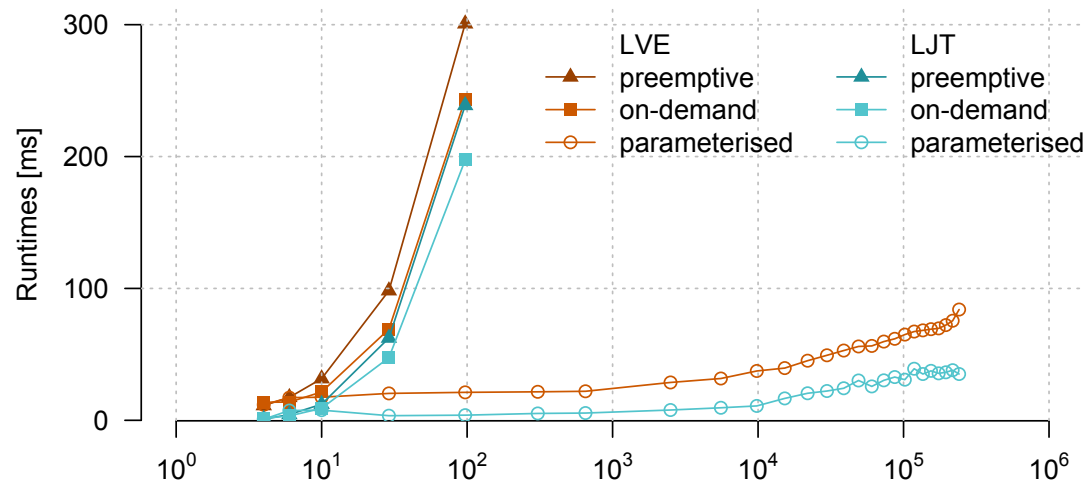
Accumulated runtimes in milliseconds , seven queries, grounded model size: 100,000
Evidence at 0% and 20% on PRVs with one parameter

LJT: Evidence

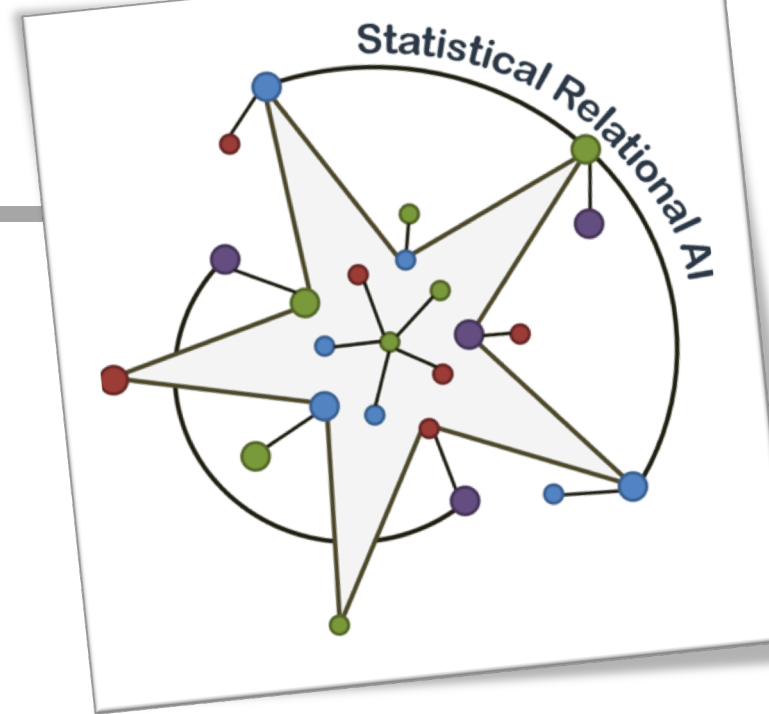


Runtimes in milliseconds for each LJT step
Evidence from 0% to 100% in 10% steps on PRVs with one parameter

LJT: Parameterised Queries



Runtimes in milliseconds, one query, grounded model size between 10 and 100,000



Appendix

FO jtree construction

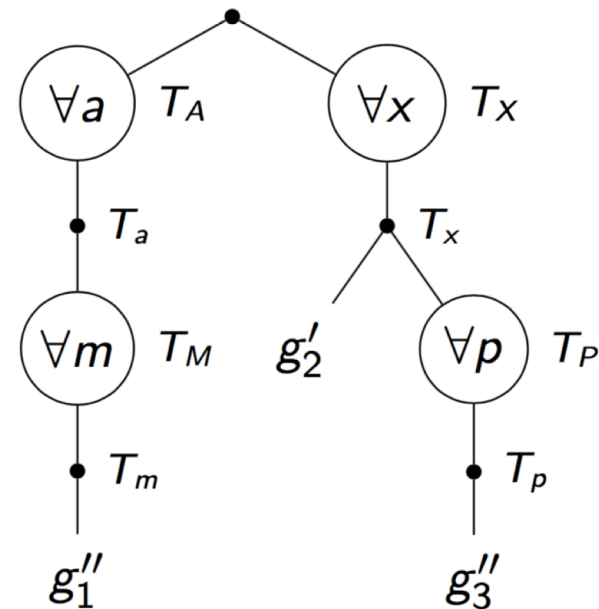
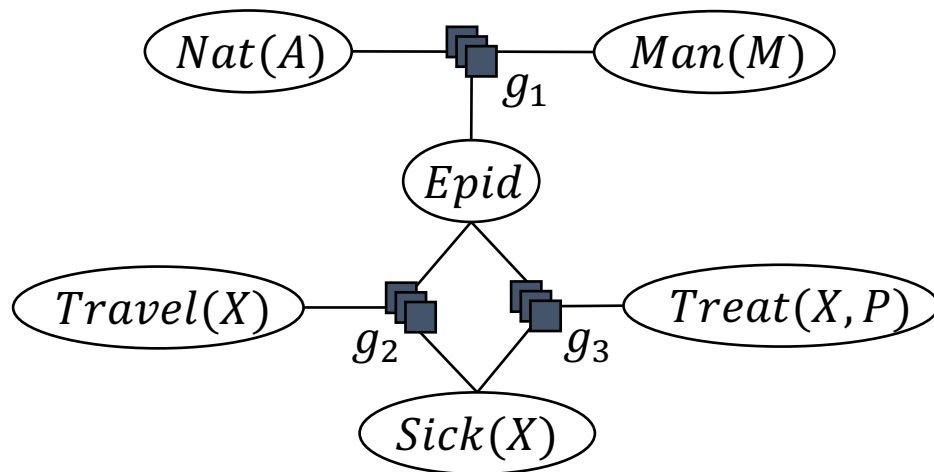
FO Jtree Construction

- Propositional junction tree construction
 - Triangulation, compute maximum spanning tree, ...
 - Hypergraph partitioning
 - Decomposition tree (dtree), clusters, ...
- First-order: logical variables Taghipour et al. (2013b)
 - First-order decomposition trees (**FO dtrees**)
 - FO dtrees have node properties (cutset, context, **cluster**)
 - (FO) dtree + clusters = (FO) jtree
 - Heuristic to build an FO dtree
(logical variables guide the construction)

FO Dtree: Data Structure

Taghipour et al. (2013b)

- (L)VE decomposes model into subproblems
- Represent as tree



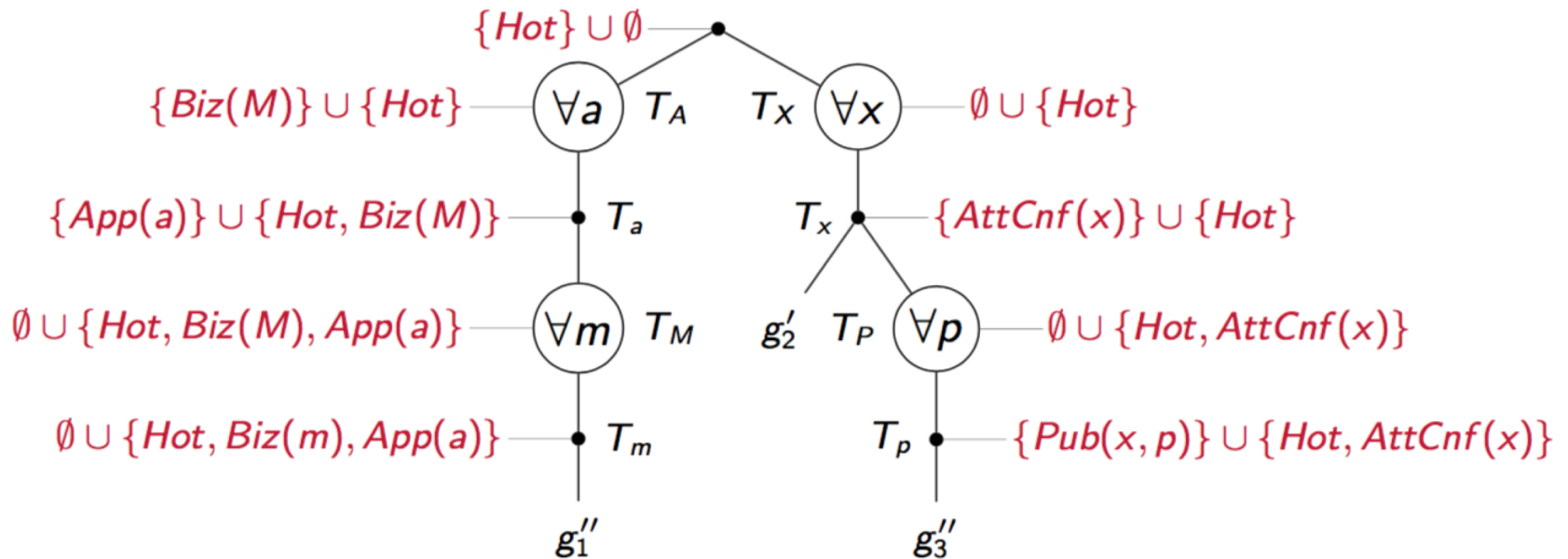
FO Dtree: Cutset, Context, Cluster

Taghipour et al. (2013b)

$$\text{cutset}(T) = \bigcup_{T_i, T_j \in \text{child}(T)} RV(T_i) \cap RV(T_j) \setminus \text{acutset}(T), \text{acutset}(T) = \bigcup_{T' \in \text{ancestor}(T)} \text{cutset}(T')$$

$$\text{context}(T) = RV(T) \cap \text{acutset}(T), RV(T) = \bigcup_{T' \in \text{child}(T)} RV(T'), RV(L) = RV(\phi_L), L \text{ leaf}$$

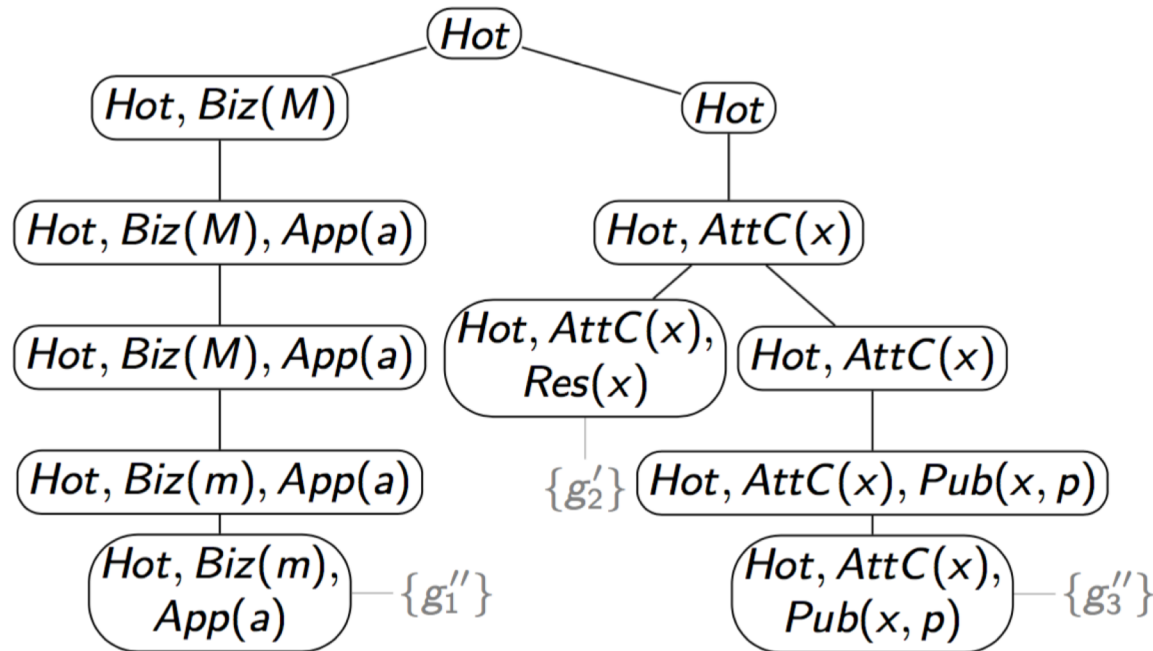
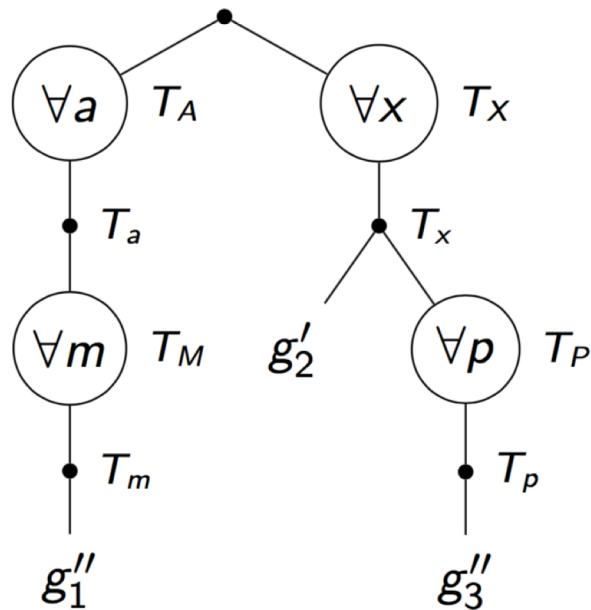
$$\text{cluster}(T) = \text{cutset}(T) \cup \text{context}(T), \text{cluster}(L) = RV(\phi_L), L \text{ leaf}$$



FO Dtree to FO Jtree

Braun and Möller (2016)

- Compute clusters per node and **minimise**



Minimising an FO Jtree

Braun and Möller (2016)

- FO jtree is **minimal**
 - If by removing a variable from any parcluster, the FO jtree stops being an FO jtree
- **Merge** parclusters
 - If neighbouring parclusters are subsets of each other

