Dynamic StarAI
Answering Continuous Queries

Tutorial at KI 2019

Tanya Braun, Marcel Gehrke, Ralf Möller
Universität zu Lübeck
Agenda: Dynamic Models and Statistical Relational AI

- Probabilistic relational models (PRMs) (Ralf)
- Answering static queries (Tanya)
- **Answering continuous queries** (Marcel)
  - Lifted Dynamic Junction Tree Algorithm (LDJT)
  - Relational interfaces
  - Taming reasoning w.r.t. lots of evidence over time
- Take home messages (Ralf)
  - LJT and LDJT research relevant for all variants of PRMs

Goal: Overview of central ideas
Lifted: Dynamic Model

- Marginal distribution query: $P(A^i_\pi | E_{0:t})$ w.r.t. the model:
  - Hindsight: $\pi < t$ (was there an epidemic $t - \pi$ days ago?)
  - Filtering: $\pi = t$ (is there an currently an epidemic?)
  - Prediction: $\pi > t$ (is there an epidemic in $\pi - t$ days?),

Gehrke et al. (2018)
Lifted Dynamic Junction Tree Algorithm: LDJT

Gehrke et al. (2018)

• **Input**
  - Temporal model $G$
  - Evidence $E$
  - Queries $Q$

• **Algorithm**
  1. Identify interface variables
  2. Build FO jtree structures $J$ for $G$
  3. Instantiate $J_t$
  4. Restore state description of interface variables from $m_{t-1}$
  5. Enter evidence $E_t$ into $J_t$
  6. Pass messages in $J_t$
  7. Answer queries $Q_t$
  8. Store state description of interface variables in $m_t$
  9. Proceed to next time step (step 3)
LDJT: Identify Interface Variables

- Use temporal conditional independences to perform inference on smaller model (Murphy (2002))

- $I_{t-1} = \{ A^i_{t-1} | \exists \phi(A) | C \in G : A^i_{t-1} \in \mathcal{A} \land A^j_{t-1} \in \mathcal{A} \}$

- Set of interface variable $I_{t-1}$ consists of all PRVs from time slice $t - 1$ that occur in a parfactor with PRVs from time slice $t$
LDJT: Construct FO jtree Structure

- Turn model in 1.5 time slice model
- Suffices to perform inference over time slice $t$
- From 1.5 time slice model construct FO jtree structure
- Ensure $I_{t-1}$ is contained in a parcluster and $I_t$ is contained in a parcluster
- Label parcluster with $I_{t-1}$ as in-cluster and parcluster with $I_t$ as out-cluster

Gehrke et al. (2018)
LDJT: Query answering

- Instantiate FO jtree structure
- Restore state description of interface variables
- Enter evidence
- Pass messages
- Query answering:
  - Find parcluster contain query term
  - Extract submodel
  - Answer query with LVE

Gehrke et al. (2018)
LDJT: Proceed in time

- Calculate $m_3$ using out-cluster ($C_3^2$)
- Eliminate $Travel(X)_3$ from $C_3^2$'s local model
- Instantiate next FO jtree and enter $m_3$
- Enter evidence and pass messages

Gehrke et al. (2018)
LDJT: Intermediate Overview

- So far only a temporal forward pass
- Reason over one time step
- Keep only one time step in memory
- Filtering queries
- Prediction queries (filtering without new evidence)
- Hindsight queries

Gehrke et al. (2018)
LDJT: Forward and Backward Pass

• Use same FO jtrees for backward pass
• Calculate a message $n$ using an in-cluster over interface variables and pass $n$ to previous time step
• LDJT needs to keep FOjtrees of previous time steps
• Different instantiation approaches during a backward pass
  • Keep all computations for all time steps in memory (not always feasible)
  • Instantiate time steps on demand (same as for the forward pass, possible due to the separation between time steps)

Gehrke et al. (2019)
LDJT: Backward Pass

- Calculate $n_t$ using in-cluster ($C_t^1$)
- Eliminate $\text{Epid}_t$ from $C_t^1$’s local model, without $m_{t-1}$
- Add $n_t$ to local model of out-cluster $C_{t-1}^2$
- Pass messages for $t-1$ to account for $n_t$

Gehrke et al. (2019)
LDJT: Instantiations during a Backward Pass

In-cluster $C^1_{t-1}$

- $Epid_{t-2}$ Sick($X$)$_{t-2}$  $Epid_{t-1}$
- $g^E$  $m_{t-2}$ $m^{21}$

Out-cluster $C^2_{t-1}$

- $Epid_{t-1}$ Sick($X$)$_{t-1}$ Travel($X$)$_{t-1}$
- $g^2_{t-1}m^{12}$ $m^{32}$ $n_t$

Gehrke et al. (2019)

In-cluster $C^1_t$

- $Epid_{t-1}$ Sick($X$)$_{t-1}$
- $Epid_t$

Out-cluster $C^2_t$

- $Epid_t$ Sick($X$)$_t$

- $Travel(X)_t$

- $g^E_t$  $m_{t-1}m^{21}$

- $n_t$

<table>
<thead>
<tr>
<th>Keep Instantiations</th>
<th>Instantiate on demand</th>
</tr>
</thead>
<tbody>
<tr>
<td>Messages to prepare for queries</td>
<td></td>
</tr>
<tr>
<td>Messages to solely calculate $n_{t-1}$</td>
<td></td>
</tr>
<tr>
<td>Additional memory for each time step</td>
<td></td>
</tr>
</tbody>
</table>
## LDJT: Instantiations during a Backward Pass

<table>
<thead>
<tr>
<th>In-cluster $C_{t-1}^1$</th>
<th>Out-cluster $C_{t-1}^2$</th>
<th>Gehrke et al. (2019)</th>
</tr>
</thead>
<tbody>
<tr>
<td>$Epid_{t-2}$ Sick($X$)$_{t-2}$</td>
<td>$Epid_{t-1}$ Sick($X$)$<em>{t-1}$ Travel($X$)$</em>{t-1}$</td>
<td>$Epid_t$ Sick($X$)$_t$ Travel($X$)$_t$</td>
</tr>
<tr>
<td>$Epid_{t-1}$</td>
<td>$Epid_{t-1}$ Sick($X$)$_{t-1}$ Treat($X, M$)$_t$</td>
<td>$Epid_t$ Sick($X$)$_t$ Treat($X, M$)$_t$</td>
</tr>
<tr>
<td>(g^E)</td>
<td>(m_{t-2} m^{21})</td>
<td>(g_t^2 m^{12} m^{32} n_t)</td>
</tr>
<tr>
<td>$m_t^3 m^{23}$</td>
<td>$m_{t-1} m^{21}$</td>
<td>$m_t^3 m^{23}$</td>
</tr>
</tbody>
</table>

### Keep Instantiations

<table>
<thead>
<tr>
<th>Messages to prepare for queries</th>
<th>$n - 1$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Messages to solely calculate $n_{t-1}$</td>
<td>$n - 1$</td>
</tr>
<tr>
<td>Additional memory for each time step</td>
<td>$n - 1$</td>
</tr>
</tbody>
</table>

\(n\) is the number of parclusters for each time step
LDJT: Instantiations during a Backward Pass

In-cluster $C_{t-1}^1$  

Out-cluster $C_{t-1}^2$

\[
\begin{array}{c}
\text{In-cluster } C_{t-1}^1 \\
\text{Out-cluster } C_{t-1}^2 \\
\text{In-cluster } C_t^1 \\
\text{Out-cluster } C_t^2
\end{array}
\]

Gehrke et al. (2019)

\[
\begin{array}{c}
\text{In-cluster } C_{t-1}^1 \\
\text{Out-cluster } C_{t-1}^2 \\
\text{In-cluster } C_t^1 \\
\text{Out-cluster } C_t^2
\end{array}
\]

\[
\begin{align*}
\text{In-cluster } C_{t-1}^1 & \quad \text{Out-cluster } C_{t-1}^2 \\
\text{In-cluster } C_t^1 & \quad \text{Out-cluster } C_t^2
\end{align*}
\]

Keep Instantiations  Instantiate on demand

<table>
<thead>
<tr>
<th>Messages to prepare for queries</th>
<th>$n - 1$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Messages to solely calculate $n_{t-1}$</td>
<td>$\leq n - 1$</td>
</tr>
<tr>
<td>Additional memory for each time step</td>
<td></td>
</tr>
</tbody>
</table>

$n$ is the number of parclusters for each time step
LDJT: Instantiations during a Backward Pass

<table>
<thead>
<tr>
<th>In-cluster $C_{t-1}^1$</th>
<th>Out-cluster $C_{t-1}^2$</th>
<th>Gehrke et al. (2019)</th>
</tr>
</thead>
<tbody>
<tr>
<td>$Epid_{t-2}$ $Sick(X)<em>{t-2}$ $Epid</em>{t-1}$</td>
<td>$Epid_{t-1} $ Sick(X)$<em>{t-1}$ Travel(X)$</em>{t-1}$</td>
<td>$Epid_{t}$ Sick(X)$<em>{t}$ Travel(X)$</em>{t}$</td>
</tr>
<tr>
<td>$g^E$ $m_{t-2}$ $m^{21}$</td>
<td>$g_{t-1}^2 m^{12}$ $m^{32}$ $n_t$ $C_{t-1}^3$</td>
<td>$g_t^2$ $m^{12}$ $m^{32}$ $C_t^3$</td>
</tr>
<tr>
<td>$Epid_{t-1}$ Sick(X)$<em>{t-1}$ Treat(X,M)$</em>{t}$</td>
<td>$n_t$</td>
<td>$Epid_{t}$ Sick(X)$<em>{t}$ Treat(X,M)$</em>{t}$</td>
</tr>
<tr>
<td>$g_{t-1}^3 m^{23}$</td>
<td>$m_{t-1}$</td>
<td>$g^E$ $m_{t-1} m^{21}$</td>
</tr>
</tbody>
</table>

### Keep Instantiations vs Instantiate on demand

| Messages to prepare for queries | $n - 1$ |
| Messages to solely calculate $n_{t-1}$ | $\leq n - 1$ |
| Additional memory for each time step | All local models |

$n$ is the number of parclusters for each time step
LDJT: Instantiations during a Backward Pass

In-cluster $C_{t-1}^1$

- $Epid_{t-2}$ Sick($X$)$_{t-2}$
- $Epid_{t-1}$

Out-cluster $C_{t-1}^2$

- $Epid_{t-1}$ Sick($X$)$_{t-1}$
- Travel($X$)$_{t-1}$

$g^E_m m_{t-2} m_{t}^2$

In-cluster $C_t^1$

- $Epid_{t-1}$ Sick($X$)$_{t-1}$
- Travel($X$)$_{t}$

Out-cluster $C_t^2$

- $Epid_{t}$ Sick($X$)$_{t}$
- Travel($X$)$_{t}$

$g_t^{3} m_{t}^{23}$

$g_{t-1}^{2} m_{t-1}^{12} m_{t}^{32} n_t$

$g_{t-1}^{3} m_{t}^{23}$

$g_{t-1}^{3} m_{t}^{23}$

$g_{t}^{2} m_{t}^{12} m_{t}^{32}$

$g_{t}^{3} m_{t}^{23}$

**Keep Instantiations**

- Messages to prepare for queries
- Messages to solely calculate $n_{t-1}$
- Additional memory for each time step: $n - 1$

**Instantiate on demand**

- $2 * (n - 1)$
- $n - 1$
- $\leq n - 1$
- All local models

$n$ is the number of parclusters for each time step
LDJT: Instantiations during a Backward Pass

<table>
<thead>
<tr>
<th>In-cluster $C_{t-1}^1$</th>
<th>Out-cluster $C_{t-1}^2$</th>
<th>Gehrke et al. (2019)</th>
</tr>
</thead>
<tbody>
<tr>
<td>$Epid_{t-2}$ $Sick(X)_{t-2}$</td>
<td>$Epid_{t-1}$ $Sick(X)<em>{t-1}$ $Travel(X)</em>{t-1}$</td>
<td>$Epid_t$ $Sick(X)_t$ $Travel(X)_t$</td>
</tr>
<tr>
<td>$g^E$ $m_{t-2}$ $m^{21}$</td>
<td>$g^E_{t-1}m^{12}$ $m^{32} n_t$ $g^E m_{t-1} m^{21}$</td>
<td>$g^E_t$ $m^{12}$ $m^{32}$ $g^E_{t-1}m^{12}$ $m^{32}$</td>
</tr>
<tr>
<td>$Epid_{t-1}$ $Sick(X)_{t-1}$ $Treat(X,M)_t$</td>
<td>$Epid_{t-1}$ $Sick(X)_{t-1}$ $Treat(X,M)_t$</td>
<td>$Epid_t$ $Sick(X)_t$ $Treat(X,M)_t$</td>
</tr>
<tr>
<td>$g^E_{t-1}m^{23}$</td>
<td>$g^E_{t-1}m^{23}$</td>
<td></td>
</tr>
</tbody>
</table>

Keep Instantiations | Instantiate on demand

| Messages to prepare for queries | $n - 1$ | $2 \cdot (n - 1)$ |
| Messages to solely calculate $n_{t-1}$ | $\leq n - 1$ | $n - 1$ |
| Additional memory for each time step | All local models | |

$n$ is the number of parclusters for each time step
LDJT: Instantiations during a Backward Pass

In-cluster $C_{t-1}^1$

Epid$_{t-2}$ Sick($X$)$_{t-2}$

Epid$_{t-1}$

$g^E$ $m_{t-2} m^{21}$

Out-cluster $C_{t-1}^2$

Epid$_{t-1}$ Sick($X$)$_{t-1}$

Travel($X$)$_{t-1}$

$g_{t-1}^2 m^{12} m^{32} n_t$

In-cluster $C_t^1$

Epid$_{t-1}$ Sick($X$)$_{t-1}$

Travel($X$)$_{t}$

$g_{t-1}^3 m^{23}$

Gehrke et al. (2019)

Out-cluster $C_t^2$

Epid$_t$ Sick($X$)$_t$

$g_t^2 m^{12} m^{32}$

$C_t^3$

Epid$_t$ Sick($X$)$_t$

Travel($X$)$_{t}$

$g_t^3 m^{23}$

---

**Keep Instantiations**

Messages to prepare for queries $n - 1$

Messages to solely calculate $n_{t-1}$ $\leq n - 1$

Additional memory for each time step All local models

**Instantiate on demand**

$n - 1$

$2 \cdot (n - 1)$

$n - 1$

Only forward ($m_t$) messages

---

$n$ is the number of parclusters for each time step
LDJT: Relational Forward Backward Algorithm

Gehrke et al. (2019)

• LDJT can answer hindsight queries, even to the first time step
• By combining the instantiation approaches, LDJT can trade off memory consumption and reusing computations
• LDJT is in the worst case quadratic to T, but normally remains linear w.r.t. T (T max # time steps)
• But does it really suffice to lift the interface algorithm?
LDJT: Preventing Unnecessary Groundings

• Groundings in inter time slice messages (especially forward messages) can lead to grounding the model for all time steps

• Elimination order predetermined in FO jtree

• Non-ideal elimination order leads to groundings
  • Minimal set of interface variables not always ideal
  • Delay eliminations for inter time slice messages to prevent unnecessary groundings
  • Simply lifting the interface algorithm does not suffice, one also needs to ensure preconditions of lifting

• Trade off between lifting and handling temporal aspects due to restrictions on elimination orders

Gehrke et al. (2018b,c)
LDJT: Preventing Unnecessary Groundings

- Depending on the settings, either lifting or handling of temporal aspects is more efficient
- Preventing groundings to calculate a lifted solution pays off
LDJT: Theoretical Analysis

• FO² is not always liftable in temporal models
  • There exists an FO² for which LDJT has to ground
  • Unrolling would allow for a lifted solution
  • Handling temporal aspects restricts elimination order
• Lifting makes the problem manageable
  • Ground width grows with instances in interface
  • Lifted width remains the same
  • Runtime exponential to width
LDJT: Additional Queries

• Conjunctive queries over different time steps (Gehrke et al. (2018 d))
  • Can be used for event detection
  • What is the probability that someone travelled from X to Y and that afterwards there is a epidemic in Y given there is an epidemic in X?

• Maximum expected utility (Gehrke et al. (2019 b,c))
  • Decision support
  • Well studied within one time step (Apsel and Brafman (2011), Nath and Domingos (2009))

• Assignment queries (Gehrke et al. (2019 d))
  • Most likely state sequence
  • Well studied for static models (Dawid (1992), Dechter (1999), de Salvo Braz et al. (2006), Apsel and Brafman (2012), Braun and Möller (2018))
Taming Reasoning

• Evidence can ground a model over time
• Non-symmetric evidence
  • Observe evidence for some instances in one time step
  • Observe evidence for a subset of these instances in another time step
  • Split the logical variable slowly over time
• Vanilla junction trees for each time step
• Forward message carries over splits, leading to slowly grounding a model over time
Evidence over Time

• $D_3(x_1) = true$

• Split $g_3^2$ into
  • $g_3^{2'}$ for $x_1$ and
  • $g_3^{2''}$ for $X \neq x_1$

• $m_3$ consists of
  • $m^{12}$
  • $m^{32}$
  • $g_3^{2'}$ and $g_3^{2''}$ with $D_3(X)$ eliminated

Gehrke et al. (2019e)
Evidence over Time

• $D_4(x_2) = true$

• Split $g_4^2$ into
  • $g_4^2'$ for $x_2$ and
  • $g_4^2''$ for $X \neq x_2$

• $m_4$ consists of
  • $m^{12}$ (containing $m_3$)
  • $m^{32}$
  • $g_4^2'$ and $g_4^2''$, with $D_4(X)$ eliminated

Gehrke et al. (2019e)
Undoing Splits

• Need to undo splits to keep reasoning polynomial w.r.t. domain sizes
• Where can splits be undone efficiently?
• How to undo splits?
• Is it reasonable to undo splits?

Gehrke et al. (2019e)
Where Can Splits Be Undone Efficiently?

- Evidence causes splits in a logical variable in the same way in all factors in a model
- LDJT always instantiates a vanilla junction tree
- Forward message carries over splits
How to Undo Splits?

• The colouring algorithm (Ahmadi et al. 2013) can efficiently identify exact symmetries
• Evidence causes differences in distributions
• Need to find approximate symmetries to undo splits caused by evidence
• Need a way to merge factors
Comparing Factors

- Comparing all marginals is expensive
- Comparing marginals of a subset of random variables can determine non-similar factors similar

<table>
<thead>
<tr>
<th>R(X)</th>
<th>A(X)</th>
<th>f</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td>false</td>
<td>0</td>
</tr>
<tr>
<td>false</td>
<td>true</td>
<td>7</td>
</tr>
<tr>
<td>true</td>
<td>false</td>
<td>4</td>
</tr>
<tr>
<td>true</td>
<td>true</td>
<td>1</td>
</tr>
</tbody>
</table>

\[ P(A(x_1 = true)) = \frac{2}{3}, \quad P(R(x_1 = true)) = \frac{5}{12} \]

Gehrke et al. (2019e)
Comparing Factors

- Potentials determine distributions
- Similar ratios in potentials lead to similar marginals and similar factors

\[
\begin{array}{|c|c|c|}
\hline
R(X) & A(X) & f \\
\hline
false & false & 4 \\
false & true & 3 \\
true & false & 2 \\
true & true & 1 \\
\hline
\end{array}
\]

\[
\begin{array}{|c|c|c|}
\hline
R(X) & A(X) & f \\
\hline
false & false & 3.9 \\
false & true & 3.1 \\
true & false & 2.1 \\
true & true & 0.9 \\
\hline
\end{array}
\]

- \( P(A(x_1 = true))): \quad \frac{4}{10}
- \( P(R(x_1 = true))): \quad \frac{3}{10}
- \( P(A(x_1 = true) \land R(x_1 = true))): \quad \frac{1}{10}
- \( |D(X)| = 1 \)
Find Approximate Symmetries

• Cosine similarity for similarity between vector

\[
\cos(\theta) = \frac{\sum_{i=1}^{n} A_i \cdot B_i}{\sqrt{\sum_{i=1}^{n} A_i^2} \cdot \sqrt{\sum_{i=1}^{n} B_i^2}}
\]

<table>
<thead>
<tr>
<th>(R(X))</th>
<th>(A(X))</th>
<th>(f)</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td>false</td>
<td>0</td>
</tr>
<tr>
<td>false</td>
<td>true</td>
<td>7</td>
</tr>
<tr>
<td>true</td>
<td>false</td>
<td>4</td>
</tr>
<tr>
<td>true</td>
<td>true</td>
<td>1</td>
</tr>
</tbody>
</table>

\[
\cos(\theta) = \frac{0 \cdot 2 + 7 \cdot 4 + 4 \cdot 2 + 1 \cdot 4}{\sqrt{0 + 49 + 16 + 1} \cdot \sqrt{4 + 16 + 4 + 16}} \approx 0.7785
\]

Gehrke et al. (2019e)
Find Approximate Symmetries

- Cosine similarity for similarity between vector

\[
\cos(\theta) = \frac{\sum_{i=1}^{n} A_i \cdot B_i}{\sqrt{\sum_{i=1}^{n} A_i^2} \cdot \sqrt{\sum_{i=1}^{n} B_i^2}}
\]

<table>
<thead>
<tr>
<th>R(X)</th>
<th>A(X)</th>
<th>f</th>
</tr>
</thead>
<tbody>
<tr>
<td>false</td>
<td>false</td>
<td>4</td>
</tr>
<tr>
<td>false</td>
<td>true</td>
<td>3</td>
</tr>
<tr>
<td>true</td>
<td>false</td>
<td>2</td>
</tr>
<tr>
<td>true</td>
<td>true</td>
<td>1</td>
</tr>
</tbody>
</table>

- \( \cos(\theta) = \frac{4 \cdot 3.9 + 3 \cdot 3.1 + 2 \cdot 2.1 + 1 \cdot 0.9}{\sqrt{16 + 9 + 4 + 1 \cdot \sqrt{15.21 + 9.61 + 4.41 + 0.81}}} \approx 0.9993 \)

Gehrke et al. (2019e)
Find Approximate Symmetries

• Cosine similarity for similarity between vector

\[
\cos(\theta) = \frac{\sum_{i=1}^{n} A_i \cdot B_i}{\sqrt{\sum_{i=1}^{n} A_i^2 \cdot \sum_{i=1}^{n} B_i^2}}
\]

\[
\begin{array}{|c|c|c|}
\hline
R(X) & A(X) & f \\
\hline
false & false & 4 \\
false & true & 3 \\
true & false & 2 \\
true & true & 1 \\
\hline
\end{array}
\]

• \( \cos(\theta) = \frac{4 \cdot 8 + 3 \cdot 6 + 2 \cdot 4 + 1 \cdot 3}{\sqrt{16 + 9 + 4 + 1 \cdot \sqrt{64 + 36 + 16 + 4}}} = 1 \)

• Cluster splits with 1-cos as distance function
# Merging Clusters

- Merge identified clusters based on distance function while accounting for groundings

| $|\mathcal{D}(X)| = 4$ | $|\mathcal{D}(X')| = 4$ | $|\mathcal{D}(X'')| = 2$ |
|---|---|---|
| $R(X)$ | $A(X)$ | $f$ | $R(X')$ | $A(X')$ | $f$ | $R(X'')$ | $A(X'')$ | $f$ |
| false | false | 4 | false | false | 7.9 | false | false | 15.7 |
| false | true | 3 | false | true | 6 | false | true | 12.2 |
| true | false | 2 | true | false | 3.9 | true | false | 8.1 |
| true | true | 1 | true | true | 2.1 | true | true | 3.8 |

| $|\mathcal{D}(X)| = 10$ |
|---|
| $R(X)$ | $A(X)$ | $f$ |
| false | false | $\frac{(4\cdot4+7.9\cdot4+15.7\cdot2)}{10} = 7.9$ |
| false | true | $\frac{(3\cdot4+6.4+12.2\cdot2)}{10} = 6.04$ |
| true | false | $\frac{(2\cdot4+3.9\cdot4+8.1\cdot2)}{10} = 3.98$ |
| true | true | $\frac{(1\cdot4+2.1\cdot4+3.8\cdot2)}{10} = 2$ |
Is It Reasonable to Undo Splits?

- Approximate forward message
- For each time step the temporal behaviour is multiplied on the forward message
- Indefinitely bounded error due to temporal behaviour
Taming Reasoning

• Need to undo splits to keep reasoning polynomial w.r.t. domain sizes

• Where can splits be undone efficiently?
  • Undo splits in a forward message

• How to undo splits?
  • Find approximate symmetries
  • Merge based on groundings

• Is it reasonable to undo splits
  • Yes, due to the temporal model behaviour (indefinitely bounded error)
Results

• DBSCAN for Clustering
• ANOVA for checking fitness of clusters

Gehrke et al. (2019e)

<table>
<thead>
<tr>
<th>π</th>
<th>Max</th>
<th>Min</th>
<th>Average</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0.0001537746121</td>
<td>0.00000000001720</td>
<td>0.0000191206488</td>
</tr>
<tr>
<td>2</td>
<td>0.00000000851654</td>
<td>0.00000000000001</td>
<td>0.0000000111949</td>
</tr>
<tr>
<td>4</td>
<td>0.00000000000478</td>
<td>0.00000000000000</td>
<td>0.0000000000068</td>
</tr>
</tbody>
</table>
Outlook

• Continue optimising
  • Parallelisation
  • Caching

• From discrete time interval to time continuous

• Preserving symmetries

• Learning?
  • Structure
  • Potentials (Idea of Baum Welch now possible)
  • Symmetries
  • *Transfer learning*

• Open world?
  • Unknown domains
  • Unknown behaviour
Wrap-up Exact Lifted Dynamic Inference

• Parfactor models for **sparse encoding**
  • Factorisation of full joint distribution
  • Logical variables to model objects

• Algorithms for exact query answering
  • **LDJT** for repeated inference
  • Extensions possible
    • Parameterised, conjunctive queries
    • Maximum expected utility
    • Assignment queries
References

• Ahmadi et al. (2013)

• Apsel and Brafman (2012)

• Apsel and Brafman (2011)

• Dawid (1992)
References

• Dechter (1999)

• De Salvo Braz et al. (2006)

• Murphy (2002)

• Nath and Domingos (2009)
Work @ IFIS

• Braun and Möller (2018b)

• Gehrke et al. (2018)

• Gehrke et al. (2018b)

• Gehrke et al. (2018c)
Work @ IFIS

• Gehrke et al. (2019)

• Gehrke et al. (2019b)

• Gehrke et al. (2019c)
Work @ IFIS

• **Gehrke et al. (2019d)**

• **Gehrke et al. (2019e)**
  Marcel Gehrke, Tanya Braun, and Ralf Möller. Lifted Taming Reasoning in Temporal Probabilistic Relational Models *Technical report*