A Priori Approximation of Symmetries in Dynamic Probabilistic Relational Models

Nils Finke[1][0000−0002−7466−0103] and Marisa Mohr[2][0000−0003−0006−6141]

[1] Oldendorff Carriers GmbH & Co. KG., 23554 Lübeck, Germany
nils.finke@oldendorff.com
[2] inovex GmbH, 22763 Hamburg, Germany
mmohr@inovex.de

Abstract. Lifted inference approaches reduce computational work as inference is performed using representatives for sets of indistinguishable random variables, which allows for tractable inference w.r.t. domain sizes in dynamic probabilistic relational models. Unfortunately, maintaining a lifted representation is challenging in practically relevant application domains, as evidence often breaks symmetries making lifted techniques fall back on their ground counterparts. In existing approaches asymmetric evidence is counteracted by merging similar but distinguishable objects when moving forward in time. While undoing splits a posteriori is reasonable, we propose learning approximate model symmetries a priori to prevent unnecessary splits due to inaccuracy or one-time events. In particular, we propose a multivariate ordinal pattern symbolization approach followed by spectral clustering to determine sets of domain entities behaving approximately the same over time. By using object clusters, we avoid unnecessary splits by keeping entities together that tend to behave the same over time. Understanding symmetrical and asymmetrical entity behavior a priori allows for increasing accuracy in inference by means of *inferred evidence* for unobserved entities to better represent reality. Empirical results show that our approach reduces unnecessary splits, i.e., improves runtimes, while keeping accuracy in inference high.

Keywords: Relational Models · Lifting · Ordinal Pattern · Symmetry

1 Introduction

To cope with uncertainty and relational information of numerous objects over time, in many real-world applications, dynamic (also called temporal) probabilistic relational models (DPRMs[1]) are employed [6]. We consider an example from logistics, specifically shipping, where the transportation of cargo using vessels (objects) changes over time (temporal) depending on supply and demand (relational). To ensure efficient query answering in DPRMs, objects of the same type and behavior, e.g., vessels transporting the same cargo, are treated together in groups, yielding a sparse representation (*lifting*). Lifted inference approaches

[1] pronounced *deeper models*

use relations in a model, allowing for tractable inference w.r.t. domain sizes [14]. Unfortunately, maintaining a sparse representation to ensure efficient query answering is challenging, as evidence breaks symmetries making lifted techniques fall back to their ground counterparts [11]. To retain a lifted representation the field of approximate inference, i.e., approximating symmetries, has emerged in research. Asymmetric evidence is counteracted by treating similar but distinguishable objects as if they were identical to obtain a lifted solution, while only introducing a small and bounded error in exchange for efficient reasoning.

For static models, Singla et al. [18] propose two algorithms for approximate lifted belief propagation and provide error bounds for them. Venugopal and Gogate [20] determine similarities between domain objects based on evidence to form clusters and project the marginal distribution of one object to all objects of a cluster. As approximate symmetries have an effect on the marginal probability distributions of the models, van den Broeck and Niepert [4] propose an approach that produces improved probability estimates on an approximate model. As all these approaches do not account for temporal behavior, Gehrke et al. [8] propose recreating a new lifted representation by merging groundings, which were introduced over time. Common to all these approaches is that in the first place groundings are allowed and then dealt with afterwards, i.e., by exploiting approximate symmetries within message passing in inference tasks.

While dealing with groundings to recover a lifted representation a posteriori has been studied extensively in research, preventing groundings a priori, i.e., before they even occur, has to the best of our knowledge not been studied yet. In this paper, we investigate how understanding model symmetries a priori can prevent groundings from occurring. Motivated by examples from seaborne transportation, we extend learning a lifted model by finding groups of entities, e.g. vessels transporting the same cargo, that behave approximately the same over time. In particular, we determine entity symmetries through an ordinal pattern symbolization approach followed by spectral clustering. Using these entity clusters, we show how to avoid groundings by keeping similar entities together. In general, as we argue, understanding model symmetries a priori to avoid groundings combines well with existing approaches by merging groundings when objects align again. Furthermore, we introduce the concept of *interconnectivity* and its potential for *inferred evidence*, e.g., evidence observed for one entity also applying to other entities. We conclude with an empirical evaluation and show that our approach significantly reduces groundings, i.e., improves runtime performance while increasing accuracy in inference as the reality is better represented by means of inferred evidence.

2 Preliminaries

We recapitulate DPRMs presented by Gehrke et al. [7], which are based on parametric factor graphs introduced by Poole [16]. We illustrate DPRMs in context of an example from shipping. DPRMs combine relational logic with a factor graph, using logical variables (logvars) as parameters for randvars (parameter-

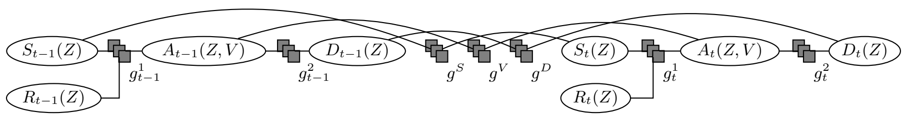


Fig. 1: Two-slice parameterized probabilistic model $G_{\rightarrow}$.

ized randvar, or PRV for short). PRVs compactly represent sets of randvars that are considered indistinguishable without further evidence.

Definition 1 (PRV). *Let* $\mathbf{R}$ *be a set of randvar names,* $\mathbf{L}$ *a set of logvar names,* Φ *a set of factor names, and* $\mathbf{D}$ *a set of entities. All sets are finite. Each logvar* L *has a domain* $\mathcal{D}(L) \subseteq \mathbf{D}$*. A* constraint *is a tuple* $(\mathcal{X}, C_{\mathbf{X}})$ *of a sequence of logvars* $\mathcal{X} = (X_1, \dots, X_n)$ *and a set* $C_{\mathcal{X}} \subseteq \times_{i=1}^{n} \mathcal{D}(X_i)$*. A* PRV $R(L_1, \dots, L_n), n \geq 0$ *is a construct of a randvar name* $R \in \mathbf{R}$ *combined with logvars* $L_1, \dots, L_n \in \mathbf{L}$*. The term* $\mathcal{R}(A)$ *denotes the (range) values of a PRV* A*.*

To represent independent relations, PRVs are linked by parametric factors (parfactors) to compactly encode the full joint distribution of the DPRM.

Definition 2 (Parfactor). *We denote a parfactor* g *by* $\phi(\mathcal{A})_{|C}$ *with* $\mathcal{A} = (A^1, \dots, A^n)$ *a sequence of PRVs,* $\phi : \times_{i=1}^{n} \mathcal{R}(A^i) \mapsto \mathbb{R}^+$ *a function with name* $\phi \in \Phi$*, and* C *a constraint on the logvars of* $\mathcal{A}$*. A PRV* A *or logvar* L *under constraint* C *is given by* $A_{|C}$ *or* $L_{|C}$*, respectively. The term* $gr(P)$ *denotes the set of all instances of* P*. An instance is a grounding of* P*, substituting the logvars in* P *with a set of entities from the constraints in* P*. The term* $lv(P)$ *refers to logvars in* P*. A parameterized model* PRM G *is a set of parfactors* $\{g^i\}_{i=1}^{n}$*, representing the full joint distribution* $P_G = \frac{1}{Z} \prod_{f \in gr(G)} f$*, where* Z *is a normalizing constant.*

Roughly speaking, DPRMs are defined by an initial model and a temporal copy pattern to describe model changes over time.

Definition 3 (DPRM). *A DPRM is a pair of PRMs* $(G_0, G_{\rightarrow})$ *where* G_0 *is a PRM representing the first time step and* $G_{\rightarrow}$ *is a two-slice temporal parameterized model representing* $\boldsymbol{A}_{t-1}$ *and* $\boldsymbol{A}_t$ *where* $\boldsymbol{A}_{\pi}$ *is a set of PRVs from time slice* π*. An inter-slice parfactor* $\phi(\mathcal{A})_{|C}$ *has arguments* $\mathcal{A}$ *under constraint* C *containing PRVs from both* $\boldsymbol{A}_{t-1}$ *and* $\boldsymbol{A}_t$*, encoding transitioning from time step* $t-1$ *to* t*. A DPRM* $(G_0, G_{\rightarrow})$ *represents the full joint distribution* $P_{(G_0, G_{\rightarrow}),T}$ *by unrolling the DPRM for* T *time steps, forming a PRM as defined above.*

Figure 1 shows a DPRM illustrating seaborne transportation that is mainly driven by supply $S_t(Z)$ and demand $D_t(Z)$ of commodities across various locations (zones Z). Vessels V move between these zones, captured by $A_t(Z, V)$, representing trade flows: Vessels are in zones with high supply (to load cargo), in zones with high demand (to discharge cargo), and in between while traveling. For transportation, a fee per ton, called freight rate $R_t(Z)$, is charged. In Figure 1, variable nodes (ellipses) correspond to PRVs, factor nodes (boxes) to parfactors.

Parfactors g^S, g^V, and g^D are so called inter-slice parfactors. The submodel to the left and to the right of these inter-slice parfactors are duplicates of each other, with the left referencing time step $t-1$ and the right referencing time step t. Parfactors reference time-indexed PRVs, namely, a boolean PRV $A_t(Z,V)$ and PRVs $S_t(Z)$, $R_t(Z)$, $D_t(Z)$ with range values $\{high, medium, low\}$. Given a DPRM, one can ask queries for probability distributions or the probability of an event given evidence.

Definition 4 (Queries). *Given a DPRM $(G_0, G_\rightarrow)$, a ground PRV Q_t, and evidence $\boldsymbol{E}_{0:t} = \{\{E_{s,i} = e_{s,i}\}_{i=1}^n\}_{s=0}^t$ (set of events for time steps 0 to t), the term $P(Q_\pi \mid \boldsymbol{E}_{0:t})$, $\pi \in \{0, \dots, T\}$, $t \leq T$, denotes a query w.r.t. $P_{(G_0,G_\rightarrow),T}$.*

In context of the shipping application, an example query for $t = 10$, such as $P(R_{10}(z_1) \mid S_{10}(z_2) = high, S_{10}(z_3) = high)$, contains a set of observations $S_{10}(z_2) = high$ and $S(z_3) = high$ as evidence. Sets of parfactors encode evidence, one parfactor for each subset of evidence that concern one PRV with the same observation.

Definition 5 (Encoding Evidence). *A parfactor $g_e = \phi_e(E(X))_{|C_e}$ encodes evidence for a set of events $\{E(x_i) = o\}_{i=1}^n$ of a PRV $E(X)$. The function ϕ_e maps the value o to 1 and the remaining range values of $E(X)$ to 0. Constraint C_e encodes the observed groundings x_i of $E(X)$, i.e., $C_e = (X, \{x_i\}_{i=1}^n)$.*

Suppose we ask for the probability distribution of supply at a time step $t = 2$ in a certain zone z_1, given that in the previous time step $t = 1$ the supply was high, i.e., $P(S_2(z_1) \mid S_1(z_1) = high$. Then evidence is encoded in parfactors g_1^1 by duplicating the parfactor and using one to encode evidence and one to represent all sets of entities that are still considered indistinguishable. Each parfactor represents a different set of entities bounded by the use of constraints, e.g., limiting the domain for the evidence parfactor to z_1. The parfactor that encodes evidence is adjusted such that all range value combinations in the parfactors distribution ϕ are dropped for $S_1(z_1) \neq high$. During message passing, the splits carry over. Thus, the parfactors g^S and g_1^2 also split into one part for z_1 and another for all other instances. Thus, under evidence a model $G_t = \{g_t^i\}_{i=1}^n$ at time step t, is split w.r.t. its parfactors such that its structure remains

$$G_t = \{g_t^{i,1}, \dots, g_t^{i,k}\}_{i=1}^n \tag{1}$$

with $k \in \mathbb{N}^+$. Every parfactor g_t^i can have up to $k \in \mathbb{N}^+$ splits $g_t^{i,j} = \phi_t^{i,j}(\mathcal{A}^i)_{|C^{i,j}}$, where $1 \leq j \leq k$ and $\mathcal{A}^i$ is a sequence of the same PRVs but with different constraint $C^{i,j}$ and varying functions $\phi_t^{i,j}$ due to evidence.

3 A Priori Approximation of Symmetries

In relational models evidence leads to splits within the models' symmetric structures, i.e., asymmetric evidence slowly grounds a lifted model over time. In the worst case a model is fully grounded, i.e., a model as defined in Eq. (1) contains

$$k = \prod_{L \in lv(\mathcal{A})} |L| \tag{2}$$

splits for every parfactor $g_t^i = \phi_t^i(\mathcal{A})_{|C^i}$ such that each object $l \in L$ is in its own parfactor split. We propose learning approximate model symmetries a priori to relieve the model from unnecessary splits due to inaccuracy or one time events. When knowing approximate model symmetries in advance, one can prevent splits through evidence, e.g., if those are only a one time blip. In general, our approach works well with any other approach undoing splits after they occurred when moving forward in time, i.e., in message passing by merging sets of entities when those align again (temporal approximate merging). As we argue, combining both kind of approaches brings together the best of both worlds:

(a) While with *determining approximate model symmetries*, we can use the full amount of historical training data to prevent groundings a priori,
(b) with *temporal approximate merging*, we can merge non-preventable parfactor splits even after they occurred.

Next, we introduce our approach for the a priori determination of model symmetries in DPRMs in two steps. First, we individually encode PRVs behavior by ordinal pattern symbolization. Secondly, we build groups with similar behavior by spectral clustering. Algorithm 1 in the Appendix outlines the corresponding pseudocode combining both steps.

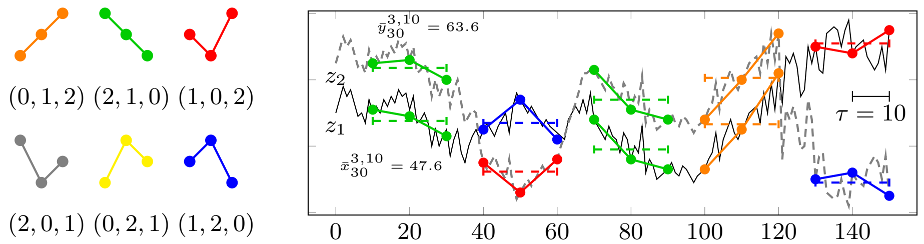
3.1 Encoding Model Behavior by Ordinal Pattern Symbolization

To describe and understand the behavior of entities and to find symmetries between them, historical observations (evidence), encoded in the models PRVs, is used. This means in particular: Every PRV represents multiple entities, e.g., vessels V or zones Z, of the same type. That is, for a PRV $S_t(Z)$, entities z represented by a logvar Z with its domain $\mathcal{D}(Z)$ of size $|\mathcal{D}(Z)|$. For each entity $z_i \in D(Z)$ from this PRV $S_t(Z)$ own observations are made over time, i.e., a time series is generated. Having $|D(Z)|$ entities in Z, we consider $|D(Z)|$ samples of time series in a data matrix

$$\mathcal{X} = ((S_t(z_i))_{i=1}^{|D(Z)|})_{t=1}^{T} \tag{3}$$

with observations $S_t(z_i)$ for every $z_i \in D(Z)$ in time $t \in \{1, \ldots, T\}$. Note that a PRV can be parameterized with more than one logvar, but for the sake of simplicity we introduce our approach using PRVs with only one logvar. Symmetry detection for m-logvar PRVs works similar to one-logvar PRVs, with the difference, that in symmetry detection entity pairs, i.e., m-tuples, are used. As an example, for any 2-logvar PRV $P_t(X, Y)$, an entity pair is a 2-tuple (x_1, y_1) with $x_1 \in D(X)$ and $y_1 \in D(Y)$.

To encode the behavior of an entity (or entity pair), we use ordinal pattern symbolization based on the works by Bandt and Pompe [1]. Ordinal patterns



(a) All possible ordinal patterns of order $d = 3$.

(b) Ordinal pattern determination of order $d = 3$ and delay $\tau = 10$ of a time series concerning two objects z_1 and z_2.

Fig. 2: Ordinal pattern determination. Note, the delay τ refers to the neighbours under consideration within a pattern. The determination of an ordinal pattern is carried out at any time step $t \in [\tau(d-1)+1, T]$. Best viewed in color.

encode the up and downs in a time series by the total order between two or more neighbours and their permutations and therefore give a good abstraction of the overall behavior or the generating process of a time series. Ordinal patterns are formally defined as follows.

Definition 6. *A vector* $(x_1, ..., x_d) \in \mathbb{R}^d$ *has ordinal pattern* $(r_1, ..., r_d) \in \mathbb{N}^d$ *of order* $d \in \mathbb{N}$ *if* $x_{r_1} \geq ... \geq x_{r_d}$ *and* $r_{l-1} > r_l$ *in the case* $x_{r_{l-1}} = x_{r_l}$*.*

Figure 2a shows all possible ordinal patterns of order $d = 3$ of a vector (x_1, x_2, x_3). For a fixed order d, there are $d!$ different ordinal patterns denoted as $o = 1, \dots, d!$. The ordinal approach has notable advantages in applications: (i) The method is conceptually simple, (ii) it is not necessary to have previous knowledge about the data range or type of time series, (iii) the ordinal approach supports robust and fast implementations [10, 15], and, (iv) compared to classical symbolization approaches such as the well-known Symbolic Aggregate ApproXimation (SAX) [5] it allows an easier estimation of a good symbolization scheme [9, 19].

To symbolize a time series $(x_1, x_2, ..., x_T) \in \mathbb{R}^T$, each time step $t \in \{d, ..., T\}$ is assigned its ordinal pattern o of order d, as shown in Figure 2b exemplarily for five time steps in each of two time series. To access overarching trend, delayed behavior is of interest, showing various details of structure of the time series. The time delay $\tau \in \mathbb{N}_{>0}$ is the delay between successive points in the symbol sequences. Finding optimal orders d and delays τ depend on the application and is a challenging problem in research [13]. Practical advice can be found in [17].

A DPRM, as introduced in Section 2, encodes sequential data using its PRVs, e.g., the PRV $S_t(Z)$ encodes supply at time step t in various zones on the globe. Figure 2b shows the time series of two continuous variables, dashed in grey and solid in black, corresponding to observations for the level of supply in two zones z_1 and z_2. Note that for the sake of simplicity we here look at supply over time with its continues range values before its discretization, since DPRMs only support discrete range values. Hereby, the PRVs range values have a total ordering, intuitively, *low* < *medium* < *high*. Note that for example, the PRVs

$A_t(Z, V)$ are boolean and thus have no total order. The further procedure can be executed in this case however on the raw data with the disadvantage that no temporal neighborhood relations are included. The figure shows symmetry in the behavior of the time series data for both variables z_1 and z_2. The time series $(x_t)_{t \in T}$ for z_2 follows the time series $(y_t)_{t \in T}$ for z_1 (or vice versa) with varying offset almost the whole time. The only exceptions are the intervals $40 < t < 60$ and $130 < t < 150$, where both curves develop contrary to each other.

Ordinal patterns are well suited to characterize an overall behavior of time series that is independent of the data range. However, the dependence on the data range is often relevant. For example, in Figure 2b the time series $(x_t)_{t \in T}$ and $(y_t)_{t \in T}$ are similar in terms of their ordinal patterns, but differ when looking at their y intercept. To address this problem, in the next step, entity clustering, we use the arithmetic mean $\overline{x}_t^{d,\tau} = \frac{1}{d} \sum_{k=1}^{d} x_{t-(k-1)\tau}$ of the time series' values corresponding to the ordinal pattern as an additional characteristic or feature of behavior. There are still other features that can be relevant. For simplicity, we only determine ordinal patterns and their means for each PRV of the DPRM, yielding a new data representation

$$\mathcal{S} \in \langle o, m \rangle^{|D(Z)| \times (T - (\tau(d-1))} \tag{4}$$

where $s_{it}\langle o, \cdot \rangle \in \mathcal{S}$ represents the ordinal pattern and $s_{it}\langle \cdot, m \rangle \in \mathcal{S}$ represents the corresponding mean $\overline{x}_t^{d,\tau}$ for an entity $z_i \in D(Z)$ at time step t of a PRV parameterized with the logvar Z. The order d and delay τ are passed in from the outside and might depend on, e.g., the frequency of the data, to capture the overarching behavior of each entity.

3.2 Entity Symmetry Approximation by Spectral Clustering

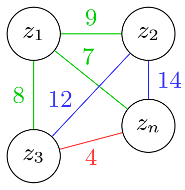
Lifted models are often used when dealing with large domains, i.e., they represent numerous objects resulting in high dimensional data. When the use of clustering is required to uncover symmetries in behavior, the curse of dimensionality [2] complicates this task. Using spectral clustering became a popular setting for problems involving high dimensional data [3].

Similarity Graph Spectral clustering is performed on a similarity graph of entities, where each node represents an entity, e.g., for the PRV $S_t(Z)$ entities $z \in \mathcal{D}(Z)$. The edges between the entities of the lifted model represent their similarity, more precisely how closely related two entities of the model are to each other. The new symbolic representation $\mathcal{S}$, containing tuples of ordinals and means, is used to measure *similar behavior* at each time step $t \in \{\tau(d-1)+1, \ldots, T\}$ individually. As an auxiliary structure, we use a square matrix $\mathcal{W} \in \mathbb{N}^{|D(Z)| \times |D(Z)|}$, where each $w_{ij} \in \mathcal{W}$ describes the similarity between entities z_i and z_j by simple counts of equal behavior over time $t \in T$. The similarity count of equal behavior of two entities z_i and z_j is given by

$$w_{ij} = \sum_{t \leq T} \Big[s_{it}\langle o, \cdot \rangle == s_{jt}\langle o, \cdot \rangle \;\&\&\; |s_{it}\langle \cdot, m \rangle - s_{jt}\langle \cdot, m \rangle| < \delta \Big], \tag{5}$$

	z_1	z_2	z_3	$\dots$	z_n
z_1	0	9	8	$\dots$	7
z_2	9	0	12	$\dots$	14
z_3	8	12	0	$\dots$	4
$\dots$	$\dots$	$\dots$	$\dots$	$\dots$	$\dots$
z_n	7	14	3	$\dots$	0

(a) Auxiliary matrix $\mathcal{W}$ containing counts denoting entity similarity.



(b) Similarity Graph as another representation form of the auxiliary matrix.

	z_1	z_2	z_3	$\dots$	z_n
z_1	24	-1	-1	$\dots$	-1
z_2	-1	35	-1	$\dots$	-1
z_3	-1	-1	24	$\dots$	-1
$\dots$	$\dots$	$\dots$	$\dots$	$\dots$	$\dots$
z_n	-1	-1	-1	$\dots$	25

(c) Graph Laplacian matrix to project the data onto a lower dimensional space.

Fig. 3: (a) Auxiliary matrix and (b) similarity graph, for the construction of the (c) Laplacian matrix used for dimension reduction.

where $[x] = 1$ if x and, 0 otherwise. Simply put, one counts if both time series of z_i and z_j share the same ordinal pattern, and the absolute difference of the two means of the corresponding ordinal pattern is smaller than $\delta > 0$. Once the auxiliary matrix $\mathcal{W}$ is filled, we can derive the similarity graph directly from the auxiliary matrix, since it is just a graphical representation of a matrix as also shown in Figures 3a and 3b. The counts $w_{ij} \in \mathcal{W}$ corresponds to the weights of the edges in the similarity graph, where zero indicates no similarity between two entities, while the larger the count, the more similar two entities are.

Spectral Clustering In the worst case, a similarity graph for a PRV $S_t(Z)$ contains $\binom{|D(Z)|}{2}$ fully-connected nodes, where Z is a logvar representing a set of entities whose entity pairs share similar behavior for least one time step. If the dimension of the similarity graph of the potentially large domains of a lifted model becomes too large, classical clustering methods do not achieve good results because in high-dimensional spaces the smallest and largest distances differ only relatively slightly [2]. While the well-known k-means algorithm assumes that the points assigned to a cluster are spherical around the cluster centre and no good clusters are found due to the relatively equal distances, spectral clustering performs a dimension reduction beforehand and is therefore well suited to uncover similar groups or symmetries in a DPRM. The general approach to spectral clustering is to use a standard clustering method such as k-means on k most relevant eigenvectors of a Laplacian matrix, a special representation of the similarity graph $\mathcal{W}$. For undirected graphs, the graph Laplacian matrix is defined as $L = D - A$, where $D \in \mathbb{N}^{|D(Z)| \times |D(Z)|}$ is a degree matrix containing the degree $d_{ij} \in D$ for each node i of the similarity graph, i.e.,

$$d_{ij} = \begin{cases} \sum_{j=1}^{|D(Z)|} w_{ij} & \text{if } i = j, \\ 0 & \text{if } i \neq j, \end{cases} \tag{6}$$

and A is the adjacency matrix corresponding to the auxiliary matrix $\mathcal{W}$, where $a_{ij} = 1$ if $w_{ij} > 0$, and 0 else. Figure 3c shows the graph Laplacian matrix for the corresponding similarity graph in Figure 3b. Based on the graph Laplacian matrix L, the first k eigenvectors corresponding to the k smallest eigenvalues of L are calculated. Considering the matrix formed by the first k eigenvectors, the rows are used for clustering by a classical clustering algorithm, such as k-means. The eigenvectors contained in the clusters $\mathcal{C}_i^{S_t(Z)}, i = 1, ..., k$ can then be traced back to the entities of an PRV $S_t(Z)$ by indices, and entity symmetry clusters

$$\mathcal{C}(S_t(Z)) = \bigcup_{i=1}^{k} \mathcal{C}_i^{S_t(Z)}. \tag{7}$$

are built with each $C_i^{S_t(Z)}$ containing a subset of entities $z \in D(Z)$. As symmetry clustering is done individually for each PRV, $\mathcal{C}$ denotes the set of all entity symmetry clusters for all PRVs.

Next, we introduce an approach utilizing symmetry clusters to prevent the model from grounding.

4 Preventing Groundings

Commonly, evidence is set in relational models when moving forward in time while answering queries in inference tasks. Here, based on the entity symmetry approximation described in Section 3, we present a procedure for preventing groundings in inference. Algorithm 2 in the Appendix presents the corresponding pseudocode. As entity symmetry approximation also works with other relational formalisms, the procedure can be included in any algorithm before setting evidence that leads to groundings. As the first part of the procedure we address how to avoid model splits, while in the second part we deal with inferred evidence.

4.1 Avoiding Model Splits with Evidence

To avoid model splits, our procedure requires a DPRM, together with a stream of evidence $\mathcal{E}$ and queries $\mathcal{Q}$ as inputs. Furthermore, order d, delay τ, and entity symmetry clusters $\mathcal{C}$ are given as inputs to Algorithm 1. Our procedure contains the following steps.

1. We first loop over all PRVs $P_t(A)$ of the DPRM to initialize for each PRV an auxiliary vector of length $|D(A)|$ that stores the number of times when observed evidence is dismissed due to dissimilar behavior to entities of its symmetry cluster. As follows, when moving forward in time $t = 0, 1, \ldots, T$ evidence $\mathbf{E}_t$ is consumed from an evidence stream $\mathcal{E}$ for that time step t.
2. Since we detect similarity clusters on a PRV level, we consider evidence $\mathbf{E}_t$ on a PRV basis, i.e., we extract evidence from $\mathbf{E}_t$ only concerning one PRV $P(A)$ at the time. We denote evidence at time step t for a PRV $P_t(A)$ from the evidence stream $\mathcal{E}$ as $\mathbf{E}_t^{P(A)}$. Evidence $\mathbf{E}_t^{P_t(A)}$ for a PRV $P_t(A)$ might contain more than one observation, so we additionally cluster $\mathbf{E}_t^{P_t(A)}$ such that we derive evidence clusters $\mathcal{M}$ and each evidence cluster contains evidence for entities that are in the same symmetry cluster $\mathcal{C}_{P_t(A)} \in \mathcal{C}$.

3. By looping over evidence cluster $\mathcal{M}_i \in \mathcal{M}$, i.e., clusters of evidence containing observations for entities in the same symmetry cluster of a PRV $P(A)$, the observation observed the most $max(x)$ is determined.
4. All other observations x in $\mathcal{M}_i$ for an entity $a_i \in D(A)$, which are different to the observation $max(x)$, are dismissed. Still, observations are only dismissed if they are no longer than a certain time period observed.
5. To ensure this, each dismissed observation is counted in the auxiliary vector created in the initialization phase. Once the count reaches the threshold of $^{d\cdot\tau}/_2$, evidence is no longer dismissed and the entity z_i is outsourced from its entity symmetry cluster into its own parfactor split.

The threshold of $^{d\cdot\tau}/_2$ is set based on the assumption that the symmetry clusters are detected based on windows of length $d \cdot \tau$ throughout the whole time series, i.e., we expect that entities at least over that time horizon align again and therefore discrepancies are allowed at least for half of the window. To allow for different threshold, d and τ can still be overwritten externally. To keep entities of similar behavior together in one parfactor groups, evidence observed for other entities is also applied to those in the same symmetry cluster, introducing the concept of *inferred evidence.*

4.2 Interconnectivity yielding Inferred Evidence

We introduce *interconnectivity* as the study of relationships that relate to the behavior of an entity or symmetry cluster in context of other entities or symmetry clusters. There are numerous different types of interconnectivity that, if properly understood, can help increase the overall accuracy of the model by making the model more representative of reality. The different forms of interconnectivity all extend to some type of symmetric behavior, e.g.,

(a) *offset symmetry* as entities with similar behavior within different data ranges, e.g., as illustrated in Fig. 2b,
(b) *inverse symmetry* as entities with contrary behavior, e.g., if some variable is increasing for some entities, that the same is decreasing for others, and
(c) *phase-shifted symmetry* as delayed similar behavior, e.g., some entities follow others with a certain delay phase shifted in time.

For example, in Figure 2b, both curves for entities z_1 and z_2 show similar behavior for almost all times with respect to their ordinal patterns, but not with respect to their y-intercept, which we considered in Section 3 by the additional characteristic or feature, i.e., the mean $\overline{x}_t^{d,\tau} = \frac{1}{d}\sum_{k=1}^{d} x_{t-(k-1)\tau}$. Depending on the parameter δ, this may lead to the entities not falling into a cluster together. Nevertheless, interconnectivity exists between the two entities or clusters, so that evidence from one cluster can be inferred for the other cluster.

In the context of our procedure of preventing groundings, not only evidence is dismissed, but evidence can and partly has to be applied to other entities for which observations are not available. More specifically, in our procedure, evidence observed for one entity or cluster is similarly applied to all other entities within another symmetry cluster based on interconnectivity, in the following denoted as *inferred evidence*. In particular, inferred evidence is necessary to prevent the model from being grounded, as, for example, entities in the same parfactor group need to always have the same observations to keep them in a group. That is, evidence for one entity contrary to other evidence for entities in the same cluster, or even no evidence, would cause new splits. The inference of evidence become possible by knowing in advance which entities share the same evidence, and is therefore a direct outcome of the approach described in Section 3.

5 Empirical Evaluation

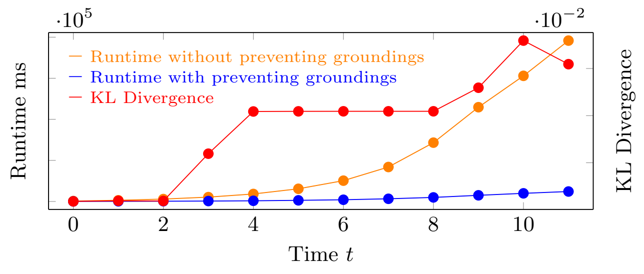
This evaluation of our proposed approach is twofold. We (a) compare runtimes in lifted inferences with and without preventing groundings, using the lifted dynamic junction tree (LDJT) algorithm as query answering algorithm on DPRMs, which was introduced in [7], and (b) compare the models accuracy with and without applying inferred evidence by comparing query answering results. Note, that we do not introduce LDJT here. Details can be found in the original work [7].

To setup a DPRM as shown in Fig. 1, we use historical vessel movements from 2020 based on automatic identification system (AIS) data[2] provided by the Danish Maritime Authority (DMA) for the Baltic Sea. As AIS data provides information about the position of a vessel, including specifications about the vessel itself, the actual supply and demand have to be calculated first. Each AIS signal contains the current geo-position and the total cargo quantity of a vessel. By dividing the Baltic Sea into zones, we derive the total amount of cargo transported between those zones based on the vessel movements and their cargo and thus obtain the data set for this evaluation. We split the data into a training (calendar weeks 1 to 40) and a test data set (remaining weeks). Data and preprocessing can be found on GitHub[3]. All data in the model are fully observable, i.e., the DPRM is derived by counting observations and building a probability distributions for each parfactor by aggregation. Besides, we determine entity symmetry clusters for each PRV of the model based on observation in the training data set. We use the model G_0 with $D(Z) = 367$ and divide all 367 zones into $k = 10$ symmetry groups using the approach described in Section 3 with parameters $d = 3$ and $\tau = 1$. To compare runtime and accuracy in inference, we set evidence based on observations in the test data, i.e., we unroll the model for $t = 11$ further time steps. We perform inference by answering prediction queries for $S_{t+\pi}(Z)$ with $\pi \in \{0, \dots, 11\}$ for each time step t and obtain a marginal distribution for each entity $z \in D(Z)$.

Figure 4 shows the runtime and accuracy to answer queries for each time step. The orange line indicates runtime for answering queries without preventing groundings, whilst the blue line indicates runtime with preventing groundings by means of our approach. The red line indicates the Kullback Leibler divergence (KLD) as measure of accuracy between the two approaches. The KLD compares

[2] https://www.dma.dk/SikkerhedTilSoes/Sejladsinformation/AIS/

[3] https://github.com/FinkeNils/Processed-AIS-Data-Baltic-Sea-2020



t	KLD	t	KLD
0	0.003600	6	0.041832
1	0.003847	7	0.041835
2	0.003786	8	0.041836
3	0.023850	9	0.051835
4	0.041755	10	0.071835
5	0.041815	11	0.061832

Fig. 4: Runtime [ms] and accuracy with and without preventing groundings.

the predicted probability distributions of each entity $z \in D(Z)$ with and without preventing groundings. The average KLD over all entities for each time step, denotes the overall accuracy for a time step. A KLD close to 0 is indicative of similar distributions, thus corresponds to a small error. In particular, Figure 4 shows that the a priori introduction of symmetry clusters to prevent groundings speeds up inference while introducing only a very small error in inference.

6 Conclusion and Future Work

Evidence often grounds dynamic probabilistic relational models over time, negating runtime benefits in lifted inference. To maintain a lifted representation, in this paper we propose an approach to detect model symmetries when learning a lifted model, which can be used in inference under evidence to avoid unnecessary splits, e.g., due to one time events. This novel approach uses ordinal pattern symbolization followed by spectral clustering for a priori approximation of model symmetries, preventing model groundings. Moreover, the concept of clusters or entity interconnectivity as a result of understanding model symmetries enables for inferred evidence, i.e., applying evidence, which is observed for one entity, also to other entities. The empirical evaluation shows that by means of our approach unnecessary groundings are reduced, i.e., improving runtime performance, while also keeping the model accuracy through leveraging inferred evidence and therefore representing the reality more realistically. In future work, we use multivariate ordinal patterns introduced by Mohr et al. [12] to incorporate entity clusters based on their partitions in the DPRM, i.e., on their parfactors. We also investigate various forms of interconnectivity between entity symmetry clusters that can help to further increase the accuracy of the model by representing the reality even better.

Appendix

Algorithm 1: Entity Symmetrie Approximation

```
   Input: Model $G$, Evidence $\mathbf{E}_{0:t}$, Order $d$, Delay $\tau$, Delta $\delta_{\bar{x}}$
1  for every PRV $P_t(A)$ in $G$ do
2      $\mathcal{X}^{|A| \times T} \leftarrow$ get all evidence from $\mathbf{E}_{0:t}$ concerning $P_t(A)$
3      $\mathcal{S}^{|A| \times (T-\tau(d-1))} \leftarrow$ init an empty symbolic representation matrix
4      for every dimension $i = 1, \ldots, |A|$ of $\mathcal{X}$ do
5          $\mathcal{S}_{i\cdot} \leftarrow$ create a time series of tuples with $\langle ordinal, mean \rangle$
6      $\mathcal{W}^{|A| \times |A|} \leftarrow$ auxiliary-matrix initialized with zeros
7      for every time step $t$ of $\mathcal{S}_{\cdot t}$ do
8          $w_{ij} \leftarrow$ do similarity counting                    // see Equation (5)
9      Create a similarity graph based on $\mathcal{W}$
10     Calculate the graph Laplacian matrix $L$ for dimensionality reduction
11     Perform Spectral Clustering based the eigenvectors of $L$
```

Algorithm 2: Preventing Groundings

```
   Input: $(G_0, G_\rightarrow)$ DPRM, $\mathcal{E}, \mathcal{Q}$ streams, $d$ order, $\tau$ delay, $\mathcal{C}$ entity
          clusters
1  for each PRV $P(A) \in G_t$ do // Initialization
2      $H_{P(A)} \leftarrow \vec{0}_n$ vector with $n = |A|$ to count for contrary behavior
3  for $t = 0, 1, ..., T$ do // Query Answering
4      Get evidence $\mathbf{E}_t$ from evidence stream $\mathcal{E}$
5      for each PRV $P_t(A) \in G_t$ do
6          $\mathbf{E}_t^{P_t(A)} \leftarrow$ get evidence concerning the PRV $P(A)$ from $E_t$
7          $\mathcal{M} \leftarrow$ cluster $\mathbf{E}_t^{P_t(A)}$ by entities using clusters $\mathcal{C}_{P_t(A)} \in \mathcal{C}$
8          for each evidence cluster $\mathcal{M}_i \in \mathcal{M}$ do
9              $max(o) \leftarrow$ get most common observation in $\mathcal{M}_i$
10             for observation $P(a_i) = o \in \mathcal{M}_i$ do // dismiss evidence
11                 if $o \neq max(o)$ and $H_{P(a_i)} < d\cdot\tau/2$ then
12                     Dismiss observation $o$ and increase counter
                        $H_{P(a_i)} \leftarrow H_{P(a_i)} + 1$
13                 else
14                     Allow Split and reset counter $H_{P(a_i)} \leftarrow 0$
15             for $a_i \in A$ of $\mathcal{X}_{P(A)}$ without observation do // inferred
                evidence
16                 Apply $max(o)$ as inferred evidence for $a_i$
17     Answer queries $Q_t$ from query stream $\mathcal{Q}$
```

References

1. Bandt, C., Pompe, B.: Permutation Entropy: A Natural Complexity Measure for Time Series. Physical Review Letters **88**(17) (Apr 2002). https://doi.org/10.1103/PhysRevLett.88.174102
2. Bellman, R.: Adaptive control processes: A guided tour. Princeton legacy library, Princeton University Press (2015), https://books.google.de/books?id=iwbWCgAAQBAJ
3. Bertozzi, A.L., Merkurjev, E.: Chapter 12 - graph-based optimization approaches for machine learning, uncertainty quantification and networks. In: Kimmel, R., Tai, X.C. (eds.) Processing, Analyzing and Learning of Images, Shapes, and Forms: Part 2, Handbook of Numerical Analysis, vol. 20, pp. 503–531. Elsevier (2019). https://doi.org/10.1016/bs.hna.2019.04.001, https://www.sciencedirect.com/science/article/pii/S157086591930002X
4. Van den Broeck, G., Niepert, M.: Lifted probabilistic inference for asymmetric graphical models. In: Proceedings of the 29th Conference on Artificial Intelligence (AAAI) (2015)
5. Chiu, B., Keogh, E., Lonardi, S.: Probabilistic Discovery of Time Series Motifs. In: Proceedings of the Ninth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. pp. 493–498. KDD '03, New York, NY, USA (2003)
6. Finke, N., Gehrke, M., Braun, T., Potten, T., Möller, R.: Investigating matureness of probabilistic graphical models for dry-bulk shipping. In: Jaeger, M., Nielsen, T.D. (eds.) Proceedings of the 10th International Conference on Probabilistic Graphical Models. Proceedings of Machine Learning Research, vol. 138, pp. 197–208. PMLR (23–25 Sep 2020)
7. Gehrke, M., Braun, T., Möller, R.: Lifted Dynamic Junction Tree Algorithm. In: Proceedings of the International Conference on Conceptual Structures. pp. 55–69. Springer (2018)
8. Gehrke, M., Möller, R., Braun, T.: Taming Reasoning in Temporal Probabilistic Relational Models. In: Proceedings of the 24th European Conference on Artificial Intelligence (ECAI 2020) (2020). https://doi.org/10.3233/FAIA200395
9. Keller, K., Maksymenko, S., Stolz, I.: Entropy determination based on the ordinal structure of a dynamical system. Discrete and Continuous Dynamical Systems - Series B **20**(10), 3507–3524 (Sep 2015). https://doi.org/10.3934/dcdsb.2015.20.3507
10. Keller, K., Mangold, T., Stolz, I., Werner, J.: Permutation Entropy: New Ideas and Challenges. Entropy **19**(3), 134 (Mar 2017). https://doi.org/10.3390/e19030134
11. Kimmig, A., Mihalkova, L., Getoor, L.: Lifted graphical models: a survey. Machine Learning pp. 1–45 (2014)
12. Mohr, M., Wilhelm, F., Hartwig, M., Möller, R., Keller, K.: New approaches in ordinal pattern representations for multivariate time series. In: Proceedings of the 33rd International Florida Artificial Intelligence Research Society Conference (FLAIRS-33). pp. 124–129. AAAI Press (2020)
13. Myers, A., Khasawneh, F.A.: On the automatic parameter selection for permutation entropy. Chaos: An Interdisciplinary Journal of Nonlinear Science **30**(3), 033130 (Mar 2020). https://doi.org/10.1063/1.5111719, https://aip.scitation.org/doi/10.1063/1.5111719, publisher: American Institute of Physics
14. Niepert, M., Van den Broeck, G.: Tractability through Exchangeability: A New Perspective on Efficient Probabilistic Inference. In: AAAI-14 Proceedings of the 28th AAAI Conference on Artificial Intelligence. pp. 2467–2475. AAAI Press (2014)

15. Piek, A.B., Stolz, I., Keller, K.: Algorithmics, Possibilities and Limits of Ordinal Pattern Based Entropies. Entropy **21**(6), 547 (Jun 2019). https://doi.org/10.3390/e21060547
16. Poole, D.: First-order Probabilistic Inference. In: Proc. of the 18th International Joint Conference on Artificial Intelligence. pp. 985–991. IJCAI Organization (2003)
17. Riedl, M., Müller, A., Wessel, N.: Practical considerations of permutation entropy: A tutorial review. The European Physical Journal Special Topics **222** (Jun 2013). https://doi.org/10.1140/epjst/e2013-01862-7
18. Singla, P., Nath, A., Domingos, P.: Approximate lifting techniques for belief propagation. In: Proceedings of the Twenty-Eighth AAAI Conference on Artificial Intelligence. p. 2497–2504. AAAI'14, AAAI Press (2014)
19. Stolz, I., Keller, K.: A General Symbolic Approach to Kolmogorov-Sinai Entropy. Entropy **19**(12), 675 (Dec 2017). https://doi.org/10.3390/e19120675
20. Venugopal, D., Gogate, V.: Evidence-based clustering for scalable inference in markov logic. In: Calders, T., Esposito, F., Hüllermeier, E., Meo, R. (eds.) Machine Learning and Knowledge Discovery in Databases. pp. 258–273. Springer Berlin Heidelberg, Berlin, Heidelberg (2014)