ASTES

*Advances in Science, Technology and Engineering Systems Journal*
*Vol. 7, No. 2, 73-93 (2022)*
www.astesj.com
Special Issue on Multidisciplinary Sciences and Engineering

**ASTES Journal**
**ISSN: 2415-6698**

# On the Construction of Symmetries and Retaining Lifted Representations in Dynamic Probabilistic Relational Models

Nils Finke*, Ralf Möller

*Institute of Information Systems, Universität zu Lübeck, 23562 Lübeck, Germany*

ARTICLE INFO

*Article history:*
*Received: 11 January, 2022*
*Accepted: 05 March, 2022*
*Online: 18 March, 2022*

*Keywords:*
*Relational Models*
*Lifting*
*Ordinal Pattern*
*Symmetry*

ABSTRACT

*Our world is characterised by uncertainty and complex, relational structures that carry temporal information, yielding large dynamic probabilistic relational models at the centre of many applications. We consider an example from logistics in which the transportation of cargoes using vessels (objects) driven by the amount of supply and the potential to generate revenue (relational) changes over time (temporal or dynamic). If a model includes only a few objects, the model is still considerably small, but once including more objects, i.e., with increasing domain size, the complexity of the model increases. However, with an increase in the domain size, the likelihood of keeping redundant information in the model also increases. In the research field of lifted probabilistic inference, redundant information is referred to as symmetries, which, informally speaking, are exploited in query answering by using one object from a group of symmetrical objects as a representative in order to reduce computational complexity. In existing research, lifted graphical models are assumed to already contain symmetries, which do not need to be constructed in the first place. To the best of our knowledge, we are the first to propose symmetry construction a priori through a symbolisation scheme to approximate temporal symmetries, i.e., objects that tend to behave the same over time. Even if groups of objects show symmetrical behaviour in the long term, temporal deviations in the behaviour of objects that are actually considered symmetrical can lead to splitting a symmetry group, which is called grounding. A split requires to treat objects individually from that point on, which affects the efficiency in answering queries. According to the open-world assumption, we use symmetry groups to prevent groundings whenever objects deviate in behaviour, either due to missing or contrary observations.*

## 1 Introduction

This paper is an extension of two works originally presented in *KI 2021: Advances in Artificial Intelligence* [1] and in *AI 2021: Advances in Artificial Intelligence – 34rd Australasian Joint Conference* [2]. Both papers study the approximation of symmetries using an ordinal pattern symbolisation approach to prevent groundings in dynamic probabilistic relational models (DPRMs)[1].

In order to cope with uncertainty and relational information of numerous objects over time, in many real-world applications, probabilistic temporal (also called dynamic) relational models (DPRMs) need often be employed [3]. Reasoning on large probabilistic models, like in data-driven decision making, often requires evaluating multiple scenarios by answering sets of queries, e.g., regarding the probability of events, probability distributions, or actions leading to a maximum expected utility (MEU). Further, reasoning on large probabilistic models is often performed under time-critical conditions, i.e., where computational tractability is essential [4]. In this respect, DPRMs, together with lifted inference approaches, provide an efficient formalism addressing this problem. DPRMs describe dependencies between objects, their attributes and their relations in a sparse manner. To encode uncertainty, DPRMs encode probability distributions by exploiting in-dependencies between random variables (randvars) using factor graphs. Factor graphs are combined with relational logic, using logical variables (logvars) as parameters for randvars to compactly represent sets of randvars that are considered indistinguishable (without further evidence). This technique is also known as *lifting* [5, 6]. A lifted representation of a probabilistic graphical model allows for a sparse representation to restrain state complexity and enables to decrease runtime complexity in inference.

To illustrate the potential of lifting, let us think of creating a probabilistic model for navigational route planning and congestion

*Corresponding Author: Nils Finke, Institute of Information Systems, Universität zu Lübeck, 23562 Lübeck, Germany, finke@ifis.uni-luebeck.de
[1] pronounced *deeper* models

avoidance in dry-bulk shipping. Dry-bulk shipping is one of the most important forms of transportation as part of the global supply chain [7, 3]. Especially the last year 2020, which was marked by the coronavirus pandemic, shows the importance of good supply chain management. An important sub-challenge in supply chain management is congestion avoidance, which has been studied in research ever-since [8]-[10]. Setting up a probabilistic model to improve planning and to avoid congestion requires identifying features, such as demand for commodities and traffic volume, affecting any routing plans. Commodities are unevenly spread across the globe due to the different mineral resources of countries. In case of excessive demand, regions where the demanded commodities are mined and supplied are excessively visited for shipping, resulting in congestion in those regions. If such a model includes only a few objects, here regions from which commodities are transported, the model might still be considerably small, but once including more regions to capture the whole market, i.e., with an increasing domain size, the complexity of the model increases. However, with an increase in complexity due to an increase in the domain size also the likelihood of keeping redundant information within the model increases. For example, in the application of route planning, multiple regions may exist that are similar in terms of features of the model. Intuitive examples are regions offering the same commodities, i.e., regions with similar mineral resources.

Lifting exactly exploits that existence: Regions which are symmetrical with respect to the features used in the model can be treated by one representative for a group of symmetrical objects to obtain a sparser representation of the model. Further, by exploiting those occurrences, reasoning in lifted representations has no longer a complexity exponential in the number of objects represented by the model, here regions, but is limited to the number of objects with asymmetries only [11, 12]. More specifically, symmetries across objects of a models domain, i.e., objects over randvars of the same type, are exploited by means of performing calculations in inference only once for groups of similarly behaving objects, instead of performing the same calculations over and over again for all objects individually. The principle of lifting applies not only to logistics but also to many other areas like politics, healthcare, or finance – just to name a few.

DPRMs encode a temporal dimension and can be used in any *online scenario*, i.e., new knowledge is on the fly encoded to enable for continuous query answering without relearning the model. In existing research, it is assumed that lifted graphical models already contain symmetries, i.e., simply speaking, a model is setup so that all objects behave according to the same probability distribution. New knowledge is then incorporated in the model with new observations for each object. Observations are encoded within the model as realisations of randvars, resulting in a split off from a symmetrical consideration, called *grounding*. Of course, if the same observation is made for multiple objects, those objects are split off together and continue to be treated as a group. Over time, models dissolve into groups of symmetrically behaving objects, i.e., symmetries are implicitly exploited. Note that in the worst case, the models are split in such way that all objects are treated individually, i.e., no symmetries are available in the model so that lifted inference can no longer be applied and all its advantages disappear.

To the best of our knowledge, existing research has not yet focused on constructing symmetries in advance instead of deriving symmetries implicitly. Constructing symmetries in advance has benefits in application and results from the characteristics of real-world applications:

(i) Certain information about objects of the model may not be available at runtime but only become known downstream. In such cases it is beneficial to infer information according to the open-world assumption from the behaviour of other object which tend to behave similar, i.e., applying an intrinsic default. On the one hand wrong information can be introduced in the model, but on the other it is likely that objects continue to behave the same as per other objects.

(ii) Symmetrical objects can behave the same in the long term, but may deviate for shorter periods of time. Even already small deviations lead to groundings in the model, which, if prevented, introduce a small error in the model, but which also is negligible in the long term.

In both cases a model grounds, which must be prevented in order to keep reasoning in polynomial time. We construct groups of objects with similar behaviour, which we denote as *symmetry clusters*, through a symbolisation scheme to approximate temporal symmetries, i.e., objects that tend to behave the same over time. Using the symmetry clusters, it is possible to selectively prevent groundings, which helps to retain a lifted representation.

This work contributes with a summary on approximating model symmetries in DPRMs based on multivariate ordinal pattern symbolisation and clustering to obtain groups of objects with approximately similar behaviour. Behaviour is derived from the realisations of randvars, which generate either a univariate or multivariate time series depending on the number of interdependent randvars in the model. In the first original conference paper [1], we present multivariate ordinal pattern symbolisation for symmetry approximation ($MOP_4SA$) for the univariate case and introduce symmetry approximation for preventing groundings ($SA_4PG$) as an algorithm to prevent groundings in inference a priori using the learned entity symmetry clusters. In the second original conference paper [2], we extend $MOP_4SA$ to the multivariate case and motivate the determination of related structural changes and periodicities in symmetry structures. Further, this work contributes with an extension of original works in [1] and [2] by

- a **comprehensive review** of $MOP_4SA$ and $SA_4PG$ with additional applications from dry-bulk shipping,
- a **complement** of the existing theoretical and experimental investigations of $MOP_4SA$ and $SA_4PG$ in the variation of its parameters, i.e., we fill in the gaps by investigating different orders and delays for ordinal patterns while also examining different thresholds in the symmetry approximation with respect to the introduced error in inference, and
- a **new approach** named $MOP_4SCD$ to detect changes in symmetry structures based on a models similarity graph, an intermediate step of $MOP_4SA$, to re-learn symmetries on demand.

Together $MOP_4SA$, $SA_4PG$ and multivariate ordinal pattern symbolisation for symmetry change detection ($MOP_4SCD$) combine into a rich toolset to identify model symmetries as part of the model

construction process, use those symmetries to maintain a lifted representation by preventing groundings a priori and detect changes in model symmetries after the model construction process.

This paper is structured as follows. After presenting preliminaries on DPRMs in Section 2, we continue in Section 3 with an review on how to retain lifted solutions through approximation and its relation to time series analysis and common approaches to determine and approximate similarities in that domain. In the following, we recapitulate MOP$_4$SA, an approach that encodes entity behaviour through ordinal pattern symbolisation in Section 4.1 and summarise approximating entity symmetries based on the symbolisation in Section 4.2. In Section 5 we outline SA$_4$PG and elaborate how to prevent groundings in DPRMs a priori with the help of entity symmetry clusters. In Section 6 we evaluate both MOP$_4$SA and SA$_4$PG in a shipping application and provide a detailed discussion on its various parameters and its effect on the accuracy in inference. In Section 7 we introduce a new approach to detect changes in symmetry structures to uncover points in time at which relearning of symmetry clusters becomes beneficial. In Section 9 we conclude with future work.

## 2 Background

In the following, we recapitulate DPRMs [5] in context of an example in logistics, specifically dry-bulk shipping. Dry-bulk shipping is one of the most important forms of transportation as part of the global supply chain [7, 3]. Especially the last year 2020, which was marked by the coronavirus pandemic, showed the importance of good supply chain management. The global supply chain was heavily affected as a result of required lock-downs all over the world, which has led to disruptions and significant delays in delivery. An important sub-challenge in supply chain management is to avoid congestion in regions/zones in which cargo is loaded, i.e., making sure that vessels arrive when those regions are not blocked by too many other vessels being anchored up in same. Congestion avoidance has always been an important topic in research [8]-[10]. As follows, we setup a DPRM to infer idle times related to global supply for commodities. As such, we formally define DPRMs and elaborate on sparse representations and more efficient inference by exploiting symmetries to enable for faster decision making.

### 2.1 A Formal Model on Congestion in Shipping

We setup a simplified DPRM to model congestion resulting in idle times in different regions/zones across the globe. To infer idle times in certain zones, we use freight rates, a fee per ton, which is paid for the transportation of cargoes and differs across zones, as a driver for operators to plan their vessel movements. E.g., an increase in idle time in a zone can be caused by a high freight rate in that same zone, resulting in an higher interest for sending vessels due to the potential to gain higher profits due to high freight rates. Of course, even though freight rates might be higher, not every vessel operator will be able to conclude business in zones which are over-crowed or have higher waiting times increasing costs for lay time. Hence, to describe the interaction between waiting times and freight rates, the idle condition and freight rates in a zone can be represented by one randvar. Freight rates itself are driven by the supply of commodities in zones represented by another randvar. Since idle conditions, freight rates and supply can be similar in multiple zones, we can develop a much smaller model and combine all randvars into one and parameterise these with a logvar to represent the set of all zones respectively. In this example one zone from the set of all zones is referred to as an object or entity, which we use interchangeably moving forward. Such a parameterised random variable is referred to as PRV for short.

**Definition 2.1 (PRV)** *Let* $\mathbf{R}$ *be a set of randvar names,* $\mathbf{L}$ *a set of logvar names,* $\Phi$ *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* $(\mathcal{X}, C_{\mathcal{X}})$ *of a sequence of logvars* $\mathcal{X} = (X_1, \ldots, X_n)$ *and a set* $C_{\mathcal{X}} \subseteq \times_{i=1}^{n} \mathcal{D}(X_i)$*. A* PRV $A(L_1, \ldots, L_n), n \geq 0$ *is a construct of a randvar name* $A \in \mathbf{R}$ *combined with logvars* $L_1, \ldots, L_n \in \mathbf{L}$*. Then, the term* $\mathcal{R}(A)$ *denotes the (range) values of a PRV* $A$*. Further, the term* $lv(P)$ *refers to the logvars and* $rv(P)$ *to the randvars in some element* $P$*. The term* $gr(P_{|C})$ *denotes the set of instances of* $P$ *with all logvars in* $P$ *grounded w.r.t. constraint* $C$*.*

The idea behind PRVs is to enable for combining objects with similar behaviour in a single randvar to come up with a sparse representation, introducing a technique called *lifting*. To model the interaction between idle times, freight rates and supply in zones across the globe, we use randvars *Idle*, *Supply* and *Rate* parameterised with a logvar $Z$ representing zones, building PRVs $Idle(Z)$, $Supply(Z)$ and $Rate(Z)$. The domain of $Z$ is $\{z_0, z_1, \ldots, z_n\}$ and range values for all PRVs are $\{high, medium, low\}$[2]. A constraint $C = (Z, \{z_1, z_2\})$ for $Z$ allows to restrict $Z$ to a subset of its domain, such as here to $z_1$ and $z_2$. Using this constraint, the expression $gr(Idle(Z)_{|C})$ evaluates to $\{Idle(z_1), Idle(z_2)\}$. To represent independent relations, PRVs are linked by a parametric factor (parfactor) to compactly encode the full joint distribution of the DPRM.

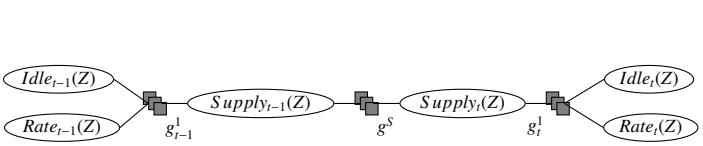
**Definition 2.2 (Parfactor)** *We denote a parfactor* $g$ *by* $\phi(\mathcal{A})_{|C}$ *with* $\mathcal{A} = (A^1, \ldots, A^n)$ *a sequence of PRVs,* $\phi : \times_{i=1}^{n} \mathcal{R}(A^i) \mapsto \mathbb{R}^+$ *a function with name* $\phi \in \Phi$*, and* $C$ *a constraint on the logvars of* $\mathcal{A}$*. A PRV* $A$ *or logvar* $L$ *under constraint* $C$ *is given by* $A_{|C}$ *or* $L_{|C}$*, respectively. An instance is a grounding of* $P$*, substituting the logvars in* $P$ *with a set of entities from the constraints 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.*

All PRVs are dependent on each other and therefore are combined through one parfactor

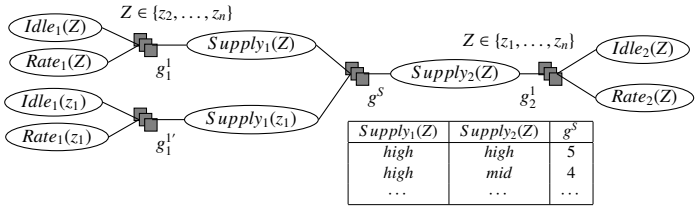
$$g^1 = \phi^1(Idle(Z), Rate(Z), Supply(Z)), \tag{1}$$

which denotes their joint probability distribution. We omit the concrete mappings of potentials to range values of $\phi^1$. To encode temporal behaviour, DPRMs follow the same idea as dynamic Bayesian networks (DBNs) with an initial model and a temporal copy pattern to describe model changes over time. DPRMs model a stationary process, i.e., changes from one time step to the next follow the same distribution.

[2] for sake of simplicity we only consider three range values here




(a) Two-slice parameterized probabilistic model (without encoded evidence).




| $Supply_1(Z)$ | $Supply_2(Z)$ | $g^S$ |
|---|---|---|
| *high* | *high* | 5 |
| *high* | *mid* | 4 |
| ... | ... | ... |

(b) Setting evidence as part of the query $P(S_2(z_1) \mid S_1(z_1) = high$ for $t = 1$.

Figure 1: Graphical representation of a slice of a dynamic probabilistic graphical model illustration how to encode evidence.

**Definition 2.3 (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* $\pi$*. 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 1a shows the final DPRM. Variable nodes (ellipses) correspond to PRVs, factor nodes (boxes) to parfactors. Edges between factor and variable nodes denote relations between PRVs, encoded in parfactors. The parfactor $g^S$ denotes a so-called inter-slice parfactor that separates the past from the present. The submodel on the left and on the right of this inter-slice parfactor are duplicates of each other, with the one on the left referring to time step $t-1$ and the one on the right to time step $t$. Parfactors reference time-indexed PRVs, namely $Idle_t(Z)$, $Rate_t(Z)$ and $Supply_t(Z)$.

## 2.2 Query Answering under Evidence

Given a DPRM, one can ask queries for probability distributions or the probability of an event given evidence.

**Definition 2.4 (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* $P(Q_\pi \mid \boldsymbol{E}_{0:t})$*,* $\pi \in \{0, \ldots, T\}$*,* $t \leq T$*, denotes a query w.r.t.* $P_{(G_0,G_\rightarrow),T}$*.*

In context of the shipping application, an example query for time step $t = 2$, such as $P(Supply_2(z_1) \mid Supply_1(z_1) = high)$, which asks for the probability distribution of supply at time step $t = 2$ in a certain zone $z_1$, given that in the previous time step $t = 1$ the supply was high, contains an observation $Supply_1(z_1) = high$ as evidence. Sets of parfactors encode evidence, one parfactor for each subset of evidence concerning one PRV with the same observation.

**Definition 2.5 (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* $\phi_e$ *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.,* $C_e = (X, \{x_i\}_{i=1}^n)$*.*

Figure 1b depicts how evidence for $t = 1$, i.e., to the left of the interslice parfactor $g^S$, is set within the lifted model.

Evidence is encoded in parfactors $g_1^{1'}$ by duplicating the original parfactor $g_1^1$ and using $g_1^{1'}$ to encode evidence and $g_1^1$ 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, i.e., limiting the domain for the evidence parfactor $g_1^{1'}$ to $\{z_1\}$ and the domain for the original parfactor $g_1^1$ to $\mathcal{D}(Z) \setminus \{z_1\}$. The parfactor that encodes evidence is adjusted such that all range value combinations in the parfactors distribution $\phi$ for $Supply_1(z_1) \neq high$ are dropped. Groundings in one time step are transferred to next time steps, i.e., also apply to further time steps, which we discuss as follows.

## 2.3 The Problem of Model Splits in Lifted Variable Elimination for Inference

As shown in Fig. 1b, evidence leads to groundings in the lifted model. Those model splits are carried over in message passing over time when performing query answering, i.e., in inference. Answering queries, e.g., asking for the probability of an event, results in joining dependent PRVs, or more specifically, joining those parfactors with *overlapping* PRVs. Figure 1b shows a sample of the probability distribution for the interslice parfactor $g^S$ which separates time steps $t-1$ and $t$. Answering the query $P(Supply_2(z_1) \mid Supply_1(z_1) = high$ as per the example in Section 2.1 to obtain the probability distribution over supply in time step $t = 2$, requires to multiply parfactors $g_1^{1'}$ with the interslice parfactor $g^S$ and the parfactor $g_2^1$. However, since in time step $t = 1$ a grounding for $z_1$ exists, the grounding is carried over to the following time step $t = 2$ as the PRV $Supply_1(z_1)$ is connected to the following time step via the interslice parfactor $g^S$. Therefore, evidence, here $Supply_1(z_1) = high$, is also set within $g^S$ dropping all range values for $Supply_1(z_1) \neq high$. This step is necessary to obtain an exact result in inference. For any other queries, this evidence is carried over to all future time steps, accordingly. 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

$$G_t = \{g_t^{i,1}, \ldots, g_t^{i,k}\}_{i=1}^n \tag{2}$$

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. Note that moving forward we use the terms *parfactor split* or *parfactor group* interchangeably.

In our example, the model is only splitted with regards to evidence for the entity $z_1$. All other entities are still considered to be indistinguishable, i.e., lifted variable elimination (LVE) can still exploit symmetries for those instances. To do so, lifted query answering is done by eliminating PRVs, which are not part of the

query, by so called *lifted summing out*. Basically, variable elimination is computed for one instance and exponentiated to the number of isomorphic instances represented. In [5], the author introduce the lifted dynamic junction tree (LDJT) algorithm for query answering on DPRMs, which uses LVE [6, 13] as a subroutine during its calculations. For a full specification of LDJT, we recommend to read on in [5]. In the worst case a model is fully grounded, i.e., a model as defined in Eq. (12) contains

$$k = \prod_{L \in lv(\mathcal{A})} |L| \tag{3}$$

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. The problem of model splits, i.e., groundings, can generally be traced back to two aspects. Groundings arise from

(a) partial evidence or unknown evidence, i.e., certain information about objects of the model may not be available at runtime and either never or only become known downstream, which we denote as *unknown inequality*, or

(b) from different observations for two ore more objects, i.e., objects show different behaviour requiring to consider those individually moving forward, which we denote as *known inequality*.

Once the model is split, those splits are carried forward over time, potentially leading to a fully ground model. By doing so, the model remains exact as new knowledge (in form of observations) is incorporated into the model in all details. Over time, however, distinguishable entities might align and can be considered as one again (in case of known inequality) or entities might have ever since behaved similarly without knowing due to less frequent evidence (in case of unknown inequality). To retain a lifted representation the field of approximate inference, i.e., approximating symmetries, has emerged in research.

# 3 Related Work on Retaining Lifted Solutions Through Approximation and the Connection to Time Series Analysis

Lifted inference approaches suffer under the dynamics of the real world, mostly due to asymmetric or partial evidence. Handling asymmetries is one of the major challenges in lifted inference and crucial for its effectiveness [14, 15]. To address that problem, approximating symmetries has emerged in related research that we discuss in the following.

## 3.1 Approximate Lifted Models

For static (non-temporal) models, in [16] the author propose to approximate model symmetries as part of the lifted network construction process. They perform Lifted Belief Propagation (LBP) [17], which constructs a lifted network, and apply Belief Propagation (BP) to it. The lifted network is constructed by simulating message passing and identifying nodes sending the same message. To approximate the lifted network, message passing is stopped at an earlier iteration to obtain an approximate instance. In [18], the author also approximate symmetries using LBP, but propose piecewise learning [19] of the lifted network. That means that the entire model is divided into smaller parts which are trained independently and then combined afterwards. In this way, evidence only influences the factors in each part, yielding a more liftable model. Besides approaches using LBP, in [20] the author propose evidence-based clustering to determine similar groundings in an Markov Logic Network (MLN). They measure the similarity between groundings and replace all similar groundings with their cluster centre to obtain a domain-reduced approximation. Since the model becomes smaller, also inference in the approximated lifted MLN is also much faster. In [15], the author propose so-called over-symmetric evidence approximation by performing low-rank boolean matrix factorisation (BMF) [21] on MLNs. They show, that for evidence with high boolean rank, a low-rank approximate BMF can be found. Simply put, finding a low-rank BMF corresponds to removing noise and redundant information from the data, yielding a more compact representation, which is more efficient as more symmetries are preserved. As with any existing approach to symmetry approximation, inference is performed on the symmetrised model, ignoring the introduction of potentially spurious marginals in the model. In [12], the author propose a general framework that provides improved probability estimates for an approximate model. Here, a new proposal distribution is computed using the Metropolis-Hasting algorithm [22, 23] on the symmetrised model to improve the distribution of the approximate model. Their approach can be combined with any existing approaches to approximate model symmetries.

Still, most of the existing research is based on static models and requires to get evidence in advance. However, the problem of asymmetric evidence is particularly noticeable in temporal models, and even more so in an *online setting*, since performing the symmetry approximation as part of the lifted network construction process is not feasible [24]. That means that it is necessary to construct a lifted temporal model once and to encode evidence as it comes in over time. Continuous relearning, i.e., reconstructing the lifted temporal model before performing query answering, is too costly. For temporal models in [25], the author propose to create a new lifted representation by merging groundings introduced over time. They perform clustering to group sub-models and perform statistical significance checks to test if groups can be merged.

In comparison to that and to the best of our knowledge, no-one has focused on preventing groundings before they even occur. To this end, we propose to learn entity behaviour in time and cluster entities that behave approximately similar in the long run and use them to accept or reject incoming evidence to prevent the model from grounding. Clustering entity behaviour requires approaches which find symmetries in entity behaviour, i.e., clustering entities which tend to behave the same according to observations made for them. As observations collected over time result in a time series our problem comes down to identify symmetries across time series.

## 3.2 From DPRMs to Time Series

In a DPRM, (real-valued) random variables observed over time are considered as time series. Let $\Omega$ be a set containing all possible states of the dynamical system, also called state space. Events are

taken from a $\sigma$-*algebra* $\mathcal{A}$ on $\Omega$. Then $(\Omega, \mathcal{A})$ is a measurable space. A sequence of random variables, all defined on the same probability space $(\Omega, \mathcal{A}, \mu)$, is called a *stochastic process*. For real-valued random variables, a stochastic process is a function

$$X : \Omega \times \mathbb{N} \to \mathbb{R}, \tag{4}$$

where $X(\omega, t) := X_t(w)$ depending on both, coincidence and time. Note that in the most simple case $\Omega$ matches with $\mathbb{R}$ and $X$ with the identity map. Then the observations are directly related to iterates of some $\omega$, i.e., there is no latency, and the $X$ itself is redundant. Over time, the individual variables $X_t(\omega)$ of this stochastic process are observed, so-called realisations. The sequence of realisations is called *time series*. With the formalism from above and fixing of some $\omega \in \Omega$, a time series is given by

$$(X_1(\omega), X_2(\omega), X_3(\omega), \dots) = (x_t)_{t\in\mathbb{N}}. \tag{5}$$

In the case $x_t \in \mathbb{R}$ the time series is called univariate, while in the case $x_t \in \mathbb{R}^m$ it is called multivariate. Note that for stochastic processes we use the capitalisation $(X(t))_{t\in\mathbb{N}}$, while for observations, i.e., paths or time series, we use the small notation $(x(t))_{t\in\mathbb{N}}$. In summary, evidence in a DPRM encoding stochastic processes $(X(t))_{t\in\mathbb{N}}$ forms a time series $(x_t)_{t\in\mathbb{N}}$ that is the subject of further consideration.

### 3.3 Symmetry Approximation in Time Series

In time series analysis, the notion of similarity, known as symmetry in DPRMs, has often been discussed in the literature [26]-[28]. In general, approaches for finding similarities in a set of time series are either (a) value-based, or (b) symbol-based. *Value-based* approaches compare the observed values of time series. By comparing the value of each point $x_t$, $t = 1, \dots, T$ in a time series $X$ with the values of each other point $y_{t'}$, $t' = 1, \dots, T'$ in another time series $Y$ (warping), they are able to include shifts and frequencies. Popular algorithms such as dynamic time warping (DTW) [29] or matrix profile [30] are discussed, e.g., in [28]. As DPRMs can encode interdependencies between multiple variables, respective multivariate procedures should be used to assess similarities. The first dependent multivariate dynamic time warping (DMDTW) approach is reported by [31], in which the authors treat a multivariate time series with all its $m$ interdependencies as a whole. The flexibility of warping in value-based approaches leads to a high computational effort and is therefore unusable for large amounts of data. Although there are several extensions to improve runtime [32] by limiting the warping path or reducing the number of data points, e.g., FastDTW [32] or PrunedDTW [33], the use of dimensionality reduction is inevitable in context of DPRMs. For dimensionality reduction, *symbol-based* approaches encode the time series observations as sequences of symbolic abstractions that match with the shape or structure of the time series. Since DPRMs encode discrete values, depending on the degree for discretisation, symbol-based approaches are preferred as they allow for discretisation directly. As far as research is concerned, there are two general ways of symbolisation. On the one hand, *classical symbolisation* partitions the data range according to specified mapping rules in order to encode a numerical time series into a sequence of discrete symbols. A corresponding and well-know algorithm is Symbolic Aggregate ApproXimation (SAX) introduced by [34]. On the other hand, as introduced by Bandt and Pompe [35] *ordinal pattern symbolisation* encodes the total order between two or more neighbours ($x < y$ or $x > y$) into so-called ordinal symbols ((0, 1) or (1, 0)). In [36], the author extend univariate ordinal patterns to the multivariate case, taking into account not only the dependencies of neighbouring values over time, but also the dependencies between spatial variables in a time series.

Specifically here, an ordinal approach has notable advantages in application: (i) The method is conceptionally simple, (ii) the ordinal approach supports robust and fast implementations [37, 38], and (iii) compared to classical symbolisation approaches such as SAX, it allows an easier estimation of a good symbolisation scheme [39, 40]. In the following, we introduce ordinal pattern symbolisation to classify similar entity behaviour.

## 4 Multivariate Ordinal Pattern for Symmetry Approximation (MOP$_4$SA)

In this section we recapitulate MOP$_4$SA, an approach for the approximation of symmetries over entities in the lifted model. MOP$_4$SA consists of two main steps, which is (a) encoding entity model behaviour through an *ordinal pattern symbolisation* approach, followed by (b) clustering entities with a similar symbolisation scheme to determine groups of entities with *approximately similar behaviour*. We have introduced MOP$_4$SA in [1] for the univariate case and extended same in [2] to the multivariate case.

### 4.1 Encoding Entity Behaviour through Ordinal Pattern Symbolisation

As mentioned in Section 3.3, approximating entity behaviour corresponds to finding symmetries in time series.

#### 4.1.1 Gathering Evidence

To find symmetries in (multivariate) time series, we use evidence which encode model entity behaviour w.r.t. a certain context, i.e., w.r.t. a parfactor. In particular, this means: Every time-index PRV $P_t(X)$ represents multiple entities $x_0, \dots, x_n$ of the same type at a specific point in time $t$. That is, for a PRV $Supply_t(Z)$, zones $z_0, \dots, z_n$ are represented by a logvar $Z$ with domain $\mathcal{D}(Z)$ and size $|\mathcal{D}(Z)|$. Note that a PRV can be parameterised with more than one logvar, but for the sake of simplicity we introduce our approach using PRVs with only one logvar throughout this paper. Symmetry detection for $m$-logvar PRVs works similarly 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 $x_1 \in \mathcal{D}(X)$ and $y_1 \in \mathcal{D}(Y)$.

A DPRM, as introduced in Section 2.1, encodes temporal data by unrolling a DPRM while observing evidence for the models PRVs, e.g., the PRV $Supply_t(Z)$ encodes supply at time $t$ in various zones $Z$ on the globe. In addition, a DPRM exploits (conditional) independencies between randvars by encoding interdependencies in parfactors. As such, parfactors describe interdependent data through its linked PRVs, e.g., the correlation between supply $Supply_t(Z)$, idle times $Idle_t(Z)$ and freight rates $Rate_t(Z)$ within a common zone

$Z$ encoded by the parfactor $g_t^1$. For each entity $z_i \in \mathcal{D}(Z)$ from the PRVs $P = \{Supply_t(Z), Idle_t(Z)$ and $Rate_t(Z)\}$ observations are made over time, i.e., a time series $((x_t^i)_{i=1}^m)_{t=1}^T$ with $x_t^i \in \mathcal{R}(P^i)$ is generated. In this work, the time series is to be assumed multivariate, containing interdependent variables, i.e., $m > 1$. Note that in [1] we consider the case $m = 1$. Having $|\mathcal{D}(Z)|$ entities in $Z$, we consider $|\mathcal{D}(Z)|$ samples of multivariate time series

$$\mathcal{X} = (((x_t^i)_{i=1}^m)_{t=1}^T)_{j=1}^{|\mathcal{D}(Z)|} \in \mathbb{R}^{m \times T \times |\mathcal{D}(Z)|}, \tag{6}$$

e.g., for $m = 3$ with observations $(x_t^1, x_t^2, x_t^3) = (Supply_t(z_j), Idle_t(z_j), Rate_t(z_j))$ for every $z_j \in \mathcal{D}(Z)$ in time $t \in \{1, \ldots, T\}$. As such, a multivariate time series is defined for several PRVs linked in a parfactor, while a univariate time series is defined for a single PRV. Identification of symmetrical entity behaviour is done on a sets of (multivariate) time series, i.e., across different (multivariate) time series that are observed for every entity individually.

### 4.1.2 Multivariate Ordinal Pattern (MOP) Symbolisation

To encode the behaviour of a time series, we use ordinal pattern symbolisation based on works from Bandt and Pompe [35]. For this, let $X_t \in \mathbb{R}^{m \times T}$ be a (multivariate) time series and $X_t \in \mathbb{R}^{m \times T \times n}$ be the reference database of $n \in \mathbb{N}$ (multivariate) time series. In case of $m = 1$, the time series is univariate. For a better understanding, we start with univariate ordinal patterns that encode the up and downs in a time series by the total order between two or more neighbours. The encoding gives a good abstraction, an approximation, of the overall behaviour or generating process. Univariate ordinal patterns are formally defined as follows.

**Definition 4.1 (Univariate Ordinal Pattern)** *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 2 shows all possible ordinal patterns of order $d = 3$ of a vector $(x_1, x_2, x_3) \in \mathbb{R}^3$.

(2 1 9) (0 1 2) (1 2 0) (0 2 1) (2 0 1) (1 0 2)

Figure 2: All $d!$ possible univariate ordinal patterns of order $d = 3$.

For a *multivariate* time series $((x_t^i)_{i=1}^m)_{t=1}^T$, each variable $x^i$ for $i \in 1, ..., m$ depends not only on its past values but also has some dependency on other variables. To establish a total order between two time points $(x_t^i)_{i=1}^m$ and $(x_{t+1}^i)_{i=1}^m$ with $m$ variables is only possible if $x_t^i > x_{t+1}^i$ or $x_t^i < x_{t+1}^i$ for all $i \in 1, ..., m$. Therefore, there is no trivial generalisation to the multivariate case. An intuitive idea, based on some theoretical discussion in [41, 42] and introduced in [36], is to store univariate ordinal patterns of all variables at a time point $t$ together into a symbol.

**Definition 4.2 (Multivariate Ordinal Pattern)** A matrix $(x_1, ..., x_d) \in \mathbb{R}^{m \times d}$ has *multivariate ordinal pattern* (MOP) of order $d \in \mathbb{N}$

$$\begin{pmatrix} r_{11} & \cdots & r_{1d} \\ \vdots & \ddots & \vdots \\ r_{m1} & \cdots & r_{md} \end{pmatrix} \in \mathbb{N}^{m \times d} \tag{7}$$

if $x_{r_{i1}} \geq ... \geq x_{r_{id}}$ for all $i = 1, ..., m$ and $r_{il-1} > r_{il}$ in the case $x_{r_{il-1}} = x_{r_{il}}$.

For $m = 1$ the multivariate case matches with the univariate case in Definition 4.1. Figure 3 shows all $(d!)^m$ possible multivariate ordinal patterns (MOPs) of order $d = 3$ and number of variables $m = 2$.

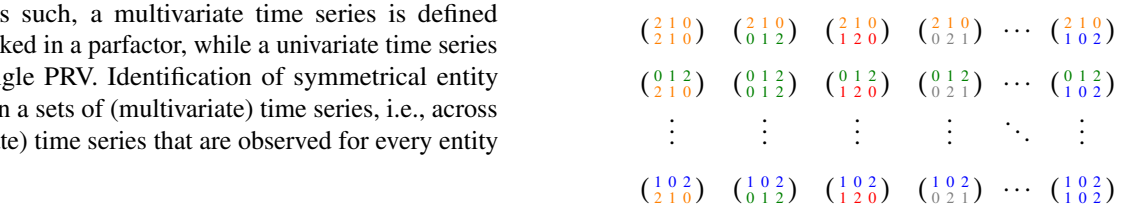


Figure 3: All $(d!)^m$ possible multivariate ordinal patterns of order $d = 3$ with $m = 2$ variables.

The number of possible MOPs $d!$ increases exponentially with the number of variables $m$, i.e., $(d!)^m$. Therefore, if $d$ and $m$ are too large, depending on the application, each pattern occurs only rarely or some not at all, resulting in a uniform distribution of ordinal patterns [36]. This has the consequence that subsequent learning procedures can fail. Nevertheless, for a small order $d$ and sufficiently large $T$ the use of MOPs can lead to higher accuracy in learning tasks, e.g., classification [36] because they incorporate interdependence of the spatial variables in the multivariate time series.

To symbolise a multivariate time series $X_t \in \mathbb{R}^{m \times T}$ each pattern is identified with exactly one of the *ordinal pattern symbols* $o = 1, 2, ..., d!$, before each point $t \in \{d, ..., T\}$ is assigned its ordinal pattern symbol of order $d \ll T$. The order $d$ is chosen much smaller than the length $T$ of the time series to look at small windows in a time series and their distributions of up and down movements. To assess long-term trends, delayed behaviour is of interest, showing various details of the structure of the time series. The time delay $\tau \in \mathbb{N}_{>0}$ is the delay between successive points in the symbol sequences.

### 4.1.3 MOP Symbolisation with Data Range Dependence

We assume that for each time step $t = \tau(d-1)+1, \ldots, T$ of a multivariate time series $((x_t^i)_{i=1}^m)_{t=1}^T) \in \mathcal{X}$, MOP is determined as described in Section 4.1.2. Ordinal patterns are well suited to characterise an overall behaviour of time series, in particular their application independent of the data range. In some applications, however, the dependence on the data range can be also relevant, i.e., time series can be similar in terms of their ordinals patterns, but differ considering their $y$-intercept. In other words, transforming a sequence

$$x = (x_t^i)_{a \leq t \leq b} \tag{8}$$

as $y = x + c$, where $c \in \mathbb{R}$ is a constant, should change $y$'s similarity to other sequences, although the shape is the same. To address the

dependence on the data range, we use the arithmetic mean

$$\overline{x}_t^{d,\tau} = \frac{1}{m}\sum_{i=1}^{m}\frac{1}{d}\sum_{k=1}^{d} x_{i,t-(k-1)\tau} \tag{9}$$

of the multivariate time series' values corresponding to the ordinal pattern, where $x_{i,t-(k-1)\tau}$ is min-max normalised, as an additional characteristic or feature of behaviour. If one of the variables changes its behaviour significantly along the intercept, the arithmetic mean uncovers this. There are still other features that can be relevant. For simplicity, we only determine ordinal patterns and their means for each parfactor $g^1$ with, e.g., PRVs ($Supply_t(Z)$, $Idle_t(Z)$, $Rate_t(Z)$), yielding a new data representation

$$\mathcal{X}' = \langle o, \overline{x}\rangle^{(T-(\tau(d-1))\times|\mathcal{D}(Z)|} \tag{10}$$

where $\langle o, \cdot\rangle_{tj} \in \mathcal{X}'$ represents the MOP and $\langle \cdot, \overline{x}\rangle_{tj} \in \mathcal{X}'$ represents the corresponding mean $\overline{x}_t^{d,\tau}$ for entity $z_j$ at time step $t$. The order $d$ and delay $\tau$ are passed in from the outside and might depend on, e.g., the frequency of the data, to capture the long-term behaviour.

## 4.2 Clustering Entities with Similar Symbolisation Scheme

After encoding the behaviour of the entities through ordinal pattern symbolisation, we identify similar entities using clustering. For this purpose, based on the derived symbolisation representation in Eq. (10), we create a similarity graph indicating the similarities based on a distance measure between entity pairs.

### 4.2.1 Creating a Similarity Graph

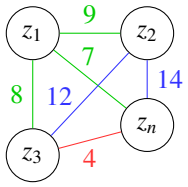
Entity similarity is measured per parfactor, i.e., per multivariate time series, separately. Therefore, multiple similarity graphs, more specifically one per parfactor, are constructed. A similarity graph for a parfactor $g_t^1$ connecting the PRVs $Supply_t(Z)$, $Idle_t(Z)$ and $Rate_t(Z)$ contains one node for each entity $z \in \mathcal{D}(Z)$ observed in form of multivariate time series. The edges of the similarity graph represent the similarity between two nodes, or more precisely, how closely related two entities of the model are. To measure similarity, we use the symbolic representation $\mathcal{X}'$, which contains tuples of multivariate ordinal numbers and mean values that describe the behaviour of an entity. The similarity of two entities $z_i$ and $z_j$ is given by counts $w_{ij}$ of equal behaviours, i.e.,

$$w_{ij} = \sum_{t\leq T}\Big[\langle o, \cdot\rangle_{it} = \langle o, \cdot\rangle_{jt} \ \wedge\ |\langle \cdot, \overline{x}\rangle_{it} - \langle \cdot, \overline{x}\rangle_{jt}| < \delta\Big], \tag{11}$$

where $[x] = 1$ if $x$ and, 0 otherwise. As an auxiliary structure, we use a square matrix $\mathcal{W} \in \mathbb{N}^{|\mathcal{D}(Z)|\times|\mathcal{D}(Z)|}$, where each $w_{ij} \in \mathcal{W}$ describes the similarity between entities $z_i$ and $z_j$ by simple counts of equal behaviour over time $t \in T$. Simply put, one counts the time steps $t$ at which both multivariate time series of $z_i$ and $z_j$ have the same MOP and the absolute difference of the mean values of the corresponding MOPs is smaller than $\delta > 0$. Finally, as shown in Figure 4b the counts $w_{ij}$ correspond to the weights of edges in the similarity graph $\mathcal{W}$, where zero indicates no similarity between two entities, while the larger the count, the more similar two entities are.

| | $z_1$ | $z_2$ | $z_3$ | ... | $z_n$ |
|---|---|---|---|---|---|
| $z_1$ | 0 | 9 | 8 | ... | 7 |
| $z_2$ | 9 | 0 | 12 | ... | 14 |
| $z_3$ | 8 | 12 | 0 | ... | 4 |
| ... | ... | ... | ... | ... | ... |
| $z_n$ | 7 | 14 | 3 | ... | 0 |

(a) Auxiliary matrix with entity similarity counts.




(b) Auxiliary matrix in form of the Similarity Graph.

Figure 4: (a) Auxiliary matrix and (b) similarity graph.

Approximating symmetries based on the similarity graph leaves us with a classical clustering problem. That means, clustering entities into groups of entities showing *enough* similarities or leaving others independent if those are too different.

### 4.2.2 Derive Entity Similarity Clusters

Theoretically, any clustering algorithm can be applied on top of the similarity graph. Each weight in the similarity matrix, or each edge weight between an an entity pair, denotes the similarity between two entities, i.e., the higher the count, the more similar the two entities are to each other. Since this contribution focuses on identifying symmetries in temporal environments, we leave the introduction of a specific clustering algorithm out here, and compare two different ones, specifically Spectral Clustering and Density-Based Spatial Clustering of Applications with Noise (DBSCAN), for the use in MOP$_4$SA as part of the evaluation in Section 6. After any clustering algorithm is run, we are left with $n$ clusters of entities for each parfactor. Formally, a symmetry cluster is defined as follows.

**Definition 4.3 (Symmetry Cluster)** *For a parfactor $g_i \in G_t$ with $G_t$ being the PRM at timestep $t$ and $g_i = \phi(\mathcal{A})_{|C}$ containing a sequence of PRVs $\mathcal{A} = (A^1, \ldots, A^n)$, a symmetry cluster $S^i$ contains entities $l \in \mathcal{D}(L)$ concerning the domain $\mathcal{D}(L)$ of one of the log-vars $L \in \boldsymbol{L}$ with $\boldsymbol{L} = \bigcup_{i=1}^{n} lv(A^i)$. Let the term $en(S)$ refer to the set of entities in any symmetry cluster $S$. Each parfactor $g_i \in G_t$ can contain up to $m$ symmetry clusters $\mathcal{S}_{|g^i} = \{S^i\}_{i=0}^{m}$, such that $en(S^i) \cap en(S^j) = \emptyset$ for $i \neq j$ and $i, j \in \{1, \ldots, m\}$. $|\top$ may be omitted in $\mathcal{S}_{|\top}$.*

In the following section we propose how to utilise symmetry clusters to prevent any lifted model from unnecessary groundings.

# 5 Symmetry Approximation for Preventing Groundings (SA$_4$PG)

As described in Section 2.3, evidence leads to groundings in any lifted model. Further, those groundings are carried over in message passing when moving forward in time leading to a fully ground model in the worst case. As follows, we propose SA$_4$PG, which uses symmetry clusters as an outcome of MOP$_4$SA to counteract any unnecessary groundings, which occur due to evidence. Since symmetry clusters denote a sets of entities, for which entities in each group tend to behave the same, also observations for each entity individually within a cluster are expected to be similar. Regardless of our approach to prevent groundings, in DPRMs entities are

considered indistinguishable after the initial model setup. Under evidence entities split off from that indistinguishable consideration and are afterwards treated individually to allow for exact inference. Nevertheless, in case one observation was made for multiple entities, those all together split off and are considered individually, but still within the group of entities for which the that observation was made. Such groundings are encoded within the DPRM in parfactor groups as shown in Eq. (12). Symmetry clusters also denote parfactor groups, with the difference that those are determined as part of the model construction process in advance. Therefore, in the model construction process, i.e., before running inference, a model will be splitted according to the clusters into parfactor groups with each group containing only entities from the respective cluster. The only difference in creating parfactor groups without evidence is that no range values are set to zero, but get a different distribution representing the group the best. SA$_4$PG is based on the assumption, that symmetry clusters *stay valid* for a certain period of time after learning them, i.e., that entities within those clusters continue to behave similarity. More specifically and w.r.t. the two types of inequality (see Section 2.3), this means, that

(a) in case of *unknown inequality*, we assume that any entity without an observation most likely continues to behave similar to the other entities within the same cluster for which observations are present, and

(b) in case of *known inequality*, we assume that certain observations dominate one cluster and therefore will be applied for all entities within the cluster.

To make one example, lets assume a symmetry cluster contains entities $z_1$, $z_2$ and $z_3$. Groundings occur whenever observations differ across entities in a symmetry cluster, e.g., grounding occurs, if (a) the observation $(Supply_1(z_1) = high, Idle_1(z_1) = high)$ of entity $z_1$ differs from observations $(Supply_1(z_i) = low, Idle_1(z_i) = mid)$ of entities $z_i$ for $i = 2, 3$, or (b) observations are only made for a subset of the entities, i.e., for entities $z_2$ and $z_3$, but not for entity $z_1$. In both cases, the entities $z_2$ and $z_3$ would be split off from their initial symmetry group, and are henceforth treated individually in a non lifted fashion. In SA$_4$PG we prevent such model splits until a certain extend. Algorithm 1 shows an outline of the overall preventing groundings approach. Preventing groundings works by consuming evidence and queries from a stream and dismissing or inferring evidence within symmetry clusters until an entity has reached an violation threshold $H$. The threshold $H$ refers to the number of times evidence was inferred or dismissed. To do not force entities to stick to their initial symmetry clusters, we relieve entities from their clusters once the threshold $H$ is received. To keep track on the number of violations, i.e., how often evidence was inferred or dismissed, we introduce a violation map as a helper data structure to store that number.

**Definition 5.1 (Violation Map)** *For a parfactor $g_i \in G_t$ with $G_t$ being the PRM at timestep t and $g_i = \phi(\mathcal{A})_{|C}$ containing a sequence of PRVs $\mathcal{A} = (A^1, \ldots, A^n)$, a violation map $v_{|g_i} : \bigcup_{i=1}^{n} gr(A^i) \to 0$ is initialised with zero values for all entities in all PRVs $\mathcal{A}$ in $g_i$. In case a PRV $A^i$ is is parameterised with more than one logvar, i.e., $m = |lv(A^i)|$ with $m > 1$, $v$ contains m-tuples as entity pairs. A model contains up to n parfactors in $G_t$. The set of violation maps is denoted by $V = \{v_{|g_i}\}_{i=0}^{n}$. Let viol$(P)$ refer to the violation count of some entity m-tuple in $V$.*

SA$_4$PG continues by taking all evidence $\mathbf{E}_t$ concerning a timestep $t = 0, 1, \ldots, T$ from the evidence stream $\mathcal{E}$. To set evidence and to prevent groundings, for each observation $E_{s,i} \in \mathbf{E}_t$ with $\mathbf{E}_t = \{E_{s,i} = e_{s,i}\}_{i=1}^{n}$ so called *parfactor partitions* are identified. A parfactor partitions is a set of parfactor groups $g_t^{i,k}$ that are all affected by evidence $E_{s,i}(x_j)$ with $x_j \in \mathcal{D}(lv(E_{s,i}))$. A parfactor group is *affected*, if

(a) the parfactor $g_t^i$ itself links the PRV $E_{s,i}$ for which an observation was made,

(b) and if the parfactor group $g_t^i$ currently represents the distribution for the specific entity $x_j$ for which the observation was made.

To make one example, observing $Supply_1(z_1) = high$, the evidence partition contains parfactor groups of the parfactors $g_t^1$ and $g^S$ since the PRV is linked to both parfactors. Further, the parfactor partition is limited to only those $i$ parfactor groups $g_t^{i,1}$ and $g_t^{i,S}$, which currently represent the distribution for the entity $z_1$. A parfactor partition containing all those parfactor groups is defined as follows.

**Definition 5.2 (Parfactor Partition)** *Every parfactor $g_t^i \in G_t$ can have up to $k \in \mathbb{N}^+$ splits such that*

$$G_t = \{g_t^{i,1}, \ldots, g_t^{i,k}\}_{i=1}^{n}. \tag{12}$$

*Each parfactor $g_t^i$ contains a sequence of PRVs $\mathcal{A}_t = (A_t^1, \ldots, A_t^n)$, which are afflicted with evidence $A_t^n(x_i) = a_{t,i}$ for any entity $x_i \in \mathcal{D}(X)$ with $X \in lv(\mathcal{A}_t)$ at timestep t leading to those splits. A parfactor partition $P_t$ denotes a set of parfactors, which are affected by new evidence $E_t(x_i) = e_t$ with*

$$P_t = \{g_t^{i,1}, \ldots, g_t^{i,l}\}_{i=1}^{n} \tag{13}$$

*and $l \leq k$ such that any parfactor group $g_t^{i,l} \in P_t$ contain the randvar $E_t$, i.e., $E_t \in rv(g_t^{i,l})$ and $g_t^{i,l}$ is limited by constraints to at least the entity $x_i$ for which the observation was made, i.e., $g_{t\,|C_e}^{i,l}$ with $C_e = (X, \{x_i\}_{i=1}^{n})$ and $x_i \in \{x_i\}_{i=1}^{n}$.*

Considering all evidence $\mathbf{E}_t$ for a time step $t$, different observations $E_{t,i} \in \mathbf{E}_t$ can result in the same parfactor partition (before those observations are encoded within the model). This holds true for all observation, which are made for the same PRV with entities being in the same parfactor group, e.g., two observation $Supply_1(z_i) = high$ and $Supply_1(z_j) = mid$ for which $\{z_i, z_j\} \in gr(g_1^{1,l})$ and $\{z_i, z_j\} \in gr(g^{S,l})$. All observations that entail the same parfactor partition are treated in SA$_4$PG as one and those observations are informally denoted as an *evidence cluster*.

Therefore, in SA$_4$PG evidence $\mathbf{E}_t$ is rearranged in a sense such that $\mathbf{E}_t$ contains multiple collections of observations, i.e.,

$$\mathbf{E}_t = \{\{E_{t,l} = e_{t,l}\}_{l=0}^{m}, \ldots, \{E_{t,l} = e_{t,l}\}_{l=0}^{m}\}, \tag{14}$$

with each element $E_{t,l}$ originally being directly in $\mathbf{E}_t$ and each subset $\{E_{t,l} = e_{t,l}\}_{l=0}^{m}$ concerning the same parfactor partition $P_t$. SA$_4$PG proceeds by processing each evidence cluster separately. Evidence of each evidence cluster is processed in a sense such that known inequalities and any uncertainty about inequality is counteracted. This is being done by identifying the *dominating observation* within

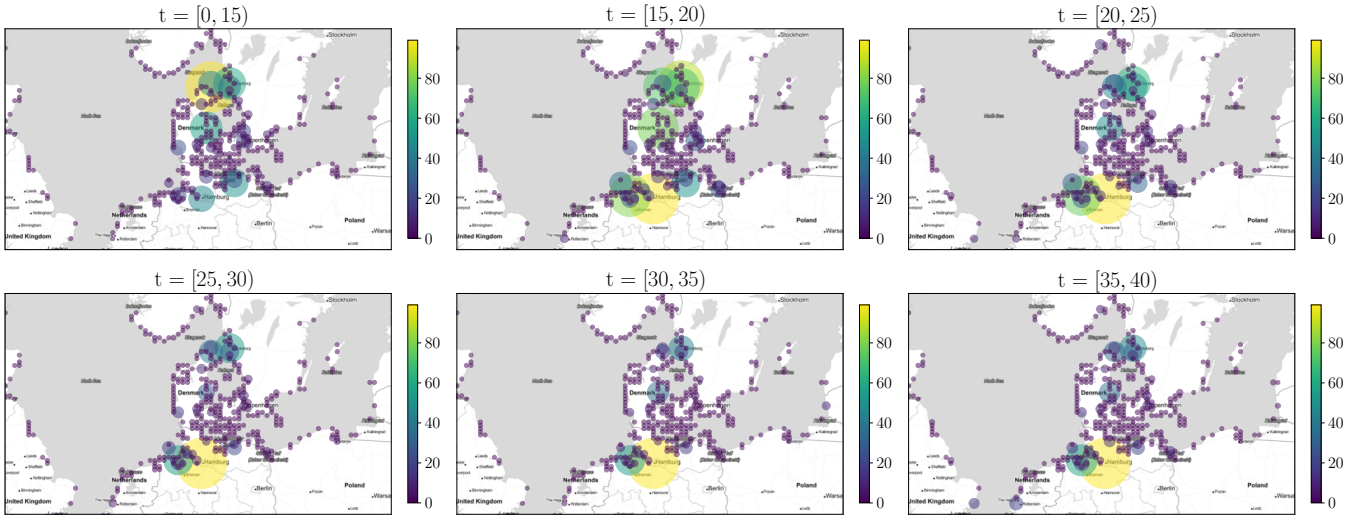



Figure 5: Pointmap showing the normalised average supply over time intervals $[t-5, t)$ in the Baltic Sea region. Best viewed in colour.

each evidence cluster $\{E_{t,l} = e_{t,l}\}_{l=0}^{m}$ and apply that observation to all entities within the respective parfactor partition $P_t$. The dominating observation $max(e_{t,l})$ is the observation that can be observed the most within the evidence cluster. Further, in case any other entities in the corresponding parfactor partition are unobserved, we also apply the dominating observation for those. For each entity for which evidence was inferred or dismissed, the violation counter is increased. In case the violation threshold $H$ for an entity is already reached, evidence is no longer inferred or dismissed, but the entity is relieved from its symmetry cluster, i.e., split off from the parfactor group, and from then on considered individually.

In the following, we evaluate MOP$_4$SA and SA$_4$PG as part of a case study from the example shipping domain.

# 6 MOP$_4$SA and SA$_4$PG in Application

Since MOP$_4$SA consists of multiple steps, namely (a) encoding entity behaviour, (b) similarity counting, and (c) clustering, we evaluate each step separately before analysing the overall fitness in conjunction with SA$_4$PG, as introduced in Section 5.

## 6.1 The AIS Dataset

To setup a DPRM as shown in Figure 1a, we use historical vessel movements from 2020 based on automatic identification system (AIS) data[2] provided by the Danish Maritime Authority for the Baltic Sea. AIS data improves the safety and guidance of vessel traffic by exchanging navigational and other vessel data. It was introduced as a mandatory standard by the International Maritime Organisation (IMO) in 2000. Meanwhile, AIS data is used in many different applications in research, such as trade flow estimation, emission accounting, and vessel performance monitoring [43]. Preprocessing for retrieving variables *Supply* and *Idle* for 367 defined *Zones* can be found on GitHub[3]. Figure 5 gives an idea on how supply evolves over time $t$ in the Baltic Sea region. Each point illustrates the normalised cargo supply amount in tons. For sake of simplicity, we only plot supply independent of idle times here. We can see, that in the beginning of the year (for $0 < t < 20$) the supply in the northern regions, i.e. the need for resources, is higher, while for the rest of the year (for $20 < t < 40$) the supply slowly decreases and increases in the southern regions. The important part here is, that the supply for $20 < t < 40$ in the respective regions is more or less constant over a longer period of time.

**Algorithm 1:** Preventing Groundings

**Input:** DPRM $(G_0, G_{\rightarrow})$, Evidence- $\mathcal{E}$ and Querystream $Q$, Order $d$, Delay $\tau$, Symmetry Clusters $C$

**for** *each parfactor* $g_i \in G_0$ **do**
  $v_{|g_i} \leftarrow$ init violoation map `// see Definition 5.1`

**for** $t = 0, 1, ..., T$ **do**
  $\mathbf{E}_t \leftarrow$ get evidence from evidence stream $\mathcal{E}$
  Rearrange $\mathbf{E}_t$ to create evidence clusters according to parfactor partition $P_t$ `// see Definition 5.2`
  **for** *each evidence cluster* $\{E_{t,l} = e_{t,l}\}_{l=0}^{m} \in \boldsymbol{E}_t$ **do**
    $max(e_{t,l}) \leftarrow$ get dominating observation
    `// Align Evidence`
    **for** *each observation in* $E_{t,l}(x_i) \in \{E_{t,l} = e_{t,l}\}_{l=0}^{m}$ **do**
      **if** $e_{t,l} \neq max(e_{t,l})$ *and* $viol(x_i) < H$ **then**
        Dismiss observation $e_{t,l}$
        $viol(x_i) \leftarrow viol(x_i) + 1$
    `// Infer Evidence`
    **for** *each unobserved entity* $x_j$ *in* $P_t$ **do**
      Set $E_{t,l}(x_j) = max(e_{t,l})$
  Answer queries $Q_t$ from query stream $Q$

The idea behind MOP$_4$SA is to identify periods of time with similar behaviour for multiple entities. That means in our application, identifying zones with similar supply (or more specifically in the multivariate context supply/idle times) over a period of time.

[2] https://www.dma.dk/SikkerhedTilSoes/Sejladsinformation/AIS/
[3] https://github.com/FinkeNils/Processed-AIS-Data-Baltic-Sea-2020-v2

Next, we look into clustering based on the similarity graph as an outcome of similarity counting after applying the symbolisation.

## 6.2 Multivariate Symbolisation Scheme for Temporal Similarity Clustering

According to the procedure as introduced in Section 4.1.3, we apply the symbolisation scheme on the multivariate supply/idle-time time series as encoded in the parfactor $g_t^i$ and create one similarity graph as the basis for clustering. We compare different clustering algorithms as follows. Unfortunately, classical clustering methods do not achieve good results in high-dimensional spaces, like for DPRMs, which are specifically made to represent large domains. Problems that classical clustering approaches have is, that the smallest and largest distances in clustering differ only relatively slightly in a high-dimensional space [44]. For DPRMs, a similarity graph, representing the similarity of entities $z \in \mathcal{D}(Z)$, contains

$$\binom{|\mathcal{D}(Z)|}{2} \tag{15}$$

fully-connected nodes in the worst case, where $Z$ is a logvar representing a set of entities whose entity pairs share similar behaviour for least one time step. Here, Eq. (15) also corresponds to the number of dimensions a clustering algorithm has to deal with.

### 6.2.1 An Informal Introduction to Clustering

Generally, clustering algorithms can be divided into the four groups (a) centroid-based clustering, (b) hierarchical clustering, (c) graph-based clustering, and (d) density-based clustering.

We already pointed out the problem that classical clustering algorithms suffer due to their distance measures, which do not work well in high dimensional spaces. Especially centroid-based clustering approaches, like the well-known $k$-means algorithm or Gaussian Mixture Models, suffer, as they expect to find spherical or ellipsoidal symmetry. More specifically, in centroid-based clustering the assumption is that the points assigned to a cluster are spherical around the cluster centre and therefore no good clusters can be found due to the relatively equal distances. In hierarchical clustering time and space complexity is especially bad since the graph is iteratively split into clusters. Graph-based clustering algorithms, like spectral clustering, is known as being especially robust for high-dimensional data due to performing dimensionality reduction before running clustering [45]. One disadvantage, which also applies to clustering algorithms above, is that the number of clusters need to be specified as a hyperparameter in advance. In contrast, in density-based clustering approaches, like DBSCAN, the number of clusters are determined automatically while also handling noise. DBSCAN is based on a high-density of points. That means, clusters are dense regions, which are identified by running with a sliding window over dense points, making DBSCAN cluster shape independent.

For these reasons, we will compare spectral clustering and DBSCAN as part of MOP$_4$SA as follows. We start by informally introducing Spectral Clustering and DBSCAN.

**DBSCAN** works by grouping together points with many nearby neighbours, denoting points lying outside those regions as noise. In DBSCAN the two parameters $\epsilon$ and *minPoint* need to be provided from the outside, which correspond to the terms *Density Reachability* and *Density Connectivity* respectively. The idea behind DBSCAN is to identify points, that are reachable from another if it lies within a specific distance from it (Reachability), identifying core, border and noise points as the result of transitively connected points (Connectivity) [46]. More specifically, a core point is a point that has $m$ points within a distance of $n$ from itself, whilst a border point has at least one core point within the distance of $n$. All other points are considered as noise. The algorithm itself proceeds by randomly picking up a point from the dataset, that means, picking one node from the similarity graph, until every point was visited. All *minPoint*-points within a radius of $\epsilon$ around the randomly chosen point are considered as one cluster. DBSCAN proceeds by recursively repeating the neighbourhood calculations for each neighbouring point, resulting in $n$ clusters.

**Spectral Clustering** involves dimensionality reduction in advance before using standard clustering methods such as $k$-means. For dimensionality reduction, the similarity graph $\mathcal{W}$ is transformed into the so-called graph Laplacian matrix $L$, which describes the relations of the nodes and edges of a graph, where the entries are defined by

$$L_{ij} := \begin{cases} \deg(z_i) & \text{if } i = j \\ -1 & \text{if } i \neq j \text{ and } w_{ij} > 0\,, \\ 0 & \text{else} \end{cases} \tag{16}$$

with $\deg(z_i) = \sum_{j=1}^{|\mathcal{D}(Z)|} w_{ij}$. For decorrelation, data in the graph Laplacian matrix $L$ is decomposed into its sequence of eigenvalues and the corresponding eigenvectors. The eigenvectors form a new uncorrelated orthonormal basis and are thus suitable for standard clustering methods. The observations of the reduced data matrix whose columns contain the smallest $k$ eigenvectors can now be clustered using $k$-means. An observation assigned to cluster $C_i$ with $i = 1, ..., k$ can then be traced back to its entity $z \in \mathcal{D}(Z)$ by indices.

We evaluate both clustering approaches as part of SA$_4$PG in Section 6.3. To improve comparability, we compare both clustering approaches as described in the next Section.

### 6.2.2 Clustering Comparison Approach

We compare DBSCAN and Spectral Clustering in MOP$_4$SA by identifying clusters with each clustering approach and use resulting clusters within SA$_4$PG respectively, i.e., run SA$_4$PG once using clusters determined by DBSCAN and once using clusters determined by Spectral Clustering.

Since DBSCAN is able to automatically determine the numbers of clusters, we use DBSCAN to identify same and provide the resulting number of clusters as a parameter when performing Spectral Clustering. As DBSCAN is capable to also classifies noise, i.e., entities, which cannot be assigned to a cluster, we use the number of points classified as noise plus the number of clusters as the number of total clusters in Spectral Clustering. Further, for DBSCAN we provide $minPoints = 2$ as the minimum number of entities in a cluster to allow for the maximum number of clusters in general. The $eps$ parameter is automatically determined using the kneedle

algorithm [47]. For Spectral Clustering we just provide parameter $k$ for the total number of clusters, which was previously determined by DBSCAN. Note that the total number of clusters $n$, which was determined by DBSCAN, does not necessarily corresponds to the best number of clusters for Spectral Clustering. Nevertheless, we obtain results that show which algorithm, given the same input $n$, is better at separating entities in the multidimensional space.

In the following, we perform a detailed comparison between the two clustering approaches as part of SA$_4$PG with different parameters for the symbolisation scheme in MOP$_4$SA using the approach to compare the two clustering mechanisms as described here.

### 6.3 Preventing Groundings

MOP$_4$SA is affected by (a) the efficiency of the clustering algorithm used, (b) the similarity measure itself, (c) and its hyper parameters such as order $d$, delay $\tau$ and $\delta$ for the arithmetic mean as defined in Eq. (9). We evaluate MOP$_4$SA as part of SA$_4$PG. Specifically, we approximate symmetry clusters using MOP$_4$SA with different settings and (i) perform inference using the the symmetry clusters to prevent the model from grounding, and (ii) compare it with exact lifted inference and calculate Kullback Leibler divergence (KLD) between query result to determine the error introduced through SA$_4$PG. A KLD with $D_{KL} = 0$ indicates that both distributions are equal. Inference in DPRMs is performed by the lifted dynamic junction tree algorithm. Details can be found in [5].

We ran 54 experiments in total with different parameter combinations $d \in \{2, 3, 4\}$, $\tau \in \{1, 2, 3\}$, $\delta \in \{0.05, 0.1, 0.15\}$ and clustering through Spectral Clustering and DBSCAN. For comparison, we perform query answering given sets of evidence, i.e., we perform inference by answering the prediction query $P(Supply_t(Z), Idle_t(Z))$ for each time step $t \in \{4, \ldots, 51\}$ and obtain a marginal distribution for each entity $z \in \mathcal{D}(Z)$. We repeat query answering three times, once without preventing any groundings, once with preventing groundings using the clusters determined by DBSCAN, and once again with preventing groundings but using clusters determined by Spectral Clustering for each parameter combinations. Note that we only discuss results for a sub-selection of the parameter combinations, which give good results in terms of accuracy in inference under preventing groundings, while Table 1 and Table 2 at the end of this paper show the full results for all parameter combinations. Table 1 and Table 2 show results for time intervals $t \in \{[5, 10), [10, 15), [15, 20), [20, 25), [25, 30), [30, 35), [35, 40), [40, 45), [45, 50)\}$.

We evaluate runtime in seconds $s$, the number of groundings $\#_{gr}$ and KLD $D_{KL}$. Note that, $\#_{gr}$ shows the number of clusters after time $t$, while $n$ shows the initial number of clusters. Thus, the number of additional groundings at a specific timestep equals to $\#_{gr}$ minus $n$. Preventing groundings aims at keeping a lifted model as long as possible. A basic prerequisite for this is that similarities exists in the data. As to that, the variable $n_{\geq 1}$ shows the number of initial clusters, which contain more than one entity directly after clustering, i.e., clusters in which similarly behaving entities have been arranged. Note that similar to $n$, $n_{\geq 1}$ does not change over time. With increasing order $d$ the number of neighbouring data points are increasing, i.e., the classification contains more long term patterns. With increasing delay $\tau$, long-term behaviour is extended even further, while also allowing for temporary deviations. For data range dependence, in similarity counting we test different delta $\delta_{\leq}$.

The number of clusters with more than one entity $n_{>1}$ relative to the total number of clusters $n$ are important in evaluating how well symmetries are exploited. When $n$ is small, i.e. when $n$ is *significantly* smaller than the total number of entities $|\mathcal{D}(Z)|$, a value of $n_{>1}$ close to $n$ is desirable since it indicates that many entities show symmetries with each other. If $n_{>1}$ is significantly smaller than $n$, then only a few entities show symmetries, which on the one hand leads to a better accuracy in the inference since many entities are considered at a ground level, but on the other hand runtime will suffer greatly. As to that, Figure 6 shows a comparison for different parameter combinations and clustering approaches. The red line denotes the total number of clusters $n$ independently of the number of entities included in a cluster, while the bars only show the number of clusters with more than one entity $n_{>1}$. Note that we also include entities, which are treated on a ground level already by the time after learning clusters, in the total number of cluster, i.e., clusters can also only include one entity. Since lifting highly depends on the degree of similarities, only those clusters with more than one entity are of interest. Each pair of bar plots correspond to a different experiment with different parameter combinations.

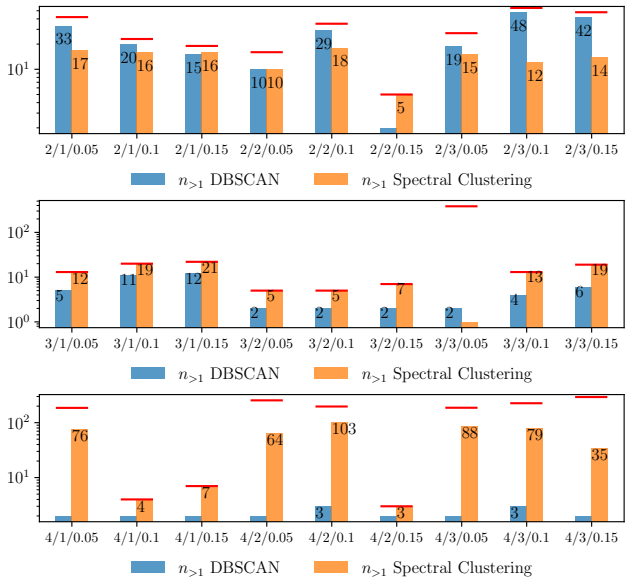
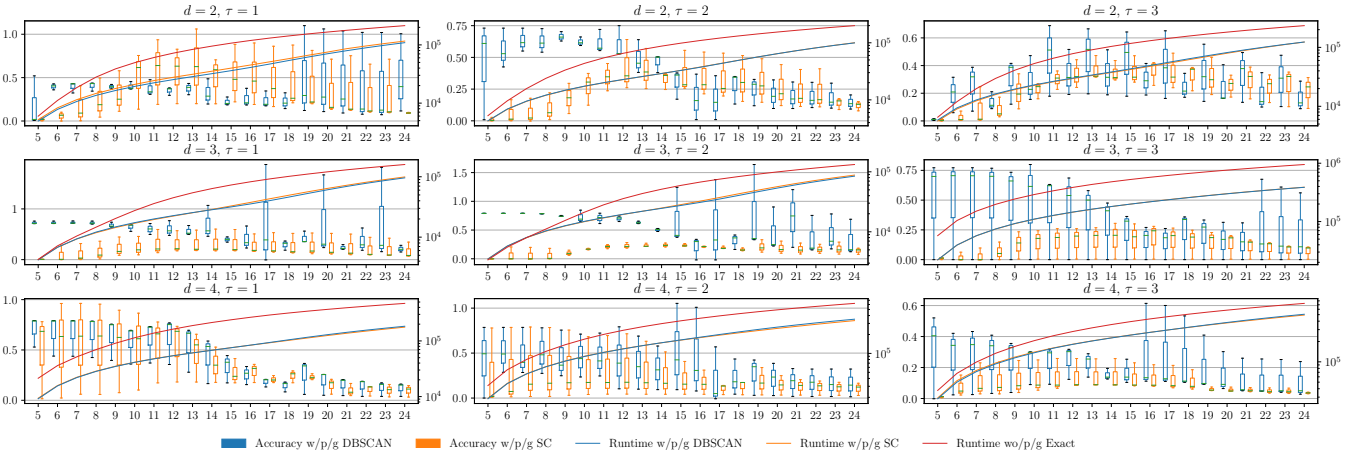
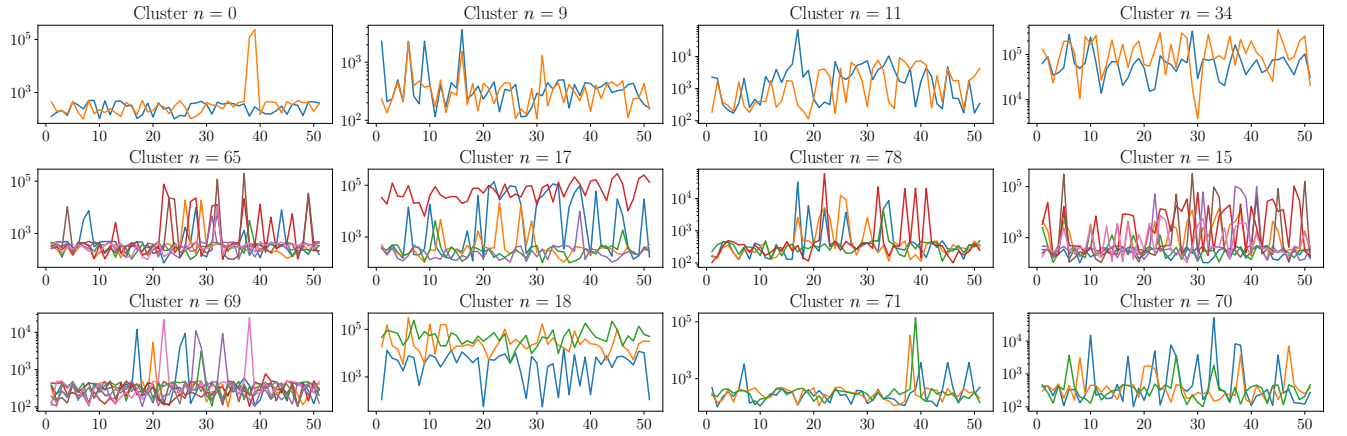



Figure 6: Comparison of number of clusters with more than one entity between DBSCAN and Spectral Clustering.

Due to space limitations we shorten order $d$, delay $\tau$ and delta $\delta_{leq}$ in the graph by the triple $d/\tau/\delta_{leq}$. From Fig. 6 it is most noticeable that the number of clusters with more than one entity $n_{\geq 1}$ for higher orders $d$ is much less than for lower orders. This intuitively makes sense, since with higher orders $d$ long term behaviour is captured much better than with lower orders and thus only a few clusters can be determined. For $d = 2$ much more clusters are identified since a smaller time span is considered resulting in a higher possibility of showing similarities. This observation also applies to increasing delays $\tau$. The experiment with order $d = 3$, delay $\tau = 3$ and delta $\delta_{\leq} = 0.05$ is a good example for cases where not many similarities have been identified, but many entities are treated on a




(a) Accuracy and runtime in inference under SA$_4$PG for different parameter combinations.




(b) Supply over time $t$ for a selection of clusters, which have been learned based on Supply/Idle time data for $0 < t \leq 4$ using Spectral Clustering.

Figure 7: Accuracy and runtime data based on query results under SA$_4$PG including raw supply data for entities within clusters.

ground level, i.e., accuracy will be good, but runtime will suffer. In the following, we look at accuracy results and will also come back to this example.

By comparing the KLD $D_{KL}$ as a result of inference without preventing groundings and with preventing groundings based on clusters determined by DBSCAN and Spectral Clustering, it is noticeable that for clusters determined by Spectral Clustering in average a lower $D_{KL}$ compared to DBSCAN results. This is explainable with better handling of higher dimensional data in Spectral Clustering. Figure 7a shows a comparison between the accuracy for both cases and different parameter combination. Each subplot corresponds to a different order $d$ and delay $\tau$, while the box-plot itself shows the variation of the accuracy over different deltas $\delta_{\leq} = \{0.05, 0.1, 0.15\}$ over time $t$. Note that we only plot data until $t = 24$ for better visibility and as the effect of any wrong evidence, which was brought in by preventing groundings, starts to level off. This happens since groundings are only prevented until the threshold $H$ is reached, i.e., any other evidence afterwards at a later timestep balance out the effect of any wrong evidence at an early timestep after learning symmetry clusters. The blue box-plots in Fig. 7a correspond to the KLD $D_{KL}$ with DBSCAN as the clustering approach in MOP$_4$SA, while the orange box-plots correspond to the KLD $D_{KL}$ with on Spectral Clustering as the clustering approach in MOP$_4$SA. The solid blue, orange and red line correspond to the runtime for answering a query for the specific time step. From the plots, we can see that for higher orders and delays, i.e., with increasing time spans each ordinal represents, that $D_{KL}$ is decreasing. Considering the total number of clusters for each experiment (see Fig. 6), this follows as not many similarities can be found in the data, but more entities are handled on a ground level, i.e., increasing accuracy. On the other hand, runtime drastically increases as symmetries are no longer exploited. Compared to exact reasoning, runtime is noticeably smaller in inference under SA$_4$PG. To lock again at the experiment with order $d = 3$, delay $\tau = 3$ (as highlighted above), the KLD $D_{KL}$ is considerably small especially for Spectral clustering, but the runtime of the inference is very poor compared to all other experiments.

In SA$_4$PG, the violation threshold $H$ is set to 5, i.e., groundings due to any inequalities are prevented for an entity $H$ times. After $t = 10$ the number of groundings $\#_{gr}$ (see Table 1) are still the same as after learning the entity similarity cluster, i.e., all groundings are prevented. It is expected that the majority of groundings are prevented in the initial timesteps after learning the clusters. Still, if entities behave similarity in early timesteps, the threshold $H$ is reached far later in time. Thus, if in clustering based on the similarity graph

the entities with similarities are identified better, then groundings will occur much later in time. The longer $D_{KL}$ stays small, the better cluster fit, i.e, the error introduced in inference through preventing groundings is kept small. In Fig. 7a we see that the accuracy suffers approximately for all experiments around $t = 10$, i.e, after 4 more timesteps after learning the clusters. Figure 7b depicts raw supply data for a selection of clusters as a result of running MOP$_4$SA based on data for $t = [0, 4)$ with parameters $d = 2$, $\tau = 1$ and $\delta = 0.05$ for symbolisation and Spectral Clustering. Even though only providing a small amount of training data, we can see that symmetrical behaviour continues for most of the clusters until approx. $t = 10$, like especially for clusters $n \in \{0, 11, 34, 65, 78, 69, 71\}$ and therefore support the insight, which we have got based on Fig. 7a. For simplicity only raw supply data is plotted even though symmetry clusters are determined based on supply/idle data.

The best results are achieved with Spectral Clustering as part of SA$_4$PG for the parameter combinations $d = 2$, $\tau = 1$, $\delta_{\leq} = 0.1$, $d = 3$, $\tau = 3$, $\delta_{\leq} = 0.1$ and $d = 3$, $\tau = 2$, $\delta_{\leq} = 0.05$, which we will also further refer to in the following Section. Generally, when reasoning under time constraints, preventing grounds is a reasonable approach as it prevents groundings in the long term and therefore speeds up inference.

Entity similarity can change over time, i.e., to further prevent the model from grounding it is beneficial to relearn symmetry structures at some time. In the following we propose MOP$_4$SCD and use it to identify points in time when relearning clusters is beneficial.

# 7 Multivariate Ordinal Pattern for Symmetry Change Detection (MOP$_4$SCD)

Symmetries in temporal models can change over time as already seen in Fig. 5. Therefore, symmetry cluster, after they have been learned, may only stay valid for a certain period of time. Further, some are valid for a longer period of time, some not. To identify points in time when relearning symmetry clusters is reasonable, we use the similarity graph as an intermediate output of running MOP$_4$SA and check if the similarity graph has changed *significantly*. More specifically, we continue running MOP$_4$SA for every timestep, but instead of for continuously relearning symmetry clusters, we prevent relearning clusters in MOP$_4$SA after the initial sync run until the graph has changed *significantly enough*. To identify such points in time with a significant change, we introduce MOP$_4$SCD taking as inputs a similarity graph for two consecutive timesteps and calculating a distance measure between both. In case the distance measure is above a certain threshold we consider those points as change points to trigger the cluster relearning process. MOP$_4$SCD is based on the assumption, that clusters no longer stay valid, if entities within a cluster no longer show the same similarity to its cluster entities as in the previous timesteps, i.e., the similarity counts is no longer proportionally scaling as before. Those entities might transition to another cluster, since its showing more similarity with another cluster. Informally, if the similarity graph changes over time in a *constant and balanced way*, symmetry clusters stay valid, but if the similarity graph changes over time in an *unbalanced manner*, i.e., if similarity counts change significantly, there is a change in the structure of the symmetry clusters. To illustrate that, let us look at Figure 8. The Figure shows a similarity graph based on which two clusters have been identified.

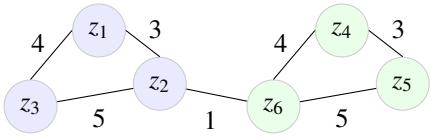



Figure 8: Overview of potential unbalanced changes in a similarity graph.

Nodes $S^1 = \{z_1, z_2, z_3\}$ (coloured in blue) denote a cluster $S^1$ and the nodes $S^2 = \{z_4, z_5, z_6\}$ (coloured in green) denote another cluster $S^2$. Both clusters are connected through nodes $z_2$ and $z_6$ since for both a similarity was measured at any timestep before learning clusters. Relearning clusters becomes necessary if the cluster structure itself changes. This happens either

(a) if similarities between entities of different clusters changes, e.g., if the similarity between $z_2$ and $z_6$ increases and might require to merge the clusters or even split them into more than two clusters, which we denote as a *unbalanced interclusteral change*,

(b) or if similarities within a cluster change disproportionately, e.g., if similarities for $S^2$ changes only for a subset of the entities such as for $z_4$ and $z_5$ but not proportionally for all entities such as $z_4$ and $z_6$ and $z_6$ and $z_5$ requiring to split the cluster even further, which we denote as a *unbalanced intraclusteral change*.

As follows we define both unbalanced interclusteral and intraclusteral change measures and combine both into a distance measure denoting the unbalanced change between consecutive timesteps. Both unbalanced inter- and intraclusteral changes are determined based on the similarity graph $\mathcal{W}^t$ from the current to the next time step $\mathcal{W}^{t+1}$ under current symmetry clusters $\mathcal{S}$, with interclusteral changes defined as

$$d_{inter}(\mathcal{W}^t, \mathcal{W}^{t+1}, \mathcal{S}) = \sum_{\substack{S^i \in \mathcal{S} \\ S^j = en(S^i) \cap en(\mathcal{S})}} \frac{\sum_{\substack{i \in en(S^i) \\ j \in en(S^j)}} \left[ w_{ij}^{t+1} = w_{ij}^t + 1 \right]}{|en(S^i)| \cdot |en(S^j)|} \tag{17}$$

where $[x] = 1$ if $x$ and, 0 otherwise for $en(S^i) \cap en(S^j) = \emptyset$ and intraclusteral changes defined as

$$d_{intra}(\mathcal{W}^t, \mathcal{W}^{t+1}, \mathcal{S}) = \sum_{S^i \in \mathcal{S}} \frac{\sum_{i,j \in en(S^i), i<j} \left[ w_{ij}^{t+1} - w_{ij}^t = 0 \right]}{|en(S^i)| \cdot |en(S^i)|} \tag{18}$$

where $[x] = 1$ if $x$ and, 0. Both $d_{inter}$ and $d_{intra}$ are merged into one combined measure with

$$d(\mathcal{W}^t, \mathcal{W}^{t+1}, \mathcal{S}) = \frac{d_{inter} + d_{intra}}{|\mathcal{S}|}. \tag{19}$$

Simply speaking, $d_{inter}$ counts the number of increases in weights across different clusters $S^i$ and $S^j$ such as shown in Fig. 8 for entities $z_2$ and $z_6$. The resulting count is normalised by dividing through the number of comparison between entity pairs of the clusters $en(S^i)$ and $en(S^j)$, resulting in measure between 0 and 1 with a value close

to 1 denoting a maximum dissimilarity. Similarly, $d_{intra}$ counts the occurrences of no weight increases within entity pairs of a similar cluster $S^i$. To ensure that entities within a cluster continue to behave the same, weights should proportionally increase equally distributed within the cluster. If there is no increase in weights most likely the entities discontinue to behave similarly. The resulting count is equally normalised with a value close to 1 denoting a maximum dissimilarity. Finally, both $d_{inter}$ and $d_{intra}$ are combined in a single measure also count normalised to determine a distance measure between 0 and 1. If $d(\mathcal{W}^t, \mathcal{W}^{t+1}, \mathcal{S}) = 0$, the change in the similarity graph is balanced, if $d(\mathcal{W}^t, \mathcal{W}^{t+1}, \mathcal{S}) > 0$, it is unbalanced. If $d(\mathcal{W}^t, \mathcal{W}^{t+1}, \mathcal{S}) > b$, $b \in \mathbb{N}_{>0}$ it may be worthwhile to (re)perform clustering and (re)build symmetry clusters.

As follows, we evaluate MOP$_4$SCD based on clusters determined by MOP$_4$SA for the same parameters as in the experiments performed in Section 6.

## 8 MOP$_4$SCD in Application

We evaluate MOP$_4$SCD based on clusters determined using MOP$_4$SA as described in Section 6. Since Spectral Clustering works better than DBSCAN in identifying clusters, we here only use clusters determined by Spectral Clustering as part of MOP$_4$SA. We run experiments 27 experiments in total for the same parameter combinations $d \in \{2, 3, 4\}$, $\tau \in \{1, 2, 3\}$ and $\delta \in \{0.05, 0.1, 0.15\}$ as in Section 6. For each experiment we calculate $d(\mathcal{W}^t, \mathcal{W}^{t+1}, \mathcal{S})$ for timesteps $t = 5, \ldots, 51$. Clusters are learned based on a similarity graph with data for $t = 1, \ldots 4$.

In this Section we discuss results for a sub-selection of the parameter combinations, which give good results in terms of accuracy in inference under preventing groundings as seen in Section 4, while Table 3 at the end of this paper shows detailed results for all parameter combinations. Each column in Table 3 shows the distance measure for consecutive timesteps, e.g., for $t = 5$, the distance is derived based on the similarity graph for timestep $t = 4$ to $t = 5$, i.e., $d(\mathcal{W}^4, \mathcal{W}^5, \mathcal{S})$. Note that since $d(\mathcal{W}^4, \mathcal{W}^5, \mathcal{S})$ is calculated for two consecutive timesteps, the distance measure has to be added up over time to derive the overall distance between more then two timesteps. Overall, the distance measure varies for different parameter combinations with in the optimal case showing an unbalanced change in weights of approximately 1.6% and in the worst case of approximately 22.4% between two consecutive timesteps. The distance measure is highly affected by the number of clusters $n$. In the case that the number of clusters with more than one entity $n_{>1}$ is considerably small compared to the total number of clusters $n$, the distance measure $d(\mathcal{W}^t, \mathcal{W}^{t+1}, \mathcal{S})$ is also considerably low since $d_{inter}$ and $d_{intra}$, see Eq. (17) and Eq. (18), always return no unbalanced change for clusters with just one entity, i.e., for entities which are already being treated on a ground level. MOP$_4$SA aims at preventing groundings to speed up inference, i.e., lead to an increase in runtime. Therefore, choosing parameters $d$, $\tau$ and $\delta$ for MOP$_4$SA and consequently for MOP$_4$SCD is a trade-of between losses in accuracy and a speed up in inference.

The parameter combinations $d = 2$, $\tau = 1$, $\delta_{\leq} = 0.1$, $d = 3$, $\tau = 3$, $\delta_{\leq} = 0.1$ and $d = 3$, $\tau = 2$, $\delta_{\leq} = 0.05$ give good results in MOP$_4$SA as shown in Section 6. Results for MOP$_4$SCD also support this. Figure 9 shows the KLD $D_{KL}$ in conjunction with results from MOP$_4$SCD. Each subplot corresponds to a different parameter combination with the blue line corresponding to the KLD $D_{KL}$, the solid red line to the distance measure $d(\mathcal{W}^t, \mathcal{W}^{t+1}, \mathcal{S})$ for two consecutive timesteps $t$ and $t + 1$ and the dashed red line for the cumulative distance measure, i.e., from $t = 0$ until the current timestep $t$. Note that the cumulative distance is log scaled and can be read of from the right $y$-axis. The highlighted red area in each subplots mark the interval when the cumulative distance measure becomes greater then 50% until it has reached 100%, i.e., with a change of 100% that all relations between all entities have been affected.

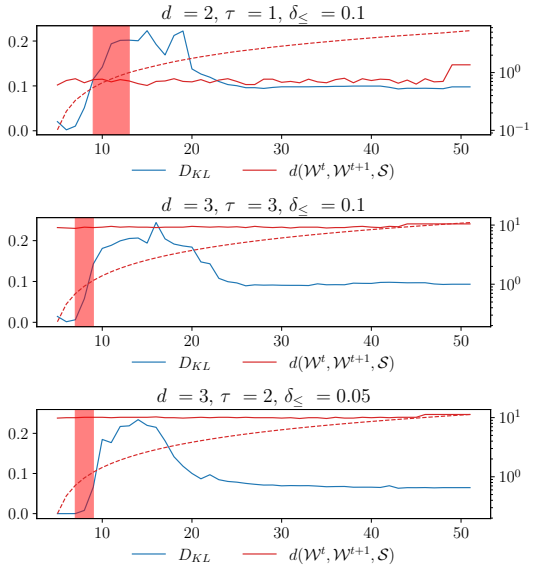



Figure 9: Results of MOP$_4$SCD for three parameter combinations which show the best results for MOP$_4$SA and SA$_4$PG. Further results can be found in the appendix.

Similar to the experiments in Section 6, clusters $\mathcal{S}$ have been determined by MOP$_4$SA based on data for $0 > t \leq 4$. For all upcoming timesteps the clusters $\mathcal{S}$ have been used to prevent groundings, i.e., execute SA$_4$PG as part of inference, see Algorithm 1. The KLD $D_{KL}$ for all experiments as shown in Fig. 9 similarly raises up to a value of approx. 0.25 with its peak around $t = 15$. In contrast $d(\mathcal{W}^t, \mathcal{W}^{t+1}, \mathcal{S})$ varies across experiments and has for the experiment with parameter combination $d = 2$, $\tau = 1$ and $\delta_{\leq}0.1$ its best value of approx. 0.1, i.e, a unbalanced change of approx. 10% over time. For the two other experiments $d(\mathcal{W}^t, \mathcal{W}^{t+1}, \mathcal{S})$ is with 0.22 similar. Correspondingly, the cumulative distance reaches a value of 0.5 at timestep $t = 9$ until it reaches a value of 1 at $t = 14$ for the first experiment, while for the two other experiments the cumulative distance reaches a value of 0.5 at $t = 7$ and a value of 1 already at $t = 9$. I.e., clusters $S$ are valid for a longer period of time using MOP$_4$SA with a parameter combination of $d = 2$, $\tau = 1$, $\delta_{\leq} = 0.1$. Further, for that parameter combination, $D_{KL}$ settles off once a cumulative distance of 1 has reached. Settling off happens due to the amount of new evidence leading to more groundings removing the effect of any wrongly introduced evidence in previous timesteps. Relearning clusters at a threshold of 0.5 is here beneficial to prevent the $D_{KL}$ from further increasing. That means relearning at $t = 9$, i.e., clusters are valid for approx. 4 timesteps after learning

them, which corresponds to a full month in our example application and therefore is a good result to considerably speed up inference while only introducing a small error in inference.

# 9 Conclusion and Future Work

Evidence lead to groundings in dynamic probabilistic relational models over time, negating runtime benefits in lifted inference. This paper provides MOP$_4$SA, SA$_4$PG and MOP$_4$SCD as a rich toolset to identify model symmetries as part of the model construction process, use those symmetries to maintain a lifted representation by preventing groundings a priori and detect changes in model symmetries after the model construction process. Preventing groundings a priori to maintain any lifted representation is important in lifted inference to preserve its runtime benefits. MOP$_4$SA detects symmetries across entities of the models domain using a multivariate ordinal pattern symbolisation approach and building a similarity graph for spectral clustering to identify sets of entities with symmetrical behaviour regarding a context of the model (symmetry clusters). Symmetry clusters are used in SA$_4$PG as part of query answering to prevent any unnecessary model splits by evidence, e.g., due to one time events. Symmetry structures can change over time, which MOP$_4$SCD detects based on the similarity graph, an intermediate output of MOP$_4$SA, and provide a distance measure denoting the degree of any unbalanced structural change to identify points in time when relearning symmetry clusters is beneficial.

The main contribution of this paper are the extension by theoretical and experimental results on the original papers [1, 2] and the introduction of MOP$_4$SCD as a mechanism to detect structural changes to complement MOP$_4$SA, SA$_4$PG as a rich toolset to prevent groundings a priori. We show, that MOP$_4$SA requires only a small amount of *training data* to come up with a good approximation of symmetry structures. Generally, MOP$_4$SA aims at determining symmetry structures which stay valid for shorter time periods. This follows, since MOP$_4$SA is not capable to capture any reoccurring patterns or periodicity, e.g., due to seasonality. MOP$_4$SA can be extended to capture such behaviour, but this would also increase the complexity of the overall approach. Due to this and since capturing symmetries for longer time spans, especially in real-world applications which normally change much faster, is not feasible, we focus with MOP$_4$SA as being a simple and easy to compute framework, requiring only few historical data points for learning, to identify symmetries for the short term future. In addition to MOP$_4$SA, MOP$_4$SCD supports in inference by identifying points in time when relearning clustering for SA$_4$PG is reasonable.

With preventing groundings a priori we complement existing approaches, which focus on retaining lifted representation after a model has already been splitted. In general, our approach works well with any other approach undoing splits after they occurred when moving forward in time, e.g., in message passing by merging sets of entities when those align again, denoted as *temporal approximate merging*, as proposed in [25]. Combining both kind of approaches brings together the best of both worlds: (a) While with *determining approximate model symmetries* a priori, we can use the full amount of historical training data to prevent groundings, (b) and with *temporal approximate merging*, we can merge non-preventable parfactor splits even after they occurred, i.e., a posterior.

Since MOP$_4$SA is designed to work with small amounts of data to provide symmetry clusters very quickly for the short term future, the overhead MOP$_4$SA and MOP$_4$SCD bring into query answering need to be kept to a minimum. Applying the symbolisation scheme to identify symmetries is already a suitable mechanism, but with the clustering approach we still depend on existing approaches, which are considerably costly. The investigation of more performant clustering approaches, e.g., taking advantage of some sort of incremental changes to clustering after the initial learning step, are left for future work.

# List of Symbols

| | |
|---|---|
| $\mathbf{R}$ | set of random variables |
| $\mathbf{L}$ | set of logical variables |
| $\Phi$ | set of factor names |
| $\mathbf{D}$ | set of entities |
| $\mathcal{D}(L)$ | domain of a logvar |
| $C, (\mathcal{X}, C_{\mathbf{X}})$ | constraint restricting logical variables |
| $A(L_1, \ldots, L_n)$ | parameterised logical variable (PRV) |
| $g, \phi(\mathcal{A})_{\vert C}$ | parfactor |
| $gr(P)$ | grounding |
| $lv(P)$ | logical variables |
| $\mathcal{R}(A)$ | range of a PRV |
| $G$ | model |
| $G_t$ | local model |
| $\mathbf{E}$ | evidence, set of events |
| $Q$ | query term |
| $\mathcal{X}$ | multivariate time series |
| $\tau$ | delay between successive time points |
| $d$ | order of ordinal pattern |
| $w_{ij}$ | similarity count |
| $\mathcal{W}$ | similarity graph |
| $S$ | symmetry cluster |
| $en(S)$ | objects in a symmetry cluster |
| $\mathbf{S}$ | set of symmetry clusters |
| $P$ | parfactor partition |
| $L$ | Laplacian matrix |
| $D_{KL}$ | Kullback-Leibler divergence |
| $\delta_{\leq}$ | mean delta |
| $d(\mathcal{W}^t, \mathcal{W}^{t+1}, \mathcal{S})$ | similarity change measure |

# References

[1] N. Finke, M. Mohr, "A Priori Approximation of Symmetries in Dynamic Probabilistic Relational Models," in S. Edelkamp, R. Möller, E. Rueckert, editors, KI 2021: Advances in Artificial Intelligence, 309–323, Springer International Publishing, Cham, 2021.

[2] N. Finke, R. Möller, M. Mohr, "Multivariate Ordinal Patterns for Symmetry Approximation in Dynamic Probabilistic Relational Models," in AI 2021: Advances in Artificial Intelligence - 34rd Australasian Joint Conference, Lecture Notes in Computer Science (LNCS), Springer International Publishing, In Press.

[3] N. Finke, M. Gehrke, T. Braun, T. Potten, R. Möller, "Investigating Matureness of Probabilistic Graphical Models for Dry-Bulk Shipping," in M. Jaeger, T. D.

Nielsen, editors, Proceedings of the 10th International Conference on Probabilistic Graphical Models, volume 138 of *Proceedings of Machine Learning Research*, 197–208, PMLR, 2020.

[4] Y. Xiang, K.-L. Poh, "Time-Critical Dynamic Decision Making," 2013.

[5] M. Gehrke, T. Braun, R. Möller, "Lifted Dynamic Junction Tree Algorithm," in Proceedings of the International Conference on Conceptual Structures, 55–69, Springer, 2018.

[6] D. Poole, "First-order Probabilistic Inference," in Proc. of the 18th International Joint Conference on Artificial Intelligence, 985–991, 2003.

[7] D. Akyar, "The Effects of Global Economic Growth on Dry Bulk Shipping Markets and Freight Rates," 2018.

[8] Z. Wang, X. Wu, K. L. Lo, J. J. Mi, "Assessing the management efficiency of shipping company from a congestion perspective: A case study of Hapag-Lloyd," Ocean & Coastal Management, **209**, 105617, 2021, doi: https://doi.org/10.1016/j.ocecoaman.2021.105617.

[9] C. Jiang, Y. Wan, A. Zhang, "Internalization of port congestion: strategic effect behind shipping line delays and implications for terminal charges and investment," Maritime Policy & Management, **44**(1), 112–130, 2017, doi: 10.1080/03088839.2016.1237783.

[10] T. Notteboom, "The Time Factor in Liner Shipping Services," Maritime Economics and Logistics, **8**, 19–39, 2006, doi:10.1057/palgrave.mel.9100148.

[11] M. Niepert, G. Van den Broeck, "Tractability through Exchangeability: A New Perspective on Efficient Probabilistic Inference," in AAAI-14 Proceedings of the 28th AAAI Conference on Artificial Intelligence, 2467–2475, AAAI Press, 2014.

[12] G. V. den Broeck, M. Niepert, "Lifted Probabilistic Inference for Asymmetric Graphical Models," CoRR, **abs/1412.0315**, 2014.

[13] N. Taghipour, D. Fierens, J. Davis, H. Blockeel, "Lifted Variable Elimination: Decoupling the Operators from the Constraint Language," Journal of Artificial Intelligence Research, **47**(1), 393–439, 2013.

[14] K. Kersting, "Lifted Probabilistic Inference," in Proceedings of the 20th European Conference on Artificial Intelligence, ECAI'12, 33–38, IOS Press, NLD, 2012.

[15] G. Van den Broeck, A. Darwiche, "On the Complexity and Approximation of Binary Evidence in Lifted Inference," in C. J. C. Burges, L. Bottou, M. Welling, Z. Ghahramani, K. Q. Weinberger, editors, Advances in Neural Information Processing Systems, volume 26, Curran Associates, Inc., 2013.

[16] P. Singla, A. Nath, P. Domingos, "Approximate Lifting Techniques for Belief Propagation," in Proceedings of the Twenty-Eighth AAAI Conference on Artificial Intelligence, AAAI'14, 2497–2504, AAAI Press, 2014.

[17] P. Singla, P. Domingos, "Lifted First-Order Belief Propagation," in Proceedings of the 23rd National Conference on Artificial Intelligence - Volume 2, AAAI'08, 1094–1099, AAAI Press, 2008.

[18] B. Ahmadi, K. Kersting, M. Mladenov, S. Natarajan, "Exploiting symmetries for scaling loopy belief propagation and relational training," Machine Learning, **92**, 91–132, 2013.

[19] C. Sutton, A. Mccallum, "Piecewise Training for Structured Prediction," Machine Learning, **77**, 165–194, 2009, doi:10.1007/s10994-009-5112-z.

[20] D. Venugopal, V. Gogate, "Evidence-Based Clustering for Scalable Inference in Markov Logic," in T. Calders, F. Esposito, E. Hüllermeier, R. Meo, editors, Machine Learning and Knowledge Discovery in Databases, 258–273, Springer Berlin Heidelberg, Berlin, Heidelberg, 2014.

[21] P. Miettinen, T. Mielikäinen, A. Gionis, G. Das, H. Mannila, "The Discrete Basis Problem," in J. Fürnkranz, T. Scheffer, M. Spiliopoulou, editors, Knowledge Discovery in Databases: PKDD 2006, Springer Berlin Heidelberg, Berlin, Heidelberg, 2006.

[22] N. Metropolis, A. W. Rosenbluth, M. N. Rosenbluth, A. H. Teller, E. Teller, "Equation of State Calculations by Fast Computing Machines," The Journal of Chemical Physics, **21**(6), 1087–1092, 1953.

[23] W. K. Hastings, "Monte Carlo Sampling Methods Using Markov Chains and Their Applications," Biometrika, **57**(1), 97–109, 1970.

[24] A. Nath, P. Domingos, "Efficient Lifting for Online Probabilistic Inference," volume 2, 2010.

[25] M. Gehrke, R. Möller, T. Braun, "Taming Reasoning in Temporal Probabilistic Relational Models," in Proceedings of the 24th European Conference on Artificial Intelligence (ECAI 2020), 2020, doi:10.3233/FAIA200395.

[26] R. Agrawal, C. Faloutsos, A. Swami, "Efficient similarity search in sequence databases," in Lecture Notes in Computer Science, volume 730, Springer Verlag, 1993.

[27] E. Keogh, K. Chakrabarti, M. Pazzani, S. Mehrotra, "Dimensionality Reduction for Fast Similarity Search in Large Time Series Databases," in Knowledge and Information Systems, 263–286, 2001, doi:10.1021/acsami.7b03579.

[28] S. Kramer, "A Brief History of Learning Symbolic Higher-Level Representations from Data (And a Curious Look Forward)," in Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence, IJCAI-20, 4868–4876, 2020.

[29] J. B. Kruskal, M. Liberman, "The Symmetric Time Warping Problem: From Continuous to Discrete," in Time Warps, String Edits and Macromolecules: The Theory and Practice of Sequence Comparison, Addison-Wesley Publishing Co., 1983.

[30] C.-C. M. Yeh, Y. Zhu, L. Ulanova, N. Begum, Y. Ding, H. A. Dau, D. F. Silva, A. Mueen, E. Keogh, "Matrix Profile I: All Pairs Similarity Joins for Time Series: A Unifying View That Includes Motifs, Discords and Shapelets," in 2016 IEEE 16th International Conference on Data Mining (ICDM), 1317–1322, 2016.

[31] F. Petitjean, J. Inglada, P. Gancarski, "Satellite Image Time Series Analysis Under Time Warping," IEEE Transactions on Geoscience and Remote Sensing, **50**(8), 2012.

[32] S. Salvador, P. Chan, "FastDTW: Toward Accurate Dynamic Time Warping in Linear Time and Space," 70–80, 2004.

[33] D. F. Silva, G. E. A. P. A. Batista, "Speeding Up All-Pairwise Dynamic Time Warping Matrix Calculation," in Proceedings of the 2016 SIAM International Conference on Data Mining, 837–845, Society for Industrial and Applied Mathematics, 2016.

[34] B. Chiu, E. Keogh, S. Lonardi, "Probabilistic Discovery of Time Series Motifs," in Proceedings of the Ninth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 493–498, 2003.

[35] C. Bandt, B. Pompe, "Permutation Entropy: A Natural Complexity Measure for Time Series," Physical Review Letters, **88**(17), 4, 2002.

[36] M. Mohr, F. Wilhelm, M. Hartwig, R. Möller, K. Keller, "New Approaches in Ordinal Pattern Representations for Multivariate Time Series," in Proceedings of the 33rd International Florida Artificial Intelligence Research Society Conference, 2020.

[37] K. Keller, T. Mangold, I. Stolz, J. Werner, "Permutation Entropy: New Ideas and Challenges," Entropy, **19**(3), 2017.

[38] A. B. Piek, I. Stolz, K. Keller, "Algorithmics, Possibilities and Limits of Ordinal Pattern Based Entropies," Entropy, **21**(6), 2019.

[39] K. Keller, S. Maksymenko, I. Stolz, "Entropy Determination Based on the Ordinal Structure of a Dynamical System," Discrete and Continuous Dynamical Systems - Series B, **20**(10), 3507–3524, 2015.

[40] I. Stolz, K. Keller, "A General Symbolic Approach to Kolmogorov-Sinai Entropy," Entropy, **19**(12), 2017.

[41] A. Antoniouk, K. Keller, S. Maksymenko, "Kolmogorov-Sinai entropy via separation properties of order-generated $\sigma$-algebras," Discrete & Continuous Dynamical Systems, **34**(5), 1793–1809, 2014.

[42] K. Keller, "Permutations and the Kolmogorov-Sinai Entropy," Discrete & Continuous Dynamical Systems, **32**(3), 891–900, 2012.

[43] D. Yang, L. Wu, S. Wang, H. Jia, K. X. Li, "How big data enriches maritime research – a critical review of Automatic Identification System (AIS) data applications," Transport Reviews, **39**(6), 755–773, 2019, doi: 10.1080/01441647.2019.1649315.

[44] R. Bellman, Adaptive control processes: A guided tour, Princeton legacy library, Princeton University Press, 2015.

[45] A. L. Bertozzi, E. Merkurjev, "Chapter 12 - Graph-based optimization approaches for machine learning, uncertainty quantification and networks," in R. Kimmel, X.-C. Tai, editors, Processing, Analyzing and Learning of Images, Shapes, and Forms: Part 2, volume 20 of *Handbook of Numerical Analysis*, 503–531, Elsevier, 2019.

[46] E. Schubert, J. Sander, M. Ester, H. P. Kriegel, X. Xu, "DBSCAN Revisited, Revisited: Why and How You Should (Still) Use DBSCAN," ACM Trans. Database Syst., **42**(3), 2017, doi:10.1145/3068335.

[47] V. Satopaa, J. Albrecht, D. Irwin, B. Raghavan, "Finding a "Kneedle" in a Haystack: Detecting Knee Points in System Behavior," in 2011 31st International Conference on Distributed Computing Systems Workshops, 166–171, 2011, doi:10.1109/ICDCSW.2011.20.

Table 1: Accuracy scores of $MOP_4SA$ in $SA_4PG$. Further results can be found in the appendix

| | | | | DBSCAN | | | | Spectral Clustering | | | | Exact | | DBSCAN | | | Spectral Clustering | | | Exact | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| $d$ | $\tau$ | $\delta_{\leq}$ | $n$ | $n_{>1}$ | $\#_{gr}$ | $s$ | $D_{KL}$ | $n_{>1}$ | $\#_{gr}$ | $s$ | $D_{KL}$ | $\#_{gr}$ | $s$ | $\#_{gr}$ | $s$ | $D_{KL}$ | $\#_{gr}$ | $s$ | $D_{KL}$ | $\#_{gr}$ | $s$ |
| | | | | **t = [05, 10)** | | | | | | | | | | **t = [10, 15)** | | | | | | | |
| | | 0.05 | 42 | 33 | 42 | 19.6 | 0.342 | 17 | 42 | 19.1 | 0.317 | 306 | 38.7 | 66 | 37.8 | 0.294 | 81 | 37.5 | 0.621 | 357 | 95.2 |
| | 1 | 0.1 | 23 | 20 | 23 | 15.3 | 0.281 | 16 | 23 | 19.8 | 0.036 | 310 | 29.8 | 54 | 29.9 | 0.397 | 64 | 39.6 | 0.185 | 357 | 78.1 |
| | | 0.15 | 19 | 15 | 19 | 14.5 | 0.441 | 16 | 19 | 14.5 | 0.126 | 268 | 25.1 | 46 | 28.4 | 0.401 | 64 | 28.8 | 0.857 | 351 | 69.6 |
| | | 0.05 | 16 | 10 | 16 | 14.1 | 0.578 | 10 | 16 | 13.9 | 0.172 | 251 | 22.8 | 47 | 27.6 | 0.535 | 56 | 27.7 | 0.388 | 348 | 65.7 |
| 2 | 2 | 0.1 | 35 | 29 | 35 | 17.6 | 0.458 | 18 | 35 | 17.5 | 0.062 | 310 | 33.2 | 75 | 34.9 | 0.487 | 69 | 34.4 | 0.444 | 359 | 87.5 |
| | | 0.15 | 5 | 2 | 5 | 12.5 | 0.724 | 5 | 5 | 12.4 | 0.014 | 180 | 15.8 | 29 | 24.4 | 0.655 | 48 | 24.8 | 0.224 | 342 | 50.9 |
| | | 0.05 | 27 | 19 | 27 | 15.7 | 0.240 | 15 | 27 | 15.7 | 0.026 | 300 | 29.1 | 71 | 31.3 | 0.442 | 57 | 30.8 | 0.249 | 358 | 78.7 |
| | 3 | 0.1 | 54 | 48 | 54 | 22.0 | 0.236 | 12 | 54 | 21.8 | 0.093 | 317 | 41.1 | 81 | 42.5 | 0.506 | 85 | 42.4 | 0.307 | 361 | 103.9 |
| | | 0.15 | 48 | 42 | 48 | 21.2 | 0.070 | 14 | 48 | 20.2 | 0.129 | 309 | 37.7 | 80 | 41.0 | 0.182 | 70 | 39.0 | 0.392 | 355 | 96.5 |
| | | 0.05 | 13 | 5 | 13 | 13.6 | 0.756 | 12 | 13 | 13.6 | 0.021 | 177 | 17.1 | 31 | 26.5 | 0.682 | 63 | 27.3 | 0.152 | 339 | 55.8 |
| | 1 | 0.1 | 20 | 11 | 20 | 15.5 | 0.697 | 19 | 20 | 14.5 | 0.283 | 194 | 20.0 | 39 | 29.5 | 0.567 | 64 | 28.8 | 0.561 | 342 | 62.0 |
| | | 0.15 | 22 | 12 | 22 | 15.2 | 0.706 | 21 | 22 | 15.1 | 0.058 | 195 | 20.3 | 40 | 29.3 | 0.620 | 67 | 30.2 | 0.207 | 344 | 63.7 |
| | | 0.05 | 5 | 2 | 5 | 12.6 | 0.782 | 5 | 5 | 12.4 | 0.011 | 165 | 15.1 | 28 | 24.4 | 0.667 | 45 | 24.7 | 0.205 | 338 | 50.1 |
| 3 | 2 | 0.1 | 5 | 2 | 5 | 12.5 | 0.781 | 5 | 5 | 12.4 | 0.147 | 165 | 15.1 | 28 | 24.4 | 0.697 | 51 | 24.9 | 0.197 | 338 | 50.0 |
| | | 0.15 | 7 | 2 | 7 | 12.8 | 0.776 | 7 | 7 | 12.7 | 0.019 | 168 | 15.4 | 25 | 24.9 | 0.599 | 48 | 25.2 | 0.246 | 340 | 51.3 |
| | | 0.05 | 383 | 2 | 383 | 249.7 | 0.000 | 1 | 383 | 247.8 | 0.001 | 385 | 244.0 | 384 | 488.7 | 0.000 | 383 | 485.8 | 0.005 | 385 | 480.0 |
| | 3 | 0.1 | 13 | 4 | 13 | 13.6 | 0.763 | 13 | 13 | 13.7 | 0.040 | 174 | 16.8 | 36 | 26.4 | 0.574 | 60 | 27.3 | 0.194 | 338 | 55.3 |
| | | 0.15 | 19 | 6 | 19 | 14.8 | 0.694 | 19 | 19 | 14.5 | 0.111 | 199 | 19.8 | 39 | 28.6 | 0.597 | 64 | 28.8 | 0.267 | 340 | 61.8 |
| | | 0.05 | 186 | 2 | 186 | 79.0 | 0.445 | 76 | 186 | 79.9 | 0.047 | 296 | 90.9 | 205 | 154.3 | 0.306 | 203 | 155.1 | 0.163 | 358 | 214.2 |
| | 1 | 0.1 | 4 | 2 | 4 | 12.5 | 0.784 | 4 | 4 | 12.6 | 0.911 | 165 | 15.0 | 27 | 24.3 | 0.676 | 23 | 24.5 | 0.712 | 338 | 49.4 |
| | | 0.15 | 7 | 2 | 7 | 12.8 | 0.779 | 7 | 7 | 12.8 | 0.644 | 167 | 15.4 | 25 | 24.9 | 0.694 | 38 | 25.0 | 0.571 | 338 | 51.1 |
| | | 0.05 | 255 | 2 | 255 | 127.3 | 0.219 | 64 | 255 | 127.9 | 0.018 | 336 | 137.4 | 277 | 251.4 | 0.155 | 261 | 249.1 | 0.043 | 366 | 297.7 |
| 4 | 2 | 0.1 | 197 | 3 | 197 | 86.1 | 0.445 | 103 | 197 | 86.9 | 0.116 | 300 | 97.4 | 213 | 168.1 | 0.381 | 209 | 169.1 | 0.171 | 363 | 228.5 |
| | | 0.15 | 3 | 2 | 3 | 12.5 | 0.779 | 3 | 3 | 12.4 | 0.779 | 165 | 14.9 | 21 | 24.2 | 0.699 | 27 | 24.2 | 0.647 | 338 | 48.9 |
| | | 0.05 | 187 | 2 | 187 | 79.7 | 0.427 | 88 | 187 | 80.6 | 0.027 | 293 | 89.1 | 199 | 155.4 | 0.277 | 196 | 156.3 | 0.076 | 360 | 212.1 |
| | 3 | 0.1 | 226 | 3 | 226 | 105.5 | 0.353 | 79 | 226 | 107.6 | 0.056 | 315 | 114.9 | 240 | 205.7 | 0.263 | 235 | 208.1 | 0.088 | 371 | 261.3 |
| | | 0.15 | 293 | 2 | 293 | 158.5 | 0.024 | 35 | 293 | 164.7 | 0.173 | 362 | 169.2 | 299 | 315.9 | 0.116 | 301 | 316.1 | 0.256 | 384 | 356.4 |
| | | | | **t = [15, 20)** | | | | | | | | | | **t = [20, 25)** | | | | | | | |
| | | 0.05 | 42 | 33 | 231 | 71.7 | 0.170 | 17 | 232 | 73.7 | 0.355 | 359 | 154.5 | 302 | 123.9 | 0.150 | 316 | 133.1 | 0.117 | 362 | 214.0 |
| | 1 | 0.1 | 23 | 20 | 229 | 58.9 | 0.324 | 16 | 230 | 69.4 | 0.201 | 359 | 127.9 | 312 | 104.8 | 0.182 | 318 | 115.3 | 0.116 | 362 | 178.4 |
| | | 0.15 | 19 | 15 | 236 | 55.2 | 0.374 | 16 | 227 | 56.3 | 0.867 | 353 | 116.6 | 305 | 98.9 | 1.029 | 314 | 100.4 | 0.584 | 356 | 164.2 |
| | | 0.05 | 16 | 10 | 242 | 55.1 | 0.145 | 10 | 238 | 55.3 | 0.280 | 351 | 111.2 | 301 | 97.6 | 0.098 | 311 | 98.5 | 0.173 | 353 | 157.0 |
| 2 | 2 | 0.1 | 35 | 29 | 230 | 67.9 | 0.354 | 18 | 214 | 64.9 | 0.527 | 359 | 143.1 | 305 | 117.1 | 0.286 | 317 | 114.6 | 0.313 | 362 | 199.6 |
| | | 0.15 | 5 | 2 | 227 | 46.7 | 0.203 | 5 | 229 | 49.4 | 0.193 | 348 | 90.1 | 301 | 83.8 | 0.162 | 308 | 87.3 | 0.102 | 350 | 129.9 |
| | | 0.05 | 27 | 19 | 251 | 63.8 | 0.471 | 15 | 224 | 58.6 | 0.344 | 360 | 130.0 | 305 | 111.3 | 0.280 | 313 | 104.7 | 0.324 | 362 | 182.2 |
| | 3 | 0.1 | 54 | 48 | 211 | 78.1 | 0.392 | 12 | 239 | 80.5 | 0.237 | 362 | 168.6 | 298 | 131.4 | 0.346 | 330 | 139.4 | 0.132 | 364 | 233.9 |
| | | 0.15 | 48 | 42 | 217 | 75.5 | 0.168 | 14 | 214 | 71.3 | 0.443 | 357 | 160.2 | 293 | 127.5 | 0.118 | 315 | 125.5 | 0.305 | 359 | 221.3 |
| | | 0.05 | 13 | 5 | 222 | 49.6 | 0.264 | 12 | 219 | 53.8 | 0.147 | 345 | 98.8 | 303 | 90.3 | 0.187 | 307 | 94.7 | 0.091 | 348 | 142.5 |
| | 1 | 0.1 | 20 | 11 | 222 | 54.6 | 0.348 | 19 | 224 | 57.1 | 0.535 | 346 | 108.3 | 297 | 98.0 | 0.291 | 309 | 100.4 | 0.446 | 349 | 155.5 |
| | | 0.15 | 22 | 12 | 225 | 55.0 | 0.753 | 21 | 222 | 58.8 | 0.176 | 347 | 111.1 | 300 | 99.4 | 0.908 | 303 | 103.0 | 0.110 | 351 | 159.5 |
| | | 0.05 | 5 | 2 | 233 | 46.8 | 0.166 | 5 | 248 | 49.2 | 0.174 | 344 | 89.2 | 301 | 84.6 | 0.193 | 309 | 88.1 | 0.087 | 347 | 128.9 |
| 3 | 2 | 0.1 | 5 | 2 | 230 | 46.8 | 0.358 | 5 | 231 | 49.4 | 0.216 | 344 | 89.0 | 300 | 84.5 | 0.774 | 297 | 86.9 | 0.255 | 347 | 128.8 |
| | | 0.15 | 7 | 2 | 217 | 46.3 | 0.995 | 7 | 228 | 49.8 | 0.217 | 345 | 91.5 | 305 | 84.8 | 0.402 | 319 | 89.4 | 0.161 | 348 | 132.3 |
| | | 0.05 | 383 | 2 | 384 | 728.7 | 0.000 | 1 | 387 | 726.9 | 0.004 | 385 | 714.6 | 385 | 968.8 | 0.000 | 388 | 968.8 | 0.004 | 385 | 950.4 |
| | 3 | 0.1 | 13 | 4 | 234 | 51.0 | 0.309 | 13 | 236 | 55.0 | 0.202 | 344 | 98.2 | 301 | 91.9 | 0.493 | 310 | 96.3 | 0.135 | 347 | 141.8 |
| | | 0.15 | 19 | 6 | 235 | 54.2 | 0.272 | 19 | 217 | 56.6 | 0.264 | 345 | 107.6 | 313 | 98.9 | 0.135 | 301 | 99.4 | 0.144 | 348 | 154.4 |
| | | 0.05 | 186 | 2 | 319 | 257.8 | 0.121 | 76 | 269 | 245.3 | 0.181 | 361 | 343.5 | 337 | 382.2 | 0.043 | 329 | 361.1 | 0.128 | 364 | 472.5 |
| | 1 | 0.1 | 4 | 2 | 231 | 46.6 | 0.272 | 4 | 211 | 45.3 | 0.307 | 344 | 88.0 | 300 | 83.8 | 0.195 | 301 | 82.0 | 0.043 | 347 | 127.2 |
| | | 0.15 | 7 | 2 | 217 | 46.3 | 0.307 | 7 | 229 | 49.6 | 0.257 | 344 | 93.4 | 304 | 84.7 | 0.192 | 295 | 87.4 | 0.156 | 347 | 134.2 |
| | | 0.05 | 255 | 2 | 344 | 401.4 | 0.059 | 64 | 309 | 382.2 | 0.039 | 367 | 460.5 | 352 | 562.2 | 0.019 | 341 | 533.9 | 0.032 | 368 | 624.7 |
| 4 | 2 | 0.1 | 197 | 3 | 324 | 275.1 | 0.697 | 103 | 260 | 264.5 | 0.158 | 365 | 362.6 | 346 | 406.3 | 0.336 | 324 | 381.1 | 0.134 | 366 | 497.2 |
| | | 0.15 | 3 | 2 | 215 | 45.0 | 0.251 | 3 | 225 | 45.8 | 0.273 | 344 | 87.1 | 303 | 81.6 | 0.179 | 297 | 82.3 | 0.197 | 347 | 126.1 |
| | | 0.05 | 187 | 2 | 315 | 253.7 | 0.482 | 88 | 267 | 248.7 | 0.068 | 362 | 340.1 | 352 | 382.7 | 0.261 | 319 | 361.2 | 0.038 | 363 | 468.2 |
| | 3 | 0.1 | 226 | 3 | 324 | 325.9 | 0.104 | 79 | 290 | 321.7 | 0.082 | 373 | 413.8 | 357 | 472.8 | 0.052 | 339 | 460.3 | 0.063 | 373 | 564.5 |
| | | 0.15 | 293 | 2 | 341 | 480.2 | 0.122 | 35 | 333 | 478.9 | 0.180 | 384 | 545.1 | 377 | 663.8 | 0.053 | 357 | 656.8 | 0.045 | 385 | 733.7 |

Table 2: Results of MOP$_4$SA for further timesteps with $t \geq 25$

| | | | | DBSCAN | | | | Spectral Clustering | | | | Exact | | DBSCAN | | | Spectral Clustering | | | Exact | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| $d$ | $\tau$ | $\delta_{\leq}$ | $n$ | $n_{>1}$ | $\#_{gr}$ | $s$ | $D_{KL}$ | $n_{>1}$ | $\#_{gr}$ | $s$ | $D_{KL}$ | $\#_{gr}$ | $s$ | $\#_{gr}$ | $s$ | $D_{KL}$ | $\#_{gr}$ | $s$ | $D_{KL}$ | $\#_{gr}$ | $s$ |
| | | | | **t = [25, 30)** | | | | | | | | | | **t = [30, 35)** | | | | | | | |
| | 1 | 0.05 | 42 | 33 | 328 | 182.7 | 0.371 | 17 | 338 | 193.2 | 0.078 | 363 | 274.0 | 334 | 248.7 | 0.356 | 345 | 255.4 | 0.073 | 366 | 334.9 |
| | | 0.1 | 23 | 20 | 328 | 155.2 | 0.102 | 16 | 340 | 167.2 | 0.093 | 362 | 229.2 | 338 | 207.6 | 0.088 | 341 | 220.7 | 0.094 | 364 | 280.7 |
| | | 0.15 | 19 | 15 | 318 | 146.5 | 0.988 | 16 | 332 | 150.4 | 0.064 | 356 | 212.3 | 322 | 195.5 | 0.442 | 338 | 201.1 | 0.052 | 358 | 261.9 |
| 2 | 2 | 0.05 | 16 | 10 | 318 | 143.7 | 0.068 | 10 | 333 | 146.2 | 0.128 | 355 | 203.5 | 322 | 191.5 | 0.052 | 340 | 195.7 | 0.110 | 358 | 250.6 |
| | | 0.1 | 35 | 29 | 324 | 172.1 | 0.105 | 18 | 337 | 171.8 | 0.076 | 362 | 256.8 | 334 | 229.4 | 0.076 | 341 | 230.3 | 0.064 | 364 | 314.5 |
| | | 0.15 | 5 | 2 | 312 | 124.5 | 0.127 | 5 | 320 | 128.8 | 0.070 | 352 | 170.1 | 314 | 167.1 | 0.136 | 325 | 171.7 | 0.068 | 355 | 210.8 |
| | 3 | 0.05 | 27 | 19 | 322 | 162.2 | 0.324 | 15 | 338 | 157.2 | 0.292 | 362 | 234.5 | 328 | 215.1 | 0.086 | 348 | 212.4 | 0.178 | 365 | 287.2 |
| | | 0.1 | 54 | 48 | 331 | 194.4 | 0.233 | 12 | 348 | 205.7 | 0.070 | 364 | 299.1 | 342 | 261.1 | 0.119 | 352 | 275.6 | 0.065 | 366 | 365.4 |
| | | 0.15 | 48 | 42 | 327 | 187.5 | 0.083 | 14 | 341 | 188.6 | 0.124 | 359 | 282.9 | 336 | 250.5 | 0.072 | 349 | 253.1 | 0.064 | 362 | 349.6 |
| | 1 | 0.05 | 13 | 5 | 313 | 134.9 | 0.143 | 12 | 323 | 140.1 | 0.064 | 349 | 187.2 | 317 | 180.7 | 0.134 | 329 | 187.2 | 0.063 | 352 | 232.2 |
| | | 0.1 | 20 | 11 | 313 | 145.7 | 0.494 | 19 | 332 | 149.9 | 0.169 | 351 | 203.2 | 320 | 194.9 | 0.440 | 338 | 200.9 | 0.097 | 354 | 251.7 |
| | | 0.15 | 22 | 12 | 314 | 148.0 | 0.656 | 21 | 326 | 152.9 | 0.079 | 352 | 208.4 | 319 | 198.6 | 0.189 | 330 | 204.3 | 0.077 | 355 | 258.0 |
| 3 | 2 | 0.05 | 5 | 2 | 311 | 125.5 | 0.149 | 5 | 320 | 131.4 | 0.071 | 349 | 169.9 | 313 | 167.4 | 0.145 | 325 | 174.4 | 0.066 | 352 | 211.0 |
| | | 0.1 | 5 | 2 | 310 | 125.4 | 0.631 | 5 | 313 | 127.8 | 0.145 | 349 | 169.0 | 312 | 167.1 | 0.590 | 322 | 170.2 | 0.129 | 352 | 210.0 |
| | | 0.15 | 7 | 2 | 311 | 126.6 | 0.142 | 7 | 335 | 133.6 | 0.112 | 350 | 173.6 | 316 | 169.5 | 0.122 | 340 | 179.1 | 0.092 | 353 | 215.5 |
| | 3 | 0.05 | 383 | 2 | 385 | 1214.0 | 0.000 | 1 | 388 | 1210.5 | 0.004 | 385 | 1186.4 | 385 | 1458.9 | 0.000 | 388 | 1454.6 | 0.004 | 385 | 1422.8 |
| | | 0.1 | 13 | 4 | 309 | 136.1 | 0.523 | 13 | 322 | 142.5 | 0.090 | 349 | 186.0 | 312 | 181.3 | 0.492 | 327 | 189.5 | 0.089 | 352 | 231.0 |
| | | 0.15 | 19 | 6 | 320 | 147.0 | 0.108 | 19 | 328 | 164.4 | 0.089 | 350 | 201.5 | 322 | 196.2 | 0.109 | 337 | 226.8 | 0.077 | 353 | 249.3 |
| | 1 | 0.05 | 186 | 2 | 341 | 508.9 | 0.037 | 76 | 360 | 490.5 | 0.108 | 364 | 602.1 | 342 | 635.8 | 0.036 | 372 | 627.3 | 0.042 | 367 | 732.7 |
| | | 0.1 | 4 | 2 | 310 | 124.1 | 0.148 | 4 | 311 | 122.5 | 0.503 | 349 | 167.1 | 312 | 165.6 | 0.139 | 317 | 164.3 | 0.524 | 352 | 207.6 |
| | | 0.15 | 7 | 2 | 312 | 126.6 | 0.143 | 7 | 313 | 129.2 | 0.105 | 349 | 176.1 | 317 | 169.8 | 0.135 | 319 | 172.4 | 0.075 | 352 | 218.2 |
| 4 | 2 | 0.05 | 255 | 2 | 355 | 725.2 | 0.015 | 64 | 361 | 698.5 | 0.025 | 369 | 789.8 | 356 | 889.8 | 0.016 | 368 | 873.8 | 0.021 | 370 | 956.1 |
| | | 0.1 | 197 | 3 | 352 | 540.4 | 0.313 | 103 | 363 | 516.2 | 0.115 | 367 | 632.7 | 353 | 676.1 | 0.303 | 368 | 658.3 | 0.102 | 369 | 772.0 |
| | | 0.15 | 3 | 2 | 312 | 122.2 | 0.134 | 3 | 309 | 122.6 | 0.156 | 349 | 165.4 | 317 | 163.8 | 0.131 | 311 | 163.8 | 0.152 | 352 | 205.5 |
| | 3 | 0.05 | 187 | 2 | 354 | 513.2 | 0.252 | 88 | 346 | 487.1 | 0.025 | 363 | 597.6 | 355 | 652.6 | 0.254 | 358 | 619.3 | 0.021 | 365 | 734.7 |
| | | 0.1 | 226 | 3 | 360 | 623.2 | 0.050 | 79 | 361 | 611.6 | 0.030 | 373 | 715.9 | 362 | 775.4 | 0.047 | 368 | 769.2 | 0.028 | 375 | 868.0 |
| | | 0.15 | 293 | 2 | 384 | 857.6 | 0.019 | 35 | 369 | 842.5 | 0.038 | 385 | 924.6 | 385 | 1054.2 | 0.013 | 374 | 1031.6 | 0.032 | 385 | 1115.7 |
| | | | | **t = [35, 40)** | | | | | | | | | | **t = [40, 45)** | | | | | | | |
| | 1 | 0.05 | 42 | 33 | 340 | 314.0 | 0.341 | 17 | 350 | 318.9 | 0.072 | 369 | 396.8 | 346 | 378.5 | 0.330 | 353 | 383.4 | 0.070 | 371 | 459.1 |
| | | 0.1 | 23 | 20 | 346 | 261.6 | 0.078 | 16 | 342 | 274.7 | 0.095 | 367 | 333.0 | 349 | 316.7 | 0.074 | 349 | 329.3 | 0.093 | 370 | 385.8 |
| | | 0.15 | 19 | 15 | 329 | 247.2 | 0.066 | 16 | 339 | 252.7 | 0.050 | 361 | 311.4 | 335 | 298.3 | 0.065 | 345 | 305.2 | 0.048 | 363 | 361.3 |
| 2 | 2 | 0.05 | 16 | 10 | 327 | 240.0 | 0.054 | 10 | 343 | 246.1 | 0.104 | 361 | 298.6 | 333 | 289.6 | 0.052 | 350 | 303.8 | 0.109 | 363 | 347.6 |
| | | 0.1 | 35 | 29 | 338 | 288.0 | 0.070 | 18 | 345 | 290.0 | 0.063 | 367 | 373.8 | 343 | 347.7 | 0.069 | 348 | 350.8 | 0.064 | 369 | 432.8 |
| | | 0.15 | 5 | 2 | 320 | 209.7 | 0.136 | 5 | 330 | 215.4 | 0.068 | 358 | 252.4 | 330 | 253.1 | 0.124 | 338 | 260.1 | 0.068 | 360 | 294.5 |
| | 3 | 0.05 | 27 | 19 | 334 | 269.6 | 0.081 | 15 | 355 | 269.0 | 0.087 | 367 | 341.2 | 340 | 324.8 | 0.078 | 359 | 326.6 | 0.082 | 370 | 395.3 |
| | | 0.1 | 54 | 48 | 347 | 329.5 | 0.107 | 12 | 353 | 344.8 | 0.064 | 369 | 438.8 | 352 | 398.8 | 0.100 | 360 | 415.8 | 0.063 | 373 | 506.7 |
| | | 0.15 | 48 | 42 | 346 | 316.1 | 0.055 | 14 | 349 | 319.3 | 0.061 | 365 | 414.6 | 353 | 383.1 | 0.052 | 352 | 385.6 | 0.063 | 368 | 479.4 |
| | 1 | 0.05 | 13 | 5 | 325 | 227.6 | 0.131 | 12 | 333 | 241.8 | 0.062 | 355 | 277.9 | 331 | 275.4 | 0.121 | 341 | 290.8 | 0.062 | 357 | 324.1 |
| | | 0.1 | 20 | 11 | 325 | 245.1 | 0.416 | 19 | 343 | 253.5 | 0.066 | 357 | 300.9 | 333 | 296.5 | 0.386 | 347 | 306.7 | 0.065 | 359 | 350.9 |
| | | 0.15 | 22 | 12 | 325 | 249.6 | 0.093 | 21 | 336 | 256.7 | 0.074 | 358 | 308.3 | 335 | 302.0 | 0.060 | 346 | 310.2 | 0.072 | 360 | 359.3 |
| 3 | 2 | 0.05 | 5 | 2 | 318 | 210.5 | 0.140 | 5 | 331 | 218.4 | 0.064 | 355 | 253.1 | 327 | 254.1 | 0.131 | 340 | 263.3 | 0.062 | 357 | 295.6 |
| | | 0.1 | 5 | 2 | 318 | 209.7 | 0.568 | 5 | 326 | 213.6 | 0.132 | 355 | 251.9 | 328 | 253.4 | 0.527 | 332 | 257.9 | 0.134 | 357 | 298.2 |
| | | 0.15 | 7 | 2 | 322 | 213.3 | 0.113 | 7 | 344 | 227.6 | 0.094 | 356 | 259.6 | 331 | 258.3 | 0.108 | 347 | 286.1 | 0.095 | 358 | 302.9 |
| | 3 | 0.05 | 383 | 2 | 385 | 1701.7 | 0.000 | 1 | 389 | 1699.4 | 0.003 | 385 | 1659.9 | 385 | 1944.8 | 0.000 | 390 | 1943.8 | 0.003 | 385 | 1898.4 |
| | | 0.1 | 13 | 4 | 317 | 227.3 | 0.472 | 13 | 332 | 237.5 | 0.091 | 355 | 276.8 | 327 | 274.4 | 0.443 | 342 | 286.3 | 0.095 | 357 | 323.4 |
| | | 0.15 | 19 | 6 | 328 | 246.2 | 0.111 | 19 | 341 | 285.6 | 0.077 | 356 | 299.6 | 335 | 298.8 | 0.101 | 349 | 338.4 | 0.075 | 358 | 349.2 |
| | 1 | 0.05 | 186 | 2 | 345 | 764.2 | 0.031 | 76 | 374 | 766.0 | 0.023 | 367 | 864.1 | 347 | 893.2 | 0.031 | 376 | 913.4 | 0.022 | 369 | 1003.2 |
| | | 0.1 | 4 | 2 | 317 | 207.7 | 0.134 | 4 | 323 | 207.2 | 0.519 | 355 | 248.8 | 326 | 250.8 | 0.125 | 332 | 251.0 | 0.465 | 357 | 290.7 |
| | | 0.15 | 7 | 2 | 323 | 213.8 | 0.134 | 7 | 323 | 216.7 | 0.076 | 355 | 261.9 | 331 | 259.1 | 0.123 | 330 | 261.7 | 0.149 | 357 | 305.5 |
| 4 | 2 | 0.05 | 255 | 2 | 356 | 1055.1 | 0.016 | 64 | 372 | 1045.3 | 0.021 | 370 | 1124.5 | 359 | 1221.2 | 0.015 | 376 | 1218.6 | 0.021 | 371 | 1291.4 |
| | | 0.1 | 197 | 3 | 355 | 812.6 | 0.295 | 103 | 371 | 804.4 | 0.098 | 370 | 914.6 | 358 | 951.2 | 0.273 | 373 | 948.6 | 0.097 | 372 | 1067.9 |
| | | 0.15 | 3 | 2 | 323 | 206.2 | 0.129 | 3 | 317 | 205.7 | 0.147 | 355 | 246.3 | 332 | 249.8 | 0.117 | 327 | 248.9 | 0.137 | 357 | 287.7 |
| | 3 | 0.05 | 187 | 2 | 356 | 785.1 | 0.245 | 88 | 360 | 753.9 | 0.020 | 368 | 865.9 | 358 | 917.9 | 0.238 | 366 | 889.9 | 0.019 | 368 | 998.0 |
| | | 0.1 | 226 | 3 | 364 | 929.7 | 0.046 | 79 | 373 | 929.2 | 0.028 | 377 | 1021.8 | 366 | 1084.0 | 0.045 | 376 | 1090.0 | 0.027 | 378 | 1175.8 |
| | | 0.15 | 293 | 2 | 385 | 1251.4 | 0.012 | 35 | 377 | 1223.1 | 0.030 | 385 | 1304.6 | 385 | 1447.7 | 0.012 | 379 | 1416.0 | 0.028 | 385 | 1494.3 |

Table 3: Distances as a result of running MOP$_4$SCD between consecutive timesteps

| $d$ | $\tau$ | $\delta_{\leq}$ | $d(\mathcal{W}^t, \mathcal{W}^{t+1}, \mathcal{S})$ **5** | **6** | **7** | **8** | **9** | **10** | **11** | **12** | **13** | **14** | **15** | **16** | **17** | **18** | **19** | **20** | **21** | **22** | **23** |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 2 | 1 | 0.05 | .067 | .067 | .068 | .065 | .066 | .068 | .067 | .066 | .063 | .065 | .069 | .068 | .062 | .061 | .064 | .069 | .064 | .067 | .068 |
| | | 0.1 | .102 | .112 | .116 | .107 | .114 | .115 | .109 | .114 | .111 | .105 | .101 | .11 | .111 | .116 | .11 | .109 | .114 | .107 | .113 |
| | | 0.15 | .135 | .132 | .142 | .125 | .136 | .142 | .142 | .138 | .14 | .135 | .137 | .133 | .139 | .132 | .129 | .135 | .131 | .134 | .143 |
| | 2 | 0.05 | .105 | .113 | .114 | .116 | .113 | .115 | .113 | .114 | .115 | .115 | .116 | .113 | .117 | .114 | .115 | .112 | .113 | .113 | .116 |
| | | 0.1 | .068 | .068 | .069 | .073 | .068 | .074 | .061 | .082 | .071 | .071 | .077 | .08 | .073 | .065 | .075 | .072 | .082 | .074 | .081 |
| | | 0.15 | .177 | .18 | .18 | .185 | .18 | .184 | .178 | .185 | .18 | .182 | .184 | .181 | .181 | .181 | .178 | .184 | .185 | .184 | .188 |
| | 3 | 0.05 | .079 | .082 | .081 | .082 | .086 | .092 | .087 | .08 | .09 | .083 | .079 | .089 | .08 | .085 | .084 | .089 | .086 | .082 | .088 |
| | | 0.1 | .037 | .038 | .037 | .039 | .039 | .039 | .037 | .035 | .038 | .039 | .034 | .039 | .038 | .037 | .036 | .038 | .038 | .038 | .038 |
| | | 0.15 | .044 | .043 | .043 | .046 | .046 | .044 | .043 | .045 | .044 | .045 | .044 | .046 | .048 | .046 | .044 | .044 | .046 | .046 | .05 |
| 3 | 1 | 0.05 | .199 | .201 | .2 | .2 | .199 | .201 | .2 | .201 | .201 | .201 | .201 | .19 | .201 | .2 | .201 | .2 | .2 | .2 | .201 |
| | | 0.1 | .214 | .219 | .219 | .218 | .219 | .218 | .217 | .216 | .216 | .218 | .218 | .217 | .217 | .217 | .219 | .217 | .218 | .217 | .219 |
| | | 0.15 | .213 | .218 | .216 | .217 | .218 | .219 | .218 | .219 | .217 | .218 | .218 | .218 | .218 | .217 | .218 | .219 | .218 | .218 | .217 |
| | 2 | 0.05 | .238 | .239 | .239 | .24 | .24 | .24 | .239 | .24 | .24 | .24 | .24 | .241 | .239 | .239 | .238 | .239 | .24 | .239 | .24 |
| | | 0.1 | .237 | .239 | .238 | .239 | .239 | .239 | .239 | .24 | .239 | .24 | .239 | .239 | .239 | .238 | .239 | .239 | .24 | .239 | .24 |
| | | 0.15 | .236 | .237 | .238 | .238 | .239 | .238 | .239 | .238 | .238 | .239 | .238 | .238 | .237 | .237 | .238 | .238 | .238 | .238 | .238 |
| | 3 | 0.05 | .001 | .001 | .0 | .001 | .001 | .001 | .001 | .001 | .001 | .001 | .001 | .001 | .001 | .001 | .001 | .001 | .001 | .001 | .001 |
| | | 0.1 | .232 | .231 | .23 | .233 | .232 | .233 | .235 | .233 | .234 | .233 | .233 | .232 | .233 | .233 | .233 | .235 | .234 | .233 | .234 |
| | | 0.15 | .223 | .226 | .223 | .228 | .227 | .227 | .226 | .227 | .23 | .227 | .227 | .227 | .229 | .228 | .228 | .229 | .226 | .227 | .228 |
| 4 | 1 | 0.05 | .056 | .055 | .056 | .055 | .056 | .056 | .056 | .055 | .055 | .056 | .056 | .056 | .056 | .056 | .056 | .056 | .056 | .056 | .056 |
| | | 0.1 | .176 | .178 | .178 | .178 | .178 | .178 | .178 | .176 | .178 | .178 | .178 | .178 | .178 | .178 | .178 | .178 | .178 | .178 | .178 |
| | | 0.15 | .211 | .21 | .213 | .213 | .213 | .213 | .213 | .213 | .212 | .212 | .213 | .213 | .213 | .213 | .213 | .213 | .213 | .213 | .213 |
| | 2 | 0.05 | .033 | .034 | .035 | .034 | .035 | .034 | .035 | .035 | .035 | .035 | .035 | .035 | .035 | .035 | .035 | .035 | .035 | .035 | .035 |
| | | 0.1 | .075 | .076 | .076 | .076 | .076 | .076 | .076 | .075 | .075 | .076 | .076 | .076 | .075 | .076 | .076 | .075 | .076 | .075 | .076 |
| | | 0.15 | .218 | .216 | .218 | .218 | .218 | .218 | .218 | .218 | .218 | .218 | .218 | .218 | .218 | .218 | .218 | .218 | .218 | .218 | .218 |
| | 3 | 0.05 | .064 | .064 | .064 | .065 | .065 | .064 | .065 | .065 | .065 | .065 | .065 | .065 | .065 | .065 | .065 | .064 | .065 | .065 | .065 |
| | | 0.1 | .049 | .05 | .049 | .05 | .05 | .05 | .05 | .05 | .05 | .05 | .05 | .05 | .05 | .05 | .05 | .049 | .05 | .05 | .05 |
| | | 0.15 | .016 | .015 | .015 | .016 | .016 | .016 | .016 | .016 | .016 | .016 | .016 | .016 | .016 | .016 | .016 | .016 | .016 | .016 | .016 |

| $d$ | $\tau$ | $\delta_{\leq}$ | $d(\mathcal{W}^t, \mathcal{W}^{t+1}, \mathcal{S})$ **24** | **25** | **26** | **27** | **28** | **29** | **30** | **31** | **32** | **33** | **34** | **35** | **36** | **37** | **38** | **39** | **40** | **41** | **42** |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 2 | 1 | 0.05 | .066 | .067 | .064 | .064 | .064 | .068 | .067 | .064 | .063 | .065 | .066 | .067 | .062 | .068 | .066 | .067 | .066 | .067 | .067 |
| | | 0.1 | .116 | .11 | .103 | .103 | .115 | .115 | .108 | .11 | .107 | .115 | .109 | .107 | .114 | .117 | .107 | .116 | .112 | .115 | .114 |
| | | 0.15 | .135 | .135 | .139 | .137 | .139 | .136 | .132 | .143 | .133 | .143 | .132 | .134 | .129 | .137 | .137 | .142 | .13 | .141 | .14 |
| | 2 | 0.05 | .111 | .115 | .113 | .114 | .111 | .11 | .11 | .112 | .108 | .112 | .11 | .111 | .112 | .112 | .115 | .114 | .11 | .113 | .109 |
| | | 0.1 | .078 | .079 | .073 | .077 | .077 | .067 | .074 | .08 | .068 | .072 | .065 | .076 | .072 | .074 | .076 | .073 | .068 | .073 | .067 |
| | | 0.15 | .184 | .185 | .182 | .182 | .182 | .182 | .178 | .183 | .172 | .184 | .175 | .179 | .183 | .177 | .184 | .186 | .176 | .187 | .18 |
| | 3 | 0.05 | .075 | .087 | .08 | .086 | .09 | .085 | .091 | .086 | .089 | .083 | .091 | .076 | .086 | .088 | .082 | .091 | .087 | .088 | .092 |
| | | 0.1 | .039 | .039 | .036 | .036 | .039 | .037 | .039 | .038 | .037 | .036 | .036 | .037 | .036 | .038 | .035 | .038 | .037 | .035 | .039 |
| | | 0.15 | .046 | .044 | .045 | .045 | .048 | .048 | .048 | .046 | .045 | .043 | .047 | .042 | .045 | .049 | .048 | .048 | .049 | .044 | .048 |
| 3 | 1 | 0.05 | .201 | .2 | .2 | .2 | .201 | .198 | .199 | .2 | .198 | .201 | .191 | .2 | .19 | .2 | .199 | .2 | .2 | .2 | .198 |
| | | 0.1 | .218 | .218 | .217 | .217 | .217 | .218 | .217 | .217 | .216 | .217 | .218 | .217 | .218 | .217 | .216 | .217 | .215 | .219 | .217 |
| | | 0.15 | .218 | .218 | .218 | .218 | .219 | .217 | .217 | .218 | .217 | .216 | .216 | .217 | .219 | .217 | .218 | .216 | .218 | .219 | .216 |
| | 2 | 0.05 | .24 | .239 | .24 | .239 | .239 | .239 | .238 | .239 | .237 | .239 | .239 | .237 | .24 | .238 | .238 | .24 | .239 | .24 | .239 |
| | | 0.1 | .239 | .239 | .239 | .239 | .239 | .239 | .238 | .239 | .238 | .237 | .238 | .238 | .24 | .237 | .238 | .239 | .239 | .239 | .239 |
| | | 0.15 | .238 | .237 | .238 | .237 | .237 | .238 | .236 | .238 | .236 | .237 | .237 | .237 | .238 | .236 | .237 | .238 | .236 | .238 | .236 |
| | 3 | 0.05 | .001 | .001 | .001 | .001 | .001 | .001 | .001 | .001 | .001 | .001 | .001 | .001 | .001 | .001 | .001 | .001 | .001 | .001 | .001 |
| | | 0.1 | .233 | .234 | .232 | .235 | .233 | .232 | .233 | .231 | .233 | .233 | .233 | .231 | .232 | .232 | .234 | .235 | .233 | .232 | .235 |
| | | 0.15 | .229 | .229 | .228 | .228 | .228 | .227 | .228 | .227 | .226 | .228 | .226 | .227 | .226 | .225 | .228 | .228 | .228 | .228 | .226 |
| 4 | 1 | 0.05 | .056 | .055 | .056 | .056 | .055 | .055 | .055 | .055 | .055 | .055 | .054 | .055 | .056 | .056 | .056 | .056 | .056 | .056 | .056 |
| | | 0.1 | .178 | .178 | .178 | .178 | .178 | .178 | .178 | .178 | .178 | .178 | .178 | .178 | .178 | .178 | .178 | .178 | .178 | .178 | .178 |
| | | 0.15 | .213 | .213 | .213 | .213 | .213 | .213 | .213 | .213 | .213 | .213 | .213 | .213 | .213 | .213 | .213 | .213 | .213 | .213 | .213 |
| | 2 | 0.05 | .035 | .035 | .035 | .034 | .035 | .034 | .035 | .035 | .035 | .034 | .035 | .034 | .035 | .034 | .035 | .035 | .035 | .035 | .035 |
| | | 0.1 | .075 | .076 | .076 | .076 | .076 | .076 | .076 | .075 | .076 | .074 | .076 | .075 | .076 | .076 | .076 | .076 | .076 | .076 | .075 |
| | | 0.15 | .218 | .218 | .218 | .218 | .218 | .216 | .218 | .218 | .218 | .218 | .218 | .218 | .216 | .218 | .218 | .218 | .218 | .218 | .218 |
| | 3 | 0.05 | .065 | .065 | .063 | .065 | .065 | .065 | .065 | .065 | .065 | .065 | .065 | .065 | .064 | .064 | .065 | .065 | .064 | .065 | .065 |
| | | 0.1 | .05 | .05 | .05 | .05 | .05 | .05 | .05 | .05 | .05 | .05 | .05 | .05 | .05 | .05 | .05 | .05 | .05 | .05 | .05 |
| | | 0.15 | .016 | .016 | .016 | .016 | .016 | .016 | .016 | .016 | .016 | .016 | .016 | .016 | .016 | .016 | .016 | .016 | .016 | .016 | .016 |