

A Generalized Framework for Mining Spatio-temporal Patterns in Scientific Data

Hui Yang
Dept. Comp. Sci. & Eng.
The Ohio State University
Columbus, Ohio, 43210

Srinivasan Parthasarathy
Dept. Comp. Sci. & Eng.
The Ohio State University
Columbus, Ohio, 43210

Sameep Mehta
Dept. Comp. Sci. & Eng.
The Ohio State University
Columbus, Ohio, 43210

ABSTRACT

In this paper, we present a general framework to discover spatial associations and spatio-temporal episodes for scientific datasets. In contrast to previous work in this area, features are modeled as geometric objects rather than points. We define multiple distance metrics that take into account objects' extent and thus are more robust in capturing the influence of an object on other objects in spatial neighborhood. We have developed algorithms to discover four different types of spatial object interaction (association) patterns. We also extend our approach to accommodate temporal information and propose a simple algorithm to derive spatio-temporal episodes. We show that such episodes can be used to reason about critical events. We evaluate our framework on real datasets to demonstrate its efficacy. The datasets originate from two different areas: Computational Molecular Dynamics and Computational Fluid Flow. We present results highlighting the importance of the identified patterns and episodes by using knowledge from the underlying domains. We also show that the proposed algorithms scale linearly with respect to the dataset size.

Categories and Subject Descriptors

H.2.8 [Database Applications]—Data Mining, Scientific Databases, Spatial Databases and GIS

General Terms

Algorithms, Experimentation

Keywords

Spatial Object Association, Spatio-temporal Association/Episode, Scientific Data

1. INTRODUCTION

Analyzing spatial data is an important problem in many application domains, including geographical information systems, bioinformatics, scientific and engineering informatics, and computer aided design. The main difference between analyzing such data and data

in a transactional form is that an object can influence the properties of objects located in the same spatial neighborhood [19] and therefore must be modeled in the analysis. Analyzing and reasoning about relationships among spatial objects is further complicated if the data is time varying in nature. Data produced from protein folding or fluid dynamics simulations, or geoinformatics datasets that track the behavior of intrusions are examples that have this additional constraint. Mining spatial relationships in these datasets is an important and interesting problem, since specific relationships may be indicators or predictors of upcoming events (e.g., vortex amalgamation and dissipation in fluid flows [18]).

Unfortunately, mining relationships among spatial objects is an extremely challenging task. First, almost all the related work done to date on this problem models features as single points [12, 26]. However, especially in physical sciences, the extent and shape of a feature can play an important role in determining its influence on neighboring objects [10, 18]. Second, there is a strong need to develop techniques to capture and reason about the interactions among features. These interactions if properly captured can help domain experts to understand the underlying processes in an effective manner. Third, one needs to develop effective techniques to incorporate temporal information in the overall analysis and reason about them. Finally, recent technological advances in computational sciences have resulted in huge amounts of data. Traditional statistical approaches to model such interactions do not scale very well to large datasets [5]. In this paper we propose a general-purpose framework to address these challenges.

To address the first challenge, a straightforward solution is to represent a feature by its Minimum Bounding Box (MBB). However, MBBs are not ideal for all situations. For example, vortices are well represented by ellipsoids [16] but not by MBBs. Alternatively, defect structures in materials [10] require irregular shape descriptors such as landmarks [14]. Our proposed framework addresses this issue by supporting flexible schemes to model and represent features of interest (see Fig. 1 for examples in 2D).

To address the second challenge, we examine the use of distance metrics that take features' shape and extent into account. Simply measuring the distance between centroids is often inadequate. We propose and evaluate the use of three distance metrics and demonstrate their suitability on different end applications. Additionally, we believe it is important to effectively model different types of interactions that capture various spatial (distance-based, topological and directional) constraints. Our framework currently supports four types of spatial object association patterns (SOAPs), *Star*, *Clique*, *Sequence*, and *minLink*, which are used to model different interactions. For example, *Clique* SOAPs in Molecular Dynamics data indicate the presence of compact defects. Compact defects are usually more stable and often govern the properties of materials. *Se-*

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.

KDD'05, August 21–24, 2005, Chicago, Illinois, USA.

Copyright 2005 ACM 1-59593-135-X/05/0008 ...\$5.00.

quence SOAPs on the other hand can indicate the formation of long extended defects from smaller point defects [15].

To address the third challenge, we have designed a simple approach that accommodates temporal information to derive spatio-temporal patterns in data produced by such scientific datasets. We define three types of events to describe SOAPs' evolutionary behavior: *formation*, *dissipation*, and *continuation*. Such events characterize the stability of different interactions. For each identified SOAP, we further construct its *spatio-temporal episodes*, marked by pairs of formation and dissipation events. We then use these episodes to infer critical events such as vortex amalgamation. We also demonstrate that, by combining episodes associated with multiple SOAP types, we are able to model the evolutionary behavior of interactions among features.

To address the final challenge, we integrate effective optimization techniques and leverage novel data structures to make the SOAP mining algorithms efficient [23]. We validate the utility and demonstrate the scalability (can easily process up to several GBs of data in minutes) of our framework on two applications drawn from the scientific and engineering community, namely, Computational Fluid Dynamics (CFD) and Computational Molecular Dynamics (CMD).

2. BASIC CONCEPTS

Spatial Feature Representation We propose three different representation schemes: parallelepiped (or parallelogram in 2D), ellipsoid (or ellipse in 2D), and landmarks based representation, where landmarks are sampled boundary points [14]. These schemes can be used to model features from a variety of scientific domains. Parallelepipeds (or parallelograms) subsume MBBs, thus are applicable to model features in relatively regular shape. Whereas ellipsoids or ellipses are appropriate for vortices. Finally, landmarks are effective to model highly irregular-shaped features such as defect structures in materials. The number of landmarks needed to represent a feature is domain dependent. The framework also supports elemental shapes such as lines and splines.

As shown in Figs. 1 (a) and (b), the shape descriptor of a parallelogram or an ellipse can be described as $\mathbf{A}_{\text{basic}} = \langle (x; y); l_1; l_2; \dots \rangle$. In cases where landmarks are used (Fig. 1c), the shape descriptor is $\mathbf{A}_{\text{landmark}} = \langle (x_i; y_i); 1 \dots i \rangle$, where $(x_i; y_i)$ is the i^{th} landmark. These descriptors can be easily extended to 3D.

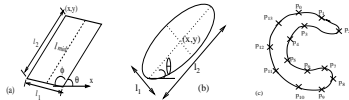


Figure 1: Shapes (a)Parallelogram (b)Ellipse (c)Irregular

Dataset Representation The dataset \mathbf{D} consists of n features extracted from $r > 1$ snapshots, taken at time steps t_1, \dots, t_r ($t_1 < \dots < t_r$). The n features are further categorized into l types, where the categorization is governed by the underlying domain. Each feature type is assigned with a unique label in $\mathbf{f} = \langle f_1; \dots; f_l \rangle$. A feature's geometric properties are captured by one of the three representation schemes described earlier. Thus a feature \mathbf{f} at time t_i can be described as a vector $\mathbf{f} = \langle t_i, loc, \mathbf{A}_{\text{geo}}, type \rangle$, where $type \in \{1, 2\}$, loc is \mathbf{f} 's position at t_i , and $\mathbf{A}_{\text{geo}} = 2\mathbf{f}\mathbf{A}_{\text{basic}}, \mathbf{A}_{\text{landmark}} \mathbf{g}$ models the geometric properties of \mathbf{f} at t_i .

Note that in the rest of the paper, we refer to a feature corresponding to the above vector as a *spatial object*. We also assume without loss of generality the existence of an ordering among the l feature types: $c_1 < c_2 < \dots < c_l$. Furthermore, we refer to a snapshot's associated time t_i as its ID.

Shape-based Distance Measurements The framework supports three distance measurements for two objects \mathbf{o}_i and \mathbf{o}_j in the same snapshot: (1) *Point-Point*: This is simply the Euclidian distance

between object centroids; (2) *Line-Line*: If \mathbf{o}_i and \mathbf{o}_j are parallelepipeds (or parallelograms), we first identify the line segment between the midpoints of the upper and lower surfaces (or sides) in each object, then compute the shortest distance between these two line segments. If \mathbf{o}_i and \mathbf{o}_j are ellipsoids (or ellipses), the line-line distance is the shortest distance between the two ellipses' major axes; and (3) *Boundary-Boundary*: This is the shortest pairwise distance between the landmarks of \mathbf{o}_i and \mathbf{o}_j . Notice that the last two metrics take objects' geometric properties into account. The framework also supports Hausdorff distance [1]. Since this distance is not applicable to the applications described in this article, we do not discuss it here.

Two objects \mathbf{o}_i and \mathbf{o}_j have a *closeTo* relationship if the distance between them is $\leq \epsilon$, a user-specified parameter. Two objects are *neighbors* if they have a *closeTo* relationship. We also define the *isAbove* relationship, a special type of directional relationship, between two objects. In a coordinate system, \mathbf{o}_i is said to have an *isAbove* relationship with \mathbf{o}_j , if the upper-left corner of \mathbf{o}_i 's MBB¹, denoted as $(x_i; y_i; z_i)$, and the upper left corner of \mathbf{o}_j 's MBB, denoted as $(x_j; y_j; z_j)$, meets the following condition: $(z_i > z_j) _ [(z_i = z_j) \wedge ((y_i > y_j) _ ((y_i = y_j) \wedge (x_i < x_j)))]$ in 3D, or $(y_i > y_j) _ [(y_i = y_j) \wedge (x_i < x_j)]$ in 2D.

Spatial Object Association Pattern (SOAP) A SOAP of size k , denoted as k -SOAP, characterizes the *closeTo* or *isAbove* relationships among k object types. The framework supports the discovery of four SOAP types: *Star*, *Clique*, *Sequence*, and *minLink* (Fig. 2). These SOAP types can be abstracted as undirected graphs. In such graphs, a node corresponds to an object-type c_i , and an edge $(c_i; c_j)$ indicates a *closeTo* or *isAbove* relationship between c_i and c_j . These SOAP types can also be represented as lists subject to certain constraints. We next describe each SOAP type in detail:

Star SOAPS (Fig. 2a) have a *center* object-type, which is required to have a *closeTo* relationship with all the other object-types in the same SOAP. Let c_{cntr} be the center of a *Star* k -SOAP \mathbf{p} , and $\mathbf{f}_{c[i]}: i \in [1, k-1]$ be the other $k-1$ object-types in \mathbf{p} , where $c[i]$ denotes the i^{th} element in \mathbf{p} , and $c[1] \dots c[k-1]$. The SOAP \mathbf{p} can then be represented as a list $\mathbf{p} = (c_{\text{cntr}}, c[1]; \dots; c[k-1]): \delta_{i \in [1, k-1]} \text{closeTo}(c_{\text{cntr}}, c[i])$

Clique SOAPS (Fig. 2b) require that a *closeTo* relationship holds between every pair of involved object-types in a SOAP. Let $\mathbf{f}_{c[i]}: i \in [1, k]$ be the k object-types in a *Clique* SOAP, it can then be described by the following list: $(c[1]; \dots; c[k]): \delta_{i, j \in [2, k]} \text{closeTo}(c[i], c[j])$

Sequence SOAPS (Fig. 2c) of size k , $\mathbf{p} = (c[i]: i \in [1, k])$, satisfy two constraints: (a) $\text{closeTo}(c[i], c[i+1]) = \text{true}$ and (b) $\text{isAbove}(c[i], c[i+1]) = \text{true}$, where $1 \leq i \leq k-1$

minLink SOAPS (Fig. 2d) are a parameterized SOAP type, where the value of *minLink* is user-specified. Let $\text{minLink} = l$, *minLink* SOAPS include all the SOAPS that have *linkage* $\geq l$. The *linkage* of a k -SOAP \mathbf{p} is defined as follows. Let n_i ($1 \leq i \leq k$) be the number of neighbors that the i^{th} object-type has in \mathbf{p} , the *linkage* of \mathbf{p} is then defined as $\min_{i \in [1, k]} n_i$. A (*minLink*= l) k -SOAP can be represented by the following list: $(c[1]; \dots; c[k]): \delta_{i \in [2, k]} \#[\text{Neighbors Of } c[i]] \geq l$, where $(c[1] \dots c[k])$. Note that the set of *minLink*= l SOAPS subsumes all the other three SOAP types.

The above SOAP types capture three basic spatial relationships: distance-based, topological, and directional. The *closeTo* relationship is both distance and topology based. As for directional relationships, we currently only consider the *isAbove* relationship, captured by *Sequence* SOAPS. Such relationships are important

¹In 3D, it is defined as the upper-left corner of its top surface.

in understanding different interacting behaviors in many scientific applications. We plan to implement other types of spatial relationships and examine their uses for different applications in the future.

A SOAP is said to be *autocorrelated* if an object-type occurs multiple times. For example, $(c_1; c_1; c_2)$ is an autocorrelated 3-SOAP as the object-type c_1 occurs twice. An *instance* of a SOAP p is the set of spatial objects that meet all the requirements specified by p . For instance, (o_i, o_j) is an instance of the *Clique* 2-SOAP (c_1, c_2) , if $o_i.type=c_1$, $o_j.type=c_2$, and $closeTo(o_i, o_j)=true$.

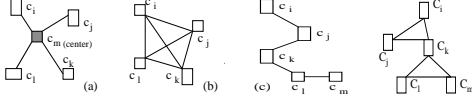


Figure 2: SOAP Types: (a)Star (b)Clique (c)Sequence (d)minLink=2

We define two measures, *support* and *realization*, to characterize the importance of a SOAP. The support of a SOAP p is the number of snapshots in the dataset where p occurs. Assume $support(p)=s$, let n_i be the number of p 's instances in the i^{th} snapshot where p appears, $realization(p)=\min n_i/g$. A pattern p is *frequent* if $support(p) \geq minSupp$, and *prevalent* if $realization(p) \geq minRealization$. Both $minSupp$ and $minRealization$ are user-specified parameters.

Spatio-temporal Episodes Spatial relationships among objects (or features) evolve over time. As a result, SOAPs of different types also evolve over time. We identify the following three evolutionary events for a SOAP p : (1) *Formation*: when the number of p 's instances changes from zero to non-zero; (2) *Dissipation*: when all p 's instances become invalid. The dissipation of a SOAP can occur due to many reasons. For example feature(s) involved in a SOAP may cease to exist or merge into a new one; and (3) *Continuation from time t_i to time t_{i+1}* : if there exists at least one instance of p in each snapshot taken in $[t_i, t_{i+1}]$. These events essentially characterize the stability of interactions among different features. They are useful to model and subsequently predict features' future spatio-temporal behaviors.

Formation and dissipation events can occur to a SOAP many times. Thus a SOAP can exist in multiple disjoint temporal intervals, where each interval starts at a formation event and ends at a dissipation event. We refer to a SOAP's continuation in each of such intervals as a *spatio-temporal episode*. Let I_p^t be the set of SOAP p 's instances at time t , an episode of p in the *discrete* interval $[t_s; t_e]$ can then be described as: $p[t_s; t_e] = \bigcup_{t=t_s}^{t_e} I_p^t$.

3. RELATED WORK

Spatial object association patterns proposed in this work share some commonalities with collocation patterns, which identify features that are frequently located in the same neighborhood. However, most work on collocation pattern mining is limited to applications where subjects of interest can be represented as points [12, 13, 26]. Furthermore, such points are located in a single map, which corresponds to a snapshot in our work. Recently, Xiong et al. proposed an approach to discover collocation patterns for extended spatial objects including line-strings and polygons [20]. For each object, they identify its neighborhood and then consider objects that have overlapped neighborhood as candidate collocation patterns. This is similar to the *Clique* SOAPs in this article. However, they only consider objects located in the same map.

The SOAP mining problem also shares some similarity with frequent subgraph mining [21]. In order to apply the conventional graph mining algorithms to discover SOAPs, the notion of nodes needs to be extended to integrate spatial properties such as location and shape. The notion of edges also needs to be modified to reflect different spatial relationships such as *closeTo*. Furthermore, it is not

straightforward to integrate topological or directional relationships into the graph mining process. Finally, *minLink* 2 SOAPs allow us to discover rigid sub-structures. This is generally not supported by graph mining algorithms.

Our research on spatio-temporal association patterns shares some of the objectives with approaches for spatial reasoning. Bailey-kellogg and Zhao [2] propose a methodology for reasoning about such problems called qualitative spatial reasoning (QSR). Their work is methodology driven and mainly focuses on conceptual topics such as data representations and manipulations. They also discuss the use of different spatial primitives to model objects of different shapes and spatial relationships among objects. Our framework is an efficient realization of their conceptual methodology for scientific data. In addition we also support the discovery of spatio-temporal episodes that is not explicitly considered in their work. Fernyhough et al. implemented techniques to detect events by identifying frequently occurring spatial relationships [4, 6]. However, their proposed technique only considers pair-wise relationships. Thus interactions involving more than two features will be missed.

4. FRAMEWORK AND ALGORITHMS

An overview of the framework is given in Fig. 3. This framework is mainly motivated by the following three scientific applications: defects simulation in Molecular Dynamics, protein structure analysis in Bioinformatics, and vortex simulation in Computational Fluid Dynamics. We have implemented techniques to detect, extract, and classify features for the first two applications [10, 22]. For the third application, we use existing algorithms to detect vortices and subsequently classify the detected vortices [8]. In this article, we focus on the four tasks enclosed by the dashed rectangle in the figure.

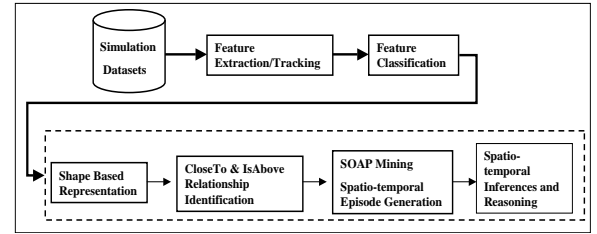


Figure 3: Overview of the framework

We organize the dataset \mathbf{D} in the following manner. The n objects are first grouped into l partitions, where each partition is composed of objects of the same type. Within each partition, objects are first ordered by the time they appeared, then by their locations at a certain time. This data organization is analogous to the vertical format used for association rule mining [25].

Equivalence Classes Based on the list-based SOAP representation, we organize SOAPs as equivalence classes. A *k-equivalence class*, denoted by *k-EquiClass*, is defined as the set of k -SOAPs that (1) are of the same SOAP type; and (2) have the same prefix, where the *prefix* of a k -SOAP consists of its first $k-1$ elements. For instance, *star* SOAPs $(c_1; c_2; c_3)$, $(c_1; c_2; c_4)$, and $(c_1; c_2; c_5)$ will be organized into the same 3-EquiClass, with $(c_1; c_2)$ being the prefix. By using equivalence classes, our mining algorithms only need to compute the *closeTo* or *isAbove* relationships among objects once (when generating 2-SOAPs). The equivalence classes also help to improve locality while generating SOAPs [25]. As a result, our algorithms can efficiently discover different types of SOAPs from large amounts of data.

Mining Star and Clique SOAPs The first step is to generate frequent 1-SOAPs. For each object-type c_i , the procedure counts the number of snapshots that contain at least one object of type c_i . If the count $\geq minSupp$, then (c_i) is a frequent 1-SOAP. The set

of all frequent 1-SOAPs is denoted by P_1 . The next step discovers 2-SOAPs and organizes them into 2-EquiClasses.

A 2-EquiClass is generated for each frequent 1-SOAP. The 2-EquiClass of 1-SOAP $(c_1)2P_1$, denoted by $E_{2;c_1}$, contains frequent 2-SOAPs in the form of $(c_1; c_j)$. To generate $E_{2;c_1}$, the procedure considers the following 2-SOAPs as candidates: $(c_1; c_j)$, where $(c_j)2P_1$. The condition $c_j > c_1$ needs to hold in the case of mining *Clique* SOAPs. For each candidate 2-SOAP $p_2=(c_1; c_j)$, the procedure identifies all the snapshots where p_2 occurs. An instance of $p_2=(c_1; c_j)$ is an object pair $(o_1; o_j)$, where $(o_1; \text{type}=c_1) \wedge (o_j; \text{type}=c_j) \wedge (\text{distance}(o_1; o_j, \text{distType}) < \text{minSupp})$. If p_2 occurs in *minSupp* snapshots, it is frequent and added to $E_{2;c_1}$. The algorithm next discovers SOAPs of size 3. Two k -SOAPs in the same k -EquiClass are combined to construct a candidate $(k+1)$ -SOAP. For each candidate $(k+1)$ -SOAP p_{k+1} derived by appending the last element of $p_{k;j}$ to $p_{k;i}$, the algorithm identifies all the snapshots where p_{k+1} occurs. $p_{k;i}$ denotes the i^{th} k -SOAP in a k -EquiClass. To compute the instances of p_{k+1} in the snapshot t_i , we combine each pair of instances from $p_{k;j}$ to $p_{k;i}$. An instance pair can be joined to produce a p_{k+1} instance if they have the same first $k-1$ objects and different last objects. For *Clique* SOAPs, an additional condition must also hold: there is a *closeTo* relationship between the two instances' last objects. SOAPs with the same prefix are organized into one equivalence class as being generated. The process stops when it has found all the frequent SOAPs.

The above procedure discovers SOAPs that are frequent and have realization 1. We use a simple example to explain why we need to keep all the SOAPs that have realization 1. Let $f_{a_1; b_1; b_2 g}$ be the only objects in a snapshot and they are neighbors. Assume $\text{minRealization}=2$, in order to derive the two instances $(a_1; b_1)$ and $(a_1; b_2)$ of the 2-SOAP $(a; b)$, the 1-SOAP (a) must be maintained even if its realization is 1 in the snapshot. Therefore, if $\text{minRealization} > 1$, an additional step is needed to identify the SOAPs that are frequent and have realization > 1 .

Mining Sequence SOAPs Unlike mining *star* or *clique* SOAPs, which uses two k -SOAPs in the same equivalence class to generate a candidate $(k+1)$ -SOAP ($k \geq 2$), the algorithm joins one k -SOAP and one 2-SOAP. The first two steps discover all the frequent 1-SOAPs and 2-SOAPs respectively. These two steps are the same as that of mining *Star* or *Clique* SOAPs except that a valid *Sequence* 2-SOAP instance also needs to meet the *isAbove* relationship other than being neighbors. For each k -SOAP $p_k=(c_{11}; \dots; c_{1k})$, the algorithm first locates the 2-EquiClass $E_{2;c_{1k}}$ in which every 2-SOAP is in the form $(c_{1k}; c_{1j})$: $\text{closeTo}(c_{1k}; c_{1j}) \wedge \text{isAbove}(c_{1k}; c_{1j})$. A set of candidate $(k+1)$ -SOAPs are then generated by combining p_k with each 2-SOAP in $E_{2;c_{1k}}$. A candidate $(k+1)$ -SOAP p_{k+1} is frequent if it appears in *minSupp* snapshots. Instances of p_{k+1} in the snapshot t_i are generated by combining each pair of instances from $p_{k;i}$ and $p_{2;j}$. An instance pair, $I_{k;i}$ and $I_{2;j}$ from $p_{k;i}$ and $p_{2;j}$ respectively, can be combined if the last object in $I_{k;i}$ is the same as the first object in $I_{2;j}$. For the same reason explained before, if $\text{minRealization} > 1$, an additional step is required to mark SOAPs being both prevalent and frequent. The algorithm stops when no more SOAPs can be discovered.

Mining minLink SOAPs The algorithm starts by identifying all frequent and prevalent 1-SOAPs and 2-SOAPs, which are the same as mining *Clique* SOAPs. To generate a $(k+1)$ -SOAP, the algorithm takes one k -SOAP and one 1-SOAP. For each k -SOAP p_k , the algorithm first collects the set of features (1-SOAPs) that have a *closeTo* relationship with at least one feature in p_k . These features can be easily identified from 2-EquiClasses. Let C be the set of all such 1-SOAPs. A candidate $(k+1)$ -SOAP p_{k+1} is then generated by appending a feature $c_i \in C$ to p_k . However, the same p_{k+1}

can be generated multiple times. For example, consider the following 2-SOAPs $p_1:(c_1; c_2)$ and $p_2:(c_1; c_3)$, p_1 can be joined with c_3 to form a 3-SOAP $(c_1; c_2; c_3)$. Similarly c_2 can be joined with p_2 to obtain the same 3-SOAP. To avoid such redundant computation, the algorithm checks if a candidate SOAP has been encountered before processing it. It proceeds to process the candidate SOAP if it has not been processed. Similarly, instances of a candidate $(k+1)$ -SOAP p_{k+1} , obtained from appending a feature c_i to a k -SOAP p_k , are derived by combining each pair of instances from p_k and c_i . An instance I_k of p_k can be combined with an object of type c_i if they meet the following conditions: (1) o_i is not in I_k ; (2) o_i has a *closeTo* relationship with at least one object in I_k ; and (3) the *linkage* of the new instance is *minLink*.

Note that when $\text{minLink} > 1$, to discover all the frequent $(k+1)$ -SOAPs, the algorithm needs to identify all the frequent k -SOAPs with *linkage* 1. This is because the *linkage* of SOAPs is not anti-monotone. For instance, the *linkage* of the *Clique* SOAP $(a; b)$ is 1. However, the *linkage* of the *Clique* SOAP $(a; b; c)$ is 2. Thus, when $\text{minLink} > 1$, the algorithm needs to mark SOAPs that have enough *linkage*. For the same reason stated before, if $\text{minRealization} > 1$, an extra step is needed to label SOAPs being prevalent and frequent.

We also implement effective optimization techniques so that the mining algorithms are capable to handle large out-of-core datasets. Please refer to [23] for details.

Analyzing Spatio-temporal Episodes For each frequent SOAP, we construct its spatio-temporal episodes by identifying the associated formation and dissipation events. We then use these episodes to address two important issues: (1) to reason about critical events such as the merging of multiple features; and (2) to model how interactions among a certain set of features evolve over time.

The first issue is addressed by analyzing individual episodes. Let $p[t_s^i; t_e^i]$ be an episode of SOAP p . For each instance of p appearing in $[t_s^i; t_e^i]$, we compute the size of the instance's minimum bounding box (MBB), where the MBB of a SOAP instance encompasses every object in the instance and is minimum in size. We then make inferences on critical events by observing the changing trend of the area of an instance's MBB. For instance, if the MBB of an instance decreases from t_s^i to t_e^i , it is possible that the involved objects will merge at its corresponding SOAP's dissipation.

Post-Trans.	Star	Clique	Sequence
Pre-Trans.			
Star	X	Increasing Interaction → Object amalgamation	Decreasing interaction between the center object and some objects
Clique	Decreasing Interaction	X	Decreasing interaction among some objects
Sequence	Formation of a center object	Increasing Interaction → Object amalgamation	X

Figure 4: SOAP transitions

The second issue is addressed by two steps. Let $F=f_{c_1; \dots; c_1 g}$ be the set of features of interest. We first find all the episodes that are associated with F . These episodes can be of different SOAP types. We then order all these episodes by their formation time. The resulting sequence of SOAP episodes is then used to model the interactions among features in F . As mentioned earlier, different SOAP types characterize different types of interactions. Therefore, a transition between two different SOAP types (e.g., from *Star* to *Clique*) indicates a significant change having occurred to an interaction. Such transitions can be easily identified from the episode sequence described above. Furthermore, we can reason about critical events based on these transitions [23]. Fig. 4 summarizes the transitions between the three SOAP types and their implications, which are encoded in our reasoning algorithms.

5. EXPERIMENTAL EVALUATION

In this section we evaluate our framework on datasets from two different scientific domains: Computational Molecular Dynamics and Computational Fluid Dynamics. We also evaluate the performance and scalability of the framework on large datasets. Extensive experiments were also conducted on datasets drawn from the bioinformatics domain. Due to space constraint, we do not present the results here. Please refer to [23] for details.

5.1 Case Study 1: Molecular Dynamics Simulation Datasets

The Molecular Dynamics (MD) dataset is generated using the Object-oriented High-performance Multi-scale Multi-resolution Simulator for material science (OHMMS) [15]. A slice of Silicon (Si) lattice is simulated to understand the creation of stable structures and dimer rows. The dataset consists of 4000 Silicon slices, corresponding to 4000 snapshots. There are 94 atoms in each slice.

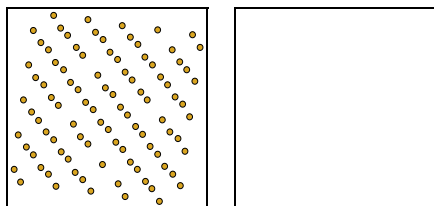


Figure 5: (a) Si surface at t_0 (b) Si surface at t_{3999}

Creation of stable structures Figs. 5(a) and (b) show the slice of Si lattice at $t=0$ and at $t=3999$ respectively. At $t=0$, there are no stable structures in the lattice. However, stable structures are formed at $t=3999$ and can be easily spotted from Fig. 5(b). These structures corresponds to the pentagon-like rings formed by five Si atoms. Our framework can discover these structures automatically by mining $minLink=1$ SOAPs from the dataset. Two atoms are considered as neighbors if the distance between them is 2.7\AA .

Creation of dimer rows A dimer is defined as a pair of con-

Figure 6: (a) Dimers on Si surface at t_{2000} (b) Dimers on Si surface at t_{2500} (c) Dimer on Si surface at t_{3999}

nected atoms with their bond length falls within a threshold (2.7\AA in our evaluation). In order to be considered as a dimer, two atoms should also be oriented at an angle of around 45 degrees. A *dimer row* is created when individual dimers align themselves in a particular fashion. For example, Figure 6(c) shows a detected dimer row (enclosed in the rectangle). The figure also shows other dimer rows at different evolutionary stages. Dimer rows were discovered by mining sequence SOAPs. We consider two dimers as neighbors if their distance is 5\AA .

SOAP episodes Fig. 6 illustrates a SOAP episode identified in the dataset. Fig. 6(b) shows two 2-SOAPs (boxed in the figure) that have no interaction. However, after a few time steps a new dimer is created between them. This new dimer acts as a link between the two 2-SOAPs, which leads to the formation of a SOAP of 5 dimers 6(c). Such an episode involves two SOAP events: the amalgamation of two 2-SOAPs and the creation of a 5-SOAP.

5.2 Case Study 2: Vortex Simulation Datasets

The vortex dataset is generated by implementing a simplistic version of the algorithm proposed by Christian [3]. The dataset consists of 1970 snapshots, with around 200 vortices in each snapshot. The number of vortices in each frame is changing over time, as an existing vortex can dissipate, new vortex can be created, or two vortices can merge to form a new vortex.

Fig. 7 shows three simulation snapshots at different times. Fig. 7 (a) shows the initial configuration of the vortices with no SOAPs. Fig. 7 (b) shows the creation of a SOAP. This SOAP is formed because the two vortices are moving towards each other and eventually their distance falls in the distance threshold. Fig. 7(c) also shows a very interesting result. The two vortices involved in the SOAP came closer and eventually merged into a new vortex. This event is captured by the SOAP’s dissipation (resulting from an amalgamation of two features). We were also able to make inference on such events by identifying the changing trend of the corresponding instance’s MBB. Note that SOAP dissipation does not necessarily imply dissipation of features.

Figure 7: (a) Vortices at t_0 (b) SOAP Formation at t_{90} (c) Vortex Merging at t_{104}

Spatio-temporal episode evaluation As mentioned earlier, combining episodes associated with different SOAP types can be used to model the evolving nature of interactions among different features. For the same set of features, the formation of different SOAP types at different time indicates a potential change in their interacting behavior, since different types of SOAPs characterize different types of interactions.

Figure 8: Evolving SOAP: (7 8 96 99) starts as a Clique (t_0), evolves into a Star (t_{20}), eventually out of interaction range (SOAP dissipation) (t_{28})

Fig. 8 demonstrates how the interactions among four vortices evolve over time and are captured by combining multiple episodes formed at different time. The four vortices form a *clique* SOAP at t_0 . This *clique* SOAP continues for 11 time steps and then evolves into a *star* SOAP at time t_{12} , which continues for another 15 time steps (Fig. 8(b)). A *clique* SOAP points to the presence of compact spatial patterns, where each feature is interacting with every other feature. The transition of a *clique* SOAP to *Star* implies that features are moving away from each other and may cease to interact after some time. Fig. 8(c) demonstrates exactly this behavior.

Other transitions such as from *star* to *clique* are also identified for different sets of features. In the case that a SOAP evolves from *Star* to *Clique*, we observe that features are moving towards each other and thereby increasing the level of interactions. This may eventually lead to the merging of multiple features.

5.3 Performance Evaluation

In this section we present results on scalability of our framework. All the experiments were carried out on a Pentium 1.7GHz machine of 1GB memory. We applied the SOAP mining algorithms to a large Molecular Dynamics dataset produced by OHMMS. The dataset consists of 387,999 slices of Si lattice and is of size 1.5GB. We first identify all the dimers in each slice and then discover different SOAP types in the dataset.

Fig. 9 shows the time taken to discover all types of SOAPS at different support thresholds for the 1.5GB dataset. First of all, even for a large dataset we are able to mine all SOAPS in reasonable amount of time. Time increases as support threshold decreases, because more SOAPS will be generated at a lower support value. However, this increase in time is linear. For example, at 20% support we take 180 seconds to discover all the *star* SOAPS, the time just increases by 50 seconds for a very low support threshold of 0.5%. Figure 9 shows the performance results on the 1.5GB dataset. Please note that *minLink* SOAP subsumes all the other SOAP types. As a result, the algorithm can compute all the other SOAP types within roughly the same time (negligible overhead) it takes to compute *minLink* patterns.

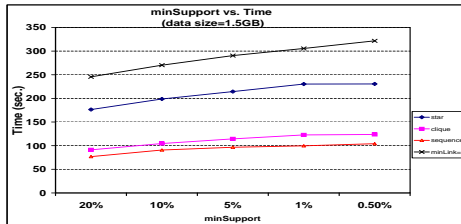


Figure 9: Running Time on 1.5GB Dataset (minDist=5Å)

5.4 Impact of Distance Metrics

In order to capture different interactions among evolving objects, it is critical to use an appropriate distance metric. In this section we present results demonstrating the importance of using distance metrics that are capable of taking objects' shape and extent into account. We compare the number of SOAPS generated from using an object-based distance metric and that from the commonly used point-based metric. Due to space constraint, we only present results on the vortex dataset, where each vortex is represented as an ellipse.

As mentioned earlier, the ellipse-based shape representation and boundary-boundary distance make the most sense in this case. To justify this, we compare the numbers of SOAPS discovered by using B-B distance and C-C distance. As shown in Table 1, we are able to get twice as many SOAPS based on Boundary-Boundary distance compared to Centroid-Centroid distance. Results on MD datasets were very similar in spirit, however due to lack of space we are not presenting the results.

Type	Star	Clique	Seq.	Total
C-C	586	169	2166	2921
B-B	1315	329	6240	7885

Table 1: #SOAPS Discovered in Vortex Data

6. CONCLUSION

In this paper, we present a general framework for mining spatial associations and spatio-temporal episodes for scientific datasets. Features are modeled as geometric objects rather than points. We define multiple distance metrics that take into account objects' shape and extent and thus are more robust in capturing the influence of an object on other objects in its spatial neighborhood. We have developed algorithms to discover four different types of spatial object association patterns across multiple maps. We also extend our ap-

proach to mine for spatial temporal episodes and thereby present a methodology for reasoning about critical events.

Empirical results on three real case study applications, drawn from the scientific and engineering disciplines serve to validate the framework. We show that the discovered interactions are meaningful and can be used to uncover important spatial and spatio-temporal patterns in the underlying scientific domain. We further demonstrate that the different association pattern types our framework when used in conjunction can provide an effective mechanism to support qualitative spatio-temporal reasoning. Performance studies carried out on large out-of-core datasets indicate that the framework is both efficient and scalable.

We are currently extending the framework to include other types of association patterns, for example, patterns concerning both topological and neighborhood relationships. We are also implementing new approaches towards more robust spatio-temporal inferences and reasoning. Our framework also currently targets the discovery of important frequent spatial interactions, but in many cases rare interactions can also be important. We plan to extend our work to address this limitation.

Acknowledgments: We thank J. Wilkins and T. Lenosky for providing and helping validate the MD results, and thank R. Machiraju, D. Thompson, K. Marsolo and D. Polshakov for valuable comments, discussion and validation of results pertaining to CFD and bioinformatics.

7. REFERENCES

- [1] M. J. Atallah. A linear time algorithm for the hausdorff distance between convex polygons. *Information Processing Letters*, 17:207–209, 1983.
- [2] C. Bailey-Kellogg and F. Zhao. Qualitative spatial reasoning: extracting and reasoning with spatial aggregates. *AI Mag.*, 24(4):47–60, 2004.
- [3] J. Christian. Numerical simulation of hydrodynamics by the method of point vortices. *Journal of Computational Physics*, 135, pages 189–197, 1971.
- [4] A. Cohn and S.M. Hazarika. Qualitative spatial representation and reasoning: an overview. *Fundam. Inf.*, 46(1-2):1–29, 2001.
- [5] N. Cressie. *Spatial Statistics*. John Wiley and Sons, 1991.
- [6] J. Fernyhough et al. Event recognition using qualitative reasoning on automatically generated spatio-temporal models from visual input. In *IJCAI97 Workshop on Spatial and Temporal Reasoning*.
- [7] C. Henze. Feature detection in linked derived spaces. In *IEEE VIS*, 1998.
- [8] M. Jiang et al. Feature mining paradigms for scientific data. In *SDM*, 2003.
- [9] H. Mannila and H. Toivonen. Discovering generalised episodes using minimal occurrences. In *SIGKDD*, 1996.
- [10] S. Mehta et al. Dynamic classification of defect structures in molecular dynamics simulation data. In *SDM*, 2005.
- [11] S. Mehta et al. Detection and visualization of anomalous structures in molecular dynamics simulation data. In *IEEE VIS*, 2004.
- [12] Y. Morimoto. Mining frequent neighboring class sets in spatial databases. In *SIGKDD*, 2001.
- [13] R. Munro et al. Complex spatial relationships. In *ICDM*, 2003.
- [14] C. R. Rao and S. Suryawanshi. Statistical analysis of shape of objects based on landmark data. *Proc Natl Acad Sci USA*, 93(22):12132–12136, 1996.
- [15] D.A. Richie et al. Real-time multiresolution analysis for accelerated molecular dynamics simulations. In *American Phys. Soc. Mar. Mtng.*, 2001.
- [16] A. Sadarjoen et al. Selective visualization of vortices in hydrodynamic flows. In *IEEE VIS*, 1998.
- [17] S. Shekhar and Y. Huang. Discovering spatial co-location patterns: A summary of results. *Lecture Notes in Computer Science*, 2001.
- [18] D. Silver and X. Wang. Volume tracking. *IEEE VIS*, 1996.
- [19] W. R. Tobler. A computer movie simulating urban growth in the detroit region. *Economic Geography* 46:234-230, 1970.
- [20] H. Xiong et al. A framework for discovering co-location patterns in data sets with extended spatial objects. In *SDM*, 2004.
- [21] X. Yan and J. Han. gSpan: Graph-based substructure pattern mining. In *ICDM*, 2002.
- [22] H. Yang et al. Discovering spatial relationships between approximately equivalent patterns. In *BIOKDD*, 2004.
- [23] H. Yang et al. A generalized framework for mining spatio-temporal patterns in scientific data. In *OSU-CISRC-5/05-TR14, Ohio State University*, 2005.
- [24] Kenneth Yip. Structural inferences from massive datasets. In *IJCAI*, 1997.
- [25] M.J. Zaki et al. New algorithms for fast discovery of association rules. In *SIGKDD*, 1997.
- [26] X. Zhang et al. Fast mining of spatial collocations. In *SIGKDD*, 2004.