Geoinformatica (2013) 17:387–412
DOI 10.1007/s10707-012-0157-8

Spatio-temporal polygonal clustering with space and time as first-class citizens

Deepti Joshi · Ashok Samal · Leen-Kiat Soh

Received: 18 March 2011 / Revised: 21 January 2012
Accepted: 1 March 2012 / Published online: 15 March 2012
© Springer Science+Business Media, LLC 2012

Abstract Detecting spatio-temporal clusters, i.e. clusters of objects similar to each other occurring together across space and time, has important real-world applications such as climate change, drought analysis, detection of outbreak of epidemics (e.g. bird flu), bioterrorist attacks (e.g. anthrax release), and detection of increased military activity. Research in spatio-temporal clustering has focused on grouping individual objects with similar trajectories, detecting moving clusters, or discovering convoys of objects. However, most of these solutions are based on using a piece-meal approach where snapshot clusters are formed at each time stamp and then the series of snapshot clusters are analyzed to discover moving clusters. This approach has two fundamental limitations. First, it is *point-based* and is not readily applicable to polygonal datasets. Second, its static analysis approach at each time slice is susceptible to inaccurate tracking of dynamic cluster especially when clusters change over both time and space. In this paper we present a spatio-temporal polygonal clustering algorithm known as the Spatio-Temporal Polygonal Clustering (STPC) algorithm. STPC clusters spatial polygons taking into account their spatial and topological properties, treating time as a first-class citizen, and integrating density-based clustering with moving cluster analysis. Our experiments on the drought analysis application, flu spread analysis and crime cluster detection show the validity and robustness of our algorithm in an important geospatial application.

Keywords Spatio-temporal data mining · Temporal data analysis · Polygonal clustering · Drought analysis · Trend analysis

1 Introduction

The increasing numbers of tracking devices, sensors, and global positioning satellites have led to the generation of gigabytes of spatio-temporal data with relative ease for a variety of geo-physical

D. Joshi (✉) · A. Samal · L.-K. Soh
Department of Computer Science and Engineering, University of Nebraska-Lincoln, Lincoln, NE 68588-0115, USA
e-mail: djoshi@cse.unl.edu

variables. Detecting spatio-temporal clusters, i.e. clusters of objects similar to each other occurring together across space and time, has important real-world applications. For example, understanding the earth's atmosphere and the various geophysical processes that occur across time has been defined as a "Grand Challenge" [12]. Some important applications include climate change analysis, drought analysis, detection of outbreak of epidemics (e.g. bird flu), bioterrorist attacks (e.g. anthrax release), and detection of increased military activity [11]. The detection and use of spatio-temporal clusters can enable us to discover trends and patterns that will in turn enable us to learn from the past, and be better prepared for the future [1].

In order to detect spatio-temporal clusters of objects several challenges need to be addressed. The first problem to be addressed is the fair treatment of the temporal dimension along with the spatial dimension. Incorporation of time within the spatial algorithms has always been a challenge within the GIS community. In many critical applications time is treated as a continuous variable just as space. In such situations objects or events may be connected to each other across the temporal dimension, instead of the spatial dimension. If the dataset is atemporal, that is, the data represents the characteristics at one time instant only, trends or changes occurring with time are impossible to detect. The other main challenge with spatio-temporal datasets is the high correlation among data points that exists within such datasets. The various samples collected across the spatial and temporal dimensions cannot be assumed to be independent of each other. Finally, while several objects studied across the spatial dimension can be treated as points, the anthropogenic representation of space tends to be polygonal. Polygonal spatial objects are much more complex in nature as compared to point objects as they add a set of structural and topological attributes to the problem which are non-existent within the point datasets.

Past research has focused on the discovery of spatio-temporal clusters by grouping objects with similar trajectories, detecting moving clusters, or discovering convoys of objects [3–5, 9, 14]. All of this work is point-based and with the assumption that these points are mobile. Detecting spatio-temporal clusters of phenomena such as drought and disease outbreaks, however, is a fundamentally different problem as the geographic space is divided into a set of *polygons* (e.g. states, counties, etc.), and the polygons themselves do *not* move with the passage of time. However, a drought or disease may move across these fixed set of polygons spreading across several counties with the passage of time.

Therefore, the current techniques for detecting spatio-temporal clusters are inadequate for the aforementioned problems because of the following reasons:

1) The current techniques are all point-based where only the longitude and the latitude of the object are used. If polygons—naturally rich objects with topological and structural properties—are represented as points, a significant amount of information is lost [6, 7]. For example, the physical length of shared boundary between two polygons is lost in point representation.
2) These algorithms follow a time slicing approach; that is, snapshot clusters are formed at each time stamp and then a comparison is made between the clusters across various timestamps to detect moving clusters. This approach translates to performing spatial clustering at each time stamp, and therefore it does not cluster entities that occur across different time stamps leading to an unbalanced treatment of space and time. While performing spatio-temporal clustering, time must be treated as a 'first-class citizen', i.e. time must be given equal importance as space. This is important to accurately track the dynamic clusters especially when clusters change significantly over time and space. An example is given in Section 2.3 to demonstrate the loss of information when giving more importance to the spatial dimension as compared to the temporal dimension.

3) Convoys [3, 4, 14] are discovered in an object dataset where the objects move across space in time however their non-spatial attributes remain constant; for example, a convoy of vehicles moving along a highway. But not all objects move across space and time without changing their non-spatial attributes. For example, studying the movement of drought clusters, while the underlying polygons may remain constant, their non-spatial attributes indicating the presence or absence of drought may change with time. Thus, the convoy detection algorithms would not work because the objects themselves do not move. Moreover, the attributes of the objects change.

We present a clustering algorithm for spatio-temporal polygonal datasets known as the Spatio-Temporal Polygonal Clustering (STPC) algorithm. STPC is based on the density-based clustering principle as this clustering paradigm naturally adapts to concepts such as spatial autocorrelation and Tobler's first law of geography– 'All things are related, but nearby things are more related than distant things' [13] (see Section 2.1 for details). STPC takes into account the spatial and topological properties of the polygons while taking into account the spatial neighborhood of the polygons as done previously in [6]. Furthermore, while the current density-based algorithms (e.g. DBSCAN [2], P-DBSCAN [6]) dis-regard the temporal dimension of a polygon, we re-define the neighborhood of a polygon to not only take into consideration the spatial neighborhood of the polygon, but also the temporal neighborhood of a polygon. As a result of taking the spatio-temporal neighborhood of a polygon into account, we are able to treat both space and time as 'first-class citizens'—a feat that the other algorithms are not able to achieve.

In other words, STPC detects a spatio-temporal cluster by spanning across both the spatial and temporal dimensions, thus treating them equally by considering the spatio-temporal neighborhood of a polygon, and therefore produces a cluster which can be viewed as a single cloud (a 3-dimensional object—with space being 2-dimensional and time being a single dimension) in the spatio-temporal space. To illustrate this point further the following figure (Fig. 1a) shows a spatio-temporal cluster discovered using STPC. Figure 1b displays the spatial organization of the cluster at three different time instances. In Fig. 1b at time instance t_3 the members of the cluster are highlighted by four different ovals encircling the polygons. These polygons are not contiguous in space and hence with a time-slicing approach, will never be clustered together. However, as we can see from Fig. 1a, they are indeed members of the same cluster since they are connected in the temporal dimension.

Thus, from Fig. 1, we can see that STPC treats time as a continuous variable similar to space. This enables us to capture many of the similar properties of a cluster across the temporal dimension as those that can be captured only across the spatial dimension by the current state of the art. For example, the clusters discovered by STPC may be both spatially and temporally contiguous, or both spatially dis-contiguous and temporally dis-contiguous, or contiguous in any one dimension and dis-contiguous in the other. Therefore, STPC

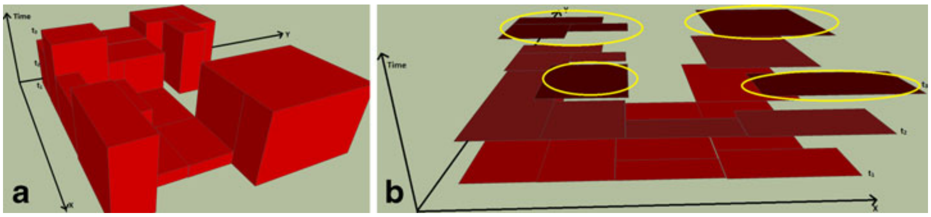


Fig. 1 **a** A three-dimensional model of a spatio-temporal cluster obtained using STPC **b** members of the spatio-temporal polygonal cluster at different time instances

presents a more natural and cohesive method to detect spatio-temporal clusters, as opposed to using the time slicing approach which treats time as an additional attribute, rather than treating it as a continuous dimension similar to space.

In this paper we study the geospatial space that is divided into polygons. *Thus the polygons themselves do not move in time, but their non-spatial attributes or properties may change with time*. We examine the accuracy and efficiency of our algorithms in drought analysis, by discovering the spatio-temporal drought clusters in the state of Nebraska, in United States. Followed by which we discover the swine flu spread clusters within the state of California in United States, and finally we discover spatio-temporal crime clusters within the city of Lincoln, Nebraska. As a part of these experiments we show the effect of the different parameters of STPC, along with the robustness and scalability of our algorithm. Furthermore, while doing the drought analysis we compare and contrast the results of STPC with other spatio-temporal clustering algorithms presented in the literature and described in Section 2.2. The experimental results are presented in Section 4. We have shown that our algorithm outperforms other spatio-temporal clusters by retaining the maximum information about the clusters across space and time, and preventing one cluster from being split into two or more clusters. In other words, STPC is most capable of capturing big shifts within the spatio-temporal clusters, and maintaining the history of the cluster.

The rest of the paper is organized as follows. Section 2 presents a brief introduction to the density-based principles as defined by [2] and the related work to spatio-temporal clustering. Section 3 goes over the framework for our spatio-temporal neighborhood and presents the algorithm STPC. Section 4 presents our experimental results, and Section 5 gives our conclusion and future work.

2 Background and related work

In this section we present a brief background on the principles of density-based clustering—on which our STPC algorithm is grounded—and the state of the art in discovering moving clusters. We also illustrate an example that demonstrates the difference in performing snapshot clustering and detecting convoys, versus treating both space and time as first class variables and detecting moving clusters.

2.1 Density-based clustering principles

A density-based clustering algorithm hinges upon the assumption that a valid cluster must have sufficient density. As suggested by Tobler's first law of Geography [13] and properties such as spatial autocorrelation [15], phenomenon occurring in space and time naturally adapt to the density-based clustering paradigm. Ester et al. proposed a density-based clustering algorithm for point datasets, called DBSCAN [2]. Here we list the main concepts of density-based clustering for points as defined in [2]. Let D be a database of points.

Definition 1: (ε-neighborhood of a point) The ε-neighborhood of a point p, denoted by $N_\varepsilon(p)$, is defined by $N_\varepsilon(p) = \{q \in D | dist(p, q) \leq \varepsilon\}$.

Definition 2: (directly density-reachable) A point p is directly density-reachable from a point q wrt. (ε, $MinPts$) if (1) $p \in N_\varepsilon(q)$ and (2) $\|N_\varepsilon(q)\| \geq MinPts$ (core point condition).

Definition 3: (density-reachable) A point p is density reachable from a point q wrt. (ε, $MinPts$) if there is a chain of points $p_1,\ldots,p_n$, $p_1=q$, $p_2=p$ such that for all i: p_{i+1} is directly density-reachable from p_i.

Definition 4: (density-connected) A point p is density connected to a point q wrt ε, and if there is a point o such that both, p and q are density-reachable from o wrt (ε, *MinPts*). Density-connectivity is a symmetric relation. For density reachable points, the relation of density-connectivity is also reflexive.

Definition 5: (cluster) A cluster C *wrt.* (ε, *MinPts*) is a non-empty subset of D satisfying the following conditions:

1. $\forall p,q$: if $p \in C$ and q is density-reachable from p wrt. (ε, *MinPts*) and $q \in C$. (Maximality)
2. $\forall p,q \in C{:}p$ is density-connected to q *wrt.* (ε, *MinPts*). (Connectivity)

Definition 6: (noise) Let $C_1,\ldots,C_k$ be the clusters of the database D wrt. parameters (ε, *MinPts*), then we define the noise as the set of points in the database D that do not belong to any cluster C_i, i.e. $noise = \{p \in D | \forall i : p \notin C_i\}$.

The above density-based concepts are applied in the DBSCAN clustering algorithm that is used by the traditional moving cluster and convoy detection algorithms on point datasets. While applying the same principles of density-based clustering to polygons in the spatial and temporal dimensions, we, have extended the idea of the ε-neighborhood of a point that takes into account only the spatial neighborhood of the points to a spatio-temporal neighborhood of a polygon that takes into account both the spatial and temporal neighborhoods of polygons. By considering the temporal neighborhood of the polygon, we treat time as a 'first-class citizen' along with space (see Section 3).

2.2 Detecting spatio-temporal clusters

Below we discuss the state-of-art in clustering algorithms detecting moving clusters in space and time. We discuss the Moving Cluster algorithm, the Coherent Moving Cluster algorithm, the Valid Convoy Discovery algorithm, and the Cluster Over Time algorithm. Followed by which we list the disadvantages of these algorithms, and explain how our approach is different.

The general approach followed by the well accepted *Moving Cluster* (MC) algorithm [9] is as follows: A density-based clustering (e.g., DBSCAN) is first performed on the moving objects at each timestamp to find snapshot density-connected clusters of arbitrary shapes and then the intersection snapshot clusters appearing during consecutive timestamps is detected as a moving cluster if they share at least a certain number of objects in common which is defined with respect to a threshold θ $\left(\frac{|c(t) \cap c(t+1)|}{|c(t) \cup c(t+1)|}\right) \geq \theta$ where $c(t)$ and $c(t+1)$ denote two adjacent snapshot clusters at time t and $t+1$ respectively.

Jeung et al. [4] extended MC and proposed the *Coherent Moving Cluster* algorithm (CMC). CMC first performs density-based clustering at each timestamp to find snapshot clusters of arbitrary shapes. The following two conditions are then tested: first, a convoy must have clusters in at least k consecutive time stamps (lifetime constraint), and second, of the intersection of two consecutive snapshot clusters must have at least m objects in common. If both the conditions are true then it is detected as a convoy.

Yoon & Shahabi [14] stated that the MC algorithms described above have a critical problem with accuracy – they tend to miss larger convoys and retrieve invalid ones where the density-connectivity among the objects is not completely satisfied. To overcome this problem, the authors proposed the algorithm VCoDA (*Valid Convoy Discovery Algorithm*). This algorithm works in two phases: first a set of all partially connected convoys is discovered from a given set of moving objects using the PCCD algorithm [14], and then the density-connectivity of each partially connected convoy is validated using the DCVal algorithm [14] to obtain a complete set of valid convoys. The first phase of VCoDA extends the CMC algorithm by scanning through the entire time span, and updating a set of density connected snapshot clusters incrementally by consecutive ones with sufficient objects in common under four operations (i.e., insert, extend, delete, and return). This approach is further extended in the second phase such that the density-connectivity of each partially connected convoy is incrementally verified at every timestamp either by immediate, single re-clustering, or recursive validation.

In [10] a simple set of formulas was proposed to predict which paired objects will move in the ε-neighborhood of each other. From these pair-wise ε-neighborhood relationships, a COOT (Core Object Over Time) algorithm is constructed to identify which objects will become core objects of future density-based clusters. This information reveals where, when, and how long the dense concentrations of objects may happen. Contents of density-based clusters over time can also be constructed using the Clusters Over Time (COT) algorithm with higher space and computation cost.

The first three algorithms mentioned above apply a time-slice approach i.e. they first perform spatial clustering using DBSCAN at each time stamp in order to discover spatial clusters. These are then compared consecutively in the presence of some user defined constraints, and spatio-temporal clusters or moving clusters are detected. While VCoDA is an improvement over the other moving cluster discovery algorithms (MC and CMC), as this algorithm discovers the maximum number of valid convoys or moving clusters, some post-processing of the results is required to prune out the invalid convoys. The authors in [10] claim to move away from the time-slice approach by detecting core objects over time. However, their algorithm—COT—fails to do so completely. This is because they compare the core objects previously detected across small time intervals to detect spatio-temporal clusters across that particular time interval. Thus one may argue that the algorithm detects several incomplete clusters, i.e. several small clusters which should in fact be a part of the same cluster are detected. Moreover all the above described algorithms assume that the dataset consists of objects with similar non-spatial attribute values. They cannot distinguish between objects that lie within the ε-neighborhood of each other but are not similar to each based on their non-spatial attributes. While this may seem trivial, however, producing clusters using the density-based scheme that have uniform non-spatial attributes cannot be simply implemented using a pair-wise distance measurement methodology. This is because while density-connectivity is a transitive property flowing within the cluster from one object to the next, similarity between objects based on non-spatial attributes is not a transitive property. Finally all the above mentioned techniques are all point-based, and cannot be directly extended to polygons.

We adopt the idea of producing spatio-temporal clusters using the density-based clustering methodology in our approach as well. However, instead of performing spatial clustering across each time interval, we follow an approach that allows us to detect clusters spanning across the spatial and temporal dimensions simultaneously. We also incorporate a strategy that allows us to detect clusters with similar non-spatial attributes. Finally our approach is designed to cluster polygons instead of points. In Section 4 we present a comparison of these point-based algorithms with our polygon—based algorithm STPC.

2.3 An example

Consider the polygons shown in Fig. 2. The figure shows the movement of drought across the counties (represented as orange polygons) with the passage of time. The centroids of the counties are marked as dots. The temporal dimension is discretized for the purposes of detecting counties experiencing drought at the same time, and in order to detect the temporal

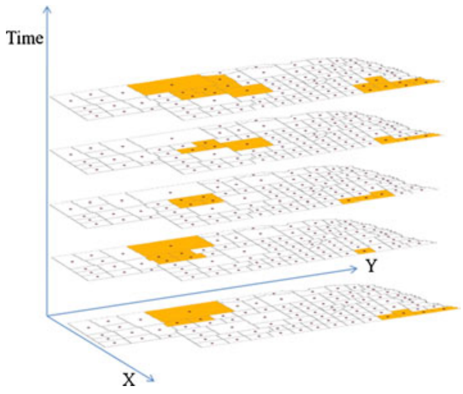


Fig. 2 Sample dataset of polygons with drought at each time stamp. The centroids are shown as *dots* within each polygon

neighbors—these neighbors may be a nanosecond apart, or a week apart, or a year apart, and will be the characteristic of the dataset being clustered. However, when clustering is performed, we consider these neighbors in a temporal continuum just like we consider spatial neighborhood as a continuum. In essence, we consider the notion of temporal neighbors in the clusters along the time dimension while conventional techniques consider one time slice at a time and do not consider temporal neighbors.

Traditional trajectory clustering algorithms discussed on the previous sub-section perform snapshot density-based clustering (using algorithms such as DBSCAN [2]) at each time-stamp. Due to the snapshot clustering approach, and constraints such as the minimum number of objects (m)—i.e. the minimum number of common points that must be present within the two consecutive clusters and the lifetime constraint (k)—i.e. the minimum number of time stamps across which the cluster must exist, sudden changes that may happen in the cluster are not captured as part of the evolving cluster and instead viewed as the stop points of the cluster. As a result, the cluster breaks into multiple clusters leading to loss of information about the structure and contents of the cluster. For example, for the drought polygons shown in Fig. 2, if we set m=2 (i.e., the minimum number of objects), and k=2 (i.e., the lifetime constraint), two clusters are detected as shown in Fig. 3a. The orange

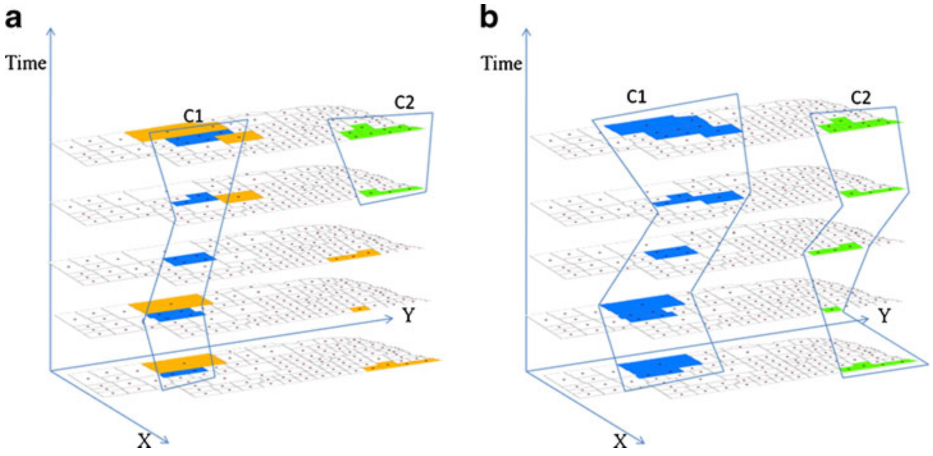


Fig. 3 **a** Point-based spatio-temporal clusters formed using snapshot clustering approach. **b** Polygonal spatio-temporal clusters using time as a first-class citizen

polygons in Fig. 3a are the drought polygons that do not get included in any cluster due to the constraints of minimum number of points that must be common within each sub-cluster formed at each time stamp and the minimum lifetime constraint. Thus, it can be seen that information may be lost using the techniques described in the previous section.

Our approach, on the other hand, gives time equal importance as space, and performs spatio-temporal clustering across time and space concurrently. As a result, two spatio-temporal clusters are detected (Fig. 3b). In a way, our approach considers not just n polygons, i.e. the number of polygons at any timestamp, but $n \times t$ (i.e., n polygons multiplied with t time stamps) objects, and clusters them together without any bias. As a result no information is lost as compared to the traditional approach. For example, the cluster (C2) shown in Fig. 3a becomes a part of a bigger cluster—cluster C2 in Fig. 3b. This information was not captured by the point-based approaches because there were none/or only one common points within the snap-shot clusters at time stamp *t1*, *t2* and time stamp *t4*. However, because of the use of a spatio-temporal neighborhood as described in Section 3, the complete cluster is detected using our proposed approach. Furthermore, the second cluster (C1) detected by our approach is also more complete as compared to the cluster C1 shown in Fig. 3a. This is because of considering the polygons spatial and topological properties into account that were lost in the point representation.

3 Spatio-temporal polygonal clustering

Consider a set of polygons that exist in space and time. Each polygon has three categories of attributes associated with them. The first category is the space 'where' the polygon exists and is referred to by a set of spatial attributes, the second category is the time 'when' the polygon exists and is referred to by a set of temporal attributes, and the third category is 'what' the polygon is, and is referred to by a set of non-spatial, non-temporal attributes. A polygon that can be represented using all the above three categories of attributes is known as a spatio-temporal polygon.

In this section we first present the density-based concepts for spatio-temporal polygons that are based on the density-based concepts for points in [2], and on the density-based concepts for polygons defined in [6]. Please note that in [6] we do not consider the temporal dimension. Furthermore, we do not consider the non-spatial attributes of the polygons in clustering. Next, we formalize the problem of spatio-temporal polygonal clustering, followed by which we present our algorithm STPC that is used to detect spatio-temporal clusters. Finally, we formally present the structure of a spatio-temporal polygonal cluster in terms of the spatial dimension keeping time constant, and in terms of the temporal dimension keeping space constant.

3.1 Density-based concepts for spatio-temporal polygons

Here we adapt the density-based concepts for points as proposed by Ester et al [2] to spatio-temporal polygons. Please note that Definitions 1 to 8 do not take into account the non-spatial attributes of the polygons, and are only based on the spatial and temporal dimensions of the polygonal dataset.

Definition 1: A *spatio-temporal polygon* p_s^t is a polygon that exists at the location indexed by *s and* at the time interval indexed by *t*.

The index *s* specifying the location of a polygon serves as an identifier—a label—for the area covered by the polygon in the two-dimensional space which may be stored as a sequence of vertices of the polygon. Similarly, the time index t serves to discretize the temporal dimension, and identify a point or interval in time when the polygon exists. (Henceforth, all polygons are spatio-temporal polygons unless specified otherwise.)

Definition 2A: A *spatial neighbor* of polygon p_s^t is any polygon p_k^t such that $dist(p_s^t, p_k^t) \leq \varepsilon$ where $dist(p_s^t, p_k^t)$ is any distance function that computes the physical distance between two polygons. Several distance functions have been proposed in the past for computing the distance between two polygons. Please see [8] for further details. We specifically apply the modified Hausdorff distance function defined in [7]. Note that the temporal aspect is constant or reduced to a fixed interval or time instant in this case.

$$SNeighbor\left(p_i^{t'}, p_j^{t''}|\varepsilon\right) \text{ istrueiff } dist\left(p_i^{t'}, p_j^{t''}\right) \leq \varepsilon \text{ and } t' = t''.$$

Definition 2B: The *spatial neighborhood* of a polygon p_s^t given ε, $SN(p_s^t|\varepsilon)$, is the set of the spatial neighbors of the polygon, that is $p_j^{t''} \in SN(p_s^t|\varepsilon)$ *if* $SNeighbor\left(p_s^t, p_j^{t''}|\varepsilon\right)$.

Definition 3A: A *temporal neighbor* of a polygon p_s^t is a polygon that occupies at least some of the space indexed by s, $\varphi(p_s^t)$, at the time intervals $t \pm \omega$ where ω is a user-defined parameter to define the extent of the temporal neighborhood of a polygon. Note here, in contrast to the spatial neighborhood, the spatial dimension is instead held to a constant space.

$$TNeighbor\left(p_i^{t'}, p_j^{t''}|\omega\right) \text{ is true iff } \varphi\left(p_i^{t'}\right) \cap \varphi\left(p_j^{t''}\right) \neq \phi \text{ and } |t' - t''| \leq \omega.$$

where $\varphi(p_i^{t'})$ refers to the area covered by the polygon $p_i^{t'}$ at time instant t'.

For example, for a static set of polygons such as geospatial polygons, if ω=3, the temporal neighbors of polygon $p_i^{t'}$ is the set of spatio-temporal polygons $TNeighbors(p_i^{t'}) = \{p_i^{t'-3}, p_i^{t'-2}, p_i^{t'-1}, p_i^{t'+1}, p_i^{t'+2}, p_i^{t'+3}\}$.

Definition 3B: The *temporal neighborhood* of a polygon p_s^t given ω, $TN(p_s^t|\omega)$, is the set of the temporal neighbors of the polygon, that is $p_j^{t''} \in TN(p_s^t|\omega)$ *if* $TNeighbor\left(p_s^t, p_j^{t''}|\omega\right)$.

Definition 4: The *spatio-temporal neighborhood* of polygon p_s^t given ε and ω, $STN(p_s^t|\varepsilon, \omega)$, is the union of the spatial neighborhood and the temporal neighborhood.

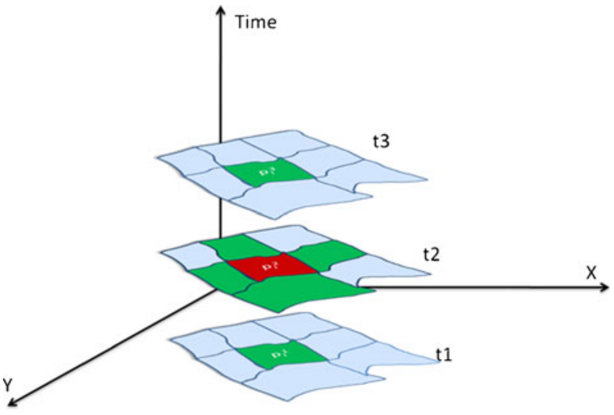
$$STN(p_s^t|\varepsilon, \omega) = SN(p_s^t|\varepsilon) \cup TN(p_s^t|\omega)$$

Figure 4 shows the spatio-temporal neighborhood of polygon p_s^{t2}. The red polygon is the polygon p_s^{t2}, and the green polygons form the spatio-temporal neighborhood of the polygon p_s^{t2}, taking ω=1 and ε=1.

Definition 5: *Core spatio-temporal polygon* is a polygon that has at least *MinPoly* number of polygons in its spatio-temporal neighborhood i.e $STN(p_s^t|\varepsilon, \omega) \geq MinPoly$.

Definition 6: A spatio-temporal (ST) polygon $p_j^{t'}$ is *directly density-reachable* from another spatio-temporal polygon $p_j^{t''}$ wrt. ε, ω and *MinPoly* if (1) $p_j^{t'} \in STN\left(p_j^{t''}|\varepsilon, \omega\right)$, (2) $\left|STN\left(p_j^{t''}|\varepsilon, \omega\right)\right| \geq MinPoly.$

Fig. 4 Spatio-temporal neighborhood (*green polygons*) of polygon p_s^{t2} (*red polygon*)



Definition 7: A polygon $p_j^{t'}$ is *density reachable* from another polygon $p_i^{t''}$ wrt. ε, ω and *MinPoly* if there is a chain of ST-polygons $p_1^{t'}, ..., p_n^{t''}$, $p_1^{t'} = p_j^{t'}, p_n^{t''} = p_i^{t''}$ such that $\forall i : p_{i+1}^{t'}$ is directly density-reachable from $p_j^{t'}$.

Definition 8: A polygon $p_j^{t'}$ is *density connected* to another polygon $p_i^{t''}$ wrt. ε, ω and *MinPoly* if there is a polygon p_k^t such that both, $p_j^{t'}$ and $p_i^{t''}$ are density-reachable from p_k^t wrt. ε, ω and *MinPoly*. Density-connectivity is a symmetric relation. For density reachable polygons, the relation of density-connectivity is also reflexive.

3.2 Problem definition

Given a set of spatio-temporal polygons $P = \{p_1^{t1}, p_2^{t2}, ..., p_n^{tm}\}$, and input parameters ε (used to define the spatial extent of the spatio-temporal neighborhood of a polygon), ω (used to define the temporal extent of the spatio-temporal neighborhood of a polygon), *MinPoly* (used to identify a core polygon), and α (used to identify the similarity between two or more polygons), the problem is to divide the dataset P into a set of clusters $C = \{C_1, C_2, ...C_k\}$ such that each polygon within a cluster is either spatially connected to another polygon within the cluster, or is temporally connected to another polygon within the cluster, and is similar to all other polygons within the cluster. In other words, a spatio-temporal cluster C_k must satisfy the following conditions:

1. $\forall p_j^{t'}, p_i^{t''}$: if $p_j^{t'} \in C_k$ and $p_i^{t''}$ is density-reachable from $p_j^{t'}$ wrt. ε, ω and *MinPoly* then $p_i^{t''} \in C_k$. (Maximality)
2. $\forall p_j^{t'}, p_i^{t''} \in C_k : p_j^{t'}$ is density-connected to $p_i^{t''}$ wrt. ε, ω and *MinPoly*. (Connectivity)
3. $\forall p_j^{t'}, p_i^{t''} \in C_k,\ d_N\left(p_j^{t'}, p_i^{t''}\right) = 0$ (Strong Uniformity) or $\forall p_j^{t'}, p_i^{t''} \in C_k,\ d_N\left(p_j^{t'}, p_i^{t''}\right) \leq \alpha$ (Weak Uniformity)where $d_N\left(p_j^{t'}, p_i^{t''}\right)$ is the distance between the non-spatial attributes of the polygons and is computed using the Euclidean distance function, and α is a user-defined input parameter. Figure 5 shows an example of a spatio-temporal cluster.

3.3 Spatio-Temporal Polygonal Clustering (STPC) algorithm

In order to detect spatio-temporal clusters, we propose a new algorithm called Spatio-Temporal Polygonal Clustering (STPC) algorithm. Unlike other algorithms as defined in

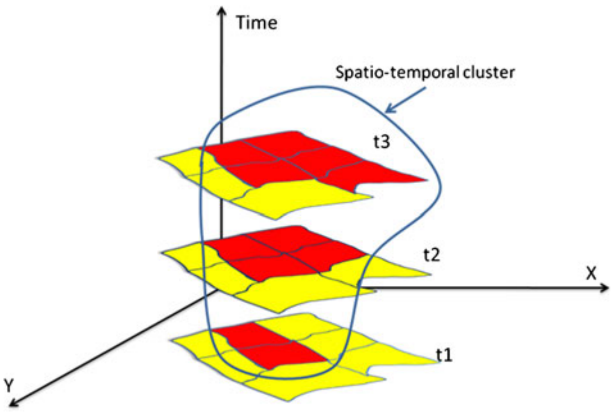
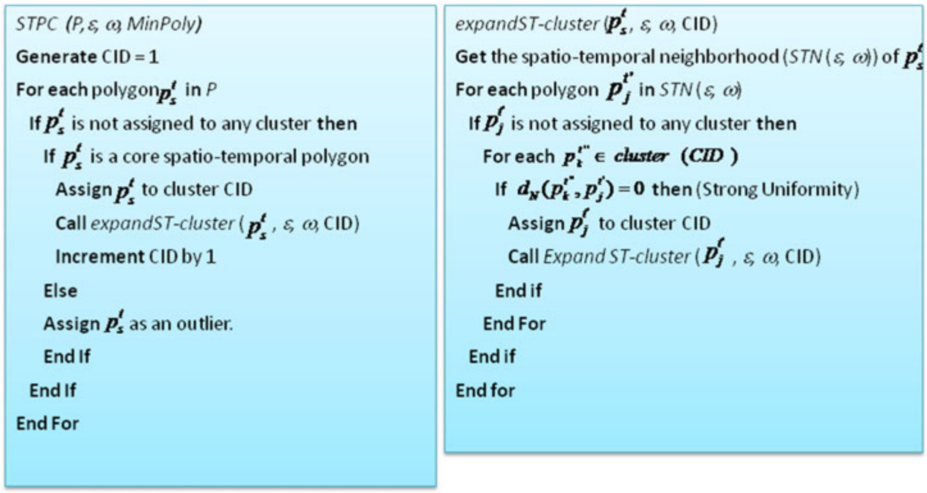


Fig. 5 A drought spatio-temporal cluster (*red polygons*)

Section 2.2, STPC does not perform density-based clustering at each time stamp. STPC (presented in Fig. 6), instead, uses the density-based concepts defined above for spatio-temporal polygons in order to detect a spatio-temporal cluster that extends both in space and over time, thus treating time as a first-class citizen along with space, and removing the need to find the intersection of snapshot clusters across consecutive time stamps. Please note: we discretize time in order to detect the temporal neighbors of a polygon—these neighbors may be a nanosecond apart, or a week apart, or a year apart, as that will be the characteristic of the dataset being clustered. However, during the clustering process, the temporal neighbors are considered continuously just as the spatial neighbors are considered continuously. Hence, both space and time are treated as first class citizens in our clustering approach.

Given a dataset of spatio-temporal polygons (P) STPC begins with a polygon (p_s^t) that has not been assigned to any cluster previously. If p_s^t meets the necessary conditions to be defined as a core spatio-temporal polygon, it is assigned a new cluster CID. It then calls the *expandST-Cluster* method that examines each polygon that falls in the spatio-temporal neighborhood of p_s^t based on ε and ω. If a spatio-temporal neighbor has similar non-



```
STPC ($P, \varepsilon, \omega, MinPoly$)
Generate CID = 1
For each polygon $p_s^t$ in $P$
  If $p_s^t$ is not assigned to any cluster then
    If $p_s^t$ is a core spatio-temporal polygon
      Assign $p_s^t$ to cluster CID
      Call expandST-cluster ($p_s^t$, $\varepsilon$, $\omega$, CID)
      Increment CID by 1
    Else
    Assign $p_s^t$ as an outlier.
    End If
  End If
End For
```

```
expandST-cluster ($p_s^t$, $\varepsilon$, $\omega$, CID)
Get the spatio-temporal neighborhood ($STN(\varepsilon, \omega)$) of $p_s^t$
For each polygon $p_j^{t'}$ in $STN(\varepsilon, \omega)$
  If $p_j^t$ is not assigned to any cluster then
    For each $p_i^{t''} \in$ cluster (CID)
      If $d_N(p_i^{t''}, p_j^{t'}) = 0$ then (Strong Uniformity)
        Assign $p_j^t$ to cluster CID
        Call Expand ST-cluster ($p_j^t$, $\varepsilon$, $\omega$, CID)
      End if
    End For
  End if
End for
```

Fig. 6 The Spatio-Temporal Polygonal Clustering (STPC) algorithm with strong uniformity

spatial attributes to p_s^t, and to every other spatio-temporal polygon that has already been assigned to cluster CID, i.e. if $d_N\left(p_j^{t'}, p_i^{t''}\right) = 0$ the neighbor is assigned to the same cluster as p_s^t, and the *expandST-Cluster* method is subsequently called recursively on this neighbor. This process is repeated as long as there exists a polygon that has not been assigned to a cluster.

It should be noted that the test $d_N\left(p_j^{t'}, p_i^{t''}\right) = 0$ is valid only in the case of categorical non-spatial attributes, and the case reduces to a strong uniformity one. For example, if the polygons are classified as "drought" or "no-drought" polygons, and $d_N\left(p_j^{t'}, p_i^{t''}\right) = 0$; i.e., if either both the polygons $p_j^{t'}$ and $p_i^{t''}$ are classified as drought polygons, or they both are classified as no-drought polygons, only then both the polygons will be assigned to the same cluster. In order to handle non-spatial attributes having continuous values or ordinal values, the test $d_N\left(p_j^{t'}, p_i^{t''}\right) \leq \alpha$ can be used instead (for weak uniformity), where α is a user-defined parameter.

Complexity analysis of STPC As our algorithm follows the structure of the density-based clustering algorithm DBSCAN, the time complexity of our algorithm will be the same as DBSCAN—which is $O(n^2)$ without the use of an indexing structure, where n is the number of polygons—if the non-spatial attributes of the polygons are not taken into account (NOTE: DBSCAN does not take the non-spatial attributes of objects into consideration). If an indexing structure such as a R* tree is used, then the time complexity will be reduced to $O(n \log n)$. Looking more closely, we find that the time complexity of our algorithm can be re-represented as $O(t\times n \log (t\times n))$, where t denotes the number of discretized temporal intervals or instances used for clustering the dataset, whereas the time complexity for the conventional piecemeal approach is $O(t\times (n \log n))$ as DBSCAN algorithm will be called t times.

With the addition of the non-spatial attributes, STPC finds clusters of spatio-temporal polygons that are not only located close to each other in the spatial and temporal dimensions, but are also similar to each other in the non-spatial attribute domain. Thus, the time complexity of STPC increases from $O(n^2)$ to $O(n^3)$. However, as STPC is treating a set of geospatial polygons, which themselves do not move with time, the spatial neighbors for each polygon will remain the same across each discretized time stamp or interval. Thus, if the spatial neighbors for a given ε are pre-computed and provided as input to the algorithm, then the time complexity of STPC will be $O(n\times t\times m\times k)$ where m is the maximum number of neighbors, k is the maximum number of polygons assigned to a cluster, and in most cases $m < n$ and $k <$ n.

In order to see how STPC scales with the number of polygons, we randomly assigned a drought index of 0 or 1 to the census tracts of the state of Nebraska for 10 weeks. For each run we increased the number of census tracts starting with 50 and going up to 500. Thus the number of polygons at each run were 500 (n=50, t=10), 1,000 (n=100, t=10), and so on. The results obtained are shown in the following figures. Figure 7 shows the graph of the actual runtime of STPC in milliseconds compared to the $O(n^3)$ and $O(n \log n)$.

3.4 Selecting input parameters

In order to select the appropriate ε for a polygonal dataset, one of the following strategies may be followed:

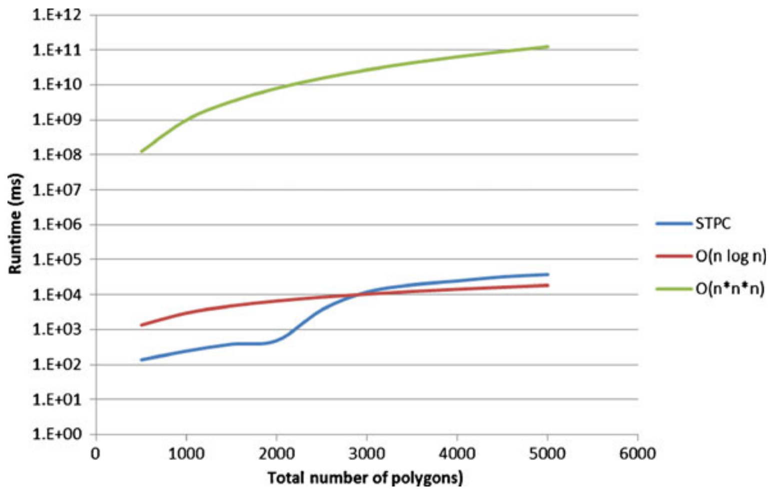


Fig. 7 Graph of STPC run times with different number of polygons compared to $O(n^3)$and $O(n\ log\ n)$

1) Using a distance function such as the Hausdorff distance function, compute the pair-wise distance between the polygons within the dataset. Based on the set of the pair-wise distances, ε may be selected as the: mode of the set, the median distance value within the set, or the average distance. However, please note that a bigger ε value may result in the aggregation of two or more clusters within the same cluster.
2) Based on the knowledge of the dataset, the user may identify two polygons that must never be clustered together. Compute the geographic distance between these polygons, and then set ε to a value smaller than the resulting distance between the two polygons.

In order to select the appropriate ω, the user needs to determine if the domain is such that the cluster may disappear and re-appear over the same spatial extent within a certain period of time. In such cases selecting an $\omega>1$ will be more appropriate. However, if the domain is such that once the cluster disappears at a certain time stamp or time interval, the same cluster cannot re-appear, then selecting $\omega=1$ will be sufficient.

3.5 Properties of a spatio-temporal polygonal cluster

In order to further illustrate the significance of the spatio-temporal clusters detected using STPC, we present a new methodology to examine the spatio-temporal clusters. A spatio-temporal cluster (ST-cluster) as detected by STPC consists of polygons that are located close to each other across the spatial and temporal dimensions. Figure 8a shows a simple ST-cluster where the spatial dimension has been reduced down to one dimension instead of two. This view of the ST-cluster provides an interface to measure the dynamics occurring within each cluster. Thus, each ST-cluster can be viewed either as a set of ST-slices (Fig. 8b) or as a set of TS-slices (Fig. 8c). Studying the ST-slices and the TS-slices will give us novel information about the dynamics of the spatio-temporal clusters such as the movement patterns, prediction capabilities, and trend analysis of various phenomena as they occur across space and time. For example, studying the ST-slices allows us to investigate how a cluster changes spatially over time—such as the spread of flood (shrinking or expanding) or infections over time. Studying the TS-slices allows us to investigate, on the other hand, how

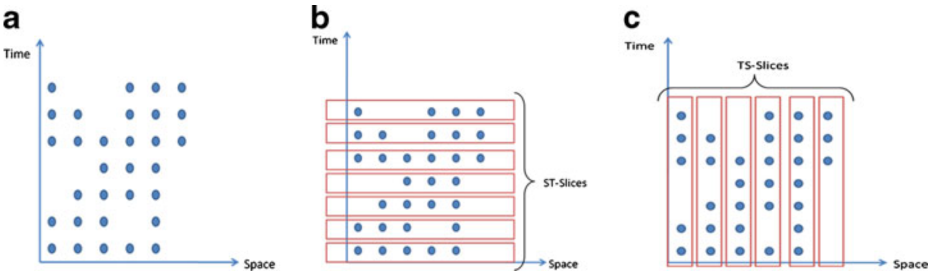


Fig. 8 **a** A simplistic spatio-temporal cluster **b** ST-slices of the spatio-temporal cluster **c** TS-slices of the spatio-temporal cluster

a cluster changes temporally over a connected group (neighbors) of polygons in space. For example, studying the TS-slices of a spatio-temporal polygonal cluster based on crops grown, will allow us to detect the change in the duration of growing season for crops moving from the southwest region of the US to the midwest region.

Next, we present the formal definitions of ST-slices and TS-slices. A spatio-temporal cluster C_S^T consists of:

1. A set of spatio-temporal (ST)-slices $\left\{c_{\{s_1\}}^{t_1}, c_{\{s_2\}}^{t_2}, \ldots, c_{\{s_m\}}^{t_n}\right\}$, where $t_i \in T$ such that $\{s_i\}$ represents the set of polygons $\left\{p_{s1}^t, p_{s2}^t, \ldots, p_{sr}^t\right\}$ that form each ST-slice at *fixed time interval index t*. Henceforth, for simplicity each ST-slice will be represented as $c_{\{s\}}^t$. The space occupied by a ST-slice $\varphi\left(c_{\{s\}}^t\right) = \cup_{i=1}^r \varphi\left(p_{s_i}^t\right)$.
2. A set of temporal-spatial (TS)-slices $\left\{c_{s_1}^{\{t_1\}}, c_{s_2}^{\{t_2\}}, \ldots, c_{s_m}^{\{t_n\}}\right\}$, where $\{t_i\}$ represents the set of polygons $\left\{p_{s1}^{t1}, p_{s1}^{t2}, \ldots, p_{s1}^{tr}\right\}$ that form each TS-slice *at fixed space index* $s1$. Henceforth, for simplicity each TS-slice will be represented as $c_s^{\{t\}}$.

Property 1 If $\omega=1$, then $\varphi\left(c_{\{u\}}^{t1}\right) \cap \varphi\left(c_{\{v\}}^{t2}\right) \neq \phi$, i.e. the intersection of the space of two consecutive ST-slices of a spatio-temporal cluster C_S^T cannot be empty. However, if $\omega>1$ then $\varphi\left(c_{\{u\}}^{t1}\right) \cap \varphi\left(c_{\{v\}}^{t2}\right)$ maybe Ø, i.e. if $\omega>1$, then the intersection of the space of two consecutive ST-slices of a C_S^T maybe empty.

Proof We prove the above axiom in two parts. First, the proof for—If $\omega=1$, $\varphi\left(c_{\{u\}}^{t1}\right) \cap \varphi\left(c_{\{v\}}^{t2}\right) \neq \phi$ is as follows:

The spatio-temporal cluster C_S^T begins by selecting any random spatio-temporal polygon p_s^t, and checking to see if it is a core polygon. If the polygon is indeed a core polygon, its spatio-temporal neighborhood is extracted from the dataset and assigned to the same cluster as the polygon p_s^t itself. Next step is to check if any of the spatial or temporal neighbors are core polygons themselves, and if yes, their neighbors are extracted from the dataset as well, and assigned to the same cluster as p_s^t. This process goes on until no other spatio-temporal polygon gets assigned to the same cluster as p_s^t anymore. Thus, when the first polygon—, polygon p_s^t, gets assigned to the cluster C_S^T, along with its spatio-temporal neighbors, the cluster C_S^T will consist a maximum of three ST-slices. The first ST-slice will consist of only the temporal neighbors of polygon p_s^t at time instant *t—1*, the second ST-slice will consist of the spatial neighbors of the polygon p_s^t and the polygon p_s^t itself (i.e., at time *t*), and the third ST-slice will consist of the temporal neighbors of the polygon p_s^t at time instant *t+1*. As defined before, the temporal neighbors of the polygon p_s^t are the polygons that may exist at time instances $t\pm\omega$, and that occupy *at least* some of the space that was occupied by p_s^t. Thus the intersection of space occupied by any two consecutive ST-slices of a spatio-temporal cluster C_S^T cannot be empty, i.e. $\varphi\left(c_{\{u\}}^{t1}\right)\cap\varphi\left(c_{\{v\}}^{t2}\right)\neq\phi$. Hence proved.

Second the proof for If $\omega>1$, then $\phi\left(c_{\{u\}}^{t1}\right)\cap\phi\left(c_{\{v\}}^{t2}\right)$ maybe Ø is as follows:

If $\omega>1$, when the first polygon, polygon p_s^t, gets assigned to the cluster C_S^T, along with its spatio-temporal neighbors, the cluster C_S^T will consist a maximum of $2\omega+1$ ST-slices. However, as the polygon p_s^t may be designated as a core polygon if it has the required density of MinPoly polygons taking into consideration its spatial neighbors and temporal neighbors across 2ω time intervals, it may happen that temporal neighborhood of polygon p_s^t may be empty at time interval *t*—1. Thus, in this case, the intersection of ST-slice of cluster C_S^T at time interval *t* and at time interval *t*—1 will be empty. Hence proved.

Based on property 1, the following properties of a spatio-temporal cluster (ST-cluster) can be deciphered:

1. If $\omega=1$, then a ST-cluster is contiguous in the temporal dimension
2. If $\omega>1$, then a ST-cluster may lose its temporal contiguity, that is the spatio-temporal cluster may disappear and re-appear at the same location within its lifetime.

4 Experimental analysis

In order to analyze and show the robustness of our algorithm STPC, we first compare its results with other spatio-temporal cluster detection algorithms. Further we study the properties of other parameters of STPC by applying it to the swine flu dataset for the state of California. Finally, in order to show the scalability of our algorithm we have applied it to the crime dataset for the city of Lincoln, NE.

4.1 Comparative analysis using the drought dataset

In this section we compare and contrast STPC with 4 other spatio-temporal cluster detection algorithms. These are the MC, CMC, VCoDA, and COT algorithms (see Section 2 for a brief description of these algorithms). We have applied all five algorithms to a real-world application that aims at finding moving drought clusters over time and space. For our comparative study, we have used the drought dataset for the state of Nebraska.

Dataset description The state of Nebraska has 93 counties. At the end of each week, the U. S. Drought Monitor determines whether each county is in a state of drought based on various measurements of the water cycle. Each county may have regions that experience different levels of drought—severe drought, extreme drought, etc. For our experiments we only take into account whether a county has drought or no drought as a binary decision. 20 weeks of data from Jan 2009 to June 2009 were used for this experiment. Figure 9 shows a representation of the Nebraska counties as spatio-temporal polygons.

The MC and the CMC algorithms have been designed to cluster point datasets. Therefore, the input to both these algorithms is shown in Fig. 9a, where each polygon is represented by

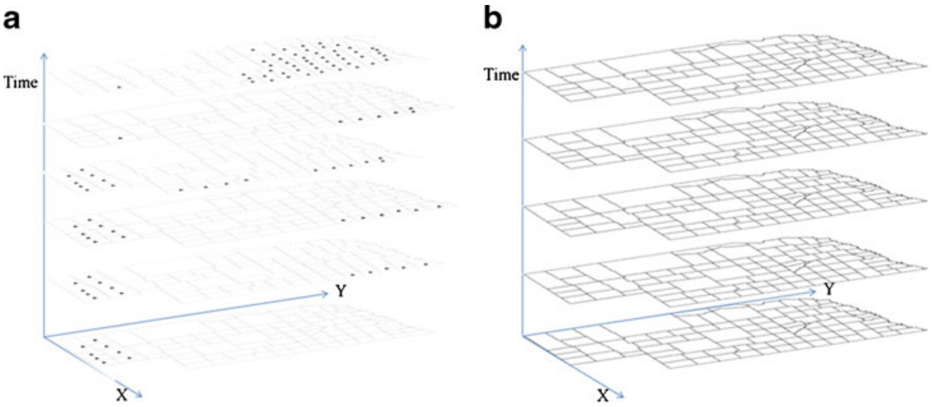


Fig. 9 **a** Point representation of drought counties of Nebraska—dataset for the MC and CMC algorithms **b** Counties of the state of Nebraska—dataset for the STPC algorithm. The discrete time scale for both the datasets is weekly

its centroid. In addition, please note that both these algorithms only work with the point locations, and do not take into account the non-spatial attributes of the points, and therefore, cannot distinguish between drought and no-drought polygons. Thus, the input needs to be filtered to contain only those polygons at each timestamp which are experiencing drought. The STPC algorithm, on the other hand, has been designed to handle spatio-temporal polygonal datasets and thus its input is the set of polygons as shown in Fig. 9b along with their non-spatial attributes.

Results To evaluate our results, we have used as ground truth the drought monitor maps (http://drought.unl.edu/dm/archive.html) produced by the U.S. Drought Monitor (Fig. 10). The drought maps for Nebraska from Jan 2009 to June 2009 show that there are three drought clusters and one no-drought cluster.

As defined in Section II(B), the MC algorithm takes as input the parameters ε, θ, and *MinPts*. For our experiments, we have used ε=52 miles, θ=0.5, and *MinPts*=3. The CMC algorithm takes as input the parameters ε, *k*, *m*, and *MinPts*. For our experiments we have used ε=52 miles, *k*=3, *m*=3, and *MinPts*=3. VCoDA takes as input the parameters ε, *k* and *m*. For our experiments, we have used ε=52 miles, *k*=3 and *m*=3. The STPC algorithm is applied using

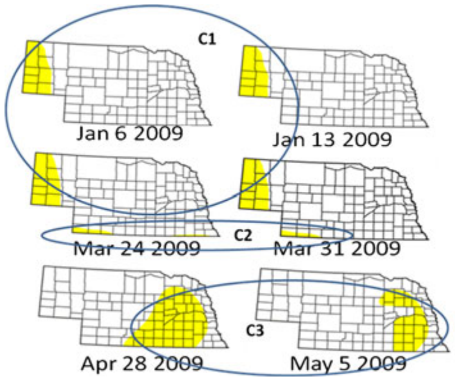


Fig. 10 Sample drought monitor maps from *http://drought.unl.edu/dm/archive.html* showing the three drought clusters

ε=52 miles, ω=1, α=0, and *MinPoly*=3. The COT algorithm only takes as input the parameters ε and *MinPts*. For our experiments, we have used ε=52 miles and MinPts=*3*. Please note that ε=52 miles was selected by computing the pair-wise Hausdorff distance between all the polygons in the dataset, and then finding the mode of the all the distance values.

Using the aforementioned parameters, the MC algorithm discovers 5 drought clusters, the CMC algorithm discovers 4 drought clusters, VCoDA discovers 7 drought clusters, STPC discovers 3 drought clusters along with one no-drought cluster, and the COT algorithm discovers 8 drought clusters. Based on the number of drought and no-drought clusters, only the STPC algorithm produces the results same as the ground truth. Furthermore, other than STPC, none other algorithm is able to discover the no-drought cluster because they do not have the ability to distinguish between the objects being clustered based on their non-spatial attributes. Finally, when we compared the clusters discovered by STPC with the ground truth, we found that they were the same clusters. The result obtained by STPC is shown in Fig. 11.

Please note that the results of STPC cannot be attributed to parameter tuning, but as a logical outcome by a systematic procedure. For our experiments, we have computed the pair-wise distance between all polygons and used the mode of all the distances to be used as the valie for ε. The same ε (=52 miles) was used for all the algorithms, namely—MC, CMC, STPC, VCoDA, and COT. In addition, we also used the same MinPts/ MinPolys (=3) for all the algorithms. The key reason for the difference obtained in the results for the other algorithms versus STPC is that STPC considers the spatio-temporal neighborhood of a polygon, and therefore sufficient density can be achieved even if one polygon continues to have the same properties across time, and does not have any spatial neighbors with the attributes. As a result STPC can detect clusters where the density of objects changes drastically from one time stamp to the next. The second key difference between the algorithms is that only STPC has the capability to incorporate the non-spatial attributes of the polygons. The other algorithms only look at the spatial attributes of the objects. Thus they are incapable of distinguishing between the drought and no-drought polygons for example.

Upon comparing the results of STPC with the clusters obtained by other algorithms mentioned above, we found that other algorithms discovered clusters that we indeed part of the clusters discovered by STPC. But none of the other algorithm successfully discovered complete clusters as found at STPC and shown in Fig. 11. For example, the trailing end of cluster 1 (C1) shown in Fig. 11 was not discovered by any other algorithm as with only one polygon at each time stamp the density condition is not satisfied. However, as STPC is based

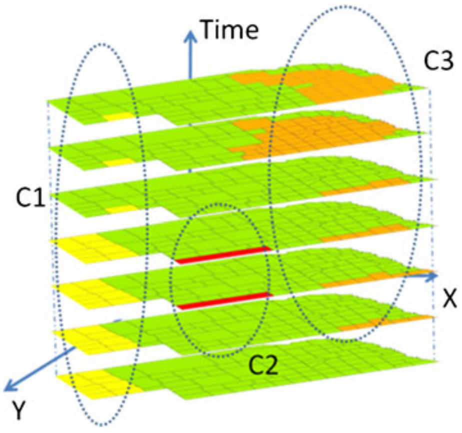


Fig. 11 Result of the STPC algorithm—the three smaller clusters are the drought clusters

on the spatio-temporal neighborhood of a polygon rather than only the spatial neighborhood of the polygon, even with a single polygon at each time stamp, the density condition is satisfied. Similarly, the third cluster discovered (C3 in Fig. 11) by STPC is divided into two or more clusters by every other algorithm because of the extreme density changes within the cluster from one time stamp to another. The comparison of the results produced by MC, CMC, VCoDA, STPC and COT is further demonstrated in Fig. 12 where the charts map the movements of the clusters across space and time. The charts show the number of polygons that belong to a cluster at a particular time stamp. These help to further visualize the density changes occurring within each cluster with the passage of time. VCoDA and COT algorithms discover clusters with constant density only, the MC and CMC algorithms are more robust to fluctuating densities, but even these algorithms are not as flexible as STPC which can capture sudden shifts most effectively.

Furthermore, upon visually inspecting the charts in Fig. 12 we can better describe the dynamics of the clusters produced by the various algorithms. For example, for the three clusters

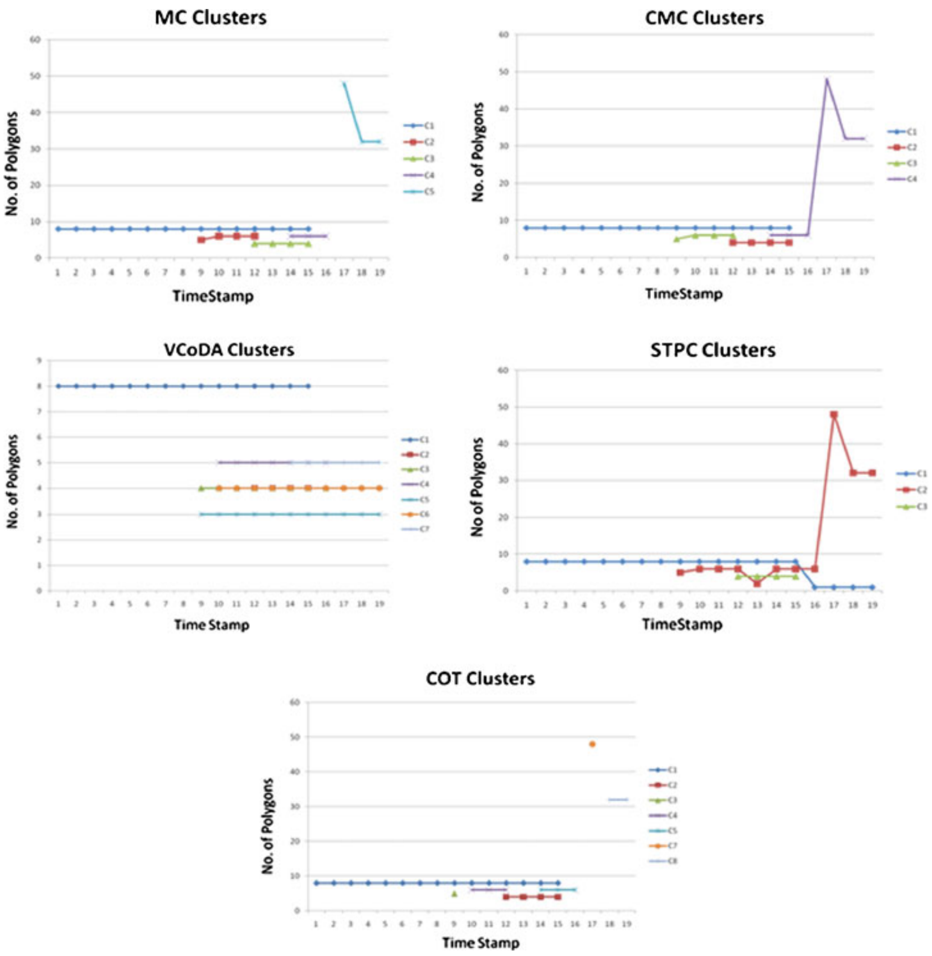


Fig. 12 Cluster densities across space and time as discovered by the MC, CMC, VCoDA, STPC, and COT algorithms for the NE drought dataset

tracked by STPC: (1) cluster C1 remains constant for some time and then contracts, (2) cluster C2 remains constant during its lifetime, and (3) cluster C3, after remaining constant for three weeks, contracts to only two polygons and then after expanding a little, suddenly expands across many polygons. This information on the cluster dynamics provides users with another level of insight for decision making. For example based on this dataset one may decide to track cluster C3 more closely to investigate the reasons for the contraction and the subsequent expansion, such as water usage and allocation, and corresponding mitigation policies.

4.2 Application of STPC on flu dataset

In order to show the robustness of our algorithm we have applied STPC to the swine flu dataset for the state of California. In this experiment we observe the properties of the two main parameters of STPC—namely ε and ω where ε dictates the possible extent of the spatial neighborhood and ω defines the possible extent of the temporal neighborhood.

Dataset description The dataset for this experiment comprises of the counites of the state of California on a weekly temporal scale from May 28, 2009 to July 16, 2009. Thus the total number of polygons in this dataset is $58\times8=464$ where 58 is the total number of counties in California, and 8 is the total number of weeks for which the data about the counties is collected. The non-spatial attributes for this experiment is the number of new swine flu cases discovered in each county during each time interval. However, we convert the dataset into categorical data by changing the number of new swine flu cases greater than one to 1, and number of new swine flu cases less than one to 0.

Results For this experiment we applied STPC on the California swine flu dataset using different values of ε and ω. The α and MinPoly parameters remain the same for all the experiments as we are using a categorical dataset in which there can be the strong uniformity case where the non-spatial distance between polygons can only be zero, i.e. $\alpha=0$. We first applied STPC using $\varepsilon=98$ miles, $\omega=1$, $\alpha=0$, and MinPoly=3. Here $\varepsilon=98$ miles is the mode of the pair-wise Hausdorff distance values between the polygons within the dataset. The result is that we discover 5 spatio-temporal clusters of new swine flu cases. The result is shown in Fig. 13. Next, we applied STPC using $\varepsilon=200$ miles, $\omega=1$, $\alpha=0$, and MinPoly=3. Here $\varepsilon=200$ miles is the median of the pair-wise Hausdorff distance values between the polygons within the dataset. The result is that we discover 1 spatio-temporal cluster of new swine flu cases. The result is shown in Fig. 14. Finally, we applied STPC using $\varepsilon=98$ miles, $\omega=5$, $\alpha=0$, and MinPoly=3. The result is that we discover 5 spatio-temporal clusters of new swine flu cases. The result is shown in Fig. 15.

Upon comparing the clusters shown in Figs. 13 and 14, we can see that more polygons with new swine flu cases are included in the spatio-temporal cluster shown in Fig. 14. Thus when using a smaller ε we can detect more clusters (Fig. 13), but there may be some polygons with the same non-spatial attributes as the polygons within the clusters that are not included within the cluster. On the other hand, upon comparing Figs. 14 and 15 we can see that the same number of polygons is included in the spatial-temporal clusters in both the cases, even though the number of clusters discovered in Fig. 15 is much more than the number of clusters discovered in Fig. 14. Finally, upon comparing the clusters shown in Figs. 13 and 15, we find the same number of clusters, but the total number of polygons included in the clusters in Fig. 15 is more than Fig. 13. This can especially be noted in cluster C1 in both the figures. In Fig. 15 we can see that cluster C1 is detected much earlier than in Fig. 13. This information was lost in the cluster discovered in Fig. 13 because of the parameter $\omega=1$ which stipulates spatio-temporal clusters with temporal contiguity.

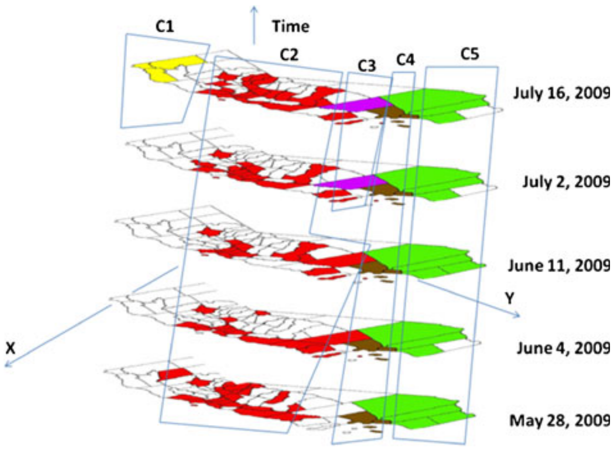


Fig. 13 Clusters discovered by STPC with ε=98 miles, ω=1, α=0, and MinPoly=3

Thus, if we use a small ε but increase the temporal extent for the spatio-temporal neighborhood of a polygon by selecting ω>1 there is a greater chance of including more polygons with the same non-spatial attributes within the spatio-temporal clusters discovered.

4.3 Application of STPC on crime dataset

Dataset description For this experiment we obtained the dataset from the chief of police of the city of Lincoln, NE, USA. The dataset consists of the time, date, type, and the location of the crime committed over 5 years between 2005 and 2009. The total number of crimes recorded is 153,404. The city of Lincoln has 186 census block groups. These form the base polygons for our experiments and are shown in Fig. 16. The temporal scale used for this set of experiments is daily. The total number of polygons within the dataset is thus 339,450 (=186×5×365). For each polygon the number of different types of crime that occur each day within the polygon are considered as the non-spatial attributes for the polygons.

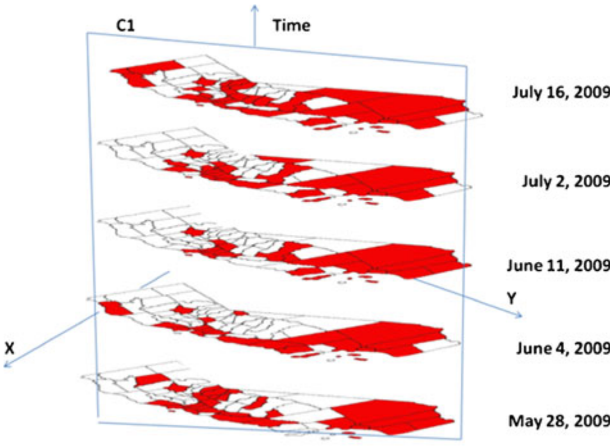


Fig. 14 Clusters discovered by STPC with ε=200 miles, ω=1, α=0, and MinPoly=3

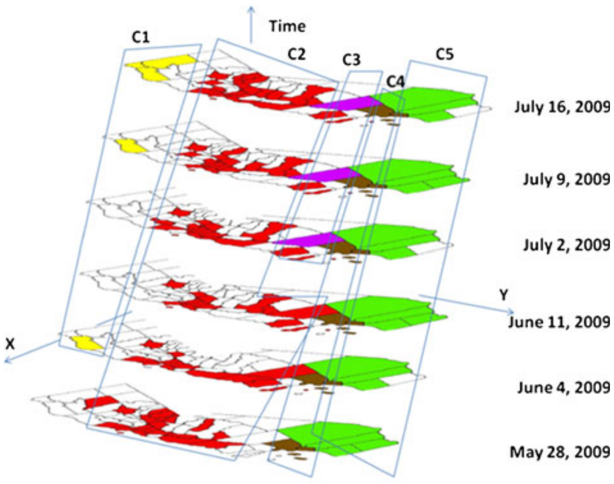


Fig. 15 Clusters discovered by STPC with ε=98 miles, ω=5, α=0, and MinPoly=3

Results In the following we show the analysis of the spatio-temporal clusters discovered by STPC for one type of crime—assaults. The total number of assault cases in the city of Lincoln from January 2005 until December 2009 is 22,314. The total number of clusters discovered by STPC using different parameter values is listed in Table 1 along with the average number of polygons per cluster and the range of polygons within the clusters.

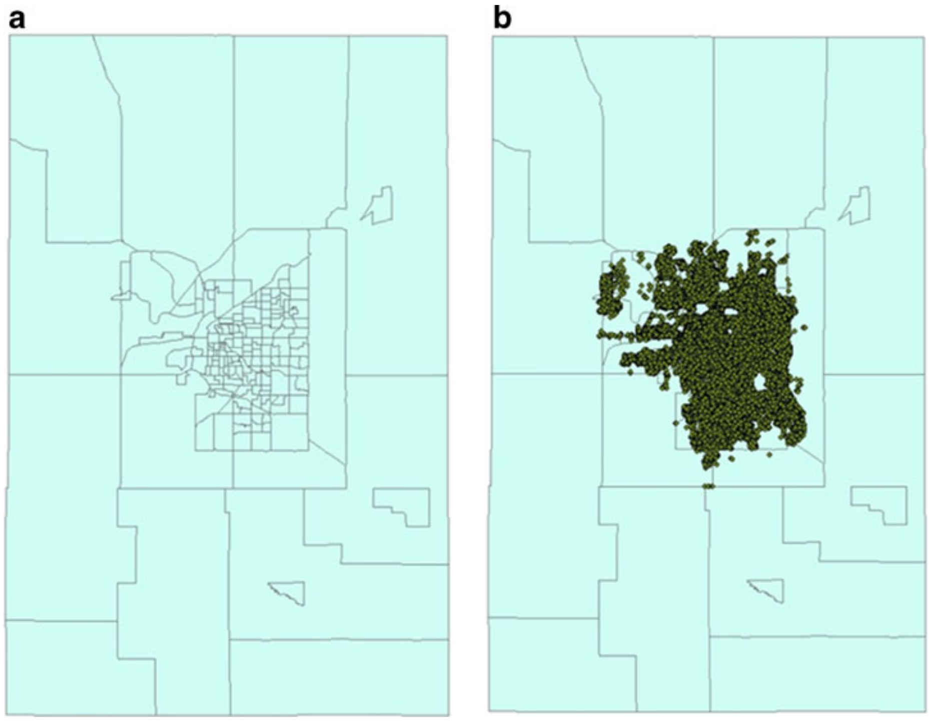


Fig. 16 **a** Census block groups in the city of Lincoln, NE **b** Crime locations for the years of 2005–2009 in the city of Lincoln, NE

Table 1 Assault clusters discovered by STPC using different parameter values

ε	ω	α	*MinPoly*	# clusters	Average # polygons per cluster	Range of polygons per cluster
0.6 miles	1	5	3	1,132	5.5	0.93
0.6 miles	1	5	10	123	15.0	0.78
0.6 miles	1	5	20	0	NA	NA
1.3 miles	1	5	3	1,216	8.7	0.96
1.3 miles	1	5	10	301	21.6	0.88
1.3 miles	1	5	20	121	29.8	0.73
1.3 miles	1	5	30	19	35.8	0.41

Table 1 shows that the input parameter of MinPoly has a big effect on the clustering results, as the smaller this number is, a a larger number of clusters will be detected. For example, when MinPoly=3, the number of clusters discovered for different ε values is 1,132 and 1,216. Whereas, when MinPoly=20, the number of clusters discovered are 0 and 121. This is because with a larger MinPoly we are forcing the core polygons to be closer to the center of the entire dataset with a large number of surrounding polygons. On the other hand, with a smaller MinPoly, a core polygon may also lie near the periphery of the polygonal dataset. Furthermore, with MinPoly=3, the density may be achieved only by taking into account the temporal neighbors of the polygons, without having any spatial neighbors. The number of clusters discovered is further augmented by the value of ε. A larger ε will allow distant polygons to be included within the same cluster, and therefore allow more number of clusters to be discovered. Thus when MinPoly=20, and ε=0.65 miles, no clusters are detected. This is because no core polygon is discovered that satisfies this criteria. However, when MinPoly=20, and ε=1.3 miles, 121 clusters are detected, as now there are polygons within the dataset that satisfy this criteria.

Further we closely inspect a few selected assault clusters. These are shown in Fig. 17. Each cluster is represented by a two-dimensional graph. The x-axis denotes the spatial dimension where each polygon is represented using its identification number. Thus the number 98 along the x-axis represents the space occupied by the polygon with the ID 98. The y-axis shows the temporal dimension of the spatio-temporal clusters. As the crime dataset is from the time period of January 2005 until December 2009 on a daily scale, each day is represented using a unique number. Thus, the number 263 on the y-axis refers to the day of September 20, 2005. To interpret each graph in Fig. 17, let us look at Cluster 4. This cluster expands from day 264 to day 269, and covers a total of 26 polygons (IDs: polygons with IDs 49, 56–57, 92, 94–99, 101–105, 107–108, 110–111, 114–117, 119, 122, 134). Furthermore, moving from day to 264 to 265, for example, we can see that only one polygon with ID 95 continues to experience cases of assault, whereas the other polygons (IDs: 56, 94, 97, 119, and 122) do not have any assault cases, instead new polygons (IDs: 99, 104–105, and 134) experience assault cases.

From Fig. 17 we can see that the clusters 4, 6, and 9 roughly spread across the same set of polygons, however occur a year apart from each other. As they are all assault clusters, we can decipher that these assault caess occurred on the same space during the same time each year for three consecutive years of 2005 to 2007. The polygons involved in these clusters are shown in Fig. 18 which shows the spatio-temporal Cluster 6. A closer inspection of these polygons shows that these polygons are along the heart of the downtown of Lincoln, NE, where most of the bars are located. Furthermore, the time period of mid-September until the beginning of October is when the fall semester

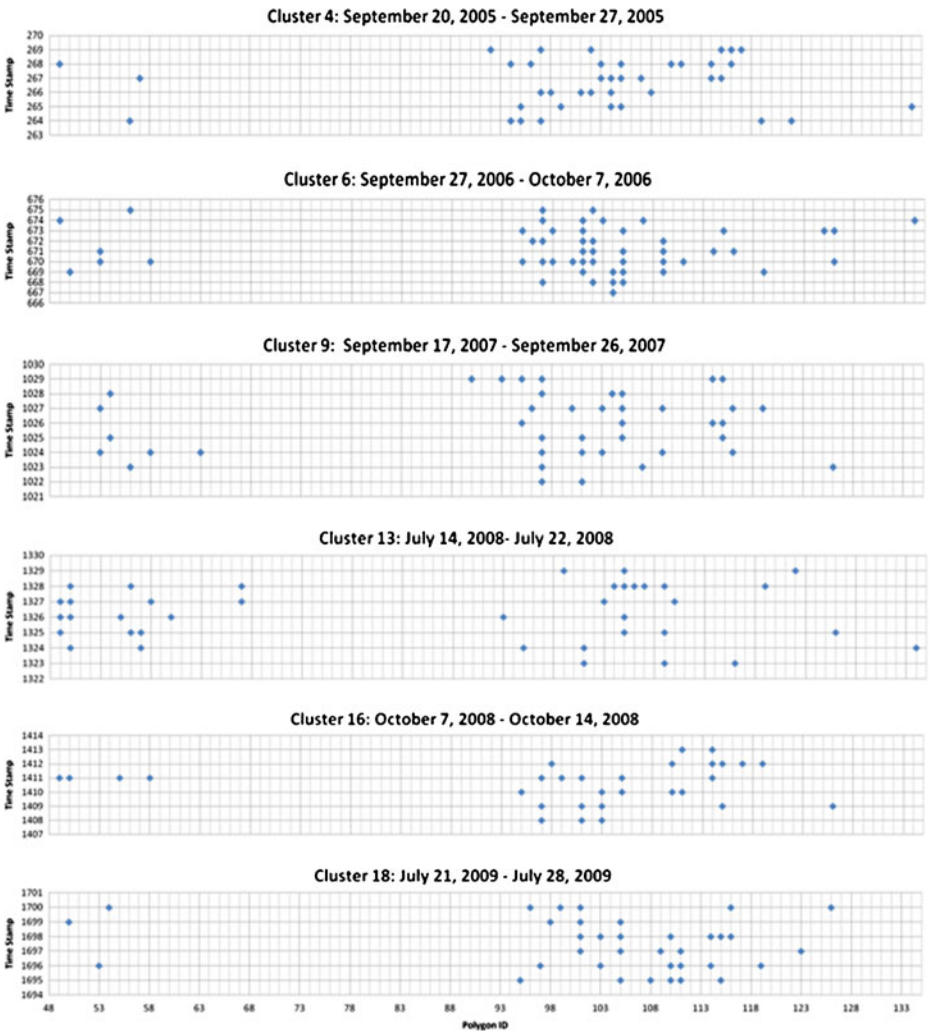


Fig. 17 Selected assault spatio-temporal clusters discovered by STPC using the parameter values: ε=1.3 miles, ω=1, α=5, and MinPoly=30, with space shown as one-dimension along the x-axis, and time along the y-axis

at the University of Nebraska-Lincoln which is located close to the downtown area is in full swing, and the weather is most favorable for people to be outdoors. It is interesting to note that the clusters seem to move to an earlier time period (from September to July) in the following 2 years (2008 and 2009).

5 Conclusion and future work

We have presented a spatio-temporal clustering algorithm called Moving Polygonal Clustering (STPC) that intrinsically incorporates time in the clustering process of spatio-temporal

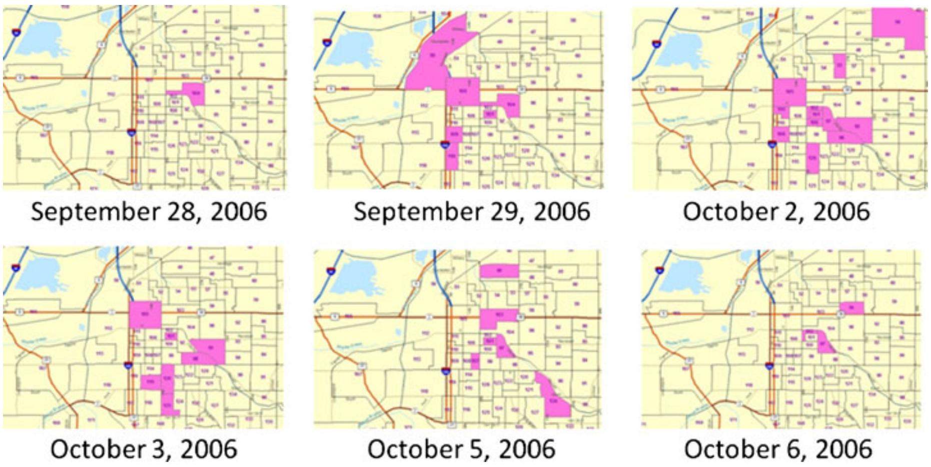


Fig. 18 The spatio-temporal cluster 6 in Fig. 17 spanning from September 28, 2006 until October 6, 2006

polygons. The STPC algorithm is based on the density-based clustering principle as this clustering paradigm naturally adapts to concepts such as spatial autocorrelation and Tobler's first law of geography. Furthermore, our algorithm treats time as a first class citizen, and thus gives equal importance to both space and time.

A unique property of our algorithm is that it naturally maintains the history of a cluster. Thus if a cluster fragments into two or more smaller clusters, our algorithm will track the fragmented clusters to the original cluster as long as the fragmented cluster has a temporal neighbor that belongs to the original cluster. Also, if a cluster suddenly contracts and then expands immediately, it will be divided into two or more smaller clusters by MC or CMC. STPC, on the other hand, will be able to capture the sudden movements within a cluster, and will thus be able to retain a unified structure.

As a part of future work, we will test the scalability of our algorithm by taking into consideration the drought dataset for the whole of Unites States of America. Further, we will perform experiments with different values for ω, i.e. change the extent of the temporal neighborhood to see the effect. One possible outcome would be the concatenation of two or more spatio-temporal clusters into one spatio-temporal cluster with a time gap in between where the cluster disappears and then re-appears within the same spatial vicinity. Also, currently STPC detects moving clusters across fixed space. We plan to extend STPC to consider moving polygons such as cells of human activities or viruses that could move spatially and change their shapes. In addition, we will extend our framework to detect movements of a cluster in other dimensions than space and time. For example, the varying intensity of drought within spatio-temporal drought clusters. Finally, we will also extend our approach to work with dynamic datasets with changing velocity across time, and real-time datasets.

We are also working on developing a stand-alone java 3D application to visualize the 3-D clusters in the three dimensional space. Next, we will develop a tool to be added in the ArcGIS toolbox that will allow users to form and visualize 3-D clusters within the ArcMap interface.

References

1. Aamodt G, Samuelsen SO, and Skrondal A (2006) A simulation study of three methods for detecting disease clusters. Int J Health Geogr, 5–15.
2. Ester M, Kriegel HP, Sander J, and Xu X (1996) A density-based algorithm for discovering clusters in large spatial databases with noise. Second Int Conf Knowl Discov Data Min, 226–231.
3. Hwang SY, Lee CM, and Lee CH (2008) Discovering moving clusters from spatio-temporal databases. Eighth Int Conf Intell Syst Des Appl, 111–114.
4. Jeung H, Yiu XZ, Jensen C, Shen H (2008) Discovery of convoys in trajectory databases. Proc VLDB Endow 1:1068–1080
5. Jeung H, Shen HT, and Zhou X (2008) Convoy queries in spatio-temporal databases. IEEE Int Conf Data Eng, ICDE'08, 1457–1459.
6. Joshi D, Samal AK, and Soh LK (2009) Density-based clustering of polygons. IEEE Symp Ser Comput Intell Data Min, 171–178.
7. Joshi D, Samal AK, and Soh LK (2009) A dissimilarity function for clustering geospatial polygons. 17th Int Conf Adv Geogr Inform Syst (ACM SIGSPATIAL GIS 2009), 384–387.
8. Joshi D, Soh LK, Samal A, Zhang J (2011) A dissimilarity function for geospatial polygons, technical report TR-UNL-CSE-2011-0010. Department of Computer Science and Engineering, University of Nebraska, Lincoln
9. Kalnis P, Mamoulis N and Bakiras S (2005) On discovering moving clusters in spatio-temporal data. Symp Spatial Temporal Databases, 364–381.
10. Lai C and Nguyen NT (2004) Predicting density-based spatial clusters over time. In Proc Fourth IEEE Int Conf Data Min.
11. Neill DB, Moore AW, Sabhnani MR and Daniel K (2005) Detection of emerging space-timeclusters. In Proc 11th ACM SIGKDD Intl Conf on Knowledge Discov Data Min.
12. Stolorz P et al (1995) Fast spatio-temporal data mining of large geophysical datasets. In Proc First Int Conf Data Min KDD—95, 300–305.
13. Tobler W (1970) A computer movie simulating urban growth in the Detroit region. Econ Geogr, 234–240.
14. Yoon H and Shahabi C (2009) Accurate discovery of valid convoys from moving object trajectories. IEEE Int Conf Data Min Workshops, 636–643.
15. Zhang P, Huang Y, Shekhar S and Kumar V (2003) Exploiting spatial autocorrelation to efficiently process correlation-based similarity queries. In Proc of the 8th Intl Symp on Spatial and Temporal Databases.

Deepti Joshi received the Bachelor of Art in English Literature from the Delhi University, Masters of Science in Applied Computer Science from the Northwest Missouri State University, and a Ph.D. degree in Computer Science from the University of Nebraska-Lincoln. She is currently an assistant professor at The Citadel. Her research interests include geospatial computing, data mining, and Volunteered Geographic Information systems.

Ashok Samal received the Bachelor of Technology degree in computer science from the Indian Institute of Technology, Kanpur, India, in 1983, and the Ph.D. degree in computer science from the University of Utah, Salt Lake City. He is a Professor with the Department of Computer Science and Engineering at the University of Nebraska. His current research interests include geospatial computing, data mining, image analysis, and computer science education.

Leen-Kiat Soh received the BS (with highest distinction), MS, and PhD (with honors) degrees in electrical engineering from the University of Kansas. He is currently an associate professor in the Department of Computer Science and Engineering at the University of Nebraska. His primary research interests are in multiagent systems and intelligent agents, especially in coalition formation and multiagent learning. He has applied his research to computer-aided education, intelligent decision support, and distributed GIS. He is a member of the IEEE, the ACM, and the AAAI.

Copyright of GeoInformatica is the property of Springer Science & Business Media B.V. and its content may not be copied or emailed to multiple sites or posted to a listserv without the copyright holder's express written permission. However, users may print, download, or email articles for individual use.