Detect Irregularly Shaped Spatio-Temporal Clusters For Decision Support

Weishan Dong, Xin Zhang, Zhongbo Jiang, Wei Sun, Lexing Xie, Arun Hampapur
IBM Research
{dongweis,zxin,jiangzb,weisun}@cn.ibm.com, {xlx,arunh}@us.ibm.com

Abstract—**Many real-world applications call for the use of detecting *unusual* clusters (abnormal phenomena or significant change) from spatio-temporal data for decision support, e.g., in disease surveillance systems and crime monitoring systems. More accurate detection can offer stronger decision support to enable more effective early warning and efficient resource allocation. Many spatial/spatio-temporal clustering approaches have been designed to detect *significantly unusual* clusters for decision support. In this paper, we focus on more accurately detecting irregularly shaped unusual clusters for point processes and propose a novel approach named EvoGridStatistic. The original problem is mathematically converted to an optimization problem and solved by estimation of distribution algorithm (EDA), which is a powerful global optimization tool. We also propose a prospective spatio-temporal cluster detection approach for surveillance purposes, named EvoGridStatistic-Pro. Experiments verify the effectiveness and efficiency of EvoGridStatistic-Pro over previous approaches. The scalability of our approach is also significantly better than previous ones, which enables EvoGridStatistic-Pro to apply to very large data sets in real-world application systems.**

I. Introduction

In recent years, there has been increasing interest in spatial and spatio-temporal data analysis. Clustering approaches are often adopted for discovering abnormal spatial/spatio-temporal patterns for decision support. Many approaches have been proposed to detect regularly shaped clusters with their significance [1]-[8]. However, in real-world applications, clusters are often irregularly shaped. In these cases, the performance of previous approaches can be significantly reduced that usually extra large areas are identified compared with the ground truth. On the other hand, being able to more accurately detect clusters when they are irregularly shaped can bring significant efficiency in terms of fast emergency respond, precise resource allocation and distribution, and correct understanding of the situation. In this paper, we focus on solving this problem and propose a novel approach to more accurately detect irregularly shaped spatio-temporal clusters.

The main idea of existing spatial/spatio-temporal cluster detection approaches can be summarized as follows. By performing spatial clustering and statistical hypothesis testing, areas with *unusually* high density of cases relative to underlying population (baseline information) can be identified. Here cases refer to reported disease cases or crime activities, etc. Usually the clusters are detected through maximizing a likelihood function, and a p-value is calculated to justify the statistical significance. If the p-value is smaller than a significance level (e.g., 0.05), then it is determined that the cluster can rarely be observed and an alarm will be triggered. When extending spatial clustering to spatio-temporal clustering, another dimension of time is added into the analysis. Spatio-temporal clustering strives to find clusters in a 3D space. It can serve as a retrospective analysis tool to analyze historical data, and can also be used for prospective surveillance in detecting events such as outbreaks of infectious disease at an early stage. In many existing approaches, the scanning window used to identify spatial clusters is restricted to regular shapes like circles, ellipses or rectangles, for the purpose of computational convenience. In spatio-temporal analysis, a regular cylindric window is often used instead. The differences between above spatial/spatio-temporal clustering and traditional clustering analysis are obvious. In classical data mining, the main goal of clustering is to maximize both the homogeneity within each cluster and the heterogeneity among different clusters. Traditional clustering techniques, some of which have also been applied to spatial/spatiao-temporal data, such as DBScan [9], STING [10], and their variants, usually do not consider baseline information. They can discover areas with dense samples but the significance of detected clusters is neglected. Therefore, when used for discovering *unusual* patterns, traditional clustering techniques have limitations.

To remedy the shortcomings of previous spatial/spatio-temporal clustering approaches only using regular window, in this paper, we propose a novel approach named *evolutionary grid statistic* (EvoGridStatistic) to detect irregularly shaped spatial clusters, and also propose EvoGridStatistic-Pro to extend to prospective spatio-temporal analysis. In EvoGridStatistic, cluster detection is converted to a binary string coded combinatorial optimization problem. A grid map is constructed and overlaid on input data. A binary string indicating a set of connected grid cells maximizing an evaluation function is figured out by estimation of distribution algorithm (EDA) [11]. Each connected region indicates a candidate cluster, which can asymptotically describe any irregular shape. Grid-based clustering algorithms which only find dense areas have already been proposed (e.g., STING [10]). However, as mentioned above, when the objective is to find *unusual* clusters, areas with dense points do not guarantee themselves to be identified, since the significance is also dependent on the baseline information. The underlying clustering problem is totally different. Compared with using regularly shaped scanning window, additional flexibility introduced by grid map also further enlarges the search space and makes the

978-1-4577-0574-8/11/$26.00 ©2011 IEEE

optimization problem more difficult. Therefore, a powerful global optimization tool, EDA, is adopted. EDA has been successfully applied to many hard optimization problems, and has been proved to be superior to traditional evolutionary algorithms for optimization tasks such as genetic algorithms in many cases [11][12]. We also propose EvoGridStatistic-Pro to extend to spatio-temporal analysis. The spatio-temporal point data process is segmented into a sequence of chunks by their arrival time. EvoGridStatistic is used to compare the newly arrived chunk with historical chunk, and detect whether significant clusters exist in the newly arrived chunk. Experiments show the efficiency and the effectiveness of EvoGridStatistic-Pro in its ability to detect irregularly shaped spatio-temporal clusters. EvoGridStatistic(-Pro) has advantages over traditional approaches in the following aspects: 1) EvoGridStatistic(-Pro) can more accurately detect a cluster when it is irregularly shaped. 2) The computation time of EvoGridStatistic(-Pro) can be easily controlled by user-defined grid map size despite of the amount of data. 3) Compared with previous approaches that can also detect arbitrary shaped clusters (e.g., [13]), the parameters of EvoGridStatistic(-Pro) have clear physical implications and are thus much easier to set.

The remainder of this paper is organized as follows. In Section II, related work is discussed. In Section III, we propose EvoGridStatistic and its prospective spatio-temporal extension, EvoGridStatistic-Pro. In Section IV, experimental studies are given. Conclusions are drawn in Section V.

II. Related Work

The spatial scan statistic proposed by Kulldorff [1] has become one of the most widely used approaches to detect unusual clusters. In this approach, exhaustive search is used to test every potential area by moving a window across the entire data set. For each candidate area, a likelihood is calculated. The area with the maximum likelihood is regarded as the most likely cluster. Spatio-temporal clustering methods (e.g., [2][3]) have also been developed based on the central idea of purely spatial scan statistic. A free software SaTScan [14] has been developed implementing these approaches. We will include SaTScan in experiments as benchmark. Many times, because of computational reasons, several approaches including the spatial scan statistic [1][4][7][8] restrict their scanning window to regular shapes like circles, ellipses or rectangles to reduce the size of search space. A cylindric window is often used when scanning for spatio-temporal clusters [2][3]. Another methodology is to segment the data into chunks by time, and then perform clustering based on the chunks to identify abnormal changes [5][13].

The regularly shaped window is a natural choice covering many real-world situations, especially when the true cluster is compact. However, such limitations apparently reduce the capability to detect irregularly shaped clusters in complex scenarios. Many attempts have already been made to solve this problem. In general, prior work can be categorized into two groups. The first one deals with zone data, which means the data is spatially aggregated to zones whose boundaries are predefined and precise locations of each data case are unavailable. In this case, clusters refer to a set of connected zones. Approaches that fall in this group include upper level set scan statistic [15], simulated annealing scan [16], and flexible spatial and space-time scan statistic [17][18]. The second group deals with point data, meaning that for each case a distinct point location is assigned. EvoGridStatistic(-Pro) aims to deal with such kind of data. Because of the absence of zone boundary, it is harder to decide clusters' scope. The search space is enlarged compared with the previous group. Therefore, approaches that fall into this group are mostly based on heuristic search. For instance, Sahajpal et. al [19] proposed a genetic algorithm (GA) to find clusters shaped as intersections of circles, and Iyengar [20] used a heuristic approach to find spatio-temporal clusters shaped as square pyramids. However, these approaches still pose a little strict limitation to the cluster shape. For instance, an irregularly shaped cluster such as L-shaped or ring-shaped will be hard to accurately detect for all these approaches. Another approach belonging to this group is the support vector machine (SVM) based clustering approach [13], which detects arbitrarily shaped cluster. However, it is well known in the SVM community that the kernel width and the margin parameters significantly influence results. Cluster detection is unsupervised also make it difficult to find optimal settings for the parameters. Another common problem of all previous approaches is the computational complexity. Let N denote the total amount of data points, the spatial scan statistic and its variants cost $O(N^2)$ to identify the most likely cluster [14]. The worst-case computation time of SVM based approach is also $O(N^2)$ when comparing 2 data chunks [13].

III. Method

A. EvoGridStatistic

We begin by introducing *evolutionary grid statistic* (EvoGridStatistic) for spatial clustering. The main idea is to overlay a grid map on data, and then choose to mark some of the grid cells to best describe clusters. Since the shape of connected marked cells can be flexible, it can asymptotically describe irregularly shaped cluster. Whether a cell is marked or not is decided by maximizing a fitness function, which is used to evaluate clusters. The corresponding maximization problem is solved by EDA, whose details will be introduced later. The main steps of EvoGridStatistic are shown in Fig. 1. Note that all indices start from zero. The scan order of a grid map is row-based, similar to index a matrix.

Fig. 2 demonstrates an instance of G, which can help understand the main idea of EvoGridStatistic. The marked cells are colored in yellow, while the rest are unmarked. An instance of G can be regarded as a 2D binary image with $m \times n$ pixels. A binary string can be encoded to represent the 2D image following (1) in Fig. 1. We can directly use connected component labeling algorithms in image processing field to identify all the connected regions of marked cells for a given string. Function f in Step 4 of Fig. 1 is the evaluation

```
EvoGridStatistic
1) Initialize $C$ as an empty set: $C \leftarrow \emptyset$.
2) Overlay a $m \times n$ grid map $G$ covering the entire data set. Point
   $O$ defines the upper left corner of $G$. $G(i, j)$ denotes a grid cell
   on the $i^{th}$ row $j^{th}$ column $(0 \leq i < m, 0 \leq j < n)$. Each cell
   of $G$ has two possible states, marked and unmarked.
3) Encode $G$ by a binary string $\vec{X}$ with length $m \times n$. Let $X_k$ denote
   the $k^{th}$ bit of $\vec{X}$ $(0 \leq k < m \times n)$.
   $X_{i \times n + j} = \begin{cases} 1, & \text{if } G(i,j) \text{ is marked during EDA evolution;} \\ 0, & \text{otherwise.} \end{cases}$   (1)
4) Call EDA to find a string $\hat{\vec{X}}$ maximizing a fitness function $f$.
   (Details see Sect. III-C.)
5) Decode $\hat{\vec{X}}$ by reversing (1) to extract connected regions of
   identified candidate clusters. Calculate p-value for each cluster.
6) $C \leftarrow C \bigcup$ clusters with p-value $\leq \theta$, where $\theta$ is a significance
   level threshold (e.g., 0.05).
7) Output $C$.
```

Fig. 1. Main steps of EvoGridStatistic.

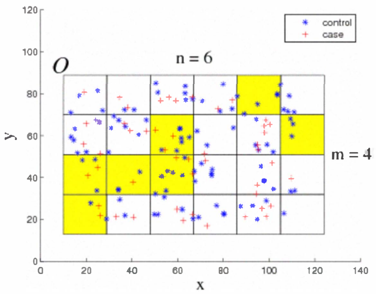


Fig. 2. Demonstration of a 4 × 6 grid map overlayed on a sample data set. O denotes the upper left corner point. Marked cells are shown in yellow. The corresponding encoded string is 000010001001111000100000 according to (1). There are 3 connected regions of marked cells if considering 4-connectedness.

function of candidate clusters, whose details will be given in the next section.

B. Evaluating Clusters

Kulldorff has proposed a likelihood function for clusters based on Bernoulli model [1]. The likelihood function is adopted in EvoGridStatistic to evaluate clusters. Under the Bernoulli model, the baseline data point is called "control", and the newly observed data point is called "case". Let n_Z denote the number of cases in area Z, n_G the total number of cases in the entire data set, N_Z the total number of cases and controls in area Z, and N_G the total number of cases and controls in the entire data set. The likelihood function of Z with unusually high density is defined as follows:

$$L(Z) = \left(\frac{n_Z}{N_Z}\right)^{n_Z} \cdot \left(1 - \frac{n_Z}{N_Z}\right)^{N_Z - n_z} \cdot \left(\frac{n_G - n_Z}{N_G - N_Z}\right)^{n_G - n_Z} \cdot \left(1 - \frac{n_G - n_Z}{N_G - N_Z}\right)^{(N_G - N_Z) - (n_G - n_Z)} \qquad (2)$$

where $\frac{n_Z}{N_Z} > \frac{n_G - n_Z}{N_G - N_Z}$.

Now we define the fitness function f in EvoGridStatistic. We use the same notation in Fig. 1 that let $\vec{X}$ denote an encoded binary string. Let $R_{\vec{X}}$ denote the set of all connected regions of $\vec{X}$. Each $Z \in R_{\vec{X}}$ represents an area, i.e., a candidate cluster. It is easy to calculate n_Z and N_Z for a given Z, and we can use (2) to calculate its likelihood. The fitness function used to evaluate $\vec{X}$ is defined as:

$$f(\vec{X}) = \sum_{Z \in R_{\vec{X}}, \frac{n_Z}{N_Z} > \frac{n_G - n_Z}{N_G - N_Z}} \left(\log L(Z) - \alpha \cdot E_Z\right) , \qquad (3)$$

where E_Z denotes the amount of "empty" cells of Z, and α is a penalization factor. Here "empty" means no case or control point within a cell. From (2), we also see that the likelihood remains unchanged no matter how many empty cells are included in Z. The term $(-\alpha \cdot E_Z)$ is used to penalize clusters with unnecessarily large size. Coefficient α controls the appearance of unnecessary empty cells. By maximizing f, EvoGridStatistic can locate potential clusters. In the hypothesis testing stage, the significance level of clusters is calculated as follows. Because the distribution of $L(Z)$ is unknown, Monte Carlo simulation is used to calculate p-values. Replicated data sets are generated randomly under the null hypothesis of no clustering, and $L(Z)$ of each candidate cluster is calculated for both the original input data set and all the replications. If the $L(Z)$ of a cluster detected in the original input data set is among the highest 5%, the cluster is significant at 0.05 level, i.e., p-value $\leq$ 0.05.

C. Optimization via EDA

So far, the original cluster detection problem has been converted to an optimization problem. The objective now becomes to find a binary string maximizing (3). We use estimation of distribution algorithm (EDA) [11][12] to solve this problem. In EDA, a population of individuals is evolved over generations. Each individual represents a potential solution to the problem, whose goodness is measured by a fitness function. Initial population is usually randomly generated. During evolution, a probabilistic model is built based on selected "good" individuals. This model describes the global distribution of promising solutions. New individuals are generated by sampling from this model. Let n_{pop} denote the population size, and n_{sel} the selected size. A general framework of EDA is shown in Fig. 3. Each iteration refers to one generation of evolution. The iteration usually stops when the population converges or a maximum number of generations is reached.

In EvoGridStatistic, an individual in EDA is a binary string. The genes of an individual refer to the bits of a string. Fitness of an individual is calculated by (3). EDA is used to find a string of 0/1 combinations maximizing (3) and thus determine which cells are marked. There are EDAs employing different probabilistic models. In our current implementation, we use the univariate marginal distribution algorithm (UMDA) [11][12] to solve the $(m \times n)$-dimensional problem. UMDA adopts the simplest form of probabilistic model that assumes all the genes of an individual are independent with each other.

```
EDA
1) Initialize a population $\mathcal{P}$ by generating $n_{pop}$ individuals.
2) Repeat until a stopping criterion is met.
    a) Select $n_{sel} \leq n_{pop}$ individuals from $\mathcal{P}$ according to their fitness values.
    b) $p(\vec{x}) \leftarrow$ Estimate a probability distribution function from the selected individuals.
    c) $\mathcal{P}' \leftarrow$ Sample new individuals from $p(\vec{x})$.
    d) Combine the best individual(s) in $\mathcal{P}$ and $\mathcal{P}'$ to create new $\mathcal{P}$.
3) Output the best individual.
```

Fig. 3. A general EDA framework.

```
EvoGridStatistic-Pro
Repeat when $D_t$ $(t \geq 1)$ arrives.
    1) Treat $D_t$ as case data and $D_{t-1}$ as control data. Call EvoGridStatistic.
    2) Trigger alarm if output of EvoGridStatistic $C \neq \emptyset$.
    3) $t \leftarrow t + 1$.
```

Fig. 4. EvoGridStatistic-Pro.

Therefore, the joint probability distribution is the product of the marginal probabilities of all the genes as:

$$p(\vec{x}) = \prod_{i=1}^{m \times n} p(x_i) \ . \tag{4}$$

The probability that gene i has value x_i is estimated as:

$$p(x_i) = \frac{\sum_{j=1}^{n_{sel}} \delta_j(X_i = x_i)}{n_{sel}} \ , \tag{5}$$

where $\vec{X}$ denotes a selected individual, and $\delta_j(X_i = x_i) = 1$ if the i^{th} gene of the j^{th} selected individual is x_i and 0 otherwise. For our binary coded problem, x_i only has 2 possible values, 1 and 0. Therefore, building the probability model and sampling new individuals is very efficient in EvoGridStatistic.

D. Extending to Prospective Spatio-Temporal Analysis

We can easily extend EvoGridStatistic to prospective spatio-temporal analysis. The spatio-temporal data is segmented into a sequence of chunks by their arrival time. A historical chunk is treated as "control". A newly arrived data chunk is regarded as "case". The "case" chunk can be compared with the "control" chunk to determine whether significant clusters exist in the "case" chunk. By doing so, we are able to give early warning when spatio-temporal clusters emerge, expand or move. The resulting algorithm is called EvoGridStatistic-Pro. Let D_t denote a data chunk arrived at time t. The flow of EvoGridStatistic-Pro is shown in Fig. 4. In EvoGridStatistic-Pro, we compare the newly arrived data chunk with only the last chunk to detect anomalies.

E. Discussion

1) Parameters: The size of grid cells is defined by m and n. Larger m and n can lead to finer spatial granularity, but can also make the optimization problem grow larger. Larger n_{pop} and n_{sel} will also be needed to guarantee global convergence of EDA when $m \times n$ is large. To our experience, $n_{pop} = 1000$ and $n_{sel} = 500$ work well enough when $m \times n \leq 2 \times 10^3$. Such settings are also commonly used in previous studies of EDA on many real-world problems [11]. The effectiveness of α depends on the range of $\log L(Z)$. In most cases so far we encountered, the value of $\log L(Z)$ varies from order 10^0 to 10^2. Therefore $\alpha = 0.01$ can be a reasonable choice, and it works well in all our experiments. One explanation of α is that it helps to find the largest connected component that is statistically significant, even if there are a few empty cells, hence a small value suffice. For efficiency, 4-connectedness is used when finding connected regions of marked cells.

2) Computation Time: The primary computation in EvoGridStatistic(-Pro) is the EDA evolution. The dimensionality of the optimization problem, $m \times n$, primarily determines the convergence speed of EDA. It also determines the computation time in finding connected regions for a string. EvoGridStatistic(-Pro) benefits from the grid map that its computation time only rely on the number of grid cells, i.e., $m \times n$. As a result, the overall computation time of EvoGridStatistic (one iteration in EvoGridStatistic-Pro) can be completely controlled by m and n, which is independent of the amount of data. This is an obvious advantage when dealing with very large data sets. Since EDA is a randomized search algorithm, so far, theoretical analysis on computational complexity of EDA is still an open problem [11]. Therefore, it is hard to formulate $O(\cdot)$ bound for EvoGridStatistic(-Pro). However, as can be seen in experiments, for the particular cluster detection problem, EvoGridStatistic(-Pro) can usually generate satisfactory results within quite acceptable time.

IV. Experiments

We will evaluate EvoGridStatistic-Pro on several data sets with the following settings. Through all experiments, we set $\alpha = 0.01$, $n_{pop} = 1000$ and $n_{sel} = 500$. Truncation selection for UMDA is adopted, that is, the n_{sel} best individuals are selected to build the probabilistic model. Elitism strategy is also used so that in every generation, the best individual survives directly into the next generation. In the sampling stage, $(n_{pop} - 1)$ new individuals are sampled to construct $\mathcal{P}'$. The new population $\mathcal{P}$ is constructed by union of the best individual and $\mathcal{P}'$. Iteration of UMDA stops when the best fitness value found remains unchanged for 10 consecutive generations. In Monte Carlo simulation, 999 runs are performed to calculate p-value with precision 0.001. Only clusters with p-value $\leq \theta = 0.05$ are reported. All experiments are done on a computer with 2.53GHz CPU and 3G memory.

A. Test of 3 Scenarios

3 data sets, "emerging", "expanding" and "moving" are used here (see Fig. 5). In all the 3 data sets, the data points' x-y coordinates are spatially distributed between $[0, 20]$. The z axis represents time (week) with range $[1, 7]$. We use fixed O=$(0, 20)$, square cell with width=1, and m=n=20 for

EvoGridStatistic-Pro on all 3 data sets, without further tuning. Using the framework of EvoGridStatistic-Pro, we can also substitute EvoGridStatistic in the iteration with the spatial scan statistic using circular window [1] and using elliptic window [6] with 3 available levels (strong, medium, none) of non-compactness penalty on the elliptic shape. The resulting 4 algorithms serve as benchmarks of EvoGridStatistic-Pro.

Fig. 6 shows the results of EvoGridStatistic-Pro and the benchmarks (denoted by SaTScan hereafter). In the figure, the red lines outline the true clusters. The yellow polygons denote the results of EvoGridStatistic-Pro. In all the tests, EvoGridStatistic-Pro and SaTScan detect the clusters and trigger alarms at exactly the same time frame. For the "emerging" data set, a new circular cluster emerges away from an existing cluster since week 3. However, only 15 points are not significant at the 0.05 level. Until week 4, the emerging pattern can be clearly observed and detected immediately by both EvoGridStatistic-Pro and SaTScan. EvoGridStatistic-Pro recognizes a cluster very close to the ground truth. Whereas, SaTScan, using either circular or elliptic window, tends to cover unnecessarily large area where rare data points are observed. For the "expanding" data set, the cluster expansion starts in week 4. The true anomaly is a ring-shaped cluster with large scale. SaTScan fails to accurately detect the entire area of the true cluster. EvoGridStatistic-Pro again recognizes the true cluster by approximating the shape with union of cells. For the "moving" data set, the true anomaly is the L shape, which starts to appear in week 5. EvoGridStatistic-Pro almost perfectly detects the anomaly. However such a cluster is too hard to accurately detect by circular or elliptic window. In a word, the simple prospective framework of EvoGridStatistic-Pro performs very well in terms of quickly detecting a cluster once it becomes significant. EvoGridStatistic also exhibits overwhelming better performance over SaTScan in its ability to detect irregularly shaped clusters. Current implementation of EvoGridStatistic-Pro is a single-threaded sequential Java program. In the experiments, the CPU time of a typical run of an evolution costs less than 3 seconds and the algorithm usually converges in around 20 iterations.

B. Scalability Test

To test the scalability of EvoGridStatistic-Pro and SaTScan, we generate 4 synthetic large data sets: D1K contains 10^3 cases and 10^3 controls; D10K contains 10^4 cases and 10^4 controls; D50K contains 5×10^4 cases and 5×10^4 controls; D100K contains 10^5 cases and 10^5 controls. The scalability of EvoGridStatistic-Pro is decided by EvoGridStatistic, so EvoGridStatistic can be tested to draw the conclusion. SaTScan using circular window and elliptic window without non-compactness penalty is also involved in comparison. We use the same parameter settings as in previous experiments of EvoGridStatistic, but with varying grid cell sizes. On each data set, square cell with width=0.5, 1.0, 1.5 are tested. For each cell size setting, we make sure the grid map covers the entire data set, thus m and n are determined accordingly, i.e., m=n=7, 10, 20. Fig. 7 demonstrates their CPU time increments as the sample size increases. Fig. 8 shows the CPU time increment of EvoGridStatistic against the dimensionality ($m \times n$) of the optimization problem. Because SaTScan fails to run on D100K due to out-of-memory error, and SaTScan using elliptic window can not finish in acceptable time on D50K, these results are not shown in the figure.

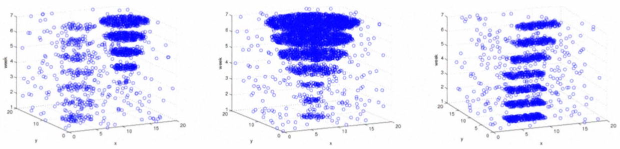

(a) "Emerging": starts from week 3. (b) "Expanding": starts from week 4. (c) "Moving": starts from week 5.

Fig. 5. The 3 data sets.

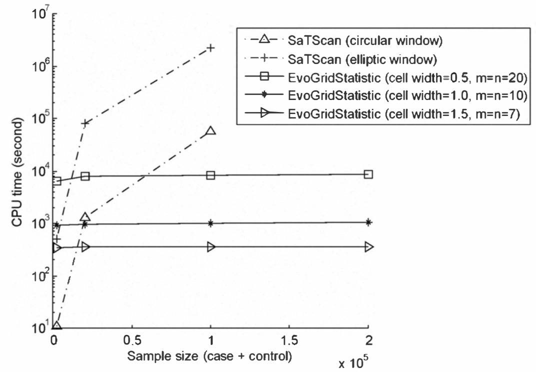


Fig. 7. Comparison of CPU time against sample size (D1K: 2×10^3, D10K: 2×10^4, D50K: 10^5, D100K: 2×10^5) between SaTScan and EvoGridStatistic.

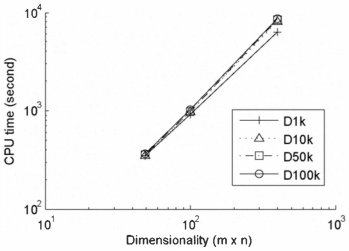
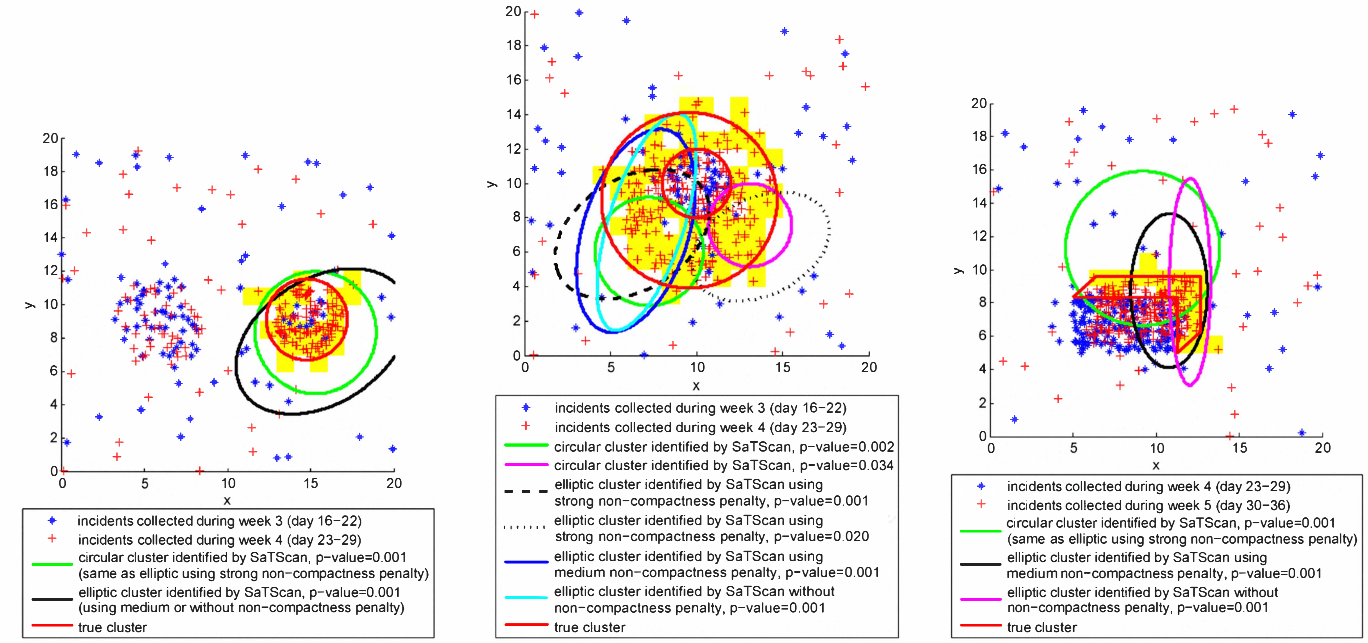


Fig. 8. EvoGridStatistic's CPU time against dimensionality ($m \times n$ = 7×7, 10×10 and 20×20) on the 4 data sets.

From Fig. 7 we can see that the computational time of SaTScan increases fast as the data set size increases since it is of the order $O(N^2)$. We can also expect SVM based methods to show similar scalability since its computational complexity is of the same order (see Section II). Contrarily, EvoGridStatistic exhibits good scalability that its computation time keeps nearly constant as the data set size increases as long as the dimensionality of the optimization problem ($m \times n$) is fixed, which verifies our previous analysis. We can also see from Fig. 8 that even when the dimensionality increases, when using fixed n_{pop} and n_{sel} in EDA as we did in experiments, EvoGridStatistic's computation time also grows approximately linear.



(a) "Emerging": detected in week 4

(b) "Expanding": detected in week 4

(c) "Moving": detected in week 5

Fig. 6. Results on the 3 data sets. On each data set, EvoGridStatistic-Pro detects one cluster (connected square cells in yellow) with p-value=0.001. Compared with SaTScan results shown in circles and ellipses, EvoGridStatistic-Pro identifies the true clusters more accurately.

V. Conclusions

In this paper, we propose EvoGridStatistic and its spatio-temporal extension, EvoGridStatistic-Pro, to detect irregularly shaped usual clusters for spatio-temporal point data. They can be effectively applied to real-world scenarios where very irregularly shaped spatial/spatio-temporal clusters easily appear. However, traditional approaches using regular shaped scanning windows usually fail in the same scenarios. EvoGridStatistic(-Pro) also has two significant advantages over previous approaches designed to detect arbitrarily irregular shaped clusters: First, the computation time is controllable by the number of grid cells, and independent with the amount of data. Second, the parameters in EvoGridStatistic(-Pro) have clear physical implications and are easy to set, even though the nature of cluster detection is unsupervised.

Acknowledgement

The authors are grateful to Wei Chang for his sharing the 3 data sets.

References

[1] M. Kulldorff, "A spatial scan statistic," *Communications in Statistics: Theory and Methods*, vol. 26, pp. 1481-1496, 1997.

[2] M. Kulldorff, W. Athas, F. Feuer, B. Miller, and C. Key, "Evaluating cluster alarms: A space-time scan statistic and brain cancer in Los Alamos," *Am J Public Health*, vol. 88, pp. 1377-1380, 1998.

[3] M. Kulldorff, "Prospective time-periodic geographical disease surveillance using a scan statistic," *Journal of the Royal Statistical Society: Series A (Statistics in Society)*, vol. 164, pp. 61-72, 2001.

[4] D.B. Neill and A. W. Moore, "Rapid detection of significant spatial clusters," *ACM SIGKDD*, pp. 256-265, 2004.

[5] D.B. Neill, A. W. Moore, M. Sabhnani, and K. Daniel, "Detection of emerging space-time clusters," *ACM SIGKDD*, pp. 218-227, 2005.

[6] M. Kulldorff, L. Huang, L. Pickle, L. Duczmal, "An elliptic spatial scan statistic," *Statistics in Medicine*, vol. 25, pp. 3929-3943, 2006.

[7] N. Levine. *CrimeStat III: A Spatial Statistics Program for the Analysis of Crime Incident Locations.* The National Institute of Justice, Washington, DC, 2002.

[8] J. Conley, M. Gahegan, and J. Macgill, "A genetic approach to detecting clusters in point data sets," *GEOGR ANAL*, vol. 37, pp. 286-314, 2005.

[9] M. Ester, H.P. Kriegel, J. Sander, and X. Xu, "A density-based algorithm for discovering clusters in large spatial databases with noise," *ACM SIGKDD*, pp. 226-231, August 1996.

[10] W. Wang, J. Yang, and R.R. Muntz, "STING: A Statistical Information Grid Approach to Spatial Data Mining,", *VLDB*, pp. 186-195, 1997.

[11] P. Larrañaga and J.A. Lozano, *Estimation of Distribution Algorithms: A New Tool for Evolutionary Computation.* Kluwer Academic Publishers, 2002.

[12] H. Mühlenbein and G. Paaß, *From recombination of genes to the estimation of distributions I. Binary parameters,* Lecture Notes in Computer Science 1411, In Parallel Problem Solving from Nature - PPSN IV, 1996, pp. 178-187.

[13] W. Chang, D. Zeng, and H. Chen, "A stack-based prospective spatio-temporal data analysis approach," *Decision Support Systems*, vol. 45, pp. 697-713, 2008.

[14] M. Kulldorff and Information Management Services, Inc. SaTScan^{TM} v8.0: Software for the spatial and space-time scan statistics. http://www.satscan.org/, 2009.

[15] G.P. Patil and C. Taillie, "Upper level set scan statistic for detecting arbitrarily shaped hotspots," *Environ. Ecol. Stat.*, vol. 11, pp. 183-197, 2004.

[16] L. Duczmal, R. Assunção, "A simulated annealing strategy for the detection of arbitrarily shaped spatial clusters," *Comput. Statist. Data Anal.* vol. 45, pp. 269-286, 2004.

[17] T. Tango, K. Takahashi, "A flexibly shaped spatial scan statistic for detecting clusters," *Int J Health Geogr* 4:11, 2005.

[18] K. Takahashi, M. Kulldorff, T. Tango and K. Yih, "A flexibly shaped space-time scan statistic for disease outbreak detection and monitoring," *Int J Health Geogr* 7:14, 2008.

[19] R. Sahajpal, G.V. Ramaraju, and V. Bhatt, "Applying niching genetic algorithms for multiple cluster discovery in spatial analysis," *International Conference on Intelligent Sensing and Information Processing*, 2004.

[20] V. S. Iyengar, "On Detecting Space-Time Clusters," *ACM SIGKDD*, 2004.