Discovery of Co-evolving Spatial Event Sets *

Jin Soung Yoo[†]   Shashi Shekhar[†]   Sangho Kim[†]   Mete Celik[†]

Abstract

A spatial co-located event set represents a subset of spatial events whose instances are located in a spatial neighborhood. The discovery of co-evolving spatial event sets involves finding co-located event sets whose spatial prevalence variations over time are similar to a specific query sequence. Mining co-evolving spatial event sets is computationally challenging due to the high computational cost of finding co-located event instances on continuous geographic space, large temporal space and a composite interest measure, i.e., the spatial prevalence time sequence of a co-located event set. We propose a novel method for mining co-evolving spatial event sets. We analyze the proposed algorithm in terms of correctness and completeness, and experimentally evaluate the algorithm.

Keywords

Spatio-temporal data mining, Co-evolution pattern, Co-located events

1   Introduction

A spatial co-located event set represents a subset of spatial events whose instances are located in a spatial neighborhood. Examples of spatial events include outbreaks of disease, crime hot-spots, climate observations, distributions of plant species, mobile service request types, etc. Frequent spatial co-located event sets, i.e., co-location patterns [6, 13, 15], give important insights for many application domains such as Earth science, ecology, environmental management, public safety, public health, business, etc.

However, the spatio-temporal nature of datasets used in the various application domains brings intriguing questions regarding co-location pattern analysis. Scientists in these domains are often interested in understanding the evolution of co-location patterns among events. In this paper, we tackle the problem of temporal aspects of co-location pattern analysis, i.e., how the co-location patterns change over time. Specifically, we focus on identifying co-located event sets whose temporal occurrences are correlated with a special time series. For example, we can find that the co-occurrence of climate phenomena such as droughts and wild fires in Australia is similar to the variation of El Niño index values over the last 50 years [11]. Figure 1 (a) shows a spatio-temporal dataset consisting of instances of several spatial events over different time, each event type represented by a distinct shape. The interest of a co-located event set can be measured by its prevalence. A high prevalence value indicates that the spatial events likely show up together in a spatial neighborhood. In the illustrative example of Figure 1 (b), the time sequences of prevalence values of co-located event sets, e.g., {'⊡', '*'} and {'+', '×'}, are represented with a specific query time sequence which is depicted as a solid line. We can notice that the prevalence time sequence of {'+', '×'} is similar to the query time sequence. Thus a *co-evolving spatial event set* is a co-located event set whose spatial prevalence time sequence is similar to a specific query sequence in a given threshold. Identifying the patterns of co-evolving spatial events is useful in many applications. For example, ecologists rely on the discovery of patterns of events to understand the relationships between living organisms and their environment. It is of great interest to them to learn, for example, that certain animal behaviors show similar frequency patterns with the variation of fruit amounts in a forested region per month [1].

Mining co-evolving spatial event sets presents challenges due to the following reasons: First, identifying spatial co-located event sets is computationally expensive by itself since the instances of spatial events are embedded in a continuous space and share neighbor relationships. Second, we have to consider a composite interest measure, e.g., spatial prevalence time sequence, rather than a scalar numeric interest measure, e.g., spatial prevalence value. Exponentially increasing computational costs of generating the spatial prevalence time sequences of all combinatorial candidate event sets become prohibitively expensive. Third, the similarity functions for measuring the degree of consistency with a query time sequence are also computationally expensive with increases of time space. In this paper, we propose
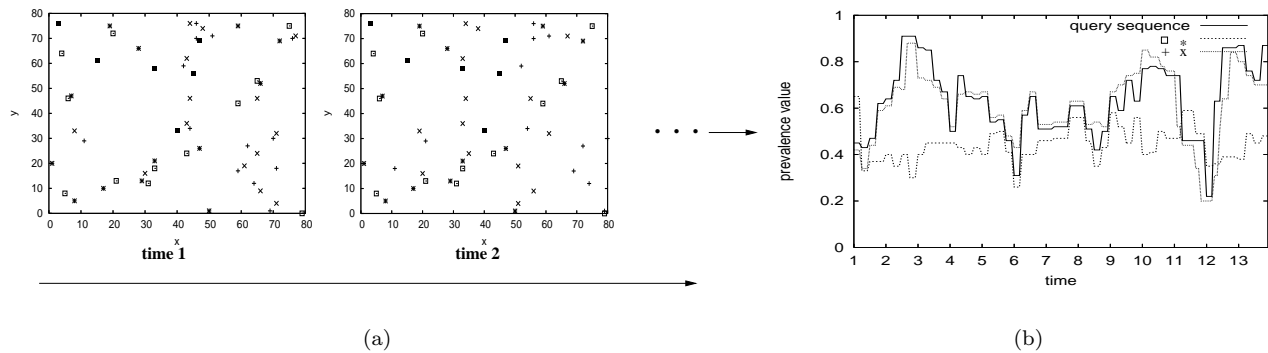
Figure 1: An illustrative example of co-evolving spatial events (a) A spatio-temporal dataset (b) Spatial prevalence time sequences and a specified query sequence

a novel algorithm to efficiently mine co-evolving spatial event sets.

Related Work: To our knowledge, researchers have yet to tackle the problem of mining co-evolving spatial event sets. In the spatial association mining literature, [8, 6, 13, 15] proposed different approaches for mining spatial co-location patterns. [8] adopted space partitioning for identifying neighboring objects for a frequent neighboring feature set, and used support count as the interest measure. [6] defined a statistically meaningful interest measure for spatial co-location patterns and proposed an instance join-based co-location mining algorithm. [13, 15] proposed to materialize spatial neighbor relationships for efficient co-location pattern mining. However, none of these works considers the temporal domain of the co-location pattern. Otherwise, in the temporal association mining literature, recent efforts have attempted to capture special temporal profiles of association patterns in market basket transaction datasets. [9] identified cyclic association rules, which discover periodically repetitive frequent patterns. [7] explored the problem of finding frequent itemsets along with calendar-based patterns which are defined with a calendar schema, e.g, year, month, and day. [14] proposed a similarity-based time-profiled association mining in a time-stamped transaction dataset. These methods are not directly applicable for mining co-evolving spatial event sets since there is no explicit transaction concept in a spatio-temporal dataset.

Contributions: In this paper, we discover co-evolving spatial event sets, i.e., co-located event sets whose spatial prevalence variations are similar to a specific query sequence. The concept of our previous similarity-based time-profiled association pattern [14] is extended to discover co-evolving spatial event sets.

The followings are our primary contributions: First, we provide a formal problem definition of co-evolving spatial event set mining. Second, we explore the event-level upper bound and the instance-level upper bound of a spatial prevalence time sequence, and adopt the lower bounding distance concept of a Euclidean distance-based similarity measure proposed in [14]. The upper bounds of spatial prevalence time sequences and the monotonicity property of the lower bounding distance make it possible to effectively reduce the search space of spatio-temporal events and to efficiently reduce expensive procedures for finding co-located instances. Third, we propose a novel co-evolving spatial event set mining algorithm. We analytically show that the proposed algorithm is complete and correct, i.e., there are no false droppings or false admissions in finding the similar co-located event sets. Finally, we experimentally evaluate the proposed algorithm. The experimental results show that the proposed algorithm outperforms a naive approach.

Outline: The remainder of the paper is organized as follows. Section 2 formally defines the problem of mining co-evolving event sets from a spatio-temporal dataset, and presents the basic concept of spatial co-location mining. The modeling of co-evolution patterns and our algorithmic design concepts are discussed in Section 3 and Section 4. Section 5 presents our algorithm for finding co-evolving spatial event sets. The proofs of correctness and completeness of the algorithm are given in Section 6. Section 7 presents the experimental evaluation. The conclusion and future work are discussed in Section 8.
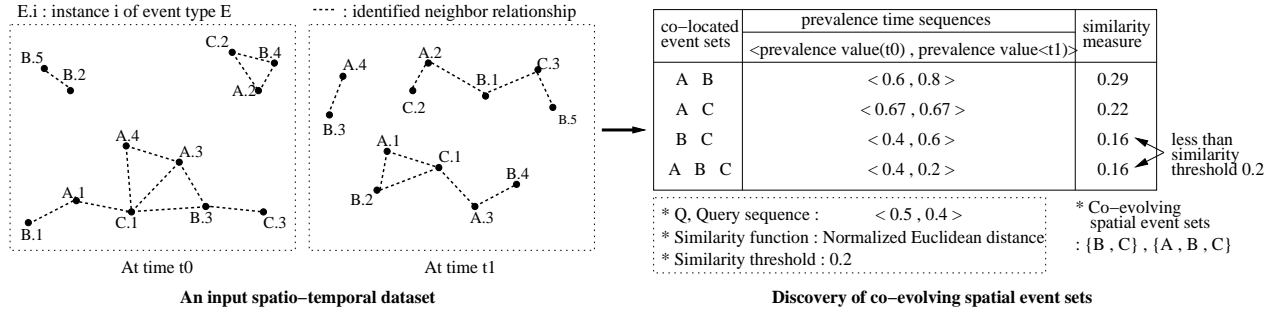
305

Figure 2: An example of mining co-evolving spatial event sets

2 Problem Statement and Basic Concepts

In this section, we provide the formal problem statement for the discovery of co-evolving spatial event sets. Then we describe the basic concept of spatial co-location pattern mining.

2.1 Problem Statement

Given:

1) A spatial framework SF

2) A time framework TF which can be divided into a set of disjoint time slots, $TF = t_0 \cup \ldots \cup t_{n-1}$.

3) A set of spatio-temporal events $E = \{e_1, \ldots, e_m\}$ and a set of their instance objects ST where each instance object $\in ST$ is a vector $<$ event type, instance id, location, time $>$, where location $\in SF$ and time $\in TF$.

4) A spatial neighbor relationship R over locations

5) A query time sequence $\vec{Q} = < q_0, \ldots, q_{n-1} >$ over TF

6) A time sequence similarity function: $f_{similarity}(\vec{P}, \vec{Q})$

7) A similarity threshold θ.

Develop:

An algorithm to find spatial co-located event sets whose prevalence variations over times are similar to a given time sequence.

Objective:

Find a complete and correct set of co-located events $C \subseteq E$ which satisfies $f_{similarity}(\vec{P_C}, \vec{Q}) \leq \theta$, where $\vec{P_C} = < p_0, \ldots, p_{n-1} >$ is the time sequence of spatial prevalence values of a co-located event set C over time slots $t_0, \ldots, t_{n-1}$.

We assume that a query time sequence $\vec{Q}$ is in the same scale as the prevalence measure of spatial co-located event sets or can be transformed to the same scale.

Figure 2 shows an example of the mining of co-evolving spatial event sets with a small spatio-temporal dataset related to two time slots t_0 and t_1. The prevalence time sequences of possible co-located event sets are generated using a spatial prevalence measure,

e.g., participation index (in Section 2.2). When a query sequence is $<0.5, 0.4>$, normalized Euclidean distance (in Section 3) is used as a similarity function, and a similarity threshold is 0.2, the output of the co-evolving spatial event set mining is {B, C} and {A, B, C} since the similarity values between their prevalence time sequences and the query time sequence are less than the given threshold.

2.2 Basic Concepts of Co-location Mining

Given a set of spatial events, a set of their instances, and a spatial neighbor relationship, a co-location(i.e., co-located event set) is a subset of spatial events whose instances form a clique using the neighbor relationship. Figure 3 shows an example dataset with three spatial events, A, B and C. Each object is represented by its event type and the unique instance id in each event type, e.g., A.1. For example, when a spatial neighbor relationship R is a distance metric and its threshold value is d, two spatial objects are neighbors if they satisfy the neighbor relationship, e.g., $R(A.1, B.1)$ $\Leftrightarrow distance(A.1, B.1) \leq d$. The identified neighbor relationships are described with dotted lines in Figure 3. A co-location instance is a set of objects which includes an object of each event type in the co-location and forms a clique relationship among them. For example, in Figure 3, {A.1, B.1} is an instance of co-location {A, B}, and {A.2, B.4, C.2} is an instance of co-location {A, B, C}. The interest of a co-location pattern can be measured by its prevalence. We use the participation index proposed in [6] as a co-location prevalence measure. The **participation index** $Pi(C)$ of a co-location $C = \{e_1, \ldots, e_k\}$ is defined as a minimum of participation ratio values of events in the co-location C, i.e., $Pi(C) = \min_{e_i \in C}\{Pr(C, e_i)\}$. The **participation ratio** $Pr(C, e_i)$ of event e_i in a co-location $C = \{e_1, \ldots, e_k\}$ is the fraction of objects of events e_i in the neighborhood of instances of co-location $C - \{e_i\}$, i.e., $Pr(C, e_i) = \frac{Number\ of\ distinct\ objects\ of\ e_i\ in\ instances\ of\ C}{Number\ of\ objects\ of\ e_i}$.
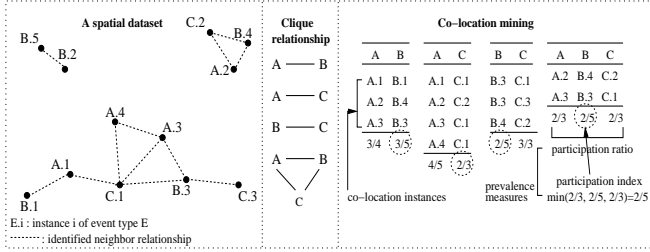
306

Figure 3: Spatial Co-location Pattern

For example, in the dataset of Figure 3, event A has four instance objects, event B has five instance objects, and event C has three instance objects. Consider the prevalence values of co-location $c=\{A, B, C\}$. The instances of co-location c are $\{A.2, B.4, C.2\}$ and $\{A.3, B.3, C.1\}$. The participation ratio of event A in the co-location c, $Pr(c, A)$ is $\frac{2}{4}$ since only A.2 and A.3 among four event A objects are involved in the co-location instances. $Pr(c, B)$ is $\frac{2}{5}$ and $Pr(c, C)$ is $\frac{2}{3}$. Thus the participation index of co-location c, $Pi(c)$, is $min\{Pr(c, A), Pr(c, B), Pr(c, C)\} = \frac{2}{5}$. A high participation index value indicates that the spatial events in a co-location pattern likely show up together.

3 Modeling Co-evolution Patterns

The participation index has been successfully used in spatial co-location mining since it represents the spatial statistical significance of a pattern [6]. We use a time sequence of participation index values as an interest measure for the variation of prevalences of a co-located event set over time. The granularity of the time period of the sequence is application dependent, e.g., day, week, month.

DEFINITION 3.1. *Given a spatio-temporal dataset $ST = ST_0 \cup \ldots \cup ST_{n-1}$ where ST_i is a set of spatio-temporal event objects occurring in time slot i, $i=0,\ldots,n-1$, the* **spatial prevalence time sequence** *of a co-located event set C, $\vec{P}_C =< p_0^C, \ldots, p_{n-1}^C >$ is the sequence of the participation index values of C over time slots, i.e.,*

$$\vec{P}_C =< Pi_{ST_0}(C), \ldots, Pi_{ST_{n-1}}(C) >,$$

where $Pi_{ST_j}(C)$, $0 \leq j < n - 1$, is the participation index value of a co-located event set C in a dataset ST_j at time slot j.

For example, in Figure 2, the participation index of a co-located event set $\{A, B\}$ at time slot $t0$ is 0.6 and its participation index at time slot $t1$ is 0.8. Thus, the spatial prevalence time sequence of $\{A, B\}$ is <0.6, 0.8>.

Next, we propose using *normalized Euclidean distance* [4] as a similarity function between a query time sequence and the prevalence time sequences.

DEFINITION 3.2. *For two time sequences $\vec{P} = < p_0, \ldots, p_{n-1} >$ and $\vec{Q} = < q_0, \ldots, q_{n-1} >$, the normalized Euclidean distance between $\vec{P}$ and $\vec{Q}$, $D(\vec{P}, \vec{Q})$, is defined as*

$$D(\vec{P}, \vec{Q}) = \sqrt{\frac{\sum_{i=0}^{n-1}(p_i - q_i)^2}{n}},$$

where n is the number of time slots.

For example, in Figure 2, the normalized Euclidean distance between the prevalence time sequence of a co-located event set $\{A, B\}$, <0.6, 0.8> and a query sequence, <0.5, 0.4> is 0.29.

Euclidean distance(i.e., $\mathcal{L}_2$ norm) is the most popular class of similarity measure in the time-series literature [3, 12, 5]. The normalized Euclidean distance of a prevalence time sequence $\vec{P}$ can be thought of as the deviation to a query sequence $\vec{Q}$, i.e., $\sqrt{\frac{\sum_{i=0}^{n-1}(p_i - q_i)^2}{n}} = \sigma(\vec{P})$.

4 Algorithmic Design Concepts

In this section, we discuss our algorithmic design concepts for mining co-evolving spatial event sets.

4.1 Co-located Event Instance Filtering

Identifying the instances of co-located event sets is computationally expensive since the instances of spatial events are embedded in a continuous space and share a variety of spatial relationships. A large fraction of the computation time is devoted to identifying the instances of co-location patterns. [15] proposes a method to materialize neighbor relationships to find co-location instances efficiently. We adopt the method for finding co-located event instances at each time slot. First, the method materializes the star neighbor relationships of a spatial dataset instead of finding all maximal clique relationships directly which is computationally expensive. Figure 4 illustrates the materialization of the star neighbor relationships of a spatial dataset at a time slot. The star neighborhood of an object is defined to a set of the center object and the objects in its neighborhood area whose event types are greater than the event type of the center object in a lexical order. In Figure 4, the star neighborhood areas of objects, for example, A.1 and B.4, are represented by dotted circles whose radius is a user specific spatial neighbor distance. The solid lines in each circle represent a star neighbor relationship with the center object. A.1

F.i : instance i of feature type F
——— : star neighbor relationship of A.1, B.4
d : neighbor distance

Disjoint star neighborhoods at time t0

Center feature	Center object	Neighbor objects
A	A.1	A.1 , B.1 , C.1
	A.2	A.2 , B.4 , C.2
	A.3	A.3 , B.3 , C.1
	A.4	A.4 , C.1
B	B.1	B.1
	B.2	B.2
	B.3	B.3 , C.1, C.3
	B.4	B.4 C.2
	B.5	B.5
C	C.1	C.1
	C.2	C.2
	C.3	C.3

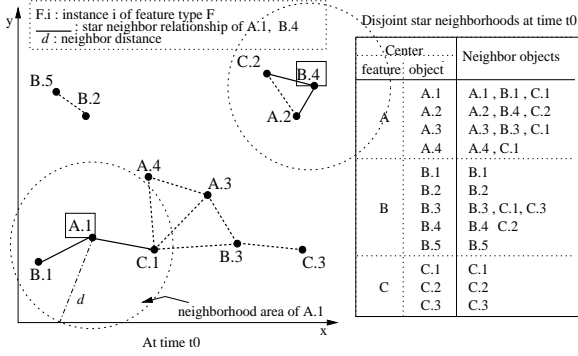neighborhood area of A.1

At time t0

Figure 4: Neighborhood materialization for co-located event instance filtering

has two neighboring objects, B.1 and C.1. The star neighborhood of A.1 is {A.1, B.1, C.1} including the center object A.1. If we consider A.2 as a center object, the star neighborhood of A.2 is {A.2, B.4, C.2}. Next consider the neighborhood of B.4, which has two neighbor objects, A.2 and C.2. A.2 is not included in the star neighborhood set of B.4 since the neighbor relationship between A.2 and B.4 is already reflected in the star neighborhood set of A.2. This materialization method is called disjoint star neighborhood partition (or edge partition in graph term) since it stores neighbor relationship information without any duplication. A set of all the disjoint star neighborhoods of the spatial dataset is listed in Figure 4. We can find the co-located event instances having a clique relationship from the star neighborhood set. First we find the star instances of a candidate co-located event set. For example, {A.1, B.1, C.1} is a star instance of {A, B, C}. Then, to check the cliqueness of {A.1, B.1, C.1}, we examine if the other objects {B.1, C.1} except the center object of the star instance, A.1, are a clique or not using the instances of a co-located event set {B, C}. The detailed algorithm will be described in Section 5.

4.2 Similar Co-located Event Set Filtering

We have several filtering steps for efficiently finding co-evolving co-located event sets. They are two event-level filterings, a coarse filtering, and a refinement filtering by true similarity measure. Before discussing these filtering schemes, we provide the following important properties of our spatial prevalence time sequence.

LEMMA 4.1. *The element values of the spatial prevalence time sequence of a co-located event set are* **monotonically non-increasing** *with the size of co-located event set at each time slot.*

Proof. The participation index is monotonically non-

increasing with increasing size of the co-located event set [6]. Thus, the participation index time sequence values follow the same monotonicity property at the disjoint time slot.

DEFINITION 4.1. *For a prevalence time sequence $\vec{P} = < p_0, \ldots, p_{n-1} >$ and a query time sequence $\vec{Q} = < q_0, \ldots, q_{n-1} >$, the* **lower bounding distance** *between $\vec{P}$ and $\vec{Q}$ is defined as*

$$D_{lb}(\vec{P}, \vec{Q}) = \sqrt{\frac{\sum_{i=0}^{n-1} f(p_i, q_i)}{n}},$$

$$where\ f(p, q) = \begin{cases} 0, & if\ p \geq q \\ (q - p)^2, & if\ p < q \end{cases}$$

The lower bounding distance considers the subsequences of the prevalence time sequence and the query time sequence at time slots where the element participation index is less than the corresponding query sequence value. For example, consider the prevalence time sequence of co-located event set {B, C}, <0.4, 0.6> in Figure 2. The lower bounding distance to the query sequence <0.5, 0.4> is $\sqrt{\{(0.4 - 0.5)^2 + 0\}/2}$=0.07.

LEMMA 4.2. *The lower bounding distance between the prevalence time sequence of a co-located event set and a query time sequence is* **monotonically non-decreasing** *with the size of the co-located event set.*

Proof. According to Definition 4.1, the lower bounding distance between $\vec{P}_C =< p_0^C, \ldots, p_{n-1}^C >$ for a size k co-located event set $C = \{e_1, \ldots, e_k\}$ and $\vec{Q} =< q_0, \ldots, q_{n-1} >$ is $D_{lb}(\vec{P}_C, \vec{Q}) = (\sum_{i=0, p_i^C < q_i}^{n-1} (q_i - p_i^C)^2/n)^{\frac{1}{2}}$. For a size $k + 1$ co-located event set $C' = C \cup \{e'\}$, where $e' \notin C$ and its prevalence time sequence $\vec{P}_{C'} =< p_0^{C'}, \ldots, p_{n-1}^{C'} >$, we need to prove that $D_{lb}(\vec{P}_C, \vec{Q}) \leq D_{lb}(\vec{P}_{C'}, \vec{Q})$. According to Lemma 4.1, the participation index is non-increasing with the size of co-located event set. Thus the participation index of C' is equal to or less than the participation index of C at all time slots such that $p_0^C \geq p_0^{C'}, \ldots, p_{n-1}^C \geq p_{n-1}^{C'}$, and $q_i - p_i^C \leq q_i - p_i^{C'}$ where $p_i^C < q_i$ and $p_i^{C'} < q_i$, $0 \leq i < n$. Thus, we can get $(\sum_{i=0, p_i^C < q_i}^{n-1} (q_i - p_i^C)^2/n)^{\frac{1}{2}} \leq (\sum_{i=0, p_i^{C'} < q_i}^{n-1} (q_i - p_i^{C'})^2/n)^{\frac{1}{2}}$, i.e., $D_{lb}(\vec{P}_C, \vec{Q}) \leq D_{lb}(\vec{P}_{C'}, \vec{Q})$.

For example, in Figure 2, $D_{lb}(\vec{P}_{AB}, \vec{Q})$=0, and $D_{lb}(\vec{P}_{ABC}, \vec{Q})$=0.07. Thus $D_{lb}(\vec{P}_{AB}, \vec{Q}) \leq D_{lb}(\vec{P}_{ABC}, \vec{Q})$. If $D_{lb}(\vec{P}_{AB}, \vec{Q})$ does not satisfy a given similarity threshold, $D_{lb}(\vec{P}_{ABC}, \vec{Q})$ also does not satisfy the threshold. Lemma 4.2 ensures that the lower bounding distance can be used to effectively reduce the co-evolving co-located event set search space.

4.2.1 Event-level filtering : We have two event-level filtering procedures that do not require examining co-located event instances and calculating true prevalence time sequences. They use the monotonicity property of the lower bounding distance. The first event-level filtering prunes a candidate event set if the lower bounding distance of a subset of the candidate event set does not satisify a given similarity threshold. The second event-level filtering is done by the estimated upper bound of the prevalence time sequence of a candidate event set and its lower bounding distance. We define the event-level upper bound of the prevalence time sequence of a co-located event set using the prevalence time sequences of its subsets.

DEFINITION 4.2. *Let C_k be a size k co-located event set and $A = \{C_{k-1}^1, \ldots, C_{k-1}^k\}$ be a set of all size $k-1$ subsets of C_k, where $C_{k-1}^j \subset C_k$, $1 \leq j \leq k$. Let $\vec{P}_{C_{k-1}^j} = <p_0^{C_{k-1}^j}, \ldots, p_{n-1}^{C_{k-1}^j}>$ be the spatial prevalence time sequence of $C_{k-1}^j \in A$. The* **event-level upper bound** *of spatial prevalence time sequence of C_k, $\vec{EU}_{C_k} = <u_0^{C_k}, \ldots, u_{n-1}^{C_k}>$ is $< \min\{p_0^{C_{k-1}^1}, \ldots, p_0^{C_{k-1}^k}\}, \ldots, \min\{p_{n-1}^{C_{k-1}^1}, \ldots, p_{n-1}^{C_{k-1}^k}\} >$.*

For example, in the example of Figure 2, the event-level upper bound sequence of {A, B, C} is $<\min(0.6, 0.67, 0.4), \min(0.8, 0.67, 0.6)>=<0.4, 0.6>$. If the lower bounding distance to the event-level upper bound of a co-located event set does not satisfy a given threshold, the candidate co-located event set is pruned.

4.2.2 Coarse filtering : We have a scheme to filter candidate event sets before doing expensive clique check operations for finding the co-located event instances. We explore the instance-level upper bound of the prevalence time sequence using the star instances of a candidate co-located event set. The instance-level upper bound is defined as following.

DEFINITION 4.3. *Let C_k be a size k co-located event set. The* **instance-level upper bound** *of the prevalence time sequence of C_k, $\vec{IU}_{C_k} = <u_0^{C_k}, \ldots, u_{n-1}^{C_k}>$ is $< p'_0, \ldots, p'_{n-1} >$, where p'_i is the participation index of star instances of a co-located C_k at time slot i where $1 \leq i \leq n-1$.*

If the lower bounding distance to the instance-level upper bound of a co-located event set does not satisfy a given threshold, the candidate co-located event set is pruned.

4.2.3 Refinement filtering : Finally, we filter similar co-located event sets using the true participation
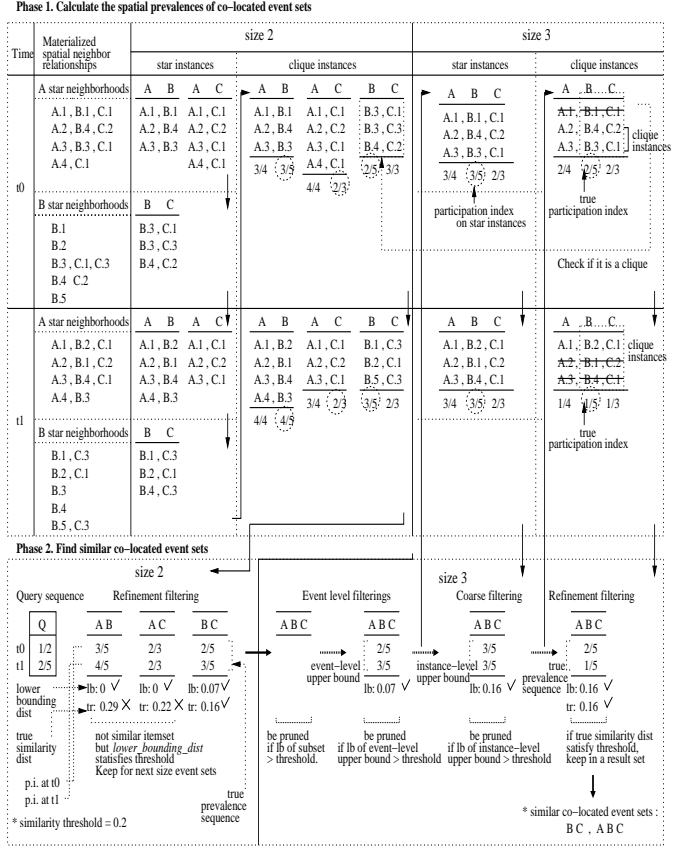

Figure 5: An example of a CE-COLOC algorithm trace

index values from exact clique co-located instances. The result set includes only co-located event sets whose similarity values satisfy a given threshold value.

5 Algorithm for Mining Co-evolving Spatial Event Sets

A naive method for finding co-evolving spatial event sets can follow a two-step procedure. First, it finds spatial instances of all possible co-located event sets, calculates their participation index values, and generates their prevalence time sequences over all time slots. Second, it searches the generated prevalence time sequences similar to a query sequence. In this step, advanced time series search methods using spatial indexing schemes can be used [3, 5]. However, as the number of both the event types and the time points increases, the computation cost to calculate the spatial prevalence values of all combinations of event sets becomes prohibitively expensive. The operation to find spatially co-located event instances requires especially extensive computation. We propose a one-step approach to combine the generation of prevalence time sequences with the sequence search. We develop an algorithm for mining Co-

Inputs:
```
E:a set of spatial event types
ST:a spatio-temporal dataset
    <event type, instance id, x, y, time>
r:a spatial neighbor relationship
```
$TF = \{t_0, \ldots, t_{n-1}\}$:a time slot frame
$\vec{Q}$: A query time sequence
D: A similarity function
θ: A similarity threshold

Output: All event sets whose prevalence time sequences are similar to $\vec{Q}$ under D and θ

Variables :
k: event set size
C_k: Set of size k candidate event set
$SN = \{SN_{t_0}, \ldots, SN_{t_{n-1}}\}$:   A set of star neighborhoods over time periods
$\vec{P}_k$:   Set of prevalence time sequences of size k event sets
L_k: Set of size k itemsets whose lower bounding distance of $\vec{P}_k \leq \theta$
S_k:Set of size k itemsets whose true similarity value $\leq \theta$
SI_k:star instances of size k candidate co-located event sets
CI_k:clique instances of size k candidate co-located event sets

Main:
1) SN=gen_star_neighborhoods(ST, TF, r);
2) $C_1 = E$;
3) C_2=generate_candidate_eventsets(C_1);
4) $\vec{P}_2$= generate_prevalence_sequences(C_2,SN);
5) (S_2, L_2)= find_similar_eventsets($\vec{P}_2$,$\vec{Q}$,D,θ);
6) $k = 3$;
7) **while** (not empty L_{k-1}) **do**
8) C_k=generate_candidate_eventsets(L_{k-1});
9) C_k=do_event-level_candidate_filtering
         (C_k, $\vec{P}_{k-1}$, $\vec{Q}$, D, θ);
10) **for** $t \in TF$ **do**
11) SI_k=filter_star_instances(C_k,SN_t);
12) **end do**
13) C_k=do_coarse_candidate_filtering(C_k, SI_k, $\vec{Q}$, D, θ);
14) CI_k=filter_clique_instances(C_k,SI_k);
15) $\vec{P}_k$=generate_prevalence_sequences(C_k,CI_k);
16) (S_k, L_k)=find_similar_eventsets (C_k, $\vec{P}_k$, $\vec{Q}$, D, θ);
17) $k = k + 1$;
18) **end**
19) **return** $\bigcup(S_2, \ldots, S_k)$;

Algorithm 1: CE-COLOC Algorithm

Evolving spatial CO-LOCated event sets(CE-COLOC). Figure 5 illustrates a trace example of the CE-COLOC algorithm. It has two phases: phase one calculates the spatial prevalence values of co-located event sets and phase two finds similar co-located event sets. The two phases work interactively with increase in size of event set. Algorithm 1 shows the pseudo code. The detailed explanation of the code is given as follows.

Generate a set of star neighborhoods per a time slot from an input spatio-temporal dataset(Step 1): Given a spatio-temporal dataset, a spatial neighbor relationship and a time slot frame, first find all neighboring object

pairs of the dataset using a geometric method such as plane sweep [2], or a spatial query method using quaternary tree or R-tree [10] per each time slot. The star neighborhoods are generated by grouping the neighboring objects per each object. Figure 5 lists the star neighborhood set sorted by the event type and event id of the center object at each time slot. It does not include the neighborhoods of the last event C since the neighborhoods have only the center object as shown in Figure 4 and our co-located event sets are considered from size 2.

Find similar size 2 co-located event sets(Steps 3-5): All pairs($k = 2$) of event types become size 2 candidate event sets(C_2). By scanning the materialized neighborhood sets, the participation index values of size 2 co-located event sets are calculated per each time slot and their spatial prevalence time sequences($\vec{P}_2$) are generated. If the similarity value between the prevalence time sequence of a candidate event set and a query sequence is not greater than a given threshold, the candidate event set is added to a result set(S_2). On the fly, the lower bounding distance between the prevalence time sequence and the query sequence is calculated. If its value is not greater than the similarity threshold, the event set is added to L_2 for generating the next size candidate event sets. For example, in Figure 5, only event set {B, C} is included in the result set since the true similarity value between $\vec{P}_{BC}$ and $\vec{Q}$, 0.16, is less than 0.2. However, the lower bounding distances of {A, B}, {A, C} and {B, C} satisfy the threshold value. Thus they all are kept into L_2 for generating the next size candidate event sets(C_3).

Generate size $k > 2$ candidate event sets and filter by event-level prunings (Steps 8-9) : All size k ($k > 2$) candidate event sets(C_k) are generated with size $k - 1$ event sets(L_{k-1}) whose lower bounding distances are not greater than a given similarity threshold. Here, we have two event-level prunings by the non-decreasing monotonocity property of lower bounding distance. First, if a size $k - 1$ subset of a generated size k event set is not in the L_{k-1}, the candidate event set is eliminated. Second, we have another pruning using the event-level upper bound sequence of a candidate event set. The upper bound sequence is generated using the prevalence time sequences of their subsets. For example, in Figure 5, the event-level upper bound of a candidate event set {A, B, C} is $< 2/5, 3/5 >$ by Definition 4.2. If the lower bounding distance between the upper bound prevalence sequence and a query sequence is greater than the threshold value, the event set is removed from the set of candidate event sets.

Filter the star instances of candidate co-located event sets(Step 11): The star instances of a candidate

event set are gathered from the star neighborhoods whose center object event type is the same as the first event of the candidate event set at each time slot. For example, the instances of {A, B, C} are gathered from the event 'A' star neighborhood set.

Filter candidate co-located event sets by a coarse pruning(Step 13):   From the star instances of a candidate event set, the coarse participation index values of the candidate event set are calculated over time slots, and the instance-level upper bound sequence is generated. If the lower bounding similarity value between the upper bound sequence and the query sequence is greater than the similarity threshold, the candidate co-located event set is removed from the set of candidate event sets.

Filter clique co-located event instances(Step 14): The clique instances of current candidate co-located event sets are filtered by looking up the clique instances of events except the first event of the candidate co-located event set. For example, in Figure 5, to check the cliqueness of a star instance {A.1, B.1, C.1} of {A, B, C}, we examine if a subinstance {B.1, C.1} but not A.1 is in the set of clique instances of {B, C}. This instance look-up operation can be performed efficiently by an instance key which is composed of the ids of objects in the instance.

Generate true prevalence time sequences and find similar co-located event sets (Step 15-16):   The true prevalence time sequence of a candidate event set is calculated using true participation index values from exact clique co-located event instances. The true similarity value between the prevalence time sequence and the query sequence is calculated. If the value satisfies the threshold, the co-located event set is included in the result set(S_k). On the fly, the lower bounding distance between the true prevalence time sequence and the query sequence is calculated. If its value is less than or equal to a given similarity threshold, the event set is added to L_k for generating the next size candidate event sets(C_{k+1}). The size of the examined event set is increased to $k = k + 1$. The above procedures(step 7 - step 18) are repeated until no item in L_k remains.

6   Analytical Analysis

We analyze our algorithm in terms of completeness and correctness. Completeness means that the CE-COLOC algorithm finds all co-evolving spatial event sets whose prevalence values are similar to a query sequence under a similarity threshold. Correctness means that the similarity values between the prevalence time sequences of all found co-evolving event sets and the query sequence are below a given similarity threshold. First, we introduce related theorems.

LEMMA 6.1. *For two sequences $\vec{P}$ and $\vec{Q}$, the normalized Euclidean distance $D(\vec{P}, \vec{Q})$ and the lower bounding distance $D_{lb}(\vec{P}, \vec{Q})$ have the following inequality :*

$$D_{lb}(\vec{P}, \vec{Q}) \leq D(\vec{P}, \vec{Q})$$

Proof. According to Definition 3.2 and 4.1,
$D_{lb}(\vec{P}, \vec{Q}) = \sqrt{\sum_{i=0, p_i < q_i}^{n-1} (p_i - q_i)^2 / n}$
$\leq \sqrt{\sum_{i=0}^{n-1} (p_i - q_i)^2 / n} = D(\vec{P}, \vec{Q}).$

LEMMA 6.2. *Let $\vec{EU}_{C_k} = <u_0^{C_k}, \ldots, u_{n-1}^{C_k}>$ be the event-level upper bound sequence of a co-located event set C_k and $\vec{P}_{C_k} = <p_0^{C_k}, \ldots, p_{n-1}^{C_k}>$ be the prevalence time sequence of C_k. The elements in two sequences hold the following inequalities: $u_0^{C_k} \geq p_0^{C_k}, \ldots, u_{n-1}^{C_k} \geq p_{n-1}^{C_k}$.*

Proof. According to Definition 4.2,
$u_0^{C_k} = min\{p_0^{C_{k-1}^1}, \ldots, p_0^{C_{k-1}^k}\}, \ldots,$
$u_{n-1}^{C_k} = min\{p_{n-1}^{C_{k-1}^1}, \ldots, p_{n-1}^{C_{k-1}^k}\}$, where $p_h^{C_{k-1}^j}$ is the participation index of jth size $k - 1$ subset of C_k at time slot h, $1 \leq j \leq k$ and $0 \leq h \leq n - 1$. The participation index value does not increase with the event set size increasing by Lemma 4.1. Thus $p_h^{C_{k-1}^1} \geq p_h^{C_k}, \ldots, p_h^{C_{k-1}^k} \geq p_h^{C_k}$. We get $u_h^{C_k} = min\{p_h^{C_{k-1}^1}, \ldots, p_h^{C_{k-1}^k}\} \geq p_h^{C_k}$ at each time slot h.

LEMMA 6.3. *Let $C = \{e_1, \ldots, e_k\}$ be a size k co-located event set and SI be a set of star instances of C. The participation index of C from SI is not less than the true participation index of C.*

Proof. The participation ratio of e_1 from SI is the maximum possible probability that the objects of event e_1 of C have clique relationships with the objects of the other events $e_2, \ldots, e_k$ in C since only objects of event e_1 in the star instances can be included in a clique co-location instance of C. The participation ratio of $e_j(1 < j \leq k)$ from SI is also the maximum possible probability that the objects of event e_j have clique relationships with the objects of event e_1 in C since our neighbor relationship is symmetric. Thus the participation index of C calculated from the star instances is not less than the true participation index of C, $min_{e_i \in C}\{possible \, max \, Pr(C, e_i)\} \geq min_{e_i \in C}\{Pr(C, e_i)\}$.

LEMMA 6.4. *Let $\vec{IU}_{C_k} = <u_0^{C_k}, \ldots, u_{n-1}^{C_k}>$ be the instance-level upper bound sequences of a co-located event set C_k and $\vec{P}_{C_k} = <p_0^{C_k}, \ldots, p_{n-1}^{C_k}>$ be the prevalence time sequence of C_k. The elements in the two sequences hold the following inequalities: $u_0^{C_k} \geq p_0^{C_k}, \ldots, u_{n-1}^{C_k} \geq p_{n-1}^{C_k}$.*

Proof. Each element value of an instance-level upper bound of an event set C_k is the participation index from the star instances of C_k at each time slot by Definition 4.3. According to Lemma 6.3, the value is not greater than the true participation index of C_k at each time slot. Thus all element values of the instance-level upper bound of C_k are equal to or greater than all element values of the true prevalence time sequence of C_k at each time slot.

THEOREM 6.1. *The CE-COLOC algorithm is complete.*

Proof. The completeness of the algorithm can be shown by the following four facts. First, we will show that in step 8 of Algorithm 1, the *generate_candidate_eventsets* function using event sets whose lower bounding distance satisfies the threshold does not miss a true event set. Let C_{k-1} be a subset of a size k candidate event set C_k, and the upper bounding distance between the prevalence time sequence of C_{k-1}, $P_{C_{k-1}}$ and a query sequence $\vec{Q}$, $D_{lb}(\vec{P}_{C_{k-1}}, \vec{Q})$ be greater than θ. According to Lemma 6.1 and Lemma 4.2, $D_{lb}(\vec{P}_{C_{k-1}}, \vec{Q}) \leq D_{lb}(\vec{P}_{C_k}, \vec{Q}) \leq D(\vec{P}_{C_k}, \vec{Q})$. Thus if $D_{lb}(\vec{P}_{C_{k-1}}, \vec{Q}) > \theta$, $D(\vec{P}_{C_k}, \vec{Q}) > \theta$ and C_k is not a similar event set.

Second, we will show that the *do_event − level_candidate_filtering* function in step 9 does not miss a true event set. By Lemma 6.2, all element values of the event-level upper bound of C_k are equal to or greater than all element values of the true prevalence time sequence of C_k at each time slot. $D_{lb}(E\vec{U}_{C_k}, \vec{Q}) \leq D_{lb}(\vec{P}_{C_k}, \vec{Q}) \leq D(\vec{P}_{C_k}, \vec{Q})$. Thus if $D_{lb}(E\vec{U}_{C_k}, \vec{Q}) > \theta$, $D(\vec{P}_{C_k}, \vec{Q}) > \theta$ and C_k is not a similar event set.

Third, we will show that the *do_coarse_candidate_filtering* function in step 13 does not miss a true event set. By Lemma 6.4, all element values of the instance-level upper bound of C_k are equal to or greater than all element values of the true prevalence time sequence of C_k at each time slot. $D_{lb}(I\vec{U}_{C_k}, \vec{Q}) \leq D_{lb}(\vec{P}_{C_k}, \vec{Q}) \leq D(\vec{P}_{C_k}, \vec{Q})$. Thus if $D_{lb}(I\vec{U}_{C_k}, \vec{Q}) > \theta$, $D(\vec{P}_{C_k}, \vec{Q}) > \theta$ and C_k is not a similar event set.

Finally, the *find_similar_eventsets* function in step 5 and 16 using true prevalence time sequences does not miss a true co-located event set.

THEOREM 6.2. *The CE-COLOC algorithm is correct.*

Proof. The correctness can be guaranteed by steps 5 and 16 in Algorithm 1. The *fine_similar_eventsets* includes only event sets whose similarity value is not greater than the user-specified threshold.

7 Experimental Evaluation

We conducted several experiments to examine the effect of our algorithm design. We compared the performance of our CE-COLOC algorithm and the naive approach described in Section 5 using synthetic datasets. In the naive method, we used the same method in Section 4.1 for finding co-located event instances. However, the naive method does not have any pruning scheme. In the experiment, we focused on parameter values which give a workload on co-evolution pattern finding, e.g., the number of time slots and the number of event types.

Synthetic spatio-temporal datasets were generated using a method similar to the spatial data generator used in [6]. We generated a spatial dataset per each time slot. Overall spatial frame size was 5000×5000. The whole frame was divided into regular grids whose side length was a neighbor distance R. The default spatial neighbor distance R was 10. The following is the procedure we used to generate general synthetic datasets in which co-located event instances were randomly distributed. First we generated P initial patterns whose average size was S. The event types of each pattern were randomly chosen from E events. Average I instances per each pattern were generated. For locating a co-located event instance, first we chose a grid cell randomly. All points of the instance were randomly located within the chosen neighborhood grid cell. The procedure was repeated at each time slot with the number of time slots T. The number of default event types E was 20. At each time point, the number of co-located event sets P was 20, the average size of co-located event set S was 5 and the average number of co-located event instances I was 150. The default time slots T was 30. Query time sequences were randomly generated in the scale of the participation index at each time slot. The default similarity threshold was 0.2.

In the first experiment, we compared the overall execution time of the two algorithms, CE-COLOC and the naive method, using different similarity threshold values. As can be seen in Figure 6 (a), the CE-COLOC algorithm showed much less execution time compared with the naive method under low threshold values. The CE-COLOC showed increases with increases in the similarity threshold. By contrast, the naive method showed a stable execution time because it generates the prevalence time sequences of all combination event sets independent of the thresholds and then compares them with a query sequence. In the second experiment, we examined the effect of different numbers of time slots. Figure 6 (b) shows that both algorithms increased their execution time with increases in the number of time slots. The CE-COLOC algorithm was more scalable with number of time slots. In the third experiment
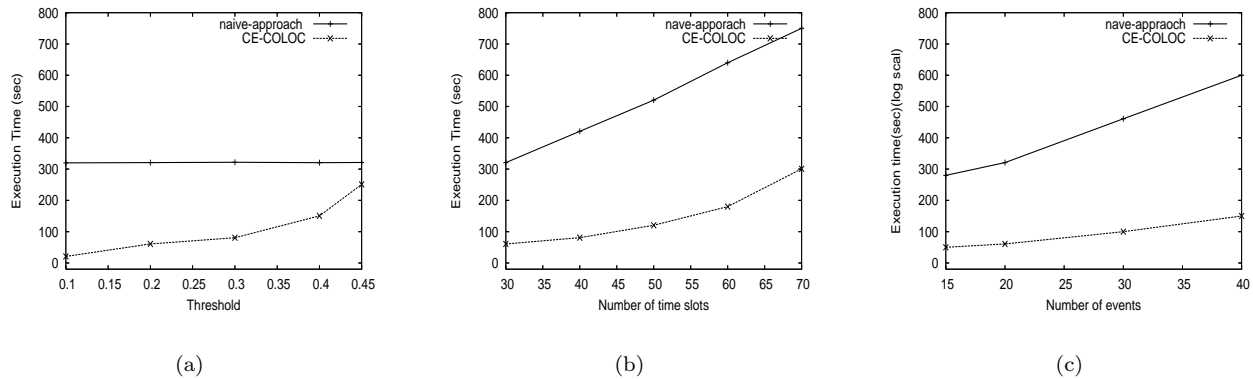
Figure 6: (a) Effect of thresholds (a) Effect of number of time slots (b) Effect of number of events

of Figure 6 (c), we compared the performances with datasets of different number of event types. Overall execution time increased gradually with the increase of event type number under a fixed average size of co-located event instances, 5 (the maximal size was 13).

8 Conclusions and Future Work

We explored the problem of mining co-evolving spatial event sets from a spatio-temporal dataset and proposed a novel algorithm to discover co-evolving co-located event sets. The proposed algorithm substantially reduces the search space of spatio-temporal event sets with the use of several filtering schemes. We shows that the algorithm is correct and complete in finding co-evolving co-located event sets. The initial experimental results show that our algorithm outperforms a naive method.

In the future, we plan to examine the behavior of our algorithm in more skewed datasets and real-world datasets. In addition, we plan to consider other similarity functions[5] rather than a Euclidean distance based measure for discovering co-evolving spatial event sets, and to explore the computational structures in spatio-temporal data.

References

[1] University of Minnesota, *Spatio-temporal Analysis Techniques for Ecology Behavior*, http://www.cs.umn.edu/research/chimps/.

[2] M. Berg, M. Kreveld, M. Overmars and O. Schwarzkopf, *Computational Geometry*, Springer, ISBN=3540656200, 2000.

[3] C. Faloutsos, M. Ranganathan and Y.Manolopoulos, *Fast subsequence matching in time-series database*, in the Proceedings of ACM SIGMOD Conference, 1993.

[4] D. Goldin, T. Millstein and A. Kutlu, *Bounded similarity querying for time-series data*, Information and Computation, 2004.

[5] D. Gunopulos and G. Das, *Time Series Similarity Measures and Time Series Indexing*, SIGMOD Record, Volume 30, Number 2, 2001.

[6] Y. Huang, S. Shekhar and H. Xiong, *Discovering Co-location Patterns from Spatial Datasets:A General Approach*, IEEE Transactions on Knowledge and Data Engineering, Volume 16, Number 12, 2004.

[7] Y. Li, P. Ning, X. S. Wang and S. Jajodia, *Discovering Calendar-Based Temporal Assocation Rules*, In the Proceeding of International Symposium Temporal Representation and Reasoning(TIME), 2001.

[8] Y. Morimoto, *Mining Frequent Neighboring Class Sets in Spatial Databases*, In the Proceeding of ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 2001.

[9] B. Ozden, S. Ramaswamy and A. Silberschatz, *Cyclic Association Rules*, In the Proceeding of IEEE Int. Conference on Data Engineering, 1998.

[10] S. Shekhar and S. Chawla, *Spatial Databases: A Tour*, Prentice Hall, ISBN 0130174807, 2003.

[11] G. H. Taylor, *Impacts of El Nino on Southern Oscillation on the Pacific Northwest*, http://www.ocs.orst.edu/reports/enso_pnw.html.

[12] B.K. Yi and C. Faloutsos, *Fast Time Sequence Indexing for Arbitrary $\mathcal{L}_p$ norms*, In the Proceeding of VLDB Conference, 2000.

[13] J.S. Yoo and S. Shekhar, *A Partial Join Approach for Mining Co-location Patterns*, In the Proceeding of ACM International Symposium on Advances in Geographic Information Systems(ACM-GIS), 2004.

[14] J.S. Yoo, P. Zhang and S. Shekhar, *Mining Time-Profiled Associations: An Extended Abstract*, In the Proceeding of the Pacific-Asia Conference on Data Mining and Knowledge Discovery, 2005.

[15] J.S. Yoo, S. Shekhar and M. Celik, *A Join-less Approach for Co-location Pattern Mining: A Summary of Results*, in the Proceeding of IEEE International Conference on Data Mining(ICDM), 2005.