Discovering Co-location Patterns from Spatial Datasets: A General Approach

Yan Huang, *Member, IEEE,* Shashi Shekhar, *Fellow, IEEE,* and Hui Xiong, *Student Member, IEEE*

Abstract

Given a collection of boolean spatial features, the co-location pattern discovery process finds the subsets of features frequently located together. For example, the analysis of an ecology dataset may reveal symbiotic species. The spatial co-location rule problem is different from the association rule problem, since there is no natural notion of transactions in spatial data sets which are embeded in continuous geographic space. In this paper, we provide a transaction-free approach to mine co-location patterns by using the concept of proximity neighborhood. A new interest measure, a participation index, is also proposed for spatial co-location patterns. The participation index is used as the measure of prevalence of a co-location for two reasons. First, this measure is closely related to the cross-K function, which is often used as a statistical measure of interaction among pairs of spatial features. Second, it also possesses an anti-monotone property which can be exploited for computational efficiency. Furthermore, we design an algorithm to discover co-location patterns. This algorithm includes a novel multi-resolution pruning technique. Finally, experimental results are provided to show the strength of the algorithm and design decisions related to performance tuning.

Yan Huang is with the Department of Computer Science and Engineering, University of North Texas, USA. E-mail: huangyan@unt.edu.

Shashi Shekhar is with the Department of Computer Science and Engineering, University of Minnesota, 200 Union Steet SE, Minneapolis, MN 55455, USA. E-mail: shekhar@cs.umn.edu.

Hui Xiong is with the Department of Computer Science and Engineering, University of Minnesota, 200 Union Street SE, Minneapolis, MN 55455, USA. E-mail: huix@cs.umn.edu.

Manuscript received August 26, 2002; revised June 17, 2003; accepted April 27, 2004.

Corresponding Author: Hui Xiong. Phone: 612-626-8084. Fax: 612-626-1596.

Index Terms

Co-location Patterns, Spatial Association Rules, Participation Index.

I. Introduction

Co-location patterns represent subsets of Boolean spatial features whose instances are often located in close geographic proximity. Figure 1 shows a dataset consisting of instances of several Boolean spatial features, each represented by a distinct shape. A careful review reveals two co-location patterns, i.e., $\{$'+','$\times$'$\}$ and $\{$'o','*'$\}$. Real-world examples of co-location patterns include symbiotic species, e.g., the Nile Crocodile and Egyptian Plover in ecology. Boolean spatial features describe the presence or absence of geographic object types at different locations in a two dimensional or three dimensional metric space, such as the surface of the Earth. Examples of Boolean spatial features include plant species, animal species, road types, cancers, crime, and business types.

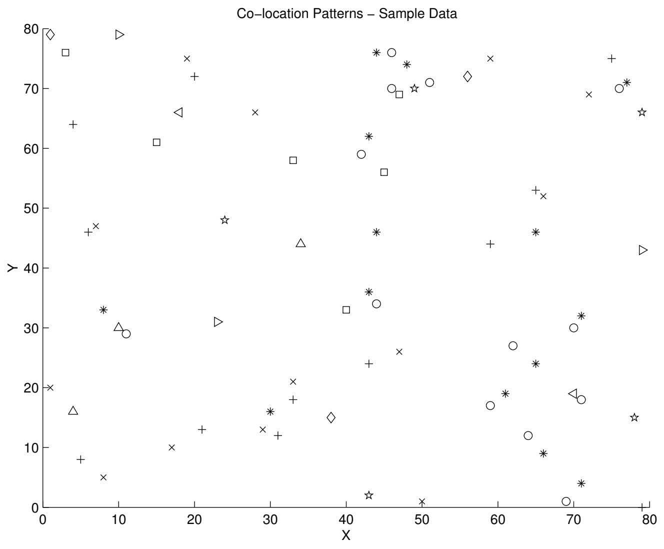


Fig. 1. Co-location Patterns Illustration.

Co-location rules are models to infer the presence of spatial features in the neighborhood of instances of other spatial features. For example, "Nile Crocodiles $\rightarrow$ Egyptian Plover" predicts the presence of Egyptian Plover birds in areas with Nile Crocodiles.

We formalize the co-location rule mining problem as follows: Given 1) a set T of K spatial feature types $T = \{f_1, f_2, \ldots, f_K\}$ and their instances $I = \{i_1, i_2, \ldots, i_N\}$, each $i_i \in$ I is a vector $<$ instance-id, spatial feature type, location $>$ where location $\in$ spatial framework S and 2) a neighbor relation $\mathcal{R}$ over instances in I, efficiently find all the co-located spatial features in the form of subsets of features or rules.

A. Related Work:

Approaches to discovering co-location rules in the literature can be categorized into two classes, namely spatial statistics and data mining approaches. Spatial statistics-based approaches use measures of spatial correlation to characterize the relationship between different types of spatial features. Measures of spatial correlation include the cross-K function with Monte Carlo simulation [6], mean nearest-neighbor distance, and spatial regression models [5]. Computing spatial correlation measures for all possible co-location patterns can be computationally expensive due to the exponential number of candidate subsets given a large collection of spatial Boolean features.

Data mining approaches can be further divided into a clustering-based map overlay approach and association rule-based approaches. A clustering-based map overlay approach [9], [8] treats every spatial attribute as a map layer and considers spatial clusters (regions) of point-data in each layer as candidates for mining associations. Given X and Y as sets of layers, a clustered spatial association rule is defined as $X \Rightarrow Y(CS, CC\%)$, for $X \bigcap Y = \emptyset$, where CS is the clustered support, defined as the ratio of the area of the cluster (region) that satisfies both X and Y to the total area of the study region S, and $CC\%$ is the clustered confidence, which can be interpreted as $CC\%$ of areas of clusters (regions) of X intersect with areas of clusters (regions) of Y.

Association rule-based approaches can be divided into transaction-based approaches and distance-based approaches. Transaction based approaches focus on defining transactions over space so that an *Apriori*-like algorithm [2] can be used. Transactions over space can be defined by a reference-feature centric model [12]. Under this model, transactions are created around instances of one user-specified spatial feature. The association rules are derived using the *Apriori* [2] algorithm. The rules found are all related to the reference feature. However, generalizing this paradigm to the case where no reference feature is specified is non-trivial. Also, defining transactions around locations of instances of all features may yield duplicate counts for many candidate associations.

A distance-based approach was proposed concurrently by Morimoto [15] and us [19]. Morimoto defined distance-based patterns called k-neighboring class sets. In his work, the number of instances for each pattern is used as the prevalence measure, which does not possess an anti-monotone property by nature. However, Morimoto used a non-overlapping-instance constraint to get the anti-monotone property for this measure. In contrast, we developed an event centric model, which does away with the non-overlapping-instance constraint. We also defined a new prevalence measure called the participation index. This measure possesses the desirable anti-monotone property. A more detailed comparison of these two works is presented in Section VI.

B. Contributions:

This paper extends our work [19] on the event centric model and makes the following contributions. First, we refine the definition of distance-based spatial co-location patterns by providing an interest measure called the participation index. This measure not only possesses a desirable anti-monotone property for efficiently identifying co-location patterns but also allows for formalizing the correctness and completeness of the proposed algorithm. Furthermore, we show the relationship between the participation index and a spatial statistics interest measure, the cross-K function. More specifically, we show that the participation index is an upper-bound of the cross-K function. Second, we provide an algorithm to discover co-location patterns from spatial point datasets. This algorithm includes a novel multi-resolution filter, which exploits the spatial auto-correlation property of spatial data to effectively reduce the search space. An experimental evaluation on both synthetic and real-world NASA climate datasets is provided to compare alternative choices for key design decisions.

C. Outline and Scope:

Section II describes our approach for modeling co-location patterns. Section III proposes a family of algorithms to mine co-location patterns. We show the relationship between the participation index and an estimator of the cross-K function and provide an analysis of the algorithms in the area of correctness, completeness, and computational efficiency in section IV. In section V, we present the experimental setup and results. Section VI provides a comparison between our work and the work by Morimoto [15]. Finally, in section VII, we present the conclusion and future work.

This paper does not address issues related to the edge effects or the choice of the neighborhood size and interest measure thresholds. Quantitative association, e.g. (A, A) and quantitative association rules, e.g. $(A \Rightarrow A)$, are beyond the scope of this paper.

II. Our Approach for Modeling Co-location Patterns

This section defines the event centric model, our approach to modeling co-location patterns. We use Figure 2 as an example spatial dataset to illustrate the model. In the figure, each instance is uniquely identified by $T.i$, where T is the spatial feature type and i is the unique id inside each spatial feature type. For example, $B.2$ represents the instance 2 of spatial feature B. Two instances are connected by edges if they have a spatial neighbor relationship.

A **co-location** is a subset of boolean spatial features. A **co-location rule** is of the form: $c_1 \Rightarrow c_2(p, cp)$, where c_1 and c_2 are co-locations, $c_1 \bigcap c_2 = \emptyset$, p is a number representing the prevalence measure, and cp is a number measuring conditional probability.

An important concept in the event centric model is proximity neighborhood. Given a reflexive and symmetric neighbor relation $\mathcal{R}$ over a set S of instances, a **$\mathcal{R}$-proximity neighborhood** is a set $I \subset S$ of instances that form a clique [4] under the relation $\mathcal{R}$. The definition of neighbor relation $\mathcal{R}$ is an input and should be based on the semantics of the application domains. The neighbor relation $\mathcal{R}$ may be defined using spatial relationships (e.g. connected, adjacent [1]), metric relationships (e.g. Euclidean distance

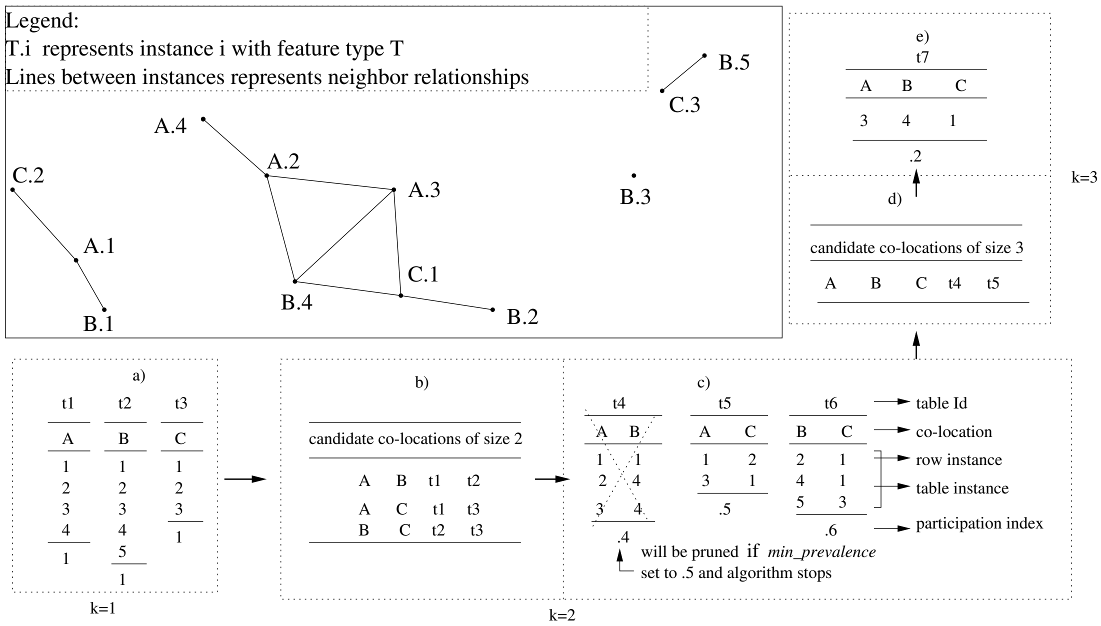

Legend:
T.i represents instance i with feature type T
Lines between instances represents neighbor relationships
A.1
A.2
A.3
A.4
B.1
B.2
B.3
B.4
B.5
C.1
C.2
C.3
a)
t1 t2 t3
A B C
1 2 3 4 | 1
1 2 3 4 5 | 1
1 2 3 | 1
k=1
b)
candidate co-locations of size 2
A B t1 t2
A C t1 t3
B C t2 t3
c)
t4: A B — 1 1, 2 4, 3 4 — .4
t5: A C — 1 2, 3 1 — .5
t6: B C — 2 1, 4 1, 5 3 — .6
will be pruned if min_prevalence set to .5 and algorithm stops
table Id
co-location
row instance
table instance
participation index
k=2
d)
candidate co-locations of size 3
A B C t4 t5
e)
t7
A B C
3 4 1
.2
k=3

Fig. 2. Spatial Dataset to Illustrate the Event Centric Model

[15]) or a combination (e.g. shortest-path distance in a graph such as a road-map). The $\mathcal{R}$-proximity neighborhood concept is different from the neighborhood concept in topology [14] since some supersets of a $\mathcal{R}$-proximity neighborhood may not qualify to be $\mathcal{R}$-proximity neighborhoods.

Two $\mathcal{R}$-proximity neighborhoods I_1 and I_2 are $\mathcal{R}$-reachable to each other if $I_1 \bigcup I_2$ is a $\mathcal{R}$-proximity neighborhood. A $\mathcal{R}$-proximity neighborhood I is a **row instance** (denoted by $row\_instance(c)$) of a co-location c if I contains instances of all the features in c and no proper subset of I does so. For example, $\{A.3,\ B.4,\ C.1\}$ is a row instance of co-location $\{A,\ B,\ C\}$ in the spatial dataset shown in Figure 2. But $\{A.2,\ A.3,\ B.4,\ C.1\}$ is not a row instance of co-location $\{A,\ B,\ C\}$ because its proper subset $\{A.3,\ B.4,\ C.1\}$ contains instances of all features in $\{A,\ B,\ C\}$. In another example, $\{A.2, A.4\}$ is not a row instance of co-location $\{A\}$ because its proper subset $\{A.2\}$ (or $\{A.4\}$) contains instances of all the features in $\{A\}$. The **table instance**, $table\_instance(c)$, of a co-location c is the collection of all row instances of c. In Figure 2, t1, t2, t3, t4, t5, t6, and t7 represent table instances. For instance, t5= {{A.1, C.2}, {A.3, C.1}} is a table instance of the co-location {A, C}.

The **participation ratio** $pr(c, f_i)$ for feature type f_i in a size-k co-location $c = \{f_1, \dots, f_k\}$ is the fraction of instances of feature f_i $\mathcal{R}$-reachable to some row instance of co-location $c - \{f_i\}$. The

participation index $pi(c)$ of a co-location $c = \{f_1, \ldots, f_k\}$ is $min_{i=1}^{k}\{pr(c,\ f_i)\}$. The participation index is used as the measure of prevalence of a co-location for two reasons. First, the participation index is closely related to the cross-K function [16], [17] which is often used as a statistical measure of interaction among pairs of spatial features. Second, it also possesses an anti-monotone property which can be exploited for computational efficiency. The participation ratio can be computed as $\frac{\pi_{f_i}(|table\_instance(c)|}{|table\_instance(f_i)|}$, where π is the relational projection operation with duplication elimination. For example, in Figure 2, row instances of co-location $\{A, B\}$ are $\{\{A.1, B.1\}$, $\{A.2, B.4\}, \{A.3, B.4\}\}$. Only two ($B.1$ and $B.4$) out of five instances of spatial feature B participate in co-location $\{A, B\}$. So $pr(\{A, B\},\ B) = 2/5 = 0.4$. Similarly, $pr(\{A, B\}, A)$ is 0.75. The participation index $pi(\{A, B\})$ = min(0.75, 0.4) = 0.4.

The **conditional probability** $cp(c_1 \Rightarrow c_2)$ of a co-location rule $c_1 \Rightarrow c_2$ is the fraction of row instances of c_1 $\mathcal{R}$-reachable to some row instance of c_2. It is computed as $\frac{|\pi_{c_1}(table\_instance(\{c_1 \bigcup c_2\}))|}{|table\_instance(\{c_1\})|}$, where π is the relational projection operation with duplication elimination. For example, in the co-location rule $A \Rightarrow C$ in Figure 2, the conditional probability of this rule is equal to $\frac{|\pi_A(table\_instance(\{A \bigcup C\}))|}{|table\_instance(\{A\})}$ $= \frac{2}{4} = 50\%$.

III. Co-location Mining Algorithm

In this section, we introduce a co-location mining algorithm. Note that the prevalence measure used in Figure 3 is the participation index and that a co-location pattern is prevalent if the values of its participation index is above a user specified threshold.

As shown in Figure 3, the algorithm takes a set ET of spatial event types, a set E of event instances, user-defined functions representing spatial neighborhood relationships as well as thresholds for interest measures, i.e. prevalence and conditional probability. The algorithm outputs a set of prevalent co-location rules with the values of the interest measures above the user defined thresholds.

The initialization steps (i.e., steps 1 and 2 in Figure 3) assign starting values to various data-structures used in the algorithm. We note that the value of the participation index is 1 for all co-locations of size 1. In other words, all co-locations of size 1 are prevalent and there is no need for either the computation of a prevalence measure or prevalence-based filtering. Thus, the set C_1 of candidate co-locations of size

```
Input:      (a) $E$ = {Event-ID, Event-Type, Location in Space} representing a set of events
            (b) $ET$ = {Set of $K$ boolean spatial event types}
            (c) Spatial relationship $\mathcal{R}$
            (d) Minimum prevalence threshold $\theta$ and conditional probability threshold $\alpha$.
            Note that participation index is the measure of prevalence.
Output:     A set of co-locations with prevalence and conditional probability values greater
            than user-specified minimum prevalence and conditional probability thresholds.
Variables:  k: co-location size
            $C_k$: set of candidate size-$k$ co-locations
            $T_k$: set of table instances of co-locations in $C_k$
            $P_k$: set of prevalent size-$k$ co-locations
            $R_k$: set of co-location rules of size-$k$
            $T\_C_k$: set of coarse-level table instances of size-k co-locations in $C_k$
Steps:
 1.         Co-location size $k = 1$; $C_1 = ET$; $P_1 = ET$;
 2.         $T_1$=generate_table_instance($C_1$, E);
 3.         if (fmul=TRUE) then
 4.             $T\_C_1$=generate_table_instance($C_1$,multi_event($E$));
 5.         Initialize data structure $C_k, T_k, P_k, R_k, T\_C_k$ to be empty for $1 < k \le K$;
 6.         while(not empty $P_k$ and $k < K$) do {
 7.             $C_{k+1}$ = generate_candidate_colocation($C_k$, k);
 8.             if(fmul=TRUE) then
 9.                 $C_{k+1}$=multi_resolution_pruning($\theta$,$C_{k+1}, T\_C_k$,multi_rel($\mathcal{R}$));
10.             $T_{k+1}$=generate_table_instance($\theta$, $C_{k+1}$, $T_k$,$\mathcal{R}$);
11.             $P_{k+1}$=select_prevalent_colocation($\theta$, $C_{k+1}$, $T_{k+1}$);
12.             $R_{k+1}$=generate_colocation_rule($\alpha$, $P_{k+1}$, $T_{k+1}$);
13.             $k = k + 1$;
14.         }
15.         return union($R_2, \ldots, R_{k+1}$);
```

Fig. 3. Overview of the Algorithms

1 as well as the set P_1 of prevalent co-locations of size 1 are initialized to ET, the set of event types. The set T_1 of table instances of size 1 co-location is created by sorting the set E of event instances by event types. If a multi-resolution pruning step is desired, the set of events are discretized into coarse level instances. The set T_C_1 of coarse-level table instances of size 1 co-locations is generated by sorting the coarse-level event instances by event types.

The proposed algorithms for mining co-location rules iteratively perform four basic tasks, namely generation of candidate co-locations, generation of table instances of candidate co-locations, pruning, and generation of co-location rules. These tasks are carried out inside a loop iterating over the size of the co-locations. Iterations start with size 2 since our definition of prevalence measure allows no pruning for co-locations of size 1.

A. Generation of Candidate Co-locations

We could rely on a combinatorial approach and use *apriori_gen* [2] to generate size $k + 1$ candidate co-locations from size k prevalent co-locations.

The *apriori-gen* function takes as argument P_k, the set of all prevalent size k co-locations. The function works as follows. First, in the *join* step, we join P_k with P_k. This step is specified in a SQL-like syntax as follows:

```
insert into $C_{k+1}$
select $p$.f$_1$, $p$.f$_2$,..., $p$.f$_k$, $q$.f$_k$, $p$.table_instance_id, $q$.table_instance_id
from $P_k$ $p$, $P_k$ $q$
where $p$.f$_1$ = $q$.f$_1$, ..., $p$.f$_{k-1}$ = $q$.f$_{k-1}$, $p$.f$_k$ < $q$.f$_k$;
```

Next, in the *prune* step, we delete all co-locations $c \in C_k$ such that some size $k - 1$ subset of c is not in P_k:

```
forall co-location $c \in C_{k+1}$ do
    forall size $k-1$ subsets $s$ of $c$ do
        if ($s \notin P_k$) then delete $c$ from $C_{k+1}$;
```

Note that the column f_i of P_k refers to the i feature of co-locations in table P_k and the column table_instance_id of table P_k refers to table instances of appropriate co-locations.

B. Generation of Table Instances of Candidate Co-locations

Computation for generating size $k + 1$ candidate co-locations can be expressed as the following join query:

```
forall co-location $c \in C_{k+1}$
    insert into $T_c$ /* $T_c$ is the table instance of co-location $c$ */
    select $p$.instance$_1$, $p$.instance$_2$, ..., $p$.instance$_k$, $q$.instance$_k$
    from $c$.table_instance_id$_1$ $p$, $c$.table_instance_id$_2$ $q$
    where $p$.instance$_1$=$q$.instance$_1$, ..., $p$.instance$_{k-1}$=$q$.instance$_{k-1}$, ($p$.instance$_k$,
            $q$.instance$_k$) $\in \mathcal{R}$;
end;
```

The query takes the size $k+1$ candidate co-location set C_{k+1} and table instances of the size k prevalent co-locations as arguments and works as follows: $c.table\_instance\_id_1$ and $c.table\_instance\_id_2$ specify the table instances of the two co-locations joined in $apriori\_gen$ to produce c. Here, a sort-merge join is preferred because the table instances of each iteration can be kept sorted for the next iteration. This follows from a similar property of *apriori-gen* [2]. Sort order is based on an ordering of the set of feature types to order feature types in a co-location to form the sort-field. Finally, all co-locations with empty table instances will be eliminated from C_{k+1}.

The join computation for generating table instances has two constraints, a spatial neighbor relationship constraint ($(p.instance_k, q.instance_k) \in \mathcal{R}$) and a combinatorial distinct event-type constraint ($p.instance_1 = q.instance_1, \ldots, p.instance_{k-1} = q.instance_{k-1}$). We examine three strategies for computing this join: a geometric strategy, a combinatorial strategy, and a hybrid strategy. These are described in forthcoming subsections. Exploration of other join strategies is beyond the scope of this paper but we may explore such strategies in future work.

Geometric Approach: The geometric approach can be implemented by neighborhood relationship-based spatial joins of table instances of prevalent co-locations of size k with table instance sets of prevalent co-locations of size 1. In practice, spatial join operations are divided into a filter step and a refinement step [18] to efficiently process complex spatial data types such as point collections in a row instance. In the filter step, the spatial objects are represented by simpler approximations such as the MBR - Minimum Bounding Rectangle. There are several well-known algorithms, such as plane sweep [3], space partition [11] and tree matching [13], which can then be used for computing the spatial join of MBRs using the overlap relationship; the answers from this test form the candidate solution set. In the refinement step, the exact geometry of each element from the candidate set and the exact spatial predicates are examined along with the combinatorial predicate to obtain the final result.

Combinatorial Approach: The combinatorial join predicate (i.e. $p.instance_1 = q.instance_1$, $p.instance_2 = q.instance_2$, $\ldots$, $p.instance_{k-1} = q.instance_{k-1}$) can be processed efficiently using a sort-merge join strategy [10], since the set of feature types is ordered and tables $c.table\_instance\_id_1$ and $c.table\_instance\_id_2$ are sorted. The resulting tuples are checked for the spatial condition $((p.instance_k, q.instance_k) \in \mathcal{R})$ to get the row-instance in the result. In Figure 2, table 4 of co-location $\{A, B\}$ and table 5 of co-location $\{A, C\}$ are joined to produce the table instance of co-location $\{A, B, C\}$ because co-location $\{A, B\}$ and co-location $\{A, C\}$ were joined in *apriori_gen* to produce co-location $\{A, B, C\}$ in the previous step. In the example, row instance $\{3, 4\}$ of table 4 and row instance $\{3, 1\}$ of table 5 are joined to generate row instance $\{3, 4, 1\}$ of co-location $\{A, B, C\}$ (Table 7). Row instance $\{1, 1\}$ of table 4 and row instance $\{1, 2\}$ of table 5 fail to generate row instance $\{1, 1, 2\}$ of co-location $\{A, B, C\}$ because instance 1 of B and instance 2 of C are not neighbors.

Hybrid Approach: The hybrid approach chooses the more promising of the spatial and combinatorial approaches in each iteration. In our experiment, it picks the spatial approach to generate table instances for co-location patterns of size 2 and the combinatorial approach for generating table instances for co-location patterns of size 3 or more.

C. Pruning

Candidate co-locations can be pruned using the given threshold θ on the prevalence measure. In addition, multi-resolution pruning can be used for spatial dataset with strong auto-correlation [6], i.e., where instances tend to be located near each other.

Prevalence-based Pruning: We first calculate the participation indexes for all candidate co-locations in T_{k+1}. Computation of the participation indexes can be accomplished by keeping a bitmap of size cardinality(f_i) for each feature f_i of co-location c. One scan of the table instance of c will be enough to put 1s in the corresponding bits in each bitmap. By summarizing the total number of 1s (p_{f_i}) in each bitmap, we obtain the participation ratio of each feature f_i (divide p_{f_i} by $|$instance of $f_i|$). In Figure 2 c), to calculate the participation index for co-location $\{A, B\}$, we need to calculate the participation ratios for A and B in co-location $\{A, B\}$. Bitmap b_A= (0,0,0,0) of size four for A and bitmap b_B = (0,0,0,0,0) of size 5 for B are initialized to zeros. Scanning of table 4 will result in b_A= (1,1,1,0) and b_B = (1,0,0,1,0). Three out of four instances of A (i.e., 1, 2, and 3) participate in co-location $\{A, B\}$, so the participation ratio for A is .75. Similarly, the participation ratio for B is .4. Therefore, the participation index is min$\{$.75, .4$\}$ = .4.

After the participation indexes are determined, prevalence-based pruning is carried out and non-prevalent co-locations are deleted from the candidate prevalent co-location sets. For each remaining prevalent co-location c after prevalence-based pruning, we keep a counter to specify the cardinality of the table instance of c. All the table instances of the prevalent co-locations in this iteration will be kept for generation of the prevalent co-locations of size $k + 2$ and discarded after the next iteration.

Multi-resolution Pruning: Multi-resolution pruning is learned on a summary of spatial data at a coarse resolution using a disjoint partitioning, e.g., pagination imposed by leaves of a spatial index or a grid. A new neighbor relationship $\mathcal{R}^c$ on partitions is derived from relationship $\mathcal{R}$ so that two partitions are $\mathcal{R}$ neighbors if any two instances from each of the two partitions are $\mathcal{R}$ neighbors. We combine all instances of a spatial feature f in each partition s in the partitioning as a new coarse instance $< s, f, m >$ in the coarse space, where m is the number of instances of spatial point feature f in cell s. For each candidate co-location generated by *apriori_gen*, we generate its coarse table instance using new coarse instances, new neighbor relationship $\mathcal{R}^c$, and its coarse participation index based on the coarse table instance. Multi-resolution pruning eliminates a co-location if its coarse participation indexes fall below the threshold, because coarse participation never under-estimates the participation index, as shown in section 4.2.

We now illustrate multi-resolution pruning by using a simple recti-linear grid for simplicity. In Figure 4 a), different shapes represent different point spatial feature types. Every instance has a unique ID in its spatial feature type and is labeled below it in the figure. Two instances are defined as neighbors if they are in a common $d \times d$ square. A grid with uniform cell size d is super-imposed on the dataset. Cells

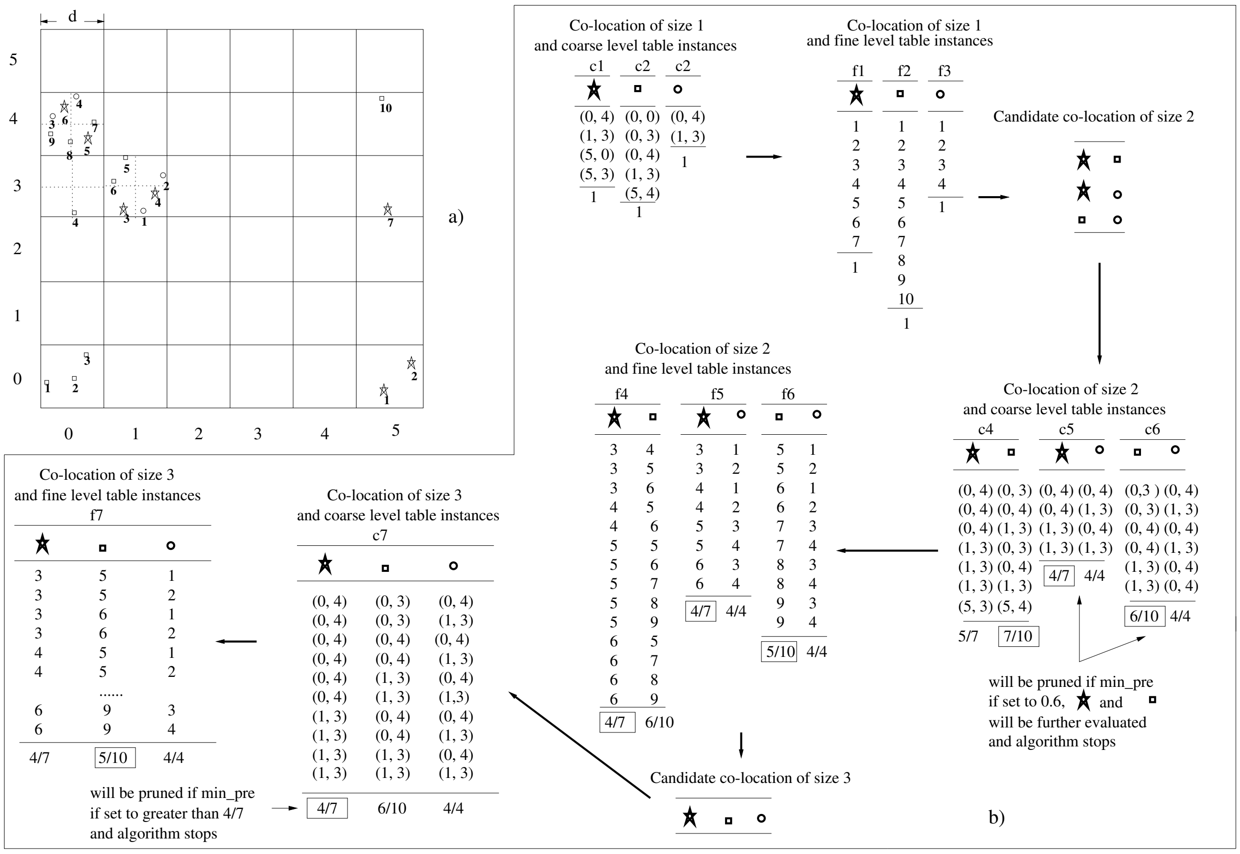

a)
b)
Co-location of size 1 and coarse level table instances
Co-location of size 1 and fine level table instances
Candidate co-location of size 2
Co-location of size 2 and coarse level table instances
Co-location of size 2 and fine level table instances
Candidate co-location of size 3
Co-location of size 3 and coarse level table instances
Co-location of size 3 and fine level table instances
will be pruned if min_pre if set to 0.6, and will be further evaluated and algorithm stops
will be pruned if min_pre if set to greater than 4/7 and algorithm stops

Fig. 4. Co-location Miner Algorithm with Multi-resolution Pruning Illustration

(i,j) refer to cells with an x-axis index of i and a y-axis index of j. In this grid, two cells are coarse-neighbors if their centers are in a common square of size $d \times d$, which imposes an 8-neighborhood (North, South, East, West, North East, North West, South East, South West) on the cells. For example, cell-pairs $\{(0,3),(0,4)),((0,3),(1,3)\}$ and $\{(0,4),(1,3)\}$ illustrate coarse-neighbors. This coarse-neighborhood definition guarantees that two cells are neighbors if there exists a pair of points from each of the two cells which are neighbors in the original dataset. The process of multi-resolution pruning is shown as follows.

First, we generate coarse table instances of candidate co-locations of size $k+1$ by joining the coarse table instances with the coarse-neighbor relationships.

Next, we calculate the participation indexes for all candidate co-locations based on the coarse table instances. For each spatial feature f_i, we add up all the counts of point instances in each coarse instance with 1s in its corresponding bitmap (p_{f_i}) and divide this by $|$instance of $f_i|$ to get the coarse-participation ratio of feature f_i. For example, in Figure 4 a), coarse $pr((\star, \circ), \star) = 4/7$ since there are 2 coarse row instances of $\{\star, \circ\}$, each containing 2 fine-grain instances of $\star$ and totally 7 fine-grain instances of $\star$. Similarly, coarse $pr(\{\star, \circ\}, \circ) = 4/4 = 1$, yielding coarse participation index $pi(\{\star, \circ\}) = min(4/7, 4/4)$ = 4/7. Figure 4 b) shows coarse table instances of co-locations $\{\star, \Box\}$, $\{\star, \circ\}$ and $\{\Box, \circ\}$. If the threshold for prevalence is set to 0.6, then co-location c_5 and c_6 can be pruned by multi-resolution pruning. We also note that the sizes of coarse table instances are smaller than the sizes of table instances at fine resolution. This shows the possibility of computation cost saving via multi-resolution pruning for clustered datasets. Finally, the examples in Figure 4 b) show that the coarse participation ratios and participation indexes never underestimate the true participation indexes of the original dataset.

D. Generating Co-location Rules

The generate_colocation_rule function generates all the co-location rules with the user defined conditional probability threshold α from the prevalent co-locations and their table instances. The conditional probability of a co-location rule $c_1 \Rightarrow c_2$ in the event centric model is the probability of c_1 reachable to a $\mathcal{R}$-proximity neighborhood containing all the features in c_2. It can be calculated as: $\frac{|\pi_{c_1}(table\_instance(c_1 \bigcup c_2))|}{|table\_instance(c_1)|}$, where π is a projection operation with duplication elimination. Bitmaps or other data structures can be used for efficient computation using the same strategies for prevalence-based pruning.

IV. Analysis of the Co-location Mining Algorithms

Here, we analyze the co-location mining algorithms in the areas of statistical interpretation of co-location patterns, completeness, correctness, and computational complexity.

A. Statistical Interpretation of the Co-location Patterns

In spatial statistics [6], interest measures such as the cross-K function, a generalization of Ripley's K-function [16], [17] (and variations such as the L-function and G function) are used to identify co-located spatial feature types. The cross-K function $K(h)$ for binary spatial features is defined as follows: $K_{ij}(h) = \lambda_j^{-1}E[\text{number of type } j \text{ instances within distance } h \text{ of a randomly chosen type } i \text{ instance}]$, where λ_j is the density (number per unit area) of type j instances and h is the distance. Without edge effects [7], the cross-K-function could be estimated by: $\hat{K}_{ij}(h) = \frac{1}{\lambda_i \lambda_j W} \sum_k \sum_l I_h(d(i_k, j_l))$, where $d(i_k, j_l)$ is the distance between the k'th instance of type i and the l'th instance of type j, I_h is the indicator function assuming value 1 if the distance between instance i_k and j_l $d(i_k, j_l) \leq h$, and value 0 otherwise, and W is the area of the study region. $\lambda_j \times \hat{K}_{ij}(h)$ estimates the expected number of type j instances within distance h of a type i instances. The variance of the cross-K function can be estimated by Monte Carlo simulation [6] in general and by a close form equation under special circumstances [6]. In Figure 5, the cross-K functions of the two pairs of spatial features, i.e., $\{$'+','x'$\}$ and $\{$'o','*'$\}$, are well above the spatial complete randomness curve $y = \pi * h^2$, while the cross-K functions of the other random two pairs of spatial features, i.e., $\{$'*','x'$\}$ and $\{$'*','+'$\}$, are very close to complete spatial randomness. This figure does not show the confidence band.

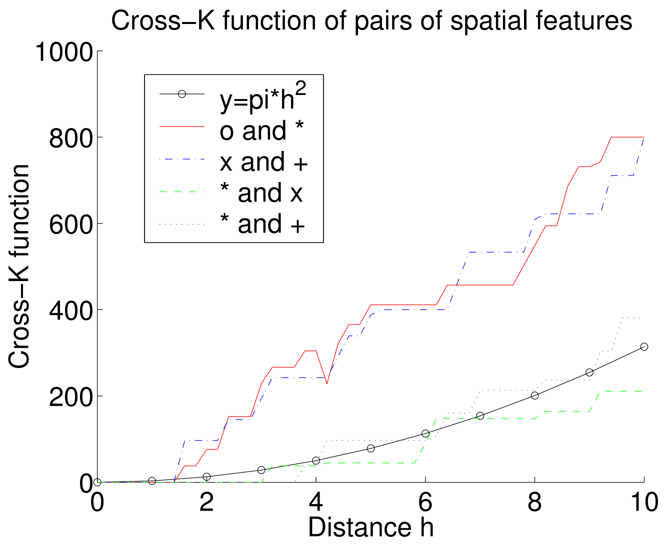


Fig. 5. Ripley's Cross-K Function

We compare the participation index with the cross-K function in an attempt to provide an independent method for evaluating co-location patterns. In particular, we explore the correctness of co-locations using the following characterization of the relationship between the cross-K function and co-locations.

Lemma 1: Participation index $pi(\{A, B\})$ for co-location $\{A, B\}$ is an upper-bound on $\frac{\hat{K_{AB}(h)}}{W}$, where $\hat{K_{AB}}(h)$ is the estimation of the cross-K function of co-location $\{A, B\}$ for a proximity neighborhood defined by distance $\leq h$, and W is the total area of the region.

Proof: From the definition of the participation index and the definition of cross-K function we have $pi(\{A,B\}) = \min\{\frac{|\pi_A(table\_instance(A,B))|}{|A|}, \frac{|\pi_B(table\_instance(A,B))|}{|B|}\}$ and $\frac{\hat{K_{AB}(h)}}{W} = \frac{1}{W} \cdot \frac{1}{\lambda_A \lambda_B W} \sum_k \sum_l I_h(d(A_k, B_l)) = \frac{|table\_instance(A,B)|}{|A| \times |B|}$. We need to show only that $\frac{|\pi_A(table\_instance(A,B))|}{|A|} \geq \frac{|table\_instance(A,B)|}{|A| \times |B|}$ and $\frac{|\pi_B(table\_instance(A,B))|}{|B|} \geq \frac{|table\_instance(A,B)|}{|A| \times |B|}$. To prove the first inequality, we need to show only that $|\pi_A(table\_instance(\{A,B\}))| \geq \frac{|table\_instance(\{A,B\})|}{|B|}$. This is obvious because the total number of instances of A with at least one instance of type B nearby (left side) is always greater or equal to the average number of instances of type A around an instance of type B (right side). The second inequality could be proved in a similar manner.

Lemma 2: The table instance $table\_instance(\{A,B\})$ of a binary co-location $\{A,B\}$ has enough information to compute the estimator $\frac{\hat{K_{AB}(h)}}{W}$ of the cross-K function for $h = d$, where distance d defines the proximity neighborhood.

Proof: Since $\frac{\hat{K_{AB}(h)}}{W} = \frac{1}{W} \cdot \frac{1}{\lambda_A \lambda_B W} \sum_k \sum_l I_h(d(A_k, B_l)) = \frac{|table\_instance(A,B)|}{|A| \times |B|}$, this lemma holds.

Lemma 1 may be used to establish the correctness of co-location rules with respect to the threshold defined by $\frac{\hat{K}_{AB}(h)}{W}$, and Lemma 2 may be used to establish the co-location miner as an algorithm to efficiently compute $\hat{K}_{AB}$ for selected co-locations, particularly when the multi-resolution filter is effective. We are at present aware of only the use of the cross-K function to characterize pairwise spatial interactions. We plan to explore spatial statistics research literature to look for measures of spatial interaction among more than two features and compare those measures to the participation index.

B. Completeness and Correctness

Lemma 3 (Anti-monotone): The participation ratio and participation index are anti-monotone (monotonically non-increasing) as the size of the co-location increases.

Proof: The participation ratio is anti-monotonic because a spatial feature instance that participates in a row instance of a co-location c also participates in a row instance of a co-location c' where $c' \subseteq c$. The participation index is also anti-monotonic because 1) the participation ratio is anti-monotonic and 2) $pi(c \bigcup f_{k+1}) = \min_{i=1}^{k+1}\{pr(c \bigcup f_{k+1}, f_i)\} \leq \min_{i=1}^{k}\{pr(c \bigcup f_{k+1}, f_i)\} \leq \min_{i=1}^{k}\{pr(c, f_i)\} = pi(c)$.

Lemma 4: The coarse participation index computed by multi-resolution pruning never underestimates the participation indexes of the original dataset. The candidate co-location set found is a superset of the prevalent co-location set on the original dataset.

Proof: When co-location size = 1, the value of the coarse participation index and the true participation index is 1, so Lemma 4 is trivially true. Suppose Lemma 4 is true for co-locations size=k. Let us consider the case that co-location size is equal to $k+1$. For each candidate co-location c of size $k+1$ generated from the *apriori_gen* by joining c_1 and c_2 of size k, we generate its coarse instance table by joining the coarse instance tables of c_1 and c_2. Because Lemma 4 is true for co-locations of size k, the candidate co-location set of size k found is a superset of the prevalent co-location set on the original dataset. Thus c_1 and c_2 are in the candidate co-location set in the previous iteration and their coarse level table instances are available to be joined to produce the coarse level table instance of c. The table join to produce the coarse table instance of c has the following property: if $\mathcal{R}(p_1, p_2)$ is in the original dataset, then coarse $\mathcal{R}^c(cell\, c_1, cell\, c_2)$ will be in the coarse-level dataset given $p_1 \in c_1$ and $p_2 \in c_2$. When we calculate the coarse participation index, any spatial feature instance which participates in the co-location in the original dataset will contribute to the counts during the coarse participation ratio calculation. So the coarse participation ratios never underestimate the true participation ratios, implying that the coarse participation index never underestimates the true participation index and that the pruning will not eliminate any truly prevalent co-location. Thus the candidate co-location set after multi-resolution pruning is a superset of the prevalent co-location set on the original dataset.

Lemma 5 (Completeness): The Co-location Miner algorithm is complete.

Proof: The schema level pruning using *apriori_gen* is complete due to the monotonicity of the participation index as proved in Lemma 3. Then we prove that the join of the table instances of c_1 and c_2 to produce the table instance of c is complete. According to the proximity neighborhood definition, any subset of a proximity neighborhood is a proximity neighborhood too. For any instance $I = \{i_1, \ldots, i_{k+1}\}$ of co-location c, subsets $I_1 = \{i_1, \ldots, i_k\}$ and $I_2 = \{i_1, \ldots, i_{k-1}, i_{k+1}\}$ are neighborhoods, i_k and i_{k+1} are neighbors, and I_1 and I_2 are row instances of C_1 and C_2 respectively. Joining I_1 and I_2 will produce I. Enumeration of the subsets of each of the prevalent co-locations ensures that no spatial co-location rules with both high prevalence and high conditional probabilities are missed. We then prove that multi-resolution pruning does not affect completeness. By Lemma 4, the co-location set found is a superset of the prevalent co-location set on the original dataset. Thus multi-resolution pruning does not falsely eliminate any prevalent co-location.

Lemma 6 (Correctness): The Co-location Miner is correct.

Proof: We will show only that the row instance of each co-location is correct, as that will imply the correctness of the participation index values and that of each co-location meeting the user specified threshold. An instance $I_1 = \{i_{1,1}, \ldots, i_{1,k}\}$ of $c_1 = \{f_1, \ldots, f_{k+1}\}$ and an instance $I_2 = \{i_{2,1}, \ldots, i_{2,k}\}$ of $c_2 = \{f_1, \ldots, f_{k-1}, f_{k+1}\}$ is joined to produce an instance $I_{new} = \{i_{1,1}, \ldots, i_{1,k}, i_{2,k}\}$ of $c = \{f_1, \ldots, f_{k+1}\}$ if: 1) all elements of I_1 and I_2 are the same except $i_{1,k}$ and $i_{2,k}$; 2) $i_{1,k}$ and $i_{2,k}$ are neighbors. The schema of I_{new} is apparently c, and elements in I_{new} are in a proximity neighborhood because I_1 is a proximity neighborhood and $i_{2,k}$ is a neighbor of every element of I_1.

C. Computational Complexity Analysis

This subsection examines the strategies for generating candidate co-locations, the evaluation of the multi-resolution pruning strategy, and the effect of noise. First, there are two basic strategies for generating table instances of candidate co-locations, namely the geometric approach and the combinatorial approach. For generating size-2 co-locations, the combinatorial approach ends up being the nest-loop join strategy with an asymptotic complexity of $O(N^2)$, while the geometric approach has the CPU cost[†] of $O(NlogN + M)$, where N is the total number of instances of all features and M is the number of intersections. When the dataset is sparse, the cost of the combinatorial approach will be much higher. However, when generating table instances of co-locations of size 3 or more, the combinatorial approach becomes cheaper than the

[†]The I/O costs of the geometric approach and the combinatorial approach are similar.

geometric approach. This is due to its exploitation of the sort-merge join strategy while keeping each table instance sorted. In a hybrid approach, we pick the cheaper of the two basic strategies in each iteration to achieve the best overall cost.

Second, let us compare the cost of the *Co-location Miner* algorithm with and without the multi-resolution pruning step. Let $T_{mcm}(k)$ and $T_{cm}(k)$ represent the costs of iteration k of the *Co-location Miner* algorithm with and without the multi-resolution pruning.

$$\begin{aligned} T_{mcm}(k) &= T_{apriori\_gen(C(prev,k))} + T_{prune(C(cand,k+1),grid\,data)} + T_{prune(C(sub\_cand,k+1),data)} \\ T_{cm}(k) &= T_{apriori\_gen(C(prev,k))} + T_{prune(C(cand,k+1),data)} \end{aligned} \tag{1}$$

In Equation 1, $T_{apriori\_gen(C(prev,k))}$ represents the cost of *apriori_gen* based on the prevalent co-location set of size k. Here, resolution is not relevant since *apriori_gen* works on the spatial feature level only. $T_{prune(C(cand,k+1),grid\,data)}$ represents the cost for multi-resolution pruning on the coarse level dataset in iteration k. After coarse-level pruning, we only need to search the leftover subset of the original dataset. $T_{prune(C(sub\_cand,k+1),data)}$ represents the cost for fine level instance pruning on the leftover subsets of the original dataset. In addition, $T_{prune(C(cand,k+1),data)}$ represents the cost for fine level instance pruning on the original dataset in iteration k.

The bulk of time is consumed in generating table instances and calculating the participation indexes; thus the ratio can be simplified as:

$$\frac{T_{mcm}(k)}{T_{cm}(k)} \approx \frac{T_{prune(C(cand,k+1),grid\,data)} + T_{prune(C(sub\_cand,k+1),data)}}{T_{prune(C(cand,k+1),data)}} \tag{2}$$

Furthermore, we assume that the average time to generate a table instance in the original dataset is $T_{orig}(k)$ for iteration k and the average time to generate a table instance in the grid dataset is $T_{grid}(k)$ for iteration k. The number of candidate co-locations generated by the *apriori_gen* is $|C_{k+1}|$ and the number of candidate co-locations after the coarse instance level pruning is $|C'_{k+1}|$, Equation 2 can be written as:

$$\frac{T_{mcm}(k)}{T_{cm}(k)} \approx \frac{|C_{k+1}| \times T_{grid}(k) + |C'_{k+1}| \times T_{orig}(k)}{|C_{k+1}| \times T_{orig}(k)} = \frac{T_{grid}(k)}{T_{orig}(k)} + \frac{|C'_{k+1}|}{|C_{k+1}|} \tag{3}$$

The first term of the ratio is controlled by the "clumpiness" (the average number of instances of the spatial features per grid cell) of the locations of spatial features. The second term is controlled by the filtering efficiency of the coarse instance level pruning. When the locations of spatial features are clustered, the sizes of the fine level table instances are much greater than the sizes of the coarse level table instances and the time needed to generate fine level table instances is greater than the time needed to generate coarse level table instances. In our experiments, as described in the next section, we use the parameter m_{clump}, which controls the number of instances clumping together for each spatial feature, to evaluate the first term, and we use the parameter $m_{overlap}$, which represents the possible false candidate ratios, to evaluate the second term. From the formula, we can see that the co-location miner with multi-resolution pruning is likely to be more efficient than the co-location miner without multi-resolution pruning when the locations of spatial features are clustered and the false candidate ratio is high.

V. Experimental Performance Evaluation

Figure 6 describes the experimental setup to evaluate the impact of design decisions on the relative performance of the co-location miner algorithm. We evaluated the performance of the algorithms with synthetic and real-world NASA climate datasets. Synthetic datasets are generated using a methodology similar to methodologies used to evaluate algorithms for mining association rules [2]. Synthetic datasets allow better control towards studying the effects of interesting parameters.

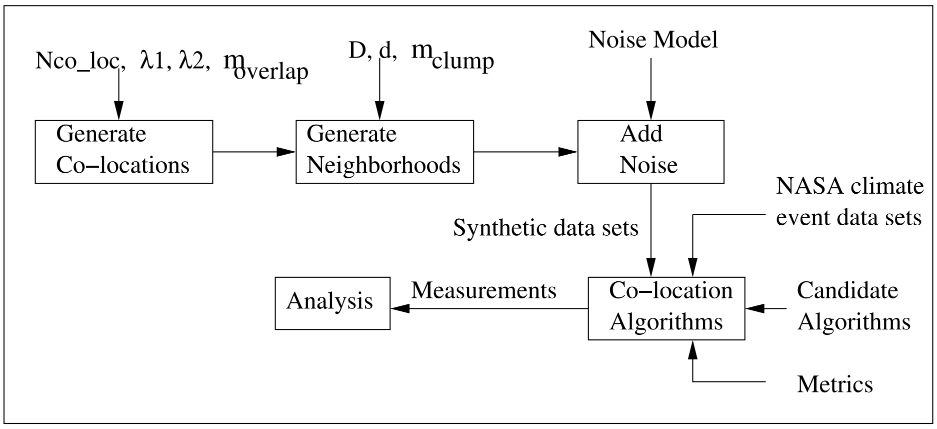


Fig. 6. Experimental Setup and Design

A data-flow diagram of the data generation process is shown in Figure 6. The process began with the generation of core co-location subsets of spatial features. To generate a subset of features, we first chose the size of the subset from a Poisson distribution with mean (λ_1). Then a set of features for this core co-location pattern was randomly chosen. For each core co-location, $m_{overlap}$ maximal co-locations were generated by appending one more spatial feature to a core co-location. The larger $m_{overlap}$ is, the more false candidate *apriori_gen* generates. The size of each table instance of each co-location was chosen from another Poisson distribution with mean λ_2. Next, we generated the set of proximity neighborhoods for co-locations instances using the size of their table instances from the previous step. m_{clump} point locations for each feature in the co-location were embedded inside a proximity neighborhood of size d. The locations of proximity neighborhoods were chosen at random in the overall spatial framework. For simplicity, the shape of the overall spatial framework was a rectangle of size $D_1 \times D_2$ and the size of each proximity neighborhood was $d \times d$. The final step involved adding noise. The model for noise used two parameters, namely the ratio of noise features r_{noise_f} and the number of noise instances p_{noise_n}. Noise was added by generating a set of instances of features from a set of noise features disjoint with the features involving generation of core co-locations and placing the instances at random locations in the global spatial framework.

TABLE I
PARAMETERS USED TO GENERATE THE SYNTHETIC DATA

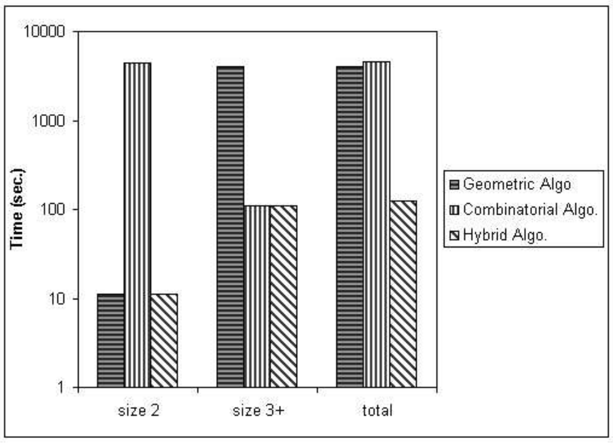
Parameter	**Definition**	**C1**	**C2**
N_{co_loc}	The number of core co-locations	5	4
λ_1	The parameter of the Poisson distribution to define the size of the core co-locations	5	5
λ_2	The parameter of the Poisson distribution to define the size of the table instance of each co-location when $m_{clump} = 1$	50	50
$D_1 \times D_2$	The size of the spatial framework	$10^6 \times 10^6$	$250 \times 1{,}000$
d	The size of the square to define a co-location	10	10
r_{noise_f}	The ratio the of number of noise features over the number of features involved in generating the maximal co-locations	.5	.5
r_{noise_n}	The number of noise instances	50,000	1,000
$m_{overlap}$	The number of co-location generated by appending one more spatial feature for each core co-location	1	1
m_{clump}	The number of instances generated for each spatial feature in a proximity neighborhood for a co-location	1	1

The real-world NASA climate data used in our experiments contain monthly measurements of various monthly numeric climate variables, e.g., precipitation and sea surface temperature, over a period of twelve years, starting in January 1982. Events, such as drought, wet, and hot, are defined via statistical thresholding using mean and standard deviation as detailed in [20].

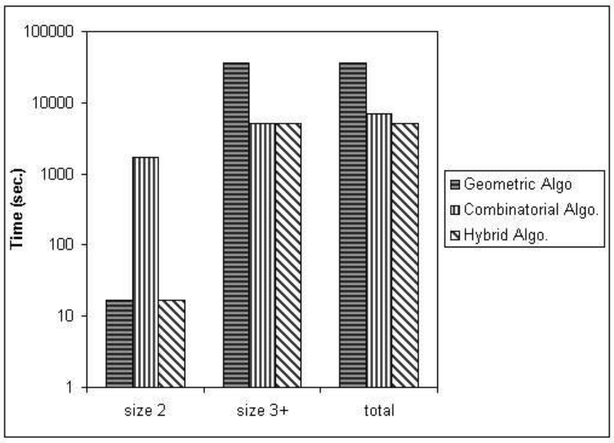
Our experiments were performed on a Sun Ultra 10 workstation with a 440 MHz CPU and 128 Mbytes memory running the SunOS 5.7 operating system.

A. Comparing Strategies for Generating Table Instances

In this subsection, we compare the geometric, the combinatorial, and the hybrid strategies using synthetic and real-world NASA climate datasets. The synthetic dataset, generated using parameter values in column C1 of Table I, used a rectangle spatial framework of size $10^6 \times 10^6$, a square proximity neighborhood of size 10×10, an average co-location size of 5, an average table instance size of 50 when $m_{clump} = 1$, a noise feature ratio of 0.5, a noise number of 50,000, and an overlapping degree of 1.



(a)



(b)

Fig. 7. (a) Relative Performance of Geometric, Combinatorial, and Hybrid Algorithms on the synthetic dataset. (b) Relative Performance of Geometric, Combinatorial, and Hybrid Algorithms on the NASA climate dataset.

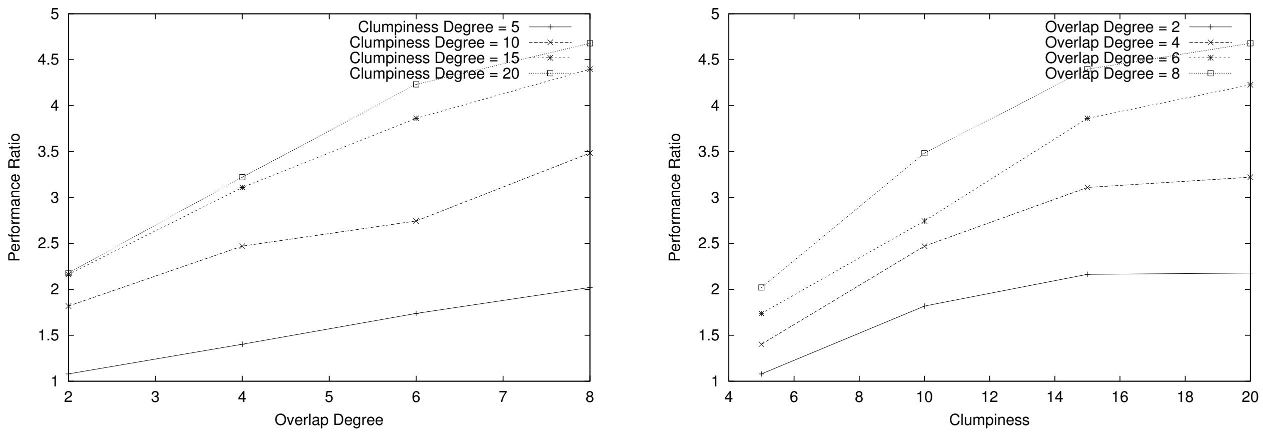
Figure 7 (a) shows the execution times for the three candidates with the prevalence threshold set to 0.9. In the figure, the first column reports the execution time needed to discover co-locations of size 2. As can be seen, the geometric strategy is faster than the combinatorial strategy for generating size-2 co-locations. Spatial-join data structures help the geometric algorithm in this step. Also, the remaining columns in Figure 7 (a) report the total execution time to discover all the co-locations as well as the time to discover co-locations of size 3 or more, given prevalent co-locations of size 2. In these cases, the combinatorial algorithm is orders of magnitude faster than the geometric algorithm. A sort-merge join strategy (e.g *apriori-gen* [2]) helps the combinatorial algorithm. The hybrid strategy uses the geometric algorithm for discovering prevalent co-locations of size 2 and the combinatorial algorithm for discovering larger co-locations. Thus, it is expected to achieve the best overall performance.

Similar trends were also observed for the NASA climate event dataset, as shown in Figure 7 (b). In this experiment, the prevalence threshold is set to 0.3 and the grid size is 4 by 4. All events are extracted at the threshold 1.5 using Z-score transformation.

B. Effect of the Filter

The effect of the multi-resolution pruning filter was evaluated with spatial datasets generated using parameter values shown in column C2 of Table I. We used a rectangular spatial framework of size 250×1000, a square proximity neighborhood of size 10×10, an average co-location size of 5, an average table instance size of 50 when $m_{clump} = 1$ a noise feature ratio of 0.5, a noise number of 1000, a core co-location size of 4, and an overlapping degree of 1. Spatial framework sizes were proportional to the total number of instances to avoid unexpected patterns created by overcrowding of instances. The overlapping degree ($m_{overlap}$) was set from 2 to 8 and the clumpiness measure (m_{clump}) was set from 5 to 20 to generate other datasets. We ran the *Co-location Miner* with and without multi-resolution pruning on these datasets. Prevalence thresholds were set to the estimation of the actual prevalences from the generation of the datasets.

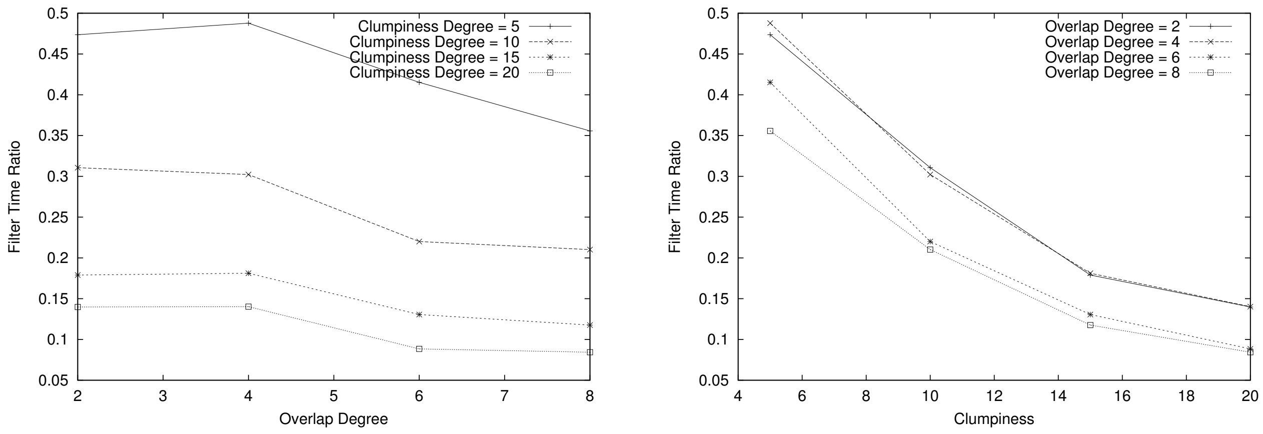
Figure 8 summarizes the performance gain by using multi-resolution pruning. The x-axis represents the overlap degree, which controls the false candidates generated by *apriori_gen* in the first figure or the "clumpiness" of locations of instances of spatial feature type in the second figure. The y-axis represents the ratio of run-time of the *Co-location Miner* without multi-resolution pruning to the run-time with multi-



(a) (b)

Fig. 8. Performance Ratio a) By Overlap Degree b) By Clumpiness Degree

resolution pruning. The results show that, as the degree of overlap and the number of false candidates increase, the running time is reduced by a factor of 1 to 4.5.



(a) (b)

Fig. 9. Filter Time Ratio (a) By Overlap Degree (b) By Clumpiness Degree

Figure 9 summarizes the ratio of the computation time for multi-resolution pruning and for prevalence-based pruning. The x-axis represents the overlap degree or the "clumpiness" of the locations of each feature type. The overhead of multi-resolution pruning as a fraction of prevalence-based pruning decreases when the degree of overlap or clumpiness increases. Clumpiness affects the overhead, reducing it from 0.45 to 0.1.

C. *Effect of Noise*

The base dataset, generated using parameter values in column C1 of Table I, used a rectangle spatial framework of size $10^6 \times 10^6$, a square proximity neighborhood of size 10×10, an average co-location size of 5, an average table instance size of 50 when $m_{clump} = 1$, a noise feature ratio of 0.5, a noise number of 50,000, and an overlapping degree of 1. Then we increased the noise instances up to 800,000 and measured the performance, as shown in Figure 10. The execution time for discovering co-locations of size 2 and 3+ are shown in the figure. The results show that noise level affects the execution time to discover co-locations of size 2 but does not affect the execution time to discover larger co-locations given co-locations of size 2. In other words, noise is filtered out during the determination of co-locations of size 2.

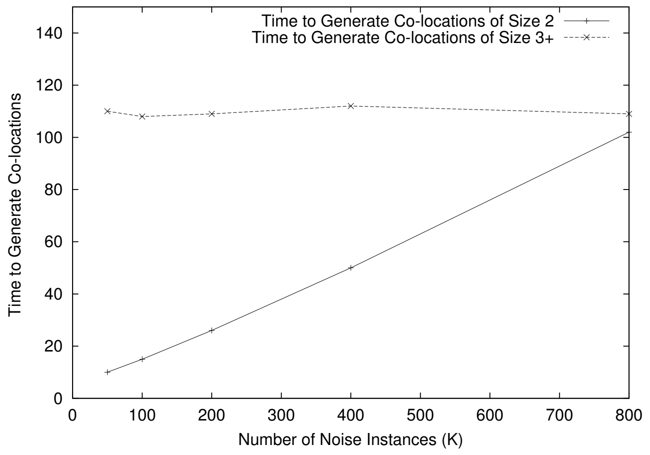


Fig. 10. Noise Effect on the Co-location Miner

VI. Discussion

In this section, we present a detailed comparison of our approach with the closest related work [15] for pattern semantics and algorithmic ideas.

- **Pattern Semantics**. Morimoto [15] defined distance-based patterns called k-neighboring class sets. In this work, the number of instances for a pattern is used as its prevalence measure. However, this measure may not possess an anti-monotone property if the instances overlap; that is, the number of instances may increase with the increase of the pattern size. For example, in Figure 11, there are two

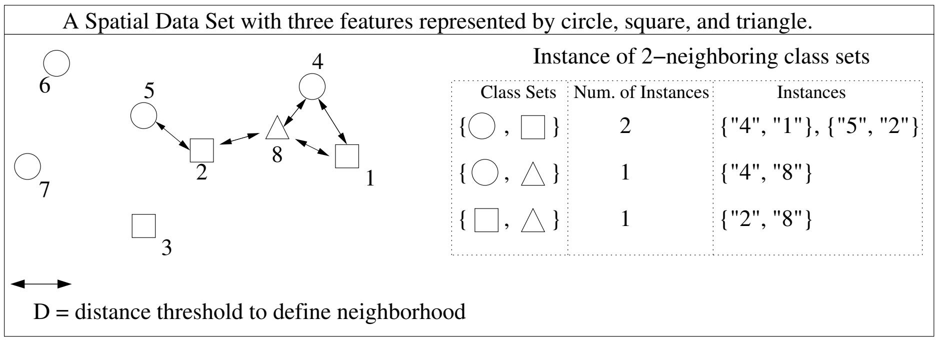

A Spatial Data Set with three features represented by circle, square, and triangle.
Instance of 2−neighboring class sets
Class Sets
Num. of Instances
Instances
2
{"4", "1"}, {"5", "2"}
1
{"4", "8"}
1
{"2", "8"}
D = distance threshold to define neighborhood

Fig. 11. A spatial dataset with three features represented by circle, square, and triangle. This sample data set is from Morimoto's paper [15]. In total, there are eight instances for three features. An edge connects two instances if the distance between these two instances is less than the distance threshold.

overlapping instances: {"2", "8"} and {"1", "8"} of the set {square, triangle}. These two instances share one common instance {"8"} of the triangle feature. Indeed, there is only one instance of the triangle feature. In other words, the number of instances of the set {square, triangle} is greater than the number of instance of its subset {triangle}. To deal with this issue, Morimoto [15] used the following constraint to get the anti-monotone property.

"any point object must belong to only one instance of a k-neighboring class set."

As shown in Figure 11, with this constraint, only one instance {2, 8} is specified for the 2-neighboring class set {square, triangle}. However, this constraint may lead to the following difficulty as described in [15].

"Instances of k-neighboring class set for k $>$ 2 may (be) different depending on the order of the class as added into the class set. Therefore, the support value of a k-neighboring class set for k $>$ 2 may be slightly different"

Our approach does not need the constraint of "any point object must belong to only one instance", since we do not use the number of instances for a pattern as its prevalence measure. We propose the participation index as the prevalence measure, which possesses a desirable anti-monotone property. A unique subset of co-location patterns can be specified using a threshold on the participation index without consideration of algorithmic details such as the order of examination of instances of a co-location. In addition, the correctness and completeness of co-location mining algorithms can be defined using the participation index.

- **Algorithmic Ideas**. We have noted that a direct performance comparison of our algorithm with the algorithm by Morimoto [15] is not very meaningful due to the difference in the prevalence measure, and thus the set of identified patterns. Nonetheless, we provide a comparison of the algorithmic ideas now. Morimoto [15] provided an iterative algorithm for mining neighboring class sets with $k + 1$ features from those with k features. In his algorithm, a nearest neighbor based spatial join was applied in each iteration. More specifically, a geometric technique, a voronoi diagram, was used to take advantage of the restriction that "any point object must belong to only one instance of a k-neighboring class set". This algorithm considers a pure geometric join approach. In contrast, our co-location mining algorithm considers a combinatorial join approach in addition to a pure geometric join approach to generate size k+1 co-location patterns from size-k co-location patterns. Our experimental results show that a hybrid of geometric and combinatorial methods results in lower computation cost than either a pure geometric approach or pure combinatorial approach. In addition, we apply a multi-resolution filter to exploit the spatial auto-correlation property of spatial data for effectively reducing the search space.

VII. Conclusion and Future Work

In this paper, we formalized the co-location problem and showed the similarities and differences between the co-location rules problem and the classic association rules problem as well as the difficulties in using traditional measures (e.g. support, confidence) created by implicit, overlapping and potentially infinite transactions in spatial datasets. We proposed the notion of user-specified proximity neighborhoods in place of transactions to specify groups of items and defined interest measures that are robust in the face of potentially infinite overlapping proximity neighborhoods. A key observation was that some properties of proximity neighborhood cliques obey the downward inclusion property necessary for apriori-based rule mining. The cardinality of table instances does not obey this property but the proposed participation index does, allowing interactive pruning. In addition, the participation index has a spatial statistical interpretation as an upper-bound on the cross-K function, a classical spatial statistical measure of association for binary spatial features. In contrast, related approaches [12], [15] have not provided spatial statistical interpretations of their results.

The Co-location Miner, an algorithm for mining co-location patterns, was presented and analyzed for correctness, completeness and computation cost. Design decisions in the proposed algorithm were evaluated using theoretical and experimental methods. Empirical evaluation shows that the geometric strategy performs much better than the combinatorial strategy when generating size-2 co-locations; however, it becomes slower when generating co-locations with more than 2 features. The hybrid strategy integrates the best features of the above two approaches. Furthermore, when the locations of the features tend to be spatially clustered, which is often true for spatial data due to spatial-autocorrelation, the computation cost of the co-location miner can be significantly reduced with a multi-resolution filter.

Several questions remain open. First, the choice of neighbor relation $\mathcal{R}$ does impact the performance of the proposed algorithms. We plan to examine statistical methods, e.g. inter-instance distance histograms, to develop guidelines for the selection of $\mathcal{R}$. Second, the co-location mining problem should be investigated to account for extended spatial data types, such as line segments and polygons. Also, we considered only boolean features here. In the real world, the features can be categorical and continuous. There is a need to extend the co-location mining framework to handle continuous features. Third, we plan to evaluate the impact of multi-resolution filtering on overall performance of the proposed algorithms using real world datasets which exhibit strong spatial auto-correlation. Finally, if locations of features change over time, it is possible for us to identify some spatio-temporal association patterns. Quantitative association, e.g., (A,A), and quantitative association rules, e.g. ($A \Rightarrow A$), may also be explored in the future.

Acknowledgment

We thank Raymod T. Ng.(University of British Columbia,Canada), Jiawei Han (Simon Fraser University, Canada), James Lesage, and Sucharita Gopal(Boston University) for valuable insights. We also thank C.T. Lu, Weili Wu, and Pusheng Zhang (University of Minnesota) for their valuable feedback. We are grateful to Kim Koffolt for her timely and detailed feedback to help improve the readability of this paper.

REFERENCES

[1] Open GIS Consortium, Inc. *http://www.opengis.org/.*

[2] R. Agarwal and R. Srikant. Fast Algorithms for Mining Association Rules. In *Proc. of the 20th Int'l Conference on Very Large Data Bases*, 1994.

[3] L. Arge, O. Procopiuc, S. Ramaswamy, T. Suel, and J. Vitter. Scalable Sweeping-Based Spatial Join. In *Proc. of the Int'l Conference on Very Large Databases*, 1998.

[4] C. Berge. *Graphs and Hypergraphs.* American Elsevier, 1976.

[5] Y. Chou. *Exploring Spatial Analysis in Geographic Information System.* Onward Press, ISBN: 1566901197, 1997.

[6] N.A.C. Cressie. *Statistics for Spatial Data.* Wiley and Sons, ISBN:0471843369, 1991.

[7] P. Dixon. Ripley's K Function. *http://www.stat.iastate.edu/preprint/articles/2001-18.pdf.*

[8] V. Estivill-Castro and I. Lee. Data Mining Techniques for Autonomous Exploration of Large Volumes of Geo-referenced Crime Data. In *Proc. of the 6th International Conference on Geocomputation*, 2001.

[9] V. Estivill-Castro and A. Murray. Discovering Associations in Spatial Data - An Efficient Medoid Based Approach. In *Proc. of the Second Pacific-Asia Conference on Knowledge Discovery and Data Mining*, 1998.

[10] G. Graefe. Sort-merge-join: An Idea Whose Time Has(h) Passed? In *Proc. of IEEE Conf. on Data Engineering*, 1994.

[11] D. J. DeWitt J. M. Patel. Partition Based Spatial-Merge Join. In *Proc. of the ACM SIGMOD Conference on Management of Data*, June 1996.

[12] K. Koperski and J. Han. Discovery of Spatial Association Rules in Geographic Information Databases. In *Proc. of the 4th International Symposium on Spatial Databases*, 1995.

[13] S. T. Leutenegger and M. A. Lopez. The Effect of Buffering on the Performance of R-Trees. In *Proc. of the Int'l Conference on Data Engineering*, 1998.

[14] S. Lipschutz. *Schaum's Outline of General Topology.* McGraw-Hill Trade, ISBN: 0070379882, 1968.

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

[16] B.D. Ripley. The Second-order Analysis of Stationary Point Process. *Journal of Applied Probability*, 13:255–266, 1976.

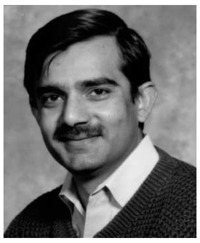
[17] B.D. Ripley. Modelling spatial patterns. *Journal of the Royal Statistical Society*, Series B 39:172–192, 1977.

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

[19] S. Shekhar and Y. Huang. Co-location Rules Mining: A Summary of Results. In *Proc. 7th Intl. Symposium on Spatio-temporal Databases*, 2001.

[20] P. Tan, M. Steinbac, V. Kumar, C. Potter, and S. Klooster. Finding Spatio-Temporal Patterns in Earth Science Data. In *KDD Workshop on Temporal Data Mining*, 2001.

Yan Huang received her B.S. degree in Computer Science from Peking University, Beijing, China, in July 1997 and Ph.D. degree in Computer Science from University of Minnesota, Twin-cities, MN, USA, in July 2003. She is currently an assistant professor at the Computer Science and Engineering Department of University of North Texas, Denton, TX, USA. Her research interests include databases, spatial databases, data mining, and geographic information systems (GIS). She is a member of IEEE Computer Society and ACM SIGMOD.



Shashi Shekhar received the B. Tech degree in Computer Science from the Indian Institute of Technology, Kanpur, India, in 1985, the M.S. degree in Business Administration and the Ph.D. degree in Computer Science from the University of California, Berkeley, CA, USA, in 1989. He is currently a Professor of Computer Science at the University of Minnesota, Minneapolis, MN, USA. His research interests include spatial databases, spatial data mining, geographic and information systems (GIS), and intelligent transportation systems. He is a co-author of a textbook on Spatial Databases (Prentice Hall, 2003, ISBN 0-13-017480-7) and has published over 100 research papers in peer-reviewed journals, books, conferences, and workshops. He is a co-Editor-in-Chief of Geo-Informatica: An International Journal on Advances of Computer Science for GIS and has served on the editorial boards of IEEE Transactions on Knowledge and Data Engineering as well as the IEEE-CS Computer Science & Engineering Practice Board. He also served as a program co-chair of the ACM Intl. Workshop on Advances in Geographic Information Systems, 1996. He is serving as a member of the mapping science committee of the National Research Council National Academy of Sciences (2004-6) and has served as a member of the Board of Directors of the University Consortium on GIS (2003-2004). Dr. Shekhar is a Fellow of the IEEE Computer Society and a member of the ACM.

Hui Xiong received the B.E. degree in Automation from the University of Science and Technology of China, Hefei, China, in 1995 and the M.S. degree in Computer Science from the National University of Singapore, Singapore, in 2000. He is currently a Ph.D. candidate in the Department of Computer Science and Engineering at the University of Minnesota, Minneapolis, MN, USA. His research interests include data mining, spatial databases, geographic and information systems, and statistical computing. He is a student member of the IEEE Computer Society and the ACM.