Discovering Spatial Co-location Patterns: A Summary of Results

Shashi Shekhar[1⋆] and Yan Huang[1⋆]

Computer Science Department, University of Minnesota, 200 Union Street SE, Minneapolis, MN-55455,USA
{shekhar,huangyan}@cs.umn.edu
http://www.cs.umn.edu/research/shashi-group

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 the frequent co-location of a fire ignition source feature with a needle vegetation type feature and a drought feature. The spatial co-location rule problem is different from the association rule problem. Even though boolean spatial feature types (also called spatial events) may correspond to items in association rules over market-basket datasets, there is no natural notion of transactions. This creates difficulty in using traditional measures (e.g. support, confidence) and applying association rule mining algorithms which use support based pruning. We propose a notion of user-specified neighborhoods in place of transactions to specify groups of items. New interest measures for spatial co-location patterns are proposed which are robust in the face of potentially infinite overlapping neighborhoods. We also propose an algorithm to mine frequent spatial co-location patterns and analyze its correctness, and completeness. We plan to carry out experimental evaluations and performance tuning in the near future.

1 Introduction

Widespread use of spatial databases [8,21,22,28] is leading to an increasing interest in mining interesting and useful but implicit spatial patterns [7,13,17,20, 26]. Efficient tools for extracting information from geo-spatial data, the focus of this work, are crucial to organizations which make decisions based on large spatial datasets. These organizations are spread across many domains including ecology and environmental management, public safety, transportation, public health, business, travel and tourism [3,12,15,9,23,26,29]. We will focus on the application domain of ecology where scientists are interested in finding frequent co-occurrence among boolean spatial features, e.g. drought, El Nino, substantial

⋆ Supported in part by the Army High Performance Computing Research Center under the auspices of the Department of the Army, Army Research Laboratory Cooperative agreement number DAAH04-95-2-0003/contract number DAAH04-95-C-0008.

C.S. Jensen et al. (Eds.): SSTD 2001, LNCS 2121, pp. 236–256, 2001.

increase in vegetation, substantial drop in vegetation, extremely high precipitation. etc.

Association rule finding [11]is an important data mining technique which has helped retailers interested in finding items frequently bought together to make store arrangements, plan catalogs, and promote products together. Spatial association rules [14] are spatial cases of general association rules where at least one of the predicates is spatial. Association rule mining algorithms [1,2,10] assume that a finite set of disjoint transactions are given as input to the algorithms. In market basket data, a transaction consist of a collection of item types purchased together by a customer. Algorithms like *apriori* [2] can efficiently find the frequent itemsets from all the transactions and association rules can be found from these frequent itemsets.

Many spatial datasets consist of instances of a collection of boolean spatial features (e.g., drought, needle leaf vegetation). While boolean spatial features can be thought of as item types, there may not be an explicit finite set of transactions due to the continuity of the underlying space. If spatial association rule discovery is restricted to a reference feature (e.g., city) [14] then transactions can be defined around the instances of this reference feature. Generalizing this paradigm to the case where no reference feature is specified is non-trivial. Defining transactions around locations of instances of all features may yield duplicate counts for many candidate associations. Defining transactions by partitioning space independent of data distribution is an alternative. However, imposing artificial transactions via space partitioning often undercounts instances of tuples intersecting the boundaries of artificial transactions or double-counts instances of tuples co-located together.

In this paper, we give different interpretation models for the spatial co-location rules accompanied by representative application domains. We define the spatial co-location rule as well as interest measures, and propose an algorithm to find co-location rules. We provide a detailed analysis of the proposed algorithm for correctness, completeness, and computational efficiency. We are working on an experimental evaluation in the context of an ecological application with datasets from NASA.

1.1 An Illustrative Application Domain

Many ecological datasets [16,18] consist of raster maps of the Earth at different times. Measurement values for a number of variables (e.g., temperature, pressure, and precipitation) are collected for different locations on Earth. Maps of these variables are available for different time periods ranging from twenty years to one hundred years. Some variables are measured using sensors while others are computed using model predictions.

A set of events, i.e., boolean spatial features, are defined on these spatial variables. Example events include drought, flood, fire, and smoke. Ecologists are interested in a variety of spatio-temporal patterns including co-location rules. Co-location patterns represent frequent co-occurrences of a subset of boolean spatial

Table 1. Examples of interesting spatio-temporal ecological patterns. Net Primary Production (NPP) is a key variable for understanding the global carbon cycle and the ecological dynamics of the Earth

Pattern #	Variable A	variable B	Examples of interesting patterns
P1	Cropland Area	Vegetation	Higher cropland area alters NPP
P2	Precipitation Drought Index	Vegetation	Low rainfall events lead to lower NPP
P3	Smoke Aerosol Index	Precipitation	Smoke aerosols alter the likelihood of rainfall in a nearby region
P4	Sea Surface Temperature	Land Surface Climate and NPP	Surface ocean heating affects regional terrestrial climate and NPP

features. Examples of interesting co-location patterns in ecology are shown in Table 1.

The spatial patterns of ecosystem datasets include:

a. **Local co-location patterns** represent relationships among events at a common location, ignoring the temporal aspects of the data. Examples from the ecosystem domain include patterns P1 and P2 of Table 1. These patterns can be discovered using algorithms [2] for mining classical association rules.

b. **Spatial co-location patterns** represent relationships among events happening in different and possibly nearby locations. Examples from the ecosystem domain include patterns P3 and P4 of Table 1.

Additional varieties of co-location patterns may exist. Furthermore, the temporal nature of general ecosystem data gives rise to many other time related patterns. We focus on the above co-location patterns in this paper.

1.2 Related Work and Our Contributions

Approaches to discover co-location rules in the literature can be categorized into two classes, namely spatial statistics and association rules. Spatial statistics-based [5,6] approaches use measures of spatial correlation to characterize the relationship between different types of spatial features. Measures of spatial correlation include chi-square tests, correlation coefficients, and regression models as well as their generalizations using spatial neighborhood relationships. Computing spatial correlation measures for all possible co-location patterns can be computationally expensive due to the exponential number of candidates given a large collection of spatial boolean features.

Association rule-based approaches [14] focus on the creation of transactions over space so that an *apriori* like algorithm [2] can be used. Some practitioners use ad-hoc windowing to create transactions, leading to problems of under counting or over counting in determination of prevalence measures, e.g., support. Another approach is based on the choice of a reference spatial feature [14] to mine all association rules of the following form:

$$is\_a(X, big\_city) \wedge adjacent\_to(X, sea) \Rightarrow close\_to(X, us\_boundary)(80\%)$$

where at least one of the predicates is a spatial predicate. Users are asked to specify the spatial features which they are interested in first in a form that specifies the main spatial feature of interest set and the relevant spatial features. An example is finding within the map of British Columbia the strong spatial association relationships between large towns and other "near_by" spatial features including mines, country boundary, water and major highways. The set of large towns is the main spatial feature set of interest while the mines, country boundary, etc. are the relevant spatial features. The algorithm [14] uses two-step computation: first, association rules are generated at a coarse level, e.g. , g_close_to, which is efficient by using R-tree or fast MBR (Minimum Bounding Rectangle) techniques, and then only the spatial features with support higher than minimum support are passed to fine level(e.g. adjacent_to) rule generation. The association rules are derived using the *apriori* [2] algorithm. This approach does not find more general co-location patterns involving no reference spatial feature on the left-hand side of the association rules. For example, consider the co-location pattern of (drought, pine-needle-vegetation) being in a neighborhood implying high probability of a fire-ignition event.

Contributions: This paper makes following contributions. First, it defines event centric spatial co-location patterns using neighborhoods in place of transactions for spatial application domains with no single reference spatial feature. Second, it defines a new spatial measure of conditional probability as well as a new monotonic measure of prevalence to allow iterative pruning. Third, it proposes the **Co-location Miner** algorithm, a correct and complete algorithm to mine prevalent co-location rules. The proposed algorithm has innovative ideas such as *generalized apriori_gen* to efficiently enumerate the neighborhoods of interest. It is also space efficient in that it discards intermediate results at the earliest opportunity. Fourth, the paper provides proofs of correctness and completeness of the **Co-location Miner** algorithm in the presence of various performance optimizations. Finally, the paper provides a detailed complexity analysis of the proposed algorithm.

1.3 Outline and Scope

Section 2 formulates the problem of mining co-location rules. Section 3 describes approaches of modeling co-location problems and their associated prevalence and conditional probability measures. Section 4 describes the challenges in designing an efficient algorithm to mine event centric co-location patterns and proposes the algorithm **Co-location Miner**. Section 5 provides analysis of the proposed algorithm in the areas of correctness, completeness, and computational efficiency. Finally, Section 6 presents the conclusion and future work.

The scope of this paper is limited to co-location rules in two dimensional Euclidean space. Issues beyond the scope of the paper include other spatial patterns, spatio-temporal co-locations as well as system implementation issues such as selection of index, buffering policy etc.

2 Problem Formulation and Basic Concepts

In a market basket data mining scenario, association rule mining is an important and successful technique. We recall a typical definition from the literature. Let $I = \{i_1, i_2, \ldots, i_m\}$ be a set of literals, called items. Let D be a set of transactions, where each transaction T is a set of items such that $T \subseteq I$. An association rule is of the form $X \Rightarrow Y$, where $X \subseteq I$, $Y \subseteq I$, and $X \cap Y = \phi$. $Pr(X)$ is the fraction of transactions containing X. $Pr(X \cup Y)/Pr(X)$ is called **confidence** of the rule and $Pr(X \cup Y)$ is called **support** of the rule [1]. An association is a subset of items whose support is above the user specified minimum support. A popular example of an association rule is $Diapers \Rightarrow Beer$ which means "People buying diapers tend to buy beer." Substantial literature is available on techniques for mining association rules [1,2,11,19,24,25,27].

The spatial co-location problem looks similar but in fact is very different from the association rule mining problem because of the lack of transactions. In market basket data sets, transactions represent sets of item types bought together by customers. The purpose of mining association rules is to identify frequent item sets for planning store layouts or marketing campaigns. In the spatial co-location rule mining problem, transactions are often not explicit. The transactions in market basket analysis are independent of each other. Transactions are disjoint in the sense of not sharing instances of item types. In contrast, the instances of Boolean spatial features are embedded in a space and share a variety of spatial relationships (e.g. neighbor) with each other. We formalize the event centric co-location rule mining problem as follows:

Given:

1) a set T of K Boolean spatial feature types T=$\{f_1, f_2, \ldots, f_K\}$

2) a set of N instances P=$\{p_1 \ldots p_N\}$, each $p_i \in P$ is a vector <instance-id, spatial feature type, location> where spatial feature type $\in$ T and location $\in$ spatial framework S

3) A neighbor relation R over locations in S 4) Min prevalence threshold value, min conditional probability threshold

Objectives:

1) **Completeness**: We say an algorithm is complete if it finds all spatial co-location rules which have prevalences and conditional probabilities greater than user specified thresholds.

2) **Correctness**:We say an algorithm is correct if any spatial co-location rules it finds has prevalence and conditional probabilities greater than user specified thresholds.

3) **Computational efficiency**: IO cost and CPU cost to generate the co-location rules should be acceptable

Find:

Co-location rules with high prevalence and high conditional probability

Constraints:

1) R is symmetric and reflexive

2) Monotonic prevalence measure

3) Conditional probability measures are specified by the event centric model

4) Sparse data set, i.e., the number of instance of any spatial features is << cardinality(P)

3 Approaches of Modeling the Co-location Rules Problem

Given the difficulty in creating explicit disjoint transactions from continuous spatial data, this section defines approaches to model co-location rules. We will use Fig. 1 as an example spatial dataset to illustrate different models. In Fig. 1, a uniform grid is imposed on the underlying spatial framework. For each grid l, its neighbors are defined to be the nine adjacent grids (including l). Spatial feature types are labeled beside their instances. We define the following basic concepts to facilitate the description of the different models.

Definition 1 *A* **co-location** *is a subset of boolean spatial features.*

Definition 2 *A* **co-location rule** *is of the form:* $C_1 \rightarrow C_2(p, cp)$ *where* C_1 *and* C_2 *are co-locations,* p *is a number representing the prevalence measure and* cp *is a number measuring conditional probability.*

The prevalence measure and the conditional probability measure, called interest measures, are defined differently in different models. They will be described shortly.

The **reference feature centric model** is relevant to application domains focusing on a specific boolean spatial feature, e.g. cancer. Domain scientists are interested in finding the co-locations of other task relevant features (e.g. asbestos,

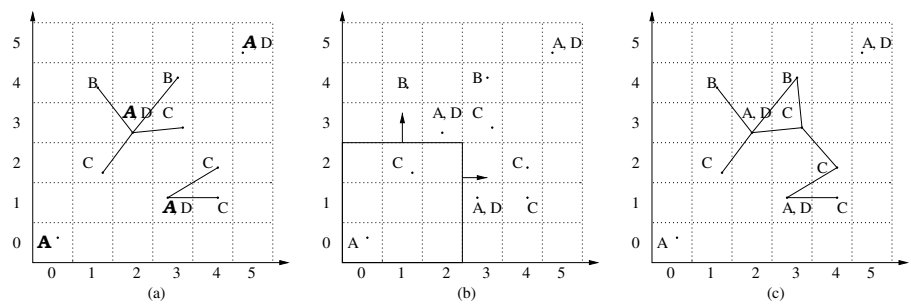

Fig. 1. Spatial dataset to illustrate different co-location models. Spatial feature types are labeled beside their instances. The 9 adjacent cells of a cell l (including l) are defined to be l's neighbors. a) Reference feature centric model. The instances of A are connected with their neighboring instances of B and C by edges. b) Window centric model. Each 3 X 3 window corresponds to a transaction. c) Event centric model. Neighboring instances are joined by edges.

Table 2. Reference feature centric model: transactions are defined around instances of feature A relevant to B and C in Fig. 1a

Instance of A	Transaction
(0,0)	ø
(2,3)	$\{B, C\}$
(3,1)	$\{C\}$
(5,5)	ø

Table 3. Interest measures for different models

Model	Items	transactions defined by	Interest measures for $C_1 \rightarrow C_2$	
			Prevalence	Conditional probability
local	boolean feature types	partitions of space	fraction of partitions with $C_1 \cup C_2$	$Pr(C_2$ in a partition given C_1 in the partition)
reference feature centric	predicates on reference and relevant features	instances of reference feature C_1 and C_2 involved with	fraction of instance of reference feature with $C_1 \cup C_2$	$Pr(C_2$ is true for an instance of reference features given C_1 is true for that instance of reference feature)
window centric	boolean feature types	possibly infinite set of distinct overlapping windows	fraction of windows with $C_1 \cup C_2$	$Pr(C_2$ in a window given C_1 in that window)
event centric	boolean feature types	neighborhoods of instances of feature types	participation index of $C_1 \cup C_2$	$Pr(C_2$ in a neighborhood of C_1)

other substances) to the reference feature. This model enumerates neighborhoods to "materialize" a set of transactions around instances of the reference spatial feature. A specific example is provided by the spatial association rule [14].

For example, in Fig. 1a, let the reference feature be A, the set of task relevant features be B and C, and the set of spatial predicates include one predicate named "*close_to*". Let us define *close_to*(a, b) to be true if and only if b is a's neighbor. Then for each instance of spatial feature A, a transaction which is a subset of relevant features $\{B, C\}$ is defined. For example, for the instance of A at (2,3), transaction $\{B, C\}$ is defined because the instance of B at (1,4) (and at (3,4)) and instance of C at (1,2) (and at (3,3)) are *close_to* (2,3). The transactions defined around instances of feature A are summarized in Table 2.

With "materialized" transactions, the support and confidence of the traditional association rule problem [2] may be used as prevalence and conditional probability measures as summarized in Table 3. Since one out of two non-empty transactions contains instances of both B and C and one out of two

non-empty transactions contain C in Table 2, an association rule example is: $is\_type(i, A) \wedge \exists j\, is\_type(j, B) \wedge close\_to(j, i) \rightarrow \exists k\, is\_type(k, C) \wedge close\_to(k, i)$ with $\frac{1}{1} * 100\% = 100\%$ probability.

The **window centric model** is relevant to applications like mining, surveying and geology, which focus on land-parcels. A goal is to predict sets of spatial features likely to be discovered in a land parcel given that some other features have been found there. The window centric model enumerates all possible windows as transactions. In a space discretized by a uniform grid, windows of size kXk can be enumerated and materialized, ignoring the boundary effect. Each transaction contains a subset of spatial features of which at least one instance occurs in the corresponding window. The support and confidence of the traditional association rule problem may again be used as prevalence and conditional probability measures as summarized in Table 3. There are 16 3X3 windows corresponding to 16 transactions in Fig. 1b. All of them contain A and 15 of them contain both A and B. An example of an association rule of this model is: *an instance of type A in a window* $\rightarrow$ *an instance of type B in this window* with $\frac{15}{16} = 93.75\%$ probability. A special case of the window centric model relates to the case when windows are spatially disjoint and form a partition of space. This case is relevant when analyzing spatial datasets related to the units of political or administrative boundaries (e.g. country, state, zip-code). In some sense this is a local model since we treat each arbitrary partition as a transaction to derive co-location patterns without considering any patterns across partition boundaries. The window centric model "materializes" transactions in a different way from the reference feature centric model.

The **event centric model** is relevant to applications like ecology where there are many types of boolean spatial features. Ecologists are interested in finding subsets of spatial features likely to occur in a neighborhood around instances of given subsets of event types. For example, let us determine the probability of finding at least one instance of feature type B in the neighborhood of an instance of feature type A in Fig. 1c. There are four instances of type A and only one of them have some instance(s) of type B in their 9-neighbor adjacent neighborhoods. The conditional probability for the co-location rule is: *spatial feature A at location* $l \rightarrow$ *spatial feature type B in 9-neighbor neighborhood is 25%.*

Neighborhood is an important concept in the event centric model. Given a reflexive and symmetric neighbor relation R, we can define neighborhoods of a location l as follows:

Definition 3 *A* **neighborhood** *of l is a set of locations $L = \{l_1, \ldots, l_k\}$ such that l_i is a neighbor of l i.e. $(l, l_i) \in R(\forall i \in 1 \ldots k)$.*

This definition satisfies the following two conditions from Topology [28]:

T1. Every location is in some neighborhood because of the reflective neighbor relationship.

T2. The intersection of any two neighborhoods of any location l contains a neighborhood of l.

We generalize the neighborhood definition to a collection of locations.

Definition 4 *For a subset of locations L' if L' is a neighborhood of every location in* $L = \{l_1, \ldots, l_k\}$ *then L' is a* **neighborhood** *of L.*

In other words, if every l_1 in L' is a neighbor of every l_2 in L, then L' is a neighborhood of L.

The definition of neighbor relation R is an input and is based on the semantics of application domains. It may be defined using topological relationships (e.g. connected, adjacent), metric relationships (e.g. Euclidean distance) or a combination (e.g. shortest-path distance in a graph such as road-map). In general there are infinite neighborhoods over continuous space and it may not be possible to materialize all of them. But we are only interested in the locations where instances of spatial feature types (events) occurs. Even confined to these locations, enumerating all the neighborhoods incurs substantial computational cost because support-based pruning cannot be carried out before the enumeration of all the neighborhoods is completed and the total number of neighborhoods is obtained. Thus the participation index is proposed to be a prevalence measure as defined.

Definition 5 $I = \{i_1, \ldots, i_k\}$ *is a* **row instance** *of a co-location* $C = \{f_1, \ldots, f_k\}$ *if* i_j *is an instance of feature* $f_j (\forall j \in 1, \ldots, k)$ *and I is a neighborhood of I itself.*

In other words, if elements of I are neighbors to each other, then I is an instance of C. For example, {(3,1),(4,1)} is an instance of co-location $\{A, C\}$ in Fig 1c using a 9-neighbor relationship over cells of a grid.

Definition 6 *The* **table instance** *of a co-locatin* $C = f_1, \ldots, f_k$ *is the collection of all its row instances.*

Definition 7 *The* **participation ratio** $pr(C, f_i)$ *for feature type* f_i *of a co-location* $C = \{f_1, f_2, \ldots, f_k\}$ *is the fraction of instances of* f_i *which participate in any row instance of co-location C. It can be formally defined as* $\frac{|distinct(\pi_{f_i}(all\ row\ instances\ of\ C))|}{|instances\ of\ \{f_i\}|}$ *where* π *is a relational projection operation.*

For example, in Fig 1c, instances of co-location $\{A, B\}$ are {(2,3), (1,4)} and {(2,3)), (3,4)}. Only one instance (2,3) of spatial feature A out of four participates in co-location $\{A, B\}$. So $pr(\{A, B\}, A) = \frac{1}{4} = .25$.

Definition 8 *The* **participation index** *of a co-location* $C = \{f_1, f_2, \ldots, f_k\}$ *is* $\prod_{i=1}^{k} pr(C, f_i)$.

In Fig. 1c, participation ratio pr($\{A, B\}, A$) of feature A in co-location $\{A, B\}$ is .25 as calculated above. Similarly $pr(\{A, B\}, B)$ is 1.0. The participation index for co-location $\{A, B\}$ is .25 X 1.0 = .25.

Note that participation index is monotonically non-increasing with the size of the co-location increasing since any spatial feature participates in a row instance

of a co-location C of size $k+1$ will participates in a row instance of a co-location C' where $C' \subseteq C$.

The conditional probability of a co-location rule $C_1 \rightarrow C_2$ in the event centric model is the probability of finding C_2 in a neighborhood of C_1 or it can be formally defined as:

Definition 9 *The conditional probability of a co-location rule* $C_1 \rightarrow C_2$ *is* $\frac{|distinct(\pi_{C_1}(all\ row\ instances\ of\ C_1 \cup C_2))|}{|instances\ of\ C_1|}$ *where* π *is a projection operation.*

4 Event Centric Approach and Algorithms

There are numerous challenges in mining spatial co-location patterns in the event centric model. These include efficient enumeration of row instances of co-locations, efficient computation of prevalence for pruning, efficient computation of conditional probability, and generation of co-location rules. We briefly discuss these in Section 4.1 before describing the **Co-location Miner** algorithm in Section 4.2.

4.1 Challenges and Solutions

Neighborhood (i.e. co-location row instance) enumeration is a major challenge and a key part of any co-location mining algorithm. It can be addressed via a combinatorial method like *apriori* [2] or a geometric approach e.g. *spatial-self-join*. A combinatorial method formularizes the problem as a smart clique enumeration problem from a graph based on the definition of neighbors. A geometric *spatial join* approach using a plane sweep method scans the underlying space and stops at anchor points to collect neighborhood information. Both methods may use optimizations at system level via spatial database techniques such as spatial indexes. We propose a combinatorial approach in the next section and a plane sweeping method is under exploration.

Co-location row instances are enumerated before measures of prevalence and conditional probability are computed at co-location level. Computing prevalences and conditional probabilities from instances of co-locations is non-trivial, especially when the number of spatial features is large as well. Computation of these measures may require efficient strategies for projection and duplicate elimination. We use bitmaps to eliminate duplicates and calculate the participation index in our **Co-location Miner** algorithm, which will be introduced shortly.

A spatial co-location rule's conditional probability measure may not be calculated directly from its prevalence measures (e.g. participation index). For a candidate co-location $C = \{f_1, f_2, \ldots, f_k\}$ we need to calculate the conditional probabilities for each possible co-location rule $C' \rightarrow C - C'$ where C' is an arbitrary subset of C. An important finding is that we only need the table instance for co-location C and cardinalities of co-locations of size $< |C|$ to calculate the conditional probabilities. We generate prevalent co-locations in order of increasing sizes. We then generate all co-location rules $C' \rightarrow C - C'$ for each prevalent

co-location C in the current iteration and each non-trivial subset C' of C. By generating prevalent co-location rules in increasing sizes , we always have the cardinalities of the table instance of co-location C' available to calculate the conditional probability of the candidate co-location rule $C' \rightarrow C - C'$.

4.2 Co-location Miner Algorithm

Co-location Miner is an algorithm to generate all the co-location rules with prevalences and conditional probabilities above a use defined *min_prevalence* and *min_cond_prob*.

Co-location Miner
input:
1) K boolean spatial instance types and their instances:
P=$\{ < f_i, \{I\} > | f_i \in \{f_1, f2, \ldots, f_K\}, I \subseteq S$ where S is the set of all interested locations$\}$
2) A symmetric and reflexive neighbor relation R
3) A user specified minimum threshold prevalence measure (*min_prevalence*)
4) A user specified minimum conditional probability (*min_cond_prob*)
output:
co-location rule sets with partition index $>$ *min_prevalence* and
conditional probability $>$ *min_cond_prob*
method:
1) prevalent size 1 co-location set along with their table instances= P
2) Generate size 2 co-location rules
3) **for** size of co-locations in (2, 3, ... , $K-1$) **do**
4) Generate candidate prevalent co-locations using the *generalized apriori_gen* algorithm
5) Generate table instances and prune based on neighborhood
6) Prune based on prevalence of co-locations
7) Generate co-location rules
8) **end**;

Explanation of the detailed steps of the algorithm

Step 1 initializes the prevalent size 1 co-location set with the input P of the algorithm. The participation indexes of singleton co-locations are 1 and all singleton co-locations are prevalent.

Example: Fig. 2a shows the size 1 co-locations, i.e. A, B, and C, and their table instances for example dataset in Fig. 3.

Step 2 generates prevalent co-location rules of size 2. Due to lack of pruning for singleton co-locations, it is more efficient to use *spatial join* using neighbor relationship in place of *generalized apriori_gen* and then neighbor-based pruning like in generation of co-location rules of size 3 or more. The spatial inner join of the instances of all spatial features will produce pairs of instances with neighbor relation R. A minor modification of a sweeping-based spatial join [4], which eliminates pairs of instances in a neighborhood with the same spatial feature type will produce all table instances of size 2 co-locations. We order

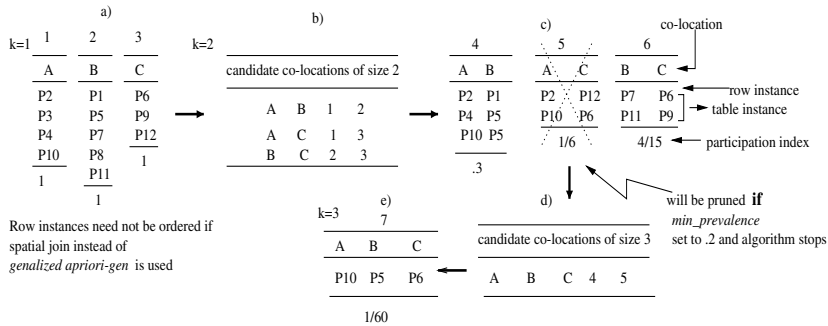

Fig. 2. Co-location Miner Algorithm Illustration on Example Database

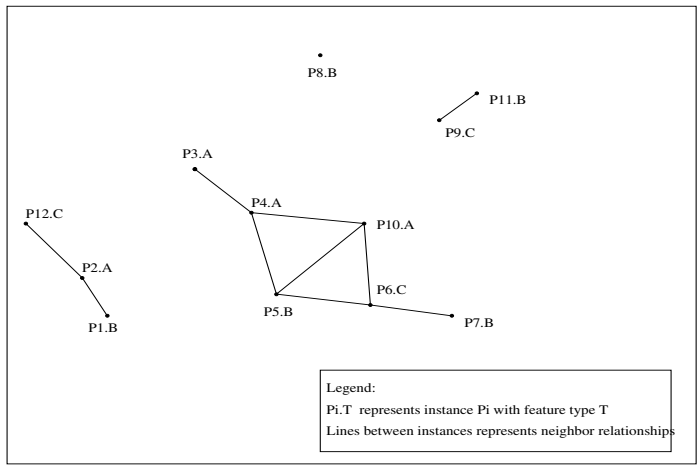

Fig. 3. Example Database

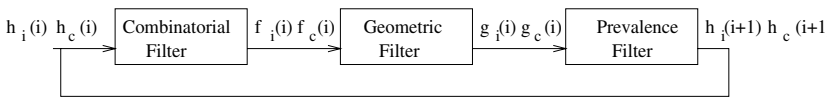

Fig. 4. Algorithm Co-location Miner Illustration

the row instances in the table instance of each co-location in increasing lexicographic order. Finally, we compute the participation index of each co-location, do prevalence-based pruning if necessary, calculate and maintain the cardinality of each prevalent co-location, and generates co-locations as described in the following steps in the loop.

Example: Figure 2b may be produced by sweeping-based spatial join implementation of the query:

select p', p''
from $\{p_1, \ldots, p_{12}\}$ p', $\{p_1, \ldots, p_{12}\}$ p''

where p'.feature $\neq p''$.feature, $p' \leq p''$, $(p', p'') \in R$

Step 3 to **Step 8** loops through 2 to $K-1$ to generate prevalent co-locations of size 3 or more, iterating on increasing values of sizes of co-locations. It breaks whenever an empty co-location set of some size is generated.

Step 4 use *generalized apriori_gen* to generate candidate prevalent co-locations of size $k+1$ from prevalent co-locations of size k along with their table instances. The *generalized apriori_gen* function is an adoption of the *apriori_gen* algorithm (see **Appendix**) function of the *apriori* [2]. The *generalized apriori_gen* function takes as argument C_k, the set of all prevalent size k co-locations. The function works as follows. First, in the *join* step, we join C_k with C_k:

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

The last two columns (id$_1$ and id$_2$) of table C_{k+1} keep track of the table instances of any pair of co-locations of size k whose *join* produce a co-location of size $k+1$.

Next, in the *prune* step, we delete all co-locations $c \in C_{k+1}$ such that some k-subset of c is not in C_k(Recall that it is also done in *apriori_gen* [2] because of the monotonicity property of prevalence measure):

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

Example: If the size 2 co-location set is $\{\{A, B\}, \{A, C\}\}$, the *join* step will produce $\{\{A, B, C\}\}$. The *prune* step will delete $\{A, B, C\}$ from $\{\{A, B, C\}\}$ because $\{B, C\}$ is not a prevalent co-location of size 2.

Step 5 generates all the table instances of candidate co-locations of size $k+1$ which passed the filter of step 4. Co-locations with empty table instances will be eliminated from the candidate prevalent co-location set of size $k+1$. It takes size $k+1$ candidate co-location set C_{k+1} as an argument and works as follows.

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

Then all co-locations with empty table instance will be eliminated from C_{k+1}. **Example:** In Fig. 2, Table 4 of co-location $\{A, B\}$ and Table 5 of co-location $\{A, C\}$ are joined to produce table instance of co-location $\{A, B, C\}$ because co-location $\{A, B\}$ and co-location $\{A, C\}$ were joined in *generalized apriori_gen* to produce co-location $\{A, B, C\}$ in the previous step. In the example, row instance $\{P_{10}, P_5\}$ of table 4 and row instance $\{P_{10}, P_6\}$ of table 5 are joined to generate row instance $\{P_{10}, P_5, P_6\}$ of co-location $\{A, B, C\}$ (Table 7).

Step 6 calculates the participation indexes for all candidate co-locations in C_{k+1} and it prunes co-locations using prevalence threshold. Computation of the participation index for a co-location C requires scanning of its table instance to compute participation ratios for each feature in the co-location. This computation can be modeled as *project-unique* operation on columns of the table instance of C. This can be accomplished by keeping a bitmap of size |instance of 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 corresponding bits in each bitmap. By summarizing total number of 1s (p_{f_i}) in each bitmap, we get the participation ratio of each feature f_i (divide p_{f_i} by |instance of f_i|. In Fig. 2c, 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= (1,0,1,1) and b_B = (1,1,0,0,0). Three unique instances P_2, P_4, and P_{10} of instance A out of 4 participate in co-location $\{A, B\}$. So the participation ratio for A is .75. Similarly, the participation ratio for B is .4. The participation index is .75x.4 = .3. After we get the participation indexes, prevalence-based pruning is carried out and non-prevalent co-locations and their table instances are deleted from the candidate prevalent co-location sets. For each left 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.

Step 7 generates all the co-location rules with user defined *min_prev* and *min_cond_prob*.

For each prevalent co-location C, we enumerate every subset C' of C and calculate the conditional probability measure for the spatial co-location rule: $C' \rightarrow C - C'$. 1) We project the table instance of C on C' to get CC. 2) Calculate the cardinality of CC after duplicate elimination to get the N_p. 3) Divide N_p by the cardinality of C' (which has already been calculated and kept in the previous iterations) to get the conditional probability. 4) Produce:$C' \rightarrow C - C'$ if the conditional probability is above user specified threshold.

5 Analysis

This section presents an analysis of the proposed algorithm for correctness, completeness and computational efficiency.

5.1 Completeness

Lemma 1: algorithm Co-location Miner is complete:
Proof:

Step 1 initializes the size 1 co-locations to be all the feature types and the table instance of a co-location to include all its instances. It is complete.

The *spatial join* of **Step 2** will produce all pairs (p', p'') of instances where p'.feature $\neq p''$.feature and p' and p'' are neighbors. Any row instance of any size 2 co-location satisfying those two conditions in the join predicate will be generated. The proof of completeness under pruning in subsequent steps of size 2 co-location rules is the same as the proof of completeness of step 3 to step 7 which will be shown shortly.

The loop from **Step 3** to **Step 8** iterates through all the co-locations of size 2 to $k-1$ to produce co-locations of size 3 or more. It only breaks when an empty co-location set is produced. We prove the completeness of the substeps inside the loop of iteration k as follows. *Generalized apriori_gen* algorithm in **Step 4** will not miss any prevalent co-locations for the following reasons. According to the monotonicity of the participation index measure, every subset of a prevalent co-location $C = \{f_1, \ldots, f_{k+1}\}$ is a prevalent co-location in the previous iteration and in particular, $C_1 = \{f_1, \ldots, f_k\}$ and $C_2 = \{f_1, \ldots, f_{k-1}, f_{k+1}\}$ are prevalent co-locations. The *join* step of step 4 will produce C. The *prune* step only deletes candidate prevalent co-locations whose one or more subset is not prevalent. It is not possible for the pruned candidate prevalent co-locations to be prevalent due to the monotonicity of the participation index. So, the *prune* step will not destroy the completeness of the algorithm. **Step 5** joins the table instances of C_1 and C_2 to produce the table instance of C where C_1, C_2, and C are as defined above. According to the neighborhood definition, any subset of a neighborhood is a 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. So, step 5 is complete. **Step 6** guarantees the completeness by the correct calculation of the participation index and use of monotonicity of participation index measure, which says any superset of a non-prevalent co-location is non-prevalent and can be pruned. In **Step 7**, enumeration of the subsets of each of the prevalent co-locations ensures no spatial co-location rules with both high prevalence and high conditional probabilities is missed.

5.2 Correctness

Lemma 2: Co-location Miner algorithm is correct:
Proof:

Step 1 is correct because the participation indexes of singleton co-locations are 1 and all singleton co-local ions are prevalent.

The correctness of **step 2** with respect to size 2 prevalent co-locations is based on the fact that neighborhood-based spatial join correctly computes the union of table instances of all size 2 co-locations.

Table 4. Notations for Cost Analysis

$h_i(i)$	average number of row instances in the table instance of a prevalent co-locations of size i
$h_c(i)$	total number of prevalent co-locations of size i
$f_i(i)$	average number of row instances in the table instance of a candidate prevalent co-location of size $i+1$ in iteration i after *generalized apriori_gen* algorithm in step 4 (combinatorial filter)
$f_c(i)$	total number of candidate prevalent co-locations of size $i+1$ in iteration i after *generalized apriori_gen* algorithm in step 4 (combinatorial filter)
$g_i(i)$	average number of row instances in the table instance of a candidate prevalent co-location of size $i+1$ in iteration i after generation of table instances and neighborhood-based pruning in step 5 (Geometric filter)
$g_ci(i)$	total number of candidate prevalent co-locations of size $i+1$ in iteration i after generation of table instances and neighborhood-based pruning in step 5 (Geometric filter)

In **Step 4 and Step 5** every candidate prevalent co-location along with it table instance generalized by the *generalized apriori_gen* algorithm and spatial join is correct because of the following reasons. We compute the table instance of a co-location $C = \{f_1, \ldots, f_{k+1}\}$ by joining the table instance of the co-location $C_1 = \{f_1, \ldots, f_k\}$ and the table instance of $C_2 = \{f_1, \ldots, f_{k-1}, f_{k+1}\}$. For each instance $I_1 = \{i_{1,1}, \ldots, i_{1,k}\}$ of C_1 and each instance $I_2 = \{i_{2,1}, \ldots, i_{2,k}\}$ of C_2 we generate an instance $I_{new} = \{i_{1,1}, \ldots, i_{1,k}, i_{2,k}\}$ of C 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 neighborhood because I_1 is a neighborhood and $i_{2,k}$ is a neighbor of every element of I_1.

Step 6 is correct because the participation index calculation method based on bitmaps correctly computes each participation ratio.

In **Step 7**, calculation of the conditional probability of each rule $C' \rightarrow C - C'$ for each prevalent co-location C is correct due to the relationship of this measure to the table instance of co-location C and the cardinality of co-location C'. We compute the number of row instances of C' which appear in some row instances of C by using *relationalproject* on the table instance of C onto C' and then calculate the unique number of results.

5.3 Computational Complexity

The bulk of the execution time of the Co-location Miner algorithm is used to compare pairs of spatial feature types for equality, check the neighbor relationship of pairs of instances, and put 1s to corresponding bits of bitmaps to calculate the participation ratios. We characterize computational complexity of **Co-location Miner** using the notation in Table 4.

We summarize the computational complexity of the generation of prevalent co-locations (excluding step 7 which we are searching more efficient algorithms)

in terms of CPU cost assuming the total number of instances is N and the total number of spatial feature types is K. The cost is just a approximation since taking the average size of each table instance of a co-location of size i simplifies the analysis.

Step 1 is the initialization step; we do not need to order instances of each co-location since step 2 will use spatial join to produce candidate prevalent co-locations of size 2:

$$cost_{step\,1} = O(N);$$

Step 2 uses an efficient spatial join algorithm such as a sweeping-based algorithm [4] to produce candidate prevalent co-locations of size 2, summarize row instances to their corresponding co-location table, order co-locations and table instances of each co-location in lexicographic order, and then perform prevalence based pruning. Let the average size of the table instances before prevalence-based pruning be $h_i'(2)$ and the total number of candidate prevalent co-locations be $h_c'(2)$. The last term is the cost of calculating the participation indexes which will be discussed in the cost of step 6.

$$cost_{step\,2} = O(cost_{spatial join} + h_c'(2)\log h_c'(2) + h_c'(2)h_i'(2)\log h_i'(2) + 2h_c'(2)h_i'(2))$$

The analysis of costs inside the loop from **Step 3** to **Step 8** involve several substeps. We analyze them in iteration i as follows.

Step 4 does a $self-join$ of the ordered prevalent co-locations of size i. This is accomplished by starting from the smallest co-location and continously joining it with subsequent co-locations until a failure occurs. Continuing this process with all the co-locations in increasing order will produce all the candidate prevalent co-locations of size $i+1$. Successful joins will produce candidate prevalent co-locations of size $i+1$ whose cost is $\mathrm{O}(if_i(i+1)f_c(i+1))$. The number of failing joins is bounded by $\mathrm{O}(h_i(i)h_c(i))$ and the cost of failing joins is $ih_i(i)h_c(i)$.

$$cost_{step\,4} = O(if_i(i+1)f_c(i+1) + ih_i(i)h_c(i) + cost_{prune})$$

Step 5 joins pairs of table instances which passed the pruning of step 4 and produces table instances of candidate prevalent co-locations of size $i+1$. Row instances in each table instance are ordered. We join the row instances in two table instances in increasing order. Successful joins produce row instances of candidate prevalent co-locations of size $i+1$ whose cost is $\mathrm{O}((i+cost_{neighbor\,checking})g_i(i+1)g_c(i+1))$. Failing joins can be categorized into two cases: those that fail before neighbor checking step and those failed in the neighbor checking step The number of the first case is bounded by $\mathrm{O}(2f_i(i+1)f_c(i+1)(f_c(i+1)-1)/2)$ and the number of the second case is bounded by $\mathrm{O}((i+cost_{neighbor\,checking})g_i(i+1)g_c(i+1))$ assuming the it is proportional to the number of table instances after neighborhood pruning.

$$cost_{step\,5} = O((i + cost_{neighbor\,checking})g_i(i+1)g_c(i+1) + f_i(i+1)f_c(i+1)^2)$$

Step 6 scans all the table instances of the candidate prevalent co-locations, calculates the participation ratio for each feature, and prunes the non-prevalent

co-locations. We keep one bitmap of size $|f_i|$ for each feature $|f_i|$ of each candidate prevalent co-location. One scan of all the table instances of the candidate prevalent co-location being scanned will be enough to put corresponding bits in corresponding bitmaps to 1. By counting number of 1s in each bitmap, we can easily calculate the participation ratios (and participation index).

$$cost_{step\,6} = O(g_c(i+1)g_i(i+1) + i * max\_number\_of\_instance\_of\_all\_features)$$

The total cost inside loop is:

$$cost_{loop} = \sum_{i=3}^{max(K, max\ co-location\ size)} cost_{step\,4} + cost_{step\,5} + cost_{step\,6} + cost_{step\,7}$$

So the total cost is:

$$cost_{total} = cost_{step\,1} + cost_{step\,2} + cost_{loop}$$

From the formulas, it is clear that the cost is very sensitive to the ratios of pruning in each pruning step. If the dataset is sparse and neighbor relation is well chosen the pruning will be well done and the algorithm is efficient.

6 Conclusion and Future Work

In this paper, we formalized the co-location problem and have shown the similarities and differences between the co-location rules problem and the classical association rules problem as well as the difficulties in using the traditional measures(e.g. support,confidence) created by inexplicit, overlapping and potentially infinite transactions in spatial data sets. We proposed user-specified neighborhoods notion in place of transactions to specify groups of items and define interest measures which are robust in face of potentially infinite overlapping neighborhoods. We define a new spatial measure of conditional probability as well as a new monotonic measure of prevalence to allow iterative pruning. Our proposed co-location miner algorithm employs innovative ideas such as the *generalized apriori_gen* function to efficiently enumerate the neighborhoods of interest. We provide proofs of correctness and completeness for the **Co-location Miner** algorithm in the presence of various performance optimizations.

Our future plan is to carry out experimental evaluation and performance turning. Plane sweeping algorithms are also under exploration.

Acknowledgments. We thank Raymod T. Ng.(University of British Columbia, CA) and the other reviewers of 7th International Symposium on Spatio-temporal Databases for helping us improving the presentation of the paper and making the definitions more precise. We thank Jiawei Han (Simon Fraser University, CA), James Lesage, and Sucharita Gopal(Boston University) for valuable insights during the discussion on spatial data mining at the Scientific Data Mining workshop

2000 held at the Army High Performance Computing Research Center. We also thank C.T. Lu, Xiaobin Ma, Hui Xiong, and Pusheng Zhang (University of Minnesota) for their valuable feedback on early versions of this paper. We would like to thank Steve Klooster et. al. from NASA for the examples in Table 1.

References

1. R. Agarwal, T. Imielinski, and A. Swami. Mining association rules between sets of items in large databases. In *In Proc. of the ACM SIGMOD Conference on Management of Data, pages 207-216*, may 1993.
2. R. Agarwal and R. Srikant. Fast algorithms for Mining association rules. *VLDB*, may 1994.
3. P.S. Albert and L.M. McShane. A Generalized Estimating Equations Approach for Spatially Correlated Binary Data: Applications to the Analysis of Neuroimaging Data. *Biometrics(Publisher: Washington, Biometric Society, Etc.), 1:627-638*, 1995.
4. 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.
5. Y. Chou. In *Exploring Spatial Analysis in Geographic Information System, Onward Press, (ISBN: 1-56690-119-7)*, 1997.
6. N. Cressie. In *Statistics for Spatial Data, Wiley-Interscience, (ISBN:0-471-00255-0)*, 1993.
7. G. Greenman. Turning a map into a cake layer of information. *New York Times*, Feb 12 2000.
8. R.H. Guting. An Introduction to Spatial Database Systems. In *Very Large Data Bases Jorunal(Publisher: Springer Verlag)*, October 1994.
9. R.J. Haining. Spatial Data Analysis in the Social and Environmental Sciences. In *Cambridge University Press, Cambfidge, U.K*, 1989.
10. J. Han and Y. Fu. Discovery of multiple-level association rules from large databases. In *In Proc. 1995 Int. Conf. Very Large Data Bases, pages 420-431, Zurich, Switzerland*, September 1995.
11. J. Hipp, U. Guntzer, and G. Nakaeizadeh. Algorithms for Association Rule Mining - A General Survey and Comparison. In *In Proc. ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2000.
12. Issaks, Edward, and M. Svivastava. Applied Geostatistics. In *Oxford University Press, Oxford*, 1989.
13. K. Koperski, J. Adhikary, and J. Han. Spatial Data Mining: Progress and Challenges. In *In Workshop on Research Issues on Data Mining and Knowledge Discovery (DMKD'96)*, 1996.
14. K. Koperski and J. Han. Discovery of Spatial Association Rules in Geographic Informa tion Databases. In *In Proc. Fourth International Symposium on Large Spatial Data bases, Maine. 47-66*, 1995.
15. P. Krugman. Development, Geography, and Economic theory. In *MIT Press, Camgridge, MA*, 1995.
16. Z. Li, J. Cihlar, L. Moreau, F. Huang, and B. Lee. Monitoring Fire Activities in the Boreal Ecosystem. *Journal Geophys. Res,. 102(29):611-629*, 1997.
17. D. Mark. Geographical Information Science: Critical Issues in an Emerging Cross-disciplinary Research Domain. In *In NSF Workshop*, February 1999.

18. D.C. Nepstad, A. Verissimo, A. Alencar, C. Nobre, E. Lima, P. Lefebvre, P. Schlesinger, C. Potter, P. Moutinho, E. Mendoza, M. Cochrane, and V. Brooks. Large-scale Improverishment of Amazonian Forests by Logging and Fire. *Nature, 398:505-508*, 1999.
19. J.S. Park, M. Chen, and P.S. Yu. Using a Hash-Based Method with Transaction Trimming for Mining Association Rules. In *IEEE Transactions on Knowledge and Data Engineering, vol. 9, no. 5, pp. 813-825*, Sep-Oct 1997.
20. J.F. Roddick and M. Spiliopoulou. A Bibliography of Temporal, Spatial and Spatio-temporal Data Mining Research. *ACM Special Interest Group on Knowledge Discovery in Data Mining(SIGKDD) Explorations*, 1999.
21. S. Shekhar and S. Chawla. Spatial Databases: Issues, Implementation and Trends. *Prentice Hall (under contract)*, 2001.
22. S. Shekhar, S. Chawla, S. Ravada, A. Fetterer, X. Liu, and C.T. Lu. Spatial Databases: Accomplishments and Research Needs. *IEEE Transactions on Knowledge and Data Engineering, 11(1)*, Jan-Feb 1999.
23. S. Shekhar, T.A. Yang, and P. Hancock. An Intelligent Vehicle Highway Information Management System. *Intl Jr. on Microcomputers in Civil Engineering (Publisher: Blackwell Publishers), 8(3)*, 1993.
24. R. Srikant and R. Agrawal. Mining Generalized Association Rules. In *Proc. of the 21st Int'l Conference on Very Large Databases, Zurich, Switzerland*, 1997.
25. R. Srikant, Q. Vu, and R. Agrawal. Mining Association Rules with Item Constraints. In *Proc. of the 3rd Int'l Conference on Knowledge Discovery in Databases and Data Mining, Newport Beach, California*, Aug 1997.
26. P. Stolorz, H. Nakamura, E. Mesrobian, R.R. Muntz, E.C. Shek, J.R. Santos, J. Yi, K. Ng, S.Y. Chien, R. Mechoso, and J.D. Farrara. Fast Spatio-Temporal Data Mining of Large Geophysical Datasets. In *Proceedings of the First International Conference on Knowledge Discovery and Data Mining, AAAI Press, 300-305*, 1995.
27. C. Tsur, J. Ullman, C. Clifton, S. Abiteboul, R. Motwani, S. Nestorov, and A. Rosenthal. Query Flocks: a Generalization of Association-Rule Mining. In *Proceedings of 1998 ACM SIGMOD, Seattle*, 1998.
28. M.F. Worboys. GIS: A Computing Perspective. In *Taylor and Francis*, 1995.
29. Y. Yasui and S.R. Lele. A Regression Method for Spatial Disease Rates: An Estimating Function Approach. *Journal of the American Statistical Association, 94:21-32*, 1997.

Appendix: The Apriori Algorithm

Table 5. Notation

k-itemset	An itemset having k items
L_k	Set of large k-itemsets (those with minimum support). Each member of this set has two fields: i) itemset and ii) support count
C_k	Set of candidate k-itemsets (potentially large itemsets). Each member of this set has two fields: i) itemset and ii) support count.
$\overline{C_k}$	Set of candidate k-itemsets when the TIDs of the generating transactions are kept associated with the candidates.

Algorithm Apriori [2]:

1) L_1 = {large 1-itemsets};
2) **for** (k =2; $L_{k-1} \neq \emptyset$; k++} **do begin**
3) C_k = apriori-gen(L_{k-1}); //New candidates
4) **forall** transactions $t \in D$ **do begin**
5) C_t = subset(C_k, t); //Candidates contained in t
6) **forall** candidates $c \in C_t$ **do**
7) c.count++;
8) **end**
9) $L_k = \{c \in C_k | c.\text{count} \geq \text{minsup}\}$
10) **end**
11) Answer = $\bigcup_k L_k$;

The **apriori-gen** function takes as argument L_{k-1}, the set of all large (k-1)-itemsets. The function works as folows. First, in the *join* step, we join L_{k-1} with L_{k-1}:

insert into C_k
select p.item$_1$, p.item$_2$, ..., p.item$_{k-1}$, q.item$_{k-1}$
from L_{k-1} p, L_{k-1} q
where p.item$_1$ = q.item$_1$, ..., p.item$_{k-2}$ = q.item$_{k-2}$,p.item$_{k-1}$ < q.item$_{k-1}$;

Next, in the *prune* step, we delete all itemsets $c \in C_k$ such that some $(k-1)$-subset of c is not in L_{k-1}:

forall itemsets $c \in C_k$ **do**
forall $(k-1)$-subsets s of c **do**
if ($s \notin L_{k-1}$) **then**
delete c from C_k;