Performance Evaluation of Co-location Miner

Shashi Shekhar*, Yan Huang *, Hui Xiong *

June 19, 2001

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) as well as association rule mining algorithms using support based pruning. We recently defined the problem of mining spatial co-location patterns and proposed the *Co-location Miner* [25], an algorithm for mining co-locations. In this paper, we present an experimental performance evaluation of *Co-location Miner*. For the purpose of comparison, we consider two other approaches, namely the pure geometric approach and the pure combinatorial approach. Empirical evaluation shows that the pure geometric method performs much better than the pure combinatorial method when generating size 2 co-locations; however, it becomes much slower when generating co-locations with more than 2 features. Co-location Miner integrates the best features of the above two approaches and provides the best overall performance. Experimental results also show that *Co-location Miner* is robust in the face of noise and scales up gracefully with increases in the number of spatial feature types, maximum size of co-location patterns, and the number of instances of spatial features.

Keywords: spatial data mining, Geographic Information System, spatial co-location rules, association rules.

*Computer Science Department, University of Minnesota 200 Union Street SE, Minneapolis, MN-55455,USA.Support in part by the Army High Performance Computing Research Center under the auspices of Department of the Army, Army Research Laboratory Cooperative agreement number DAAH04-95-2-0003/contract number DAAH04-95-C-0008. [*shekhar*, *huangyan*, *huix*]@cs.umn.edu

1 Introduction

Widespread use of spatial databases [9, 23, 24, 32] is leading to an increasing interest in mining interesting and useful but implicit spatial patterns [8, 15, 19, 22, 30, 26, 5]. 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, 14, 17, 10, 27, 30, 33]. Here we focus on the application domain of ecology, where scientists are interested in finding frequent co-occurrences among boolean spatial features, e.g., drought, El Nino, substantial increase in vegetation, substantial drop in vegetation, and extremely high precipitation.

Association rule finding [13]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 [16] are spatial cases of general association rules where at least one of the predicates is spatial. Association rule mining algorithms [1, 2, 11, 13, 12] assume that a finite set of disjoint transactions are given as input to the algorithms. In market basket data, a transaction consists 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). Figure 1 shows the frequent co-occurrences of some spatial feature types represented by different shapes. As can be seen spatial features in sets {'+', '×'} and {'o', '*'} tend to be located together. 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) [16] 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.

We recently defined the problem of mining spatial co-location patterns and proposed *Co-location Miner* [25], an algorithm for mining co-locations. Reviewers considered co-location patterns to be a more natural approach for mining association-like patterns in spatial datasets since this pattern does not require the creation of transactions. In this paper, we present an experimental performance evaluation of *Co-location Miner*. For the purpose of comparison, we consider two other approaches, namely the pure the geometric approach and the pure combinatorial approach. The geometric approach uses spatial join (e.g. rectangle join) to enumerate neighborhoods. It scans the nearby regions of anchor locations to generate larger neighborhoods. The combinatorial approach enumerates interested neighborhoods after performing spatial feature level pruning. Its performance depends on the effectiveness of the spatial feature level pruning. Empirical evaluation shows that the geometric method performs much better than the combinatorial method when generating size 2 co-locations; however it becomes much slower when generating co-locations with more than 2 features. The reason is that a geometric method such as spatial join keeps the information of its nearby regions but lacks spatial feature level pruning, while

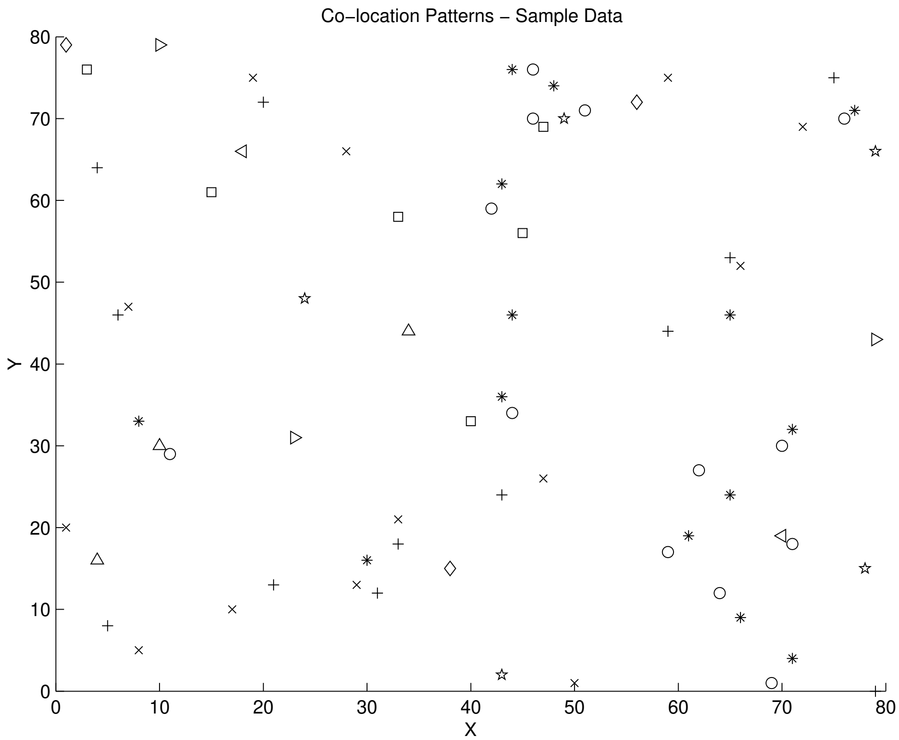

Figure 1: Spatial Co-location Patterns Illustration. Shapes represent different spatial feature types. Spatial features in sets {'+', '×'} and {'o', '*'} tend to be located together.

the combinatorial method benefits from spatial feature level pruning but suffers from having to scan a larger portion of the database without local information when the spatial feature level pruning is not effective. The *Co-location Miner* algorithm integrates the best features of the above two approaches by starting from the geometric method and switching to the combinatorial approach when the number of co-locations increases and provides the best overall performance. Experimental results also show that *Co-location Miner* is robust in the face of noise and scales up gracefully with increases in number of spatial feature types, maximum size of co-location patterns, and the number of instances of spatial features.

1.1 An Illustrative Application Domain

Many ecological datasets [18, 20] 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 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 features. Examples of interesting co-location patterns in ecology are shown in Table 1.

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

The spatial patterns of ecosystem datasets include:

a. **Local co-location patterns**, which 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 [13] for mining classical association rules.

b. **Spatial co-location patterns**, which 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. In this paper, we focus on the co-location patterns described above.

1.2 Related Work and Our Contributions

Approaches to discovering co-location rules in the literature can be categorized into two classes, namely spatial statistics and association rules. Spatial statistics-based [6, 7] 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 [13] 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 the determination of prevalence measures, e.g., support. Another approach is based on the choice of a reference spatial feature [16] to mine all association rules of the following form:

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

where at least one of the predicates is a spatial predicate. Users specify reference spatial feature (e.g. big cities in Canada) and other relevant spatial features (e.g. US boundaries, population, transportation, and sea). The algorithm [16] uses a 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 not involving reference spatial features but involving other features. For example, consider the co-location of transportation network in Canada with US boundaries.

We present an experiment design and performance evaluation of *Co-location Miner*, an algorithm we proposed in [25] for mining co-locations, using a wide range of datasets with different properties. For the purpose of comparison, we provide additional simple algorithms to explain design decisions in *Co-location Miner*. We consider two other approaches for co-location mining: the pure geometric method and the pure combinatorial method. The geometric method retrieves the neighbors of a location by keeping the information of nearby regions. Panes-weep techniques [4] are used for computing the spatial join. The combinatorial method formularizes the problem as a smart clique enumeration problem from a graph based on the definition of neighbors and uses techniques similar to the *apriori* [2] algorithm. The combinatorial method depends heavily on spatial feature level pruning. We compare and contrast the pure geometric method and the pure combinatorial method experimentally. Empirical evaluations show that the pure geometric method performs better than the pure combinatorial method when generating size 2 co-locations; however, it becomes much slower when generating co-locations with more than 2 features. The reason is that the geometric method scans a smaller portion of the whole dataset by keeping the information of nearby regions while the combinatorial method benefits more from being able to prune based on spatial feature types when the sizes of co-locations are increasing. *Co-location Miner* combines the best features of the above two approaches and provides the best overall performance. Through experiments we also show that *Co-location Miner* is robust in the face of noise and scales up gracefully with increases in the number of spatial feature types, maximum size of co-location patterns and the number of instances of spatial features, assuming sparse spatial datasets.

1.3 Outline and Scope

Section 2 formulates the problem of mining co-location rules. Section 3 describes approaches for modeling co-location problems and their associated prevalence and conditional probability measures. In Section 4, we present various options in each step of the co-location mining algorithms and describe three algorithms, namely the pure geometric, the pure combinatorial, and the hybrid algorithms. Section 5 present the experiment design and evaluates the algorithms based on decisions made in each step of the experiment design. The conclusion and future work are presented in Section 6.

The scope of this paper is limited to evaluation of performance when producing co-locations in two dimensional Euclidean space; the paper does not consider the generation of co-location rules. The relative performance of different algorithms for generating co-location rules is likely to be the same as that for generating co-locations and we plan to explore this in future work. Issues beyond the scope of the paper include other spatial patterns such as spatio-temporal co-locations.

2 Basic Concepts and Problem Formulation

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 the **confidence** of the rule and $Pr(X \cup Y)$ is called the **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, 13, 21, 28, 29, 31].

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 that they do not share 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. Prevalence measures (e.g. in association rules) and conditional probability (e.g. confidence in association rules) need to be redefined without a transaction definition.

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 dataset, i.e., the number of instances of any spatial features is << cardinality(P)

3 Modeling Co-location Rules

Given the difficulty in creating explicit disjoint transactions from continuous spatial data, this section defines approaches to modeling co-location rules. We use Figure 2 as an example spatial dataset As can be seen in Figure 2, one location can associate with more than one spatial feature type. Spatial feature types are labeled beside their instances. Instances with interested spatial relationships are connected by edges. We define the following basic concepts to facilitate the description of the different models.

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, 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, need to be defined in spatial application domains and will be described after describing two models of interpretation, namely the reference feature centric model and the event centric model.

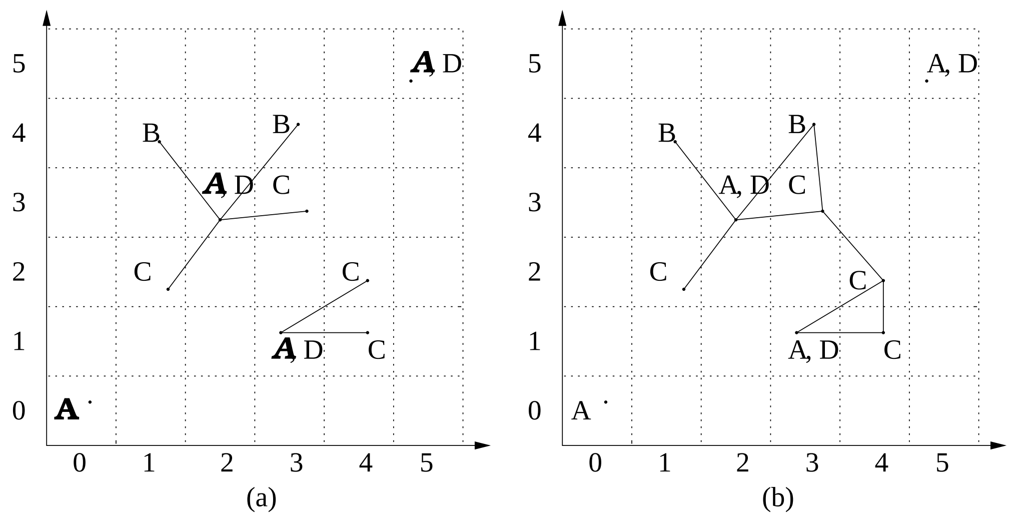

Figure 2: Spatial dataset to illustrate different co-location models. Spatial feature types are labeled beside their instances. a) The reference feature centric model. A is the referenced feature. B and C are relevant features. The instances of A are connected with their neighboring instances of B and C by edges. b) The event centric model. Neighboring instances are joined by edges.

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, 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 [16]. For example, in Figure 2 a), 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 the 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 shown in Table 2, since one out of two non-empty transactions contains instances of both B and C, and one out of two non-empty transactions contains C, 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.

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

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

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 Figure 2 b). There are four instances of type A and only one has some instance(s) of type B in its neighborhood. The conditional probability for the co-location rule is: *spatial feature A at location l → spatial feature type B in 9-neighbor neighborhood is 25%.*

Neighborhood is an important concept in the event centric model. The definition of a neighbor relation is an input and is based on the semantics of application domains. Neighbor relation 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). A formal definition appears in [25] and is reproduced in the **Appendix A** for reference. 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 [25] is proposed to be a prevalence measure.

Here we introduce several concepts necessary for defining prevalence and conditional probability measures for spatial datasets. $I = \{i_1, \ldots, i_k\}$ 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 has pairwise interested neighboring relationships. For example, {(3,1),(4,1)} is an instance of co-location $\{A, C\}$ in Figure 2 b). The **table instance** of a co-location $C = \{f_1, \ldots, f_k\}$ is the collection of all its row instances. 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. This ratio 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 Figure 2 c), 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 $pr(\{A,B\},A) = \frac{1}{4} = .25$. The **participation index** of a co-location $C = \{f_1, f_2, \ldots, f_k\}$ is $min\{pr(C, f_i)\}, i = 1, \cdots, k$. We use the minimal value instead of the product of the participation ratios [25] because minimal value is more meaningful to end users. In Figure 2 b), 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 min$\{.25, 1.0\}$ = .25. Note that the participation index is monotonically non-increasing with the size of the co-location increasing since any spatial feature that 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. It can be formally defined as: 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.

The lack of transactions in spatial co-location mining creates fundamental differences between association rule mining and co-location rule mining. The major differences are presented in Table 3.

Table 3: Association Rules Vs. Co-location Rules

Criteria	Association Rule Mining	Co-location Rule Mining
Item Types	Product types	Spatial Features
Item Collections	Transactions $\{T_i\}$	Neighborhoods
Prevalence ($A \rightarrow B$)	Support: $p(A \cup B \in T_i)$	Participation Index
Conditional Probability ($A \rightarrow B$)	Confidence: $p(B \in T_i \mid A \in T_i)$	Conditional Probability $p(B \in Neighborhood\,of\,L \mid A\,at\,L)$

4 Description of Three 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 challenges and then we describe three algorithms, namely the pure geometric algorithm, the pure combinatorial algorithm, and the hybrid algorithm in Section 4.1. In our recent work, we used the product of the participation ratios to define the participation index. In this paper we define the participation index to be the minimal of the participation ratios in this paper. The change in definition is due to feedback from end users. This change does not impact the *Co-location Miner* algorithm.

4.1 Challenges

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 the system level via spatial database techniques such as spatial indexes.

Co-location row instances are enumerated before measures of prevalence and conditional probability are computed at the 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.

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 plan to discuss the mining of co-location rules in future work.

We present three algorithms to meet the above challenges after we present a high level description of main steps of the algorithms.

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) Initialization
2) Generate size 2 co-location rules
3) Generate size 3 or more co-location rules

4.2 The Geometric Approach

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.

Step 2 generates prevalent co-locations of size 2.
The process of generating prevalent co-location of size $k + 1$ can be based on spatial joins of table instances of prevalent co-location of size k, using spatial-join predicate of common neighborhood.

A simple approach is to use a single spatial self-join of union of table instances of all prevalent co-locations of $sizek$ to generate intermediate results which could be summarized to get the prevalent co-location of size $k + 1$. This approach is particularly effective in Step 2, since all co-locations of size 1 need to be considered to generate co-location of size 2. Given tables feature_type(id, name) and instances(point_id, location_ coordinates, feature_type_id) and stored procedure participation_index(feature_type_id, feature_type_id), computation for step 2 can be expressed as the following spatial-self-join query:

select I_1.feature_type_id, I_2.feature_type_id,
    participation_index(I_1.feature_type_id, I_2.feature_type_id)
from instance I_1, instance I_2
where I_1.feature_type_id > I_2.feature_type_id
    and neighbor(I_1.location_coordinates, I_2.location_coordinates)
group by (I_1.feature_type_id, I_2.feature_type_id)
having participation_index(I_1.feature_type_id, I_2.feature_type_id) > min_prevalence

We modify a plane-sweeping based spatial join algorithm [4] to check for (I_1.feature_type_id > I_2.feature_type_id and neighbor(I_1.location_coordinates, I_2.location_coordinates))on the fly in the core algorithm. The result of spatial self-join is sorted by co-location, i.e.(I_1.feature_type_id, I_2.feature_type_id), to compute participation index and prevalent co-locations of size 2.

Step 3 generate co-locations rules of size 3 or more.

This step can be computed in a similar manner with step 2 using bounding box, MOBR (minimum orthogonal bounding Rectangle), of row-instances of prevalent co-locations of smaller size. Given tables co-location_of_size_k(id, subset_of_features) and size_k_row_instances(id, co-location_id, MOBR, set of location of feature instance), this computation can be expressed as follows:

select I_1.co-location_id, I_2.co-location_id,
    participation_index(I_1.feature_type_id, I_2.feature_type_id)
from size_k_row_instance I_1, size_k_row_instances I_2
where all_pair_neighbor(I_1.set_of_location, I_2.set_of_location)
  and differs_by_exactly_1(I_1.co_location_id.subset_of_features, I_2.co_location_id.subset_of_features)
group by (I_1.co_location_id, I_2._co_location_id)
having participation_index(I_1.feature_type_id, I_2.feature_type_id) > min_prevalence

The stored procedure all_pair_neighbor() ensures that every pair of locations in the cross-product of I_1.set_of_locations and the I_2.set_of_location are related by the given neighbor relationship R.

A minimum orthogonal bounding box (set_of_location) can be used to design a fast filter before checking each pair. A stored procedure differs_by_exactly_1() ensures that the set- differences of co-location (I_1.co-location_id, I_2.co-location_id) and (I_2.co-location.subset_ of_features, I_1.co-location.subset_of_features) are singleton sets. This helps efficient enumeration of cliques among features as shown in *apriori* algorithm, and may be checked in a pipe-line fashion during computation of spatial join. We use a minor modification of a plane-sweep based spatial join for this computation.

4.3 The Combinatorial Approach

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: Figure 4 a) shows the size 1 co-locations, i.e. A, B, and C, and their table instances for the example dataset in Figure 3.

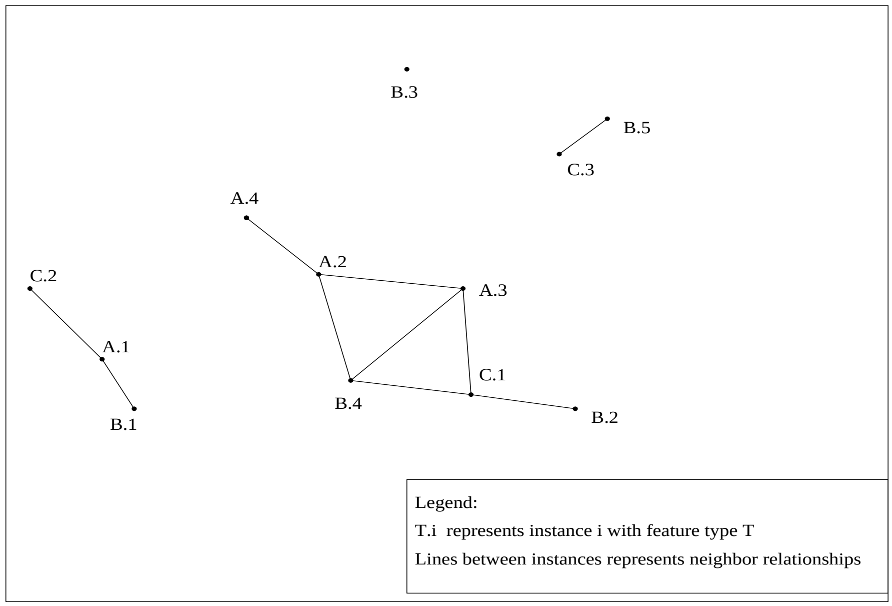

Figure 3: Example Database

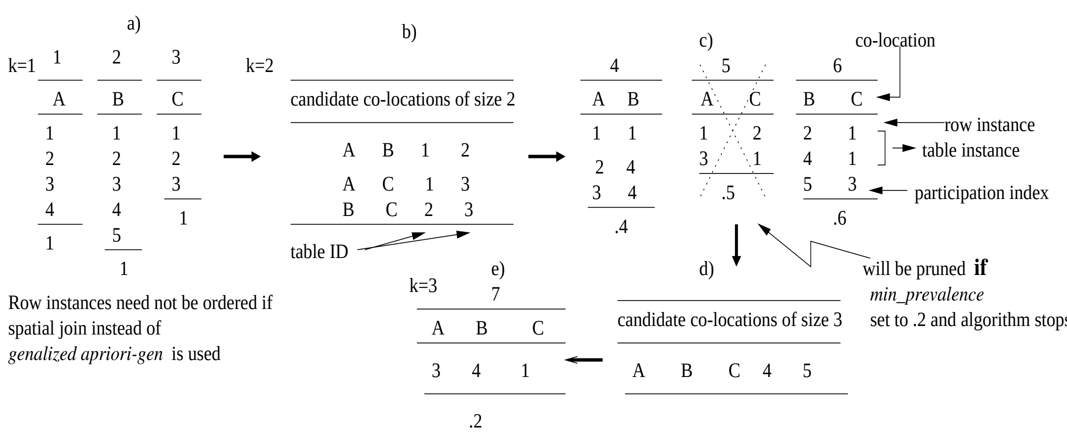

Figure 4: Combinatorial Algorithm Illustration on Example Database

Step 2 generates prevalent co-location rules of size 2.

The combinatorial approach to generate co-locations of size 2 is the same as the approach to generate co-locations of size 3 or more. And the algorithm to calculate participation indexes and prunes accordingly stays the same as described in the generation of co-locations of size 3. But since all spatial features are singleton co-locations with participation indexes equal to 1, spatial feature level pruning is not effective in this step.

Step 3 works using the following sub-steps:

3.1) **for** size of co-locations in (2, 3, ... , $K-1$) **do**

3.2) Generate candidate prevalent co-locations using the *generalized apriori_gen* algorithm

3.3) Generate table instances and prune based on neighborhood

3.4) Prune based on prevalence of co-locations

3.5) Generate co-location rules

3.6) **end**;

Step 3.1 to **Step 3.3** 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. The loop breaks whenever an empty co-location set of some size is generated.

Step 3.2 uses *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 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* produces 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 this is also done in *apriori_gen* [2] because of the monotonicity property of the 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 3.3 generates all the table instances of candidate co-locations of size $k+1$ which passed the filter of step 3.2. Co-locations with empty table instances will be eliminated from the candidate prevalent co-location set of size $k+1$. This step 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 Figure 4, 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 *generalized 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).

Step 3.4 calculates the participation indexes for all candidate co-locations in C_{k+1} and prunes the co-locations using the prevalence threshold. Computation of the participation index for a co-location C requires scanning its table instance to compute participation ratios for each feature in the co-location. This computation can be modeled as a *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 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 4 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 $\{A, B\}$. Thus the participation ratio for A is .75. Similarly, the participation ratio for B is .4. The participation index is min{.75, .4} = .4. 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 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.

Step 3.5 generates all the co-location rules with the 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: $C' \to C - C'$. 1) 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:$C' \to C - C'$ if the conditional probability is above user specified threshold.

4.4 The Hybrid Approach - Co-location Miner

The geometric approach suffers from a lack of spatial feature level pruning but it keeps the information of nearby regions. The combinatorial approach does not keep local information but it benefits from spatial feature level pruning. Since all the participation indexes of singleton co-locations are 1, making all singleton co-locations prevelant and spatial feature level pruning is not effective when generating co-locations of size 2, the geometric approach is preferred in this step. However, when generating co-locations of size 3 or more, spatial feature level pruning is the dominant optimization technique, making the combinatorial method the preferred approach in this step. By integrating the advantages of the geometric and combinatorial approaches, we get a hybrid approach which uses the geometric approach for **Step 1** and **Step 2** and switches to combinatorial approach for **Step 3**.

5 Experiment Design and Performance Evaluation

Figure 5 describes the experimental setup to evaluate relative performance of the alternative algorithms and to study the effects of different parameters on Co-location Miner algorithm.

We evaluates the performance of *Co-location Miner* with synthetic data generated using a methodology similar to methodologies used to evaluate algorithms for mining association rules [2]. Synthetic datasets allow better control towards studying effects of interesting parameters, e.g. number of co-locations (N_{co_loc}), expected size (λ_1) of maximal co-locations, etc. The list of the parameters is presented in Table 4. A data-flow diagram of data generation process is shown in Figure 5. The process began with the generation of subsets of spatial features (maximal co-locations). 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 co-location pattern was chosen. To simplify interpretation of the results, co-locations were kept mutually disjoint. 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 neighborhoods for each co-location using the size of their table instances from the previous step. Each row in the table instance of a co-location was embedded inside a neighborhood of size d by generating point locations for each feature in the co-location. The locations of neighborhoods were chosen at random in the overall spatial framework. For simplicity, the shape of the overall spatial framework was a square of size $D \times D$ and the size of each neighborhood was $d \times d$. The final step was to add noise via a local process as well as a global process. The model for local noise in each neighborhood uses two parameters, namely the ratio of noise features $r_{noise\_feature}$ and the ratio of local noise instances p_{local_noise}. The local noises were generated by choosing features involving generation of maximal co-locations randomly and placing them into the spatial framework at random. Global noise was added by generating a set of instances of features in a set of noise features disjoint with the features involving generation of maximal co-locations and placing those at random locations in the global spatial framework. A sample dataset is shown in Figure 1. This small dataset was produced by generating 2 co-locations of ($\lambda_1 = 2$) size 2 (N_{co_loc}) with table instance sizes 11 and 7 ($\lambda_2 = 10$) respectively. Both the local noise ratio and the global noise ratio were .5. Global noise was generated from 6 noise features ($r_{noise\_feature}$).

The collections of spatial datasets used are listed in Table 6. All datasets used a square spatial framework of size $10^6 \times 10^6$ and a square neighborhood of size 10×10. The number of maximal co-locations varied from 30 to 250, with average size being 3 or 5. For example, the dataset Case_{comp}

had 30 maximal co-locations with an average size of 3. The average number of row instances for a maximal co-location varied from 30 to 400. For example, the average number of rows in the maximal co-locations in dataset Case_{comp} was 30. The noise parameters and the total size of resulting datasets are also reported in Table 6. These datasets are used by the co-location algorithm driver module to collect the performance statistics, as shown in Figure 5. The driver calls the candidate algorithm (i.e. the pure geometric algorithm, the pure combinatorial algorithm, and the hybrid algorithm - *Co-location Miner*. Each algorithm was instrumented to measure execution time for discovering co-locations of size 2 as well the execution time for co-locations of size 3 or more. The execution time was measured using the "time" utility in the C++ language on a Sun Ultra 10 work station with a 440 MHz CPU, and 128 Mbytes memory running the SunOS 5.7 operating system. The absolute value of the execution time is not of interest to the main hypothesis. We also reported the total number of all row instances for all co-locations. We presented it by using the total number of all row instances of co-locations of size 3 since it was usually the largest. We intend to collect logical metrics (e.g. number of operations) in future work to evaluate the cost model. The last module in the experimental setup was responsible for summarizing the measurements in the form of plots and tables reported in the following section.

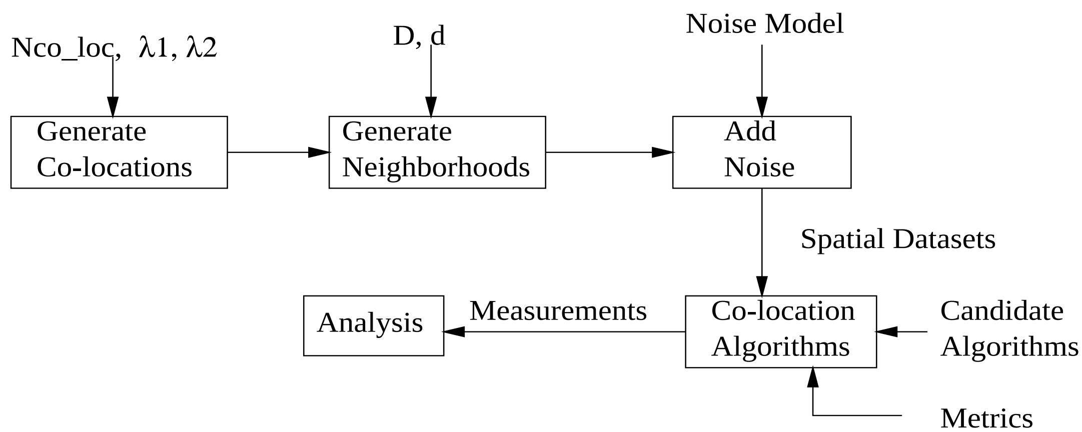

Figure 5: Experimental Setup and Design

5.1 Comparison of Candidate Algorithms

The candidate algorithms were compared using the dataset Case_{comp} described in the first row of Table 6. This dataset was small to keep the execution time of all the algorithms within a few hours. It has 30 co-locations with an average size of 3 features per co-locations and 30 row-instances per co-location. Table 5 show the execution times for the three candidate algorithms for different values of min_prevalence. The second column reports the execution time needed to discover co-locations of size 2. As can be seen, the geometric algorithm is orders of magnitude faster than the combinatorial algorithm. Spatial join data-structures help geometric algorithm in this step. The remaining column reports 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 for different values of min-prevalence. In this case, the combinatorial algorithm is orders of magnitude faster than the geometric algorithm. Combinatorial pruning (e.g apriori-gen [2]) helps the combinatorial algorithm. Figure 6 shows these trends graphically.

Table 4: Parameters Used to Generate Synthetic Data

N_{co_loc}	The number of co-locations
λ_1	The parameter of the Poisson distribution to define the size the co-locations
λ_2	The parameter of the Poisson distribution to define the size of the table instance of each co-location
D	The size of the spatial framework
d	The size of the square to define a co-location
$r_{noise\_feature}$	The ratio the of number of noise features over the number of features involved in generating the maximal co-locations
r_{noise_local}	The ratio of the number of noise instances from a set of non-noise features over the number of instances involved in generating the maximal co-locations
r_{noise_global}	The ratio of the number of noise instances from a set of noise features over the number of instances involved in generating the maximal co-locations

The *Co-location Miner* is a hybrid, it uses the geometric algorithm for discovering prevalent co-locations of size 2 and the combinatorial algorithm for discovering larger co-locations. Thus, it achieves the best performance overall. Our experimental results confirms this as shown in Table 5.

Table 5: Relative Performance of Geometric, Combinatorial, and Hybrid Algorithms (sec.)

Algorithm	Size 2	Participation Index Threshold							
		0		0.1		0.25		0.5	
		Size 3+	Total	size 3+	Total	Size 3+	Total	Size 3+	Total
Geometric Algo.	16	28215	28231	12070	12086	306	342	16	32
Combinat. Algo.	708	18	726	7	805	1	709	1	709
Co-location Miner	16	18	34	7	23	1	17	1	17

5.2 Effect of Parameters on Performance of Co-location Miner

We investigated the effects of three parameters, namely number of maximal co-locations, average size of table instances per co-location, and noise, on the overall execution time of the *Co-location Miner*. The goal was to rank the parameters in terms of their influence.

Datasets Case_1, Case_2, Case_3, Case_4, and Case_5 were used to study the effect of number of maximal co-locations on the execution time of the *Co-location Miner*. These datasets had co-locations with an average size of 5, an average table instance size of 50, and identical values for parameters to the noise

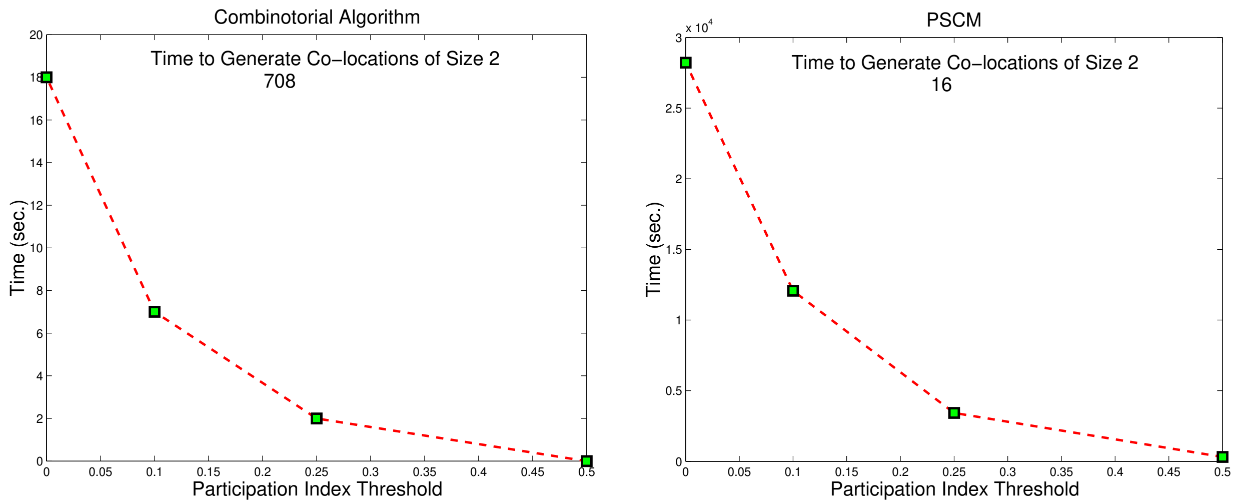

Figure 6: Total Exec. Time to Generate Co-locations of Size 3 or more (Time to Generate Co-locations of Size 2 is Labeled at the Upper Right–hand Corner of Each Graph)

model. The number of maximal co-location varied from 50 to 250. Figure 7(a) shows the effect of the number of maximal co-locations on the execution time for different values of min_prevalence Execution time rises sharply as the number of maximal co-locations increases. It appears that the rise in execution cost for high values of min_prevalence is not as sharp when the number of maximal co-locations increases. We plan to study this further in future work. Figure 7 (b) shows the total size (#row instances) of all table instances of prevalent co-locations of size 3 produced by the algorithm for different values of min_prevalence. The shapes of the curves in Figure 7 (a) and Figure 7(b) are similar, indicating strong relationship between total execution time and total size of all table instance. This insight may be useful in cost model development.

Next we studied the effect of the average size of table instances of maximal co-locations on the execution time of *Co-location Miner*, using datasets Case_1, Case_6, Case_7, Case_8. These datasets had 50 maximal co-locations with an average of 5 feature types. The noise model parameters were identical across datasets. The average size of table instances vary from 50 to 400, changing the total size of the spatial datasets from 38,778 to 313,458 including noise. The experimental results are shown in Figure 8(a). As can be seen, an increase in the average size of table instances means increased execution cost. However, the increase in execution time seems sub-linear. Figure 8(b) shows the total number of rows for prevalent co-locations of size 3 as a function of the average size of table instances of maximal co-locations and min_prevalence. The trends are similar to those in Figure 8(a). A comparison of Figure 7 and Figure 8 reveals that the effect of the average size of table instances is weaker than the effect of the number of maximal co-locations.

Finally, we studied the effect of noise on the execution time of *Co-location Miner* using dataset Case_1, Case_9, Case_{10} and Case_{11}, which are described in Table 6. These datasets had 50 maximal co-locations with an average of 5 features and 50 row instances. The ratio of global noise instance to co-location features is varied from .5 to 2 and ratio of noise features increased accordingly to avoid too many instances

Table 6: Synthetic Data Parameters

Cases	D	d	N_{co_loc}	λ_1	λ_2	$r_{noise\_feature}$	r_{noise_local}	r_{noise_local}	N_{inst}
Case_{comp}	1×10^6	10	30	3	30	.5	2	1	14,546
Case_1	1×10^6	10	50	5	50	1	.5	.5	38,778
Case_2	1×10^6	10	100	5	50	1	.5	.5	75,714
Case_3	1×10^6	10	150	5	50	1	.5	.5	108,201
Case_4	1×10^6	10	200	5	50	1	.5	.5	169,184
Case_5	1×10^6	10	250	5	50	1	.5	.5	224,121
Case_1	1×10^6	10	50	5	50	1	.5	.5	38,778
Case_6	1×10^6	10	50	5	100	1	.5	.5	78,093
Case_7	1×10^6	10	50	5	200	1	.5	.5	156,628
Case_8	1×10^6	10	50	5	400	1	.5	.5	313,458
Case_1	1×10^6	10	50	5	50	1	.5	.5	38,778
Case_9	1×10^6	10	50	5	50	2	.5	1	51,278
Case_{10}	1×10^6	10	50	5	50	3	.5	1.5	45,028
Case_{11}	1×10^6	10	50	5	50	4	.5	2	57,528

of any one noise feature while the ratio of local instances was kept the same to maintain the co-location patterns. A higher value implied higher noise, i.e. each neighborhood contained instances of noise features which were not part of any maximal co-location. The curves in Figure 9(a) show the execution times for different levels of noise for discovering co-locations of size 3 or more given prevalent co-locations of size 2 for different values of min_prevalence. The execution time for discovering co-locations of size 2 is shown using numbers just along the x-axis. We note that noise-level affects the execution time to discover co-location of size 2 but 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. Figure 9(b) shows the total number of row instances for prevalent co-locations of size 3. The trends are similar to those in Figure 9(a).

6 Conclusion and Future Work

In this paper, we have presented an experimental design and performance evaluation of *Co- location Miner*, which combines the best features of the geometric approach and the combinatorial approach for mining co-location pattern. Experimental results show that *Co-location Miner* is robust in the face of noise and scales up gracefully with increases in the number of spatial feature types, maximum size of co-location patterns, and the number of instances of spatial features.

In our future work, we will investigate the process of generating co-location rules. We also plan to study the problem of co-location mining when spatial events are represented as other spatial types such as lines and polygons.

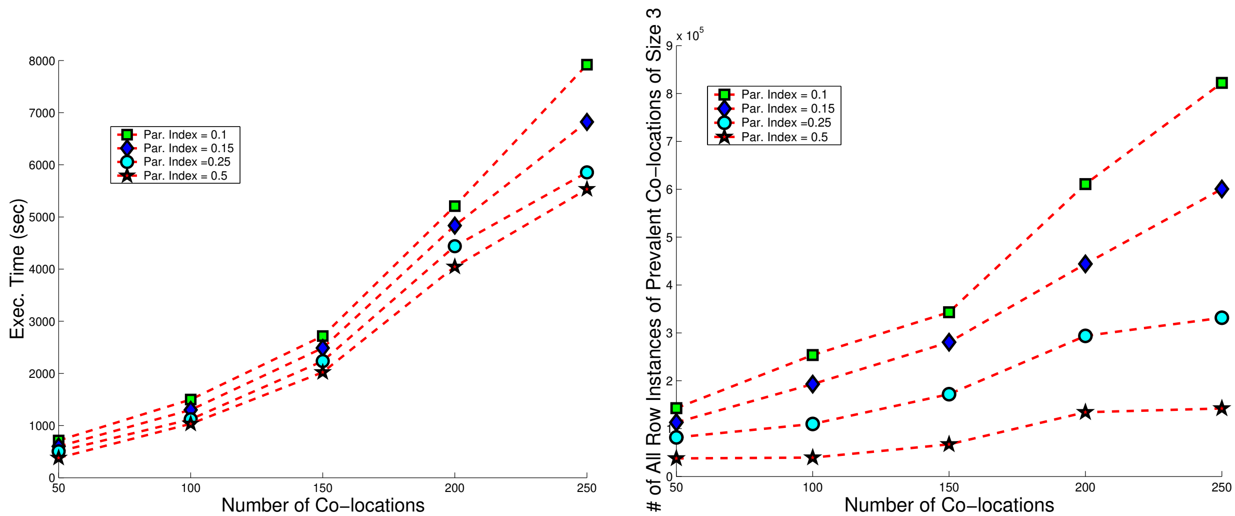

Figure 7: Experimental Results of $Case_1$, $Case_2$, $Case_3$, $Case_4$, $Case_5$ a) Exec. Time of Different Participation Index Threshold. b) Total Number of Row Instances in All **Prevalent** Co-locations of Size 3 for Different Participation Index Threshold.

Acknowledgments

We thank Mukund Deshpande (University of Minnesota) for suggesting use of min instead of product in defining participation index measure. 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, and Pusheng Zhang (University of Minnesota) for their valuable feedback on early versions of this paper. We would like to thank Chris Potter and Steve Klooster from NASA for the examples in Table 1.

References

[1] R. Agrawal, T. Imielinski, and A. Swami. Mining association rules between sets of items in large databases. In *Proc. of the ACM SIGMOD Conference on Management of Data, pages 207-216*, May 1993.

[2] R. Agrawal 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] S. Chawla, S. Shekhar, W. Wu, and U. Ozesmi. Extending Data Mining for Spatial Applications: A Case Study in Predicting Nest Locations. In *Proc. Intl. Conf. on ACM SIGMOD Workshop on Research Issues in Data Mining and Knowledge Discovery*, 2000.

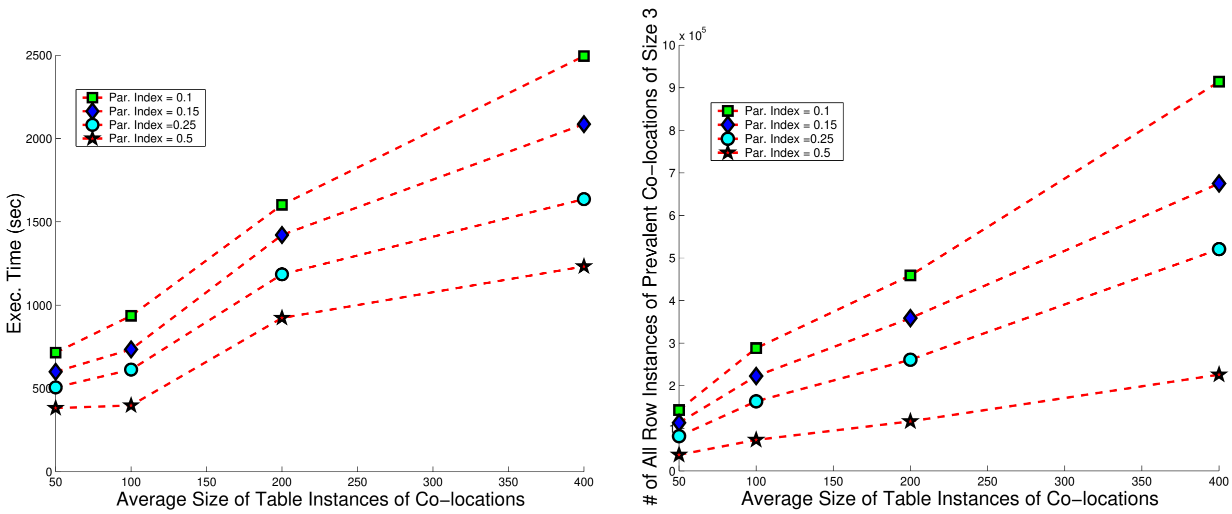

Figure 8: Experimental Results of $Case_1$, $Case_6$, $Case_7$, $Case_8$ a) Exec. Time of Different Participation Index Threshold. b) Total Number of Row Instances in All **Prevalent** Co-locations of Size 3 for Different Participation Index Threshold.

[6] Y. Chou. In *Exploring Spatial Analysis in Geographic Information System, Onward Press, (ISBN: 1-56690-119-7)*, 1997.

[7] N. Cressie. In *Statistics for Spatial Data, Wiley-Interscience, (ISBN:0-471-00255-0)*, 1993.

[8] G. Greenman. Turning a map into a cake layer of information. *New York Times*, Feb 12 2000.

[9] R.H. Guting. An Introduction to Spatial Database Systems. In *Very Large Data Bases Jorunal(Publisher: Springer Verlag)*, October 1994.

[10] R.J. Haining. Spatial Data Analysis in the Social and Environmental Sciences. In *Cambridge University Press, Cambfidge, U.K*, 1989.

[11] J. Han and Y. Fu. Discovery of multiple-level association rules from large databases. In *Proc. 1995 Int. Conf. Very Large Data Bases, pages 420-431, Zurich, Switzerland*, September 1995.

[12] J. Han, J. Pei, and Y. Yin. Mining Frequent Patterns without Candidate Generation. In *Proc. ACM-SIGMOD Conf. on Management of Data*, 2000.

[13] J. Hipp, U. Guntzer, and G. Nakaeizadeh. Algorithms for Association Rule Mining - A General Survey and Comparison. In *Proc. ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2000.

[14] Issaks, Edward, and M. Svivastava. Applied Geostatistics. In *Oxford University Press, Oxford*, 1989.

[15] K. Koperski, J. Adhikary, and J. Han. Spatial Data Mining: Progress and Challenges. In *Workshop on Research Issues on Data Mining and Knowledge Discovery (DMKD'96)*, 1996.

[16] K. Koperski and J. Han. Discovery of Spatial Association Rules in Geographic Informa tion Databases. In *Proc. Fourth International Symposium on Large Spatial Data bases, Maine. 47-66*, 1995.

[17] P. Krugman. Development, Geography, and Economic theory. In *MIT Press, Camgridge, MA*, 1995.

[18] 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.

[19] D. Mark. Geographical Information Science: Critical Issues in an Emerging Cross-disciplinary Research Domain. In *NSF Workshop*, February 1999.

[20] 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.

[21] 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.

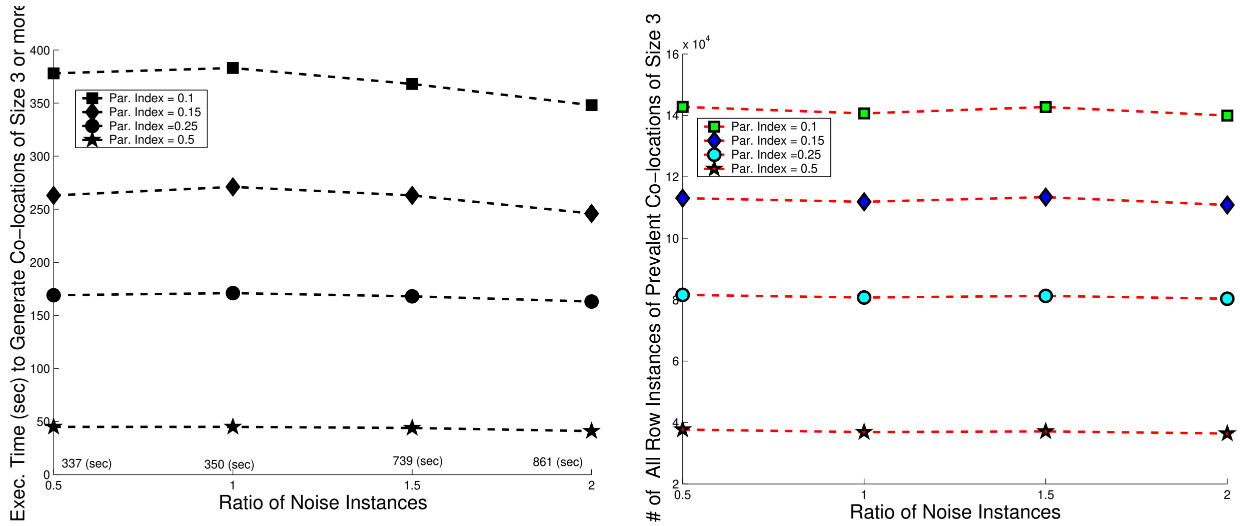

Figure 9: Experimental Results of Case$_1$, Case$_9$, Case$_{10}$, Case$_{11}$. Execution time to generate co-locations of size 2 for different noise ratios is labeled on the x-axis. a) Exec. Time to Generate Co-locations of Size 3 or More for Different Participation Index Threshold. b) Total Number of Row Instances in All **Prevalent** Co-locations of Size 3 for Different Participation Index Thresholds.

[22] 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.

[23] S. Shekhar and S. Chawla. Spatial Databases: Issues, Implementation and Trends. *Prentice Hall (under contract)*, 2001.

[24] 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.

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

[26] S. Shekhar, C.T. Lu, and P. Zhang. Detecting Graph-based Spatial Outliers: Algorithms and Applications. In *The Seventh ACM SIGKDD Intl. Conf. on Knowledge Discovery and Data Mining*, 2001.

[27] 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.

[28] R. Srikant and R. Agrawal. Mining Generalized Association Rules. In *Proc. of the 21st Int'l Conference on Very Large Databases, Zurich, Switzerland*, 1997.

[29] 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.

[30] 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.

[31] 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.

[32] M.F. Worboys. GIS: A Computing Perspective. In *Taylor and Francis*, 1995.

[33] 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: Definition of Neighborhood

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 1 *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 [32]:

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 2 *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.