# Technical Report

Department of Computer Science
and Engineering
University of Minnesota
4-192 EECS Building
200 Union Street SE
Minneapolis, MN 55455-0159 USA

TR 02-033

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

Yan Huang, Shashi Shekhar, and Hui Xiong

October 10, 2002

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

Yan Huang*, Shashi Shekhar, Hui Xiong
Computer Science Department, University of Minnesota
200 Union Street SE, Minneapolis, MN-55455,USA
[*huangyan*, *shekhar*, *huix*]@cs.umn.edu

August 23, 2002

**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 a family of algorithms to mine frequent spatial co-location patterns. Experimental results are provided to show the strength of each algorithm and design decisions related to performance tuning.

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

*Contact author. This work was supported by the Army High Performance Computing Research Center under the auspices of the Department of the Army, Army Research Laboratory cooperative agreement number DAAD19-01-2-0014, the content of which does not necessarily reflect the position or the policy of the government, and no official endorsement should be inferred

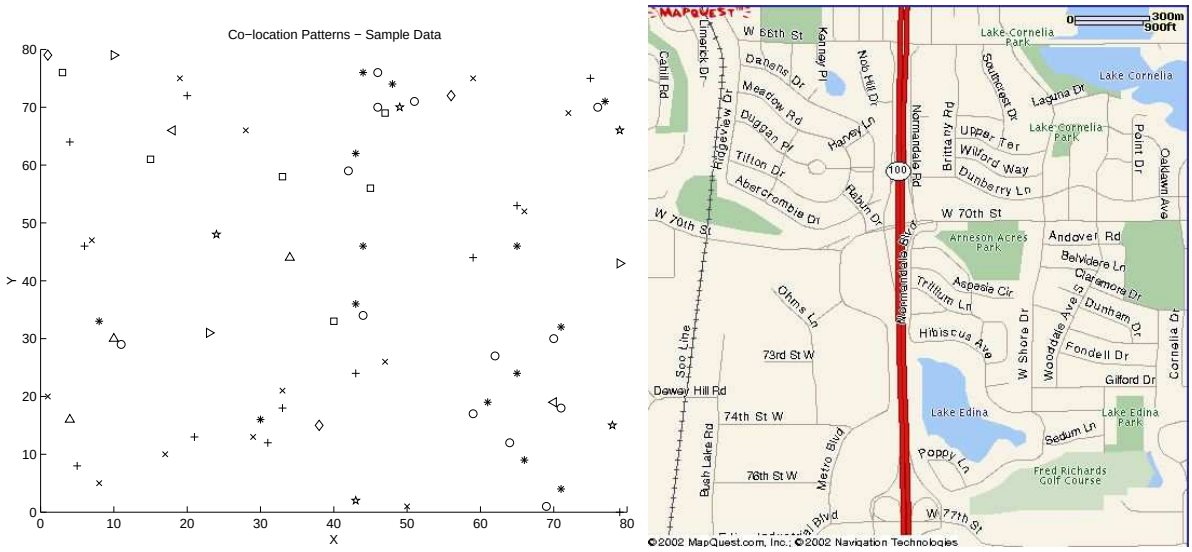
# 1 Introduction

Widespread use of spatial databases [10, 24, 25, 36] is leading to an increasing interest in mining interesting and useful but implicit spatial patterns [9, 16, 20, 23, 32, 26, 28, 5, 29, 35]. For example, E-services are growing along with mobile computing infrastructures such as PDAs and celluar phones. Finding E-services frequently located together is of interest to businesses that want to conduct location sensitive market promotions such as promoting a taxi service for customers who reserve an E-ticket in some locations. In ecology, 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, extremely high precipitation, etc. 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, and tourism [3, 14, 18, 11, 32, 37].

Association rule finding [13, 1, 2, 13, 22, 30, 31, 33] 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 [17] are spatial cases of general association rules where at least one of the predicates is spatial. Association rule mining algorithms [1, 2, 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 instances of boolean spatial features (e.g., drought, needle leaf vegetation). Figure 1 a) shows the frequent co-occurrences of some point spatial feature types represented by different shapes. As can be seen, instances of spatial features in sets {'+', '×'} and {'o', '*'} tend to be located together. Figure 1 b) shows an instance of co-location patterns among extended spatial features, namely road-types, on an urban road map. Highways often have frontage roads nearby in large metropolitan areas, e.g. Minneapolis. Identification of such co-locations is useful in selecting test-sites for evaluating in-vehicle navigation technology [38]. 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.

We formalize the co-location rule mining problem as follows: Given 1) a set T of $K$




(a) (b)

Figure 1: a) Point Spatial Co-location Patterns Illustration. Shapes represent different spatial feature types. Spatial features in sets {'+', '×'} and {'o', '*'} tend to be located together. b) Line String Co-location Patterns Illustration. Highway 100 and Normandale Road are co-located for several hundred meters. Highways are often co-located with frontage roads.

spatial feature types $T = \{f_1, f_2, \ldots, f_K\}$ and their instances $P = \{p_1, p_2, \ldots, p_N\}$, each $p_i \in$ P is a vector < instance-id, spatial feature type, location > where location $\in$ spatial framework S, 2) A symmetric and reflexive neighbor relation R over locations in S, 3) Min prevalence threshold ($min\_prevalence$) and min conditional probability ($min\_cond\_prob$); efficiently find correct and complete set of co-location rules with participation index $> min\_prevalence$ and conditional probability $> min\_cond\_prob$.

**Related Work:** Approaches to discovering co-location rules in the literature can be categorized into two classes, namely spatial statistics and association rules. Spatial statistics-based approaches use measures of spatial correlation to characterize the relationship between different types of spatial features. Measures of spatial correlation include cross-$k$ function with Monte Carlo simulation [7], chi-square tests, correlation coefficients, and regression models [6] 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 candidate

subsets given a large collection of spatial boolean features.

Association rule-based approaches focus on the creation of transactions over space so that an *apriori* like algorithm [2] can be used. Transactions over space can be defined using a window-centric model [27], a reference-feature centric model [17] or an ad-hoc data-partition [21] approach. 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 $k \times k$ 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 be used as prevalence and conditional probability measures as summarized in Table 2 (see **Appendix B**). There are four windows corresponding to four transactions in Figure 2 a). Two windows contain $B$ and only one contains both $B$ and $C$. An example of an association rule of this model is: $an\ instance\ of\ type\ B\ in\ a\ window \rightarrow an\ instance\ of\ type\ C\ in\ this\ window$ with $\frac{1}{2} = 50\%$ 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.

Another approach is based on the choice of a reference spatial feature [17]. 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 [17]. Transactions are created around instances of one user-specified spatial feature. The association rules are derived using the *apriori* [2] algorithm. The rules found are all related to the reference feature. For example, consider the spatial dataset in Figure 2(a) with three feature types, namely $A, B$ and $C$. Each feature type has two instances. The neighbor relationship between instances is shown as an edge. Co-location $(A, B)$ and $(B, C)$ may be considered to be frequent in this example. Figure 2(b) shows transactions created by choosing $C$ as the reference. Co-location

$(A, B)$ will not be found since it does not involve the 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 an ad-hoc data-partition approach [21] attempts to measure the frequency of a co-location pattern by grouping the spatial instances into disjoint partitions. However, imposing artificial disjoint 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 addition, there may be multiple partitions yielding distinct sets of transactions, which in turn yields different values of support of the co-location. Figure 2 c) shows two possible partitions for the dataset of Figure 2 a), along with the support for co-location $(A, B)$. The ad-hoc approach is partitioning sensitive and thus the prevalence measure is ill-defined. A different grouping may result in different values of the support measure and thus different co-location patterns.

In recent work, we developed an event centric model. For point spatial feature which provides a transaction-free approach by using the concept of neighborhood. 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 a). There are two instances of type $A$ and both have some instance(s) of type $B$ in their neighborhoods. The conditional probability for the co-location rule is: *spatial feature A at location* $l \rightarrow$ *spatial feature type B in neighborhood is 100%*. This yields a well-defined prevalence measure(i.e. support) without the need for transactions. Figure 2 d) illustrates that our approach will identify both $(A, B)$ and $(B, C)$ as frequent patterns.

**Our Contributions:** This paper extends our recent work [27] on the event centric model and makes the following new contributions. First, it presents a generalized algorithm to discover co-location patterns from point spatial datasets. The generalized algorithm includes a novel multi-resolution filter step. Second, it also provides proofs of correctness and completeness for the generalized algorithm. Finally the paper provides an experimental performance evaluation to compare alternative choices for key design decisions, such as the use of a multi-resolution filter.

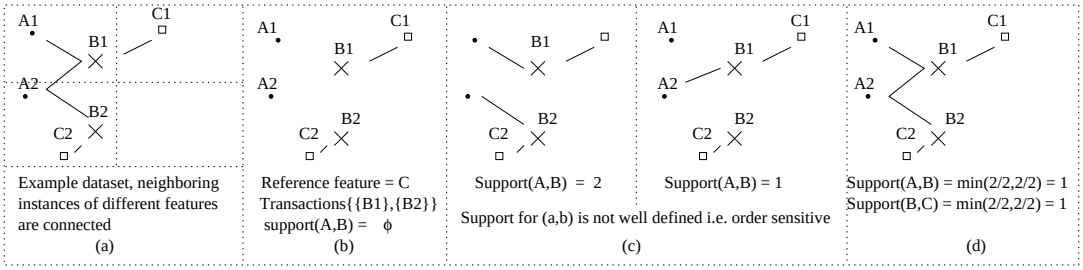



Figure 2: Example to Illustrate Different Approaches to Discovering Co-location Patterns a) Example dataset. Grid is imposed to illustrate window center model b) Ad hoc data partition approach. Support measure is ill-defined and order sensitive c) Reference feature centric model d) Event centric model

**Outline:** Section 2 describes our approach for modeling co-location patterns and the associated measures of prevalence and conditional probability. Section 3 proposes a family of algorithms to mine co-location patterns; an analysis of the algorithms in the areas of correctness, completeness, and computational efficiency is presented in section 4. We present the experimental setup and results in section 5. Finally, section 6 presents the conclusion and future work.

# 2 Our Approach for Mo eling Co-location Patterns

Given the difficulty in creating explicit disjoint transactions from continuous spatial data, this section defines our approach, namely **the event centric** model, to model co-location patterns. We use Figure 3 as an example spatial dataset to illustrate the model. In the figure, each instance is uniquely identified by $T.i$, where $T$ is the spatial feature type and $i$ is the unique id inside each spatial feature type. We define the following basic concepts to facilitate the description of our model.

A **co-location** is a subset of boolean spatial features. A **co-location rule** is of the form: $C_1 \rightarrow C_2(p, cp)$, where $C_1$ and $C_2$ are co-locations, $C_1 \cap C_2 = \emptyset$, $p$ is a number representing the prevalence measure, and $cp$ is a number measuring conditional probability. Prevalence measures and conditional probability measures, called interest measures, are defined differently in different models. The window centric and the reference feature centric models "materialize" transactions and thus can use traditional support and confidence measures.

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: A **neighborhood** of $l$ is a set of locations $L = \{l_1, \ldots, l_k\}$ such that $l_i$ is a neighbor of

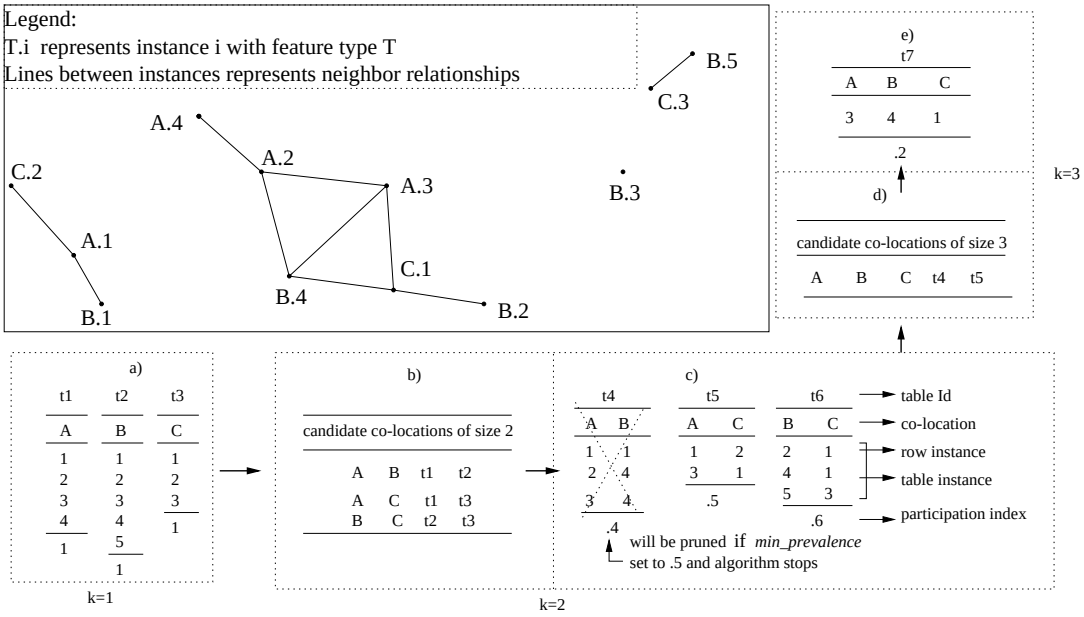



Figure 3: Spatial dataset to illustrate event centric model

$l$ i.e. $(l, l_i) \in R\,(\forall i \in 1 \ldots k)$. This definition satisfies the following two conditions from Topology [36]: First, every location is in some neighborhood because of the reflective neighbor relationship. Second, 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. 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 the application domains. The neighbor relation $R$ 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 a road-map). 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 in the next paragraph to be a prevalence measure to facilitate pruning.

$I = \{i_1, i_2, \ldots, i_k\}$ 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, i.e. elements of $I$ are neighbors to each other. For example, $\{A.3, B.4, C.1\}$ is an instance of co-location $\{A, B, C\}$ in the illustration spatial dataset shown in Figure 3. The table instance of a co-location $C = \{f_1, \ldots, f_k\}$ is the collection of all row instance of C.

The **participation ratio** $Pr(C, f_i)$ for feature type $f_i$ in a co-location $C = \{f_1, \ldots, f_k\}$ is the fraction of instances of $f_i$ which participate in any row instance of co-location $C$. The ratio can be computed as $\frac{|distinct(\pi_{f_i}(table\,instance\,of\,C))|}{|instance\,of\,\{f_i\}|}$, where $\pi$ is a relational projection operation. For example, in Figure 3, row instances of co-location $\{A, B\}$ are $\{(A.1, B.1), (A.2, B.4), (A.3, B.4)\}$. Only two out of five instances, $B.1$ and $B.4$ of spatial feature $B$, participate in co-location $\{A, B\}$. So $Pr((A, B), B) = 2/5 = 0.4$.

The **participation index** of a co-location $C = \{f_1, \ldots, f_k\}$ is $min_{i=1}^{k}\{Pr(C, f_i)\}$. In Figure 3, the participation ratio $Pr(\{A, B\}, B)$ of feature $B$ in co-location $\{A, B\}$ is 0.4 as calculated above. Similarly, $Pr(\{A, B\}, A)$ is 0.75. The $participation\_index(A, B)$ = min(0.75, 0.4) = 0.4. The **conditional probability** of a co-location rule $C_1 \rightarrow C_2$ is the probability of finding an instance of $C_2$ in a neighborhood of instance of $C_1$. It can be computed as $\frac{|distinct(\pi_{C_1}(all\,row\,instance\,of\,C_1 \cup C_2))|}{|instance\,of\,C_1|}$ where $\pi$ is a projection operation.

# 3 ining o-location atterns

As shown in Figure 4, the algorithm takes a set ET of spatial event types, a set E of event instances, user-defined functions representing spatial neighborhood relationships as well as interest measures (e.g prevalence, conditional probability) and thresholds for prevalence based pruning. It assumes that the prevalence measure is monotonic in size of co-locations. The algorithm outputs a set of prevalent co-location rules with the values of the interest measures.

The initialization step assigns starting values to various data-structures used in the algorithm. We note that our prevalence measure evaluates to 1 for all co-locations of size 1. In other words, all co-locations of size 1 are prevalent and there is no need for either the computation of a prevalence measure or prevalence-based filtering. Thus, the set $C_1$ of candidate co-locations of size 1 as well as the set $P_1$ of prevalent co-locations of size 1 are initialized to ET, the set of boolean spatial event types. The set $T_1$ of table instances of size 1 co-location is created by sorting the set E of event instances by event types.

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

**Input:**
(a) E = {Event-ID, Event-Type, Location in Space} representing a set of events;
ET = {Set of boolean spatial event types};
(b) Neighborhood relationship function; pair of spatial points;
(c) Interest measure function (e.g. prevalence, conditional probability);
(d) Threshold on prevalence measure and conditional probability;

**Output:**
A set of co-locations with values of interest measures (i.e. prevalence, conditional probability) satisfying threshold.

**Data Structure:**
k = Co-location size
$C_k$ = set of candidate size k co-locations in iteration k = 1, 2, ..., P
$T_k$ = set of table instances of co-locations in $C_k$ for k = 1, 2, ..., P
$P_k$ = set of prevalent size k co-locations for k = 1, 2, ..., P
$R_k$ = set of co-location rules of size k for k = 1, 2, ..., P
$T\_C_k$ = set of coarse-level table instances of size k co-location in $C_k$ for k= 1, 2, ..., P

**Steps:**
Co-location-size k = 1
$C_1$ = ET;
$P_1$ = ET;
$T_1$ = generate_table_instance($C_1$, E);

Initialize data structure $C_k$, $T_k$, $P_k$, $R_k$, $T\_C_k$ to be empty for k >1
while(not empty $P_k$) do{
$C_{k+1}$ = generate_candidate_colocation ($C_k$, k);
if (fmul = true) then {
$C_{k+1}$ = multi_resolution_pruning ($C_{k+1}$);
}
$T_{k+1}$ = generate_table_instance ($q$, $C_{k+1}$, $T_k$);
$P_{k+1}$ = select_prevalent_colocation($q$, $C_{k+1}$, $T_{k+1}$);
$R_{k+1}$ = generate_colocation_rule ($P_{k+1}$, $T_{k+1}$);
}
return union ($R_2$, ..., $R_{k+1}$);

Figure 4: Overview of algorithms

## 3.1 Generation of Candidate Co-locations

The participation ratio is monotonically non-increasing with the size of the co-location increasing because any spatial feature that participates in a row instance of a co-location $c$ participates in a row instance of a co-location $c'$ where $c' \subseteq c$. The participation index is also monotonic because 1) the participation ratio is monotonic 2) $pi(c \cup f_{k+1}) = \min_{i=1}^{k+1}\{pr(c \cup f_{k+1}, f_i)\} \leq \min_{i=1}^{k}\{pr(c \cup f_{k+1}, f_i)\} \leq \min_{i=1}^{k}\{pr(c, f_i)\} = pi(c)$. Given this property, a spatial feature level pruning approach can be effective. We could also rely on a combinatorial approach and use *apriori_gen* [2] to generate size $k+1$ candidate co-locations from size $k$ prevalent co-locations.

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

is specified in a SQL-like syntax as follows:

```
insert into C_k
select p.item_1, p.item_2,..., p.item_{k-1}, q.item_{k-1}, p.table_instance_id, q.table_instance_id
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 ∈ C_k do
   forall (k − 1)-subsets s of c do
         if (s ∉ L_{k-1}) then delete c from C_k;
```

Note that $L_{k-1}$, $L_k$, and $C_k$ are nested tables [34] where columns table_instance_id refer to table instance of appropriate co-locations.

## 3.2 Generation of Table Instances of Candidate Co-locations

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

```
forall co-location c ∈ C_{k+1}
   insert into T_c /* T_c is the table instance of co-location c */
   select p.instance_1, p.instance_2, ..., p.instance_k, q.instance_k
   from c.table_instance_id_1 p, c.table_instance_id_2 q
   where p.instance_1=q.instance_1, ..., p.instance_{k-1}=q.instance_{k-1}, (p.instance_k,
         q.instance_k) ∈ R;
end;
```

It takes the size $k+1$ candidate co-location set $C_{k+1}$ and table instances of the size $k$ prevalent co-locations as arguments and works as follows: $c$.table_instance_id$_1$ and $c$.table_instance_id$_2$ specify the table instances of the two co-locations joined in *apriori_gen* to produce $c$. Here, a sort-merge join is preferred because the table instances of each iteration can be kept sorted for the next iteration. This follows from a similar property of apriori-gen [2]. Sort order is based on an ordering of the set of item-types to order item-types in a co-location to form the sort-field. Finally, all co-locations with empty table instance will be eliminated from $C_{k+1}$.

The join computation for generating table instances has two constraints, namely a spatial neighbor relationship constraint ($p$.instance$_k$, $q$.instance$_k$) and a combinatorial distinct event-type constraint ($p$.instance$_1$=$q$.instance$_1$, ..., $p$.instance$_{k-1}$=$q$.instance$_{k-1}$).

We examine three strategies for computing this join, a geometric strategy, a combinatorial strategy and a hybrid strategy. These are described in forthcoming subsections. Exploration of other join strategies is beyond the scope of this paper but we may explore such strategies in future work.

**Geometric Approach:** This approach can be implemented by neighborhood relationship-based spatial joins of table instances of prevalent co-locations of size $k$ with table instance sets of prevalent co-locations of size 1. In practice, spatial join operations are divided into a filter step and a refinement step [24] to efficiently process complex spatial data types such as point collections in a row instance. In the filter step, the spatial objects are represented by simpler approximations such as the MBR - Minimum Bounding Rectangle. There are several well-known algorithms, such as plane sweep [4], space partition [15] and tree matching [19], which can then be used for computing the spatial join of MBRs using the overlap relationship; the answers from this test form the candidate solution set. In the refinement step, the exact geometry of each element from the candidate set and the exact spatial predicates are examined along with the combinatorial predicate to obtain the final result.

**Combinatorial Approach:** The combinatorial join predicate (i.e. $p.\text{instance}_1 = q.\text{instance}_1, \ldots, p.\text{instance}_{k-1}=q.\text{instance}_{k-1}$) can be processed efficiently using a sort-merge join strategy [8], since the set of feature types is ordered and tables $c.\text{table\_instance\_id}_1$ and $c.\text{table\_instance\_id}_2$ are sorted. The resulting tuples are checked for the spatial condition ($p.\text{instance}_k$, $q.\text{instance}_k$) to get the row-instance in the result.

**Example 1** *In Figure 3, table 4 of co-location* $\{A, B\}$ *and table 5 of co-location* $\{A, C\}$ *are joined to produce the table instance of co-location* $\{A, B, C\}$ *because co-location* $\{A, B\}$ *and co-location* $\{A, C\}$ *were joined in apriori_gen to produce co-location* $\{A, B, C\}$ *in the previous step. In the example, row instance* $\{3, 4\}$ *of table 4 and row instance* $\{3, 1\}$ *of table 5 are joined to generate row instance* $\{3, 4, 1\}$ *of co-location* $\{A, B, C\}$ *(Table 7). Row instance* $\{1, 1\}$ *of table 4 and row instance* $\{1, 2\}$ *of table 5 fail to generate row instance* $\{1, 1, 2\}$ *of co-location* $\{A, B, C\}$ *because instance 1 of B and instance 2 of C are not neighbors.*

**Hybrid approach:** This approach chooses the more promising of the spatial and combinatorial approaches in each iteration. In our experiment, it picks the spatial approach

to generate table instances for co-locations of size 2 and the combinatorial approach for generating table instances for co-locations greater than size 2.

### 3.3 Pruning

Candidate co-locations can be pruned using the given threshold $\theta$ on the prevalence measure. In addition, multi-resolution pruning can be used for spatial dataset with strong auto-correlation [7], i.e., where instances of each spatial feature types tend to be located near each other.

**Prevalence-Based Pruning:** We first calculate the participation indexes for all candidate co-locations in $T_{k+1}$. Computation of the participation indexes can be accomplished by keeping a bitmap of size cardinality($f_i$) for each feature $f_i$ of co-location $C$. One scan of the table instance of $C$ will be enough to put 1s in the corresponding bits in each bitmap. By summarizing the total number of 1s ($p_{f_i}$) in each bitmap, we obtain the participation ratio of each feature $f_i$ (divide $p_{f_i}$ by |instance of $f_i$|). In Figure 3 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 the participation indexes are determined, 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.

**Multi-resolution Pruning:** Multi-resolution pruning is learned on a summary of spatial data at a coarse resolution using a simple recti-linear grid. We combine all instances of a spatial feature $f$ in each cell $(x, y)$ in the grid as a new coarse instance $< (x, y), f, m >$ in the coarse space where $m$ is the number of instances of spatial point feature $f$ in cell $(x, y)$. For each candidate co-location generated by *apriori_gen*, we generate its coarse

table instance using new coarse instances and its coarse participation index based on the coarse table instance. Multi-resolution pruning eliminates a co-location if its coarse participation indexes fall below the user given threshold. We illustrate the idea with one example now.

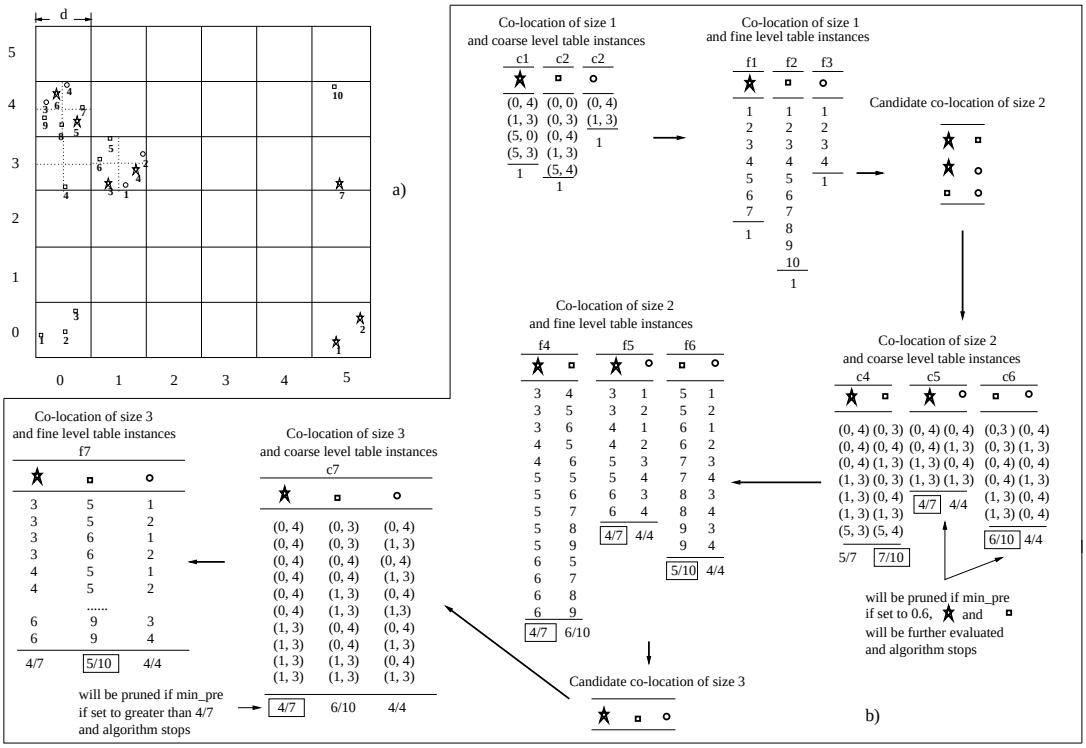



Figure 5: Multi-resolution Co-location Miner Algorithm Illustration

In Figure 5 a), different shapes represent different point spatial feature types. Every instance has a unique ID in its spatial feature type and is labeled below it in the figure. A grid with uniform cell size $d$ is super-imposed on the dataset. Cells (i,j) refer to cells with an x-axis index of i and a y-axis index of j. In this grid, two cells are coarse-neighbors if their centers are in a common square of size $d \times d$, which imposes an 8-neighborhood(North, South, East, West, North East, North West, South East, South West) on the cells. For example, cell-pairs ((0, 3), (0, 4)), ((0, 3), (1, 3)) and ((0, 4), (1, 3)) illustrate coarse-neighbors. This coarse-neighborhood definition guarantees that two cells are neighbors if there exists a pair of points from each of the two cells which are neighbors in the original dataset.

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

Next we calculate the participation indexes for all candidate co-locations based on

the coarse table instances. For each spatial feature $f_i$, we add up all the counts of point instances in each coarse instance with 1s in its corresponding bitmap ($p_{f_i}$) and divide this by |instance of $f_i$| to get the coarse-participation ratio of feature $f_i$. For example, coarse $Pr((\star, \circ), \star) = 4/7$ since there are 4 coarse row instances of $(\star, \circ)$ and 7 fine-grain instances of $\star$. Similarly, coarse $Pr((\star, \circ), \circ) = 4/4 = 1$, yielding coarse participation $index(\star, \circ) = min(4/7, 4/4) = 4/7$. Figure 5 b) shows coarse table instances of co-locations $(\star, \square)$, $(\star, \circ)$ *and* $(\star, \square)$. If the threshold for prevalence is set to 0.6, then co-location c5 can be pruned by multi-resolution pruning. We also note that the sizes of coarse table instances are smaller than the sizes of table instances at fine resolution. This shows the possibility of computation cost saving via multi-resolution pruning for clustered datasets. Finally, the examples in Figure 5 b) show that the coarse participation ratios and participation indexes never underestimate the true participation indexes of the original dataset.

### 3.4 Generating Co-location Rules

The gen_rules generates all the co-location rules with the user defined *min_prev* and *min_cond_prob*. 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: $\frac{|distinct(\pi_{C_1}(all\,row\,instance\,of\,C_1 \cup C_2))|}{|instance\,of\,C_1|}$ where $\pi$ is a projection operation. Bitmaps or other data structures can be used for efficient computation using the same strategies for prevalence-based pruning.

## 4 nalysis of t e o-location ining lgorit ms

In this section, we analyze the co-location mining algorithms in the areas of completeness, correctness, and computational complexity.

### 4.1 Completeness and Correctness

**Lemma 1** *The participation ratio and participation index are monotonically non-increasing as the size of the co-location increases.*

**Proof:** The participation ratio is monotonic because a spatial feature instance that participates in a row instance of a co-location $c$ participates in a row instance of a co-location

$c'$ where $c' \subseteq c$. The participation index is also monotonic because 1) the participation ratio is monotonic and 2)$pi(c \cup f_{k+1}) = \min_{i=1}^{k+1}\{pr(c \cup f_{k+1}, f_i)\} \leq \min_{i=1}^{k}\{pr(c \cup f_{k+1}, f_i)\} \leq \min_{i=1}^{k}\{pr(c, f_i)\} = pi(c)$. Given this property, spatial feature level pruning can be effective. We can also rely on a combinatorial approach and use *apriori_gen* [2] to generate size $k+1$ candidate co-locations from size $k$ prevalent co-locations.

**Lemma 2** *The coarse participation index computed by multi-resolution pruning never underestimates the true participation indexes of the original dataset. The candidate co-location set found is a superset of the prevalent co-location set on the original dataset.*

**Proof** : When co-location size = 1, the value of the coarse participation index and the true participation index is 1, so Lemma 2 is trivially true. Suppose Lemma 2 is true for co-locations size=k. Let us consider the case that co-location size is equal to k+1. For each candidate co-location $C$ of size $k+1$ generated from the *apriori_gen* by joining $C_1$ and $C_2$ of size $k$, we generate its coarse instance table by joining the coarse instance tables of $C_1$ and $C_2$. Because Lemma 2 is true for co-locations of size $k$, the candidate co-location set of size $k$ found is a superset of the prevalent co-location set on the original dataset. Thus $C_1$ and $C_2$ are in the candidate co-location set in the previous iteration and their coarse level table instances are available to be joined to produce the coarse level table instance of $C$. The table join to produce the coarse table instance of $C$ has the following property: if $Neighbor_R(p_1, p_2)$ is in the original dataset, then coarse $Neighbor(cell\, c_1, cell\, c_2)$ will be in the coarse-level dataset given $p_1 \in c_1$ and $p_2 \in c_2$. When we calculate the coarse participation index, any spatial feature instance which participates in the co-location in the original dataset will contribute to the counts during the coarse participation ratio calculation. So the coarse participation ratios never underestimates the true participation ratios, implying that the coarse participation index never underestimates the true participation index and that the pruning will not eliminate any truly prevalent co-location. Thus the candidate co-location set after multi-resolution pruning is a superset of the prevalent co-location set on the original dataset.

**Lemma 3** *The Co-location Miner algorithm is complete.*

**Proof** : The *spatial join* produces 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 these two conditions in the join predicate will be generated. The schema

level pruning using *apriori_gen* is complete due to the monotonicity of the participation index as proved in Lemma 1. Then we prove that the join of the table instances of $C_1$ and $C_2$ to produce the table instance of $C$ is complete. According to the 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 $I$. Enumeration of the subsets of each of the prevalent co-locations ensures that no spatial co-location rules with both high prevalence and high conditional probabilities are missed. We then prove that multi-resolution pruning does not affect completeness. By Lemma 2, the co-location set found is a superset of the prevalent co-location set on the original dataset. Thus multi-resolution pruning does not falsely eliminate any prevalent co-location.

**Lemma 4** *The Co-location Miner is correct.*

**Proof**: We will only show that the row instance of each co-location is correct, as that will imply the correctness of the participation index values and that of each co-location meeting the user specified threshold. An instance $I_1 = \{i_{1,1}, \ldots, i_{1,k}\}$ of $C_1 = \{f_1, \ldots, f_{k+1}\}$ and an instance $I_2 = \{i_{2,1}, \ldots, i_{2,k}\}$ of $C_2 = \{f_1, \ldots, f_{k-1}, f_{k+1}\}$ is joined to produce an instance $I_{new} = \{i_{1,1}, \ldots, i_{1,k}, i_{2,k}\}$ of $C = \{f_1, \ldots, f_{k+1}\}$ if: 1) all elements of $I_1$ and $I_2$ are the same except $i_{1,k}$ and $i_{2,k}$; 2) $i_{1,k}$ and $i_{2,k}$ are neighbors. The schema of $I_{new}$ is apparently $C$, and elements in $I_{new}$ are in a neighborhood because $I_1$ is a neighborhood and $i_{2,k}$ is a neighbor of every element of $I_1$.

## 4.2 Computational Complexity Analysis

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

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

combinatorial approach becomes cheaper than the geometric approach. This is due to its exploitation of the sort-merge join strategy while keeping each table instance sorted, resulting in the computation complexity of $O(N)$ where N is the size of table instances. In a hybrid approach, we pick the cheaper of the two basic strategies in each iteration to achieve the best overall cost.

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

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

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

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

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

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

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

The first term of the ratio is controlled by the "clumpiness" (the average number of instances of the spatial features per grid cell) of the locations of spatial features. The second term is controlled by the filtering efficiency of the coarse instance level pruning.

When the locations of spatial features are clustered, the sizes of the fine level table instances are much greater than the sizes of the coarse level table instances and the time needed to generate fine level table instances is greater than the time needed to generate coarse level table instances. In our experiments, as described in the next section, we use the parameter $m_{clump}$, which controls the number of instances clumping together for each spatial feature, to evaluate the first term, and we use the parameter $m_{overlap}$, which represents the possible false candidate ratios to evaluate the second term. From the formula, we can see that the *Multi-resolution Co-location Miner* is likely to be more efficient than the *Co-location Miner* when the locations of spatial features are clustered and the false candidate ratio is high.

# 5 Experi ental erfor ance Evaluation

## 5.1 Experiment Design

Figure 6 describes the experimental setup to evaluate the impact of design decisions on the relative performance of the co-location miner algorithm. We evaluated the performance of the algorithms with synthetic data generated using a methodology similar to methodologies used to evaluate algorithms for mining association rules [2]. Synthetic datasets allow better control towards studying the effects of interesting parameters. A data-flow diagram of the data generation process is shown in Figure 6. The process began with the generation of core co-location subsets of spatial features. To generate a subset of features, we first chose the size of the subset from a Poisson distribution with mean ($\lambda_1$). Then a set of features for this core co-location pattern was randomly chosen. For each core co-location, $m_{overlap}$ maximal co-locations were generated by appending one more spatial feature to a core co-location. The larger $m_{overlap}$ is, the more false candidate *apriori_gen* generates. The size of each table instance of each co-location was chosen from another Poisson distribution with mean $\lambda_2$. Next, we generated the set of neighborhoods for co-locations instances using the size of their table instances from the previous step. $m_{clump}$ point locations for each feature in the co-location were embedded inside a neighborhood of size $d$. The locations of neighborhoods were chosen at random in the overall spatial framework. For simplicity, the shape of the overall spatial framework was a rectangle of size $D_1 \times D_2$ and the size of each neighborhood was $d \times d$. The final step involved adding noise. The model for noise used two parameters, namely the ratio of noise features $r_{noise\_f}$ and the number of noise instances $p_{noise\_n}$. Noise was added

Table 1: Parameters Used to Generate the Synthetic Data

| **Parameter** | **Definition** | **C1** | **C2** |
|---|---|---|---|
| $N_{co\_loc}$ | The number of core co-locations | 5 | 4 |
| $\lambda_1$ | The parameter of the Poisson distribution to define the size of the core co-locations | 5 | 5 |
| $\lambda_2$ | The parameter of the Poisson distribution to define the size of the table instance of each co-location when $m_{clump} = 1$ | 50 | 50 |
| $D_1 \times D_2$ | The size of the spatial framework | $10^6 \times 10^6$ | $250 \times 1{,}000$ |
| $d$ | The size of the square to define a co-location | 10 | 10 |
| $r_{noise\_f}$ | The ratio the of number of noise features over the number of features involved in generating the maximal co-locations | .5 | .5 |
| $r_{noise\_n}$ | The number of noise instances | 50,000 | 1,000 |
| $m_{overlap}$ | The number of co-location generated by appending one more spatial feature for each core co-location | 1 | 1 |
| $m_{clump}$ | The number of instances generated for each spatial feature in a neighborhood for a co-location | 1 | 1 |

by generating a set of instances of features from a set of noise features disjoint with the features involving generation of core co-locations and placing those at random locations in the global spatial framework.

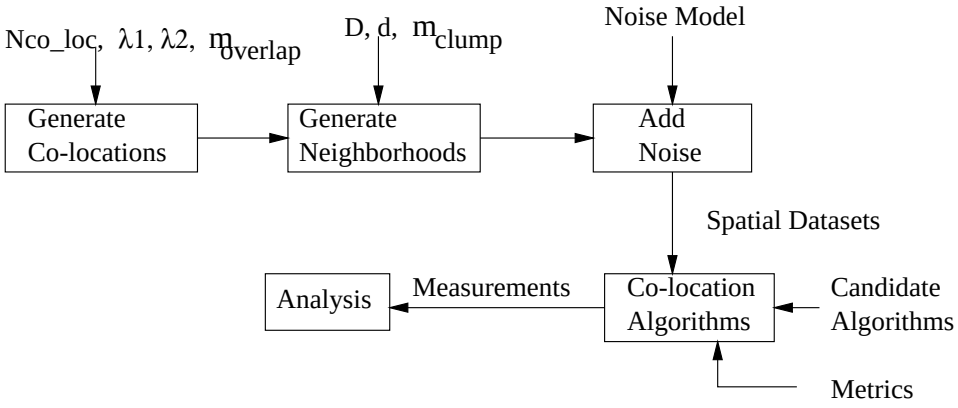
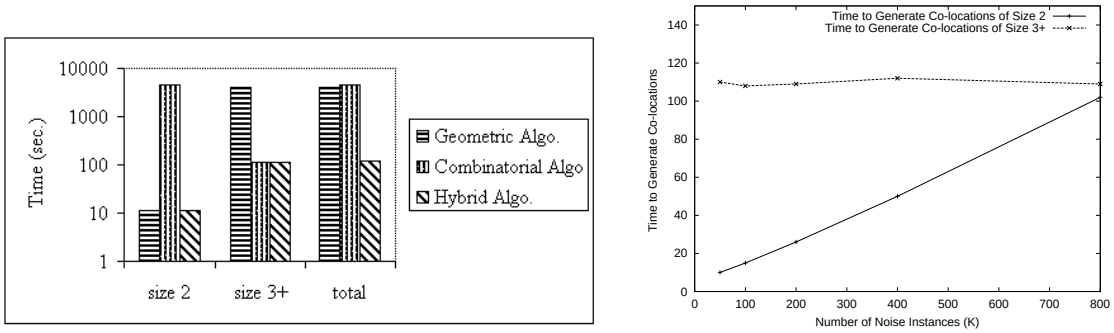



Figure 6: Experimental Setup and Design

Datasets were used by the co-location algorithm driver module to collect the performance statistics, as shown in Figure 6. The execution time was measured using the "time" function 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 measurements are summarized in the form of plots and tables reported in the following section.

## 5.2 Comparing Strategies for Generating Table Instances

We compared the geometric, the combinatorial, and the hybrid strategies. The base dataset, generated using parameter values in column C1 of Table 1, used a rectangle spatial framework of size $10^6 \times 10^6$, a square neighborhood of size $10 \times 10$, an average co-location size of 5, an average table instance size of 50 when $m_{clump} = 1$, a noise feature ratio of 0.5, a noise number of 50,000, and an overlapping degree of 1. Figure 7 (a) shows the execution times for the three candidates with the prevalence threshold set to .9. The second column reports the execution time needed to discover co-locations of size 2.



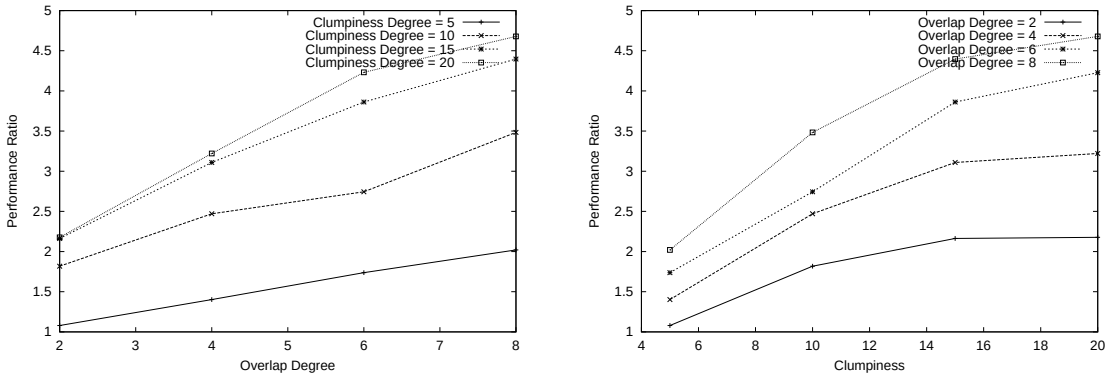

(a) (b)

Figure 7: (a)Relative Performance of Geometric, Combinatorial, and Hybrid Algorithms (b) Noise Effect on Co-location Miner

As can be seen, the geometric strategy is faster than the combinatorial strategy for generating size-2 co-locations. Spatial-join data structures help the geometric algorithm in this step. The remaining columns report the total execution time to discover all the co-locations as well as the time to discover co-locations of size 3 or more, given prevalent co-locations of size 2. In these cases, the combinatorial algorithm is orders of magnitude faster than the geometric algorithm. A sort-merge join strategy (e.g apriori-gen [2]) helps the combinatorial algorithm. The hybrid strategy uses the geometric algorithm for discovering prevalent co-locations of size 2 and the combinatorial algorithm for discovering larger co-locations. Thus, it is expected to achieve the best overall performance. Our experimental results confirm this, as shown in Figure 7 (a).

## 5.3 Effect of the Filter

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

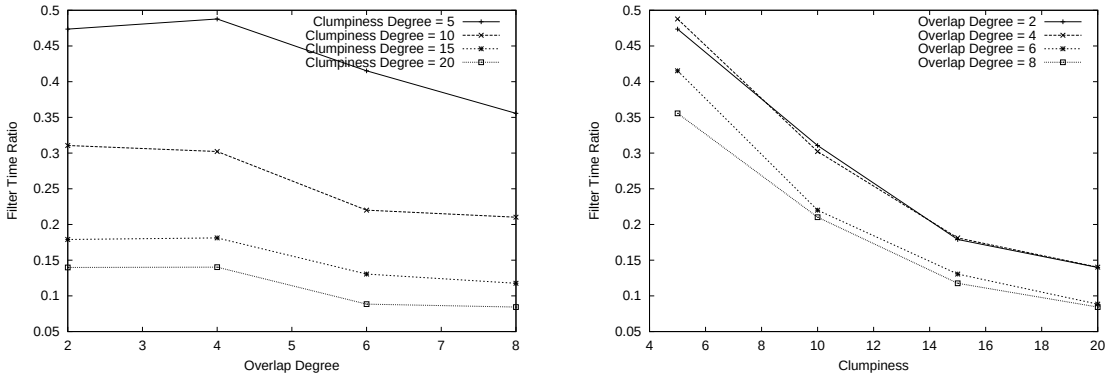



(a) (b)

Figure 8: Performance Ratio a) By Overlap Degree b) By Clumpiness Degree

Figure 8 summarizes the performance gain by using the Multi-resolution filter. The x-axis represents the overlap degree, which controls the false candidates generated by *apriori_gen* in the first figure or the "clumpiness" of locations of instances of spatial feature type in the second figure. The y-axis represents the ratio of run-time of the *Co-location Miner* without the multi-resolution filter to the run-time with the multi-resolution filter. The results show that, as the degree of overlap and the number of false candidates increase, the running time is reduced by a factor of 1 to 4.5.

Figure 9 summarizes the ratio of the computation time for multi-resolution pruning and that for prevalence-based pruning. Similarly, the x-axis represents the overlap degree




(a) (b)

Figure 9: Filter Time Ratio (a) By Overlap Degree (b) By Clumpiness Degree

or the "clumpiness" of the locations of each spatial feature type. The overhead of the multi-resolution filter as a fraction of prevalence-based pruning decreases when the degree of overlap or clumpiness increases. Clumpiness strongly affects the overhead, reducing it from 0.45 to 0.1.

### 5.4 Effect of Noise

The base dataset, generated using parameter values in column C1 of Table 1, used a rectangle spatial framework of size $10^6 \times 10^6$, a square neighborhood of size $10 \times 10$, an average co-location size of 5, an average table instance size of 50 when $m_{clump} = 1$, a noise feature ratio of 0.5, a noise number of 50,000, and an overlapping degree of 1. Then we increased the noise instances up to 800,000 and measured the performance, as shown in Figure 7 (b). The execution time for discovering co-locations of size 2 is and 3+ are shown in the figure. We note that noise-level affects the execution time to discover co-locations of size 2 but does not affect the execution time to discover larger co-locations given co-locations of size 2. In other words, noise is filtered out during the determination of co-locations of size 2.

## 6 onclusion an Future Work

In this paper, we formalized the co-location problem and showed the similarities and differences between the co-location rules problem and the classic association rules prob-

lem as well as the difficulties in using traditional measures (e.g. support, confidence) created by implicit, overlapping and potentially infinite transactions in spatial data sets. We proposed the notion of user-specified neighborhoods in place of transactions to specify groups of items and defined interest measures that are robust in face of potentially infinite overlapping neighborhoods. We defined a new spatial measure of conditional probability as well as a new monotonic measure of prevalence to allow iterative pruning. *The Co-location Miner*, a generalized algorithm for mining co-location patterns was presented and analyzed for correctness, completeness and computation cost. Design decision in the proposed algorithm were evaluated using theoretical and experimental methods. Empirical evaluation shows that the geometric strategy performs much better than the combinatorial strategy when generating size-2 co-location; however, it becomes slower when generating co-locations with more than 2 features. The hybrid strategy integrates the best features of the above two approaches. Experimental results show that *the Co-location Miner* is tolerant of noise and provides the best overall performance. Furthermore, when the locations of the features tend to be spatially clustered, which is often true for spatial data due to spatial-autocorrelation, the Multi-resolution filter will significantly reduce computation cost of proposed algorithm.

Several questions remain open. The co-location mining problem should be investigated to account for extended spatial data types, such as line segments, polygons and circles. We briefly discuss the potential approach to deal with extended spatial objects in Appendix A. We considered only boolean spatial features here. In the real world, the features can be categorical and continuous. There is a need to extend the co-location mining framework to handle continuous features. Finally, if the locations of features change over time, it should be possible for us to identify some temporal-spatial co-location patterns.

## cknowle g ents

We thank Raymod T. Ng.(University of British Columbia,Canada), Jiawei Han (Simon Fraser University, Canada), James Lesage, and Sucharita Gopal(Boston University) for valuable insights. We also thank C.T. Lu, Xiaobin Ma, and Pusheng Zhang (University of Minnesota) for their valuable feedback on early versions of this paper. We would also like to express our thanks to Kim Koffolt for her timely and detailed feedback to help improve the readability of this paper.

# References


[1] R. Agarwal, 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. Agarwal and R. Srikant. Fast algorithms for Mining association rules. In *Proc. of the 20th VLDB*, 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.

[6] Y. Chou. Exploring spatial analysis in geographic information system. *Onward Press (ISBN: 1566901197)*, 1997.

[7] N.A.C. Cressie. Statistics for spatial data. *John Wiley and Sons, (ISBN:0471843369)*, 1991.

[8] G. Graefe. Sort-merge-join: An idea whose time has(h) passed? In *Proc. of IEEE Conf. on Data Engineering*, 1994.

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

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

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

[12] 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.

[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. *Oxford University Press, Oxford*, 1989.

[15] D. J. DeWitt J. M. Patel. Partition Based Spatial-Merge Join. In *Proc. of the ACM SIGMOD Conference on Management of Data, pp.259-270, Montreal, Canada*, June 1996.

[16] 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.

[17] K. Koperski and J. Han. Discovery of Spatial Association Rules in Geographic Informa020111366X tion Databases. In *Proc. Fourth International Symposium on Large Spatial Data bases, Maine. 47-66*, 1995.

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

[19] S. T. Leutenegger and M. A. Lopez. The Effect of Buffering on the Performance of R-Trees. In *Proc. of the ICDE Conf., pp 164-171*, 1998.

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

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

[22] 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.

[23] 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.

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

[25] 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.

[26] S. Shekhar and Y. Huang. Categorization of Spatial Data Mining Techniques. *Scientific Data Mining, working chapter*, 2001.

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

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

[29] S. Shekhar, P. Schrater, W.R. Raju, and W. Wu. Spatial Contextual Classification and Prediction Models for Mining Geospatial Data. *IEEE Transactions on Multimedia (special issue on Multimedia Databases)*, 2002.

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

[31] 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.

[32] 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.

[33] 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.

[34] J. Ullman. Principles of Database and Knowledge-Base Systems. *Vol.1. Computer Science Press*, 1988.

[35] W. Wang, J. Yang, and R. Muntz. STING: A statistical information grid approach to spatial data mining. In *Matthias Jarke, Michael J. Carey, Klaus R. Dittrich, Frederick H. Lochovsky, Pericles Loucopoulos, and Manfred A. Jeusfeld, editors, Twenty-Thrid International Conference on Very Large Data Bases, pages 186-195, Athens, Greece*, 1997.

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

[37] 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.

[38] Yilin Zhao. Vehicle Location and Navigation Systems. *Artech House ITS Series (ISBN: 0890068615)*, April 1997.

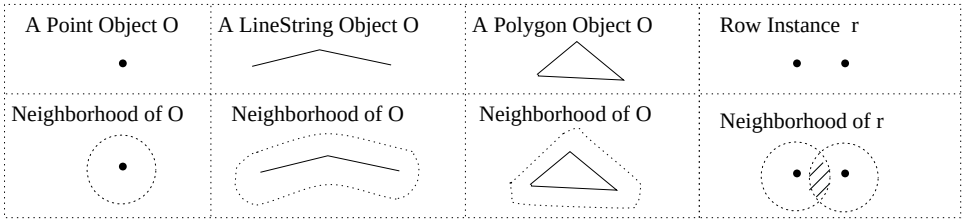



Figure 10: Neighborhood Illustration for various Spatial Objects

# ppen ix : pproac es for xten e bjects

We will use a Euclidean distance based neighborhood here to simplify the discussion. $N(p)$, the size-$d$ Euclidean neighborhood of a point location $p$ ,is a circle of radius $d$ with $p$ as its center. $N(o)$, the size-$d$ neighborhood of an extended spatial object (e.g polygon, line-string) is defined by a buffer operation as shown in Figure 10. Object $o_i$ and $o_j$ are e-neighbors if and only if $o_i \cap N(o_j)$ as well as $o_j \cap N(o_i)$ are non-empty. Euclidean neighborhood $N(f_j)$ of feature $f_j$ is the union of $N(i_l)$ for all instance $i_l$ of $f_j$. Euclidean neighborhood $N(C)$ for a co-location $C = \{f_1, \ldots, f_k\}$ is the intersection of $N(f_i)$ for $f_i$ in $C$.

$I = \{i_1, i_2, \ldots, i_k, B\}$ 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, i.e. elements of $I$ are neighbors to each other. The last element $B$ in $I$ represents $\bigcap_{i_j \in I} N(i_j)$.

The **participation ratio** $Pr(C, f_i)$ for feature type $f_i$ in a co-location $C = \{f_1, \ldots, f_k\}$ is $Pr(p \in \bigcap_{i_j \in C} N(f_j) | p \in f_i)$, the probability that a point $p$ in $f_i$ has points belonging to all features in $C$ within a neighborhood of $P$, assuming a symmetric neighbor relationship. We illustrate the definition of the participation ratio for extended linear object features using length. A similar measure for polygonal features can be defined using area in place of length. Participation ratio $pr(C, f_i)$ for linear spatial feature $f_i$ can be computed as $\frac{length(intersection(f_i, \bigcap_{f_j \in C} N(f_j)))}{length(f_i)}$. For example, an analysis of an urban road map may reveal 100 miles of freeway, of which 25 miles are within $N(frontage\,road)$, yielding $Pr((freeway,\ frontage\,road) = 25/100 = 0.25)$. Participation ratio $Pr(C, f_l)$ can be computed from table instance of $C$ in terms of $\frac{length(union_{I \in table\,instance\,of\,C}(intersection(i_j, B(I))))}{length(union_{i_j \in f_l}(i_j))}$.

The **conditional probability** of a co-location rule $C_1 \rightarrow C_2$ is the probability of finding an instance of $C_2$ in a neighborhood of instance of $C_1$, i.e. $Pr(p \in N(C_2) | p \in N(C_1))$. It can be computed as $\frac{area(N(C_1 \bigcup C_2))}{area(N(C_1))}$ using the table instances of $C_1$ and $C_1 \bigcup C_2$.

# ppen ix B: Interest easures for Different o els

Table 2: 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 $C_1)$ |