

A Join-less Approach for Co-location Pattern Mining: A Summary of Results

Jin Soung Yoo *, Shashi Shekhar, Mete Celik
Computer Science Department, University of Minnesota, Minneapolis, MN, USA
jyoo,shekhar,mcelik@cs.umn.edu

Abstract

Spatial co-location patterns represent the subsets of features whose instances are frequently located together in geographic space. Co-location pattern discovery presents challenges since the instances of spatial features are embedded in a continuous space and share a variety of spatial relationships. A large fraction of the computation time is devoted to identifying the instances of co-location patterns. We propose a novel join-less approach for co-location pattern mining, which materializes spatial neighbor relationships with no loss of co-location instances and reduces the computational cost of identifying the instances. The join-less co-location mining algorithm is efficient since it uses an instance-lookup scheme instead of an expensive spatial or instance join operation for identifying co-location instances. We prove the join-less algorithm is correct and complete in finding co-location rules. The experimental evaluations using synthetic datasets and real world datasets show the join-less algorithm performs more efficiently than a current join-based algorithm and is scalable in dense spatial datasets.

1 Introduction

The explosive growth of spatial data and widespread use of spatial databases emphasize the need for the automated discovery of spatial knowledge. Spatial data mining [7, 8] is the process of discovering interesting and previously unknown, but potentially useful patterns from spatial databases. Extracting interesting patterns from spatial datasets is more difficult than extracting the corresponding patterns from traditional numeric and categorical data due to the complexity of spatial data types, spatial relationships and spatial autocorrelation [10].

A spatial co-location pattern represents a subset of spa-

tial features whose instances are frequently located in a spatial neighborhood. For example, ecologists have found that Nile Crocodiles and Egyptian Plover birds are frequently co-located. The co-location rule, i.e., Nile Crocodile \rightarrow Egyptian Plover, predicts the presence of Egyptian Plover birds in areas with Nile Crocodiles. Spatial co-location patterns may yield important insights for many applications. For example, a mobile service provider may be interested in mobile service patterns frequently requested by geographically neighboring users. The frequent neighboring request sets can be used for providing attractive location-sensitive advertisements, etc. Other application domains include Earth science, public health, biology, transportation, etc.

Co-location rule discovery presents challenges due to the following reasons: First, it is difficult to find co-location instances since the instances of spatial features are embedded in a continuous space and share neighbor relationships. A large fraction of the computation time is devoted to identifying the co-location instances. Second, it is non-trivial to reuse association rule mining algorithms [3, 5] for co-location pattern mining since there are no pre-defined transactions in many spatial datasets. Thus, a current co-location mining algorithm [9] uses a join-based approach to find co-location instances. Its computational performance suffers, however, due to the large number of joins required as the number of features and their instances increases.

In this paper, we propose a method to materialize the neighbor relationships of a spatial dataset with no duplication of the neighbor relationships and no loss of co-location instances, and present a novel join-less approach for co-location pattern mining. The join-less co-location mining algorithm reduces the computational cost of identifying the instances of co-location patterns using an instance-lookup scheme, and also has a coarse pruning step which can filter candidate co-locations without finding exact co-location instances. We analytically prove our join-less algorithm is correct and complete, i.e., there are no false droppings or false admissions in finding co-location rules. The experimental evaluations using synthetic datasets and real world datasets show the join-less co-location mining algorithm outperforms the join-based algorithm and is scalable

*This work was partially supported by NSF grant 0431141 and Oak Ridge National Laboratory grant. The content of this work does not necessarily reflect the position or policy of the government and no official endorsement should be inferred.

in dense spatial datasets.

The remainder of the paper is organized as follows. Section 2 gives an overview of the basic concepts of co-location pattern mining and the problem definition, and discusses related works. In Section 3, we presents our join-less approach for co-location pattern mining. In Section 4, the analytical analysis of the join-less co-location mining algorithm is given. Section 5 presents the experimental evaluation. The conclusion and future work are discussed in Section 6.

2 Co-location Pattern Mining

In this section, we describe the basic concepts of co-location pattern mining and the problem definition, and discuss the related works.

2.1 Basic Concepts

Given a set of spatial features F , a set of their instances S , and a neighbor relationship R over S , a **co-location** C is a subset of spatial features $C \subseteq F$ whose instances $I \subseteq S$ form a clique using a neighbor relationship R . A **co-location rule** is of the form: $C_1 \rightarrow C_2(p, cp)$, where $C_1 \cap C_2 = \emptyset$, p is the prevalence measure, and cp is the conditional probability. For example, when a spatial neighbor relationship R is a Euclidean distance metric and its threshold value d , two spatial objects are neighbors if they satisfy the neighbor relationship, e.g., $R(A.1, B.1) \Leftrightarrow (\text{distance}(A.1, B.1) \leq d)$. Figure 1 (a) shows an example dataset with three spatial features, A, B and C. Each object is represented by its feature type and the unique instance id of each feature type, e.g., A.1. Identified neighbor objects are connected by solid lines. The instance of a co-location is a set of objects which includes an object of each feature type in the co-location and forms a clique relationship among them. For example, in Figure 1 (a), $\{A.2, B.4, C.2\}$ is an instance of co-location $\{A, B, C\}$ since $\text{feature}(A.2)=A$, $\text{feature}(B.4)=B$ and $\text{feature}(C.2)=C$, and $R(A.2, B.4)$, $R(A.2, C.2)$ and $R(B.4, C.2)$.

The interest of a co-location pattern can be measured by its prevalence and conditional probability [9]. The **conditional probability** $Pr(C_1|C_2)$ of a co-location rule $C_1 \rightarrow C_2$ is the fraction of instances of C_2 in the neighborhood of instances of C_1 , i.e., $Pr(C_1|C_2) = \frac{\text{Number of distinct instances of } C_1 \text{ in instances of } C_1 \cup C_2}{\text{Number of instances of } C_1}$. The participation index is used as a co-location prevalence measure. First, the **participation ratio** $Pr(C, f_i)$ of feature f_i in a co-location $C = \{f_1, \dots, f_k\}$ is the fraction of objects of features f_i in the neighborhood of instances of co-location $C - \{f_i\}$, i.e., $Pr(C, f_i) = \frac{\text{Number of distinct objects of } f_i \text{ in instances of } C}{\text{Number of objects of } f_i}$. The **participation index** $Pi(C)$ of a co-location $C = \{f_1, \dots, f_k\}$

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

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

For example, the participation index value of a size 3 co-location is not greater than the participation index value of any size 2 co-location, e.g., $Pi(\{A, B, C\}) = \frac{2}{5} \leq Pi(\{A, B\}) = \frac{3}{5}$ in Figure 1 (c). Please refer to [9] for the proof of Lemma 1.

2.2 Problem Definition

The formal problem definition for the co-location pattern mining is as follows. We focus on finding a correct and complete set of co-location rules with reducing the computation cost.

Given:

1) A set of spatial features $F = \{f_1, \dots, f_n\}$ and a set of their instances $S = S_1 \cup \dots \cup S_n$ where $S_i (1 \leq i \leq n)$ is a set of instances of feature f_i and each instance $\in S$ is a vector $\langle \text{feature type, instance id, location} \rangle$, where location \in a spatial framework

2) A neighbor relationship R over locations

3) A minimum prevalence threshold (min_prev) and a minimum conditional probability threshold (min_cond_prob)

Find:

A set of co-location rules with participation index $\geq \text{min_prev}$ and conditional probability $\geq \text{min_cond_prob}$.

Objective:

1) Find a correct and complete set of co-location rules.

2) Reduce the computation cost.

Constraints:

1) R is a distance metric based neighbor relationship and has a symmetric property.

2) The spatial dataset is a point dataset.

2.3 Related Work

The problem of mining association rules based on spatial relationships was first discussed in [5]. The work discovers

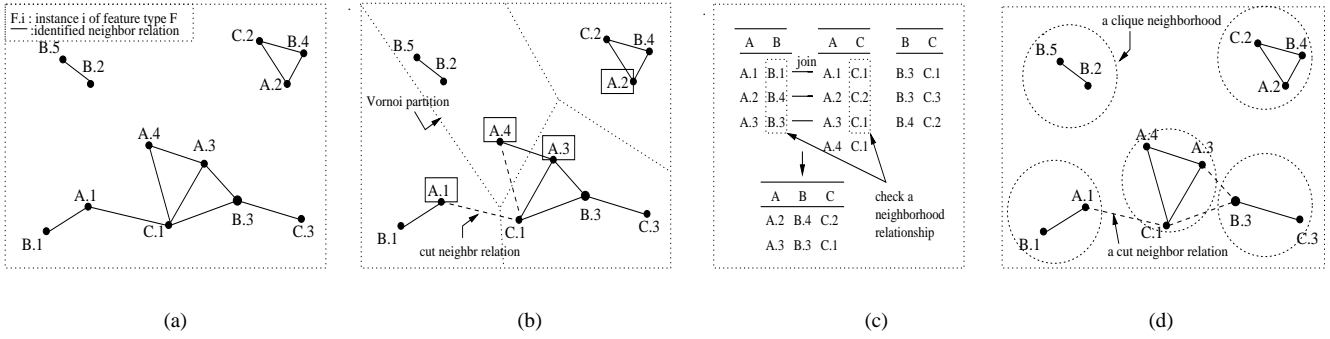


Figure 1. Different approaches for finding co-location instances (a) Example dataset (b) Space partition (c) Instance join (d) Clique partition and partial join

the subsets of spatial features frequently associated with a specific feature, e.g., cancer. Directly applying this method to a co-location problem may not capture our co-location meaning with no specific reference feature.

Previous works on spatial co-location mining have presented different approaches for identifying co-location instances. [6] uses space partitioning for identifying neighboring objects for a frequent neighboring feature set. Figure 1 (b) shows the space partition method to find the neighboring objects of a subset of features, $\{A, C\}$. First, it decides the partition center points with base objects, e.g., feature A objects, A.1, A.2, A.3 and A.4, and decomposes the space from the partitioning points using a geometric approach, i.e., Voronoi diagram, and then finds feature C objects within a distance threshold from the partitioning point in each partition area. In this example, the identified neighboring objects of $\{A, C\}$ are $\{A.3, C.1\}$ and $\{A.2, C.2\}$. However, note that $\{A.1, C.1\}$ and $\{A.4, C.1\}$ are also neighboring objects of $\{A, C\}$ but they are not found by the disjoint space partitions. Thus the distinct space partitioning approach may miss co-location instances across partition areas and generate incorrect results.

[9] proposes an instance join-based co-location mining algorithm similar to *apriori-gen* [3]. First, after finding all neighbor pair objects (size 2 co-location instances) using a geometric method, the method finds the instances of size $k (> 2)$ co-locations by joining the instances of its size $k - 1$ subset co-locations where the first $k - 2$ objects are common. Figure 1 (c) shows the procedure to generate the instances of co-location $\{A, B, C\}$. The instances of co-location $\{A, B\}$ and the instances of co-location $\{A, C\}$ are joined with the first objects, and then the neighbor relationships between the second objects are checked. This approach finds correct and complete co-location instance sets. However, the join-based approach is computationally ex-

pensive with the increase of co-location patterns and their instances. [11] proposes a partial join approach. It transactionizes a continuous spatial data into a set of disjoint clique neighborhoods while keeping track of the spatial neighbor relations not modeled by the transactionization as shown in Figure 1 (d). This approach reduces the number of expensive join operations dramatically in finding co-location instances. However, the performance depends on the distribution of the spatial dataset, exactly the number of cut neighbor relations.

3 A Join-less Approach for Co-location Pattern Mining

In this section, we discuss a join-less approach for mining co-location patterns. First, we describe our method to materialize spatial neighbor relationships, and then present the join-less co-location algorithm.

3.1 Neighborhood Materialization

The ideal neighborhood materialization for co-location mining is to find all maximal clique relationships from an input dataset. However, it is computationally expensive. We propose to materialize disjoint star neighbor relationships as a framework for efficient co-location mining.

Definition 1 Given a spatial object $o_i \in S$ whose feature type is $f_i \in F$, the **star neighborhood** of o_i is defined as a set of spatial objects $T = \{o_j \in S | o_i = o_j \vee (f_i < f_j \wedge R(o_i, o_j))\}$, where $f_j \in F$ is the feature type of o_j and R is a neighbor relationship.

We define the star neighborhood of an object is a set of the center object and objects in its neighborhood whose fea-

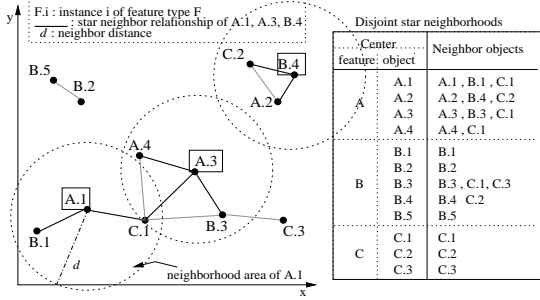


Figure 2. Neighborhood materialization

feature types are greater than the feature type of the center object in a lexical order. Figure 2 illustrates the method to materialize neighbor relationships of a spatial dataset. The neighborhood areas of objects A.1, A.3, and B.4 are represented by dotted circles whose radii are a user specific neighbor distance. The black solid lines in each circle represent a star neighbor relationship with the center object. A.1 has two neighboring objects, B.1 and C.1. The star neighborhood of A.1 is $\{A.1, B.1, C.1\}$ including the center object A.1. In the case of A.3, three neighboring objects are present, A.4, B.3 and C.1. However, A.4 is not included in the star neighborhood set of A.3 since we focus on co-location relationships among different feature types. Next consider the neighborhood of B.4. B.4 has two neighbor objects, A.2 and C.2. However, A.2 is not included in the star neighborhood set of B.4 since the neighbor relationship between A.2 and B.4 is already reflected in the star neighborhood set of A.2. A set of all star neighborhoods of the spatial dataset is listed in Figure 2.

Definition 2 Let $I = \{o_1, \dots, o_k\} \subseteq S$ be a set of spatial objects whose feature types $\{f_1, \dots, f_k\}$ are different. If all objects in I are neighbors to the first object o_1 , I is called a **star instance** of co-location $C = \{f_1, \dots, f_k\}$.

In Figure 2, a subset of the A.1 star neighborhood including A.1, $\{A.1, B.1, C.1\}$ is a star instance of $\{A, B, C\}$.

3.2 Join-less Co-location Mining Algorithm

The join-less co-location mining algorithm has three phases. The first phase converts an input spatial dataset into a set of disjoint star neighborhoods. The second phase gathers the star instances of candidate co-locations from the star neighborhood set, and coarsely filters candidate co-locations by the prevalence value of the star instances. The third phase filters co-location instances from the star instances, and finds prevalent co-locations and generates co-location rules. Figure 3 illustrates a join-less algorithm trace. Algorithm 1 shows the pseudo code.

PHASE I	Level 1	PHASE II	Level 2	Level 3
Feature A star neighborhoods	A	A B A C	A B C	A B C
A.1, B.1, C.1	A.1	A.1, B.1 A.1, C.1	A.1, B.1, C.1	A.1, B.1, C.1
A.2, B.4, C.2	A.2	A.2, B.4 A.2, C.2	A.2, B.4, C.2	A.2, B.4, C.2
A.3, B.3, C.1	A.3	A.3, B.3 A.3, C.1	A.3, B.3, C.1	A.3, B.3, C.1
A.4, C.1	A.4	A.4, C.1	A.4, C.1	
	4			3/4 2/3
Feature B star neighborhoods	B	B C	B C	
B.1	B.1		B.3, C.1	
B.2	B.2		B.3, C.3	
B.3, C.1, C.3	B.3		B.4, C.2	
B.4, C.2	B.4			
B.5	B.5			
	5			
Feature C star neighborhoods	C			
C.1	C.1			
C.2	C.2			
C.3	C.3			
	3			
PHASE III		A B A C B C	A B C	
		A.1, B.1 A.1, C.1 B.3, C.1	A.1, B.1, C.1	
		A.2, B.4 A.2, C.2 B.3, C.3	A.2, B.4, C.2	
		A.3, B.3 A.3, C.1 B.4, C.2	A.3, B.3, C.1	
		3/4 2/3 4/4 2/3	2/4 2/3	
				clique instances
				participation index

Figure 3. Join-less algorithm trace

Convert a spatial dataset to a set of disjoint star neighborhoods (Step 1): Given an input dataset and a neighbor relationship, first find all neighboring object pairs using a geometric method such as plane sweep [4], or a spatial query method using quaternary tree or R-tree [8]. The star neighborhoods are generated by grouping the neighboring objects per each object. Figure 3 shows the star neighborhoods sorted by the feature type of the center objects.

Generate candidate co-locations (Step 4): First, we initialize all features to size 1 prevalent co-locations by the definition of the participation index measure. The number of instances per each feature can be known during the scan of the input spatial dataset for materializing the neighbor relationships. Size k ($k > 1$) candidate co-locations are generated from prevalent size $k - 1$ co-locations. Here, we have a feature level filtering of co-locations. If any subset of a candidate co-location is not prevalent, the candidate co-location is pruned.

Filter the star instances of candidate co-locations from the star neighborhood set (Step 6): The star instances of a candidate co-location are gathered from the star neighborhoods whose center object feature type is the same as the first feature of the candidate co-location. For example, the instances of a candidate co-location $\{B, C\}$ are gathered from the feature B star neighborhoods, and the instances of $\{A, B, C\}$ are gathered from the feature A star neighborhoods. Notice that the number of candidate co-locations examined in each star neighborhood is much smaller than the number of actual candidate co-locations.

Select coarse prevalent co-locations using their star instances (Step 9): The size 2 star instances are clique instances since our neighbor relationship is symmetric (step 8). Thus, we go to step 12 to find prevalent co-locations. For size 3 or more, we need to check if the star instance is

Algorithm 1 Join-less co-location mining algorithm

Inputs

$F = \{f_1, \dots, f_n\}$: a set of spatial feature types
 S : a spatial dataset, R : a neighbor relationship
 min_prev , min_cond_prob

Output

A set of all prevalent co-location rules with participation index $\geq min_prev$ and conditional probability $\geq min_cond_prob$

Variables

$T = \{T_{f_1}, \dots, T_{f_n}\}$: a set of star neighborhoods
 C_k : a set of size k candidate co-locations
 SI_k : star instances of size k candi co-locations
 CI_k : clique instances of size k candi co-locations
 P_k : a set of size k prevalent co-locations
 R_k : a set of size k co-location rules

Method

```
1)  $TD = gen\_star\_neighborhoods(F, S, R)$ ;  
2)  $P_1 = F$ ;  $k = 2$ ;  
3) while (not empty  $P_{k-1}$ ) do  
4)    $C_k = gen\_candidate\_co\_locations(P_{k-1})$ ;  
5)   for  $t \in T$  do  
6)      $SI_k = filter\_star\_instances(C_k, t)$ ;  
7)   end do  
8)   if  $k = 2$  then  $CI_k = SI_k$   
9)   else do  $C_k = select\_coarse\_prev\_co\_location$   
           ( $C_k, SI_k, min\_prev$ )  
10)       $CI_k = filter\_clique\_instances(C_k, SI_k)$ ;  
11)   end do  
12)    $P_k = select\_prev\_co\_location(C_k, CI_k, min\_prev)$ ;  
13)    $R_k = gen\_co\_location\_rules(P_k, min\_cond\_prob)$ ;  
14)    $k = k + 1$ ;  
15) end do  
16) return  $\bigcup(R_2, \dots, R_k)$ ;
```

a clique instance. Before this procedure, we have a coarse filtering step of co-locations. We filter the candidate co-locations using the participation index from their star instances. For example, in Figure 3, the participation index of candidate co-location $\{A, B, C\}$ from the star instances is $\frac{3}{5}$. If it is less than a user specified minimum prevalent threshold, the candidate co-location $\{A, B, C\}$ is pruned without examining exact co-location instances.

Filter co-location instances(Step 10): From the star instances of a candidate co-location, we filter its co-location instances by looking up all the instances of the co-location of features except the first feature of the candidate co-location. For example, to check the cliqueness of a star instance $\{A.1, B.1, C.1\}$ of co-location $\{A, B, C\}$, we examine if a subinstance $\{B.1, C.1\}$ except A.1 is in the set of clique instances of co-location $\{B, C\}$. This instance look up operation can be performed efficiently by an instance key which is composed of the ids of objects in the instance. As shown in Figure 3, $\{A.1, B.1, C.1\}$ is not a co-location instance, but $\{A.2, B.4, C.2\}$ and $\{A.3, B.3, C.1\}$ are co-location instances.

Select prevalent co-location patterns(Step 12): The refinement filtering of co-locations is done by the participation index values calculated from their co-location instances. Prevalent co-locations satisfying the minimum prevalence threshold are selected.

Generate co-location rules(Step 13): All co-location rules satisfying a given minimum conditional probability

are generated from a set of prevalent co-locations. Steps 3-15 are repeated as the size of co-location patterns increases.

4 Analytical Analysis

We analyze our join-less co-location mining algorithm for completeness and correctness. Completeness means the join-less algorithm finds all co-location rules whose participation index and conditional probability satisfy a user specified minimum prevalence threshold and conditional probability threshold. Correctness means that all co-location rules generated by the join-less algorithm have a participation index and a conditional probability above a user-specified minimum prevalence threshold and conditional probability. First we provide related lemmas.

Lemma 2 *The star partition model does not miss any neighbor relationship of an input spatial data.*

Proof The disjoint star partition model includes all neighbor relations of each object and excludes only duplicate neighbor relations which are already included in a star neighborhood by Definition 1.

Lemma 3 *Let $C = \{f_1, \dots, f_k\}$ be a size k co-location and SI be a set of star instances of C . The participation index of C from SI is not less than the true participation index of C .*

Proof The participation ratio of f_1 from SI is the maximum possible probability that the objects of feature f_1 of C have clique relationships with the objects of the other features f_2, \dots, f_k in C since only objects of feature f_1 in the star instances can be included in a clique co-location instance of C . The participation ratio of f_j ($1 < j \leq k$) from SI is also the maximum possible probability that the objects of feature f_j have clique relationships with the objects of features f_1 in C since our neighbor relationship is symmetric. Thus the participation index of C calculated from the star instances is not less than the true participation index of C , $\min_{f_i \in C} \{possible\ max\ Pr(C, f_i)\} \geq \min_{f_i \in C} \{Pr(C, f_i)\}$.

Lemma 4 *Let an instance $I = \{o_1, \dots, o_k\}$ be a star instance of a co-location $C = \{f_1, \dots, f_k\}$. If the subinstance $\{o_2, \dots, o_k\}$ except o_1 is a clique, the instance I is a clique.*

Proof In a star instance $I = \{o_1, \dots, o_k\}$, the first object o_1 has neighbor relationships to the other objects, o_2, \dots, o_k by Definition 2. Object o_j ($2 \leq j \leq k$) has a neighbor relationship to o_1 since the neighbor relationship is symmetric and also has neighbor relationships to all the other objects o_h where $2 \leq h \leq k$ and $h \neq j$ since $\{o_2, \dots, o_k\}$ is a clique. Thus each object o_i ($1 \leq i \leq k$) has neighbor relationships to all other objects in I . I is a clique.

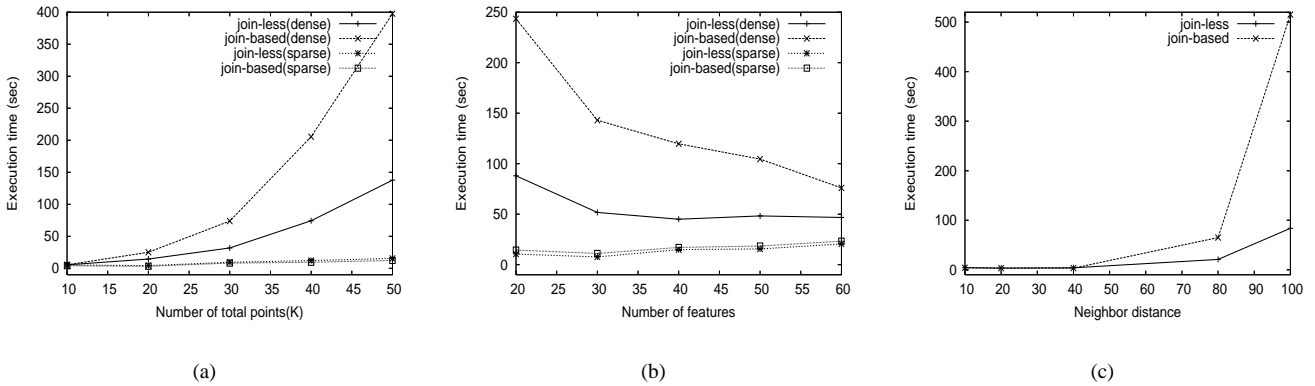


Figure 4. Scalability of the join-less algorithm: (a) by number of points, (b) by number of features, (c) by neighbor distance

Theorem 1 *The join-less co-location mining algorithm is complete.*

Proof The completeness of the join-less algorithm can be shown by the following two parts. The first is that the method to materialize the neighbor relationships of an input spatial data (step 1), the method to gather star instances(step 6), and the method to filter clique instances(step 10) are correct. The star partition model does not miss any neighbor relationship of a input spatial dataset by Lemma 2. The star instances of co-locations gathered from the star neighborhoods whose center object feature type is the same as the first feature of the co-location, have correct star neighbor relationships. Any potential co-location instance is not missed since the star instances are a super set of the clique instances. The method to filter co-location instances from the star instances does not drop a true clique instance by Lemma 4. Next, we show that the filtering steps of co-locations do not drop true co-locations. The feature level filtering by prevalent subsets(step 4) is complete by Lemma 1. The coarse filtering of co-locations(step 10) does not eliminate any true prevalent co-locations by Lemma 3. The refinement filtering(step 12) prunes only co-locations whose true participation index is less than the threshold. Step 13 ensures that no co-location rules satisfying a user specific conditional probability are missed.

Theorem 2 *The join-less co-location mining algorithm is correct.*

Proof The correctness of the join-less algorithm can be guaranteed by steps 12 and 13. Step 12 selects only co-locations whose participation indexes satisfy a user specific prevalence threshold. The generated co-location rules by step 13 also satisfy a user specific conditional probability.

5 Experimental Evaluation

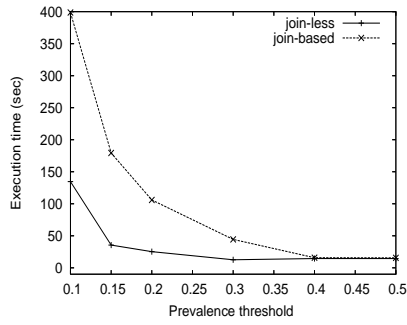
We evaluate the join-less co-location algorithm with the join-based co-location algorithm [9] using synthetic and real datasets. Synthetic datasets were generated using a spatial data generator similar to the data generator used in [9]. The number of features is 20, the average size of co-locations is 5, the neighbor distance is 10 and the prevalence threshold is 0.3. All the experiments were performed on a Sun SunBlade 1500 with 1.0 GB main memory and 177MHz CPU.

5.1 Evaluation with Synthetic Datasets

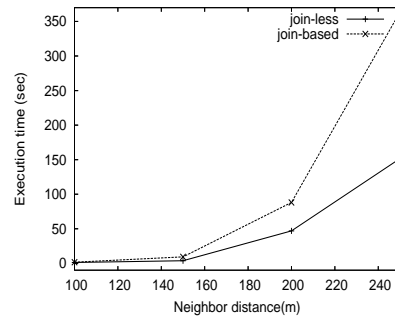
We examined the scalability of the join-less algorithm in the number of point objects, the number of feature types and distance neighbor threshold.

1) *Effect of the number of point objects:* First, we compared the effect of the number of points. We used two different spatial frames, $10,000 \times 10,000$ and $1,000 \times 1,000$. In the first frame, even if the number of points is increased from 10K to 50K, the two algorithms showed similar execution time since the datasets are still sparse. In the second frame, with the increase of number of points, the join-based algorithm execution time is dramatically increased due to the increase of data density. As shown in Figure 4 (a), the join-less algorithm shows scalability to large dense datasets.

2) *Effect of the number of features:* In the second experiment, we compared the performance of the algorithms as a function of the number of features. We also used two different dense datasets of 15K points. Figure 4 (b) shows the results. In the sparse dataset, the algorithms show similar execution time even if the number of features increases. In the dense dataset, overall execution time is decreased with the increase of features. The reason is that under the same num-



(a)



(b)

Figure 5. Real datasets: (a) A climate dataset, (b) A chimpanzee behavior dataset

ber of points, the increase of features causes the number of points per each feature to be decreased, which in turn may lead to a decrease in the number of co-location instances. Overall the join-less algorithm shows better performance.

3) *Effect of neighbor distance:* The third experiment examined the effect of different neighbor distances 10, 20, 40, 80 and 100. As shown in Figure 4 (c), the join-less algorithm shows less increase in the execution time with the increase of distance threshold. The join-based algorithm shows a rapid increase since the neighbor distance increase makes the neighborhood areas larger and increases the number of co-location instances.

5.2 Evaluation with Real Datasets

We used two different types of real world datasets. One was an Earth dataset relating climate to vegetation growth from [1]. Another dataset was an Ecology animal behavior dataset. It contained female chimpanzee behavior observation data from 1999 to 2001 from [2].

1) *Earth climate data:* The earth climate dataset includes monthly measurements of variables such as global plant growth, e.g., Net Primary Production(NPP), and climate variables, e.g., precipitation(PREC) on latitude-longitude spherical grids. For example, (NPP-Hi, PREC-Low) is one of the co-location patterns discovered, where Hi(Low) denotes an unusually high(low) value of the measurements. The total number of event features was 18. The total number of feature instances was 15,515. We used 4 as a neighborhood distance which means 4 cells (each grid cell is 1 degree \times 1 degree). Figure 5 (a) presents the execution time of the three algorithms as a function of the prevalence threshold. The join-less method shows much better performance at the lower threshold values. The performance difference between the partial join method and the join-based

method is relatively small because the cut relation ratio was almost 0.8.

2) *Ecology animal behavior data:* The animal behavior dataset has 24 chimpanzee features. We assigned a unique instance id to different location points per chimpanzee id. The total number of point instances was 698. Figure 5 (b) presents the execution time of the algorithms by different neighbor distances. The execution time of the join-less method increases more slowly than the other methods with the increase of distance and shows better performance.

6 Conclusion and Future Work

In this paper, we propose a join-less co-location mining algorithm with a neighborhood materialization. The algorithm is efficient since it does not require expensive spatial joins or instance joins for identifying co-location instances. The experimental evaluation shows the join-less method outperforms the join-based method and is scalable to dense datasets. As future work, we plan to explore methods to answer temporal questions such as how a co-location changes over time as well as methods to identify objects showing similar moving patterns.

References

- [1] Discovery of patterns in the global climate system using data mining. <http://www.ahpcrc.umn.edu/nasa-umn/>.
- [2] Nsf-sei project: Spatio-temporal data analysis for ecology behavior. <http://www.cs.umn.edu/research/chimps/>.
- [3] R. Agarwal and R. Srikant. Fast algorithms for Mining association rules. In *Proc. of Int'l Conference on Very Large Databases(VLDB)*, 1994.
- [4] M. Berg, M. Kreveld, O. M., and O. Schwarzkopf. *Computational Geometry*. Springer, 2000.

- [5] K. Koperski and J. Han. Discovery of Spatial Association Rules in Geographic Information Databases. In *Proc. of Int'l Symposium on Large Spatial Data bases, Maine*. 47-66, 1995.
- [6] Y. Morimoto. Mining Frequent Neighboring Class Sets in Spatial Databases. In *Proc. ACM SIGKDD Int'l Conference on Knowledge Discovery and Data Mining*, 2001.
- [7] J. Roddick and M. Spiliopoulou. A Bibliography of Temporal, Spatial and Spatio-Temporal Data Mining Research. In *Proc. SIGKDD Explorations 1(1)*: 34-38, 1999.
- [8] S. Shekhar and S. Chawla. *Spatial Databases: A Tour*. Prentice Hall, 2003.
- [9] S. Shekhar and Y. Huang. Co-location Rules Mining: A Summary of Results. In *Proc. of Int'l Symposium on Spatio and Temporal Database(SSTD)*, 2001.
- [10] S. Shekhar, P. Zhang, Y. Huang, and R. Vatsavai. *Trends in Spatial Data Mining, as a book chapter in Data Mining: Next Generation Challenges and Future Directions*. H. Kargupta, A. Joshi, K. Sivakumar and Y. Yesha (editors), AAAI/MIT Press, 2004.
- [11] J. Yoo and S. Shekhar. A Partial Join Approach for Mining Co-location Patterns. In *Proc. of ACM Int'l Symposium on Advances in Geographic Information Systems(ACM-GIS)*, 2004.