

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. The experimental evaluations show the join-less algorithm performs more efficiently than a current join-based algorithm and is scalable in dense spatial datasets.

1 Introduction

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

*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.

Co-location pattern 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 [1, 3] for co-location pattern mining since there are no pre-defined transactions in many spatial datasets. Thus, a current co-location mining algorithm [6] 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. The experimental evaluations show the join-less co-location mining algorithm outperforms the join-based algorithm and is scalable in dense spatial datasets.

Related Work: The problem of mining association rules based on spatial relationships was first discussed in [3]. The work discovers 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. [4] uses space partitioning for identifying neighboring objects for a frequent neighboring feature set. However, the distinct space partitioning approach may miss co-location instances across partition areas and generate incorrect results. [6] proposes an instance join-based co-location mining algorithm similar to *a priori-gen* [1]. The join-based approach is computationally expensive with the increase of co-location patterns

and their instances. [7] proposes a partial join approach to reduce the number of expensive spatial join operations in finding co-location instances. However, the performance depends on the distribution of the spatial dataset.

The remainder of the paper is organized as follows. Section 2 presents our join-less approach for co-location pattern mining. Section 3 presents the experimental evaluation. The conclusion and future work are discussed in Section 4.

2 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 the basic concepts of co-location pattern mining. Then, we describe our method to materialize spatial neighbor relationships, and present the join-less co-location algorithm.

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., $R(A.1, B.1) \Leftrightarrow (distance(A.1, B.1) \leq d)$. 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. The interest of a co-location pattern can be measured by its prevalence and conditional probability [6]. 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., $Pr(C_1|C_2) = \frac{Number\ of\ distinct\ instances\ of\ C_1\ in\ instances\ of\ C_1 \cup C_2}{Number\ of\ instances\ of\ C_1}$.

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., $Pr(C, f_i) = \frac{Number\ of\ distinct\ objects\ of\ f_i\ in\ instances\ of\ C}{Number\ of\ objects\ of\ f_i}$. The **participation index** $Pi(C)$ of a co-location $C = \{f_1, \dots, f_k\}$ is defined as $Pi(C) = \min_{f_i \in C} \{Pr(C, f_i)\}$.

2.2 Neighborhood Materialization

The ideal neighborhood materialization for co-location mining is to find all maximal clique relationships from an

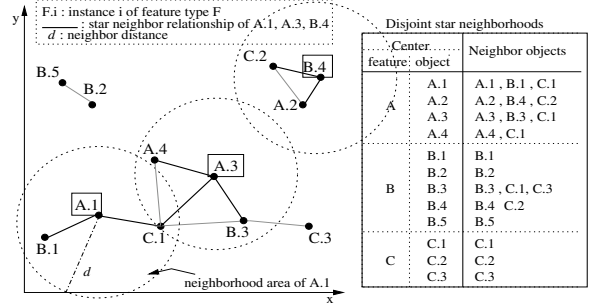


Figure 1. Neighborhood materialization

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.

The star neighborhood of an object is a set of the center object and objects in its neighborhood whose feature types are greater than the feature type of the center object in a lexical order. Figure 1 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 1.

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

PHASE I	Level 1	PHASE II		Level 2	Level 3		
Feature A star neighborhoods	A	A B	A 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		star instances
A.2, B.4, C.2	A.2	A.2, B.4	A.2, 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.4, C.1	A.4		A.4, C.1				
	4				3/4	2/3	
Feature B star neighborhoods	B			B C			participation index star instances.
B.1	B.1			B.3, C.1			if < threshold, it's pruned
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						single instance count
C.1	C.1						
C.2	C.2						
C.3	C.3						
	3						Check if it is a clique
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		clique instances
		A.3, B.3	A.3, C.1	B.4, C.2	A.3, B.3, C.1		
			A.4, C.1				
		3/4	2/3	2/5	2/3		participation index

Figure 2. Join-less algorithm trace

2.3 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 2 illustrates a join-less algorithm trace. Algorithm 1 shows the pseudo code.

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 [2], or a spatial query method using quaternary tree or R-tree [5]. The star neighborhoods are generated by grouping the neighboring objects per each object. Figure 2 shows the star neighborhoods sorted by the feature type of the center objects.

Generate candidate co-locations(Step 4): 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 5): 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

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) $TD = \text{gen_star_neighborhoods}(F, S, R)$;
- 2) $P_1 = F$; $k = 2$;
- 3) while (not empty P_{k-1}) do
- 4) $C_k = \text{gen_candidate_co_locations}(P_{k-1})$;
- 5) for $t \in T$ do
- 6) $SI_k = \text{filter_star_instances}(C_k, t)$;
- 7) end do
- 8) if $k = 2$ then $CI_k = SI_k$
- 9) else do $C_k = \text{select_coarse_prev_co_location}(C_k, SI_k, min_prev)$
- 10) $CI_k = \text{filter_clique_instances}(C_k, SI_k)$;
- 11) end do
- 12) $P_k = \text{select_prev_co_location}(C_k, CI_k, min_prev)$;
- 13) $R_k = \text{gen_co_location_rules}(P_k, min_cond_prob)$;
- 14) $k = k + 1$;
- 15) end do
- 16) return $\bigcup(R_2, \dots, R_k)$;

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. For size 3 or more, we need to check if the star instance is a clique instance. Before this procedure, we have a coarse filtering step of co-locations. If the participation index calculated from the star instances of a candidate co-location is less than a prevalent threshold, the candidate co-location 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.

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-

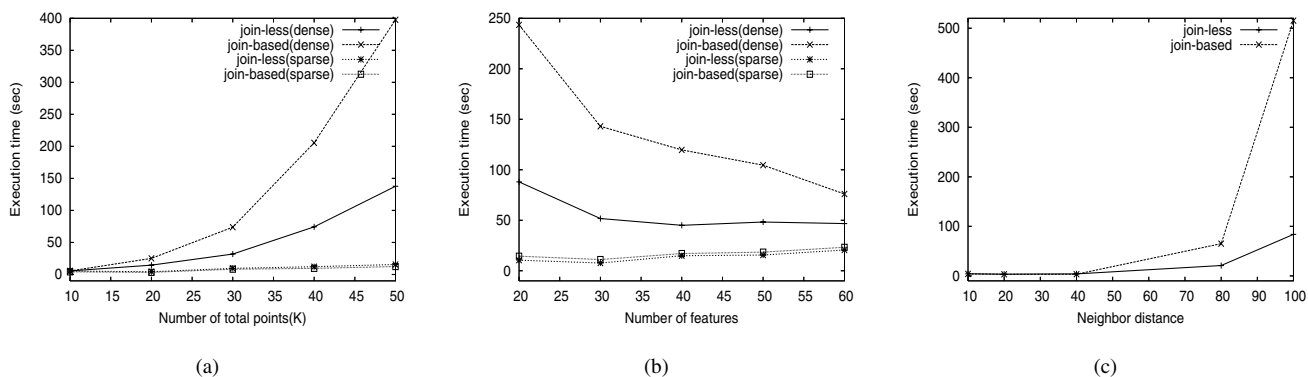


Figure 3. Efficiency of the join-less co-location mining algorithm

15 are repeated as the size of co-location patterns increases.

3 Experimental Evaluation

We evaluate the join-less co-location algorithm with the join-based co-location algorithm [6] using synthetic datasets. Synthetic datasets were generated using a spatial data generator similar to the data generator used in [6]. 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 Sun-Blade 1500 with 1.0 GB main memory and 177MHz CPU.

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 3 (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 3 (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 number 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 3 (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.

4 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] R. Agarwal and R. Srikant. Fast algorithms for Mining association rules. In *Proc. of Int'l Conference on Very Large Databases(VLDB)*, 1994.
- [2] M. Berg, M. Kreveld, O. M., and O. Schwarzkopf. *Computational Geometry*. Springer, 2000.
- [3] 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.
- [4] Y. Morimoto. Mining Frequent Neighboring Class Sets in Spatial Databases. In *Proc. ACM SIGKDD Int'l Conference on Knowledge Discovery and Data Mining*, 2001.
- [5] S. Shekhar and S. Chawla. *Spatial Databases: A Tour*. Prentice Hall, 2003.
- [6] 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.
- [7] 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.