Mining Confident Co-location Rules without A Support Threshold

Yan Huang, Hui Xiong, Shashi Shekhar [*]
University of Minnesota at Twin-Cities
{huangyan,huix,shekhar}@cs.umn.edu

Jian Pei
State University of New York at Buffalo
jianpei@cse.buffalo.edu

ABSTRACT

Mining co-location patterns from spatial databases may reveal types of spatial features likely located as neighbors in space. In this paper, we address the problem of *mining confident co-location rules without a support threshold.* First, we propose a novel measure called the *maximal participation index.* We show that every confident co-location rule corresponds to a co-location pattern with a high maximal participation index value. Second, we show that the maximal participation index is non-monotonic, and thus the conventional Apriori-like pruning does not work directly. We identify an interesting weak monotonic property for the index and develop efficient algorithms to mine confident co-location rules. An extensive performance study shows that our method is both effective and efficient for large spatial databases.

Keywords

spatial data mining, confident co-location rules

1. INTRODUCTION

Spatial data mining becomes more interesting and important as more spatial data have been accumulated in spatial databases [9, 11, 12, 4, 6, 7]. Spatial patterns are of great values in many applications. For example, in mobile computing, to provide location-sensitive promotions, it is demanding to find services requested frequently and located together from mobile devices such as PDAs.

Mining *spatial co-location patterns* [10, 8, 3] is an important spatial data mining task with broad applications. To illustrate the idea of spatial co-location patterns, let us consider the events in Figure 1. In the figure, there are various types of spatial objects denoted by different symbols. As can be seen, objects of $\{‘+’, ‘\times’\}$ and $\{‘o’, ‘*’\}$ tends to be located together, respectively.

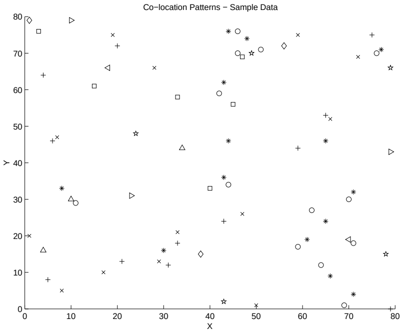

Figure 1: Spatial Co-location Patterns Illustration

In [10], efficient algorithms have been proposed to mine spatial co-location patterns from spatial databases. A set of spatial features form a pattern if, for each spatial feature, at least $s\%$ objects having the feature are neighbours to some objects having other features in the pattern. However this method biases towards popular spatial features when both frequent and rare features are involved. Many important rules involve both frequent and rare features where the above mentioned approach could fail. For example, in a case settled in 1996, *PG&E*'s nearby plant was leaching chromium 6, a rust inhibitor, into Hinkley's water supply, and the suit blamed the chemical for dozens of symptoms, ranging from nosebleeds to breast cancer, Hodgkin's disease, miscarriages and spinal deterioration. The prosecutors proved that "*chromium 6 contaminated water → nosebleeds, breast cancer, . . . , in their nearby region with high probability*". This is a typical confident co-location rule involving both frequent and rare events because although nosebleeds are quite common and chromium 6 contaminated water is rare the later factor implies the former one strongly.

Unfortunately, since the rule in the above example is *confident* but not *popular*, they cannot be found using any previous methods, such as the one in [10, 8]. We need to explore new methods to solve the problem. In general, the challenges of mining confident spatial co-location rules lay in two aspects. First, *how to identify and measure confident spatial co-location rules?* Rare events are often ignored when they are happening together with popular events. Many measures are based on *frequency* such that rare events are unfavorable. Second, *how to mine the patterns efficiently?* Even though we have a good measure for confident patterns, it is still challenging to find all the patterns efficiently. One dominant

[*] This work was supported was partially supported by NASA grant No. NCC 2 1231 and 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 he policy of the government, and no official endorsement should be inferred

ACM SAC Melbourne, Florida, USA
Copyright 2003 ACM 0-58113-624-2/03/03 ...$5.00.

obstacle is that *the maximal participation index is not monotonic w.r.t. co-location pattern containment relation.* Thus, the conventional apriori-like [1] pruning technique cannot be applied.

Our contributions. In this paper, we propose a novel measure called *maximal participation index*, which incorporates confident spatial co-location patterns regardless of the freqencies of the events involved. We show that finding confident spatial co-location rules can be achieved by finding confident co-location patterns w.r.t. the maximal participation index. We propose two algorithms. The first algorithm is a rudimentary extension of the Apriori-like [1] solutions. Our second method explores an interesting *weak monotonic property* of the maximal participation index, and uses the property to push the maximal participation index threshold deep into the mining. It achieves good performance in most cases. The experimental results show that our methods are effective and efficient for mining large spatial databases.

The remainder of the paper is organized as follows. In Section 1.1, we review related work. Section 2 presents an overview of a prevalent co-location pattern mining framework [10] and its limitations. In Section 3, we extend the framework in Section 2 to handle confident patterns, and show that the problem of mining confident co-location rules can be solved by mining confident co-location patterns. Efficient algorithms for the mining are developed in Section 4. An extensive performance study is reported in Section 5. We conclude the paper in Section 6.

1.1 Related Work

We categorize the related work of mining co-location patterns into spatial statistics approaches and combinatorial approaches.

In spatial statistics, dedicated techniques such as cross k-functions with Monte Carlo simulations [3] have been developed to test the co-location of two spatial features. However, the Monte Carlo simulation could be expensive. Another approach is to arbitrarily partition the space into a lattice. For each cell of the lattice, count the number of instances of each spatial feature. Pairwise correlation of spatial features could be found by tests such as χ^2 [3]. Arbitrary partitioning may loss neighboring instances across borders of cells.

In market basket data sets, finding highly correlated items without support pruning [2] is a close analogy to our problem, but without a spatial component. The related algorithms mostly rely on sampling and hashing. Thus, the results may not be complete.

The spatial co-location pattern mining framework presented in [10, 8] biased on popular events. It may miss some highly confident but "infrequent" co-location rules by using only "*support*"-based pruning.

To the best of our knowledge, mining confident co-location rules without support threshold in the spatial context has not been investigated systematically yet.

2. PREVALENT CO-LOCATION RULES IN SPATIAL DATABASES

In a spatial database S, let $F = \{f_1, \ldots, f_k\}$ be a set of *boolean spatial features*. Let $I = \{i_1, \ldots, i_n\}$ be a set of n instances in the spatial database S, where each instance is a vector $\langle$instance-id, location, spatial features$\rangle$. The spatial feature f of instance i is denoted as $i.f$. We assume that a neighborhood relation R over pairwise locations in S exists.

The objective of the co-location rule mining is to find rules in the form of $A \rightarrow B$, where A and B are subsets of spatial features. For example, a rule "{*traffic jam, police*} → *car accident* (80%)" means that, when there are a traffic jam and policemen, there is a probability of 80% that a car accident is in a nearby region. Here, 80% is the conditional probability of the rule.

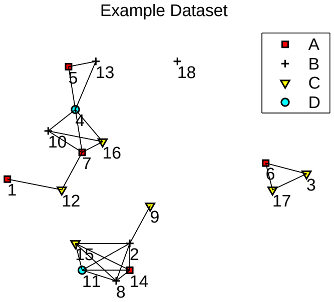

Figure 2: Example Data set

To capture the concept of "nearby", the concept of user-specified neighbor-sets was introduced. A *neighbor-set* L is a set of instances such that all pairwise locations in L are neighbors. For example, in Figure 2, neighborhood relation R is defined based on Euclidean distance and neighboring instances are linked by edges. $\{3, 17\}$, $\{5, 13\}$, and $\{7, 10, 16, 4\}$ are all neighbor-sets. A *co-location pattern* (or pattern in short) C is a set of spatial features, i.e., $C \subseteq F$. A neighbor-set L is said to be a *row instance* of co-location pattern C if every feature in C appears in an instance of L, and there exists no proper subset of L does so. We denote all row instances of a co-location pattern C as $rowset(C)$. In other words, $rowset(C)$ is the set of neighbor-sets where spatial features in C co-locate.

For a co-location rule $R: \ A \rightarrow B$, the *conditional probability* of R is defined as

$$\frac{|\{L \in rowset(A) | \exists L' \text{ s.t. } (L \subseteq L') \wedge (L' \in rowset(A \cup B))\}|}{|rowset(A)|}$$

In words, the conditional probability is the probability that a neighbor-set in $rowset(A)$ is a part of a neighbor-set in $rowset(A \cup B)$. Intuitively, the conditional probability p indicates that, whenever we observe the occurrences of the spatial features in A, the probability to find the occurrence of B in a nearby region is p.

Example 1. *To find the conditional probability of* $\{A, B\} \rightarrow \{C\}$*, we firsts identify* $\{A, B\}$*'s rowset, i.e.,* $\{\{5, 13\}, \{7, 10\}, \{14, 2\}, \{14, 8\}\}$*. Please note that* $\{7, 10, 4\}$ *is not a row instance of* $\{A, B\}$*, since it has a proper subset* $\{7, 10\}$ *does so. Then we identify* $\{A, B, C\}$*'s rowset, i.e.,* $\{\{7, 10, 16\}, \{14, 2, 5\}\}$*. Among all of the 4 row instances of* $\{A, B\}$*, two of them, i.e.* $\{7, 10\}$ *and* $\{14, 2\}$*, are subsets of some neighbor-sets in the rowset of* $\{A, B, C\}$*. So the condition probability of* $\{A, B\} \rightarrow \{C\}$ *is* $\frac{2}{4} = 50\%$*.*

Given a spatial database S, to measure the implication strength of a spatial feature in a co-location pattern, a *participation ratio* $pr(C, f)$ can be defined as

$$pr(C, f) = \frac{|\{r | (r \in S) \wedge (r \text{ is in a row instance of } C)\}|}{\{r | (r \in S) \wedge (r.f = f)\}|}$$

In words, a feature f has a partition ratio $pr(C, f)$ in pattern C means wherever the feature f is observed, with probability $pr(C, f)$, all other features in C are also observed in a neighbor-set.

In spatial application domain, there exist no natural transactions. Thus in [10], a *participation index* is proposed to measure the implication strength of a pattern from spatial features in the pattern. For a co-location pattern C, the participation index $PI(C) = \min_{f \in C}\{pr(C, f)\}$. In words, wherever a feature in C is observed, with a probability of at least $PI(C)$, all other features in C can be observed in a neighbor-set. A high participation index value indicates that the spatial features in a co-location pattern likely show up together.

Given a user-specified *prevalent threshold* min_prev, a co-location pattern is called *prevalent* if $PI(C) \geq min\_prev$. Interestingly, the prevalent measure here plays a role similar to the "*support*" measure in the mining frequent patterns from tradition databases.

As shown below, both the participation ratio and the participation index are monotonic w.r.t. the size of co-location patterns. (Because of lack of space, please refer to [5] for proofs of all lemmas)

Lemma 1. *Let C and C' be two co-location patterns such that $C \subset C'$. Then, for each feature $f \in C \cap C'$, $pr(C, f) \geq pr(C', f)$. Furthermore, $PI(C) \geq PI(C')$.*

It is interesting to note that, in the above prevalent co-location pattern mining framework, some confident co-location rules with rare events may be unfortunately missed.

Example 2. *Let us consider co-location pattern $C = \{$chromium 6 contaminated water, nosebleeds,miscarriages$\}$. Suppose $pr(C,$ chromium 6 contaminated water $) = 85\%$, $pr(C,$ nosebleed$) = 0.1\%$ and $pr(C,$ miscarriages$) = 1\%$. Then, $PI(C) = \min\{85\%, 0.1\%, 1\%\} = 0.1\%$. As can be seen, even though chromium 6 contaminated water has strong implication to nosebleed and miscarriages, unfortunately, the whole co-location pattern is weak in the term of participation index.*

Can we extend the framework to mine such confident patterns and rules even though their participation index values are low? In other words, *can we mine confident rule without "*support*"-based pruning?"* That is the topic of the next two sections.

3. MAXIMAL PARTICIPATION INDEX

There is one important observation about confident co-location patterns: "*even though the participation index of the whole pattern could be low, there must be some spatial feature(s) with high participation ratio(s)*". In the pattern of Example 2, $P = \{$chromium 6 contaminated water, nosebleeds, miscarriages$\}$, the participation *index* is low, since chromium 6 contaminated water sources are rare. However, the participation *ratio* of "*chromium 6 contaminated water*" in the pattern is pretty high.

The above observation motivates our extension of the participation index framework. For a co-location pattern C, we define the *maximal participation index* of a spatial feature f as $maxPI = \max_{f \in C}\{pr(C, f)\}$. In words, a high maximal participation index value indicates that there are some spatial features strongly imply the pattern.

Using confident co-location patterns, we can generate confident co-location rules. In general, given a confident pattern $C = \{f_1, \ldots, f_k\}$. We sort all spatial features in C in the participation ratio descending order. Without loss of generality, suppose that, for $(1 \leq i \leq l \leq k)$, $pr(C, f_i) \geq min\_conf$. Then, we can generate a rule $R : \ f_1 \cdots f_l \rightarrow f_{l+1} \cdots f_k$. The rule carries the information that *if a spatial feature f_i $(1 \leq i \leq l)$ is observed in some location, then the probability of observing all other spatial features in $C - \{f_i\}$ in a neighbor-set is at least $maxPI(C)$.*

Given a *confidence threshold min_conf*, the problem of **mining confident co-location patterns in a spatial database** is *to find the complete set of co-location patterns C such that $maxPI(C) \geq min\_conf$.*

While the extension of participation index to maximal participation index is intuitive, there is no easy way to extend the existing level-by-level Apriori-like [1] algorithm to mine confident patterns w.r.t. a maximal participation index threshold. The dominant obstacle is that *maximal participation index is not monotonic w.r.t. the pattern containment relation*, as shown in the following example.

Example 3. *In Figure 2, the set of spatial features $\{B, C\} \subset \{A, B, C\}$. However, $maxPI(\{B, C\}) = \max\{\frac{3}{5}, \frac{3}{6}\} = 60\% \leq maxPI(\{B, C, D\}) = \max\{\frac{2}{5}, \frac{2}{6}, \frac{2}{2}\} = 100\%$! (Please refer [5] for rowsets and maxPIs)*

Now, the challenge becomes *how we can push the confidence threshold to prune the search space*. That is the topic of the next section.

4. ALGORITHMS

In this section, we will develop efficient algorithms for mining confident co-location patterns from spatial databases. We propose two methods. The first method is a rudimentary extension of the Apriori [1] algorithm. The second method is based on an interesting weak monotonic property of the maximal participation index.

4.1 A Rudimentary Algorithm

In many applications, very rare events could be just noise. Thus, we may have a minimal prevalent threshold min_prev and a confidence threshold min_conf such that we only want to find patterns P with $PI(P) \geq min\_prev$ and $maxPI(P) \geq min\_conf$. Based on this observation, we develop an Apriori-like algorithm called Min-Max algorithm as follows. We use the minimal prevalent threshold min_prev to do Apriori-like pruning, then filter out patterns failed the maximal participation index threshold by a postprocessing. Limited by space, the details are omitted here. Please refer [5] for details. One advantage of the Min-Max algorithm is that the user can specify the prevalence of patterns she wants to see by the min_prev value. The major disadvantage of the algorithm is that, if the user want to find the complete answer, the algorithm has to generate a huge number of candidates and test them, even though the confidence threshold min_prev is high.

4.2 Pruning By A Weak Monotonic Property

Is there any property of the maximal participation index we can use to get efficient algorithms for mining confident rules? Let us re-examine Example 2. Pattern $P = \{$chromium 6 contaminated water, nosebleeds,miscarriages$\}$ has three proper subsets such that each subset has exactly 2 features. Feature "*chromium 6 contaminated water*" has a high participation ratio, and it participates in two out of the three subsets. Since the participation ratio is monotonic (Lemma 1), the maximal participation index values of the two proper subsets containing "*chromium 6 contaminated water*" must be higher or equal to that of P. In other words, at most one 2-subpattern of P can have a lower maximal participation index value.

The above observation can be generalized to a pattern with l features and we have the following *weak monotonic* property.

Lemma 2 (Weak monotonicity). *Let P be a k-co-location pattern. Then, there exists at most one $(k-1)$-subpattern P' such that $P' \subset P$ and $maxPI(P') < maxPI(P)$.*

Based on the weak monotonic property if a k-pattern is confident, at least $(k-1)$ out of its k subpatterns with $(k-1)$ features are confident. Therefore, we can revise the candidate generation process, such that a k-pattern having at most one non-confident $(k-1)$-subpattern should be generated. The idea is illustrated in the following example.

Example 4. *Suppose the maximal participation index values of $\{A, B, C\}$, $\{A, C, D\}$ and $\{B, C, D\}$ are all over the confidence threshold, but that of $\{A, B, D\}$ is not. We still should generate a candidate $P = \{A, B, C, D\}$, since it is possible that $maxPI(P)$ passes the threshold.*

To achieve this, we need a systematic way to generate the candidates. Please note that, in apriori, for the above example, $\{A, B, C, D\}$ is generated only if $\{A, B, C\}$ and $\{A, B, D\}$ (differ only in their last spatial feature) are both frequent. However, in the confident pattern mining, it is possible that $\{A, B, D\}$ is not confident, while $\{A, B, C, D\}$ is still confident. The candidate generation here is tricky.

In general, two co-location patterns from the confident k-pattern set P_k, i.e. $P \in P_k$ and $P' \in P_k$, can be joined and generate a candidate $(k+1)$-pattern in C_{k+1} if and only if P and P' have one different feature in the last two features. For example, even though $\{A, B, D\}$ is not confident, candidate $\{A, B, C, D\}$ can be generated by $\{A, B, C\}$ and $\{A, C, D\}$.

We will illustrate the correctness of the above candidate generation method in Lemma 3 and Example 5.

With the revised candidate generator, the mining algorithm is presented as follows.

Algorithm maxPrune

Input: A spatial database S, a neighbourhood relation R, a confidence threshold min_conf.

Output: Co-location patterns P such that $maxPI(P) \geq min\_conf$.

Method:

1. let $k = 2$; generate C_2, the set of candidates of confident 2-patterns and their rowsets, by geometric methods;
2. For each $C \in C_k$ calculate $maxPI(C)$ from C's rowset $rowset(C)$; Let P_k be the subset of C_k such that for each $P \in P_k$, $maxPI(P) \geq min\_conf$;
3. generate C_{k+1}, the set of candidates of confident $(k+1)$-patterns, as illustrated in Example 4 ; if $C_{k+1} \neq \emptyset$, let $k = k + 1$, go to Step 2;
4. output $\cup_i P_i$

The algorithm does not need minimal prevalence threshold and finds all confident co-location patterns.

To make sure the candidate generation does not miss any confident co-location, we need to prove candidate $(k+1)$-patterns C_{k+1} generated by the maxPrune algorithm is a superset of the actual confident $(k+1)$-patterns P_{k+1}. This is proved in the following lemma.

Lemma 3. *Let P be a confident k-pattern $(k \geq 2)$. Then, there exist two $(k-1)$ patterns P_1 and P_2 such that (1) $P_1 \subset P$, $P_2 \subset P$, (2) P_1 and P_2 share either the k^{th} or the $(k-1)^{th}$ feature in P, but not both, and (3) both P_1 and P_2 are confident.*

Example 5. *Suppose $min\_conf = 0.85$. Initially all singleton co-location patterns are confident since they have $maxPI = 1$. A general geometric method is used to generate candidate confident 2-patterns and their rowsets. From their rowsets, we calculate their $maxPI$. Only $P_2 = \{\{A, C\}, \{A, D\}, \{B, C\}, \{C, D\}\}$ are confident.*

Then, we generate candidates 3-patterns. In detail, $\{A, C\}$ joins $\{A, D\}$, $\{A, C\}$ joins $\{B, C\}$, $\{A, C\}$ joins $\{C, D\}$, $\{A, D\}$ joins $\{C, D\}$, and $\{B, C\}$ joins $\{C, D\}$ to generate candidate 3-patterns $\{\{A, B, C\}$, $\{A, B, D\}$, $\{A, C, D\}$, and $\{B, C, D\}\}$. The rowsets of the candidates are generated by joining the rowsets of the two 2-patterns leading to the candidate.

We go back to step 2. From their rowsets, we calculate the maximal participation index values for the candidates of 3-patterns. We get $P_3 = \{\{A, B, D\}, \{A, C, D\} \{B, C, D\}\}$, which are the confident patterns. Then, we generate the candidates of confident 4-patterns. In detail, $\{A, B, D\}$ joins $\{A, C, D\}$ according to Lemma 3. We thus generate candidate 4-pattern $\{A, B, C, D\}$, as illustrated in Example 4. Rowsets of $\{A, C, D\}$ and $\{B, C, D\}$ are joined to produce the rowset of $\{A, B, C, D\}$. Its maxPI is calculated and it is confident. The algorithm proceeds similarly. It can be verified that $C_5 = \emptyset$ and thus the algorithm stops.

Compared to the Min-Max algorithm, the maxPrune algorithm does not need any minimum prevalence threshold and finds the complete set of confident co-locations with any prevalence. In the process of mining the complete set of highly confident co-locations, the maxPrune algorithm generates much less candidate co-location patterns compared to that of min-max with $prev\_min = 0$, and thus lowers down the costs of expensive rowset generation and test dramatically.

5. PERFORMANCE EVALUATION

To evaluate the performance of the two algorithms, Min-Max and maxPrune, we conducted an extensive performance study on synthetic datasets. All the experiments were performed on a Pentium III 550MHz PC machine with $4G$ megabytes main memory, running Redhat 6.1. All methods were implemented using C++. The experimental results are consistent. Limited by space, we report only the results on some representative datasets.

We made up a data synthetic generator. Our Data generator is similar to the one in [1], with some proper extensions to produce spatial datasets. The major parameters of the data generator are demonstrated as follows. For synthetic dataset $I100k.C10.R50$, we generate $100k$ instances (denoted as $I100k$). There are up to 50 co-location patterns with very high confidence but very low prevalence (denoted as $R50$). We achieved this by binding a spatial feature to a pre-generated potential co-location pattern, and making those bound spatial features not prevalent. The number of attributes in a co-location pattern yields to a Possion distribution, while the mean is 10 (denoted as $C10$). For all the datasets, the total number of spatial attributes is 100 and the total number of pre-generated potential co-location pattern is 500.

We first evaluated the performance of the two algorithms on mining a dataset that has no co-location patterns which are with high confidence and low prevalence. Dataset $I100kC5R0$ is used. Clearly, this condition favors algorithm Min-Max. We varied the confidence threshold from 2.5% to 4%. Figure 3 shows the run time of the algorithm maxPrune and the algorithm Min-Max with respect to different min_conf thresholds.

For the Min-Max algorithm, we chose the min_prev values as 0%,0.05%,0.5%, and 2.5%, respectively. Only when the min_prev value was set to 0%, the algorithm Min-Max finds all the confident co-location patterns. Because algorithm

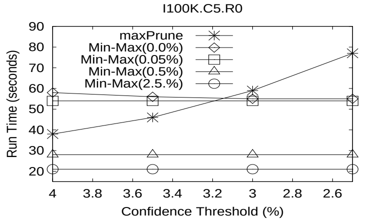

Figure 3: Run Time without High Conf. and Low Prev. Co-locations

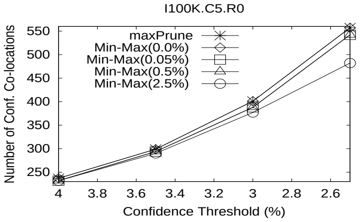

Figure 4: Conf. Co-locations Found without High Conf. and Low Prev. Co-locations

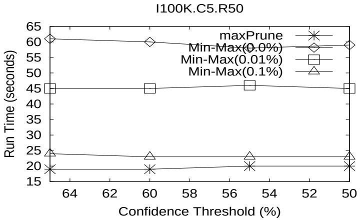

Figure 5: Run Time with High Conf. and Low Prev. Co-locations

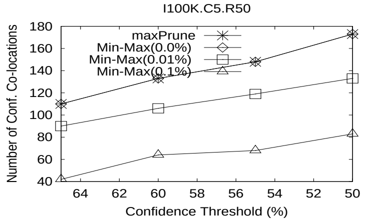

Figure 6: Conf. Co-locations Found with High Conf. and Low Prev. Co-locations

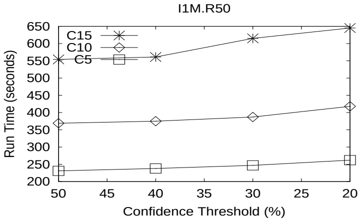

Figure 7: Scalability of the maxPrune Algorithm w.r.t. Conf. Threshold

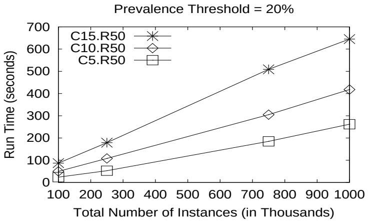

Figure 8: Scalability of the maxPrune Algorithm w.r.t. Number of Instances

Min-Max has to generate all co-location patterns with participation index value over *min_prev*, the run time of Min-Max is not sensitive to the change of confidence threshold. To the contrast, the run time of maxPrune increases as the *min_prev* threshold decreases.

When $min\_prev > 0$, algorithm Min-Max may miss some patterns. When the two algorithms generate the same set of patterns, i.e., $min\_prev = 0$ for Min-Max, Min-Max outperforms maxPrune only when the confidence threshold is lower than 3.2%. In such an extreme region, most co-location patterns are confident, and algorithm maxPrune has a heavier overload on the candidate generation than Min-Max has. In all other situations, maxPrune wins. Figure 4 shows the number of confident co-location patterns found w.r.t. confident thresholds. As can be seen, using a small *min_prev* threshold, Min-Max can generate results close to maxPrune does.

Next, we compared the performance of the two algorithms on a dataset(I100K.C5.R50) where there were many confident co-locations with low prevalence. In this experiment, we treid to find confident co-locations with high $maxPI$ thresholds, say above 50%. As shown in Figure 5 and Figure 6, maxPrune outperforms Min-Max in a wide margin. To obtain a comparable runtime performance, Min-Max has to adopt a high *min_prev* threshold, and miss almost half of the confident co-location patterns.

Then we evaluated the scalability of the maxPrune algorithm w.r.t. confidence thresholds. Limited by space, we only plotted the runtime of maxPrune on dataset I1M.C15.R50 in Figure 7 as an example. The results on other datasets are consistent. As can be seen, the algorithm is scalable w.r.t. confidence thresholds. On the other hand, the more spatial features on average in a pattern, the longer the run time.

Finally, we evaluated the scalability of maxPune algorithm w.r.t. total number of instances. We fixed the confidence thresholds as 20%. The run time of maxPrune is linearly scalable to the database size as show in Figure 8.

6. CONCLUSIONS

In this paper, we identified the limitation of the conventional co-location pattern mining and proposed a novel approach to mining confident co-location patterns without "support"-based pruning. We developed efficient algorithms for the mining. An extensive performance study shows that our method is both effective and efficient. The maxPrune algorithm is scalable in mining large spatial databases.

This study opens two interesting directions for future explorations. First, as an initial study, in this paper, we considered only boolean spatial features. 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. Second, it would be interesting to mine temporal-spatial co-location patterns for moving objects.

7. REFERENCES

[1] R. Agarwal and R. Srikant. Fast algorithms for Mining association rules. In *VLDB'94*.

[2] E. Cohen, M. Datar, S. Fujiwara, A. Gionis, P. Indyk, R. Motwani, J. D. Ullman, and C. Yang. Finding interesting associations without support pruning. *Knowledge and Data Engineering*, 13(1):64–78, 2001.

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

[4] M. Ester, H.-P. Kriegel, and X. Xu. Knowledge discovery in large spatial databases: focusing techniques for efficient class identification. In *SSD'95*.

[5] Y. Huang, H. Xiong, S. Shekhar, and J. Pei. Mining confident co-location rules without a support threshold: A summary of results. In *University of Minnesota Technical Report*, 2002.

[6] E. M. Knorr and R. T. Ng. Extraction of spatial proximity patterns by concept generalization. In *KDD'96*.

[7] K. Koperski and J. Han. Discovery of Spatial Association Rules in Geographic Information Databases. In *SSD'95*.

[8] Y. Morimoto. Mining Frequent Neighboring Class Sets in Spatial Databases. In *KDD'01*.

[9] R. T. Ng and J. Han. Efficient and effective clustering methods for spatial data mining. In *VLDB'94*.

[10] S. Shekhar and Y. Huang. Co-location Rules Mining: A Summary of Results. In *SSTD'01*.

[11] S. Shekhar, C. Lu, and P. Zhang. Detecting Graph-based Spatial Outliers: Algorithms and Applications. *KDD'01*.

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