Efficiently Mining High Utility Co-location Patterns from Spatial Data Sets with Instance-Specific Utilities

Lizhen Wang, Wanguo Jiang, Hongmei Chen(✉), and Yuan Fang

Department of Computer Science and Engineering, School of Information Science and Engineering, Yunnan University, Kunming 650091, Yunnan, China
hmchen@ynu.edu.cn

Abstract. Traditional spatial co-location pattern mining attempts to find the subsets of spatial features whose instances are frequently located together in some regions. Most previous studies take the prevalence of co-locations as the interestingness measure. However, it is more meaningful to take the utility value of each instance into account in spatial co-location pattern mining in some cases. In this paper, we present a new interestingness measure for mining high utility co-location patterns from spatial data sets with instance-specific utilities. In the new interestingness measure, we take the intra-utility and inter-utility into consideration to capture the global influence of each feature in co-locations. We present a basic algorithm for mining high utility co-locations. In order to reduce high computational cost, some pruning strategies are given to improve the efficiency. The experiments on synthetic and real-world data sets show that the proposed method is effective and the pruning strategies are efficient.

Keywords: Spatial co-location patterns · High utility co-location patterns · Intra-utility · Inter-utility · Pruning

1 Introduction

In recent years, spatial data are rapidly generated and the size of spatial data sets is getting huger and huger. For example, NASA Earth Observing System has been generating more than 1 TB of spatial data per day. With the popularity of mobile devices, spatial data with location would increase faster and faster. The vast amounts of spatial data contain potential and valuable information which can help us make important decisions. There are a lot of researches on spatial data mining, including spatial association rule analysis, spatial clustering, spatial classification, and so on.

In spatial data, if the distance between two spatial instances is no more than a given distance threshold, the two instances satisfy the *neighbor relationship*. Traditional spatial co-location pattern mining aims at finding the subsets of spatial features whose instances are frequently located in neighborhoods. A *row instance* of a co-location *c* represents a subset of instances, which includes an instance of each feature in *c* and forms a clique under the neighbor relationship. All row instances of a co-location *c* make up its *table instance* denoted as *T*(*c*). Similar to the support in Association

Rules Mining (ARM), *Participation Index* (PI) is used to evaluate the prevalence of co-locations. The *PI* of a co-location c is defined as $PI(c) = \min_{f_i \in c}\{PR(c,f_i)\}$, where $PR(c,f_i)$ is the *Participation Ratio* (PR) of feature f_i in a co-location c, that is $PR(c,f_i) = \frac{|\pi_{f_i}T(c)|}{\text{Number of instances of } f_i}$, where π is the relational projection operation with duplication elimination. Participation ratio is used to evaluate the prevalence of features, and participation index is used to measure the prevalence of co-locations, which is the interestingness measure in traditional co-location mining.

Mining co-locations is very significant in the real world. For example, botanists have found that there are orchids in 80% of the area where the middle-wetness green-broad-leaf forest grows. A mobile service provider may be interested in mobile service patterns frequently requested by geographical neighboring users. Other applications include Earth science, public health, biology, transportation, etc.

In most previous studies, the importance of all features and instances are treated similarly. However, there exist some difference between features and even instances belonging to the same feature. For instance, the economic value of the rosewood is much greater than that of the ordinary pine. What's more, the value of rosewoods with different sizes is also different. So, only checking the prevalence of co-locations might be insufficient for identifying real interesting patterns. Traditional co-location mining can't find some low frequency but high interesting patterns [19], and some prevalent patterns which just reflect the common sense may be worthless to users.

Here, we use an example to illustrate the problem. Figure 1 shows the locations of instances of six kinds of plants (features), and each instance is denoted by the plant type and a numeric id, e.g. A.1, and edges among instances indicate neighboring relationships. The superscript of each instance represents its utility value, which can be considered as its price. Table 1 gives the total utility value of each kind of plant which is the sum of utility value of all instances belonging to the plant type.

In Fig. 1, according to traditional co-location mining, for the co-location {A, B, C}, *PI*({A, B, C}) = 1/4. And if the prevalence threshold is 0.3, {A, B, C} would be

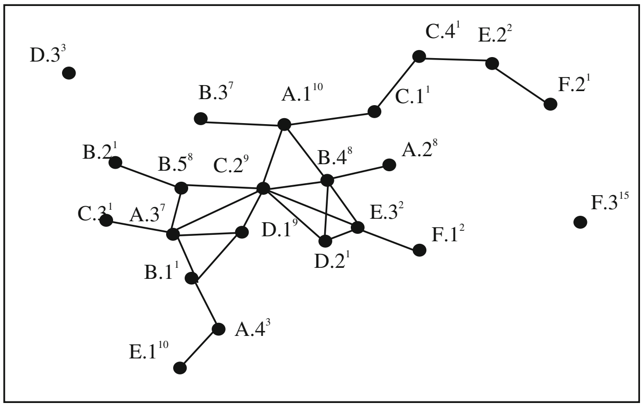

Fig. 1. An example spatial data set

Table 1. Total utility value of each plant in Fig. 1

Features	Instances	Total utility values
A	A.1^{10}, A.2^{8}, A.3^{7}, A.4^{3}	28
B	B.1^{1}, B.2^{1}, B.3^{7}, B.4^{8}, B.5^{8}	25
C	C.1^{1}, C.2^{9}, C.3^{1}, C.4^{1}	12
D	D.1^{9}, D.2^{1}, D.3^{3}	13
E	E.1^{10}, E.2^{2}, E.3^{2}	14
F	F.1^{2}, F.2^{1}, F.3^{15}	18

regarded as a non-interesting co-location. However, according to $T(\{A, B, C\}) = \{\{A.1^{10}, B.4^{8}, C.2^{9}\}, \{A.3^{7}, B.5^{8}, C.2^{9}\}\}$, the utility value of feature A's instances in {A, B, C} is 17, which accounts for 17/28 of total utility value of A. Similarly, the proportion of B is 16/25 and C is 9/12. So the utility of each feature in {A, B, C} account for a large proportion of its total utility. {A, B, C} may be interesting. However, as to the pattern {E, F}, $PI(\{E, F\}) = 2/3$, $T(\{E, F\}) = \{\{E.2^{2}, F.2^{1}\}, \{E.3^{2}, F.1^{2}\}\}$. But the proportion of E is 4/14 and F is 3/18. So the utility of each feature in {E, F} is less than 30%. {E, F} may be non-interesting.

Therefore, the traditional measure may not find interesting co-locations because the utilities of features and instances are ignored. In this paper, we focus on high utility co-location mining from spatial data sets with instance-specific utilities.

1.1 Related Work

The problem of mining spatial association rules was first discussed in [1]. The participation index for prevalent co-location mining and join-based algorithm was presented in [2, 3]. Then a lot of existing works about co-location mining are based on the participation index which satisfies the downward closure property. Join-less algorithm was introduced in [4], using a novel model to materialize spatial neighbor relationships and an instance-lookup scheme to reduce the computational cost of identifying table instances. An efficient algorithm based on iCPI-Tree was proposed in [8]. In order to mine the co-locations with rare features, a new prevalence measure called the maximal participation ratio was proposed in [9]. A new general class of interestingness measures based on the spatial distribution of co-locations and information entropy was proposed in [5]. Probabilistic prevalent co-location mining was introduced in [6] to find co-locations in the context of uncertain data. [7] studied co-location rule mining on interval data and defined new related concepts based on the semantic proximity neighborhood. An optimal candidate generation method was proposed in [17]. Complex spatial co-location mining which can deal with complex spatial relationships was introduced in [18].

The research on high utility mining was first discussed in ARM [10]. The utility of each item consists of *internal utility* and *external utility*. The internal utility represents the quantity of items in transactions and the external utility is the unit profit values of items. But the utility of itemsets doesn't satisfy the downward closure property which can improve the mining efficiency, and a two-phase algorithm for fast mining high utility itemsets was proposed in [11]. [12] introduced a novel framework to mine the interesting high utility pattern with a strong frequency affinity. An incremental mining algorithm for efficiently mining high utility itemsets was proposed to handle the intermittent data environment in [13]. UP-Growth proposed in [14] enhances the mining performance through maintaining the information of high utility itemsets by UP-tree. A novel algorithm named GUIDE and a special data structure named TMUI-tree were proposed for mining temporal maximal utility itemsets from data stream environment in [15]. [16] introduced an efficient algorithm named USpan to mine high utility sequences from large scale data with very low minimum utility.

There are more and more studies on co-location mining and high utility itemsets mining, but there were rare literatures about high utility co-location mining [19, 20]. Similar to ARM [10], [19] divided the utility of features in a co-location into *external utility* and *internal utility*. The external utility represents the unit profile and the internal utility represents the quantity of different instances of features in a table instance. The utility of a feature in a co-location is equal to the product of external and internal utilities. And a framework for mining high utility co-locations was proposed in [19]. By following the definitions in [19], [20] discussed a problem of updating high utility co-locations on evolving spatial databases.

In some real-world data, the utilities of features are different from each other and even instances belonging to the same feature may have an obvious difference in utilities. Furthermore, in some cases, the data set can't map into the model of external and internal utility. Considering the complexity of real-world data, there exist two major challenges in high utility co-location mining from spatial data sets with instance-specific utilities. One is how to define the interestingness measure reasonably to judge high utility co-locations, and another is how to mine high utility co-locations efficiently. In this paper, we try to tackle these challenges.

1.2 Our Contributions

Different from previous researches, we make the following contributions in this paper:

First, we take the instances with utilities as study objects, and the importance of features and instances is treated differently.

Second, we propose a new interestingness measure to identify high utility co-locations in spatial data sets with instance-specific utilities.

Third, we present a basic algorithm to mine high utility co-locations. In order to reduce the computational cost, some pruning strategies are given.

Finally, the extensive experiments on synthetic and real-world data sets verify that the proposed method is effective and efficient.

The remainder of the paper is organized as follows: Sect. 2 gives the related concepts for mining high utility co-locations from spatial data sets with instance-specific utilities, and a basic algorithm is presented in Sect. 3. In Sect. 4, the pruning strategies are detailed. Experimental results and evaluation are shown in Sect. 5. The conclusion and future work are discussed in Sect. 6.

2 Related Concepts

In the real world, the importance of each instance may be different. Thus, we take the instances with utilities as study objects and the utilities reflect their importance. The related concepts for mining high utility co-locations are given in this section, and Table 2 summarizes notations frequently used throughout the paper.

Table 2. Summary of notations

Notation	Definition	Notation	Definition
F	Set of spatial features	$u(f_i)$	Utility of feature f_i
f_i	i-th spatial feature	$u(f_i, c)$	Utility of feature f_i in co-location c
S	Set of features' instances	$IntraUR(f_i, c)$	Intra-utility ratio of f_i in c
$f_i \cdot j^v$	j-th instance with utility v of f_i	$InterUR(f_i, c)$	Inter-utility ratio of f_i in c
c	A co-location pattern	$UPR(f_i, c)$	Utility participation ratio of f_i in c
k	Size of c	$UPI(c)$	Utility participation index of c
R	A spatial neighbor relationship	w_1	Weighted value of *IntraUR* in computing *UPR*
$T(c)$	Table instance of c	w_2	Weighted value of *InterUR* in computing *UPR*
$u(f_i \cdot j)$	Utility of instance $f_i \cdot j$	M	A UPI threshold

Definition 1 (spatial instance with utility value). *Given a set of spatial features F and a set of their instances S. Let spatial instance $f_i \cdot j^v \in S$ be the j-th instance of feature $f_i \in F$. The utility value of $f_i \cdot j^v$ is expressed by the superscript v. We denote the utility of spatial instance $f_i \cdot j^v$ as $u(f_i \cdot j) = v$.*

According to Definition 1, every instance may have distinct utility, even if they belong to the same feature. For example, the feature A represents the rosewood. A.1^{1000} is a 100-year-old rosewood and worth \$1000, i.e., u(A.1) = 1000. A.2^{25} is a 10-year-old rosewood, which is worth \$25, and u(A.2) = 25.

The ***total utility*** of a feature $f_i \in F$ is the sum of utilities of its instances, denoted as $u(f_i) = \sum_{j=1}^{m} u(f_i \cdot j)$, where m is the number of instances belonging to f_i. For example, the total utility of feature A in Fig. 1 is u(A) = u(A.1) + u(A.2) + u(A.3) + u(A.4) = 10 + 8+ 7 + 3 = 28.

Definition 2 (utility of feature in co-location). *Given a size k co-location c = $\{f_1, f_2, \ldots, f_k\}$, we further define the sum of utilities of instances belonging to feature $f_i \in c$ in table instance T(c) as the utility of f_i in c, denoted as $u(f_i, c) = \sum_{f_i \cdot j \in \pi_{f_i}(T(c))} u(f_i \cdot j)$, where π is the relational projection operation with duplication elimination.*

For example, for c = {A, B, C} in Fig. 1, $T(c)$ = {{A.1^{10}, B.4^{8}, C.2^{9}}, {A.3^{7}, B.5^{8}, C.2^{9}}}. The utility of A in c is u(A, c) = u(A.1) + u(A.3) = 10 + 7 = 17.

Definition 3 (intra-utility ratio). *Given a size k co-location c* = {f_1, f_2, ..., f_k}, *the intra-utility ratio of feature f_i in co-location c is defined as the proportion of f_i's utility in c to its total utility, i.e.,* $IntraUR(f_i, c) = \frac{u(f_i, c)}{u(f_i)}$.

IntraUR(f_i, *c*) indicates the direct utility of feature f_i in co-location *c*, which can be regarded as its direct influence on *c*.

For example, for *c* = {A, B, C} in Fig. 1, *T*(*c*) = {{$A.1^{10}$, $B.4^8$, $C.2^9$}, {$A.3^7$, $B.5^8$, $C.2^9$}}. The intra-utility ratio of each feature in *c* is calculated as

$$IntraUR(\mathrm{A}, c) = \frac{u(\mathrm{A.1}) + u(\mathrm{A.3})}{u(\mathrm{A})} = 17/28,\ IntraUR(\mathrm{B}, c) = \frac{u(\mathrm{B.2}) + u(\mathrm{B.5})}{u(\mathrm{B})} = 16/25,$$
$$IntraUR(\mathrm{C}, c) = \frac{u(\mathrm{C.2})}{u(\mathrm{C})} = 9/12.$$

Definition 4 (inter-utility ratio). *Given a size k co-location c* = {f_1, f_2, ..., f_k}, *the inter-utility ratio of feature f_i in co-location c is defined as* $InterUR(f_i, c) = \frac{\sum_{f_j \in c, j \neq i} u(f_j, c)}{\sum_{f_j \in c, j \neq i} u(f_j)}$.

The inter-utility ratio is regard as the influence of feature f_i on other features in co-location *c*, which is an indirect influence of f_i on *c*. In a co-location *c* = {$f_1, f_2, \ldots, f_k$}, the change of feature $f_i \in c$ probably impact on the utility of other features in *c*. For example, there are various services in Location-based Service. In the package service, the sales of service A might promote the sales of service B. So, we use the inter-utility ratio to indicate the effect of a feature on other features in a co-location. In Fig. 1, the effect of feature A in co-location {A, B, C} on other features B and C is compute as

$$InterUR(\mathrm{A}, c) = \frac{u(\mathrm{B}, c) + u(\mathrm{C}, c)}{u(\mathrm{B}) + u(\mathrm{C})} = 25/37.$$

We divide the influence of feature f_i into two parts to evaluate a co-location *c* comprehensively and reasonably. One is the influence of its utility in *c* denoted as *IntraUR*(f_i, *c*), and another is the indirect influence of f_i on *c* denoted as *InterUR*(f_i, *c*).

Definition 5 (Utility Participation Ratio, UPR). *Given a size k co-location c* = {f_1, f_2, ..., f_k}, *the weighted sum of IntraUR(f_i, c) and InterUR(f_i, c) is defined as the utility participation ratio of feature f_i in co-location c, which is denoted as UPR(f_i, c)* = $w_1 \times$ *IntraUR*(f_i, *c*) + $w_2 \times$ *InterUR*(f_i, *c*), *where* $0 \leq w_1, w_2 \leq 1$ *and* $w_1 + w_2 = 1$, w_1 *represents the weighted value of IntraUR(f_i, c) and* w_2 *represents that of Inter UR(f_i, c).*

The w_1 and w_2 in Definition 5 can be used to adjust the effect of *IntraUR* and *InterUR*, which are assigned the specified values by users in application. For example, in sales volume promotion of supermarkets, if we are more care the promoted sale volume of different goods, $w_1 \leq w_2$ may be reasonable. Usually, w_1 and w_2 satisfy $w_1 \geq w_2$.

For example, in Fig. 1, if we suppose w_1 = 0.7 and w_2 = 0.3, then the UPR of each feature in c = {A, B, C} is computed as

$$\begin{aligned} UPR(\mathrm{A}, c) &= 0.7 \times IntraUR(\mathrm{A}, c) + 0.3 \times InterUR(\mathrm{A}, c) \\ &= 0.7 \times (17/28) + 0.3 \times (25/37) = 0.628. \\ UPR(\mathrm{B}, c) &= 0.7 \times IntraUR(\mathrm{B}, c) + 0.3 \times InterUR(\mathrm{B}, c) \\ &= 0.7 \times (16/25) + 0.3 \times (26/40) = 0.643. \\ UPR(\mathrm{C}, c) &= 0.7 \times IntraUR(\mathrm{C}, c) + 0.3 \times InterUR(\mathrm{C}, c) \\ &= 0.7 \times (9/12) + 0.3 \times (33/53) = 0.711. \end{aligned}$$

Definition 6 (Utility Participation Index, UPI). *Given a size k co-location c* = {f_1, f_2, ..., f_k}, *We define the minimum utility participation ratio among all features in co-location c as the utility participation index of c, i.e.,* $UPI(c) = min\{UPR(f_i,\ c), f_i \in c\}$.

A co-location pattern c is a ***high utility co-location pattern*** if and only if $UPI(c) \geq M$ holds, where M is a UPI threshold given by users.

The UPI measure extends the traditional PI measure only based on prevalence. If the utilities of instances and the influence between features in a co-location are ignored, UPI is equal to the traditional PI.

The prevalent patterns may not be high utility patterns and the high utility patterns may not be prevalent as well, which can be proved by patterns {E, F} and {A, B, C} in Fig. 1. If w_1 = w_2 = 0.5 and M = 0.3, *UPI*({E, F}) = 0.226 and *PI*({E, F}) = 0.667, while *UPI*({A, B, C}) = 0.628 and *PI*({A, B, C}) = 0.25. Because of full consideration into the difference of each instance, our interestingness measure is more reasonable. However, different from the traditional interestingness measure, UPI does not satisfy the downward closure property which is a very efficient pruning strategy for mining prevalent co-locations. Therefore, finding all high utility patterns directly is time-consuming. For example, for c = {A, D} in Fig. 1, $T(c)$ = {{A.3^7, D.1^9}}. Given w_1 = w_2 = 0.5, we can get *UPI*({A, D}) = 0.471. But the super pattern $c' = \{\mathrm{A},\ \mathrm{C},\ \mathrm{D}\}$ of $c, T(c') = \{\{\mathrm{A}.3^7,\ \mathrm{C}.2^9,\ \mathrm{D}.1^9\}\}$, and *UPI*({A, C, D}) = 0.485. So, we have the inequality $UPI(c') > UPI(c)$.

3 A Basic Algorithm

In this section, we present a basic algorithm for mining the high utility co-locations defined in Sect. 2. The basic algorithm has three phases. The first one is to materialize the spatial neighbor relationships. The spatial data set is converted into the star neighborhood partition model in [4]. The second one is to generate candidate co-locations and compute their table instances. The third one is to compute the UPI of each candidate co-location and find high utility co-locations. The second and third phases are repeated with the increment of co-locations' size. Algorithm 1 shows the pseudocode of the basic algorithm.

Algorithm 1. Basic Algorithm

Input: $F=\{f_1,f_2,\ldots,f_n\}$: a set of spatial features; S: a set of spatial instances; R: a spatial neighbor relationship; w_1: the weighted value of *IntraUR*; w_2: the weighted value of *InterUR*; M: the UPI threshold

Output: a set of all co-location patterns with UPI $\geq M$

Steps:

```
1) $SN = gen\_star\_neighborhoods(F, S, R)$;
2) $H_1 = F$; $k$=2;    //$H_k$: a set of size $k$ high utility co-locations
3) While (not empty $H_{k-1}$ and $NonH_{k-1}$) do
                    //$NonH_k$: a set of size $k$ non-high utility co-locations
4)     $C_k=gen\_candi\_colocations(H_{k-1}, NonH_{k-1})$;
           //$C_k$: a set of size $k$ candidate co-locations
5)     $CTI_k=gen\_table\text{-}instances(C_k, SN)$;
           //$CTI_k$: a set of table instances of size $k$ candidates
6)     $Compute\_UPI(C_k, w_1, w_2)$;
7)     $H_k=select\_utility\_colocations(C_k, CTI_k, M)$;
8)     $NonH_k=C_k - H_k$;
9)     $k=k + 1$;
10) End do
11) Return $\cup \{H_1, H_2,\cdots, H_{k-1}\}$
```

Initialization (Step 1–2): Given a spatial data set and a spatial neighbor relationship, find all neighboring instance pairs using a geometric method such as mesh generation or plane sweep [4]. The star neighborhoods can be generated from the neighbor instance pairs by lexicographical order [4]. After generating the star neighborhood set (*SN*), we initialize all size 1 co-locations with utility participation index 1.0, which means all size 1 co-locations are high utility co-locations. Then, we add all size 1 co-locations into H_1.

Generating Candidate Co-locations (Step 4): A size k ($k \geq 2$) candidate co-locations in C_k is generated from a size $k-1$ co-location c in H_{k-1} or $NonH_{k-1}$ and a new feature f_s which is not included in c and greater than all features of c in lexicographical order, i.e., $C_k = \{c'|c' = c \cup \{f_s\}, \forall c \in H_{k-1} \cup NonH_{k-1}, f_s > \forall f_i \in c\}$.

Specially, the size 2 candidate co-locations in C_2 can be generated from the star neighborhood set directly.

Calculating the UPIs of Candidate Co-locations (Step 5–6): The size 2 co-locations' table instances can be gathered from the star neighborhood set directly. For size k ($k > 2$) co-locations, their table instances need to be extended by size $k-1$ co-locations' table instances. For example, the table instance of co-location {A, B, C} can be generated from the table instance of co-location {A, B}. Then, we can compute the UPI of each candidate co-location according to the Definitions 5 and 6.

Identifying High Utility Co-locations (Step 7–8): We can filter high utility co-locations by the UPIs of candidate co-locations and the given UPI threshold M. Then, high utility co-locations are added into H_k and non-high utility co-locations are added into $NonH_k$.

Steps 3–10 are repeated with the increment of size k.

In Fig. 1, if $w_1 = w_2 = 0.5$ and $M = 0.5$, we can get the high utility co-locations {A, B}, {A, C}, {A, B, C}, {B, C}, {B, D}, {C, D}, and {C, E}. The basic algorithm tests all possible patterns and computes their UPI accurately. So, it is complete and correct, but it is inefficient. In the next section, we would give some pruning strategies to improve the efficiency of the basic algorithm.

4 Pruning Strategies

In this section, we will introduce some pruning strategies to promote the efficiency of the basic algorithm. Traditional co-location mining based on PI can efficiently find all prevalent co-locations due to the downward closure property. But there is no a similar method to find all high utility co-locations due to the non-existence of the downward closure property. Similar to traditional co-location mining, the most time-consuming component in mining high utility co-locations is to generate the table instances of candidate patterns. In order to improve the efficiency of the basic algorithm, we have to early identify some non-high utility candidate co-locations without generating their table instances. The following pruning strategies are used to prune the non-high utility candidate patterns ahead of time.

Lemma 1. *For* $n_1 \geq m_1 > 0$, $n_2 \geq m_2 > 0$, *there exists the following inequality*:

$$\frac{m_1 + m_2}{n_1 + n_2} \leq \max\{\frac{m_1}{n_1}, \frac{m_2}{n_2}\}$$

Proof: Given $n_1 \geq m_1 > 0$, $n_2 \geq m_2 > 0$. If $\frac{m_1}{n_1} \geq \frac{m_2}{n_2}$, then there exists $\frac{m_1+m_2}{n_1+n_2} - \frac{m_1}{n_1} = \frac{m_2 n_1 - m_1 n_2}{n_1(n_1+n_2)} \leq 0$. So, $\frac{m_1+m_2}{n_1+n_2} \leq \frac{m_1}{n_1}$ holds. Similarly, if $\frac{m_2}{n_2} \geq \frac{m_1}{n_1}$, then $\frac{m_1+m_2}{n_1+n_2} \leq \frac{m_2}{n_2}$. Therefore, $\frac{m_1+m_2}{n_1+n_2} \leq \max\{\frac{m_1}{n_1}, \frac{m_2}{n_2}\}$ holds. □

Corollary 1. *For* $k (k > 1)$ *pairs* m_i *and* n_i $(i = 1, 2, ..., k)$, *if* $n_i \geq m_i > 0$, *there exists the following inequality*: $\frac{\sum_{i=1}^{k} m_i}{\sum_{i=1}^{k} n_i} \leq max_{i=1}^{k}\{\frac{m_i}{n_i}\}$.

Definition 7 (non-high utility feature set). *Given a size k co-location* $c = \{f_1, f_2, ..., f_k\}$, *we call the set of all features in co-location c whose UPR is less than the UPI threshold M as the non-high utility feature set of c.*

For example, for c = {A, B, D} in Fig. 1, if $M = 0.4$ and $w_1 = w_2 = 0.5$, then *UPR* (A, c) = 0.257, *UPR*(B, c) = 0.215 and *UPR*(C, c) = 0.422. The non-high utility feature set of c is {A, B}.

Theorem 1. *If* c_1 *and* c_2 *are two non-high utility co-locations, and they have and only have one common feature* f_i *and it is a non-high utility feature, then the pattern* $c = c_1 \cup c_2$ *must be a non-high utility pattern, i.e.,* $c = c_1 \cup c_2$ *can be pruned.*

Proof: Because f_i is a non-high utility feature in c_1 and c_2, we have:

$$UPR(f_i, c_1) = w_1 \frac{u(f_i, c_1)}{u(f_i)} + w_2 \frac{m_1}{n_1} < M \tag{1}$$

where $m_1 = \sum_{f_j \in c_1, j \neq i} u(f_j, c_1)$ and $n_1 = \sum_{f_j \in c_1, j \neq i} u(f_j)$. And

$$UPR(f_i, c_2) = w_1 \frac{u(f_i, c_2)}{u(f_i)} + w_2 \frac{m_2}{n_2} < M \tag{2}$$

where $m_2 = \sum_{f_j \in c_2, j \neq i} u(f_j, c_2)$ and $n_2 = \sum_{f_j \in c_2, j \neq i} u(f_j)$.

For the co-location $c = c_1 \cup c_2$, the UPR of f_i in c satisfies:

$$UPR(f_i, c) = w_1 \frac{u(f_i, c)}{u(f_i)} + w_2 \frac{m_1 + m_2}{n_1 + n_2} \tag{3}$$

due to f_i is the unique common feature in c_1 and c_2.

According to Definition 2 and the concept of table instance, we have $u(f, c) \leq u(f, c')$ if f is the common feature in co-locations c and c', and $c' \subseteq c$.

And according to Lemma 1, $\frac{m_1+m_2}{n_1+n_2} \leq \max\{\frac{m_1}{n_1}, \frac{m_2}{n_2}\}$.

Therefore, we can infer $UPR(f_i, c) < M$ by (1), (2) and (3), which can judge that $c = c_1 \cup c_2$ is a non-high utility co-location. □

For example, for c_1 = {A, B, D} and c_2 = {B, E} in Fig. 1, if $w_1 = w_2 = 0.5$ and $M = 0.5$, $T(c_1) = \{\{A.3^7, B.1^1, D.1^9\}\}$ and $T(c_2) = \{\{B.4^8, E.3^2\}\}$. The UPRs of common feature B in c_1 and c_2 are $UPR(B, c_1) = 0.215 < M$ and $UPR(B, c_2) = 0.231 < M$ respectively, which satisfy the conditions of Theorem 2. So, $c = c_1 \cup c_2$ = {A, B, C, D} must be a non-high utility co-location and can be pruned.

According to the Theorem 1 and Corollary 1, we can infer the Corollary 2.

Corollary 2. *For size* 2 *non-high utility co-locations* $c_1, c_2, \ldots, c_k$ ($k > 1$), *if they have a common non-high utility feature f, then the pattern* $c = c_1 \cup c_2 \cup \ldots \cup c_k$ *must be a non-high utility pattern, i.e., c can be pruned.*

When the spatial data set is sparser or the UPI threshold M is higher, there would be large amounts of size 2 non-high utility co-locations. At that time, we could prune a large number of higher size non-high utility co-locations by combining those size 2 non-high utility co-locations.

In Fig. 1, if $w_1 = w_2 = 0.5$ and $M = 0.5$, there are size 2 non-high utility co-locations {A, E}, {B, E}, {D, E} and {E, F}. And E is a non-high utility feature. The co-locations {A, B, E}, {A, D, E}, {A, E, F}, {B, D, E}, {B, E, F}, {D, E, F}, {A, B, D, E}, {A, B, E, F}, {A, D, E, F}, {B, D, E, F} and {A, B, D, E, F} can be pruned by Corollary 2.

According to Definition 2, for a size k co-location $c = \{f_1, f_2, \ldots, f_k\}$ and $f_i \in c$, $u(f_i, c) \leq u(f_i, c')$ holds, where c' is an arbitrary size $k-1$ sub-pattern of c including f_i. So, we call the minimum of utilities of f_i in size $k-1$ sub-patterns of c including f_i as the upper bound utility of f_i in c, donated as *upbound_u*(f_i, c).

For example, for c = {A, B, C} in Fig. 1, the upper bound utility of feature A in c is *upbound_u(*A, c) = min{u(A, {A, C}), u(A, {A, B})} = min{17, 28} = 17.

Lemma 2. *Given a size k co-location $c = \{f_1, f_2, ..., f_k\}$ and its size k+1 super-pattern $c' = c \cup \{f_{k+1}\}$, the upper bound of $UPI(c')$ is computed as follows*:

$$\min\{w_1 \frac{u(f_i, c)}{u(f_i)} + w_2 \frac{\sum_{f_j \in c, j \neq i} u(f_j, c) + upbound\_u(f_{k+1}, c')}{\sum_{f_j \in c, j \neq i} u(f_j) + u(f_{k+1})}, \ 1 \leq i \leq k\}$$

Proof: If $c = \{f_1, f_2, \ldots, f_k\}$ and $c' = c \cup \{f_{k+1}\}$. As to any feature $f_i \in c$, the inequality $u(f_i, c') \leq u(f_i, c)$ holds.

So, we have $UPR(f_i, c') \leq w_1 \frac{u(f_i,c)}{u(f_i)} + w_2 \frac{\sum_{f_j \in c, j \neq i} u(f_j,c) + upbound\_u(f_{k+1},c')}{\sum_{f_j \in c, j \neq i} u(f_j) + u(f_{k+1})}$.

Based on Definition 6, we can infer that:

$$UPI(c') \leq \min\{w_1 \frac{u(f_i, c)}{u(f_i)} + w_2 \frac{\sum_{f_j \in c, j \neq i} u(f_j, c) + upbound\_u(f_{k+1}, c')}{\sum_{f_j \in c, j \neq i} u(f_j) + u(f_{k+1})}, \ 1 \leq i \leq k\}. \quad \square$$

Theorem 2. *Given a size k non-high utility co-location $c = \{f_1, f_2, ..., f_k\}$ and its size k+1 super-pattern $c' = c \cup \{f_{k+1}\}$, if there is a non-high utility feature $f_i \in c$ which satisfies $\frac{\sum_{f_j \in c, j \neq i} u(f_j,c)}{\sum_{f_j \in c, j \neq i} u(f_j)} > \frac{upbound\_u(f_{k+1},c')}{u(f_{k+1})}$, then c' is a non-high utility co-location, i.e., c can be pruned.*

Proof: For a non-high utility co-location $c = \{f_1, f_2, \ldots, f_k\}$ and $c' = c \cup \{f_{k+1}\}$, if f_i is a non-high utility feature in c and M is the UPI threshold, we have

$$UPR(f_i, c) = w_1 \frac{u(f_i, c)}{u(f_i)} + w_2 \frac{m}{n} < M \tag{4}$$

where $m = \sum_{f_j \in c, j \neq i} u(f_j, c)$ and $n = \sum_{f_j \in c, j \neq i} u(f_j)$.

According to Lemma 2, the UPR of f_i in c' satisfies the following inequality:

$$UPR(f_i, c') \leq w_1 \frac{u(f_i, c)}{u(f_i)} + w_2 \frac{m + upbound\_u(f_{k+1}, c')}{n + u(f_{k+1})}$$

According to Lemma 1, we have

$$\frac{m + upbound\_u(f_{k+1}, c')}{n + u(f_{k+1})} \leq \max\left\{\frac{m}{n}, \frac{upbound\_u(f_{k+1}, c')}{u(f_{k+1})}\right\}$$

If $\frac{m}{n} > \frac{upbound\_u(f_{k+1},c')}{u(f_{k+1})}$, the following inequality holds.

$$UPR(f_i, c') \leq w_1 \frac{u(f_i, c)}{u(f_i)} + w_2 \frac{m}{n}$$

Based on the inequality (4), we can infer that $UPR(f_i, c') < M$. So, c' must be a non-high utility co-location. □

For example, for c = {B, C, D} in Fig. 1, if $w_1 = w_2 = 0.5$ and $M = 0.5$, due to $T(c)$ = {{B.4^8, C.2^9, D.2^1}}, UPR(B, c) = 0.36, UPR(C, c) = 0.493 and UPR(D, c) = 0.268, c is a non-high utility co-location pattern. For the supper-pattern $c' =$ {B, C, D, E} of c, $upbound\_u(\mathrm{E}, c') = \min\{u(\mathrm{E}, \{\mathrm{B, C, E}\}) + u(\mathrm{E}, \{\mathrm{B, D, E}\}) + u(\mathrm{E}, \{\mathrm{C, D, E}\})\} = \min\{2, 2, 2\} = 2$. As to the feature B in {B, C, D}, we have:

$$\frac{u(\mathrm{C}, c) + u(\mathrm{D}, c)}{u(\mathrm{C}) + u(\mathrm{D})} = \frac{9+1}{12+13} > \frac{upbound\_u(\mathrm{E}, c')}{u(\mathrm{E})} = \frac{2}{14}$$

So, based on the computing results of size 3 co-locations, we can infer that the size 4 co-location {B, C, D, E} must be a non-high utility co-location.

Theorem 1, Corollary 2 and Theorem 2 are regarded as three pruning strategies to identify some non-high utility co-locations ahead of time.

5 Experimental Analysis

This section verifies the effect and efficiency of the basic algorithm and the algorithm with pruning strategies on synthetic and real data sets through experiments. The algorithms are implemented in Java 1.7 and run on a windows 8 operating system with 3.10 GHz Intel Core i5 CPU and 4 GB memory.

5.1 Data Sets

We conduct the experiments on synthetic data sets and plant data sets of the "Three Parallel Rivers of Yunnan Protected Areas". Synthetic data sets are generated using a spatial data generator similar to [3, 4], and the utilities of instances are assigned randomly between 0 and 20. In the plant data sets, we compute the utilities of plant instances according the plant price associated with size and kind of plant. The efficiency of the basic algorithm and the algorithm with pruning strategies are examined on the synthetic and real data sets.

5.2 The Quality of Mining Results

We aim at finding the high utility co-locations whose instances are frequently located together in geographic space and which have high utilities. So, we take the criterion $Q(c) = \sum_{f \in c} u(f, c) / \sum_{f \in c} u(f)$ to evaluate the quality of a mined co-location c.

In order to illustrate the interestingness measure UPI proposed in this paper is more reasonable, we compare the quality of mining results identified by different interestingness measures. They are the traditional participation index measure (PI), the traditional pattern utility ratio (PUR) proposed in [19] and the UPI proposed in our paper.

In the experiments of Fig. 2, we take the number of spatial features $|F|$ is 15, the total number of instances $|S|$ is 10000, the neighboring distance threshold d is 30, and $w_1 = 0.9$, $w_2 = 0.1$. Figure 2(a) shows the sum of quality of top-k interesting co-locations identified by the measure PI, PUR and UPI respectively. The x-axis refers to the value k, while y-axis is the sum of quality of top-k interesting patterns. Figure 2(b) shows the average quality of top-20 interesting patterns identified by the three measures over different sizes. The x-axis is the sizes of co-locations, while the y-axis is the average quality of top-20 interesting patterns. The results show that our UPI measure can identify higher quality co-locations, and it can extract top co-locations with higher average utility.

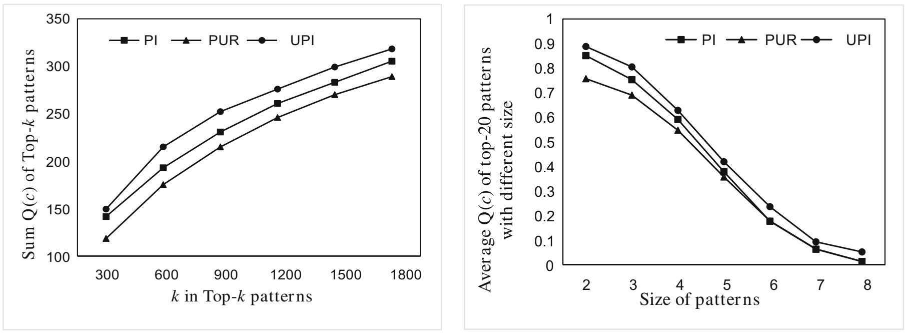

Fig. 2. Testing the quality of mining results, where (a) the sum of quality of top-k interesting patterns; (b) the average of quality of top-20 interesting patterns with different sizes.

5.3 Evaluation of Pruning Strategies

We evaluate the effect of pruning strategies with several workloads, e.g. different numbers of instances, neighbor distance thresholds, UPI thresholds and pruned rate on synthetic and real data sets.

5.3.1 Influence of the Number of Instances

We compare the running time of the basic algorithm and the algorithm with pruning strategies on synthetic and real data sets. We set $|F| = 20$, $d = 20$, $M = 0.3$, $w_1 = w_2 = 0.5$, the running time of two algorithms by increasing the number of instances is shown in Fig. 3. The x-axis represents the number of total instances and the y-axis is the running time. As a result, the performance of the algorithm with pruning strategies is better than the basic algorithm both in synthetic and real data sets. Compared with synthetic data sets, the neighbor relationships of real data sets are relatively fewer,

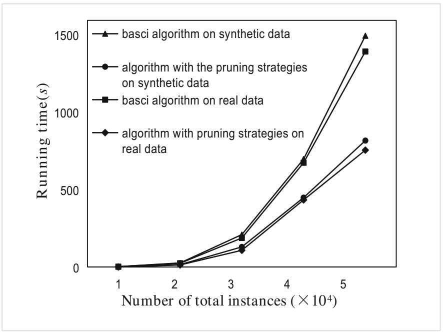

Fig. 3. The influence of the number of instances over synthetic and real data sets

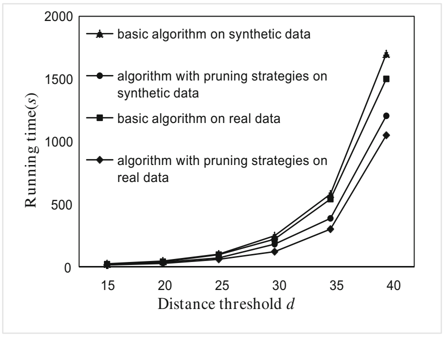

Fig. 4. The influence of the distance thresholds over synthetic and real data sets

which results in less row instances to be computed. So, in our experiments, the runtime of the algorithms on real data sets is less than that on synthetic data sets.

5.3.2 Influence of the Distance Threshold *d*

In Fig. 4, we set $|F| = 20$, $|S| = 10000$, $M = 0.3$, $w_1 = w_2 = 0.5$. We compare the running time of two algorithms by changing the distance threshold *d*, where the *x*-axis denotes the distances threshold *d* and the *y*-axis represents the running time in different data sets. From Fig. 4, we can see that the algorithm with pruning strategies is still faster than the basic algorithm. However, both algorithms have a huge time-cost with the increase of *d*. This is because with increasing *d*, there are more cliques formed, which results in huge row instances to be computed and more time consumption.

5.3.3 Influence of the UPI Threshold *M*

The parameters are set $|F| = 20$, $|S| = 10000$, $d = 15$, $w_1 = w_2 = 0.5$ in this experiment. The running time of two algorithms by changing the UPI threshold *M* is shown in Fig. 5. The *x*-axis denotes the value of the UPI threshold *M* and the *y*-axis is the running time. With the increase of *M*, the more non-high utility co-locations are pruned ahead of time, which improve the efficiency of the algorithm with pruning strategies.

5.3.4 Pruned Rate

In order to examine efficiency of the three pruning strategies (Theorem 1, Corollary 2, and Theorem 2), we count the number of candidates pruned by each pruning strategy respectively. In the experiment, we set $|F| = 15$, $|S| = 4000$, $d = 20$, $M = 0.3$, $w_1 = w_2 = 0.5$, and we randomly generate 5 different data sets whose size is similar to each other. We independently run the algorithm with pruning strategies on 5 different data sets and compute the average proportion of the candidates pruned by each strategy. The statistic result is shown in Fig. 6.

The result shows that the pruning strategies are very efficient. However, the efficiency of pruning strategies doesn't be improved in the same degree. There are two reasons. First, the process of pruning candidates would cost some time. Second, some pruned co-locations may be used to generate the table instances of super co-locations,

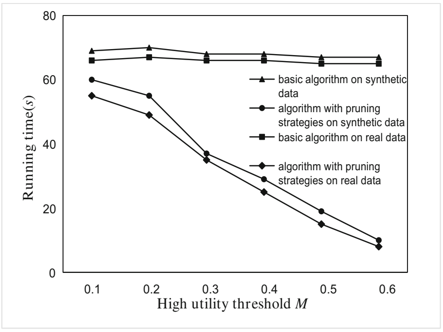

Fig. 5. The influence of the UPI threshold *M* over synthetic and real data sets

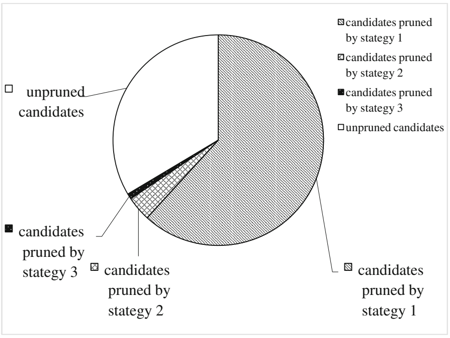

Fig. 6. The proportion of candidates pruned by each strategy

so we might have to generate the table instances of pruned co-locations, which has a negative effect on the algorithm. Fortunately, it rarely occurs in the experiments. Thus, the average efficiency of pruning strategies is obvious.

In addition, the basic algorithm and the algorithm with pruning strategies presented in this paper convert spatial data sets into the star neighborhood partition model in [4]. Algorithms in both papers store spatial neighbor relationships and table instances of current candidates. Therefore, the memory cost of our algorithms is similar to the join-less algorithm in [4]. Due to the non-existence of the downward closure property, the scalability of the basic algorithm requires improvement. From Figs. 3, 4, 5 and 6, we can see that the pruning strategies significantly reduce the overall runtime of the basic algorithm, while in some extreme cases less so. Further improvement of scalability is left for future work.

6 Conclusion and Future Work

Different from the previous researches, in this paper we take the instances with utilities as study objects which are near to real world and a new interesting measure is proposed. We combine the intra-utility ratio and the inter-utility ratio into the utility participation index for identifying high utility co-locations, which is comprehensive and reasonable. Because the utility participation index does not satisfy the downward closure property, we propose the effective pruning strategies to improve the efficiency of finding high utility co-locations. The experiments on synthetic and real data sets show that the pruning strategies significantly reduce the overall runtime of the basic algorithm. Although the algorithm with pruning strategies is better than the basic algorithm, it also shows less improvement in some extreme case. Our future work focuses on designing algorithms for bigger data sets and better pruning strategies.

Acknowledgments. This work is supported by the National Natural Science Foundation of China (61472346, 61662086), the Natural Science Foundation of Yunnan Province (2015FB114, 2016FA026), the Spectrum Sensing and borderlands Security Key Laboratory of Universities in Yunnan (C6165903), and the Program for Young and Middle-aged Skeleton Teachers of Yunnan University.

References

1. Koperski, K., Han, J.: Discovery of spatial association rules in geographic information databases. In: Egenhofer, M.J., Herring, J.R. (eds.) SSD 1995. LNCS, vol. 951, pp. 47–66. Springer, Heidelberg (1995). doi:10.1007/3-540-60159-7_4
2. Shekhar, S., Huang, Y.: Co-location rules mining: a summary of results. In: Spatio-Temporal Symposium on Databases (2001)
3. Huang, Y., Shekhar, S., Xiong, H.: Discovering co-location patterns from spatial data sets: a general approach. IEEE Trans. Knowl. Data Eng. **16**(12), 1472–1485 (2004)
4. Yoo, J.S., Shekhar, S.: A joinless approach for mining spatial co-location patterns. IEEE Trans. Knowl. Data Eng. **18**(10), 1323–1337 (2006)
5. Sengstock, C., Gertz, M., Tran Van, C.: Spatial interestingness measures for co-location pattern mining. In: SSTDM (ICDM Workshop 2012), pp. 821–826. IEEE Press, New York (2012)
6. Wang, L., Wu, P., Chen, H.: Finding probabilistic prevalent colocations in spatially uncertain data sets. IEEE Trans. Knowl. Data Eng. **25**(4), 790–804 (2013)
7. Wang, L., Chen, H., Zhao, L., Zhou, L.: Efficiently mining co-location rules on interval data. In: Cao, L., Feng, Y., Zhong, J. (eds.) ADMA 2010. LNCS (LNAI), vol. 6440, pp. 477–488. Springer, Heidelberg (2010). doi:10.1007/978-3-642-17316-5_45
8. Wang, L., Bao, Y., Lu, Z.: Efficient discovery of spatial co-location patterns using the iCPI-tree. Open Inf. Syst. J. **3**(2), 69–80 (2009)
9. Huang, Y., Pei, J., Xiong, H.: Mining co-location patterns with rare events from spatial data sets. Geoinformatica **10**(3), 239–260 (2006)
10. Yao, H., Hamilton, H.J., Butz, C.J.: A foundational approach to mining itemset utilities from databases. In: 4th SIAM International Conference on Data Mining, pp. 482–486 (2004)
11. Liu, Y., Liao, W.-K., Choudhary, A.: A two-phase algorithm for fast discovery of high utility itemsets. In: Ho, T.B., Cheung, D., Liu, H. (eds.) PAKDD 2005. LNCS (LNAI), vol. 3518, pp. 689–695. Springer, Heidelberg (2005). doi:10.1007/11430919_79
12. Ahmed, C.F., Tanbeer, S.K., Jeong, B.S., et al.: A framework for mining interesting high utility patterns with a strong frequency affinity. Inf. Sci. **181**(21), 4878–4894 (2011)
13. Hong, T.P., Lee, C.H., Wang, S.L.: An incremental mining algorithm for high average-utility itemsets. Expert Syst. Appl. **39**(8), 7173–7180 (2012)
14. Tseng, V.S., Wu, C.W., Shie, B.E., et al.: UP-Growth: an efficient algorithm for high utility itemset mining. In: 16th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 253–262. ACM, New York (2010)
15. Shie, B.E., Tseng, V.S., Yu, P.S.: Online mining of temporal maximal utility itemsets from data streams. In: ACM Symposium on Applied Computing, pp. 1622–1626. ACM, New York (2010)
16. Yin, J., Zheng, Z., Cao, L.: USpan: an efficient algorithm for mining high utility sequential patterns. In: 18th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 660–668. ACM, New York (2012)

17. Lin, Z., Lim, S.: Optimal candidate generation in spatial co-location mining. In: ACM Symposium on Applied Computing, pp. 1441–1445. ACM, New York (2009)
18. Verhein, F., Al-Naymat, G.: Fast mining of complex spatial co-location patterns using GLIMIT. In: SSTDM (ICDM Workshop 2007), pp. 679–984. IEEE Press, New York (2007)
19. Yang, S., Wang, L., Bao, X., Lu, J.: A framework for mining spatial high utility co-location patterns. In: 12th International Conference on Fuzzy Systems and Knowledge Discovery (FSKD 2015), pp. 595–601. IEEE Press, New York (2015)
20. Wang, X., Wang, L., Lu, J., Zhou, L.: Effectively updating high utility co-location patterns in evolving spatial databases. In: Cui, B., Zhang, N., Xu, J., Lian, X., Liu, D. (eds.) WAIM 2016. LNCS, vol. 9658, pp. 67–81. Springer, Heidelberg (2016). doi:10.1007/978-3-319-39937-9_6