INTERNATIONAL JOURNAL OF GEOGRAPHICAL INFORMATION SCIENCE, 2018
https://doi.org/10.1080/13658816.2018.1431839

Taylor & Francis
Taylor & Francis Group

ARTICLE

A spatial co-location mining algorithm that includes adaptive proximity improvements and distant instance references

Xiaojing Yao [a], Liujia Chen[a], Congcong Wen[a], Ling Peng[a], Liang Yang[b], Tianhe Chi[a], Xiaomeng Wang[a] and Wenhao Yu [c]

[a]Lab of Spatial Information Integration, Institute of Remote Sensing and Digital Earth, Chinese Academy of Sciences, Beijing, China; [b]State Key Laboratory of Information Security, Institute of Information Engineering, Chinese Academy of Sciences, Beijing, China; [c]Faculty of Information Engineering, China University of Geosciences, Wuhan, China

ABSTRACT

Spatial co-location pattern mining is employed to identify a group of spatial types whose instances are frequently located in spatial proximity. Current co-location mining methods have two limitations: (1) it is difficult to set an appropriate proximity threshold to identify close instances in an unknown region, and (2) such methods neglect the effects of the distance values between instances and long-distance instance effects on pattern significance. This paper proposes a novel maximal co-location algorithm to address these problems. To remove the first constraint, the algorithm uses Voronoi diagrams to extract the most related instance pairs of different types and their normalized distances, from which two distance-separating parameters are adaptively extracted using a statistical method. To remove the second constraint, the algorithm employs a reward-based verification based on distance-separating parameters to identify the prevalent patterns. Our experiments with both synthetic data and real data from Beijing, China, demonstrate that the algorithm can identify many interesting patterns that are neglected by traditional co-location methods.

ARTICLE HISTORY
Received 11 September 2015
Accepted 20 January 2018

KEYWORDS
Spatial data mining; co-location pattern mining; reward value; Voronoi diagram; generalized extreme value distribution

1. Introduction

Mining co-location patterns is a technique employed to identify relationships among different types of spatial data. This technique is applied extensively in species distribution analysis (Shekhar and Huang 2001, Sierra and Stephens 2012), public safety (Leibovici *et al*. 2014), environmental management (Akbari *et al*. 2015), site selection, and other fields. In site selection, for example, an investigation of commercial entities reveals that most banks are located around restaurants in urban areas. Based on this finding, decision makers can scientifically determine restaurant locations using a specific site selection model.

CONTACT Ling Peng plqiqi@126.com

Yao, X. J., Chen, L. J., Wen, C. C. and Yang, L. design and implement the algorithms, as well as the examples. Peng, L., Chi, T. H., Wang, X. M. and Yu, W. H. propose modifications for the paper.

Most current co-location methods involve two steps: first, different types of instances that have close relationships are connected. Second, frequency mining technology is adopted to identify frequent patterns through calculating the prevalence index. The prevalence index is the minimum participation ratio among the types of a candidate pattern based on its instance table and can measure the significance of a co-location pattern (Shekhar and Huang 2001).

1.1. *Context*

Co-location was first proposed by Shekhar and Huang (2001) and followed by a join-based algorithm using an Apriori strategy (Morimoto 2001). Subsequently, many researchers have developed a variety of interesting algorithms that yielded satisfactory results. Some of these algorithms, such as the partial-join algorithm (Yoo and Shekhar 2004), the join-less algorithm (Yoo and Shekhar 2006), the density clustering algorithm (Huang *et al*. 2008), the order-clique-based algorithm (Wang *et al*. 2009), the parallel co-location algorithm (Yoo *et al*. 2014) and the SGCT algorithm (Yao *et al*. 2016), focused on efficiency improvements. Additionally, others have expanded the data categories by introducing co-location mining algorithms for lines and polygons (Xiong *et al*. 2004), spatio-temporal data (Leibovici *et al*. 2014), uncertain data (Wang *et al*. 2013), or rare types of data (Huang *et al*. 2006). Moreover, some have enlarged the mining targets by proposing algorithms such as the negative co-location algorithm (Wan *et al*. 2008) and the co-location algorithm with constraints (Flouvat *et al*. 2015). However, all the above algorithms require a predefined distance threshold or deformation concepts (Huang *et al*. 2008) to separate close instances from distant ones. This requirement restricts their implementation in unknown regions with little prior information, such as density information.

To improve the adaptability of these mining techniques, some scholars (Wan and Zhou 2008, Qian *et al*. 2012, Sundaram *et al*. 2012, Qian *et al*. 2014, Deng *et al*. 2017) have deployed novel approaches. For example, Wan and Zhou (2008) proposed a *k*-nearest features-based co-location algorithm in which the neighbour relationships among instances depend on the number of nearest objects and *k*. However, these algorithms still demand knowledge of proximity standards, such as a *k* threshold, prior to implementation. To address this limitation, some adaptive algorithms that require no proximity standards have been developed. For instance, Sundaram *et al*. (2012) used Delaunay triangulation to find co-location instances. This approach considers nodes connected by a triangle edge to be neighbours. Qian *et al*. (2012) presented an iterative framework to discover prevalent co-location patterns. This method iteratively selects informative edges to construct a neighbour relationship graph until every significant co-location has sufficient confidence based on absolute and relative prevalence. Qian *et al*. (2014) explored a hierarchical co-location algorithm that adopts a *k*-nearest neighbour graph instead of a distance threshold to discover regional co-location patterns. However, these algorithms use topological predicates or proximity predicates that offer limited consideration of the distance decay effect, which describes how near instances are more closely related than distant instances in space (Tobler 1970). This oversight of the detailed distance values among instances affects the meaning of the mined patterns for use with high-accuracy data.

1.2. *Challenges*

Recently, some studies have addressed distance decay effects associated with co-location issues and used the kernel density model to imitate the degree of tightness among close instances, which improves the precision of the prevalent patterns (Yao *et al*. 2017, Yu *et al*. 2017). However, these approaches still have some limitations. First, the close-instance connections do not involve far instances that would introduce negative effects to the pattern's significance. Far-instance effects are mainly discussed in another related topic called 'de-location' or 'negative co-location,' whose goal is to find type groups with effects that are exclusive of each other (Wan *et al*. 2008). Similar to the current co-location methods, most de-location algorithms only consider distant instances that comply with a predefined proximity measurement. From that viewpoint, the distant instances truly affect the significance of a pattern in the context of co-location mining. However, few studies consider both close and far instances at the same time. Second, it is difficult to set an appropriate proximity standard with which to connect instances. Although some valid approaches have been proposed, they mainly use *k*-nearest or optimization mechanisms (Qian *et al*. 2012, 2014), and they still suffer from the aforementioned limitations relating to the distance values or far-instance effects. Based on the above statement, it is difficult to integrate both the adaptability characteristics and the effects of close and far instances in co-location. Such inadequacies significantly weaken the meaning of the prevalent patterns. This paper is designed to address these problems.

1.3. *Contributions*

In this paper, we propose a novel maximal co-location mining algorithm that includes adaptive proximity improvements and distant instance references (ADMC). Specifically, we present the following three innovations.

(1) We use a Voronoi-based method (Aurenhammer 1991) to connect different-type instances that have extreme spatial relationships with each other. These connected instances are representatives of all the different-type instance pairs in space and involve both close and far instances.
(2) A statistical method is proposed to adaptively estimate two 'distance-separating parameters' based on the distances between these connected instances. This method removes the requirement for a predefined proximity standard, as has been adopted by traditional algorithms, thereby making the algorithm more adaptive.
(3) Based on the aforementioned distance-separating parameters, a new method with distance decay interference is designed to measure pattern prevalence. Consequently, the prevalent patterns are identified according to a 'reward value' rather than a prevalence index. This solution considers the distance effects of both close and far instances, thus making the final patterns more trustworthy.

Our experiments use both synthetic data and real spatial data concerning points of interest (POIs) in Beijing, China, in 2014. The results show that the ADMC algorithm is more adaptive and reliable than other state-of-the-art techniques.

The remainder of this paper is organized as follows. Section 2 presents the ADMC algorithm in detail. Section 3 discusses the computational complexity of the ADMC algorithm. The performance of the ADMC algorithm for both synthetic and real datasets is discussed in Section 4. Section 5 provides conclusions and suggestions for future study directions.

2. Methods

This section discusses the details of the ADMC algorithm. To formally describe the ADMC algorithm, we assume a set of spatial event types $E = \{e_1,e_2,\ldots,e_m\}$ and instance objects $O = \{o_1,o_2,\ldots,o_n\}$ in two-dimensional Euclidean space. Each instance o_i contains information that can be denoted as <instance *ID*, event type e_j, location(*X,Y*)>. The instances of a single type e_i are stored in a set $O(e_i)$ with the cardinal number of n_i. The ultimate task is to mine all the maximal co-location patterns that satisfy a given reward threshold τ ($0 < \tau < 1$), which replaces the prevalence threshold used in traditional co-location algorithms.

The ADMC algorithm is implemented in three steps. First, using a Voronoi- based method, a size-two normalized instance table (*NT*) is constructed from *O* and *E*. Second, using a statistical approach, two distance-separating parameters are estimated from *NT*. These two parameters are used to construct a reward-based verification method to identify prevalent patterns. Third, based on a maximal mining framework (Yao *et al*. 2016), all the prevalent co-location patterns satisfying the given reward threshold are obtained. The preceding steps correspond to the subsequent three sections.

2.1. *Size-two normalized instance table construction*

This section describes the construction of the size-two normalized instance table (*NT*) using Voronoi diagrams. A general Voronoi diagram shows the partitioning of a plane into regions based on the distances between spatial instances in the plane. These instances are specified beforehand; for each instance, a region exists in which all spatial points are closer to this instance than to any others (Aurenhammer 1991). A weighted Voronoi diagram is considered if the instances have different levels of impact; in such cases, a complicated index that includes additional factors, such as the distance and instance scale, is regarded as a proximity index to replace distance alone. Here, for simplicity, we use only the general Voronoi diagram. The relative definitions are as follows.

Definition 1 (Instance pair set constrained by a type pair): Given the instances of a single type e_i, a Voronoi diagram can be constructed by regarding these instances as kernels. Then, the entire space can be divided into n_i cells. The instance pair set constrained by a type pair (e_i,e_j) is a set that stores all the instances of type e_j with their corresponding kernels of type e_i, defined as follows:

$$T(e_i, e_j) = \{(o_x, o_y) | o_y \in O(e_j),\ o_x \in O(e_i)\}, \tag{1}$$

where (o_x,o_y) is an instance pair consisting of a kernel o_x and an instance o_y of type e_j in its corresponding cell. Each instance of type e_j can find its kernel; therefore, the cardinal number of $T(e_i,e_j)$ is n_j. For the general Voronoi hypothesis, $T(e_i,e_j)$ can be acquired by the 1-nearest strategy – that is, each instance of type e_j should search out the nearest instance of type e_i.

Example 1: Suppose there are instances of two types A and B in space. Figure 1(a,b) shows the Voronoi diagram based on the instances of type A and B, respectively. We extract a sample of instances – for example, if $P = \{A_1,A_2,A_3,B_1,B_2,B_3,B_4,B_5\}$, then $T(A, B) = \{(A_2,B_1),(A_3,B_2),(A_3,B_3),(A_1,B_4),(A_1,B_5)\}$, and $T(B,A) = \{(B_2,A_1),(B_1,A_2),(B_2,A_3)\}$. The cardinal numbers of the two sets are 5 and 3, which are equal to the instance numbers of types B and A.

As Example 1 shows, the instance pairs have directionality. Specifically, the instances of (A_3,B_3), (A_1,B_4) and (A_1,B_5) in Figure 1(a) do not have connecting relationships in Figure 1(b), but the instances of (A_2,B_1) and (A_3,B_2) can be connected in both Figures. Because co-location pattern mining is a non-directional issue, we should eliminate the redundant instance connections (see Definition 2).

Definition 2 (Size-two normalized instance table): The instances of each instance pair in $T(e_i,e_i)$ for all $e_i<e_j$ are reordered by the lexicographical order of their types, forming a new set $T^*(e_i,e_i)$. The size-two normalized instance table is defined as follows:

$$NT = \{\langle o_x, o_y, Ndis(o_x, o_y)\rangle | (o_x, o_y) \in T^*(e_j, e_i) \cup^T (e_i, e_j) \text{ for all } 1 \le e_i < e_j \le m\}. \quad (2)$$

In Equation (2), $<o_x,o_y,Ndis(o_x,o_y)>$ is a triple, where o_x and o_y are instances that have connecting relationships and $Ndis(o_x,o_y)$ is their normalized distance (see Definition 3). Additionally, m is the type number. A subset containing all triples that relate to types e_i and e_j is denoted as $NT(e_i,e_j)$. NT can be interpreted as a complicated two-dimensional upper triangular matrix, as illustrated in Example 2.

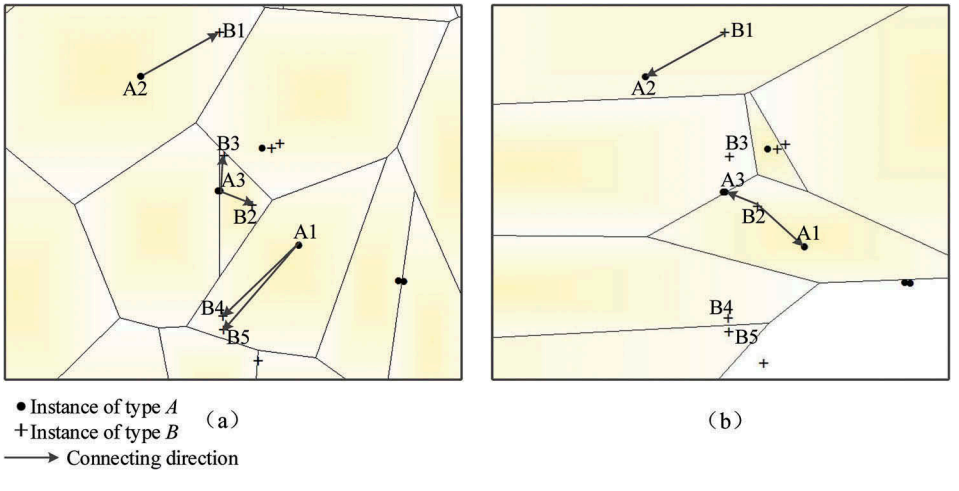

Figure 1. The Voronoi diagram based on the instances of type A and B.

Definition 3 (Normalized distance of an instance pair): The normalized distance of an instance pair (o_x,o_y) is calculated by

$$Ndis(o_x, o_y) = \frac{dis(o_x, o_y) - \min(D_T)}{\max(D_T) - \min(D_T)}, \tag{3}$$

where $dis(o_x,o_y)$ is the Euclidean distance between instances o_x and o_y, D_T is a collection that includes all the distance values of instance pairs in space, and min(·) and max(·) calculate the minimum and maximum values of the input collection, respectively. The set that contains all the normalized distances of *NT* is defined as $NorD_{NT}$ and is used to estimate the distance-separating parameters in Section 2.2.

Example 2: A new set, $T^*(B,A)$, is derived from $T(B,A) = \{(B_2,A_1),(B_1,A_2),(B_2,A_3)\}$ in Example 1. As described in Definition 2, the instance pairs in $T(B,A)$ are reordered by their types. For instance, (B_2,A_1) is changed to (A_1,B_2) because $A < B$. All instance pairs in $T(B,A)$ are processed in this way; finally, $T^*(B,A) = \{(A_1,B_2),(A_2,B_1),(A_3,B_2)\}$ is acquired. The non-redundant instance pairs are calculated by $T^*(B,A) \cup T(A,B) = \{(A_1,B_2),(A_1,B_5),(A_1,B_4),(A_2,B_1),(A_3,B_2),(A_3,B_3)\}$. Computing the normalized distances of these instance pairs finally yields $NT = \{<A_1,B_2,0.82>, <A_1,B_5,0.89>,<A_1,B_4,0.8>,<A_2,B_1,0.66>,<A_3,B_2,0.34>,<A_3,B,0.31>\}$, as illustrated by the matrix in Figure 2. Then, $NorD_{NT} = \{0.82,0.89,0.8,0.66,0.34,0.31\}$.

Based on the aforementioned definitions, we can identify that *NT* is different from the size-two instance tables of current studies in two respects. First, in addition to the instance pairs, *NT* also calculates their normalized distances. Second, the instance-connecting strategy differs. Figure 3 shows an illustration using the star model (Wang *et al*. 2009) comparing our strategy with three other classical instance-connecting strategies proposed in contemporary studies (Wan and Zhou 2008, Wang *et al*. 2009, and Sundaram *et al*. 2012).

Example 3: As shown in Figure 3, there are instances of three types *A*, *B* and *C* in space. Using the star model, Figure 3(a–d) shows the instance-connecting status of three instances B_1, B_2 and B_3 using the distance threshold strategy, *k*-nearest strategy, Delaunay triangulation strategy and our strategy, respectively. We can see that B_2 has

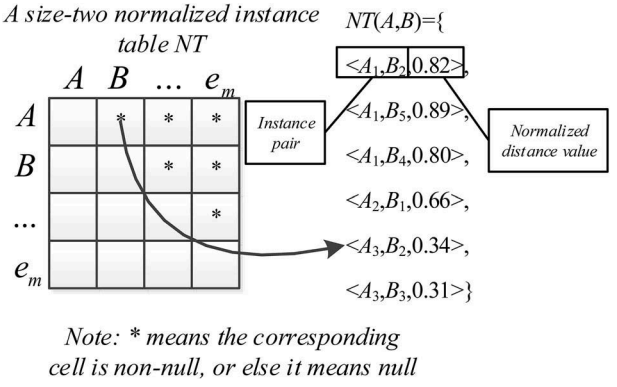

Figure 2. A size-two normalized instance table.

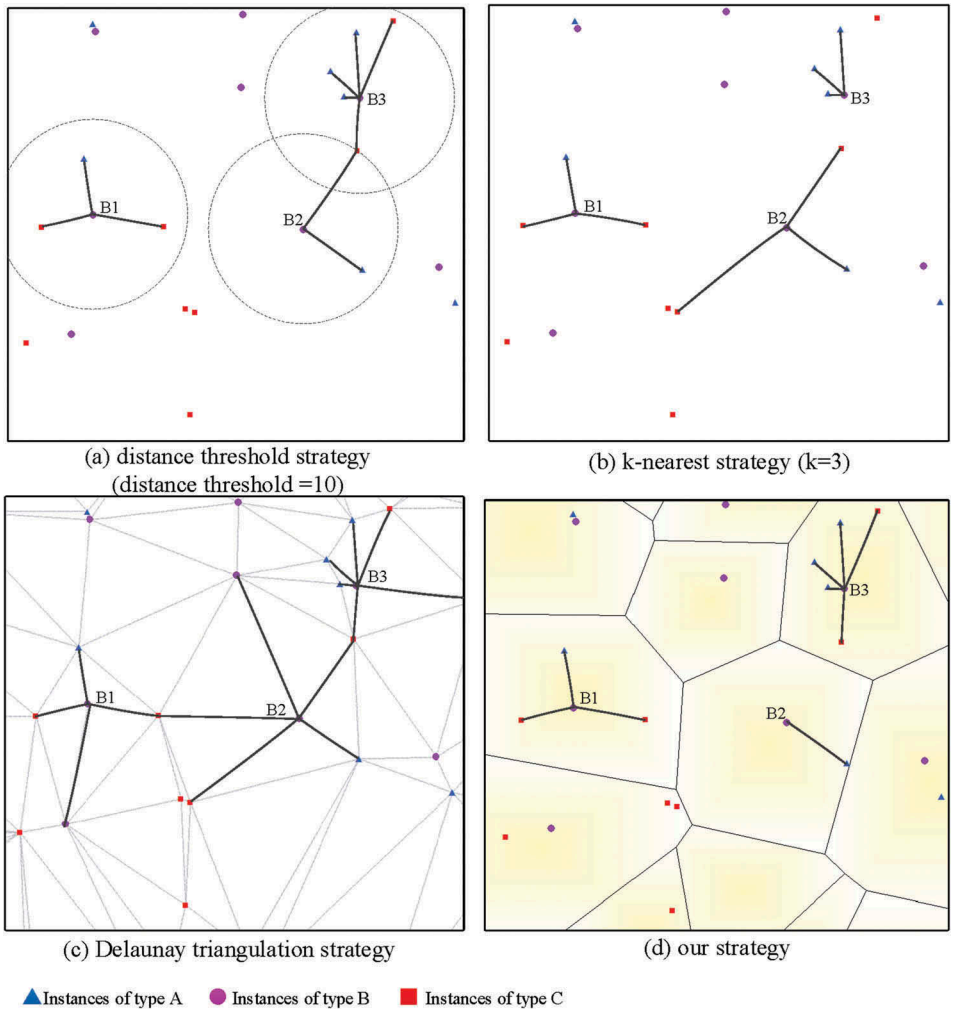

Figure 3. An illustration comparing our strategy with three other classical instance-connecting strategies proposed in contemporary studies.

connecting relationships with 2 instances, 3 instances, 5 instances and 1 instance according to these four strategies. The connecting statuses of the former two strategies will change when their proximity thresholds change.

Examples 3 highlight three properties of *NT* that differentiate *NT* from other size-two instance tables acquired by generic means.

Property 1: In *NT*, each instance is paired with at least one instance of all other types.

Proof: Based on Definition 1, each instance of type e_j will search out the nearest instance of a different type e_i when the instances of type e_j are treated as kernels. That is, each instance in $O(e_j)$ will be paired with one instance of type e_i, and vice versa. Then, the instance pairs in $NT(e_i,e_j)$ are calculated by uniting all instance pairs with types e_i and

e_j through Definition 2. Thus, each instance of type e_i will have at least one pair of type e_j, and vice versa. To construct *NT*, we traverse the above process for all different-type pairs; therefore, each instance will pair with at least one instance of another type.

Based on Property 1, we can see that even if two types have no co-location meaning, all their instances will participate in the connections because the instance pairs are formed based on the influence region of each instance rather than on a proximity standard. Example 2 clearly shows this property.

Property 2: *NT* contains not only close-instance pairs in a general sense but also relatively far pairs.

Proof: Most spatial instances are stable within their force field (Burrough 1986). If a mutual attraction exists between two types, their instances will mainly be distributed in each other's core regions where the influence is stronger in the corresponding Voronoi cell (such as the regions with darker colours in Figure 3(d)). Otherwise, they will be distributed in the regions where the influence is considerably weaker (such as the regions with lighter colours in Figure 3(d)). In this sense, the majority of the normalized distances for which the instance pairs have potential co-location relationships are lower than the average level among all normalized distances in *NT*. The average level of the normalized distance separates close instances from relatively far ones.

Property 3: The distribution of $NorD_{NT}$ tends to satisfy the Generalized Extreme Value (GEV) function.

Proof: The GEV distribution is a continuous probability distribution that is often used as an approximation to model the extremes in long sequences of random variables. The problem of modelling extreme or rare events arises in many areas. Some examples of rare events include extreme floods and snowfalls, high wind speeds and extreme temperatures. To develop appropriate probabilistic models and assess the risks caused by these events, scientists frequently use extreme value distributions (Coles *et al*. 2001). Based on Definition 1, the instance pairs in *NT* are obtained by the 1-nearest method – that is, each instance of a specific type should search out the nearest instance of a different type. Thus, these instance pairs are the most spatially related instances among all the different-type instance pairs in a plane. Accordingly, their normalized distances are the minima among those of all the different-type instance pairs. Thus, these normalized distances can be considered an outcome of an operation of extreme values, and their distribution tends to satisfy the GEV function.

Properties 2 and 3 are verified in Section 4.1, which lays the foundation for the reward-based verification method of a prevalent pattern in Section 2.3.

2.2. *Distance-separating parameter estimation*

In Property 2, the instance pairs in *NT* are not only close pairs but are also mixed with relatively distant ones. We presume there are two points that can divide these pairs into close, average and distant groups. Here, we provide a definition of these two points as follows.

Definition 4 (Distance-separating parameters): The distance-separating parameters l_{far} and l_{close} are derived from the statistical elements of all the normalized distances in *NT*, i.e. $NorD_{NT}$. These parameters are cutoff signals that divide the [0,1] distance axis into small, average and large groups and provide clues about the data density.

To detect the distribution characteristics of the normalized distances in *NT*, we use a statistical method, assuming that the space is continuous. In Property 3, we propose that the normalized distance satisfies the Generalized Extreme Value (GEV) function. The GEV function is a family of continuous probability distributions developed within extreme value theory to combine the Gumbel, Fréchet and Weibull families, also known as type I, II and III extreme value distributions. These three families can be nested into a single representation with three parameters, i.e. the location parameter μ, a scale parameter σ, and a shape parameter k. This is a skewed bell-like distribution, in which $\mu \pm stdrr$ (*stderr* is the standard deviation of the estimated location parameter μ) is the value of the x observation that is most apparent in real data (Coles *et al*. 2001). In our application, $\mu \pm stdrr$ is the value of the normalized distance that is most apparent in *NT*. Similar to the normal distribution curve, $[\mu - stdrr, \mu + stdrr]$, the peak bound of the real data in the [0,1] distance axis denotes the average level of a sample dataset. It divides the distance axis into three intervals. Distances larger than the bound are generally affiliated with the far-instance group, whereas distances smaller than the bound are affiliated with the near-instance group. Thus, the two distance-separating parameters are defined as $l_{close} = \mu - stdrr$ and $l_{far} = \mu + stdrr$.

To calculate these two parameters, we construct a frequency distribution histogram for $NorD_{NT}$ using the Freedman-Diaconis rule (Freedman and Diaconis 1981), a widely used approach for selecting the size of the bins in a histogram. Then, we employ maximum likelihood estimation (MLE) (Hosking *et al*. 1985) with a significance (α) of 0.05 to estimate the parameters of the GEV distribution function of the data. Specifically, we regard each middle value of the histogram interval as an x observation and its corresponding frequency as a y observation. The fitting function is the regressive curve of these observation pairs composed of all the x and y values. Finally, their coefficients of determination R^2 (Nagelkerke 1991) are calculated to indicate how well the observation pairs fit this function. When R^2 is satisfactory, $\mu \pm stdrr$ values are extracted as the two distance-separating parameters, as illustrated in Figure 4. Section 4.1 provides experiments on real data with a fitting analysis to visually clarify this process.

2.3. *Prevalent pattern mining*

This section provides details on mining prevalent co-location patterns from the size-two normalized instance table (*NT*) and the two distance-separating parameters (l_{far} and l_{close}) obtained in the previous sections. To offset the time costs of the distance-separating parameter estimation, we employ a fast maximal pattern-mining framework (Wang *et al*. 2009, Yao *et al*. 2016) to return all prevalent maximal co-location patterns. A prevalent maximal co-location pattern is a concise representative of the prevalent patterns, and it can be defined as follows.

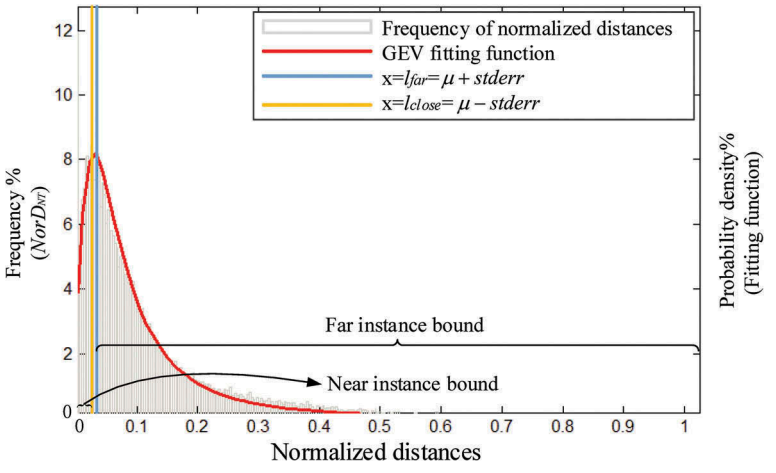

Figure 4. An illustration of distance-separating parameter extraction.

Definition 5 (Prevalent maximal co-location pattern): Given a prevalent co-location pattern $Pm = \{e_l,\ldots,e_v\}$, where $l,v\in\{1,2,\ldots,n\}$ and $Pm{\cdot}E$, if $Pm \cup e_i$ is a non-prevalent co-location for any $e_i\in E$ and $e_i \notin Pm$, the prevalent co-location Pm is called a prevalent maximal co-location pattern.

We then describe the reward value calculation for a candidate pattern and detail the mining procedure used by the ADMC algorithm.

2.3.1. *Reward-based verification for a candidate pattern*

In the Introduction, we revealed that the prevalence index calculation is replaced by a reward-based verification method to identify whether a candidate pattern is prevalent. The reward value of a candidate pattern is calculated based on the instance cliques derived from the instance pairs in *NT* and their related normalized distances. Instance cliques are acquired using a condensed instance tree construction proposed by Yao *et al.* (2016); we do not expound on that method here. Some related definitions are given below.

Definition 6 (Reward value of an instance pair): The reward value of an instance pair $R(o_x,o_y)$ reflects the effect of an instance pair (o_x,o_y) on its corresponding type pair (e_i,e_j). This is a monotonically decreasing function that uses l_{far} and l_{close}, defined as follows:

$$R(o_x,o_y)=\begin{cases}\left(\frac{l_{close}-Ndis(o_x,o_y)}{l_{close}}\right)^2, & Ndis(o_x,o_y)<l_{close}\\ -\left(\frac{Ndis(o_x,o_y)-l_{far}}{1-l_{far}}\right)^2, & Ndis(o_x,o_y)>l_{far}\\ 0, & l_{close}\le Ndis(o_x,o_y)\le l_{far}\end{cases}. \tag{4}$$

Figure 5 shows a diagram of Equation (4) with an l_{far} value of 0.6 and an l_{close} value of 0.4. Compared with most general co-location methods, which focus only on close instances, Equation (4) also considers the weakening effect of the more distant instances and presents a reward calculation when the normalized distance is $(l_{far},1]$. A value within $[l_{close}, l_{far}]$ denotes that most normalized distances fall into this bound, and such a value

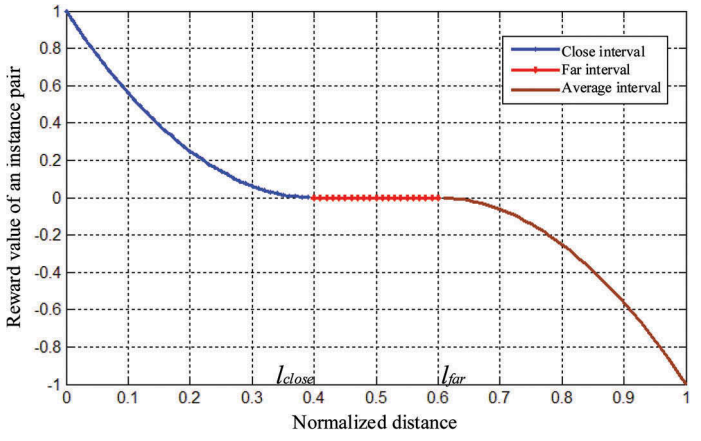

Figure 5. A diagram of Equation (4) with an l_{far} value of 0.6 and an l_{close} value of 0.4.

has less co-location meaning. Thus, [l_{close}, l_{far}] is regarded as an interval without a calculation design.

Definition 7 (Reward value of a candidate pattern, *Cm*): We first define the reward value of an instance pair (o_x,o_y) projected on *Cm* as shown below,

$$R_{Cm}(o_x, o_y) = \begin{cases} R(o_x, o_y), & \text{if}(o_x, o_y) \in \pi_{(e_i,e_j)}(insT_{Cm}) \\ -\text{abs}(R(o_x, o_y)), & \text{otherwise} \end{cases}, \tag{5}$$

where $insT_{Cm}$ is the instance table constructed from all the instance cliques of *Cm* (Shekhar and Huang 2001). Here, $\pi_{(e_i,e_j)}(insT_{Cm})$ is an instance projecting process that produces a set containing non-repetitive instance pairs of type e_i and e_j that have connecting relationships in $insT_{Cm}$. The abs(·) function calculates the absolute value of the input. Accordingly, the reward value of a type pair (e_i,e_j) projecting on *Cm* is as follows:

$$R_{Cm}(e_i, e_j) = \text{avg}\{R_{Cm}(o_x, o_y) | (o_x, o_y) \text{is in} NT(e_i, e_j) \text{ for all members}\}, \tag{6}$$

where the avg(·) function calculates the average value of the input set. The reward value of *Cm* is the minimum value of the entire reward set, which collects the reward values of all the type pairs extracted from *Cm*, i.e.

$$R_{Cm} = \min_{1 \le i < j \le k}\{R_{Cm}(e_i, e_j)\}. \tag{7}$$

From the above statement, we can infer that $R_{Cm} \in (-1,1)$. A prevalent co-location pattern should satisfy $R_{Cm} > \tau$, where τ is a reward threshold specified by the user.

Example 4: In Figure 6(b), we show a case for calculating the reward value of a candidate pattern {*A*,*B*,*C*} based on the instance-connecting status shown in Figure 6 (a). Taking the type pair {*B*,*C*} for example, we first implement the instance-projecting process based on the instance table of {*A*,*B*,*C*} and the instance pairs of {*B*,*C*} in *NT*. Based on Equation (5), we acquire the reward values of the instance pairs of {*B*,*C*} projected onto {*A*,*B*,*C*} and then acquire the reward value of {*B*,*C*} projected onto {*A*,*B*,*C*} according

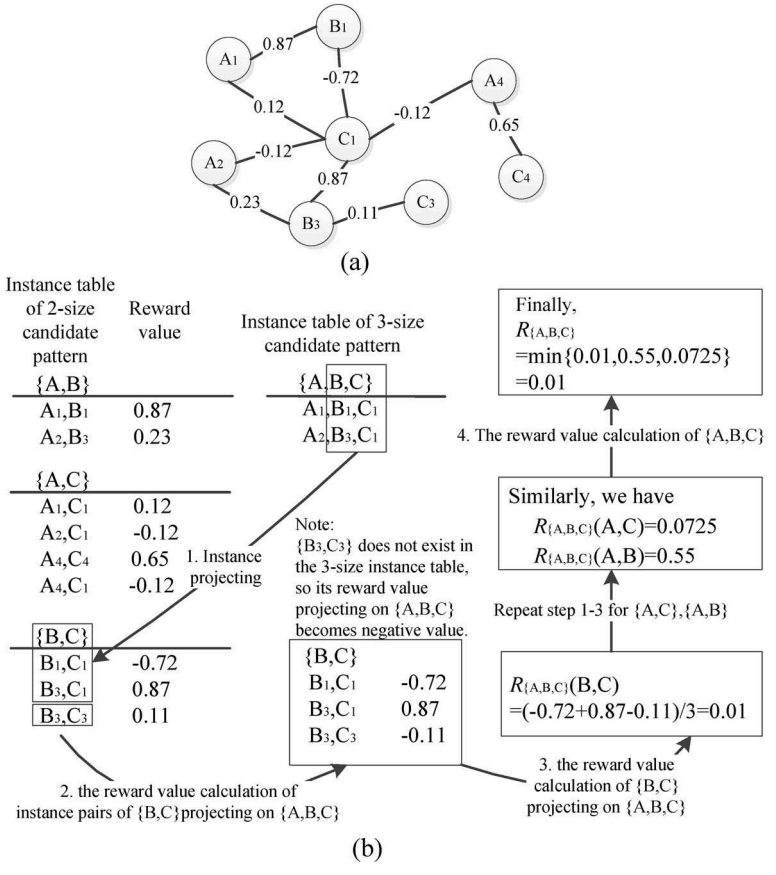

Figure 6. An example of the reward value calculation for a candidate pattern: (a) the instance-connecting statuses and their reward values and (b) the reward value calculation of {*A,B,C*}.

to Equation (6). After repeating the previous processes for the other-type pairs, such as {*A,C*} and {*A,B*}, we obtain three reward values for the three type pairs projected onto {*A, B,C*}: 0.01, 0.55 and 0.0725, respectively. Finally, we choose the minimum value as the reward value of {*A,B,C*}.

Example 4 shows that in contrast to the traditional prevalence index, the reward value mainly concerns the relationships between instances. Based on these definitions, we present a proof of the following hypothesis: the prevalent co-location pattern based on the reward-based verification has an anti-monotonicity property. This hypothesis is the basis of the pruning strategy used in the mining procedure described in Section 2.3(2).

Property 4: Suppose $Cm^* = \{e_1,\ldots,e_k,e_{k+1}\}$; then, $R_{Cm^*} \leq R_{Cm}$.

Proof: Assuming $R_{Cm} = R_{Cm}(e_i,e_j),(1 \leq i < j \leq k)$, we obtain $\pi_{(e_i,e_j)}(insT_{Cm*}) \subseteq \pi_{(e_i,e_j)}(insT_{Cm})$ based on the anti-monotonicity property of $insT_{Cm}$. Then, the following formula applies:

$$R_{Cm}(e_i, e_j) - R_{Cm*}(e_i, e_j) = \frac{\text{sum}\left\{R(o_x,o_y)+\text{abs}(R(o_x,o_y))|(o_x,o_y)\in\pi_{(e_i,e_j)}(insT_{Cm})\setminus\pi_{(e_i,e_j)}(insT_{Cm*})\right\}}{|NT(e_i,e_j)|} \geq 0,$$

where sum(·) is the summed value of the input collection and |·| is the cardinal number of the input set. Thus, we obtain $R_{Cm}(e_i, e_j) \geq R_{Cm*}(e_i, e_j) \geq R_{Cm*}$.

2.3.2. *The prevalent pattern-mining algorithm*

The pseudocode for mining all prevalent patterns is shown in Algorithm 1. However, the mining framework is not the main point of this paper; Algorithm 1 directly uses the approaches proposed by Yao *et al*. (2016) to generate the candidate maximal co-location patterns (line 3) and the prevalent maximal co-location patterns (lines 4–8), whose subsets are all prevalent co-locations.

Algorithm 1: Mining prevalent maximal co-location patterns
Input: *NT*, I_{far}, I_{close} and τ (reward threshold)
Output: *P* (a set storing all prevalent maximal co-location patterns)
1: $CP \leftarrow \emptyset$;
2: calculate all prevalent size-two co-location patterns by Equation (6);
3: use the CMCG algorithm (Yao *et al*. 2016) to generate all candidate maximal co-location patterns (*Cms*) and store them in set *CP*.
4: **for** each *Cm* in *CP*
5: {construct the instance table of C*m* by the CITC algorithm (Yao *et al*. 2016) and calculate its reward value R_{Cm} by Equation (7);
6: **if** $R_{Cm}>\tau$ {$P \leftarrow P \cup Cm$; remove *Cm* from *CP*;}
7: **else** {*Cm* in *CP* is replaced by its subsets;}
8: }

3. Computational complexity analysis

This section analyses the computational complexity of the ADMC algorithm based on its three procedures.

In Step 1, the worst complexity for constructing *NT* is $O(n\log_2(n))$, where *n* is the number of instances in space. The computational complexity of the undirected process is $O(mn)$, and that of the normalizing distances of all instance pairs in *NT* is $O(n_{NT})$, where *m* is the type number and n_{NT} is the number of instance pairs in *NT*.

In Step 2, we use traditional maximum likelihood estimation (MLE) with the complexity of $O(u^3)$, where *u* is the observation number. The computing cost of obtaining observations for the fitting process is $O(n_{NT})$. Because the observation number using the Freedman-Diaconis rule is $n_{NT}^{1/3}/2\text{IQR}(x)$, where IQR(*x*) is the interquartile range of the data, the computing cost of MLE in our algorithm is $O((n_{NT}^{1/3}/2\text{IQR}(x))^3) = O(n_{NT}/8\text{IQR}(x)^3)$. Finally, the computational complexity of Step 2 is $O(n_{NT}(1 + 1/8\text{IQR}(x)^3))$, which is approximately equal to $O(n_{NT})$.

In Step 3, the time complexity of computing the prevalent size-two co-location patterns is $O(n_{NT})$, and that of acquiring candidate maximal co-location patterns is $O(km3^{k/3})$, where *k* is the degeneracy of the prevalent size-2 co-locations. The calculation

of the prevalent co-location patterns is related to the number of candidate maximal co-location patterns. Previous studies have indicated that the long candidate verification process consumes the bulk of the execution time in maximal mining frameworks (Wang *et al*. 2009). We take only the longest candidate maximal co-location C_L for example. The worst complexity for constructing an instance tree for C_L is $O(\left|S_{C_L}^{Max}\right| * \log_2 n_L)$, where $\left|S_{C_L}^{Max}\right|$ is the maximum number of instances with a single type in C_L and n_L is element number of the set that contains connecting instances with types in C_L. The worst complexity of the reward calculation for C_L is $O(|C_L|^2{}_* n_L)$.

From the above analysis, the worst time cost of the ADMC algorithm is $O(n\log_2(n) + mn + 3n_{NT} + km3^{k/3} + t(\left|S_{C_L}^{Max}\right| * \log_2 n_L + |C_L|^2{}_* n_L)$, where t is the number of *Cm*s. Because $|C_L| < m < \left|S_{C_L}^{Max}\right| < n_L$ and $n < n_{NT} < mn$, the final time complexity is similar to $O(n\log_2(n) + mn + tn_L(\log_2 n_L + |C_L|^2)$.

4. Experiments and analysis

We used both synthetic and real datasets to design our experiments.

Synthetic datasets: Figure 7 shows the synthetic datasets produced by a synthetic data generator similar to that used by Yao *et al*. (2017). To emphasize the distance effects between instances and assure the stability of the mined results, we made some changes to the original generator. Notably, it can generate potential prevalent maximal co-locations based on predefined ranks. The design of the generator is shown in Figure 8. First, we predefined the maximal co-locations according to the predefined prevalence degree from high to low. These co-locations are stored in a permutation $P^* = \langle Pm_1^*, Pm_2^*, \ldots, Pm_n^* \rangle$, where n is the number of maximal co-locations. Second, we generate instance neighbourhoods for each maximal co-location $Pm_k^*(0<k<n)$ with input parameters r_k, d, and m_{clump_k}. The generating flow is described on the right side of Figure 8. The distances among the new appended instance neighbourhoods for the current pattern (Pm_k^* or $Pm_k^* \backslash T$ according to different conditions) are randomly distributed in $[r_k - d, r_k + d]$, and m_{clump_k} is the number of instance cliques for the current pattern. Third, we add n_{noise} random noise instances for each type to the spaces using the method proposed by Huang *et al*. (2004). Finally, we implement the comparative co-location algorithms and analyse the results. Table 1 shows the parameters of the two synthetic datasets (Cases S1 and S2).

Real dataset: The real dataset includes POIs in Beijing, China (2014). This data was also adopted by Yao *et al*. (2016, 2017) in their studies. The POIs include 288,486 items with 85 types (such as banks and restaurants), are registered legal entities with the Beijing Administration for Industry and Commerce, and do not include sensitive items. Each POI is associated with geographic coordinates and a category. We present the spatial distribution of the data in Figure 9(a) and preprocess these data using kernel density clustering (Silverman 1986) to define nine representative regions with different point densities, as shown in Figure 9(b). These regions are grouped into two series (Cases 1–4 and Cases 5–9) with densities that range from high to low. As shown in Table 2, the difference is that the areas of the regions in the former series are equivalent, whereas the regions in the latter series contain similar numbers of points.

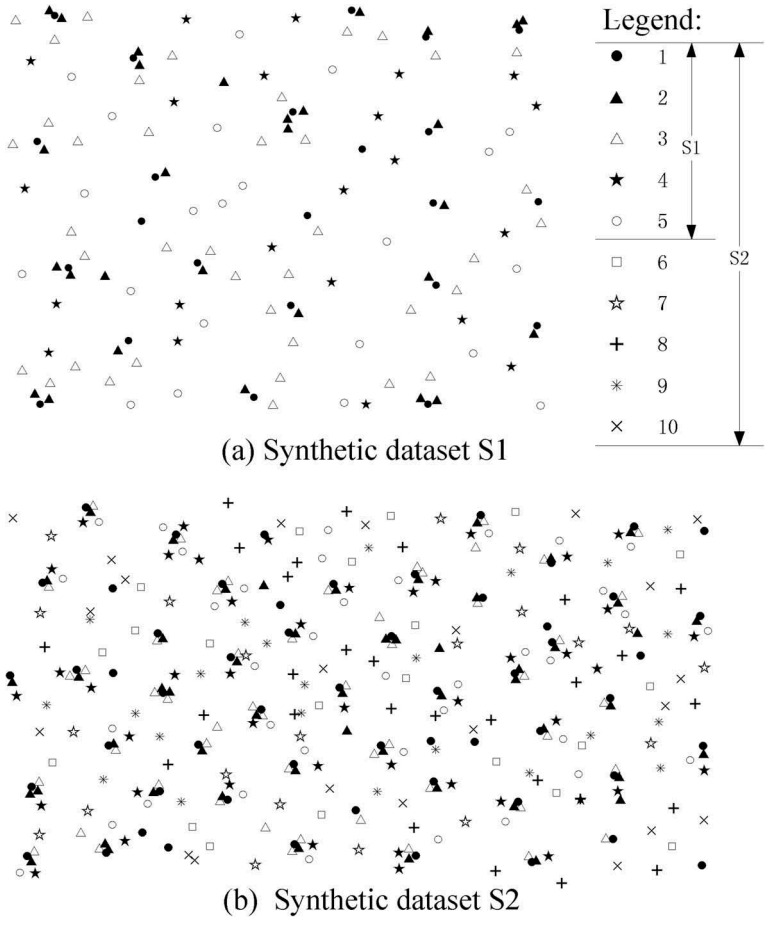

Figure 7. The synthetic datasets derived from the data generator.

We conducted three experiments using these datasets. All experiments were performed on a 3.3-GHz Centrino PC machine with 4 GB of main memory.

4.1. *Distance-separating parameter estimation results*

This experiment was designed to verify that the normalized distances in the size-two normalized instance table *NT* of different datasets tend to satisfy the GEV distribution. Using the method described in Section 2.2, we determined the appropriate distance-separating parameters for each real dataset and analysed the frequency distribution of the reward values of all type pairs based on real datasets (Cases 1–9).

Based on Cases 1–9, we constructed the frequency distribution histograms of the normalized distances in *NT* ($NorD_{NT}$) using the Freedman-Diaconis rule and discovered that most cases satisfy a type of skewed distribution with a high peak and a fat tail. The GEV distribution, normal distribution and T distribution statistically belong to this category due to their similar tracing patterns. To determine the best fitting function,

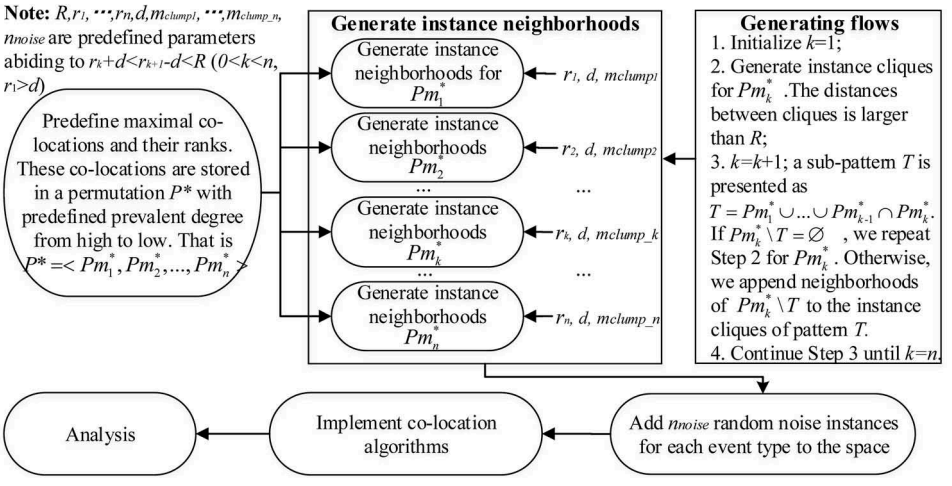

Figure 8. The design of the data generator.

Table 1. The parameters of the synthetic dataset generator for Case S1 and S2.

Parameter	Definition	Case S1	Case S2
Type number	–	5	10
Instance number	–	135	357
P^*	The permutation of the predefined co-locations	<{1,2},{1,3},{2,3}, {1,4},{1,5}>	<{1,2,3},{2,4,5},{5,7}, {8,9},{1,6},{2,10}>
R	The minimum distance between instance cliques of the predefined co-locations	20	20
$r_1, r_2, \ldots, r_n$	The distance between co-located instance pairs of the current pattern	2,6,10,14,18	2,4,6,8,10,12
d	Standard error of r	1	0.5
$m_{clump1,\ldots,clump\_n}$	The number of instance cliques for the current pattern	20,20,20,20,20	50,50,20,20,20
n_{noise}	The number of noise instances for each type	2	5

we used the fitting method proposed in Section 2.2 with a significance of 0.05 to estimate the parameters of these three functions. The coefficients of determination (R^2) are detailed in Table 3, and the graph in Figure 10 shows the fitting effects of the three aforementioned distribution functions for Case 1.

Table 3 demonstrates that, in all cases, the fitting effect of the GEV distribution is optimal relative to any of the other distributions. This phenomenon is plainly visible in Figure 10, which shows that the GEV fitting function is in superior concordance with the outline of the frequency histograms than the other two fitting functions.

This experimental result verified our inference in Section 2.1. Specifically, Cases 1, 2, 3, 5, 6, 7, 8 and 9 achieved high R^2 values ranging from 0.9972 to 0.9360. These values are extremely satisfactory for cross-sectional data. However, it is notable that the remaining case (Case 4) exhibited poor fitting effects and achieved an R^2 of 0.6397. For poor fitting cases, we used a compensatory strategy and chose the middle value of the histogram bin with the peak frequency as the final distance-separating parameters. In that case, a critical value of R^2 was needed. No definitive

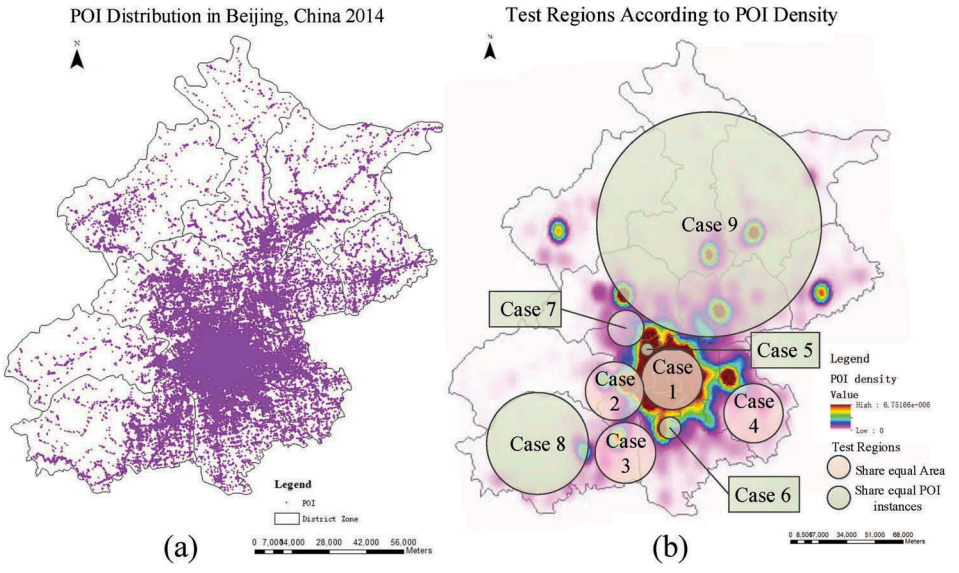

Figure 9. The real dataset.

Table 2. Statistical information for the real dataset.

Region label	Number of POIs	Area (km^2)	Number of types	Density of POIs (Number/km^2)
Case 1	106,629	250.00	80	427
Case 2	52,225	250.00	81	209
Case 3	7,811	250.00	82	31
Case 4	1,696	250.00	60	8
Case 5	12,006	16.54	71	726
Case 6	12,073	45.85	74	263
Case 7	12,085	78.32	81	154
Case 8	12,368	573.13	82	22
Case 9	12,348	2,259.51	80	5

Table 3. Fitting effects of three distribution functions for nine cases.

Region label	GEV fitting model (R^2)	Normal fitting model (R^2)	T fitting model (R^2)
Case 1	**0.9972**	0.6632	0.7392
Case 2	**0.9827**	0.5019	0.6119
Case 3	**0.9360**	0.3242	0.4267
Case 4	**0.6397**	0.3218	0.4108
Case 5	**0.9760**	0.4271	0.7462
Case 6	**0.9826**	0.3726	0.5728
Case 7	**0.9947**	0.4819	0.6078
Case 8	**0.9381**	0.4821	0.7928
Case 9	**0.9550**	0.6729	0.8329

Items in bold font are the largest values in each row.

standard exists for this approach; instead, it depends on the application context. For our application with human intervention, 0.8 is sufficient. Finally, for Cases 1, 2, 3, 5, 6, 7, 8 and 9, we employed the fitting strategy, and for Case 4, we employed the compensatory strategy (see Figure 11).

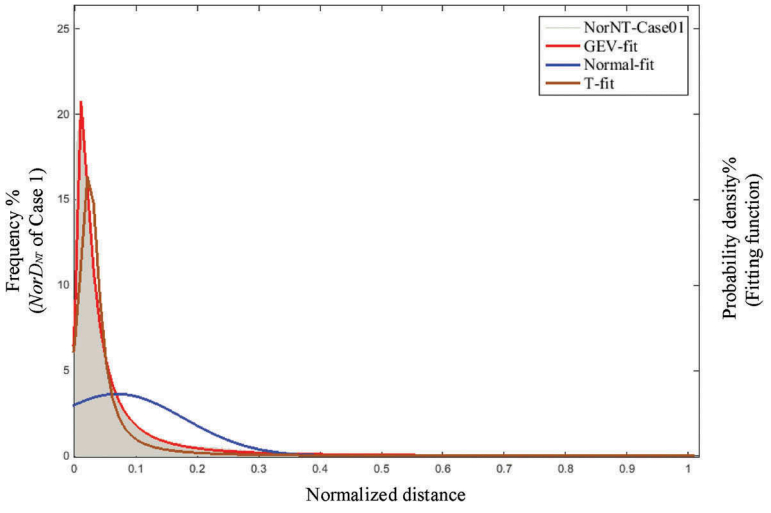

Figure 10. The fitting effects of the three functions for Case 1.

4.2. *Evaluation of the results*

From the density-adaptability and result significance, we evaluated our algorithm by using both synthetic and real datasets. We chose the DTBC algorithm for comparison because its input parameters are similar to ours.

4.2.1. *The density-adaptability evaluation of the results*

We evaluate the adaptability character of our algorithm by using a real dataset (Case 5). Table 4 shows the reward values of the top ten most prevalent patterns mined by the ADMC algorithm and their corresponding prevalence indices and rank orders calculated by the DTBC algorithm (Sundaram *et al*. 2012) based on Case5.

It shows that the prevalent co-location patterns mined by the two algorithms are different and that the reward values calculated by the ADMC algorithm are considerably larger than the prevalence indices acquired by the DTBC algorithm from the same dataset. To more deeply analyse the reward value distribution of the candidate patterns of different datasets, Figure 12 shows the grid images of reward values of all type pairs from Cases 3 and 4 (the graph series on the left side) and their corresponding frequency histograms (the graphs on the right side) based on the calculated distance-separating parameters. The grid value in the *i*th row and *j*th column is the reward value of the type pair (e_i, e_j). We can see from the graphs on the right that the reward value distributions reflect similar characteristics in cases with different densities. First, most reward values are close to 0, and only a few are close to −1 or 1. The frequency of the reward values decrease quickly from the $x = 0$ axis in both the positive and negative directions. Second, the reward values exhibit an axial-symmetric and fat-tailed distribution along the [−1, 1] axis that is similar to that of other datasets.

In Table 4, the prevalence index is theoretically between 0 and 1. However, the results of the DTBC algorithm shown in Table 4 are much lower than 1 and deliver an obscure picture of the prevalence index distribution. In this context, it is easier to set a reward

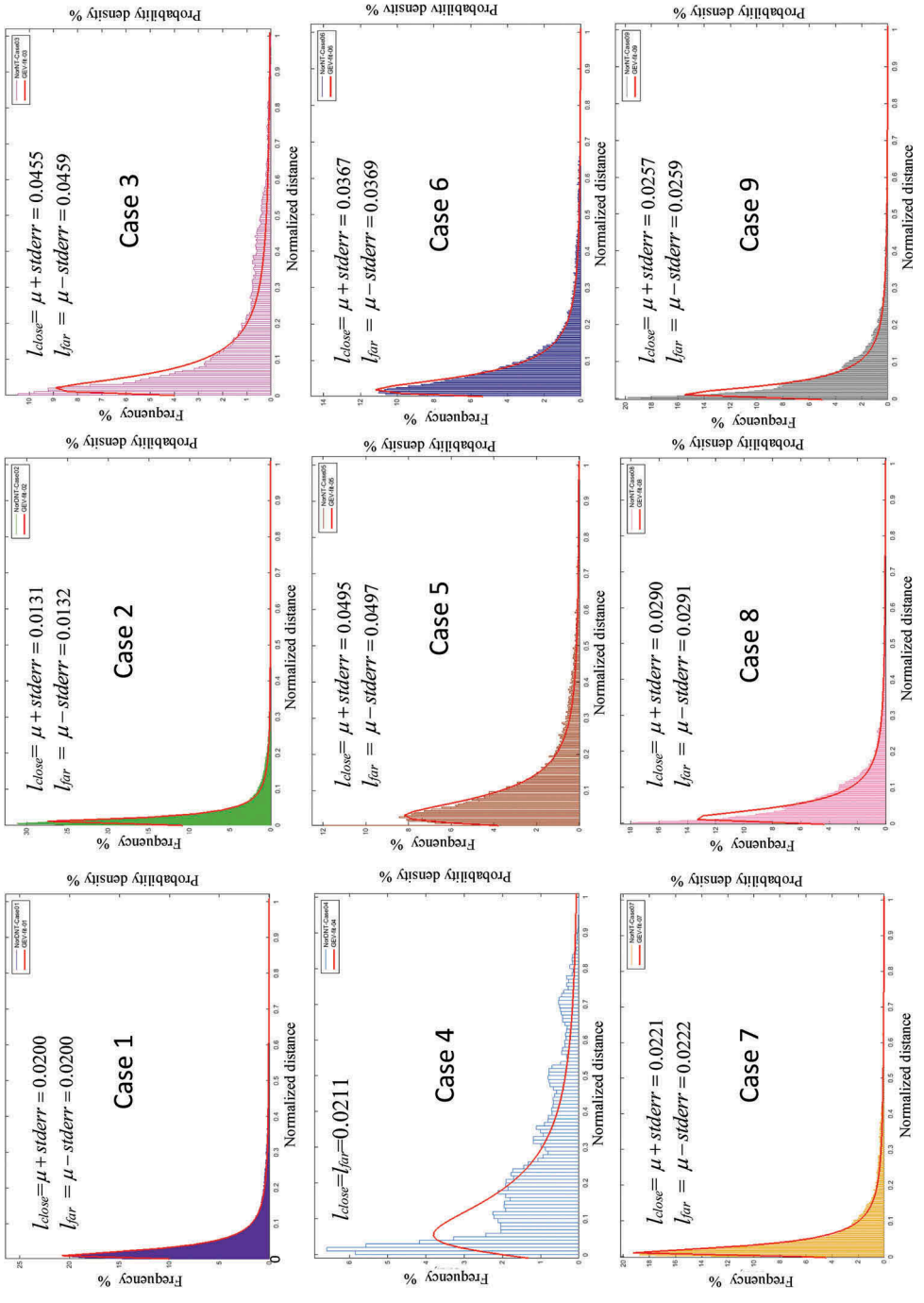

Figure 11. Distance-separating parameter extraction for Cases 1–9.

Table 4. Prevalent patterns mined by four algorithms from Case 5.

	ADMC		DTBC	
Co-location patterns	Reward value	Rank	Prevalence index	Rank
{business store, company}	0.815	1	0.223	2
{company, bank}	0.648	2	0.189	4
{beverage outlet, beauty salon}	0.561	3	0.185	5
{restaurant, snack bar}	0.555	4	0.093	21
{restaurant, beauty salon}	0.550	5	0.122	10
{snack bar, beauty salon}	0.546	6	0.115	12
{convenience store, clothing shop}	0.539	7	0.123	8
{business store, bank}	0.538	8	0.020	–
{restaurant, bank}	0.533	9	0.115	13
{restaurant, beverage outlet}	0.530	10	0.020	–

"–"Means the sorted position of the corresponding pattern exceeds 100.

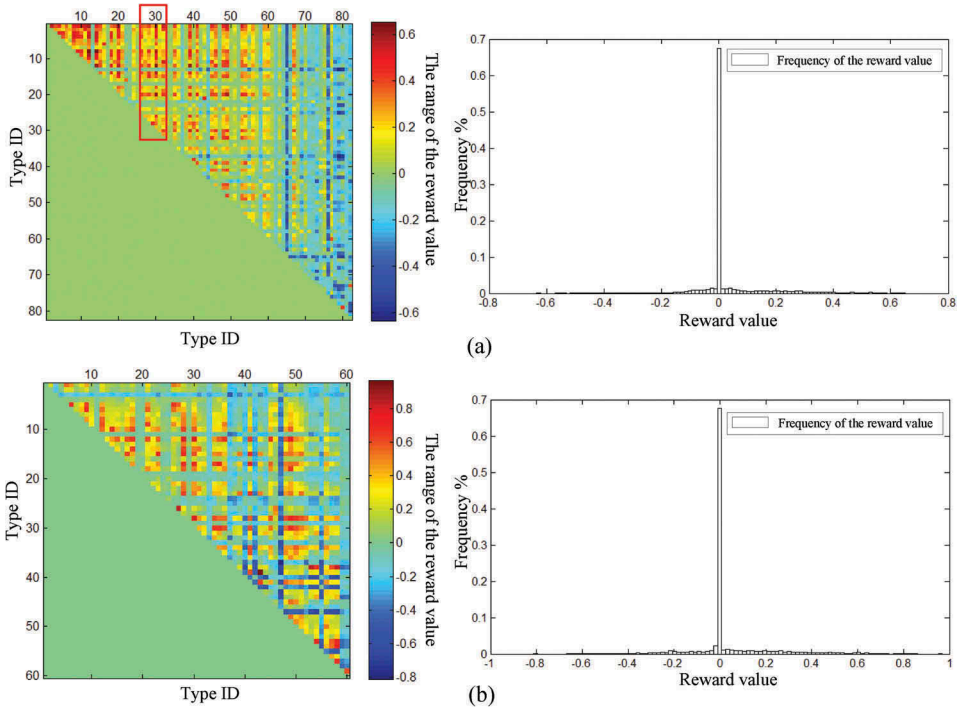

Figure 12. The grid images of reward values of all type pairs from Cases 3–4 and their corresponding frequency histograms based on the distance-separating parameters.

threshold in our algorithm than to set a prevalence threshold in the DTBC algorithm for different datasets, and the reward values of different datasets have similar distributions. Furthermore, the above characteristics of the reward values of type pairs show that the distance-separating estimation can adaptively distinguish the different effects of both close and far instances on patterns. That is, these effects are reflected by the axial symmetry of the frequency distribution of reward values. The values displayed by warm colours in the grid images and close to 1 imply that the corresponding type pairs have co-location relationships. In contrast, the values close to −1 imply that the corresponding type pairs are mutually exclusive. This finding agrees with the phenomena of the

natural world, as reported in previous studies, notably, co-location patterns are extreme relationships and are rare in nature.

Simultaneously, we find that the density of single-type instances in space influences the mining results. For example, when instances of a certain event type (such as instances of types 27–33 marked by the red rectangle in the grid image of Figure 12(a)) are abundant, they will likely have co-location relationships with other-type instances because the density of the single-type instances will influence the density of the entire dataset. Consequently, the estimated distance-separating parameters will also be disturbed. These statistical parameters reflect the comprehensive features of the dataset and will give higher weights to the patterns that contain types with abundant instances.

4.2.2. *The significant evaluation of the results*

4.2.2.1. *Evaluation on synthetic datasets.* We implemented the ADMC and DTBC algorithms using two synthetic datasets shown in Figure 7 and obtained the ranks of the mined results according to their prevalence degrees. The quality of the mined results can be evaluated by the following expression:

$$Precision = 1 - \frac{h_{result}}{h_{\max}} \times 100\%,$$

where h_{result} is the number of inversions (Margolius 2001) in the permutation of the predefined co-locations based on their calculated prevalence degrees and the predefined sequence of these co-locations and h_{max} is the maximum number of inversions in the probable permutation of the predefined co-locations. If the resulting ranks are the same as the predefined ranks, the precision is 100%.

Table 5 shows the detailed results of two comparative algorithms implemented for Cases S1 and S2. We can see from Table 5 that our algorithm performs well compared with the DTBC algorithm in precision for both synthetic datasets.

4.2.2.2. *Evaluation on real datasets.* For the real datasets, Table 6 shows the prevalent maximal patterns obtained using the ADMC algorithm and the same series of reward thresholds for Cases 1 and 5. These two datasets contain POIs from the core region of Beijing and include a multitude of businesses and entertainment venues. Geographically, the two datasets have similar morphologies. Notably, we can see that the prevalent co-locations for the two datasets using the same reward threshold mostly conform to logical patterns and the regional dataset characteristics. The prevalent co-locations mined from the two datasets with the same reward threshold also display high similarity (S_{RT}), which is calculated by the formula shown below. The similarity helps to mutually verify the significance of the results. A higher similarity value indicates higher significance for the mined results:

$$S_{RT} = \frac{(n_s/n_{c1} + n_s/n_{c2})}{2} \times 100\%,$$

where n_s is the number of the same prevalent co-locations in the two comparative datasets under the same reward threshold and n_{c1} and n_{c2} are the numbers of all the prevalent co-locations in the two datasets, respectively.

Table 5. The result evaluation of the two comparative algorithms.

Items		Case S1	Case S2
Predefined co-locations		<{1,2},{1,3},{2,3},{1,4},{1,5}>	<{1,2,3},{2,4,5},{5,7},{8,9},{1,6},{2,10}>
ADCM	Location ranks	<{1,2},{1,3},{2,3},{1,4},{1,5}>	<{1,2,3},{2,4,5},{8,9},{5,7},{1,6},{2,10}>
	Number of inversions	0	1
DTBC	Location ranks	<{1,2},{1,5},{2,3},{1,4},{1,3}>	<{1,2,3},{5,7},{8,9},{2,4,5},{1,6},{2,10}>
	Number of inversions	5	2
Maximum number of inversions		10	15
Precision of ADCM		**100%**	**93.3%**
Precision of DTBC		50%	86.7%

Items in bold font denote high precision.

Table 6. Prevalent maximal patterns with different reward thresholds based on Cases 1 and 5.

Reward threshold	Case 1	Case 5	Similarity
0.5	{parking lot, company}, {snack bar, convenience store, restaurant}, {snack bar, convenience store, beauty salon}, {restaurant, bank, company}, {business store, company}, {convenience store, restaurant, clothing shop}, {convenience store, beauty salon, clothing shop}	{convenience store, beauty salon, clothing shop}, {restaurant, company, bank}, {snack bar, beverage outlet, beauty salon}, {restaurant, snack bar, beauty salon}, {company, business store, bank}, {restaurant, beverage outlet}	54.1%
0.6	{business store, company}, {company, bank}	{business store, company}, {company, bank}	100%
0.7	–	{business store, company}	–
0.8	–	{business store, company}	–

To verify the significance of the final prevalent patterns, we chose six representative patterns and analysed the frequency distribution of the normalized distances of their corresponding instance pairs in Figure 13.

As shown in Figure 13, (a–c) are prevalent patterns in bold in Table 4. Specifically, Figure (a, b) illustrates the results for {business store, company} and {company, bank}, both of which have high prevalence indices and reward values in different algorithms. Figure (c) illustrates the results for {restaurant, beverage outlet}. These POIs are identified as prevalent patterns in the ADMC algorithm, but their sequence is significantly different in the DTBC algorithms. Figure (d) illustrates the results for {convenience store, pet clinic}, with a reward value of −0.035. We regard the distance-separating parameters of Case 5 acquired in Section 4.1 ($I_{close} = 0.0495$ and $I_{far} = 0.0497$) as the dividing lines of the frequency histograms in Figure 13. Figure (a–c) shows that the instance pairs on the left side of the red line outnumber those on the right side of the blue line. This result suggests that the close instances of the corresponding pattern are dominant among all instance pairs, and their patterns have co-locational meaning. Figure (d) shows the opposite situation–i.e. far instances predominate over close instances–such that the types in the corresponding patterns are exclusive in space. The phenomena reflected by these four patterns are consistent with their reward values. Thus, the results gained by the ADMC algorithm are both reliable and significant.

4.3. *Effects of the threshold settings*

We chose the same reward thresholds for Cases 1–6. Specifically, Figure 14 shows the time cost and number of results for the six datasets when the reward threshold was set

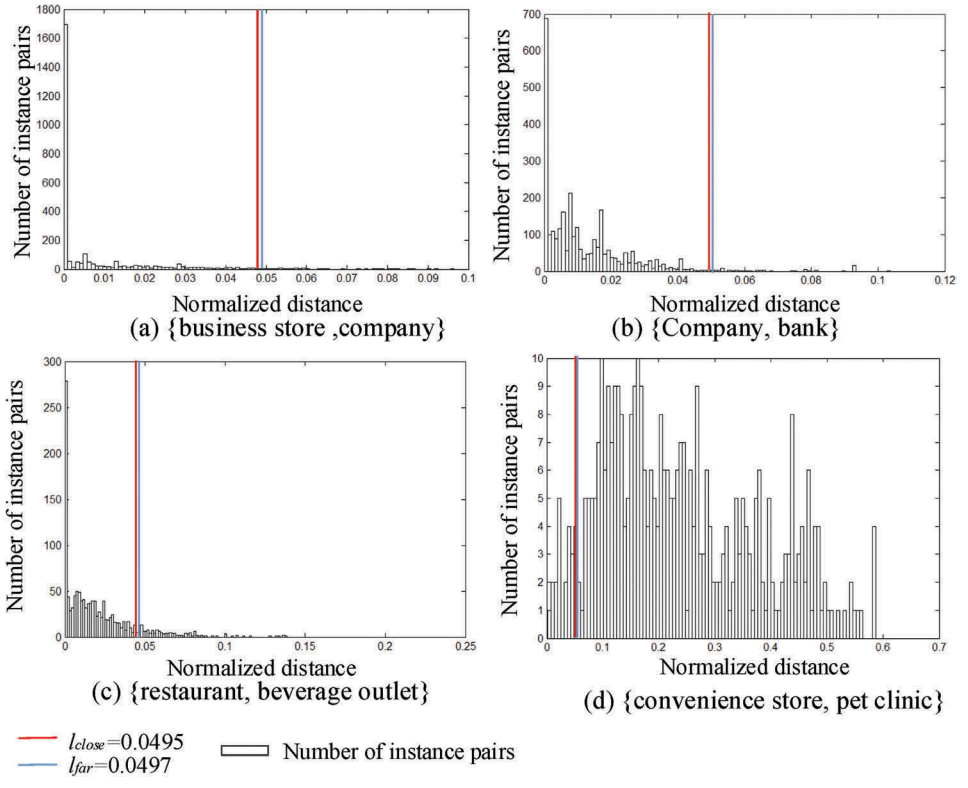

Figure 13. The normalized distance distributions of instance pairs in *NT* of representative patterns.

to 0.4–0.9 with an interval of 0.1. The left vertical axis represents the number of prevalent maximal patterns, while the right vertical axis represents the time cost of the execution. Note that the changes in the number of patterns and the execution time with the thresholds in the ADMC algorithm are similar to the changes observed in the DTBC algorithm. Thus, as the prevalence threshold decreases, the number of patterns and the time cost of the ADMC algorithm increases. This phenomenon verifies the anti-monotonicity property of our algorithm.

5. Conclusions

In this paper, we proposed the ADMC algorithm with three main features. First, Voronoi-based instance connections are implemented within the influence area of each instance. This process considers both close and far instances in space. Second, the reward-based verification for a prevalent pattern considers distance decay effects, which are neglected by general co-location algorithms. Third, the algorithm adapts to the density of the task data using a statistical method without requiring a predefined proximity standard, which avoids uncertainty. The experimental results show the advantages of our algorithm in terms of adaptability and significance.

However, our research and experiments were conducted using a general Voronoi diagram without considering the barriers or facilitating aspects of real space. Instead of

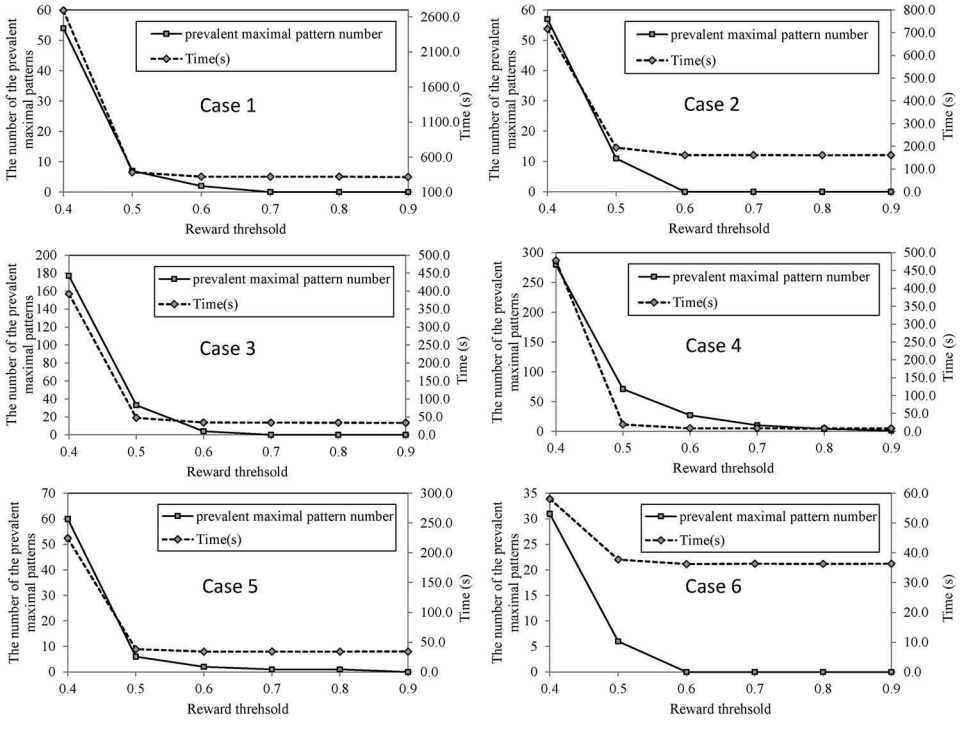

Figure 14. The time cost and the number of final results from Cases 1–6 with the reward threshold of 0.4–0.9.

Euclidean distance, a new index is required to measure the relationships between instances. Moreover, instances in the real world have complicated attributes, such as the instance level and importance, which were not considered in this study. Based on the results presented in this paper, we will explore new and improved solutions to address these problems in further studies.

Disclosure statement

No potential conflict of interest was reported by the authors.

Funding

This work was supported by the National Natural Science Foundation of China [Grant Number 41701438]; The National Science-Technology Support Plan Projects of China [Grant Number 2015BAJ02B00]; The Special Foundation of the Chief of the Institute of Remote Sensing and Digital Earth, Chinese Academy of Sciences [Grant Number Y6SJ2800CX].

ORCID

Xiaojing Yao http://orcid.org/0000-0001-9745-3150
Wenhao Yu http://orcid.org/0000-0003-1521-2674

References

Akbari, M., Samadzadegan, F., and Weibel, R., 2015. A generic regional spatio-temporal co-occurrence pattern mining model: a case study for air pollution. *Journal of Geographical Systems*, 17 (3), 249–274. doi:10.1007/s10109-015-0216-4

Aurenhammer, F., 1991. Voronoi diagrams – a survey of a fundamental geometric data structure. *ACM Computing Surveys*, 23 (3), 345–405. doi:10.1145/116873.116880

Burrough, P.A., 1986. *Principles of geographical information systems for land resources assessment*. Oxford: Clarendon and Publishing House.

Coles, S., *et al*., 2001. *An introduction to statistical modeling of extreme values*. Berlin: Springer-Verlag. ISBN 1-85233-459-2.

Deng, M., *et al*., 2017. Multi-level method for discovery of regional co-location patterns. *International Journal of Geographical Information Science*, 31 (9), 1846–1870. doi:10.1080/13658816.2017.1334890

Flouvat, F., *et al*., 2015. Domain-driven co-location mining. *Geoinformatica*, 19 (1), 147–183. doi:10.1007/s10707-014-0209-3

Freedman, D. and Diaconis, P., 1981. On the histogram as a density estimator: L2 theory. *Probability Theory and Related Fields*, 57 (4), 453–476.

Hosking, J.R.M., Wallis, J.R., and Wood, E.F., 1985. Maximum-likelihood estimation of the parameters of the generalized extreme-value distribution. *Journal of the Royal Statistical Society*, 34 (3), 301–310.

Huang, Y., Pei, J., and Xiong, H., 2006. Mining colocation patterns with rare events from spatial data sets. *Geology*, 10 (3), 239–260.

Huang, Y., Shekhar, S., and Xiong, H., 2004. Discovering colocation patterns from spatial datasets: a general approach. *IEEE Transactions on Knowledge and Data Engineering*, 16 (12), 1472–1485. doi:10.1109/TKDE.2004.90

Huang, Y., Zhang, P., and Zhang, C., 2008. On the relationships between clustering and spatial co-location pattern mining. *International Journal on Artificial Intelligence Tools*, 17 (01), 55–70. doi:10.1142/S0218213008003777

Leibovici, D.G., *et al*., 2014. Local and global spatio-temporal entropy indices based on distance-ratios and co-occurrences distributions. *International Journal of Geographical Information Science*, 28 (5), 1061–1084. doi:10.1080/13658816.2013.871284

Margolius, B.H., 2001. Permutations with inversions. *Journal of Integer Sequences*, 4 (2), 4–1.

Morimoto, Y., 2001. Mining frequent neighboring class sets in spatial databases. *In*: *The 7th ACM SIGKDD international conference on knowledge discovery and data mining*, 26–29 August 2001 San Francisco. New York: ACM, 353–358.

Nagelkerke, N.J.D., 1991. A note on a general definition of the coefficient of determination. *Biometrika*, 78 (3), 691–692. doi:10.1093/biomet/78.3.691

Qian, F., *et al*., 2012. Spatial co-location pattern discovery without thresholds. *Knowledge and Information Systems*, 33 (2), 419–445. doi:10.1007/s10115-012-0506-9

Qian, F., *et al*., 2014. Mining regional co-location patterns with kNNG. *Journal of Intelligent Information Systems*, 42 (3), 485–505. doi:10.1007/s10844-013-0280-5

Shekhar, S., and Huang, Y., 2001. Discovering spatial co-location patterns: a summary of results. *In*: C.S. Jensen, *et al*. eds. Advances in spatial and temporal databases, 12 –15 July 2001. Berlin: Springer, 236–256.

Sierra, R. and Stephens, C.R., 2012. Exploratory analysis of the interrelations between co-located boolean spatial features using network graphs. *International Journal of Geographical Information Science*, 26 (3), 441–468. doi:10.1080/13658816.2011.594799

Silverman, B.W., 1986. *Density estimation for statistics and data analysis*. New York: Chapman and Hall.

Sundaram, V.M., Thnagavelu, A., and Paneer, P., 2012. Discovering co-location patterns from spatial domain using a Delaunay approach. *Procedia Engineering*, 38, 2832–2845. doi:10.1016/j.proeng.2012.06.332

Tobler, W.R., 1970. A computer movie simulating urban growth in the Detroit region. *Economic Geography*, 46 (2), 234–240. doi:10.2307/143141

Wan, Y. and Zhou, J., 2008. KNFCOM-T: a k-nearest features-based co-location pattern mining algorithm for large spatial data sets by using T-trees. *International Journal of Business Intelligence and Data Mining*, 3 (4), 375–389. doi:10.1504/IJBIDM.2008.022735

Wan, Y., Zhou, J.G., and Bian, F.L. 2008. CODEM: a novel spatial co-location and de-location patterns mining algorithm. *In*: *The 5th international conference on fuzzy systems and knowledge discovery*, 5–27 August 2008 Jinan, China. Los Alamitos: IEEE, 576–580.

Wang, L., *et al*., 2009. An order-clique-based approach for mining maximal co-locations. *Information Sciences*, 179 (19), 3370–3382. doi:10.1016/j.ins.2009.05.023

Wang, L., Wu, P., and Chen, H., 2013. Finding probabilistic prevalent colocations in spatially uncertain data sets. *IEEE Transactions on Knowledge and Data Engineering*, 25 (4), 790–804. doi:10.1109/TKDE.2011.256

Xiong, H. *et al*., 2004. A framework for discovering co-location patterns in data sets with extended spatial objects. *In*: *The 4th SIAM international conference on data mining*, 22–24 April 2004, Lake Buena. Philadelphia, PA: SIAM Publications, Vols. 89, 78.

Yao, X., *et al*., 2016. A fast space-saving algorithm for maximal co-location pattern mining. *Expert Systems with Applications*, 63, 310–323. doi:10.1016/j.eswa.2016.07.007

Yao, X., *et al*., 2017. A co-location pattern-mining algorithm with a density-weighted distance thresholding consideration. *Information Sciences*, 396, 144–161. doi:10.1016/j.ins.2017.02.040

Yoo, J.S., Boulware, D., and Kimmey, D., 2014. A parallel spatial co-location mining algorithm based on MapReduce. *In*: *3rd IEEE international congress on big data,* 27 June-02 July 2014, Anchorage, AK. New York: IEEE, 25–31.

Yoo, J.S. and Shekhar, S. 2004. A partial join approach for mining co-location patterns. *In*: *The 12th ACM international symposium on advances in geographic information systems*, 12–13 November 2004 Washington, DC. New York: ACM, 241–249.

Yoo, J.S. and Shekhar, S., 2006. A joinless approach for mining spatial colocation patterns. *IEEE Transactions on Knowledge and Data Engineering*, 18 (10), 1323–1337. doi:10.1109/TKDE.2006.150

Yu, W., *et al*., 2017. Spatial co-location pattern mining of facility points-of-interest improved by network neighborhood and distance decay effects. *International Journal of Geographical Information Science*, 31 (2), 280–296. doi:10.1080/13658816.2016.1194423