EHUCM: An Efficient Algorithm for Mining High Utility Co-location Patterns from Spatial Datasets with Feature-specific Utilities

Yinqiao Li, Lizhen Wang(✉), Peizhong Yang, and Junyi Li

School of Information Sciences and Engineering, Yunnan University, Kunming, China
lzhwang@ynu.edu.cn

Abstract. High utility co-location pattern mining is still computationally expensive in terms of both runtime and memory consumption. In this paper, an efficient high utility co-location pattern mining algorithm, named EHUCM, is proposed to address this problem, which introduces the ideas of neighborhood materialization, participating objects of features and filtering unpromising candidate patterns to discover high utility co-location patterns more efficiently. To reduce the cost of dataset scanning, EHUCM pre-storing spatial relationships in a data structure to facilitate the search for potential candidate patterns. In addition, two effective pruning strategies are proposed in the EHUCM algorithm to improve the running overhead due to the utility measure not satisfying the downward closure property. Extensive experiments show that the EHUCM algorithm is 10 times or even 100 times faster than the traditional high utility co-location pattern mining algorithm.

Keywords: Spatial Data Mining · High utility co-location pattern · Pruning

1 Introduction

Co-location pattern mining in spatial datasets is a knowledge discovery problem. It aims at finding a set of spatial features whose objects are frequently located in the close geographic proximity [1]. This problem is useful in business [2], environmental science [3], biology [4] and many other fields. However, an important limitation of co-location pattern mining is that all features are considered equally important. This causes some important but non-prevalent patterns are missed [5]. To address this issue, Yang et al. first proposed the problem of high utility co-location pattern mining (HUCM) in spatial datasets with feature-specific utilities [6].

In contrast to co-location pattern mining, HUCM considers the case where each feature has a utility. Therefore, it can be used to discover sets of features with high utility, i.e. high utility co-location patterns (HUCPs). Whereas the utility of a co-location pattern may be lower, equal or higher than the utility of its subsets. Hence the pruning strategies based on the anti-monotonicity of prevalence in co-location pattern mining is not applicable to HUCM.

There have been several algorithms proposed for HUCM. Yang et al. [6] proposed the EPA algorithm, Wang et al. [7] proposed a base algorithm and three pruning strategies,

C. Strauss et al. (Eds.): DEXA 2021, LNCS 12923, pp. 185–191, 2021.
https://doi.org/10.1007/978-3-030-86472-9_17

and the min/max feature utility ratio algorithm were designed in [8]. Despite these research efforts, HUCM is still very expensive in terms of computation and memory. As they all mine HUCPs by generating and storing row instances of candidate patterns.

In this paper, we propose an efficient algorithm EHUCM (Efficient High Utility Co-location pattern Mining) which differs from past HUCM algorithms that generate all row instances of a candidate pattern c to compute pattern utility. It simply depends on the participating objects of each feature in c. Moreover, to reduce the cost of dataset scanning, we organize the spatial relationships in a data structure of feature-object neighbor tree that can be used to find potential candidate patterns.

2 Problem Definition

In a spatial dataset S, consider a set of spatial features F and a set of spatial objects O, each object o is represented with a tuple $<$ feature type, object id, location, utility $>$. If the distance between two objects $o, o' \in O$ is not greater than a given distance threshold d, the two objects satisfy **neighbor relationship** R. A **co-location pattern** c is a subset of spatial feature set F. The size of c is the number of features in c. A **row instance** RI of c represents a subset of objects, which includes an object of each feature in c and any two objects in RI satisfy the neighbor relationship, i.e., objects in RI form a clique. For a feature f in c, we say that an object o of f participates in c if at least one row instance of c involves o. The set of **participating objects** of f in c is noted as $Obj(f, c)$. Each feature $f_i \in F$ is associated with a positive number $v(f_i)$ called **external utility** of the feature that represents its importance. Correspondingly, the **internal utility** of f_i in c is the number of participating objects of f_i in c, denoted as $q(f_i, c) = |Obj(f_i, c)|$. Given a size k co-location pattern $c = \{f_1, f_2, \ldots, f_k\}$. The **feature utility** of a feature f_i in c is defined as $v(f_i, c) = v(f_i) \times q(f_i, c)$. The **pattern utility** of c is the sum of utility of each feature in c, defined as $u(c) = \sum_{f_i \in c} v(f_i, c)$.The pattern utility ratio of c is defined as $\lambda(c) = u(c)/U(S)$, where $U(S)$ is the total utility of S. c is a HUCP if and only if $\lambda(c) \geq$ *minutil*, where *minutil* is a user-specified minimum pattern utility ratio threshold.

Problem Definition. Given a spatial dataset S with feature-specific utilities, a distance threshold d and a utility ratio threshold *minutil*, the high utility co-location pattern mining is to find all high utility co-location patterns in S.

3 The EHUCM Algorithm

As stated above, the aim of this paper is to improve the efficiency of HUCM. In this section, we present the EHUCM algorithm.

3.1 The Search Space

Since the utility measure does not satisfy the downward closure property, all patterns in the spatial dataset need to be searched. Based on the idea that the combination of

features with the greater total utility of all objects in a spatial dataset is more likely to be a HUCP, we search the space in descending order of the total utility of all objects of each feature in a spatial dataset. Let $\succ$ be the descending order of the total utility of the feature in F.

Definition 1 (Extensible Feature Set of a Pattern). Given a co-location pattern c. Let $E(c)$ denote the set of features that can be used to extend c according to the depth-first search, so

$$E(c) = \{y | y \in F \wedge y \succ x, \forall x \in c\} \tag{1}$$

Definition 2 (Extension of a Pattern). Given a co-location pattern c. A pattern c' is a single-feature extension of c if $c' = c \cup \{z\}$ for $z \in E(c)$. Also, if $c' = c \cup Z$ where Z is a set of features $Z \in 2^{|E(c)|}$ with $Z \neq \emptyset$, c' is an extension of c.

3.2 Feature-Object Neighbor Tree (FONT)

The participating objects of each feature in a candidate pattern c are generated by scanning the dataset. To reduce the cost of dataset scanning, we adopt the idea of neighborhood materialization to organize spatial relationships in a feature-object neighbor tree (FONT) data structure. With the FONT we can easily find objects that have neighbor relationships with a given object.

Definition 3 (Object Neighbor Set). Given an object $o_i \in O$ with feature type o_i. feature $= f_i$, the object neighbor set of o_i is defined as

$$ONS(o_i) = \{o_j \big| o_j \in O \wedge R(o_i, o_j) \wedge o_j.\text{feature} \in F \setminus \{f_i\}\} \tag{2}$$

The object neighbor set of o_i includes objects that have neighbor relationships with o_i and the feature type of o_j is different from f_i.

Definition 4 (Feature-Object Neighbor Set). Given an object o_i and its object neighbor set $ONS(o_i)$, the feature-object neighbor set of o_i on feature f_i is defined as

$$FONS(o_i, f_j) = \{o_j \big| o_j \in ONS(o_i) \wedge (o_j.\text{feature} = f_j)\} \tag{3}$$

The feature-object neighbor set $FONS(o_i, f_j)$ is a subset of object neighbor set $ONS(o_i)$, and it includes all objects of f_j in $ONS(o_i)$.

Definition 5 (Feature-Object Neighbor Tree). Given the set of spatial features $F = \{f_1, f_2, \ldots, f_m\}$ and all neighbor relationships among spatial objects, the feature-object neighbor tree (FONT for short) is designed as follows. (1) The root of the FONT is marked as "null" and each feature is a child of the root. (2) The root of the feature f_i sub-tree is f_i, and the object neighbor set of all objects of f_i constitute the branch of f_i. Each branch records an object and its feature-object neighbor set.

3.3 Two Pruning Strategies

The search space of HUCPs has $2^{|F|}$-$|F|$-1 candidates, the number of candidates grows exponentially with the number of features. Therefore, in order to efficiently mine the HUCPs, two pruning strategies are proposed in this subsection, named Pattern Utility Loss Ratio and Extended Pattern Utility Ratio.

Definition 6 (Utility Loss Ratio). Given a co-location pattern c. Let $lu(c)$ represent the utility loss ratio of c, denoted as

$$lu(c) = \frac{\sum_{f_i \in c} tu(f_i) - u(c)}{U(S)} \tag{4}$$

where $tu(f_i)$ is the total utility of all objects of feature f_i in a spatial dataset S.

Lemma 1. If $lu(c) > 1\text{-}minutil$, all extended patterns of c cannot be high utility patterns.

Definition 7 (Extensible Objects of Extensible Feature of a Pattern). Given a co-location pattern c. The set of objects of feature f' in $E(c)$ is defined as

$$nei(f') = \left\{ o' \middle| o'.\text{feature} = f', \forall FONS(o', f) \neq \emptyset, f \in c \right\} \tag{5}$$

Definition 8 (Extensible Utility Ratio Upper-bound). Given a co-location pattern c. The extensible utility ratio upper-bound of c in a spatial dataset S is defined as

$$ub(c) = \frac{\sum_{f_i \in E(c)} v(f_i)|nei(f_i)|}{U(S)} \tag{6}$$

Definition 9 (Extended Pattern Utility Ratio). Given a co-location pattern c. The extended pattern utility ratio of c is defined as

$$eu(c) = \lambda(c) + ub(c) \tag{7}$$

Lemma 2. If $eu(c) < minutil$, all extended patterns of c cannot be high utility patterns.

3.4 The EHUCM Algorithm

Algorithm 1 scans the dataset once to generate the feature-object neighbor tree, and reorder the set of features F according to the descending order of the total utility of each feature in F. Then, initialize the candidate pattern c to the empty set.

Algorithm 1: The EHUCM algorithm

method:

1. $FONT$ = gen_feature_object_neighbor_tree(O, R)
2. The features in F be listed in descending order of $tu(f_i)$
3. Initialize $c = \varnothing$, $k = 1$
4. **Search(** k **)**

Algorithm 2: The Search procedure

method:

1. **while** $k \le F$.size **do**
2. $c = c \bigcup \{f_k\}$ //generates candidates
3. **if** c.size ≤ 1 **then** $k = k + 1$, **Search(** k **)**
4. **else**
5. $CS\text{-}HBS(c)$ = generate all participating objects per feature in c
6. **if** $CS\text{-}HBS(c) \neq \varnothing$ **then** calculate $\lambda(c)$
7. **if** $\lambda(c) \ge minutil$ **then** output c, $k = k + 1$, **Search(**k**)**
8. **else**
9. calculate $lu(c)$
10. **if** $lu(c) \le 1 - minutil$ **then** calculate $ub(c)$, $eu(c)$
11. **if** $eu(c) \ge minutil$ **then** $k = k + 1$, **Search(** k **)**
12. remove f_k from c, **Search(** k **)** //backtrack
13. **end**

The *Search* procedure (Algorithm 2) takes as a parameter the ordinal number of the k-th element of the set of features F. The procedure executes a loop that considers each single-feature extension of c of the form $c = c \cup \{f_k\}$, where f_k is the k-th element of F.

4 Experimental Evaluation

The experiments were conducted on a Windows 10 platform with an Intel Core i7-8700K CPU @3.70 GHz and 32 GB of RAM. We used two real spatial datasets, namely plants dataset of Three Parallel Rivers of Yunnan Protected Area and Beijing POI dataset. We compared the performance of the EHUCM algorithm with the EPA [6].

Influence of the Distance Threshold *d*. This experiment compares the running times of the two algorithms for mining HUCPs when the distance threshold is varied and the minimum utility ratio threshold parameter is fixed. Figure 1(a) and (b) shows the results. It can be observed that the EHUCM is always much faster than the EPA algorithm. For example, on Three Parallel Rivers with $minutil = 0.18$ and $d = 10900$, EHUCM is about 165 times faster than EPA.

Influence of the pattern utility ratio threshold *minutil*. The performance of algorithms was evaluated by fixing the value of the distance threshold in each dataset and varying the value of the minimum utility ratio threshold. The results are shown in Fig. 1(c) and (d). The running time of both algorithms decreases as *minutil* increases, and EHUCM

always runs in less time than EPA. For example, on Three Parallel Rivers with $d = 10500$ and $minutil = 0.16$, EHUCM is about 144 times faster than EPA.

The results of the EPA algorithm and the EHUCM algorithm in the above experiments are consistent.

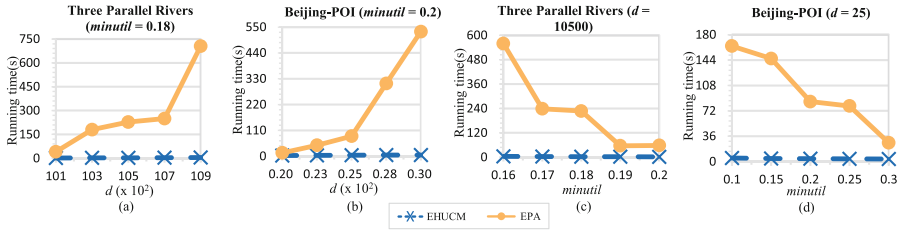

Fig. 1. Influence of the d and *minutil* on different datasets.

5 Conclusions

Since existing high utility co-location pattern mining algorithms are still very expensive in terms of both runtime and memory consumption. This paper proposes an efficient algorithm EHUCM for high utility co-location pattern mining. This algorithm differs from past algorithms that generate all row instances of candidate patterns to compute pattern utility. EHUCM simply depends on the participating objects of each feature in the candidate pattern. Because the utility measure fails to satisfy downward closure property, we propose two effective pruning strategies to prune the search space more efficiently. Extensive experimental results show that the EHUCM algorithm is efficient.

Acknowledgement. This work is supported by the National Natural Science Foundation of China (61966036, 62062066), the Project of Innovative Research Team of Yunnan Province (2018HC019).

References

1. Yang, P., Wang, L., Wang, X., Zhou, L.: SCPM-CR: a novel method for spatial co-location pattern mining with coupling relation consideration. IEEE Trans. Knowl. Data Eng. **4347**, 1–14 (2021). https://doi.org/10.1109/TKDE.2021.3060119
2. Yu, W.: Spatial co-location pattern mining for location-based services in road networks. Exp. Syst. Appl. **46**, 324–335 (2016). https://doi.org/10.1016/j.eswa.2015.10.010
3. Akbari, M., Samadzadegan, F., Weibel, R.: A generic regional spatio-temporal co-occurrence pattern mining model: a case study for air pollution. J. Geogr. Syst. **17**(3), 249–274 (2015). https://doi.org/10.1007/s10109-015-0216-4
4. Yoo, J.S., Bow, M.: A framework for generating condensed co-location sets from spatial databases. Intell. Data Anal. **23**, 333–355 (2019). https://doi.org/10.3233/IDA-173752
5. Truong, T., Duong, H., Le, B., Fournier-Viger, P.: Efficient algorithms for mining frequent high utility sequences with constraints. Inf. Sci. (Ny). **568**, 239–264 (2021)

6. Yang, S., Wang, L.: A framework for mining spatial high utility co-location patterns. In: The 12th International Conference on Fuzzy Systems and Knowledge Discovery (FSKD 2015), pp. 595–601. IEEE Press, New York (2015)
7. Wang, L., Jiang, W., Chen, H., Fang, Y.: Efficiently mining high utility co-location patterns from spatial data sets with instance-specific utilities. In: Candan, S., Chen, L., Pedersen, T.B., Chang, L., Hua, W. (eds.) DASFAA 2017. LNCS, vol. 10178, pp. 458–474. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-55699-4_28
8. Wang, X., Wang, L., et al.: Mining spatial high utility co-location patterns based on feature utility ratio. Chin. J. Comput. **42**, 1721–1738 (2019)