Expert Systems with Applications 37 (2010) 5795–5802

Contents lists available at ScienceDirect

Expert Systems with Applications

journal homepage: www.elsevier.com/locate/eswa

Algorithms for discovery of spatial co-orientation patterns from images ☆

Man-Kwan Shan *, Ling-Yin Wei

Department of Computer Science, National Chengchi University, Taiwan

ARTICLE INFO

Keywords:
Spatial co-orientation patterns
2D string
Iconic images
Spatial mining

ABSTRACT

Image mining is an important task to discover interesting and meaningful patterns form large image databases. In this paper, we introduce the spatial co-orientation patterns in image databases. Spatial co-orientation patterns refer to objects that frequently occur with the same spatial orientation, e.g. left, right, below, etc. among images. For example, an object P is frequently left to an object Q among images. We utilize the data structure, 2D string, to represent the spatial orientation of objects in an image. Two approaches, Apriori-based and pattern-growth approaches, are proposed for mining co-orientation patterns. An experimental evaluation with synthetic datasets shows the advantage and disadvantage between these two algorithms.

1. Introduction

Spatial data mining has attracted more and more attention, as the advances of information technologies. Finding meaningful patterns from spatial or image databases is important. There are three basic types of spatial relationships: distance, topological, and directional relationship (Ester, Frommelt, Kriegel, & Sander, 2000). Several studies have focused on mining spatial co-location patterns in which spatial objects are locating together closely. The spatial co-orientation pattern mining is concerned with the distance spatial relationship. Little work has been done on directional spatial relationships among the objects (Liu, Shekhar, & Chawla, 2000).

In this paper, we introduce the concept of spatial co-orientation pattern mining. Spatial co-orientation patterns refer to the spatial objects that occur frequently and collocate with the same orientation among each other. A typical but superstitious example of spatial co-orientation pattern mining is the Chinese Geomancy. As often as Asian design buildings or interior of home, the Chinese Geomancy is employed. In Chinese Geomancy, the directional relationship is an important factor. For example, Chinese Geomancy suggests the placement of a fish bowl in the north wall of a home to ward off bad influences.

Fig. 1 shows an image databases consisting of four iconic images. Each image contains several objects. Among three of these images, object D is north-western to object A, object A is south-western to object B, and object B is south-eastern to object D. Therefore, object D, A, and B constitute a spatial co-orientation pattern. Fig. 2 shows some of the spatial co-orientation patterns with occurrences no less than two.

Besides the concept of spatial co-orientation pattern mining, in this paper, we propose two algorithms to discover the spatial co-orientation patterns. Two algorithms, which are Apriori-based approach and pattern-growth approach, are proposed. In particular, to capture the spatial relationships among objects in the images, we employed the 2D string representation (Chang, Shi, & Yan, 1987) to represent symbolic pictures for Apriori-based approach and pattern-growth approach. The rest of the paper is organized as follows. Section 2 reviews the related work. In Section 3, we give the definition of the spatial co-orientation mining problem and present two algorithms for mining co-orientation patterns. The performance of two proposed algorithms is analyzed in Section 4. In Section 5, some applications of spatial co-orientation pattern mining are presented. Section 6 concludes this paper.

2. Related work

Koperski et al. proposed spatial association rules in geographic information databases. The work is finding the rules of topological relationships from spatial databases and its detail is described in Loperski and Han (1995). The main processes in this paper are filtering and refining. But it is expensive to scan the spatial databases to accomplish.

Spatial co-location mining (Huang, Shekhar, & Xiong, 2004; Shekhar & Huang, 2001; Yoo & Shekhar, 2004) is to discover frequent object classes occur together closely from spatial databases and so does (Morimoto, 2001). These researches focus on neighboring relation without dealing with relative direction among objects.

Hsu, Dai, and Lee (2003) proposed viewpoint patterns mining to find the relative distance and orientation invariant patterns. For

☆ Parts of the contents of this paper have been published in IEEE Proceedings of International Conference on Advanced Learning Technologies, 2006 and IEEE Proceedings of International Conference on System, Man, and Cybernetics, 2006.

* Corresponding author. Tel.: +886 2 29393091x67622; fax: +886 2 22341494.
E-mail address: mkshan@cs.nccu.edu.tw (M.-K. Shan).

0957-4174/$ - see front matter © 2010 Elsevier Ltd. All rights reserved.
doi:10.1016/j.eswa.2010.02.028

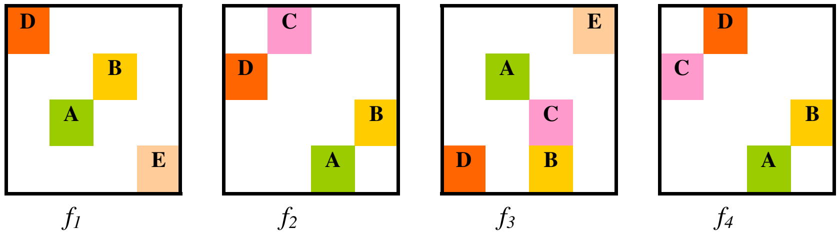

Fig. 1. An image database *SDB* of four symbolic pictures.

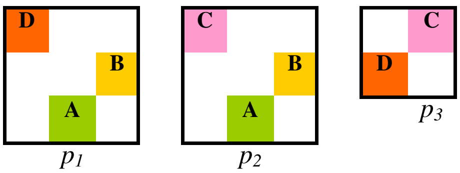

Fig. 2. Spatial co-orientation patterns of Fig. 1.

example, given two images in Fig. 3, the discovered viewpoint pattern is {A, B, C}.

In this Apriori-based approach, candidate (k + 1)-object pattern is generated by concatenating two frequent k-object patterns. However, in spatial cognition, human is more sensitive to the relative orientation than to distance. It is also useful to discover the patterns by considering only spatial orientation. If the distances between objects were not considered in viewpoint pattern mining, their approach won't work. This is because the transitive property is not hold for the relationship of spatial orientation. For example, in Fig. 3, the viewpoint pattern {A, B, D} will be discovered by their approach while in fact the spatial relationships between A and D in f_1 and f_2 are quite different. Moreover, in their work, each step of generating k-object patterns must scan databases and record the distance and orientation among all combinations of k objects. It takes too much time.

3. Co-orientation pattern mining

3.1. Problem definition

Definition 1. A symbolic picture f is a relation from $\{1,2,\ldots,m\} \times \{1,2,\ldots,n\}$ to V, where m, n is the size of the picture, and V is the set of objects contained in this picture.

Definition 2. A relative direction Rd is a relation from $V \times V$ to S, where V is the set of objects, S = {north, north-east, east, south-east, south, south-west, west, north-west}.

Definition 3. A symbolic picture f' is called a *subpicture* of a symbolic picture f iff $V' \subseteq V$, and $\forall o'_1, o'_2 \in V', o_1, o_2 \in V$, if $o'_1 = o_1$ and $o'_2 = o_2$, then $Rd(o'_1, o'_2) = Rd(o_1, o_2)$, where V' and V are the sets of objects in f' and f, respectively. The symbolic picture f is said to be the superpicture of the symbolic picture f' and to contain the symbolic picture f'.

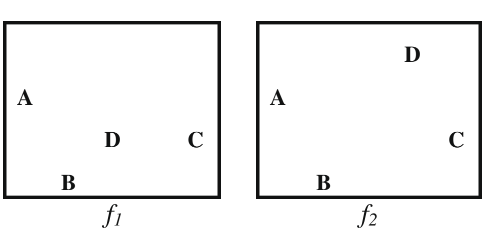

Fig. 3. Example of viewpoint patterns.

Definition 4. Given an image database $SDB = \{f_1, f_2, \ldots, f_n\}$ where each f_i is a symbolic picture. The support of a symbolic picture f is the percentage of symbolic pictures in *SDB* that contain the symbolic picture f. If the support of a symbolic picture f is greater than or equal to a given minimum support threshold, *minSup*, f is the spatial co-orientation pattern of *SDB*.

Example 1. Fig. 1 shows an image database of four symbolic pictures. The symbolic picture p_1, shown in Fig. 2, is a subpicture of the symbolic pictures f_1, f_2, and f_4, respectively. Therefore, the support of p_1 is 75%. If the minimum support threshold *minSup* is 50%, the symbolic pictures p_1, p_2, and p_4 are the maximal spatial co-orientation patterns.

3.2. 2D string representation

To discover the spatial co-orientation patterns from a database of symbolic pictures, we employ the 2D string representation to represent symbolic pictures. 2D string was originally proposed by Chang et al. (1987) for iconic indexing of image retrieval. In the 2D string approach, first, for each object in an image, the ortho-relation objects with respect to other objects are generated. The ortho-relation objects are used to characterize the relative spatial location with respect to other objects. Then, the reference points which are the central points of each ortho-relation objects constitute the symbolic picture. At last, the symbolic picture which preserves the relative spatial relationship is encoded as a 2D string.

Definition 5. An 1D string S, over a set of objects O, is represented as $S = o_1 o_2 \cdots o_m$ where $o_i \in O$ for $1 \leqslant i \leqslant m$, and m is the length of S. S is contained in another 1D string $S = o'_1 o'_2 \cdots o'_n$ if there exists integers $1 \leqslant i_1 < i_2 < \cdots < i_m \leqslant n$, $n \geqslant m$ such that $o_1 = o'_1, o_2 = o'_2, \ldots, o_m = o'_m$ (Chang et al., 1987).

Definition 6. Let V be a set of object symbols and R be the set {"=", "<"} which is used to specify the relative direction among objects. The symbol "=" denotes the "at the same spatial location", the symbol "<" denotes the "left–right or below–above spatial relationship." A *2D string* (S_x, S_y) over V is defined as $(o_1 r_{1x} o_2 r_{2x} \cdots r_{(n-1)x} o_n,\ o_{p(1)} r_{1y} o_{p(2)} r_{2y} \cdots r_{(n-1)y} o_{p(n)})$, where $o_1 o_2 \cdots o_n$ and $o_{p(1)} o_{p(2)} \cdots o_{p(n)}$ are 1D strings over V, p is a permutation function from $\{1,\ldots,n\}$ to $\{1,\ldots,n\}$, $r_{1x} r_{2x} \cdots r_{(n-1)x}$ and $r_{1y} r_{2y} \cdots r_{(n-1)y}$ are both 1D strings over R and n is the length of (S_x, S_y). A 2D string with n objects is called the size-n 2D string (Chang et al., 1987).

Example 2. The 2D string representations for the symbolic pictures f_1, f_2, f_3 and f_4 in Fig. 1 are (S_{1x}, S_{1y}) = (D < A < B < E, E < A < B < D), (S_{2x}, S_{2y}) = (D < C < A < B, A < B < D < C), (S_{3x}, S_{3y}) = (D < A < B = C < E, D = B < C < A < E) and (S_{4x}, S_{4y}) = (C < D < A < B, A < B < C < D), respectively.

Note that, to ensure the unique 2D string representation (S_x, S_y) of a symbolic picture, if two objects o_i, o_j are at the same spatial location along the vertical axis, the relative orders of o_i, o_j in S_y

should be of the same as those in S_x. For example, rather than (D < A < B = C < E, B = D < C < A < E), the symbolc picture f_3 is represented as (D < A < B = C < E, D = B < C < A < E).

Definition 7. A string S' is a *1D subsequence* of a string S, if (1) S' is contained in S, and (2) if $a'w'b'$ is a substring of S', a' matches a in S and b' matches b in S, then $r(b) - r(a) \geqslant r(b') - r(a')$ where $r(x)$, the *rank* of symbol x, is defined as one plus the number of "<" preceding this symbol x (Chang et al., 1987).

Definition 8. A 2D string (S'_x, S'_y) is a *2D subsequence* of a 2D string (S_x, S_y), denoted as $(S'_x, S'_y) \subseteq (S_x, S_y)$, if S'_x is a 1D subsequence of S_x and S'_y is a 1D subsequence of S_y. The 2D string (S_x, S_y) is said to *contain* the 2D string (S'_x, S'_y) (Chang et al., 1987).

Definition 9. Given a 2D string database $SDB_{2D} = \{(S_{1x}, S_{1y}), (S_{2x}, S_{2y}), \ldots, (S_{nx}, S_{ny})\}$. The *support* of a 2D string (S'_x, S'_y) is the percentage of the 2D strings in SDB_{2D} that contain (S'_x, S'_y). (S'_x, S'_y) is *frequent* if its support is greater than or equal to a given minimum support threshold *minSup*.

Example 3. Given the database of four 2D strings in Example 2, the support of the 2D string (D < A < B, A < B < D) is 75%. It is a 2D subsequence of the 2D strings (S_{1x}, S_{1y}), (S_{2x}, S_{2y}), and (S_{4x}, S_{4y}), while it is not a 2D subsequence of the 2D string (S_{3x}, S_{3y}). If *minSup* equals 50%, then (D < A < B, A < B < D) is one of the frequent.2D strings.

Given a database of symbolic pictures, the problem of spatial co-orientation pattern mining thus becomes the discovery of the frequent 2D strings among a database of 2D strings. Each such frequent 2D string is a spatial co-orientation pattern.

3.3. Apriori-based algorithm

In this section, we propose a Apriori-based algorithm to discover the spatial co-orientation patterns. The Apriori-based approach utilizes the downward closure property that every subpattern of a frequent pattern must be frequent. This property is a foundation for generation of correct and complete set of frequent patterns.

Our proposed Apriori-based algorithm starts by finding all frequent objects, i.e. size-1 2D strings. Then, it enters the main iteration phase. In each iteration, namely, the kth iteration, the candidate size-$(k+1)$ 2D strings are generated by joining two frequent size-k 2D strings. Then for each candidate size-$(k+1)$ 2D string, the *support* are counted and those candidates that satisfy the *support* constraint are preserved as frequent size-$(k+1)$ 2D strings.

The above sketch of the proposed Apriori-based algorithm is quite similar to that of frequent itemset or sequential pattern mining. However, the joining operation of 2D strings is more complicated. Especially, it is not as intuitive as joining of symbolic pictures. The following definition gives the condition for joining of two 2D strings.

Definition 10. Two size-k 2D strings $(S_x, S_y) = (o_1 r_{1x} o_2 r_{2x} \cdots o_{(k-1)} r_{(k-1)x} o_k,\ o_{p(1)} r_{1y} o_{p(2)} r_{2y} \cdots r_{(k-1)y} o_{p(k)})$ and $(S'_x, S'_y) = (o'_1 r'_{1x} o'_2 r'_{2x} \cdots o'_{(k-1)} r'_{(k-1)x} o'_k, o'_{p'(1)} r'_{1y} o'_{p'(2)} r'_{2y} \cdots r'_{(k-1)y} o'_{p'(k)})$ are joinable if

(1) $\forall i, 1 \leqslant i \leqslant k-1, o_i = o'_i$ and $\forall i, 1 \leqslant i \leqslant k-2 r_{ix} = r'_{ix}$.
(2) $\exists$ a monotonic function g from $\{1, 2, \ldots, n\}$ to $\{1, 2, \ldots, n\}$ such that $g(i) = j$ only if $p(i) \neq n$ and $p'(j) \neq n$ and $o_{p(i)} = o_{p'(j)}$ and $r_{(i-1)y} = r'_{(j-1)y}$.

This definition states that two size-k 2D strings $(o_1 r_{1x} o_2 \cdots o_{(k-1)} r_{(k-1)x} o_k,\ o_{p(1)} r_{1y} o_{p(2)} \cdots r_{(k-1)y} o_{p(k)})$ and $(o'_1 r'_{1x} o'_2 \cdots o'_{(k-1)} r'_{(k-1)x} o'_k, o'_{p'(1)} r'_{(1y)} o'_{p\prime(2)} \cdots r'_{(k-1)y} o'_{p'(k)})$ are joinable if they share the same prefix $o_1 r_{1x} o_2 r_{2x} \cdots o_{(k-1)}$ along the horizontal axis and the relative orders of the $(k-1)$ objects $o_1, o_2, \ldots, o_{(k-1)}$ are of the same along the vertical axis.

Example 4. The 2D strings (A < B < C < D, C < D < A < B) and (A < B < C < E, C < A < E < B) are joinable. Both have the same size-3 prefix "A < B < C" along the horizontal axis. Moreover, the relative orders of objects, A, B, C are of the same along the vertical axis. In other words, there exists a monotonic function such that $g(1) = 1$, $g(3) = 2$, and $g(4) = 4$ where $o_{p(1)} = o_{p'(1)} =$ 'C', $o_{p(3)} = o_{p'(2)} =$ 'A', and $o_{p(4)} = o_{p'(4)} =$ 'B'.

Different from the joining operation of frequent itemset mining in which two frequent size-k itemsets produce a unique size-$(k+1)$ itemset, the joining operation of two frequent size-k 2D strings may produce more than one distinct size-$(k+1)$ candidate 2D strings. For example, joining of two frequent 2D strings in Example 4 will generate three distinct size-5 candidate 2D strings which are shown in Fig. 4(b).

The detailed algorithm for the candidate generation is shown in Fig. 5. Given two joinable frequent size-k 2D strings $(S_x, S_y) = (o_1 r_{1x} o_2 \cdots o_{(k-1)} r_{(k-1)x} o_k,\ o_{p(1)} r_{1y} o_{p(2)} \cdots r_{(k-1)y} o_{p(k)})$ and $(T_x, T_y) = (o_1 r_{1x} o_2 \cdots o_{(k-1)} r'_{(k-1)x} o'_k, o'_{p'(1)} r'_{1y} o'_{p'(2)} r'_{2y} \cdots r'_{(k-1)y} o'_{p\prime(k)})$ the number of generated candidates depends on the spatial location of the two disagreed objects o_k and o'_k. This algorithm first generates the possible cases, by calling the function subGen_Candidate, along the horizontal axis and the vertical axis, respectively. Then the combinations of candidates are generated.

For the horizontal axis, if both $r_{(k-1)x}$ and $r'_{(k-1)x}$ are '<', then there are three possible cases. Otherwise, there is only one case. Fig. 4(a) and (b) is an example of the former while Fig. 4(c) is that of the latter. For the vertical axis, if these two unmatched objects, o_k and o'_k, locate in the same location among the matched objects, then there are three possible cases. Otherwise, there is only one case. Fig. 4(a) is an example of the former while Fig. 4(b) and (c) are those of the latter. In Fig. 4(a), along the vertical axis, objects D and E both locate in the slot between objects C and A. Therefore, there are three possible cases along the vertical axis. Finally, the last step of the algorithm is to prune the exception if objects are not allowed to be at the same cell. An example is the last candidate of Fig. 4(a).

3.4. Pattern-growth algorithm

In this section, we propose another efficient algorithm, which is based on the pattern-growth approach, for mining spatial co-orientation patterns. Pei et al. (2004) have proposed the pattern-growth based algorithm, PrefixSpan, for mining sequential patterns from a set of customer sequences. In this approach, the database of customer sequences is recursively projected into a set of projected databases. The discovered sequential patterns are grown in each projected database by exploring local frequent elements (Pei et al., 2004). With the concept similar to PrefixSpan, we propose the pattern-growth approach for mining frequent 2D strings.

Definition 11. Given a 2D string $(S_x, S_y) = (o_1 r_{1x} o_2 r_{2x} \cdots o_{(k-1)} r_{(k-1)x} o_k,\ o_{p(1)} r_{1y} o_{p(2)} r_{2y} \cdots r_{(k-1)y} o_{p(k)})$. A string $S' = (o'_1 r'_1 o'_2 r'_2 \cdots o'_{(h-1)} r'_{(h-1)} o'_h), h \leqslant k$, is called a *X-prefix* of (S_x, S_y), if and only if S' is an 1D subsequence of S_x .

Definition 12. Given a 2D string $(S_x, S_y) = (o_1 r_{1x} o_2 r_{2x} \cdots o_{(k-1)} r_{(k-1)x} o_k,\ o_{p(1)} r_{1y} o_{p(2)} r_{2y} \cdots r_{(k-1)y} o_{p(k)})$. Let $S' = (o_1 r_{1x} o_2 r_{2x} \cdots o_{(h-1)} r_{(h-1)x} o_h)$, $h \leqslant k$, be an X-prefix of (S_x, S_y). A string $S'' = (r_{hx} o_{(h+1)} r_{(h+1)x} o_{(h+2)} r_{(h+2)x} \cdots o_{(k-1)} r_{(k-1)x} o_k)$ is called an *X-suffix* of (S_x, S_y) with respect to X-prefix S'. A 2D string (S'_x, S'_y) is called a *suffix* of (S_x, S_y) with respect to X-prefix S' if

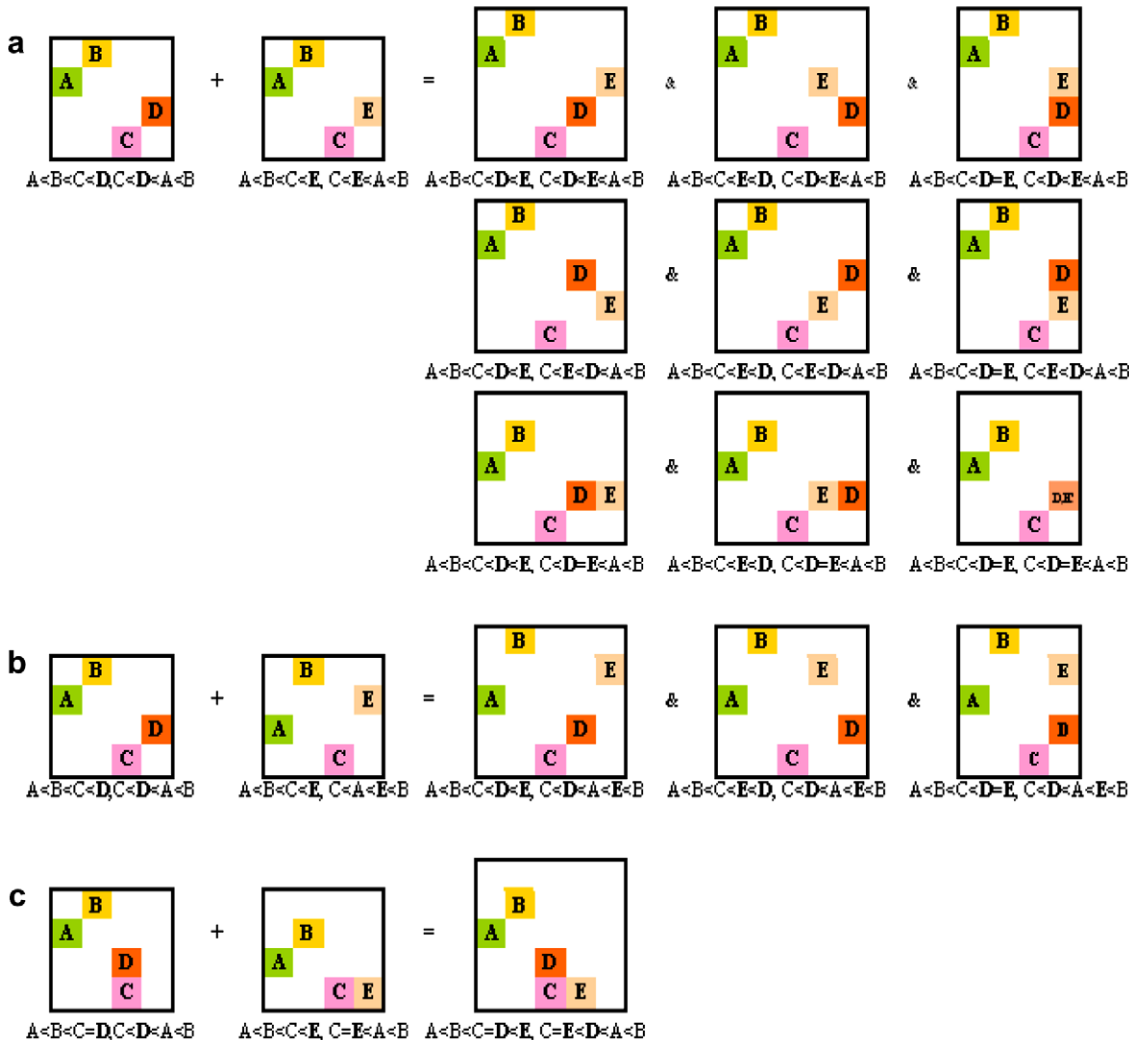

Fig. 4. Examples of candidate generation.

(1) S'_x is a X-suffix of (S_x, S_y) with respect to S' and
(2) $S'_y = (o'_{p(1)} r'_{1y} o'_{p(2)} r'_{2y} \cdots r'_{(k-1)y} o'_{p(k)})$ is a 1D subsequence of S_y where $o'_{p(i)}$ appears in X-prefix or X-suffix.

Definition 13. Let (S_x, S_y) be a spatial co-orientation pattern in *SDB*. The S_x*-projected database* collects all suffixes of 2D strings in *SDB* with regards to an X-prefix S_x.

Example 5. Given an image database *SDB* in Fig. 1. Let *minSup* be 75%. (D, D), (A, A), (B, B) and (C, C) are size-1 spatial co-orientation patterns. The D-projected databases are (<A < B, A < B < D), (<C < A < B, A < B < D < C), (<A < B = C, D = B < C < A), and (<A < B, A < B < D).

Fig. 6 shows the pattern-growth algorithm. In Fig. 6, the preprocess function transforms symbolic picture databases to 2D string databases SDB_{2D}. We scan SDB_{2D} to generate size-1 spatial co-orientation patterns FP_1. The build_projected_database function generates a projected database according to SDB_{2D} and FP_1. From step 4 to 11, we generate spatial co-orientation patterns with respect to each pattern in FP_1 and it is shown in Fig. 7. In Fig. 7, the Gen_Pattern function generates 2D strings from a co-orientation pattern *P* by scanning the projected database *PDB* and adding an object e occurring in FP_1 but not in *P* to *P*. The detail of Gen_Pattern is shown in Fig. 8. From step 3 to 10, it is continuously recursive to find co-orientation patterns with respect to each pattern *s* if *s* is frequent. The process of update_projected_database in Fig. 7 scans SDB_{2D} to generate the projected database of *s* to *PDB*.

Example 6. Given 2D string database SDB_{2D} in Fig. 1, Fig. 9 gives the entire process of miming. The order of steps in Fig. 9 is 1, 2, 3, ..., 6.

To implement the projection of database of 2D strings, similar to the pattern-growth approach for sequential pattern mining, we also use the concept of pseudo-projection without generating large physical projected database. The pseudo-projection reduces the size of projected database and it can fit in main memory. Our algorithm adopts this technique to save both space and time. Similarly, \$ indicates the X-prefix occurs in the current 2D string but its projected X-suffix is empty and $\emptyset$ indicates the X-prefix does not occur in the current 2D string. Otherwise, the horizontal axis of each pseudo-projection denotes the offset of the rightmost object of X-prefix. But the vertical axis of each pseudo-projection denotes the rank of each object of X-prefix orderly.

Example 7. Given a 2D string database SDB_{2D} in Table 1. Let *minSup* be 75%. The projected databases for X-prefixes (R), (P), (O), (R < O), (R < P) and (O < P) are shown in Table 1. One of the pseudo-projected databases with regard to the X-prefix, (R < O), is <4, 3 2>,

```
Input: Two frequent size-k 2D strings, (S_x, S_y) and (T_x, T_y)
Output: A set of candidate size-(k+1) 2D strings
Let (S_x, S_y)=(o_1 r_1x o_2 r_2x…o_(k-1) r_(k-1)x o_k, o_p(1) r_1y o_p(2) r_2y…r_(k-1)y o_p(k))
        (T_x, T_y)=(o'_1 r'_1x o'_2 r'_2x…o'_(k-1) r'_(k-1)x o'_k, o'_p'(1) r'_1y o'_p'(2) r'_2y…r'_(k-1)y o'_p'(k))
   If (S_x, S_y) and (T_x, T_y) are joinable Then {
        i=k;    j=k;
        C_(k+1)x = subGen_Candidate(S_x, T_x, i, j)
        Let i, j, be the index such that P(i)=k, p'(j)=k
        C_(k+1)y = subGen_Candidate(S_y, T_y, i, j)
        For each C_x in C_(k+1)x
            For each C_y in C_(k+1)y
                Insert (C_x, C_y) into C_(k+1)
         Prune the exception                }
Return C_(k+1)
subGen_Candidate(S, T, i, j)
Let S=o_1 r_1 o_2 r_2…o_(k-1) r_(k-1) o_k, T=o'_1 r'_1 o'_2 r'_2…o'_(k-1) r'_(k-1) o'_k;
If (i = j) then {
   switch (r_(i-1), r'_(j-1))
      case (r_(i-1) = '<' and r'_(j-1) = '<') then {
            Insert (o_1 r_1…<o_i<o'_j…r_(k-1) o_k) into C_(k+1);
            Insert (o_1 r_1…<o'_j<o_i…r_(k-1) o_k) into C_(k+1);
            Insert (o_1 r_1…<o_i=o'_j…r_(k-1) o_k) into C_(k+1); }
      case (r_(i-1) = '=' and r'_(j-1) = '<')
            Insert (o_1 r_1…r_(i-1) o_i r'_(j-1) o'_j…r_(k-1) o_k) into C_(k+1);
      case (r_(i-1) = '<' and r'_(j-1) = '=')
            Insert (o_1 r_1…r'_(j-1) o'_j r_(i-1) o_i…r_(k-1) o_k)into C_(k+1);
      else Insert (o_1 r_1…r_(i-1) o_i r'_(j-1) o'_j…r_(k-1) o_k)into C_(k+1); }
else if (i < j)
   Insert (o_1 r_1…r_(i-1) o_i…r'_(j-1) o'_j…r_(k-1) o_k)into C_(k+1);
else if (i > j)
   Insert (o_1 r_1…r'_(j-1) o'_j…r_(i-1) o_i…r_(k-1) o_k);
Return C_(k+1);
```

Fig. 5. Algorithm of candidate generation.

```
Input: A symbolic picture database SDB and a support threshold minSup
Output: A set of spatial co-orientation patterns FP
Variables: k: co-orientation pattern size
           FP_k: set of size-k spatial co-orientation pattern
SDB_2D=preprocess(SDB);
Scan SDB_2D to generate FP_1;
PDB=build_projected_database(SDB_2D,FP_1);
If (FP_1 != ∅) then
   FP=FP_1;
   i=1;
   While (i ≤ the size of FP_1 ) do {
      FP=subPGMiner(FP_1,FP,PDB,the i-th pattern in FP_1,minSup);
      i++;
   }
Return FP;
```

Fig. 6. Pattern-growth algorithm.

where 4 is the offset of O in X-axis of 2D string, (R < B = O < P, P < B = O < R), 3 is the rank of R in Y-axis of (R < B = O < P, P < B = O < R) and 2 is the rank of O in Y-axis of (R < B = O < P, P < B = O < R).

4. Experiment

We measure the efficiency of proposed two approaches by the number of images, the total number of objects, average number of objects in an image and the minimum support (Agrawal & Srikant, 1994, 1995). First, we utilized the synthetic data generator developed by IBM Almaden Research Center to generate transactions, 1D sequence, and we regard each transaction as the horizontal axis of a 2D string. For each generated item, the position along the vertical axis is determined randomly. Then each transaction is translated to a 2D string. We performed all experiments on IBM PC with 256 MB main memory and 2.40 GHz CPU. The synthetic data generation program takes the parameters shown in Table 2 and generates all images by setting $D \times D = 6 \times 6$. Note that the convention of $T6kM8N10S50\%$ means that the data set includes 6k images, the average number of objects in an image is 8, the total number of objects is 10 and the minimum support is 50%.

Fig. 10 illustrates the runtime of the Apriori-based algorithm increases more rapidly than the pattern-growth algorithm as the number of images increases. While the total number of objects

```
Input: Two sets of co-orientation patterns FP_1 and FP,
       a projected database PDB,
       a size-k co-orientation pattern P and
       the minimum support threshold minSup.
Output: A set of co-orientation patterns FP
Variable: P_k: a set of size-k 2D strings
          s: a 2D string

For each object o is in FP_1 but not in P
    P_(k+1)=Gen_Pattern(PDB, P, o, minSup);
    If (P_(k+1)!= ∅) then
        For each s in P_(k+1)
            Scan PDB to check if s is frequent;
            If s is frequent then
                Insert s to FP;
                PDB=update_projected_database(PDB, s);
                FP=subPGMiner(FP_1, FP, PDB, s, minSup);
        End
End
Return FP;
```

Fig. 7. subPGMiner function.

```
Input: A projected database PDB,
       a size-k co-orientation pattern P, an object o and
       the minimum support threshold minSup.
Output: A set of size-(k+1) 2D stirngs P_(k+1)
Variables: P_k: set of size-k 2D strings
           P_kx: size-k 1D string
           s_x,s_y: size-k 1D string

Let P=<P_x,P_y> where
    P_x=(o_1=o_2=…=o_i)<(o_i+1=o_i+2=…=o_j)<…<(o_m=o_m+1=…=o_k);
P_(k+1)x={(o_1=…=o_i)<(o_i+1=…=o_j)<…<(o_m=…=o_k=o)
        ∪(o_1=…=o_i)<(o_i+1=…=o_j)<…<(o_m=…=o_k)<(o)};
For each s_x in P_(k+1)x
    Scan PDB to check if s_x is frequent;
    If s_x is frequent then
        According to s_x, scan PDB to generate s_y;
        Insert <s_x,s_y> to P_k+1;
    Else
        Delete s_x from P_(k+1)x;
End
Return P_(k+1);
```

Fig. 8. Gen_Pattern function of Fig. 7.

increase, the effect of that of each algorithm is not obvious. However, the pattern-growth algorithm is more efficient than the Apriori-based algorithm in Fig. 11. As shown in Fig. 12, the Apriori-based algorithm and the pattern-growth algorithm takes more time as the average number of objects in an image increases. Fig. 13 demonstrates that the runtime of the Apriori-based algorithm scales up more and more rapidly than the pattern-growth algorithm as minimum support decreases. When minimum supports are 22% and 18%, Apriori-based algorithm takes less time than pattern-growth algorithm because the length of spatial co-orientation patterns is shorter.

5. Applications

5.1. Mining painting color style

One application of spatial co-orientation pattern mining is mining painting style from images (Shan, 2009). The painting style relates to the painting techniques which the artist uses to create the painting. In other words, the painting style concerns the common properties of the artist's works. We can mine the painting style of artists to finding out the artists' characteristics and then utilize those to discriminate the artist's works from others. Several features can be extracted from images, such as color, shape, texture and spatial relationship, to represent characteristics of artists' works.

Especially, MPEG-7 color layout descriptor is one of the color features to capture the spatial distribution of colors. To extract MPEG-7 color layout descriptor, an images is divided into 8×8 grids, 64 blocks. Then, representative color detection from each block, Discrete Cosine Transform (DCT) of these 64 blocks and non-linear quantization of the zigzag-scanned coefficients are performed to produce the color layout descriptor. We can use 2D string to represent such 64 blocks, and then utilize spatial co-orientation patterns to represent the painting style in terms of the color layout.

5.2. Discovering interesting patterns in basketball games

Spatial co-orientation pattern mining is useful for discovering interesting patterns in basketball game data. In basketball competition, basketball strategies, such as offense, defense and so on, are

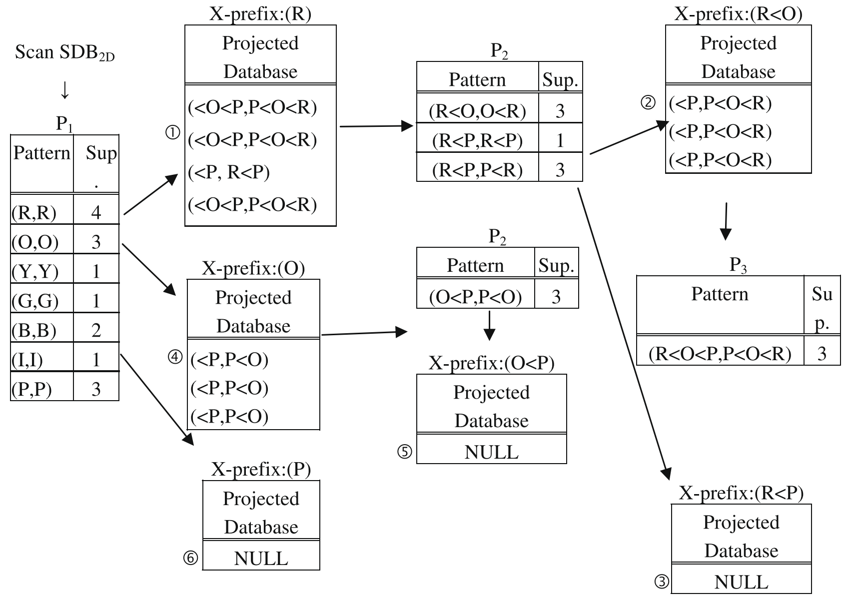

Fig. 9. Example of pattern-growth algorithm.

Table 1
String database SDB_{2D} and parts of its pseudo-projected databases.

TID	2D string	(R)	(O)	(P)	(R < O)	(R < P)	(O < P)	...
100	(R < O < P < G, G < P < O < R)	<2,4>	<3,3>	<4,2>	<3,4 3>	<4,4 2>	<4,3 2>	...
200	(R < B = O < P, P < B = O < R)	<2,3>	<4,2>	<$,1>	<4,3 2>	<$,3 1>	<$,2 3>	...
300	(B = R < P < I, I < R < B < P)	<3,2>	∅	<4,4>	∅	<4,2 4>	∅	...
400	(Y < R < O < P, Y < P < O < R)	<3,4>	<4,3>	<$,2>	<4,4 3>	<$,4 2>	<$,3 2>	...

Table 2
Parameters of experiment.

T	Number of images
D × *D*	Size of an image
M	Average number of objects in an image
N	Total number of objects
S	Minimum support

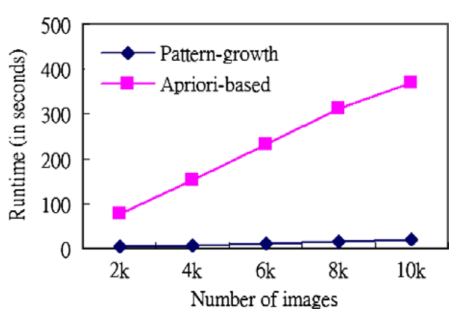

Fig. 10. Performance of these two algorithms on data set *M*6*N*10*S*0.5%.

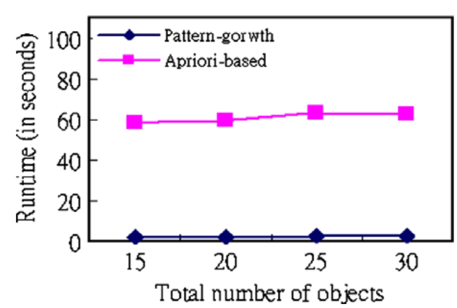

Fig. 11. Performance of these two algorithms on data set *T*6k*M*6*S*5%.

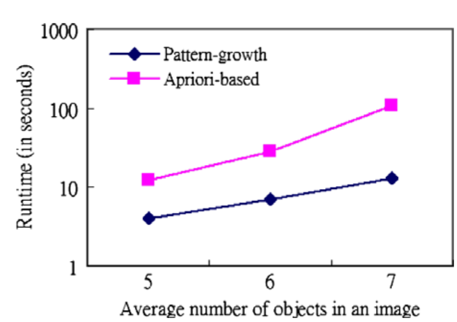

Fig. 12. Performance of these two algorithms on data set *T*3k*N*20*S*1%.

important. Coaches can detect opponent's strategies from these discovered patterns. To discover patterns from basketball videos, we first segment videos into shots and select key-frames for modeling. Then we annotate each object in key-frames by human and transfer this information into 2D strings for mining.

For instance, the Triangle offense is a famous basketball offense strategy. Chicago Bulls used this basketball offense strategy in the past, and it is still useful for Los Angeles Lakers today. Fig. 14 shows the pattern of the Triangle offense. In Fig. 14, No. 7 (Lamar Odom), No. 8 (Kobe Bryant), No. 9 (Laron Profit), No. 18 (Sasha Vujacic) and No. 31 (Chris Mihm) are NBA (National Basketball Association) players, belonging to Los Angeles Lakers, and they are set in the Triangle offense in this figure.

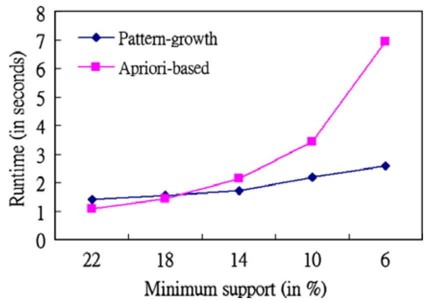

Fig. 13. Performance of these two algorithms on data set $T1kM6N10\%$.

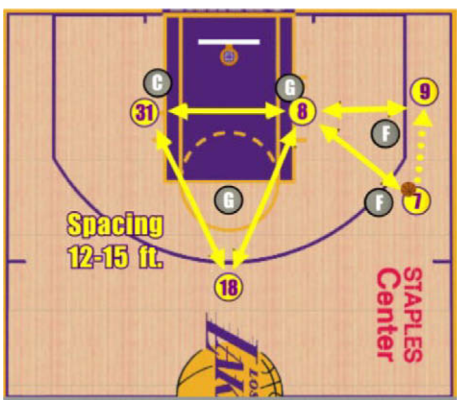

Fig. 14. Example of the Triangle offense. (The official site of the National Basketball Association, http://www.nba.com/.)

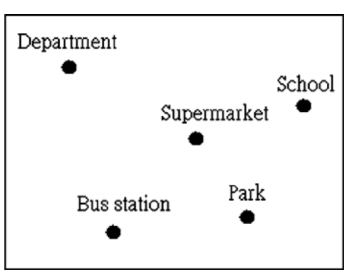

Fig. 15. Example of a cognitive map.

5.3. Analyzed tool for spatial cognitive development

Spatial co-orientation pattern mining can also be utilized as a tool for analyzing students' spatial cognitive development (Wei & Shan, 2006). Spatial cognition concerns how human interpret spatial complexity through spatial properties of objects, such as distance, direction and so on, in the world. Human construct the individual mental image according to their own spatial cognition. People commonly sketch the cognitive map, a graphical representation of spatial knowledge, to represent their mental image. In geographic education, most studies utilize cognitive maps to analyze the development of students' spatial cognitive ability. Some studies have focused on the analysis of cognitive maps drawn by students of different sexes, ages and so on, and finding out which factors influence students' spatial cognition. These studies provide education authorities with reference in compiling the source materials for geography education.

For example, students were asked to point the directional relationship among 5 landmarks according the reference, "school," and Fig. 15 is a cognitive map drawn by someone. When we transfer each cognitive map into symbolic picture, spatial co-orientation patterns mean that most students construct such distribution. According to these patterns, researchers interest in discussing which factors induce this result.

6. Conclusions

In this paper we introduce the problem of mining spatial co-orientation patterns in image databases. We utilize 2D string to represent the spatial orientation of objects in an image. We propose two algorithms, Apriori-based algorithm and pattern-growth algorithm, to solve this problem. Our experiments show the good scale-up property of these two algorithms. Pattern-growth algorithm performs more effectively than Apriori-based algorithm.

References

Agrawal, R., & Srikant, R. (1994). Fast algorithms for mining association rules. In *Proceedings of the 20th international conference on very large data bases VLDB*.

Agrawal, R., & Srikant, R. (1995). Mining sequential patterns. In *Proceedings of the 7th IEEE international conference on data engineering*.

Chang, S. K., Shi, Q. Y., & Yan, C. W. (1987). Iconic indexing by 2D strings. *IEEE Transactions on Pattern Analysis and Machine Intelligence, 9*(3).

Ester, M., Frommelt, A., Kriegel, H., & Sander, J. (2000). Spatial data mining: Database primitives, algorithms and efficient DBMS support. *Data Mining and Knowledge Discovery, 4*(2 & 3).

Hsu, W., Dai, J., & Lee, M. (2003). Mining viewpoint patterns in image databases. In *Proceedings of the 9th ACM international conference on knowledge discovery and data mining*.

Huang, Y., Shekhar, S., & Xiong, H. (2004). Discovering co-location patterns from spatial datasets: A general approach. *IEEE Transactions on Knowledge and Data Engineering, 16*(12).

Liu, X., Shekhar, S., & Chawla, S. (2000). Processing object-orientation-based direction queries: A summary of results. In *Proceedings of the 8th ACM international workshop on geographic information systems*.

Loperski, K., & Han, J. (1995). Discovery of spatial association rules in geographic information databases. In *Proceedings of the 4th international symposium on large spatial databases*.

Morimoto, Y. (2001). Mining frequent neighboring class sets in spatial databases. In *Proceedings of the 7th ACM international conference on knowledge discovery and data mining*.

Pei, J., Han, J., Mortazavi-Asl, B., Wang, J., Pinto, H., & Chen, Q. (2004). Mining sequential patterns by pattern-growth: The PrefixSpan approach. *IEEE Transactions on Knowledge and Data Engineering, 16*(11).

Shan, M. K. (2009). Discovering color styles from fine art images of Impressionism. *International Journal of Computer Science and Security, 3*(4).

Shekhar, S., & Huang, Y. (2001). Discovering spatial co-location patterns: A summary of results. In *Proceedings of the 7th international symposium on spatial and temporal databases*.

Wei, L. Y., & Shan, M. K. (2006). Mining spatial co-orientation patterns for analyzing portfolios of spatial cognitive development. In *Proceedings of the 6th IEEE international conference on advanced learning technologies ICALT*.

Yoo, J. S., & Shekhar, S. (2004). A partial join approach for mining co-location patterns. In *Proceedings of the 12th ACM international workshop on geographic information systems ACMGIS*.