Discovering Partial Periodic Spatial Patterns in Spatiotemporal Databases

R. Uday Kiran[1,2], C. Saideep[3], Koji Zettsu[1], Masashi Toyoda[2], Masaru Kitsuregawa[2,4], P. Krishna Reddy[3]
[1]National Institute of Information and Communications Technology, Tokyo, Japan
[2]The University of Tokyo, Tokyo, Japan
[3]International Institute of Information Technology-Hyderabad, Telangana, India
[4]National Institute of Informatics, Tokyo, Japan
{uday_rage,toyoda,kitsure}@tkl.iis.u-tokyo.ac.jp, saideep.chennupati@research.iiit.ac.in, zettsu@nict.go.jp, pkreddy@iiit.ac.in

***Abstract*—Finding partial periodic patterns in very large databases is a challenging problem of great importance in many real-world applications. Most previous work focused on finding these patterns in temporal (or transactional) databases and did not recognize the spatial characteristics of items. In this paper, we propose a more flexible model of partial periodic spatial pattern that may be present in spatiotemporal database. Three constraints, *maximum inter-arrival time* ($maxIAT$), *minimum period-support* ($minPS$) and *maximum distance* ($maxDist$), have been employed to determine the interestingness of a pattern in a spatiotemporal database. The $maxIAT$ controls the maximum duration in which a pattern must reappear to consider its occurrence as periodic within the data. The $minPS$ controls the minimum number of periodic occurrences of a pattern within the data. The $maxDist$ controls the maximum distance between the items in a pattern. All patterns satisfying these three constraints are returned. An efficient algorithm, called SpatioTemporal-Equivalence CLAss Transformation (ST-ECLAT), has also been described to discover all partial periodic spatial patterns in a spatiotemporal database. This algorithm employs a novel smart depth-first search technique to discover desired patterns effectively. Experimental results demonstrate that the proposed algorithm is efficient. We also present a case study in which we apply our model to find useful information in the air pollution database.**

***Index Terms*—data mining, pattern mining, periodic patterns, spatiotemporal database**

I. Introduction

Partial periodic pattern mining is an important data mining model with many real-world applications [1]. A partial periodic pattern represents a set of items that repeats itself at regular intervals in the data. It is thus useful in characterizing the cyclic behavior. Most previous works [2]–[4] focused on finding partial periodic patterns in temporal (or transactional) databases. In some applications, e.g. market basket, such a model has proved to be important and meaningful. However, in other applications, the occurrence information of items alone may not always be sufficient to represent the significance of a pattern. Consider the following examples.

- *Environmental studies.* Air pollution is a major factor affecting climate change and public health. Several air quality monitoring stations have been set up across the world to monitor air pollutants [5]. The data generated by these stations represent a spatiotemporal database. The information regarding the geographical regions (or sets of neighboring stations) where people have been regularly exposed to unhealthy or hazardous levels of air pollution may be found very useful to the Ecologists in devising environmental policies to slow and reverse the climate change and improve public health.
- *Road traffic analytics.* Traffic congestion is a serious problem in smart cities. To confront this problem, several sensors have been placed on road segments to monitor congestion. The data generated by these sensors represent a spatiotemporal database. The information regarding the neighboring road segments where people have regularly faced traffic congestion throughout a day(s) or certain time periods of a day may be found useful to the users for urban planning and traffic monitoring.

In the above examples, we can see that users may be interested in finding only those periodically occurred patterns where items are close to each other in a coordinate system. Notably, the current partial periodic pattern models are not ideal for these applications because they disregard the crucial information about the spatial (or geometric) characteristics of an item. In the literature, the spatial characteristics of an item within the data have been exploited to find spatial co-occurrence patterns [6], [7]. However, it has to be noted that these studies do not take into account the periodic occurrence information of a pattern in the data.

With this motivation, this paper proposes a more flexible model of partial periodic spatial pattern that exists in a spatiotemporal database. The proposed model considers a pattern in a spatiotemporal database as a partial periodic spatial pattern if it satisfies the user-specified *maximum inter-arrival time* ($maxIAT$), *minimum period-support* ($minPS$) and *maximum distance* ($maxDist$) constraints. The $maxIAT$ controls the maximum time interval within which a pattern must reappear in the data. The $minPS$ controls the minimum number of periodic occurrences of a pattern within the data. The $maxDist$ controls the maximum distance between any two items in

978-1-7281-0858-2/$31.00 ©2019 IEEE

a pattern. The patterns generated by the proposed model satisfy the *downward closure property*. That is, all non-empty subsets of a partial periodic spatial pattern are also partial periodic spatial patterns. This property facilitates the mining of partial periodic spatial patterns in very large databases practicable. An efficient algorithm, called SpatioTemporal-Equivalence CLAss Transformation (ST-ECLAT), has been introduced to discover all partial periodic spatial patterns in a spatiotemporal database. The proposed algorithm employs **smart depth-first search technique** to discover all desired patterns efficiently. Experimental results demonstrate that our algorithm is not only memory efficient, but also 10 to 100 times faster than a naïve extended ECLAT [8] algorithm. We also present a case study in which we apply our model to find useful information in the air pollution database.

The rest of the paper is organized as follows. Related work is presented in Section 2. Section 3 introduces the model of partial periodic spatial pattern. Section 4 describes the ST-ECLAT algorithm. Experimental results are reported in Section 5. Section 6 concludes the paper with future research directions.

II. Related Work

Tanbeer et al. [2] described a model to find full periodic-frequent patterns in a transactional database. A major limitation of this model is that it fails to discover partial periodic-frequent patterns in a transactional database. When confronted with this problem in real-world applications, researchers have tried to find partial periodic (frequent) patterns using alternative measures, such as sequence periodic ratio [4] and *period-support* [3]. It has to be noted that these studies do not take into account the spatial characteristics of an item within the database. On the contrary, the proposed model takes into account both the spatial and temporal characteristics of the items within the database.

The problem of finding spatiotemporal co-occurrence patterns (or association rules) in spatiotemporal databases has received a great deal of attention [6], [7]. Unfortunately, all spatiotemporal co-occurrence pattern mining algorithms determine the interestingness of a pattern by taking into account only its *support* and disregard the periodic occurrence information of a pattern within the data. On the contrary, the proposed study takes into account the spatiotemporal characteristics of the items within the database to find partial periodic spatial patterns.

III. Proposed Model

For brevity, we describe the model of partial periodic spatial pattern using a spatial database and a temporal database. A hypothetical air pollution data is used for illustration purposes.

Let $I = \{i_1, i_2, \cdots, i_n\}$, $n \geq 1$, be a set of geometric (or spatial) items. Let P_{i_j} denote a set of coordinates for an item $i_j \in I$. The spatial database SD is a collection of items and their coordinates. That is, $SD = \{(i_1, P_{i_1}), (i_2, P_{i_2}), \cdots, (i_n, P_{i_n})\}$. The above notion of spatial database facilitates us to capture items of various geometric forms, such as point, line, or polygon. Two items, $i_p, i_q \in I$, are said to be **neighbors** to each other if $Dist(i_p, i_q)(= Dist(i_q, i_p)) \leq maxDist$, where $Dist(.)$ is a distance function and $maxDist$ is a user-specified *maximum distance*.

TABLE I: Spatial database

Item	Coordinates
a	(0,0)
b	(3,3)
c	(0,7)
d	(5,3)
e	(5,6)
f	(7,7)
g	(7,1)

TABLE II: Neighbors

Item	Neighbors
a	bd
b	$acdefg$
c	bde
d	$abefg$
e	$bcdfg$
f	$bdeg$
g	$bdef$

Example 1. Let $I = \{a, b, c, d, e, f, g\}$ be a set of items (or air quality measuring stations). A spatial database of these items is shown in Table I. Given the distance measure as Euclidean, the distance between a and b, i.e., $Dist(a, b) = 4.24$. If the user-specified $maxDist = 6$, then a and b are considered as neighbors because $Dist(a, b) \leq maxDist$. Table II lists the neighbors of every item in Table I.

Definition 1. ***(Spatial pattern.)*** *Let $X \subseteq I$ be an itemset (or a pattern). If X contains k items, then it is called a k-pattern. A pattern X in SD is said to be a* ***spatial pattern*** *if the maximum distance between any two of its items is no more than the user-specified $maxDist$. That is, X is a spatial pattern if $max(Dist(i_p, i_q) | \forall i_p, i_q \in X) \leq maxDist$.*

Example 2. The set of items a and b, i.e., ab is a pattern. This pattern contains two items. Therefore, it is a 2-pattern. The pattern ab is also a spatial pattern because $max(Dist(a, b)) \leq maxDist$. On the contrary, $abf \supset ab$ is not a spatial pattern because $Dist(a, f) \nleq maxDist$ or $max(Dist(a, b), Dist(a, f), Dist(b, f)) \nleq maxDist$.

A transaction $t_{tid} = (tid, ts, Y)$, where $tid \geq 1$ represents the transaction identifier, $ts \in \mathbb{R}^+$ represents the timestamp and $Y \subseteq I$ is a pattern. An (irregular) **temporal database** TDB is a collection of transactions. That is, $TDB = \{t_1, t_2, \cdots, t_m\}$, $1 \leq m \leq |TDB|$, where $|TDB|$ represents the size of database. If a pattern $X \subseteq Y$, it is said that X occurs in transaction t_{tid}. The timestamp of this transaction is denoted as ts^X_{tid}. Let $TS^X = \{ts^X_{tid_a}, ts^X_{tid_b}, \cdots, ts^X_{tid_c}\}$, $tid_a, tid_b, tid_c \in (1, |TDB|)$, denote the set of all timestamps in which the pattern X has appeared in the database. The $support$ of X in TDB, denoted as $SUP(X)$, represents the number of transactions containing X in TDB. That is, $SUP(X) = |TS^X|$.

Example 3. The temporal database of the items in Table I is shown in Table III. The first transaction in this table provides the information that the stations a, b, c, e, f and g have recorded hazardous quantities[1] of air pollution at the timestamp of 1. Similar statement can be made on remaining

[1] A measured quantity of an air pollutant by a sensor is considered hazardous to the people if it exceeds a threshold value specified by the Air Quality Index Standards.

TABLE III: Running example: temporal database

tid	*ts*	items
1	1	*abcefg*
2	2	*abcf*
3	3	ab
4	5	*cdeg*
5	6	*acd*
6	7	*bcdfg*
7	8	*acde*
8	11	*acefg*
9	12	*abef*
10	13	*abdef*
11	14	*eg*
12	15	*acdfg*

transactions in Table III. The size of this temporal database, i.e., $m = |TDB| = 12$. The spatial pattern ab appears in the transactions whose timestamps are 1, 2, 3, 12 and 13. Therefore, $ts_1^{ab} = 1$, $ts_2^{ab} = 2$, $ts_3^{ab} = 3$, $ts_9^{ab} = 12$ and $ts_{10}^{ab} = 13$. The complete set of timestamps at which ab has occurred in Table III, i.e., $TS^{ab} = \{1, 2, 3, 12, 13\}$. The *support* of ab, i.e., $SUP(ab) = |TS^{ab}| = 5$.

A recurrence of a pattern X is considered $periodic$ if the inter-arrival time is no more than the user-specified *maximum inter-arrival time* ($maxIAT$). It is formally defined as follows.

Definition 2. ***(A periodic inter-arrival time of*** X ***in*** TDB***.)*** *Let* ts_a^X *and* ts_b^X, $1 \leq a < b \leq m$, *denote two consecutive timestamps at which* X *has appeared in* TDB. *The time difference between* ts_a^X *and* ts_b^X *is defined as an* ***inter-arrival time*** *of X, and denoted as* iat_p^X, $1 \leq p \leq SUP(X) - 1$. *That is,* $iat_p^X = ts_b^X - ts_a^X$. *Let* IAT^X *denote the set of all inter-arrival times of ab in* TDB. *An inter-arrival time of X,* $iat_p^X \in IAT^X$, *is considered periodic if* $iat_p^X \leq maxIAT$.

Example 4. In Table III, the spatial pattern ab has appeared consecutively in the transactions whose timestamps are 1 and 2. Therefore, an inter-arrival time of ab, i.e., $iat_1^{ab} = 2 - 1 = 1$. Similarly, other inter-arrival times of ab in Table III are: $iat_2^{ab} = 3 - 2 = 1$, $iat_3^{ab} = 12 - 3 = 9$ and $iat_4^{ab} = 13 - 12 = 1$. The set of all inter-arrival times of ab, i.e., $IAT^{ab} = \{1, 1, 9, 1\}$. If the user-specified $maxIAT = 1$, then iat_1^{ab}, iat_2^{ab} and iat_4^{ab} are considered as the periodic occurrences of ab because $iat_1^{ab} \leq maxIAT$, $iat_2^{ab} \leq maxIAT$ and $iat_4^{ab} \leq maxIAT$. On the contrary, iat_3^{ab} is considered as an aperiodic (or irregular) occurrence of ab because $iat_3^{ab} \nleq maxIAT$.

Definition 3. ***(Period-support of*** X ***in*** TDB***.)*** *The* ***period-support*** *of X, denoted as* $PS(X)$, *captures the number of periodic occurrences of X in* TDB. *That is,* $PS(X) = |\widehat{IAT^X}|$, *where* $\widehat{IAT^X} \subseteq IAT^X$ *such that if there* $\exists iat_p^X \in IAT^X$ *with* $iat_p^X \leq maxIAT$, *then* $iat_p^X \in \widehat{IAT^X}$. *In other words,* $\widehat{IAT^X}$ *represent the set of all periodic inter-arrival times of* X *in* TDB.

Example 5. The set of all periodic inter-arrival times of ab, i.e., $\widehat{IAT^{ab}} = \{1, 1, 1\}$. Thus, the *period-support* of ab, i.e., $PS(ab) = |\widehat{IAT^{ab}}| = |\{1, 1, 1\}| = 3$. In other words, ab has appeared periodically three times within the data.

Definition 4. ***(Partial periodic spatial pattern*** X***.)*** *A spatial pattern* X *is said to be a partial periodic spatial pattern if its* period-support *is no less than the user-specified* minimum period-support *(*$minPS$*). That is,* X *is a partial periodic spatial pattern if* $PS(X) \geq minPS$ *and* $max(Dist(i_p, i_q | \forall i_p, i_q \in X)) \leq maxDist$.

Example 6. If the user-specified $minPS = 3$, then ab is a partial periodic spatial pattern because $Dist(a, b) \leq maxDist$ and $PS(ab) \geq minPS$. This pattern provides the useful information that the people in the geographical regions of the stations a and b have been regularly exposed to hazardous levels of pollution. The complete set of partial periodic spatial patterns generated from Tables I and III have been listed in Table IV.

TABLE IV: All partial periodic spatial patterns generated from Tables I and III. The term 'Pat.' represents 'Pattern.'

Pat.	PS
a	4
ab	3
c	4
cd	3
b	3
d	3
e	3
f	3

Definition 5. ***(Problem definition.)*** *Given a spatial database (SD), temporal database (*TDB*),* maximum inter-arrival time *(*$maxIAT$*),* minimum period-support *(*$minPS$*) and* maximum distance *(*$maxDist$*), the aim of partial periodic spatial pattern mining is to discover all patterns in* SD *and* TDB *that satisfy the following two conditions: (i) the maximum distance between any two items in a pattern is no more than the user-specified* $maxDist$ *and (ii) the* period-support *of a pattern must be no less than the user-specified* $minPS$.

The constraints, $maxIAT$ and $minPS$, can also be expressed in percentage of $(ts_{max} - ts_{min})$ and $(|TDB| - 1)$, respectively. The ts_{min} and ts_{max} respectively denote the minimum and maximum timestamps of all transactions in TDB.

The partial periodic spatial patterns generated by the proposed model satisfy the *downward closure property*. That is, all non-empty subsets of a partial periodic spatial pattern are also partial periodic spatial patterns. This property facilitates the proposed model practicable on real-world very large databases.

IV. Proposed Algorithm

The proposed SpatioTemporal-Equivalence CLAss Transformation (ST-ECLAT) extends ECLAT [8] to find partial periodic spatial patterns in spatiotemporal database. A key difference between ECLAT and ST-ECLAT is as follows: *ECLAT performs typical depth-first search on the itemset lattice. On the contrary, proposed algorithm conducts smart depth-first search on the itemset lattice by utilizing the prior information regarding the items' neighbors.* We now describe the basic idea of ST-ECLAT algorithm.

A. Basic idea: smart depth-first search

In the smart depth-first search, we consider a partial periodic spatial pattern as a prefix pattern and explore only those immediate supersets whose items are common neighbors to all items in a prefix pattern (see Example 7). The formal definitions of *prefix pattern* and *suffix items* are given in Definitions 6 and 7, respectively. The correctness of our idea is shown in Property 1.

Example 7. In the smart depth-first search, we first identify all neighbors for every item in the database. Next, we start with the first item $a \in S$. As a is a partial periodic spatial pattern, we consider a as a prefix pattern. Next, we consider the neighbors of a in $S - a$ (i.e., $(S - a) \cap N_a = \{b, d\}$) as suffix items, and perform depth-first search on the supersets of a containing only these two items. Notice that all other items in $S - a$ are not taken into account for exploration because they implicitly fail the $maxDist$ constraint. Since ab is a partial periodic spatial pattern, we consider ab as a prefix pattern, and common neighbors of a and b in $S - ab$, i.e., $(S - ab) \cap (N_a \cap N_b) = d$ is considered as a suffix item. As abd is not a partial periodic spatial pattern, we verify for ad and end the depth-first search for item a. The above smart depth-first search is repeated for remaining items in S. This approach of exploring only common neighbors of a prefix pattern reduces the search space effectively.

Definition 6. ***(Prefix pattern.)*** *Let $PPSI = \{i_1, i_2, \cdots, i_n\}$ be an ordered set of partial periodic spatial items (or 1-patterns) with respect to a measure M (say, period-support). The prefix pattern α is an ordered subset of $PPSI$. That is, $\alpha = \{i_1, i_2, \cdots, i_k\} \subseteq PPSI,\ k \leq n$.*

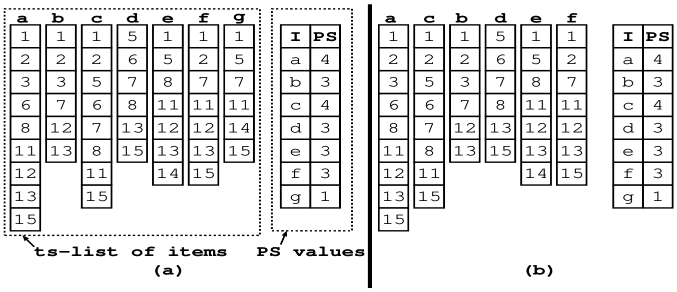
Definition 7. ***(Suffix items.)*** *Let $PPSI - \alpha$ denote an ordered set of remaining items. An item $i_p \in (PPSI - \alpha)$ and $p > k$ is a suffix item if i_p is a neighbor for all items in α. That is, $i_p \in (PPSI - \alpha)$ is a suffix item if $i_p \in \cap_{i_j \in \alpha} N(i_j)$.*

Property 1. Let $N(\alpha) = \{N(i_1) \cap N(i_2) \cap \cdots \cap N(i_k)\}$ denote the common neighbors of all items in α. If $\alpha \cup i_p$ is a partial periodic spatial pattern, then $i_p \in (N(\alpha) \cap (PPSI - \alpha))$. If $i_p \notin (N(\alpha) \cap (PPSI - \alpha))$, then $\alpha \cup i_p$ cannot be a partial periodic spatial pattern because $max(Dist(i_a, i_b) | \forall i_a, i_b \in (\alpha \cup i_p)) > maxDist$.

B. ST-ECLAT

The ST-ECLAT is presented in Algorithms 1 and 2. Since the algorithms to determine the distances between the items and the calculation of *period-support* for a pattern from its list of timestamps is straightforward, we have not presented these two algorithms in the paper. We now illustrate these two algorithms using the databases in Tables I and III. Let $maxDist = 6$, $maxIAT = 1$ and $minPS = 3$. Let Euclidean distance be the distance function.

First, we identify neighbors for each item by scanning the spatial database. The neighbors for each item in I are shown in Table II (line 1 in Algorithm 1). Next, we scan the temporal database and construct the list of timestamps (or ts-list) for every item in the database. Simultaneously, we also calculate the *period-support* for each item. Fig. 1(a) shows the items' ts-list and PS values after scanning the temporal database (line 2 in Algorithm 1). Since the proposed patterns satisfy the downward closure property, we prune the items that have *period-support* less than $minPS$. The remaining items are considered as partial periodic spatial items (or 1-patterns) and sorted in descending order of their PS value. Fig. 1(b) shows the sorted list of partial periodic spatial items along with their ts-lists and PS values (line 3 in Algorithm 1). Let S and $TS(S)$ denote the sorted list of partial periodic spatial items and their ts-lists, respectively. Next, we initialize prefix pattern $\alpha = \emptyset$ and call depthFirstSearch function by passing $\alpha, TS(S)$ and S as neighbors of α (line 4 in Algorithm 1).


ts-list of items
PS values
(a)
(b)

a	b	c	d	e	f	g
1	1	1	5	1	1	1
2	2	2	6	5	2	5
3	3	5	7	8	7	7
6	7	6	8	11	11	11
8	12	7	13	12	12	14
11	13	8	15	13	13	15
12		11		14	15	
13		15				
15						

I	PS
a	4
b	3
c	4
d	3
e	3
f	3
g	1

a	c	b	d	e	f
1	1	1	5	1	1
2	2	2	6	5	2
3	5	3	7	8	7
6	6	7	8	11	11
8	7	12	13	12	12
11	8	13	15	13	13
12	11			14	15
13	15				
15					

I	PS
a	4
b	3
c	4
d	3
e	3
f	3
g	1

Fig. 1: Finding partial periodic spatial items. (a) After scanning the temporal database and (b) Sorted list of partial periodic spatial items

The depthFirstSearch algorithm initially verifies whether there exists any neighbor for the prefix pattern α (line 1 in Algorithm 2). Since $S \neq \emptyset$, the first item in S, i.e., a and its ts-list are popped as prefix pattern (line 2 in Algorithm 2). This item is also generated as a partial periodic spatial item (line 3 in Algorithm 2). Next, we identify common neighbors in α and a, i.e., $N(\alpha \cap a) = N(a) = \{b, d\}$, as suffix items. Fig. 2(a) shows the prefix pattern a, its suffix items and their ts-lists (line 3 in Algorithm 2). Since a is a partial periodic spatial pattern, we perform $a \cup b$, generate its ts-list and suffix items (see Fig. 2(b)). Next, we calculate period-support of ab by scanning its ts-list (line 8 and 9 in Algorithm 2). As the $PS(ab) \geq minPS$, we generate ab as a partial periodic spatial pattern and recursively call the depthFirstSearch algorithm with $\alpha = ab$, $TS(d)$ and $N(ab) = d$ (lines 10 to 12 in Algorithm 2). The prefix pattern abd is generated by performing $ab \cup d$. Next, we generate ts-list of abd, i.e., $TS(abd) = TS(ab) \cap TS(d)$ as shown in Fig. 2(c). The suffix item is set to $\emptyset$. As $PS(abd) < minPS$, we consider abd as uninteresting pattern and avoid searching its supersets. Next, we generate ad and determine it as an uninteresting pattern. As the depth-first search on a and its neighboring items is completed, we choose the next item $c \in S - a$ as prefix pattern. All of the above steps are repeated until S is empty. All partial periodic spatial patterns generated from Tables I and III are shown in Table IV.

V. Experimental Results

As there exists no algorithm to find partial periodic spatial patterns in a spatiotemporal database, we show that ST-ECLAT is efficient by evaluating against a naïve extended ECLAT algorithm (referred as n-ECLAT).

A. Experimental setup

The algorithms, n-ECLAT and ST-ECLAT, have been written in Python 3 and executed on a machine with 2.5 GHz

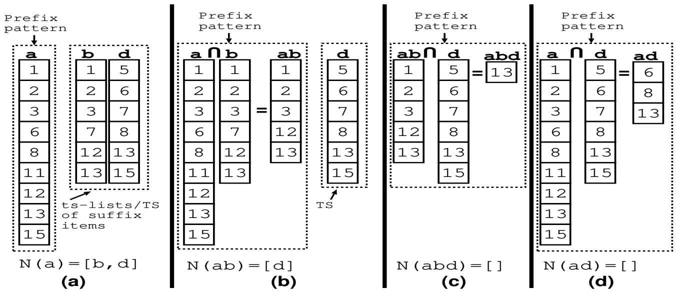


Fig. 2: Performing depth-first search on the itemset lattice using item a. (a) prefix pattern a, (b) prefix pattern ab, (c) prefix pattern abd and (d) prefix pattern ad

Algorithm 1 ST-ECLAT (Set of items (I), spatial database (SD), temporal database (TDB), maximum distance ($maxDist$), maximum inter-arrival time ($maxIAT$), minimum period-support ($minPS$)

```
Scan the spatial database and find neighbors for each item using a distance function. Let $N(i_j)$ denote the set of neighboring items for an item $i_j$.
Scan the temporal database and generate ts-list for each item. Simultaneously, calculate *period-support* for each item $i_j$ by traversing its ts-list.
Find partial periodic spatial items by pruning uninteresting items that have *period-support* less $minPS$. Sort the partial periodic spatial items in descending order of their $support$ values. Let $S$ denote this sorted list of partial periodic spatial items. Let $TS(S)$ denote the set of ts-lists for all partial periodic spatial items in $S$.
Initialize $\alpha = \emptyset$ and call depthFirstSearch($\alpha$, TS($S$), $S$).
```

processor and 8 GB RAM. The experiments have been conducted on real-world **air pollution** database.

Atmospheric Environmental Regional Observation System (AEROS) represents a set of air quality monitoring stations set up by the Ministry of Environment, Japan. Each station reports the measured air pollutants, such as PM2.5[2] and SO_2, at hourly intervals. In this paper, we confine our experimentation to PM2.5 pollutant, which is a major cause of many cardio-respiratory problems reported in Japan. The PM2.5 recordings generated by these stations from 1-April-2018 to 30-April-2018 has been transformed into a temporal database such that each transaction contains the following information: *transaction identifier, timestamp in hours, station identifiers that have recorded PM2.5 values no less than 16* $\mu g/m^3$. The pollution database contained 1600 items (or stations) with 720 transactions. It is a high dimensional dense database with minimum, average and maximum transaction lengths equal to 11, 460 and 971, respectively.

Algorithm 2 depthFirstSearch(α, $TS(N(\alpha))$, $N(\alpha)$)

```
while $TS(N(\alpha))! = \emptyset$ do
    $i_j$, ts-list($i_j$) = $TS(N(\alpha))$.pop();{extract item and its ts-list from $TS(N(\alpha))$ using pop function}
    output $\alpha \cup i_j$ as a partial periodic spatial pattern;
    set $N(\alpha \cup i_j) = N(\alpha) \cap N(i_j)$;
    set $suffixItems = \emptyset$.
    for $i_k$,ts-list($i_k$) in $TS(N(\alpha))$ do
        if $i_k \in N(\alpha \cup i_j)$ then
            generate ts-list($\alpha \cup i_k$) = ts-list($\alpha$) $\cup$ ts-list($i_j$);
            $PS(\alpha \cup i_k)$=$calculatePeriodSupport(ts\text{-}list(\alpha \cup i_k))$.
            if $PS(\alpha \cup i_k) \geq minPS$ then
                suffixItems.append($i_k$, ts-list($\alpha \cup i_k$));
    depthFirstSearch($\alpha \cup i_j$, suffixItems, $N(\alpha \cup i_j)$);
```

B. ST-ECLAT vs n-ECLAT

Figure 3a shows the number of partial periodic spatial patterns generated in Pollution database at different $minPS$ and $maxIAT$ values. The $maxDist$ value has been set to 7 kilometers. It can be observed that $maxIAT$ has a positive effect on the generation of partial periodic spatial patterns, while $minPS$ has a negative effect on the number of patterns being generated from the database.

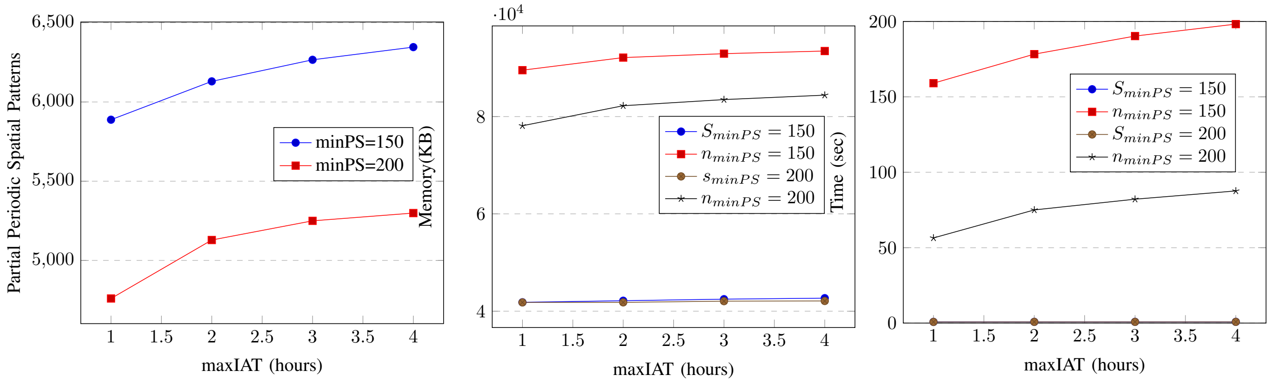
Figure 3b show the maximum memory consumed by ST-ECLAT and n-ECLAT algorithms in Pollution database at different $minPS$ and $maxIAT$ values. The $maxDist$ is set at 5 kilometers. Please note that the legends '$S_{minPS} = x$' and '$n_{minPS} = x$' in all figures respectively represent the performance results of $ST\text{-}ECLAT$ and *n-ECLAT* algorithms at $minPS = X$. The following observations can be drawn from these figures: (i) The memory requirements of ST-ECLAT and n-ECLAT algorithms depend on the number of partial periodic spatial patterns being generated at $minPS$ and $maxIAT$ values. (ii) At any given $maxIAT$ and $minPS$ value, memory consumed by ST-ECLAT is only half of that of n-ECLAT for any database. It is because the smart depth-first search technique facilitated ST-ECLAT to generate all partial periodic spatial patterns by visiting fewer itemsets in the itemset lattice.

Figure 3c show the runtime requirements of ST-ECLAT and n-ECLAT algorithms in Pollution database at different $minPS$ and $maxIAT$ values. The following observations can be drawn from these figures: (i) The runtime requirements of ST-ECLAT and n-ECLAT algorithms depend on the number of partial periodic spatial patterns being generated at $minPS$ and $maxIAT$ values. (ii) At any given $maxIAT$ and $minPS$ value, ST-ECLAT is faster than n-ECLAT by 10 to 100 times in any database. It is because the smart depth-first search technique facilitated ST-ECLAT to generate all partial periodic spatial patterns by visiting fewer itemsets in the itemset lattice.

C. A case study: finding areas where people have been regularly exposed to hazardous levels of PM2.5 pollutant

Some of the interesting patterns generated from **Pollution** database are shown in Table V. The spatial location of the

[2] Atmospheric particulate matter that have a diameter of less than 2.5 micrometers



(a) Number of patterns

(b) Memory

(c) Runtime

Fig. 3: Evaluation of n-ECLAT and ST-ECLAT algorithms

TABLE V: Few interesting patterns generated in Pollution database

S.No.	Patterns (or station ids)	color
1	{868, 872, 874, 876, 881}	Green
2	{4372, 4256, 4312, 4263, 4335,- 4230, 4236, 4329, 4377}	Red
3	{1193, 1225, 1261, 1266}	Blue
4	{3946, 3954, 3503, 4032, 3097, 3939}	Pink

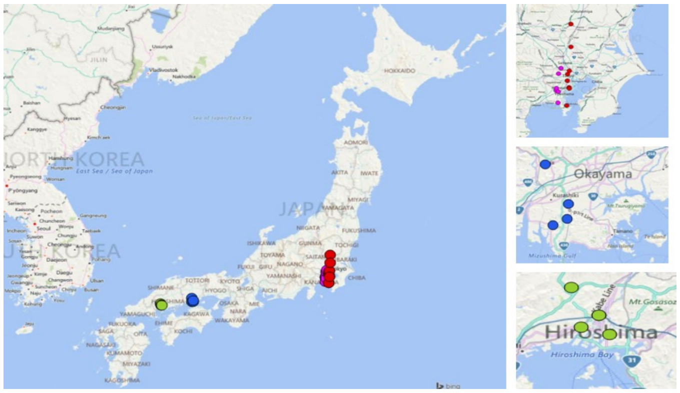


Fig. 4: Areas where people have been regularly subjected to unhealthy levels of PM2.5 concentrate

sensors present in each of these patterns are shown in Figure 4. It can be observed that most sensors in this figure are situated at bay areas (or shipyards). Thus, it can be inferred that people working or living near these bay areas have been regularly exposed to high levels of PM2.5. Such information may be found very useful to the Ecologists in devising policies to control pollution and improve public health. Please note that more in depth studies, such as finding high polluted areas on weekends or particular time intervals of a day, can also be carried out with our model.

VI. Conclusions and Future Work

This paper exploited the spatiotemporal characteristics of the items within the data and proposed a flexible model of partial periodic spatial pattern that may exist in a spatiotemporal database. A novel depth-first search technique using the prior knowledge regarding the neighbors of items has been suggested to reduce the search space and the computational cost of the proposed model. An efficient algorithm, called ST-ECLAT, has also been described to find all partial periodic spatial patterns in a spatiotemporal database. Experimental results demonstrate that the proposed algorithm is not only memory efficient, but also 10 to 100 times faster than the basic extended ECLAT algorithm. We have also presented a case study to demonstrate the usefulness of proposed patterns in real-world applications.

In this paper, we have studied the problem of finding partial periodic spatial patterns by considering static and certain data. As part of future work, we would like to extend the proposed model to data streams and uncertain databases. It is also interesting to investigate parallel algorithms to find all partial periodic spatial patterns in very large databases.

References

[1] P. Esling and C. Agon, “Time-series data mining,” *ACM Comput. Surv.*, vol. 45, no. 1, pp. 12:1–12:34, Dec. 2012.

[2] S. K. Tanbeer, C. F. Ahmed, B.-S. Jeong, and Y.-K. Lee, “Discovering periodic-frequent patterns in transactional databases,” in *Advances in Knowledge Discovery and Data Mining*, 2009, pp. 242–253.

[3] R. U. Kiran, H. Shang, M. Toyoda, and M. Kitsuregawa, “Discovering partial periodic itemsets in temporal databases,” in *Proceedings of the 29th International Conference on Scientific and Statistical Database Management*, ser. SSDBM ’17, 2017, pp. 30:1–30:6.

[4] P. Fournier-Viger, Z. Li, J. C. Lin, R. U. Kiran, and H. Fujita, “Efficient algorithms to identify periodic patterns in multiple sequences,” *Inf. Sci.*, vol. 489, pp. 205–226, 2019.

[5] “World air quality index team,” http://aqicn.org/here/, [Online; accessed 16-August-2019].

[6] W. Ding, C. F. Eick, J. Wang, and X. Yuan, “A framework for regional association rule mining in spatial datasets,” in *Proceedings of the Sixth International Conference on Data Mining*, ser. ICDM ’06, 2006, pp. 851–856.

[7] P. Mohan, S. Shekhar, J. A. Shine, J. P. Rogers, Z. Jiang, and N. Wayant, “A neighborhood graph based approach to regional co-location pattern discovery: A summary of results,” in *Proceedings of the 19th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems*, ser. GIS ’11, 2011, pp. 122–132.

[8] M. J. Zaki, “Scalable algorithms for association mining,” *IEEE Transactions on Knowledge and Data Engineering*, vol. 12, no. 3, pp. 372–390, 2000.