Indian Journal of Science and Technology, *Vol 10(20), DOI: 10.17485/ijst/2017/v10i20/109933, May 2017*
ISSN (Print) : 0974-6846
ISSN (Online) : 0974-5645

Grade-based Spatio-Temporal Sequential Pattern Mining using Support and Event Index Measures

Sunitha Gurram*

Department of CSE, Sree Vidyanikethan Engineering College, A. Rangampet, Tirupati – 517102, Andhra Pradesh, India; gurramsunitha@gmail.com

Abstract

Objectives: The knowledge on cause-effect relationships between instances of real-world entities can be gathered by extracting sequential patterns from spatio-temporal databases. The discovery of the patterns in the context of space and time is a challenging issue. The sequential pattern mining algorithms designed for traditional databases may result in the loss of spatio-temporal correlations due to the improper estimations of properties related to the time and space. The proposed work approaches the problem of designing sequential pattern mining algorithm specifically for spatio-temporal event datasets. **Methods/Statistical Analysis**: An algorithm is proposed which is based on frequency-based measures for mining frequent spatio-temporal sequential patterns. The spatio-temporal sequential pattern mining based on Support index and Event index algorithm proposes two new parameters support index and event index which are used to scrutinize the sequences extracted from the database. A data structure is also proposed to represent the spatio-temporal data for efficient pattern mining. **Findings:** The proposed algorithm generates the interesting set of frequent sequential patterns. The proposed algorithm is compared with Slicing-STS-Miner and MST-ITP and the experimental results proved that the proposed algorithm performs well with the order of two to three. **Application/Improvements:** The proposed algorithm uses frequency-based measures rather than density-based measures. Frequency-based measures take less computational time when compared to density-based measures. The proposed technique is suitable for extracting knowledge in the form of sequential patterns from spatio-temporal point databases.

Keywords: Event Databases, Frequent Pattern, Interestingness Measures, Sequential Pattern, Spatio-Temporal

1. Introduction

The spatio-temporal data is the information related to the location of the object at a specific time. Many location based services are available with the increase in the wireless and mobile technology. Various kinds of technology is used to obtain this kind of data which is related to location and time of an object like Global Positioning System (GPS), Radio Frequency Identification (RFID) tags, sensors etc[1].

Major research work is going on for discovering sequential patterns from mobile object databases. Trajectory databases are one of the specialized databases of mobile object databases which store the traveling paths taken by vehicles, humans, animal herds etc. to move from one location to another. With regard to the trajectory databases, the paths which are being followed by many moving objects are found to be repeated very often. For example people going to colleges and schools, bus routes etc. where the routes followed by these people are different. For example, during weekdays the same routes might be followed but during weekends, it is difficult to predict the path. Determining such frequent routes is referred to as spatio-temporal sequential pattern mining from trajectory databases. This mining process helps us to manage the traffic, according to the frequency of the routes being used by the public[2].

Sequential patterns help in business decision making like advertising depending on the location, examining the illness of the patients, distributed systems, transportation,

Author for correspondence

geographic information systems etc. are the spatio-temporal applications[3].

Initially, data mining is introduced for mining frequent item sets and then it extended to sequential pattern mining, later to temporal data mining. The next step of data mining is towards the mining of spatio-temporal data, which is a challenging task. The sequential pattern mining algorithms proposed for the traditional transactional databases are not suitable for spatio-temporal sequential pattern mining. The challenges of the spatio-temporal data includes the size of the objects which do not have specific shape and borders, aggregating data which have different topology and information related to geometric boundaries, different data skewed to different objectives and correlation among the data[4].

The spatial and temporal incident is depicted as an event. A specific event occurs at a particular location and time. Events can be clustered by using its event type as a similarity feature.

The sequential pattern of the spatio-temporal data is a sequence of event types. Event type E_1 will lead to event type E_2, then to event type E_3 and so on, represented as a sequential pattern $E_1 \rightarrow E_2 \rightarrow E_3 \rightarrow \ldots\, E_n$. For example, the disease may spread from one to the other like from birds to animals, from animals to human and among humans. In this paper, an algorithm is proposed called as *Spatio-Temporal Sequential pattern Mining based on Support index and Event index* (STSMSE), for mining spatio-temporal data based on new metrics referred as support index and event index. The proposed method is called as Spatio-Temporal Sequential pattern Mining based on Support index and Event index (STSMSE).

The rest of the paper is organized as follows: Next section is presented with the research work carried out in this area. In Section 3, the proposed system, STSMSE is described and the algorithm is presented. The results of the experiments are discussed in Section 4 and finally the Section 5 concludes the paper.

The sequential pattern mining on transactional databases is first introduced[5]. Transactional databases have the patterns like <(b, c), (a, b), (c)> where (b, c), (a, b), (c) are three different transactions of a particular customer. This mining process is used to find the most frequent purchases of the customer. Sequential pattern mining applications includes DNA sequences, medical treatments, web log click streams and gene structures. The data in these applications cannot be transactionized as the spatio-temporal data are continuous in nature[6]. This reason makes the present sequential pattern mining algorithms not to be appropriate to spatio-temporal data mining[7].

The data considered is discrete forms of the pre-defined spatial locations and the process of mining these sequences is similar to the mining of the transactional databases. A spatio-temporal data mining framework is proposed to discover frequent periodic patterns[8]. The event sequences are not spatio-temporal series. The drawback of this algorithm is that it is prone to alterations of the pattern occurrences.

The pattern mining of vague sequences is more concentrated in[9]. Based on particular conditions, estimated frequent spatio-temporal sequential patterns are introduced. The breadth-first search and depth first search algorithms based dynamic programming is presented in order to compute the frequency of these patterns.

In general, all the items in a pattern or all the sequential patterns are given equal importance without any preference to any pattern or an item in a pattern which is not true in the case of real sequences. Hence, Weighted Sequential Pattern Mining (WSPAN) is proposed[10]. Each item in a pattern is associated with a parameter called weight which takes different values for different items. The anti-monotone property which states the infrequent sequential pattern leads to infrequent super sequential pattern. This anti-monotone property is broken by WSPAN. The inability of generating the sequential correlated patterns with support/weight affinity is the main disadvantage of WSPAN.

Weighted Interesting Sequential Pattern mining (WIS) is[11]. This algorithm is proposed based on the pattern growth method and defines two new measures referred as w-confidence and sequential s-confidence. These measures are used to mine the weighted interesting sequential patterns with similar levels of support and/or weight. WSRP-Miner algorithm is proposed to mine weighted sequential patterns from spatio-temporal databases[12]. An RSP-Miner algorithm to mine spatio-temporal sequential patterns using region-based framework to mine global patterns is proposed[13]. Also, proposed interestingness measures to extract reliable patterns.

Based on these observations from the literature, this paper proposes Spatio-Temporal Sequential pattern Mining based on Support index and Event index (STSMSE).

2. System Model

Consider the 2-dimensional space-time representation of the occurrences of events of various event types as shown in Figure 1 and the tabular representation of the same as shown in Table 1. Space is considered on x-axis and time on y-axis.

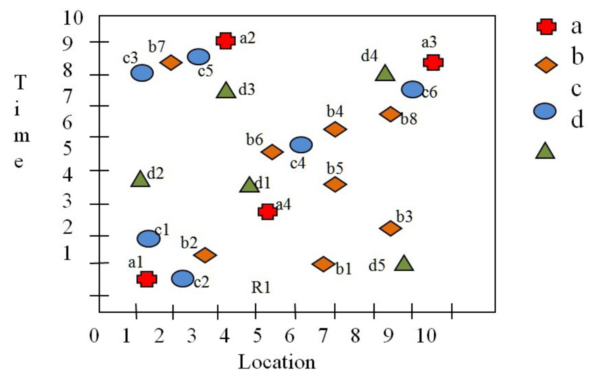

Figure 1. Occurrence of various event types.

Table 1. Sample spatio-temporal dataset

Event Type	Symbol	Event Set (Location, Time)
A		a1(1.2, 1.5), a2(3.2, 9.1), a3(8.4, 8.4), a4(4.1, 3.8)
B		b1(5.7,1.9), b2(2.8,2.2), b3(7.4,3.1), b4(6.1, 6.2), b5(6.0, 4.6), b6(4.3, 5.4), b7(2.2,8.3), b8(7.5,6.6)
C		c1(1.2,2.8), c2(2.1,1.5), c3(1.0, 8.0), c4(5.2, 5.8), c5(2.7,8.5), c6(8.0, 7.3)
D		d1(3.8,4.4), d2(1.0, 4.7), d3(3.1, 7.2), d4(7.1, 7.9), d5(7.9, 2.0)

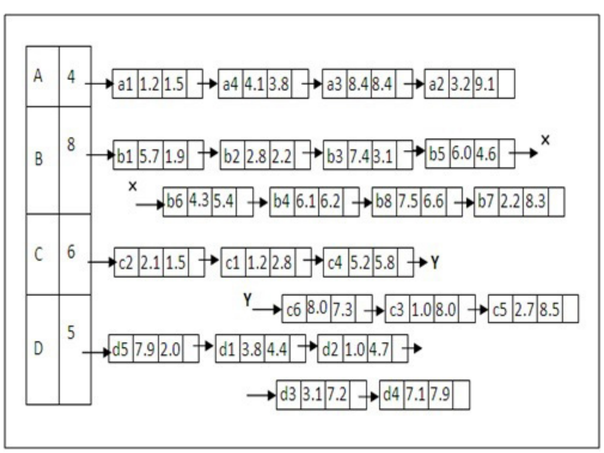

Figure 2. Data structure of the database at level-1.

The data is stored as temporally ordered list grouped based on event types to make the mining process easier. Figure 2 shows such representation of the data shown in Table 1. The dataset shown in Table 1 is stored in the form of a table with each entry as a linked list which is used to calculate the number of events easily as shown in Figure 2. The events are inserted in the linked list based on the time it took place. So, the event which is following the other event could be easily determined.

In the header node, the event type and the number of events are stored. Hence, it can be observed that the support index of every event type will be one initially. The event types which have the support index more than the threshold support index only are considered for further processing. As the events are stored according to the time they occur, the event following the other event could be identified easily. For example, a1 → {a4, a3, a2}, a4 → {a3, a2}, a3 → a2. This indicates the support of the event type sequence A → A as 6 (1 + 2 + 3). Similarly, the support of event type sequence B → B is calculated as (1 + 2 + 3 + 4 + 5 + 6 + 7) = 28, C → C as (1+2+3+4+5) = 15 and D → D as (1+2+3+4) = 10. In general if an event type E_i is having n events, then the support of the sequence $E_i \rightarrow E_i$ is given by $n\ (n-1)/2$. The other sequences also can be easily obtained by simple comparisons. The trace of algorithm to find sequence, the event sequences with respect to the sequence and the corresponding support is shown in Table 2.

Definition 1. Support Index*:* The support index is defined as the ratio of number of events in the event set to the total number of events of similar type.

Definition 2. Event Index*:* Event index is defined as the number of event sequences and is calculated as follows. Consider the event sequence $x \rightarrow Y$, where x is an event of type X and Y is the set of events which follow event x. Then event index is calculated as:

$$\sum_{j=1}^{n} x_j \,|Y \quad | \qquad \text{Eq (1)}$$

where n is the number of events of type X and $|Y|$ is the number of events in set Y and x_j is the number of times x appears in the set Y in the previous level. For example, for the sequence A → B, the event sequences are a1 → {b1, b2, b3, b4, b5, b6, b7, b8}, a2 → {b4, b5, b6, b7, b8}, hence the event index is 13 (1*8 + 1* 5 + 1*0 + 1* 0).

Then sequences of next level are determined as shown in Table 3 with the corresponding support index. Here the event sequences are determined as follows. Consider

Table 2. Support of the sequences at level-2

Sequence	Event sequences	Event Set	Support index	Event index
A → A	a1 → {a4, a3, a2} a4 → {a3, a2} a3 → a2	{a2, a3, a4}	0.75	6
A → B	a1 → {b1, b2, b3, b4, b5, b6, b7, b8} a2 → {b4, b5, b6, b7, b8}	{b1, b2, b3, b4, b5, b6, b7, b8}	1	13
A → C	a1 → {c1, c3, c4, c5, c6} a2 → {c3, c4, c5, c6} a3 → c5	{c1, c3, c4, c5, c6}	0.833	10
A → D	a1 → {d1, d2, d3, d4, d5} a2 → {d1, d2, d3, d4}	{d1, d2, d3, d4, d5}	1	9
B → A	b1 → {a2, a3, a4} b2 → {a2, a3, a4} b3 → {a2, a3, a4} b4 → {a2, a3} b5 → {a2, a3} b6 → {a2, a3} b8 → {a2, a3}	{a2, a3, a4}	0.75	17
B → B	b1 → {b2, b3, b4, b5, b6, b7, b8} b2 → {b3, b4, b5, b6, b7, b8} b3 → {b4, b5, b6, b7, b8} b4 → {b7, b8} b5 → {b4, b6, b7, b8} b6 → {b4, b7, b8} b8 → {b7}	{b2, b3, b4, b5, b6, b7, b8}	0.875	28
B → C	b1 → {c1, c3, c4, c5, c6} b2 → {c1, c3, c4, c5, c6} b3 → {c3, c4, c5, c6} b4 → {c3, c5, c6} b5 → {c3, c4, c5, c6} b6 → {c3, c4, c5, c6} b7 → {c5} b8 → {c3, c5, c6}	{c1, c3, c4, c5, c6}	0.833	29
B → D	b1 → {d1, d2, d3, d4, d5} b2 → {d1, d2, d3, d4} b3 → {d1, d2, d3, d4} b4 → {d3, d4} b5 → {d2, d3, d4} b6 → {d3, d4} b8 → {d3, d4}	{d1, d2, d3, d4, d5}	1	22
C → A	c1 → {a2, a3, a4} c2 → {a2, a3, a4} c3 → {a2, a3} c4 → {a2, a3} c5 → {a2} c6 → {a2, a3}	{a2, a3, a4}	0.75	13

Sequence	Event sequences	Event Set	Support index	Event index
C → B	c1 → {b3, b4, b5, b6, b7, b8} c2 → {b1, b2, b3, b4, b5, b6, b7, b8} c3 → {b7} c4 → {b4, b7, b8} c6 → {b7}	{b1, b2, b3, b4, b5, b6, b7, b8}	1	19
C → C	c1 → {c3, c4, c5, c6} c2 → {c1, c3, c4, c5, c6} c3 → {c5} c4 → {c3, c5, c6} c6 → {c3, c5}	{c1, c3, c4, c5, c6}	0.833	15
C → D	c1 → {d1, d2, d3, d4} c2 → {d1, d2, d3, d4, d5} c4 → {d3, d4} c6 → {d4}	{d1, d2, d3, d4, d5}	1	12
D → A	d1 → {a2, a3} d2 → {a2, a3} d3 → {a2, a3} d4 → {a2, a3} d5 → {a2, a3, a4}	{a2, a3, a4}	0.75	11
D → B	d1 → {b4, b5, b6, b7, b8} d2 → { b4, b6, b7, b8} d3 → {b7} d4 → {b7} d5 → {b2, b3, b4, b5, b6, b7, b8}	{b2, b3, b4, b5, b6, b7, b8}	0.875	18
D → C	d1 → {c3, c4, c5, c6} d2 → {c3, c4, c5, c6} d3 → {c3, c5, c6} d4 → {c3, c5} d5 → {c1, c3, c4, c5, c6}	{c1, c3, c4, c5, c6}	0.833	18
D → D	d1 → {d2, d3, d4} d2 → {d3, d4} d3 → {d4} d5 → {d1, d2, d3, d4}	{d1, d2, d3, d4}	0.8	10

the sequence (A → C) → A. Convert this sequence as A → C and C → A. In the sequence (A → C), c2 is missing in the event set. So, eliminate the event sequences related to c2 from the event sequences of C → A. Then the event sequences of (A → C) → A are obtained. Consider the example ((A→ B) → A) to calculate the event index. The event index of the sequence ((A→ B) → A) is 29 (1*3 + 1*3 + 1*3 + 2*2 + 2*2 + 2*2 + 2*2 + 2*2).

Similarly, the next levels of sequences can be easily obtained in the same procedure. Consider ((A → C) → D) → B. Write this sequence as ((A → C) → D) and D → B. d5 is missing in the event set of ((A → C) → D). Hence remove the event sequences related to d5 from the event sequences of D → B to obtain the event sequences of ((A → C) → D) → B. When the support is less than the threshold value then the corresponding sequence need not be considered for further process. In this way, the sequence patterns can be determined. After reaching the final level, only the interesting patterns are considered and the duplicates are eliminated.

3. STSMSE Algorithm

Based on the descriptions presented in Section 3, the following constitutes the algorithm for the proposed

Table 3. Support of the sequences at level-3 ((a → b) and (a → c) sequence extensions only)

Sequence	Event Sequences	Event Set	Support Index	Event Index
((A→ B) → A)	(a1 → b1) → {a4, a3, a2} (a1 → b2) → {a4, a3, a2} (a1 → b3) → {a4, a3, a2} {(a1 → b4), (a2 → b4)} → {a3, a4} {(a1 → b5), (a2 → b5)} → {a3, a4} {(a1 → b6), (a2 → b6)} → {a3, a4} {(a1→ b7), (a2 → b7)} → {a3, a4} {(a1 → b8), (a2 → b8)} → {a3, a4}	{a4, a3, a2}	0.75	29
((A→ B) → B)	b1 → {b2, b3, b4, b5, b6, b7, b8} b2 → {b3, b4, b5, b6, b7, b8} b3 → {b4, b5, b6, b7, b8} b4 → {b7, b8} b5 → {b4, b6, b7, b8} b6 → {b4, b7, b8} b8 → {b7}	{b2, b3, b4, b5, b6, b7, b8}	0.875	38
((A→ B) → C)	b1 → {c1, c3, c4, c5, c6} b2 → { c1, c3, c4, c5, c6} b3 → {c3, c4, c5, c6} b4 → {c3, c5, c6} b5 → {c3, c4, c5, c6} b6 → {c3, c4, c5, c6} b7 → {c5} b8 → {c3, c5, c6}	{c1, c3, c4, c5, c6}	0.833	44
((A→ B) → D)	b1 → {d1, d2, d3, d4, d5} b2 → {d1, d2, d3, d4} b3 → {d1, d2, d3, d4} b4 → {d3, d4} b5 → {d2, d3, d4} b6 → {d3, d4} b8 → {d3, d4}	{d1, d2, d3, d4, d5}	1	31
((A→ C) → A)	c1 → {a2, a3, a4} c3 → {a2, a3} c4 → {a2, a3} c5 → {a2} c6 → {a2, a3}	{a4, a3, a2}	0.75	18
((A → C) → B)	c1 → {b3, b4, b5, b6, b7,b8} c3 → {b7} c4 → {b4, b7, b8} c6 → {b7}	{b3, b4, b5, b6, b7,b8}	0.75	16
((A → C) → C)	c1 → {c3, c4, c5, c6} c3 → {c5} c4 → {c3, c5, c6} c6 → {c3, c5}	{c3, c4, c5, c6}	0.667	16
((A → C) → D)	c1 → {d1, d2, d3, d4} c4 → {d3, d4} c6 → {d4}	{d1, d2, d3, d4}	0.8	10

Spatio-Temporal Sequential pattern Mining based on Support index and Event index (STSMSE).

Algorithm STSMSE
Input:
Number of event types – n // the event types as X1, X2, …, Xn
Events of type X1 as $X1_1$, $X1_2$, $X1_3$, …, $X1_p$
Events of type X2 as $X2_1$, $X2_2$, $X2_3$, …, $X2_r$
Events of type Xn as Xn_1, Xn_2, Xn_3, …, Xn_s
// all the events are associated with its location and time details (TXn_i)

Output:
Number of sequences
Interesting sequences
Support index of each sequence
Event index of each sequence
begin
  for (i = 1 ; i <= n ; i++)
    for (l = 1 ; l <= n ; l++)
    for (j = 1 ; j < e ; j++) //consider number of events of type x_i as e
      for (k = 2 ; k <= e ; k++)
    if txi_j < txl_k then {
        $xi_j \rightarrow xl_k$
        add xi_k into event set
        compute support index
        compute event index
        seq++ // number of second level sequences
    }
  do { //computing higher level of sequences
    for (i =1 ; i <= seq ; i++)
    for (j = 1 ; j <= seq ; j++)
    for (k = 1 ; k <= seq ; k++)
    for (u = 1 ; u < e ; u++)
      for (v = 2 ; v <= e ; v++)
  if support index of xi → xj > $Thresh_{SI}$ &&
    event index of xi → xj > $Thresh_{EI}$ then
      if $xj_u \in$ the event set of xi → xj
      if txj_u < txk_v then {
      $xj_u \rightarrow xk_v$
      add xk_v into event set
      compute support index
        compute event index
        eliminate duplicates
          seq++
          }
  } while (all the possible sequences are found)
end

4. Results and Discussions

The proposed algorithm is evaluated using synthetic datasets generated using spatio-temporal data generator G-TERD[14]. The algorithm is implemented using java programming language. The parameters and the values of the corresponding used in the implementation are shown in Table 4.

Table 4. Simulation parameters

Parameter	Value
Event types	5
Average length of the sequence	10
Number of events in each event type	Random
Threshold value of support index	0.4
Threshold event index	40

The existing algorithm STS-Miner[15] is used to test and compare the performance of the proposed algorithm in terms of interested sequential patterns. The data during different time slots is captured. Sample of two time slots is shown in Figures 3 and 4. Figure 3 shows the event distribution in the workspace in 2-dimensional view for the time slot 10 and Figure 4 shows the distribution of events during time slot 80.

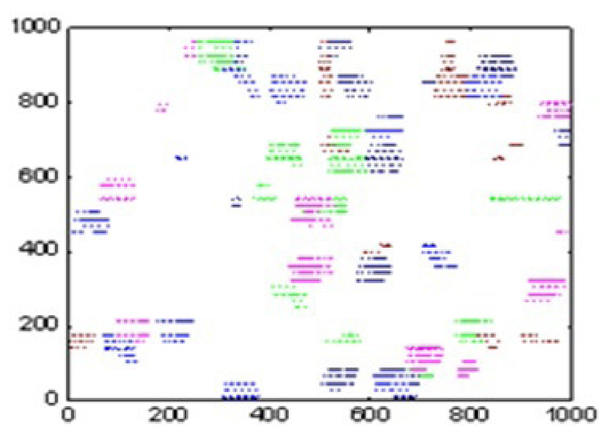

Figure 3. Event distribution in the workspace in 2-dimensional view at time slot 10.

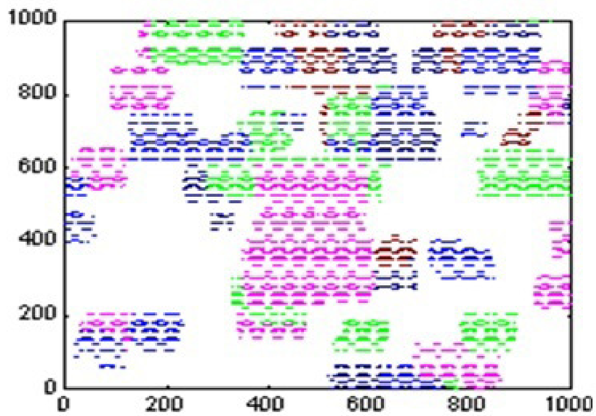

Figure 4. Event distribution in the workspace in 2-dimensional view at time slot 80.

The evaluation of the proposed approach in terms of execution time taken to completely generate the interested patterns for different sizes of data sets is performed and the comparison with respect to the STS-Miner algorithm is shown in Figures 5 and 6. The support index is considered to be 0.4 in the evaluation shown in Figure 5 and 0.2 in case of Figure 6. The event index is considered to be 40 in both the cases.

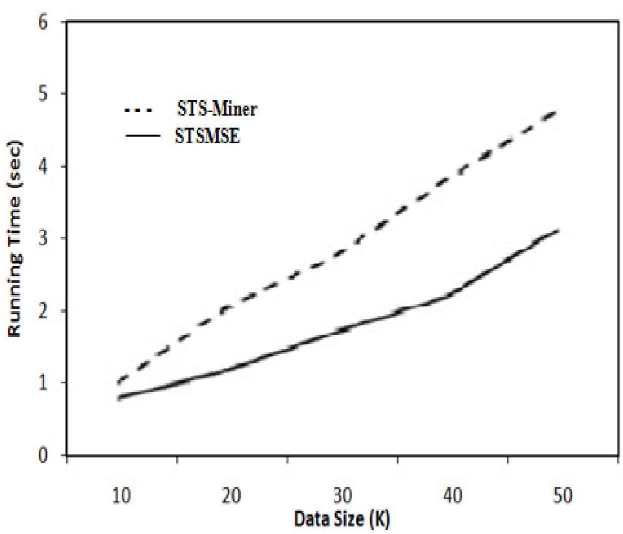

Figure 5. Mining efficiency with support index-0.4.

The performance of the STS-Miner is tested based on the specifications provided[15]. As the support index increases the number of sequences that can be generated will be reduced level by level and hence the execution time will appropriately reduce. It can be observed that the difference between the performance of the STS-Miner algorithm and the proposed system is much better when the support index is more as the proposed system generates only the interested patterns and avoids generating spurious patterns.

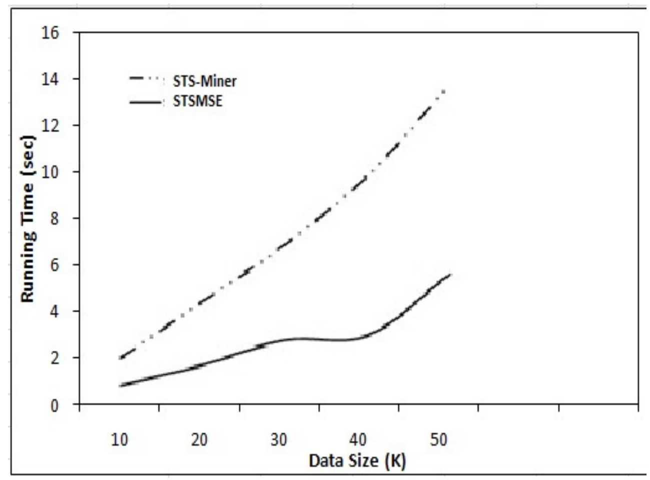

Figure 6. Mining efficiency with support index-0.2.

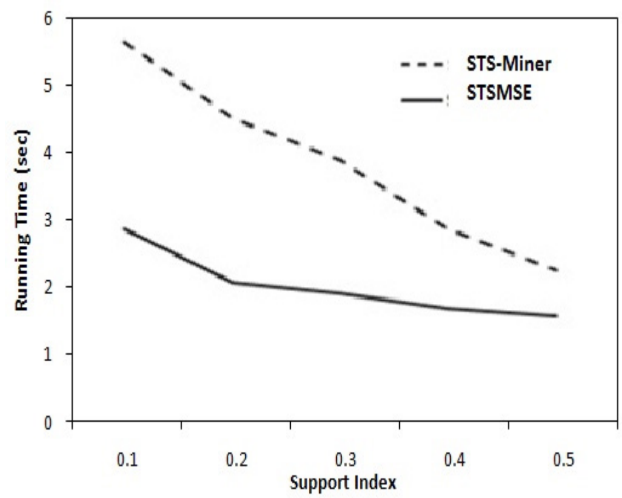

Figure 7. Mining efficiency with varying support index.

The performance of the STS-Miner algorithm and the proposed STSMSE algorithm in terms of execution time with respect to varying support index is shown in Figure 7. The experimental results shown in Figure 7 are carried out for various values of support index. The STSMSE algorithm shows better performance when compared to the STS-Miner algorithm when the support index is 0.4 and 0.5. Hence, the threshold of support index is considered to be 0.4. As stated before, there is a chance of generating more patterns and may be uninterested patterns if the support index is less and interested patterns might be lost if the support index is chosen to be high. Therefore, the threshold value needs to be chosen appropriately and carefully.

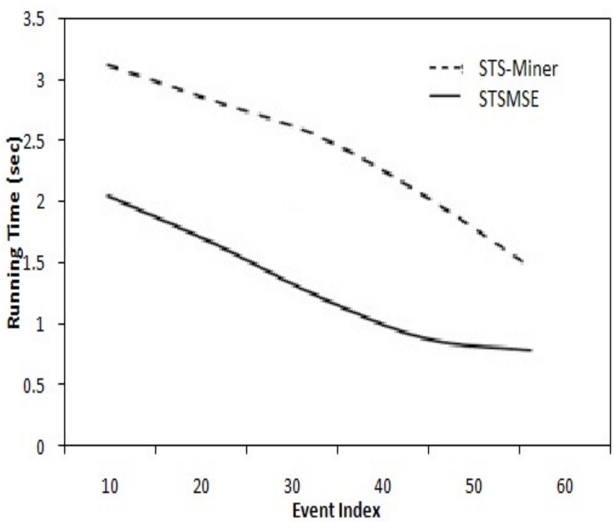

Figure 8. Mining efficiency for varying event index.

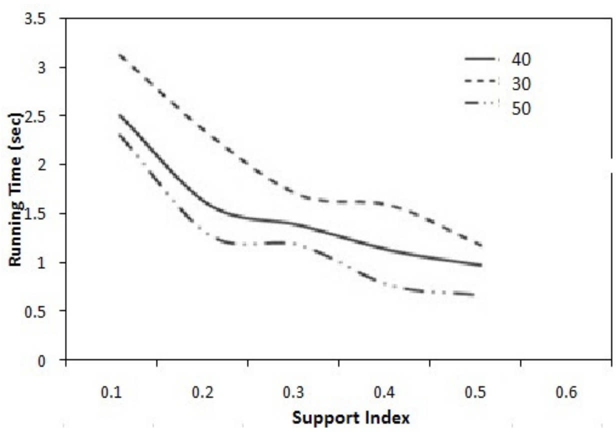

Figure 9. Mining efficiency for different event index values for varying support index.

The performance of the STS-Miner algorithm and the proposed system in terms of execution time with respect to varying event index is studies and is as shown in Figure 8. The support index considered is 0.4. The threshold value of the event index is considered to be 40 as the proposed system shows the better performance when compared to STS-Miner when the event index is 40. The event index threshold value must be chosen properly as in the case of support index.

The proposed system, STSMSE is tested for varying support index using three event index values and the results are shown in the Figure 9. Even though the performance improves for the high support index and high event index, these values need to be considered carefully based on proper analysis in order to generate all the interested patterns without missing any interested patterns or generate the spurious patterns. Very less execution time shows that the interested patterns might have been lost. The similar testing process is carried out for varying event index and three different values of support index and the evaluation results are shown in Figure 10.

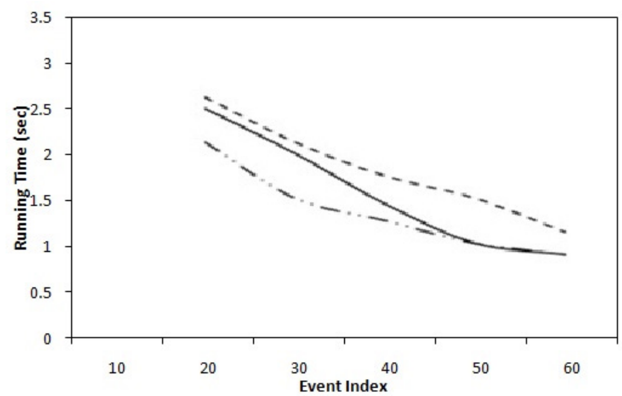

Figure 10. Mining efficiency for different support index for varying event index.

5. Conclusions

The research problem undertaken in this paper is to efficiently mine frequent spatio-temporal sequential patterns from spatio-temporal event datasets. An algorithm named STSMSE is proposed. Two frequency-based interestingness measures support index and event index which are defined to generate the interested patterns of spatio-temporal data. The spatio-temporal dataset is generated using G-TERD data generator. The experiments are carried out for varying sizes of data set, varying support index and varying event index values. The proposed STSMSE algorithm has been compared with STS-Miner algorithm and the experimental studies have proved that the proposed method generates patterns a magnitude times faster and is efficient in generating the frequent sequential patterns from the given spatio-temporal event database.

6. References

1. Verhein F. Mining complex spatio-temporal sequence patterns. Proceedings of 2009 SIAM International Conference

on Data Mining, Society for Industrial and Applied Mathematics; 2009 Apr. p. 605–16. Crossref
2. Cao H, Mamoulis N, Cheung DW. Mining frequent spatio-temporal sequential patterns. Proceedings of Fifth IEEE International Conference on Data Mining; 2005 Nov. p. 82–9. PMid: 15628913.
3. Roddick JF, Spiliopoulou M. A survey of temporal knowledge discovery paradigms and methods. IEEE Transactions on Knowledge and Data Engineering. 2002 Jul; 14(4):750–67. Crossref
4. Obulesu O, Reddy ARM. An enhanced tree mining algorithm for finding maximal periodic movements from spatio-temporal databases. Indian Journal of Science and Technology. 2016 Nov; 9(41):1–8. Crossref
5. Agrawal R, Srikant R. Mining sequential patterns. Proceedings of International Conference on Data Engineering (ICDE'95); Washington. 1995 Mar. p. 3–14. Crossref
6. Huang Y, Shekhar S, Xiong H. Discovering colocation patterns from spatial data sets: A general approach. IEEE Transactions on Knowledge and Data Engineering. 2004 Dec; 16(12):1472–85. Crossref
7. Tsoukatos I, Gunopulos D. Efficient mining of spatio-temporal patterns. Proceedings of International Symposium on Spatial and Temporal Databases; Springer Berlin Heidelberg. 2001 Jul. p. 425–42. Crossref
8. Mamoulis N, Cao H, Kollios G, Hadjieleftheriou M, Tao Y, Cheung DW. Mining, indexing and querying historical spatio-temporal data. Proceedings of Tenth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD '04); ACM, New York, NY, USA. 2004 Aug. p. 236–45. Crossref
9. Li Y, Bailey J, Kulik L, Pei J. Mining probabilistic frequent spatio-temporal sequential patterns with gap constraints from uncertain databases. Proceedings of IEEE 13th International Conference on Data Mining (ICDM); 2013 Dec. p. 448–57. Crossref
10. Yun U, Legget JJ. WSPAN. Weighted Sequential Pattern mining in large sequence databases. Proceedings of 3rd International IEEE Conference on Intelligent Systems; 2006 Sep. p. 512–7. Crossref
11. Yun U. WIS, Weighted Interesting Sequential pattern mining with a similar level of support and/or weight. ETRI Journal. 2007; 29(3):336–52. Crossref
12. Sunitha G, Reddy ARM. WRSP-Miner algorithm for mining weighted sequential patterns from spatio-temporal databases. Advances in Intelligent Systems and Computing (AISC). Springer. 2015; 379:309–17.
13. Sunitha G, Reddy ARM. A region-based framework for mining sequential patterns from spatio-temporal event databases. International Journal of Applied Engineering Research. 2014; 9(24):28161–75.
14. Tzouramanis T, Vassilakopoulos M, Manolopoulos Y. Generator for Time-evolving Regional Data (G-TERD). 2013 Feb; 6(3):207–31.
15. Huang Y, Zhang L, Zhang P. A framework for mining sequential patterns from spatio-temporal event data sets. IEEE Transactions on Knowledge and Data Engineering. 2008 Apr; 20(4):433–48. Crossref