Spatiotemporal Co-occurrence Rules

Karthik Ganesan Pillai, Rafal A. Angryk, Juan M. Banda, Tim Wylie, and Michael A. Schuh

k.ganesanpillai, angryk, juan.banda, timothy.wylie, michael.schuh@cs.montana.edu
Montana State University, Bozeman, Montana-59717

Abstract. Spatiotemporal co-occurrence rules (STCORs) discovery is an important problem in many application domains such as weather monitoring and solar physics, which is our application focus. In this paper, we present a general framework to identify STCORs for continuously evolving spatiotemporal events that have extended spatial representations. We also analyse a set of anti-monotone (monotonically non-increasing) and non anti-monotone measures to identify STCORs. We then validate and evaluate our framework on a real-life data set and report results of the comparison of the number candidates needed to discover actual patterns, memory usage, and the number of STCORs discovered using the anti-monotonic and non anti-monotonic measures.

Keywords: spatiotemporal events, extended spatial representations, spatiotemporal co-occurrence rules

1 Introduction

Spatiotemporal co-occurrence patterns (STCOPs) represent subsets of event types that occur together in both space and time. The discovery of spatiotemporal co-occurrence rules (STCORs) from STCOPs in data sets with evolving extended spatial representations is an important problem for application domains such as weather monitoring, astronomy, and solar physics, which is our application focus, and many others. Given a spatiotemporal (ST) database in which data objects are represented as polygons that continuously change their movement, shape, and size, our goal is to discover STCORs. In this paper we present a novel approach to our recent work initiated in [9], where we introduced the STCOPs mining problem, developed a general framework to discover STCOPs, and introduced an Apriori-based [1] STCOPs mining algorithm. This paper makes the following new contributions: 1) We introduce our novel Apriori-based STCOR-Miner algorithm; 2) We analyse a set of anti-monotonic and non anti-monotonic measures to discover STCORs; and 3) We verify the STCOR-Miner algorithm with a real-life data set, and provide experimental results reporting comparisons on the number of candidate pattern instances and actual pattern instances found for both types of measures, memory usage of the STCOR-Miner algorithm for our measures, and the number of STCORs discovered.

Since ST data mining is an important area, many algorithms have been proposed in literature for co-location mining in ST databases: topological pattern mining [13], co-location episodes [2], mixed drove mining [3], spatial co-location pattern mining from extended spatial representations [14], and interval orientation patterns [8]. However, none of these approaches focus on mining ST co-

occurrences from data represented as polygons evolving in time. Due to space constraints we do not give detailed information on these works; however, interested readers read our earlier paper published in [9] for detailed list of literature.

The rest of the paper is organized as follows: We review important concepts of modeling STCOPs for evolving extended spatial representations in Sec. 2. In Sec. 3, we present our proposed STCOR-Miner algorithm. Finally in Sec. 4 we present the experimental evaluation and summary of results.

2 Modeling STCOPs

Given a set of ST event types denoted $E = \{e_1, \ldots, e_M\}$, and a set of their instances $I = \{i_1, \ldots, i_N\}$, such that $M \ll N$. A STCOP is a subset of event types that co-occur in both space and time.

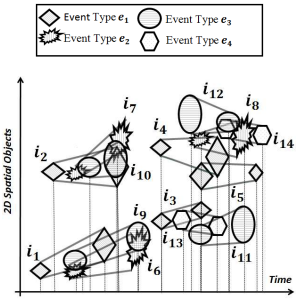

Fig. 1: An ST data set with 2D spatial objects evolving in time.

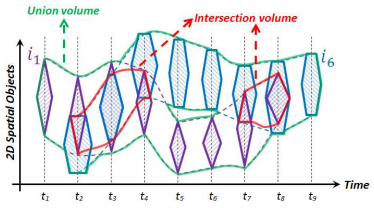

Fig. 2: Example of size-2 co-occurrence of spatiotemporal objects.

(a)

Instance ID	Event Type	Start Time (HH:MM)	End Time (HH:MM)
i_1	e_1	10:00	10:30
i_2	e_1	10:10	10:40
i_3	e_1	11:00	11:20
i_4	e_1	11:00	11:30
i_5	e_1	11:20	11:50
i_6	e_2	10:20	10:50
i_7	e_2	10:20	10:40
i_8	e_2	11:20	11:40
i_9	e_3	10:20	10:50
i_{10}	e_3	10:30	10:40
i_{11}	e_3	11:20	11:40
i_{12}	e_3	11:10	11:30
i_{13}	e_4	11:10	11:30
i_{14}	e_4	11:30	12:00

(b)

Instance of e_1	Instance of e_2	Time Instant ($t_s = 10$ minutes)	$Area(i_1 \cap i_6)$	$Area(i_1 \cup i_6)$
i_1	i_6	t_1 =10:00	0	60
i_1	i_6	t_2 =10:10	25	120
i_1	i_6	t_3 =10:20	95	115
i_1	i_6	t_4 =10:30	15	140
i_1	i_6	t_5 =10:40	0	150
i_1	i_6	t_6 =10:50	0	140
i_1	i_6	t_7 =11:00	16	130
i_1	i_6	t_8 =11:10	90	90
i_1	i_6	t_9 =11:20	0	60

$$cce_{i_1 i_6} = \frac{V(i_1 \cap i_6)}{V(i_1 \cup i_6)} = \frac{\sum_{j=t_1}^{t_9} t_s \times Area_j(i_1 \cap i_6)}{\sum_{j=t_1}^{t_9} t_s \times Area_j(i_1 \cup i_6)} = \frac{10 \times (0+25+\ldots+90+0)}{10 \times (60+120+\ldots+90+60)} = \frac{241}{1005} = 0.23$$

Fig. 3: (a) Table with temporal event information. (b) Example cce calculation.

In Fig. 1, we show an example data set, which we use to explain the concepts in detail. In Fig. 3 a), we show the *Instance ID*, *Event Type*, *Start Time*, and *End Time* of data instances from Fig. 1. This data set contains four event types. The event type e_1 has five instances, e_2 has three instances, e_3 has four instances, and e_4 has two instances. For simplicity, in this example we do not show the sequence of $2D$ shapes that reflect the ST evolution of our data.

A *size-k* STCOP is denoted as $SE = \{e_1, \ldots, e_k\}$, where $SE \subseteq E$, $SE \neq \emptyset$ and $1 < k \leq M$. We can have multiple *size-k* STCOPs derived from the set E, so to separate them we will use subscripts in future definitions, to denote uniqueness, i.e., $SE_i \neq SE_j$, and SE_i and SE_j contain different event types. *pat_instance* is a pattern instance of an STCOP SE_i, if *pat_instance* contains

an instance of *all* event types from SE_i. No proper subset of *pat_instance* is considered to be a pattern instance of SE_i. For example, $\{i_2, i_7, i_{10}\}$ is a *size*-3 ($k = 3$) pattern instance of co-occurrence $SE_i = \{e_1, e_2, e_3\}$ in the example data set. A collection of pattern instances of SE_i is a table instance of SE_i, and is denoted as *tab_instance*(SE_i). For example, $\{\{i_1, i_6, i_9\}, \{i_2, i_7, i_{10}\}\}$ is a *size*-3 ($k = 3$) *tab_instance*($SE_i = \{e_1, e_2, e_3\}$) in the example data set. An STCOR is of the form $SE_i \Rightarrow SE_j(cce, p, cp)$, where SE_i and SE_j are STCOPs, such that $SE_i \neq SE_j$, and parameters *cce*, p, and *cp* characterize the rule in the following manner: (a) *cce* is the ST co-occurrence coefficient and it indicates the strength of ST relation's occurrence that is investigated. The STCOPs mining algorithm [9] uses the ST relation Overlap for *cce*. To distinguish the ST relation from the purely spatial one, we will use, capital letter in the name of the former. Some examples of ST Overlap are $\{i_1, i_6\}$, $\{i_2, i_7\}$, and $\{i_7, i_{10}\}$ in Fig. 1. (b) p is the *prevalence measure*. The *prevalence measure* emphasizes how interesting the ST co-occurrences are based on prevalence. In our investigation, we used the participation index (*pi*), proposed in [6], as the *prevalence measure*. The *pi* is monotonically non-increasing when the size of the STCOP increases, which is exploited for computational efficiency [6]. (c) *cp* is the conditional probability [6] of our STCOR. The *cp* gives the confidence of the STCOR $SE_i \Rightarrow SE_j$.

2.1 Measures

Our STCOPs algorithm [9] uses the ST co-occurrence coefficient to calculate *cce*. The ST co-occurrence coefficient is closely related to the coefficient of areal correspondence (CAC) proposed in [12]. CAC is computed for any two or more overlapping polygons as the area of their intersection, divided by the area of union (spatial version of *Jaccard* measure [7]). In [9], we extend CAC to three dimensions (two dimensions correspond to space and the third dimension corresponds to time) and calculate the ST co-occurrence coefficient using ST volumes: (1)The Intersection volume of a *size-k* pattern instance, denoted $V(i_1 \cap i_2, \ldots, i_{k-1} \cap i_k)$, is the volume of the three dimensional object representing the Intersection of the trajectories of all instances involved in a given pattern instance. (2)The Union volume of a *size-k* pattern instance, denoted as $V(i_1 \cup i_2, \ldots, i_{k-1} \cup i_k)$, is the volume of the three dimensional object representing the Union of the trajectories of all instances involved in a given co-occurrence.

2.2 Co-occurrence Coefficient *cce*

We use the *cce* as our measure to assess the strength of the ST relation Overlap. *cce* is calculated for a *size-k* pattern instance as the ratio $J = \frac{V(i_1 \cap i_2, \ldots, i_{k-1} \cap i_k)}{V(i_1 \cup i_2, \ldots, i_{k-1} \cup i_k)}$. The symbol J represents the *Jaccard* measure [7] (Fig. 2), which is commonly accepted by data mining practitioners [7, 11]. Computing the *cce* for extended ST representations such as evolving polygons is not a trivial task. In Fig. 2, we show the movement of a pair of instances of two event types that change sizes and directions across different time instances. We also show the region of Intersection and the region of Union at different time slots. If we assume that instances $\{i_1, i_6\}$, in our example data set (Fig. 1 and Fig. 3 a)), have ST Intersection volume $V(i_1 \cap i_6) = 241$ and a ST Union volume $V(i_1 \cup i_6) = 1005$, then, $cce = \frac{V(i_1 \cap i_6)}{V(i_1 \cup i_6)} = 0.23$ (see the notes under Fig. 3 b) for detailed calculations).

Coefficients	Formula	Anti-monotonic
Jaccard (J)	$\frac{V(i_1 \cap i_2,\ldots,i_{k-1} \cap i_k)}{V(i_1 \cup i_2,\ldots,i_{k-1} \cup i_k)}$	Yes
Overlap (OMAX)	$\frac{V(i_1 \cap i_2,\ldots,i_{k-1} \cap i_k)}{\max(V(i_1),\ldots,V(i_k))}$	Yes
Cosine (N)	$\frac{\sqrt[k]{k} \times V(i_1 \cap i_2,\ldots,i_{k-1} \cap i_k)}{\sqrt[k]{\sum_{j=1}^{k} V(i_j)^k}}$	No
Dice (D)	$\frac{k \times V(i_1 \cap i_2,\ldots,i_{k-1} \cap i_k)}{\sum_{j=1}^{k} V(i_j)}$	No
Cosine (C)	$\frac{V(i_1 \cap i_2,\ldots,i_{k-1} \cap i_k)}{\sqrt[k]{V(i_1) \times V(i_2),\ldots,V(i_{k-1}) \times V(i_k)}}$	No
Overlap (OMIN)	$\frac{V(i_1 \cap i_2,\ldots,i_{k-1} \cap i_k)}{\min(V(i_1),\ldots,V(i_k))}$	No

Fig. 4: Measures evaluating ST relation Overlap (cce).

Although, we have shown calculation of our *cce* using the *Jaccard* measure, in this paper we would like to investigate alternative measures in detail. We analyze six different measures (denoted in the first column of Fig. 4) to assess the strength of ST Overlap.

2.3 Prevalence of STCOPs

The participation index $pi(SE_i)$ of an STCOP SE_i is $min_{j=1}^{k} pr(SE_i, e_j)$, where k is the size of the pattern (cardinality of SE_i), and the participation ratio $pr(SE_i, e_j)$ for a event type e_j is the fraction of the total number of instances of e_j forming ST co-occurring instances in SE_i. For example, from Fig. 1 and Fig. 3 a) we can see that the pattern instances of $SE_i = \{e_1, e_2, e_3\}$ are $\{\{i_1, i_6, i_9\}, \{i_2, i_7, i_{10}\}\}$. Only two (i_1, i_2) out of five instances of the event type e_1 participate in $SE_i = \{e_1, e_2, e_3\}$. So, $pr(\{e_1, e_2, e_3\}, e_1) = 2/5 = 0.40$. Similarly $pr(\{e_1, e_2, e_3\}, e_2) = 2/3 = 0.67$, and $pr(\{e_1, e_2, e_3\}, e_3) = 2/4 = 0.50$. Therefore the participation index of STCOP $SE_i = \{e_1, e_2, e_3\}$ is $pi(\{e_1, e_2, e_3\}) = min(0.40, 0.67, 0.50) = 0.40$. The STCOP SE_i is a *prevalent pattern* if it satisfies a user-specified minimum participation index threshold pi_{th}. In our example above, if the minimum threshold is set to $pi_{th} = 0.3$, then the STCOP $SE_i = \{e_1, e_2, e_3\}$ is a *prevalent pattern*. The conditional probability $cp(SE_i \Rightarrow SE_j)$ of an STCOR $SE_i \Rightarrow SE_j$ is the fraction of pattern instances of SE_i that satisfies the ST relation strength indicator *cce* to some pattern instances of SE_j. It is computed as, $\frac{|\pi_{SE_i}(tab\_instance(\{SE_i \cup SE_j\}))|}{|tab\_instance(\{SE_i\})|}$, where π is the relational projection operation with duplicate elimination [6].

2.4 Problem Statement

Inputs:

1. A set of ST event types $E = \{e_1, e_2, \ldots, e_M\}$ over a common ST framework.
2. A set of N ST event instances $I = \{i_1, i_2, \ldots, i_N\}$, each $i_j \in I$ is a tuple *<instance-id, event type, sequence of <2D shape, time instant> pairs>*.
3. A user-specified ST thresholds for: cce_{th}, pi_{th}, and cp_{th}.
4. A time interval of data sampling (t_s)(the same for all events).

Output: *Find the complete and correct result set of STCORs with* $cce > cce_{th}$, $pi > pi_{th}$, *and* $cp > cp_{th}$.

3 STCOR-Miner Algorithm

In this section we introduce our STCOR-Miner algorithm to mine STCORs from data sets with extended spatial representations that evolve over time. Fig. 5 gives the pseudocode of our STCOR-Miner algorithm. The inputs and outputs are as defined in Sec. 2.5. In the algorithm, steps 1 and 2 initialize the parameters and data structures, steps 3 through 10 give an iterative process to mine STCORs and step 11 returns a union of the results of the STCORs (rules of all sizes). Steps 3 through 10 continue until there is no candidate STCOPs to be mined. Next we explain the functions in the algorithm.

Variables :
(1) k the co-occurrence size (Sec. 2).
(2) C_k: a set of candidates for size-(k) STCOPs derived from *size-($k-1$)* prevalent STCOPs.
(3) T_k: set of instances of size-(k) ST co-occurrences (Sec. 2).
(4) P_k: a set of *size-(k)* prevalent STCOPs derived from *size-(k)* candidate STCOPs (Sec. 2).
(5) R_k: a set of ST co-occurrence rules derived from *size-(k)* prevalent STCOPs (Sec. 2).
(6) R_{final}: union of all STCORs (rules of all sizes).
Algorithm :
1 k=1; C_1=E; $P_1 = E$; $R_{final} = \emptyset$;
2 $T_1 = gen\_loc(C_1, I, t_s)$;
3 **while** ($P_k \neq \emptyset$) {
4 $C_{(k+1)} = gen\_candidate\_coocc(P_k)$;
5 $T_{(k+1)} = gen\_tab\_ins\_coocc(C_{(k+1)}, cce_{th})$;
6 $P_{(k+1)} = pre\_prune\_coocc(C_{(k+1)}, pi_{th})$;
7 $R_{(k+1)} = gen\_rules\_coocc(P_{(k+1)}, cp_{th})$;
8 $R_{final} = R_{final} \cup R_{(k+1)}$;
9 $k = k + 1$;
10 }
11 return R_{final};

Fig. 5: STCOR-Miner Algorithm

Step 2: In step 2, the evolution of instances of our ST events from their start time slot is projected using t_s (to increment the number of time steps between time slots). The combination of the event instance ID and time step will allow us to identify an event at a particular moment. Step 4 generates candidate STCOPs in this step. We use an Apriori-based [1] approach to generate the candidates of size-($k+1$) using ST co-occurring prevalent patterns of *size-(k)* for anti-monotonic measures. Hence, prevalent patterns of *size-(k)*, which satisfy the user-specified threshold value of a minimum participation index pi_{th}, are used to generate candidate patterns of size-($k+1$). However, for correctness, we generate candidate patterns of *size-(k+1)* from all *size-k* patterns for non anti-monotonic measures (Fig. 4). Step 5 generates table instances for candidate patterns of *size-($k+1$)*. Pattern instances for each table instance can be generated by an ST query. The geometric shapes of the instances at each time step are saved, as these geometric shapes will be used for finding the *cce* of STCOPs of size three or more. Pattern instances that have a $cce < cce_{th}$ are deleted from the table instance, if the measure used to calculate *cce* has the anti-monotonic property (i.e., J and $OMAX$). However, for non anti-monotonic measures (Fig. 4), pattern instances that do not have any volume resulting from intersection of trajectories of the event instances, are deleted from the table instances. Step 6 discovers filtered *size-($k+1$)* STCOPs by pruning $C_{(k+1)}$ that have $pi < pi_{th}$. However, please

note, if the measure used to calculate *cce* is non anti-monotonic, we will not prune the candidates based on pi_{th} value. Step 7 generates STCORs. A set of STCORs R that have *cp* greater than cp_{th} of *size-(k+1)* is generated from $P_{(k+1)}$ [6] for anti-monotonic measures. However, for non anti-monotonic measures we generate rules that have *cp* greater than cp_{th} from patterns of $P_{(k+1)}$ that have *pi* value greater than pi_{th} (note, this check is neccessary for non anti-monotonic measures because we do not prune away patterns, see Step 6). Step 8 calculates the union of rules R_{final} and R_{k+1}. The algorithm runs iteratively until no more STCOPs can be generated for anti-monotonic measures. However, for non anti-monotonic measures all the patterns are generated and in a post processing step only the patterns that satisfy pi_{th} are reported. Finally, the algorithm returns the union of all the found STCORs in Step 11.

4 Experimental Evaluation and Conclusions

In our experiments, we use a real-life data set from the solar physics domain ([5],[10]) which contains evolving instances of six different solar event types observed on 01/01/2012. We investigate STCOR-Miner with the measures to accurately capture the STCORs of solar event types represented as evolving polygons. Moreover, an interesting ordering relation on the selectivity of the boolean versions of J, $OMAX$, N, D, C, and, $OMIN$ measures is shown in [4]. We show the ordering relation of the measures on real numbers in our experiments. For all experiments, the cce_{th}= 0.01, pi_{th}= 0.1, cp_{th}= 0.6, and t_s= 30 minutes.

4.1 Conclusion on the Count of Pattern Instances

We first investigated the number (no.) of candidate *pat_instance*'s that are used to generate the pattern instances that satisfy the threshold cce_{th} for anti-monotonic and non anti-monotonic measures. In Fig. 6 (a), we show the number of pattern instances used by STCOR-Miner with anti-monotonic measures for different pattern sizes. In Fig. 6 (a), J-BCCE (OMAX-BCCE) represent the

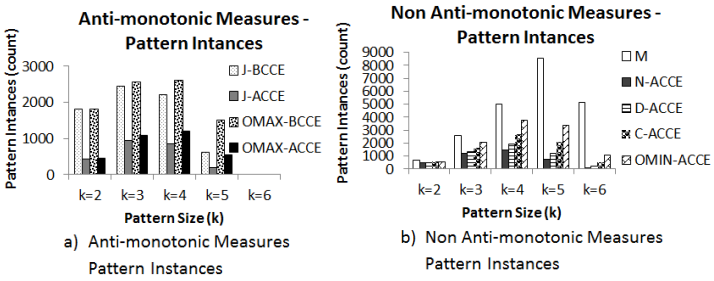

Fig. 6: Pattern Instances.

no. of candidate *pat_instance*'s generated with measure J ($OMAX$), and J-ACCE (OMAX-ACCE) represent the no. of *pat_instance*'s after filtering out the candidates that do not satisfy the threshold cce_{th} in the STCOR-Miner algorithm. From Fig. 6 (a), we can observe that the no. of candidate *pat_instance*'s and actual patterns for the measures J and $OMAX$ follows the ordering $J \leq OMAX$. In Fig. 6 (b), we show the no. of *pat_instance*'s used by the STCOR-Miner algorithm with non anti-monotonic measures for different pattern sizes.

In Fig. 6 (b), M represents the no. of candidate *pat_instance*'s generated (i.e., $V(i_1 \cap i_2, \ldots, i_{k-1} \cap i_k) > 0$), and N-ACCE, D-CCE, C-ACCE, and OMIN-ACCE represent the no. of *pat_instance*'s that satisfy the threshold cce_{th} in the STCOR-Miner algorithm (i.e., the actual patterns that are reported on the output). In comparison to the anti-monotonic measures, we keep the candidate *pat_instance*'s that do not satisfy the threshold cce_{th} for the N, D, C, and $OMIN$ measures. Moreover, from Fig. 6 (b) we can observe that the no. of *pat_instance*'s that satisfy the threshold cce_{th} for the measures N, D, C, and $OMIN$ follows the order $N \leq D \leq C \leq OMIN$.

4.2 Conclusion on Memory Usage

We now investigate the hard-drive memory usage of the STCOR-Miner algorithm candidate table instances with all the pattern instances generated, and candidate table instances after filtering the pattern instances that do not satisfy cce_{th}. In Fig. 7 (a), we show the memory usage of table instances generated with anti-monotonic measures for different pattern sizes: J-BCCE represents the memory usage of table instances for all pattern instances generated, and J-ACCE represents the memory usage after filtering out the pattern instances that do not satisfy the threshold cce_{th}. As expected, from Fig. 7 (a) we can observe that there is a drop in the memory usage after the pattern instances are filtered by applying the threshold cce_{th}. Furthermore, we can observe that the memory usage J is more expensive than $OMAX$ due to cost of union geometries needed for the calculation of J. In Fig. 7 (b), we show the memory usage of table instances used by the STCOR-Miner algorithm with non anti-monotonic measures for different pattern sizes. M represents the memory usage of table instances for all the pattern instances generated (i.e., $V(i_1 \cap i_2, \ldots, i_{k-1} \cap i_k) > 0$), and N-ACCE, D-CCE, C-ACCE, and OMIN-ACCE represent the memory usage of table instances with pattern instances that satisfy the threshold cce_{th} in the STCOR-Miner algorithm with the measures N, D, C, and $OMIN$, respectively. However, in comparison to the J and $OMAX$ we do not filter candidate pattern instances for the non anti-monotonic measures. Thus, for N, D, C, and $OMIN$, the no. of candidate pattern instances used to generate patterns of higher sizes is greater than J and $OMAX$.

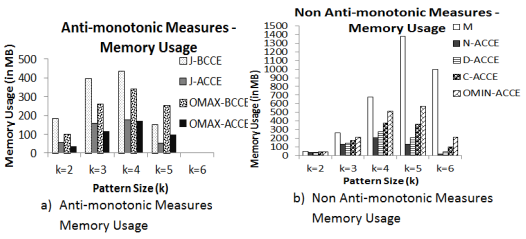

Fig. 7: Memory usage used by candidate table instances

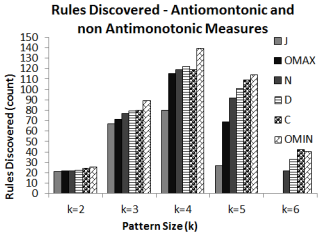

Fig. 8: Number of rules discovered

4.3 Conclusion on Discovered Rules

Finally, we investigate the no. of rules generated using the anti-monotonic and non anti-monotonic measures with the STCOR-Miner algorithm (Fig. 8). The

importance of analyzing different measures is shown here in order to accurately capture the ST characteristics of different solar events. For instance, J acts similar to measure D [7]; however, it penalizes objects with smaller Intersection volumes. It gives much lower values than D to objects which have a small Intersection volume - giving a penalty to some of our events that are small in the area and short-lasting. Similarly, the measures $OMAX$ and N also penalize objects with smaller Intersection volume. The measure $OMIN$ [7] gives a value of one if an object is totally contained with another object. We could say that it reflects inclusion, which benefits the objects that are almost equal in space and time. The measure C [7] is more resistant to the size of the objects, making it more appropriate to data sets that contain event types with different life spans and areas (sizes).

5 Acknowledgements

This work was supported by two National Aeronautics and Space Administration (NASA) grant awards, 1) No. NNX09AB03G and 2) No. NNX11AM13A.

References

1. R. Agrawal, T. Imieliński, and A. Swami. Mining association rules between sets of items in large databases. *SIGMOD Rec.*, 22(2):207–216, June 1993.
2. H. Cao, N. Mamoulis, and D. W. Cheung. Discovery of collocation episodes in spatiotemporal data. In *The 6'th Intern. Conf. on DataMining,823-827*, DC, 2006.
3. M. Celik, S. Shekhar, J. P. Rogers, J. A. Shine, and J. S. Yoo. Mixed-drove spatio-temporal co-occurence pattern mining: A summary of results, 119-128. In *The 6'th Intern. Conf. on DataMining*, DC, 2006.
4. L. Egghe and C. Michel. Strong similarity measures for ordered sets of documents in information retrieval. *Inf. Process. Manag.*, 38(6):823–848, Nov. 2002.
5. HEK. http://www.lmsal.com/isolsearch, Jan. 2012.
6. Y. Huang, S. Shekhar, and H. Xiong. Discovering colocation patterns from spatial data sets: a general approach,1472-1485. *Trans. on Know. and Data Eng.*, 2004.
7. C. D. Manning and H. Schütze. *Foundations of statistical natural language processing*. MIT Press, Cambridge, MA, USA, 1999.
8. D. Patel. Interval-orientation patterns in spatio-temporal databases, 416-431. In *DEXA (1)*, 2010.
9. K. G. Pillai, R. A. Angryk, J. M. Banda, M. A. Schuh, and T. Wylie. Spatio-temporal co-occurrence pattern mining in data sets with evolving regions, 805-812. In *ICDM Workshops*, 2012.
10. M. A. Schuh, R. A. Angryk, K. G. Pillai, J. M. Banda, and P. C. Martens. A large-scale solar image dataset with labeled event regions. In *Int. Conf. on Image Processing (ICIP)*, 2013.
11. P.-N. Tan, M. Steinbach, and V. Kumar. *Introduction to Data Mining, (First Edition)*. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 2005.
12. P. Taylor. *Quantitative Methods in Geography: An Introduction to Spatial Analysis*. Houghton Mifflin, 1977.
13. J. Wang, W. Hsu, and M. L. Lee. A framework for mining topological patterns in spatio-temporal databases, 429-436. CIKM '05, New York, 2005. ACM.
14. H. Xiong, S. Shekhar, Y. Huang, V. Kumar, X. Ma, and J. S. Yoo. A framework for discovering co-location patterns in data sets with extended spatial objects. In *SDM*, 2004.