7-2008

Searching correlated objects in a long sequence

Ken C. K. LEE

Wang-chien LEE

Donna Peuquet

Baihua ZHENG
Singapore Management University, bhzheng@smu.edu.sg

DOI: https://doi.org/10.1007/978-3-540-69497-7_28

Citation

LEE, Ken C. K.; LEE, Wang-chien; Peuquet, Donna; and ZHENG, Baihua. Searching correlated objects in a long sequence. (2008). *Scientific and Statistical Database Management: 20th International Conference, SSDBM 2008, Hong Kong, July 9-11, 2008: Proceedings*. 5069, 436-454. Research Collection School Of Information Systems.
Available at: https://ink.library.smu.edu.sg/sis_research/378

Searching Correlated Objects in a Long Sequence

Ken C.K. Lee[1], Wang-Chien Lee[1], Donna Peuquet[1], and Baihua Zheng[2]

[1] Pennsylvania State University, University Park, PA16802, USA
cklee@cse.psu.edu, wlee@cse.psu.edu, djp11@psu.edu
[2] Singapore Management University, Singapore
bhzheng@smu.edu.sg

Abstract. *Sequence*, widely appearing in various applications (e.g. event logs, text documents, etc) is an ordered list of objects. Exploring correlated objects in a sequence can provide useful knowledge among the objects, e.g., event causality in event log and word phrases in documents. In this paper, we introduce *correlation query* that finds correlated pairs of objects often appearing closely to each other in a given sequence. A correlation query is specified by two control parameters, *distance bound*, the requirement of object closeness, and *correlation threshold*, the minimum requirement of correlation strength of result pairs. Instead of processing the query by scanning the sequence multiple times, that is called *Multi-Scan Algorithm (MSA)*, we propose *One-Scan Algorithm (OSA)* and *Index-Based Algorithm (IBA)*. OSA accesses a queried sequence once and IBA considers correlation threshold in the execution and effectively eliminates unneeded candidates from detail examination. An extensive set of experiments is conducted to evaluate all these algorithms. Among them, IBA, significantly outperforming the others, is the most efficient.

1 Introduction

Many datasets, such as event logs and textual documents, organize data objects in an ordered list, i.e., *sequence*. Both the data objects and their positions are captured by the sequence where the closeness of two objects in a sequence implies their relationships. We refer objects a and b as *correlated* if they often occur closely to each other. Efficiently identifying correlated objects has a large application base. For example, finding products likely to be selected by the same customers some time after their purchase of certain products is a key to the success of recommendations [4]. Detecting events usually happened some time after some others from an event log can provide hints to determine event causality in an event analysis [8]. Figuring out words frequently appearing together in documents will help identifying key phrases used and providing better understanding of documents [6].

Motivated by the importance of identifying correlated objects in a sequence, we introduce *correlation query* in this paper. Its definition is formalized in Section 3. In a sequence, objects can be classified into *object sets*, i.e., subsets of objects categorized by certain properties of interests. Two objects are said to be close if their distance along the sequence does not exceed a threshold, specified by a query parameter *distance bound*. A correlation query is to retrieve object set pairs that have a large portion of objects close to each other. Another query parameter *correlation threshold* is specified that two object sets (that we call them an *object set pair*) satisfy a correlation query when their correlation coefficient is greater than the specified threshold. A correlation coefficient (defined by cosine function in this paper) measures the strength of correlation in two object sets whose objects are closely located. A correlation query finds all the satisfied correlated object set pairs from a sequence.

Efficiently processing a correlation query is challenging because the number of close objects is subject to the specified distance bound. The most intuitive way is to scan the queried sequence to measure the numbers of close objects, and then determine the correlation coefficients. Following this idea, we propose a scan-based algorithm, namely *Multi-Scan Algorithm (MSA)*, to serve as the baseline algorithm. It examines a pair of candidate object sets in each scan. Suppose there are n objects sets. MSA scans the whole sequence $\binom{n}{2}$ times that is very time consuming. To overcome the shortcoming of MSA, we propose another scan-based algorithm, One-Scan Algorithm (OSA), which finishes the query within one sequence scan. Scan-based algorithms, however, have serious performance deterioration when the queried sequence is very long. Since only object set pairs with high correlation coefficients are needed and worth investigation, we propose *Index-Based Algorithm (IBA)*, which builds an index for every object set to capture the positions of mapped objects in the sequence. Given two indices, the number of close objects can be determined by merging the two indices and thus the correlation coefficient is calculated. Several effective optimization techniques, such as candidate screening, group matching, and early termination, are proposed to further boost up the search performance.

We conduct an extensive set of experiments on both synthetic and real datasets to evaluate the proposed search algorithms. MSA and OSA perform stably with various sequence properties and OSA significantly outperforms MSA. IBA runs even much faster than OSA due to effectiveness of optimization techniques, especially when search criteria is strict (i.e., a large correlation threshold and a small distance bound) and the cardinalities of object sets differ a lot. We also discuss some variants of correlation query including constrained correlation query, position correlation query and correlation spectrum query. Our contributions in this paper are summarized as below:

1. We introduce a new query type, called correlation query, which retrieves correlated object set pairs based on specified distance bound and correlation threshold.
2. We analyze the characteristics of correlation query and propose two scan-based algorithms, namely *Multi-Scan Algorithm* (MSA) and *One-Scan Algorithm* (OSA).
3. We also propose *Index-Based Algorithm* (IBA), that indexes objects in a sequence, and employs optimization techniques for better search performance.

4. We introduce variants of correlation query including constrained correlation query, position correlation query and correlation spectrum query.
5. We conduct an extensive set of experiments to evaluate the performance of the proposed algorithms. The results indicate that IBA performs better than the others and it is the most efficient algorithm for this correlation query.

The remainder of the paper is organized as follows. Section 2 reviews related work about correlation analysis in related domains. Section 3 formalizes the correlation query and discusses algorithm design criteria. Section 4 details our proposed algorithms. Section 5 discusses variants of correlation query. Section 6 evaluates the performance of proposed algorithms and presents our results. Section 7 concludes this paper.

2 Related Work

Subject to application needs and data characteristics, the definitions and measurements of object correlation are different [5,10]. In statistics, correlation measures the strength and direction of a linear relationship between two random variables (e.g. education and income). Two random variables are correlated when the values of both variables increase (or decrease) with similar amplitude simultaneously. In data mining where transaction databases are usually considered, finding association among objects is one of the most important search. Result objects are those frequently appearing in same transactions [3]. Association mining finds which pairs or groups of objects are often included in same transactions.

	y	$\bar{y}$	
x	f_{xy}	$f_{x\bar{y}}$	f_x
$\bar{x}$	$f_{\bar{x}y}$	$f_{\bar{x}\bar{y}}$	$f_{\bar{x}}$
	f_y	$f_{\bar{y}}$	N

Fig. 1. A 2×2 contingency table for x and y

Finding correlated objects is fundamentally different from association mining that correlated pairs of objects may not have high frequencies but strong correlations [13]. Currently, there are a number of correlation metrics (e.g., lift, cosine, χ^2 and Pearson's correlation coefficient) defined to quantify the strength of object correlation [10]. Most of the metrics are developed based on contingency table. Figure 1 shows a 2×2 contingency table for two objects, x and y where f_{xy} is the frequency (i.e., the counts) of baskets containing both x and y at the same time, and $f_{\bar{x}\bar{y}}$ is one containing neither x nor y. $f_{x\bar{y}}$ ($f_{\bar{x}y}$) represents the number of baskets containing x (or y) only. Based on these frequencies, x and y are highly correlated if f_{xy} is relatively large to f_x and f_y. To perform such correlation analysis, all the frequencies have to be collected in advance.

There are several related research studies exploring correlation in sequences, but they are different from what we focus in this paper. Existing studies concern the correlation between individual sequences from a pool of sequences [9,14], while our work is to explore a *single long* sequence and find out the correlation among objects according to *distance bound* a query parameter. Subject to the setting of distance bound, the frequency of close objects is not fixed. Thus, counting the number of close objects in prior is no longer feasible. Thus, new and efficient algorithms that can quickly identify correlated objects are demanded.

3 Problem Formulation

A sequence, S, is a list of objects $\langle o_1, o_2, \cdots o_{|S|}\rangle$, where o_i represents an object o located at position i in S and $|S|$ is the length of S. The distance between two objects o_i and o_j where o_i can be located either before or after o_j, denoted by $\delta_{i,j}$, is equal to $|j-i|$. Two objects o_i and o_j are said to be close if their distance is not greater than a distance bound ω, i.e., $\delta_{i,j} \leq \omega$. Each object is classified to one of n object sets, i.e., $\mathcal{O} = \{O_i | i \in [1, n]\}$ according to application needs. The following is a running example.

Example 1 (Running Example). *Given a sequence $S = \langle a_1, b_2, a_3, a_4, b_5, b_6, a_7, c_8, c_9, d_{10}, d_{11}, c_{12}\rangle$ and four object sets, $\mathcal{O} = \{A, B, C, D\}$ with $A = \{a\}$, $B = \{b\}$, $C = \{c\}$ and $D = \{d\}$. The distance between a_7 and d_{10}, $\delta_{7,10}$, is 3, and that between a_7 and b_8, $\delta_{7,8}$, is 1. When ω is set to 2, a_7 and b_8 are regarded to be close but a_7 and d_{10} are not.* □

Our model considers one object in one sequence position for presentation clarity. It can be easily extended to have multiple objects located at a same position and use real number as positions [7,12]. Correspondingly, our proposed search algorithms are general enough to handle these variations. The correlation coefficient between two object sets is defined in Definition 1. We consider the *cosine* metric because of its wide acceptance. The coefficient $\phi_\omega(X, Y)$ ranges from 0 to 1. The larger the coefficient is, the stronger the correlation of two object sets exploits.

Definition 1 Object Set Correlation Coefficient. *The correlation coefficient between two object sets X and Y is defined in Equation (1).*

$$\phi_\omega(X, Y) = \frac{|XY|_\omega}{\sqrt{|X| \cdot |Y|}} \tag{1}$$

where $|X|$ and $|Y|$ are the numbers of objects in X and in Y, respectively and $|XY|_\omega$ is the number of close object pairs that depends on the setting of ω. For convenience, we omit ω from $\phi_\omega(X, Y)$ and $|XY|_\omega$ if the context is clear. ■

To calculate $\phi(X, Y)$, $|X|$, $|Y|$ and $|XY|$ have to be determined. However, it is not that straightforward to measure $|XY|$ due to a *redundant count problem.* Let us consider the first 5 objects a_1, b_2, a_3, a_4, b_5 in S in the running example. If ω is set to 2, b_2 is close to a_1, a_3 and a_4, and b_5 is close to a_3 and a_4. Based on this, while $|A|$ and $|B|$ are 3 and 2, respectively we would obtain 5 pairs of close objects (i.e., $|AB| = 5$), which is, however, incorrect. In fact, $|XY|$ represents the number of close object pairs that must be disjoint. In other words, once an object in set X is identified to be close to an object in set Y, it contributes only one to $|XY|$, no matter how many objects in set Y it is close to and vice versa. Back to the running example, we can only identify 2 disjoint close object pairs, e.g., $\langle a_1, b_2 \rangle$ and $\langle a_4, b_5 \rangle$ and $|AB|$ equals 2. Based on object set correlation coefficient, correlation query is formally defined in Definition 2 and exemplified in Example 2. Take the redundant count problem into consideration, our proposed algorithms to be discussed next guarantee the correctness of $|XY|$.

Definition 2 Correlation Query. *Given a sequence, a set of predefined object sets, $\mathcal{O}$, and two query parameters: distance bound, ω, and correlation threshold, t, a* correlation query, *$Q(S, \omega, t)$, returns all pairs of object sets $(X, Y) \in \mathcal{O} \times \mathcal{O}$ with $\phi_\omega(X, Y) > t$.* ■

Example 2. *Given a correlation query $(S, 2, 0.5)$ using S and $\mathcal{O}$ specified in Example 1, the correlation coefficients of all object set pairs are derived according to Equation (1) and listed in Figure 2.*

XY	$\lvert X\rvert$	$\lvert Y\rvert$	$\lvert XY\rvert_\omega$	$\phi_\omega(X, Y)$
AB	4	3	3	0.87
AC	4	3	1	0.29
AD	4	2	0	0.00
BC	3	3	1	0.33
BD	3	2	0	0.00
CD	3	2	2	0.82

Fig. 2. Correlation coefficients

Given the four object sets, there are 6 object set pairs. As t is set to 0.5, only AB and CD are qualified and returned as the result set. □

4 Search Algorithms

In this section, we present three algorithms for correlation query, namely, *Multi-Scan Algorithm* (MSA), *One-Scan Algorithm* (OSA) and *Index-Based Algorithm* (IBA). MSA and OSA are scan-based while IBA is an index approach.

4.1 Multi-Scan Algorithm (MSA)

Multi-Scan Algorithm (MSA) is an iterative algorithm. In each turn, it examines one pair of object sets, say X and Y, and determines the corresponding $|X|$, $|Y|$ and $|XY|$ to compute $\phi(X, Y)$. It skips objects not belonging to candidate object sets. Given n sets of data objects, MSA iterates for $\binom{n}{2}$ object set pairs.

To tackle the redundant count problem that affects the correctness of $|XY|$, we allocate a sliding window W to buffer the ω recently examined objects. An object is only compared against those objects inside W to form close object pairs. If an object can be paired with multiple objects in W, the oldest object is matched so the recent ones are reserved to match with those later examined in order to maximize $|XY|$. Once an object is paired with a new object, it is deleted from the sliding window W to prevent double counting. A counter c_{XY} carries the number of close object pairs formed so far with zero as its initial value.

Figure 3(a) depicts the pseudo-code of MSA. It consists of a big loop (line 1-15). For each iteration, it examines one object set pair. It reads one object o from S each time (line 4). It compares o against a buffer W and updates counters (i.e., c_X, c_Y and c_{XY}) and W accordingly (line 6-11). By the end of each turn, it collects the examined object sets if the calculated correlation coefficient is greater than a correction threshold, t (line 14) and returns the result (line 16). Example 3 shows how MSA determines the correlation coefficient.

Example 3. *Suppose object sets A and B are examined and ω set to 2. First, three counters c_A, c_B and c_{AB} that are used to measure $|A|$, $|B|$ and $|AB|$, respectively, are all initialized to 0, and a sliding window, W, that buffers two recently accessed objects, is initialized with $(\perp, \perp)$, (where $\perp$ means no object). The trace of MSA examining A and B is shown in Figure 3(b) where each row presents a state right after an object is examined.*

Algorithm. MSA
input: a sequence S; a set of object sets $\mathcal{O}$, dist. bound ω; corr. threshold t;
output: a result set of object set pairs R;
Begin
1. **foreach** $(X, Y) \in \mathcal{O} \times \mathcal{O} \wedge X \neq Y$ **do**
2. start at the head of S;
   $c_X \leftarrow 0$; $c_Y \leftarrow 0$; $c_{XY} \leftarrow 0$;
3. **repeat**
4. read o from S;
5. **if** $o \in X \vee o \in Y$ **then**
6. increase $c_X (c_Y)$ if $o \in X$ (Y) by 1;
7. compare o against W;
8. **if** o matches with o' **then**
9. increase c_{XY} by 1;
10. replace o' with $\cancel{o'}$ in W; add $\cancel{o}$ to W;
11. **else** add o to W;
12. **else** add $\perp$ to W;
13. **until** S end;
14. **if** $\frac{c_{XY}}{\sqrt{c_X \cdot c_Y}} > t$ **then** $R \leftarrow R \cup \{(X, Y)\}$;
15. **endforeach**
16. **return** R;
End.

(a) The pseudo-code of MSA

object	W	matched	c_A	c_B	c_{AB}
$\langle$init$\rangle$	$(\perp,\perp)$	-	0	0	0
a_1	$(\perp,a_1)$	no	1	0	0
b_2	$(\cancel{a_1},\cancel{b_2})$	$\langle a_1, b_2\rangle$	1	1	1
a_3	$(\cancel{b_2},a_3)$	no	2	1	1
a_4	(a_3,a_4)	no	3	1	1
b_5	$(a_4,\cancel{b_5})$	$\langle a_3, b_5\rangle$	3	2	2
b_6	$(\cancel{b_5},\cancel{b_6})$	$\langle a_4, b_6\rangle$	3	3	3
a_7	$(\cancel{b_6},a_7)$	no	4	3	3
c_8	$(a_7,\perp)$	no	4	3	3
c_9	$(\perp,\perp)$	no	4	3	3
d_{10}	$(\perp,\perp)$	no	4	3	3
d_{11}	$(\perp,\perp)$	no	4	3	3
c_{12}	$(\perp,\perp)$	no	4	3	3

(b) Trace of MSA for A and B

Fig. 3. Multi-Scan Algorithm

The search starts with examining a_1 ($\in A$) from S; c_A and W are updated to 1 and $(\perp, a_1)$, respectively. Next, b_2 is examined and it is close to a_1 in W. Both are marked as $\cancel{a_1}$ and $\cancel{b_2}$ so they are not available for other match and both c_B are c_{AB} are updated to 1. Next, a_3 is accessed and c_A is increased to 2. Since no buffered object available for matching, it is appended to W, while $\cancel{a_1}$ is shifted out. W becomes ($\cancel{b_2}$, a_3). Next, a_4 is scanned and W is replaced with (a_3,a_4) and c_A is increased to 3. Later, b_5 is examined and both a_3 and a_4 are close to it. To maximize c_{AB}, b_5 is matched with a_3, i.e, the older one in W and c_{AB} is updated to 2. This examination continues until S is completely scanned. At last, c_A, c_B and c_{AB} are 4, 3, and 3, respectively and hence the coefficient $\phi(A, B)$ is obtained as $c_{AB}/\sqrt{c_A \times c_B} = 3/\sqrt{4 \times 3} = 0.87$. □

MSA needs only a few counters and a ω-slot buffer. However, it is inefficient because of its blind scan of the sequence multiple times. As seen in Example 3, the last five objects scanned from S do not belong to either A or B and they do not affect $\phi(A, B)$ but MSA has to scan all of them. Similarly, when examining another pair of candidates, C and D, the head portion of the sequence that contains no related objects is also scanned. Finally, each scan incurs $O(\omega \cdot |S|)$ comparisons. Hence, the complexity of MSA is $O(n^2 \cdot \omega \cdot |S|)$.

4.2 One-Scan Algorithm (OSA)

One-Scan Algorithm (OSA) improves MSA by evaluating all object set pairs in one sequence scan. For each object set pair, it counts the numbers of close objects. During the sequence scan, it updates the respective counters. The pseudo-code of OSA is depicted in Figure 4(a). It compares each examined object o against a sliding window W and updates respective counters (line 2-12). After the scan, those with coefficient higher than the correlation threshold t are collected as a part of the query result (line 13-15) and finally the result is returned (line 16).

To address the redundant count problem, we associate objects in W with their matched partners if any. When an object $o \in O$ is examined against objects in W, it tries to match with an object available, i.e., not belonging to O and not being matched with any object belonging to O. In case multiple buffered objects are available to match, the oldest one is chosen. Example 4 illustrates OSA based on our running example.

Example 4. *Due to limited space, our discussion focuses only on object sets A, B and C and their counters c_{AB}, c_{AC} and c_{BC}. Assume that ω is set to 2. Figure 4(b) shows the trace. We use x:$\{y, z\}$ to denote a buffered object x and its paired objects, y and z. OSA first loads a_1 from S and buffers it in W, which becomes $(\perp, a_1{:}\{\})$. Next, b_2 is examined. It matches a_1 and contributes one to c_{AB}. Consequently, W becomes (a_1:$\{b_2\}$, b_2:$\{a_1\}$). Thereafter, a_3 and a_4 are studied and found that b_2 has already been matched with a_1. Now W becomes (a_3:$\{\}$, a_4:$\{\}$). Further, b_5 is matched with a_3 which is the oldest and available and c_{AB} is increased to 2. Next, b_6 is matched with a_4; thus, c_{AB} is updated to 3. For the next object a_7, no match is found. Next, c_8 is retrieved and it is matched with both b_6 and a_7. Consequently, both c_{AC} and c_{BC} are updated to 1.*

Algorithm. OSA
input: a sequence S; a set of object sets $\mathcal{O}$, dist. bound ω; corr. threshold t;
output: a result set of object set pairs R;
Begin
1. start at the head of S;
   $c_X \leftarrow 0$; $c_Y \leftarrow 0$; $c_{XY} \leftarrow 0$;
2. **repeat**
3. read o from S (assuming $o \in X$);
4. increase c_X by 1;
5. compare o with W;
6. **forall** o' in W matched with o
7. increase c_{XY} by 1 where $o' \in Y'$;
8. associate o' with o;
9. associate o with o';
10. **endforall**
11. add o and its associated objects to W;
12. **until** S end;
13. **foreach** $(X, Y) \in \mathcal{O} \times \mathcal{O} \wedge X \neq Y$
14. **if** $\frac{c_{XY}}{\sqrt{c_X \cdot c_Y}} > t$ **then** $R \leftarrow R \cup \{(X, Y)\}$
15. **endforeach**
16. **return** R;

End.

(a) The pseudo-code of OSA

exam	W	c_{AB}	c_{AC}	c_{BC}
$\langle$init$\rangle$	$(\perp, \perp)$	0	0	0
a_1	$(\perp, a_1{:}\{\})$	0	0	0
b_2	$(a_1{:}\{b_2\}, b_2{:}\{a_1\})$	1	0	0
a_3	$(b_2{:}\{a_1\}, a_3{:}\{\})$	1	0	0
a_4	$(a_3{:}\{\}, a_4{:}\{\})$	1	0	0
b_5	$(a_4{:}\{\}, b_5{:}\{a_3\})$	2	0	0
b_6	$(b_5{:}\{a_3\}, b_6{:}\{a_4\})$	3	0	0
a_7	$(b_6{:}\{a_4\}, a_7{:}\{\})$	3	0	0
c_8	$(a_7{:}\{c_8\}, c_8{:}\{a_7, b_6\})$	3	1	1
c_9	$(c_8{:}\{a_5, b_6\}, c_9{:}\{\})$	3	1	1
d_{10}	$(c_9{:}\{\}, d_{10})$	3	1	1
d_{11}	(d_{10}, d_{11})	3	1	1
c_{12}	$(d_{11}, c_{12} : \{\})$	3	1	1

(b) Trace of OSA

Fig. 4. One-Scan Algorithm

The next object is c_9 which does not find any close object and hence is simply inserted into W. The run continues until S is fully scanned. Finally, c_{AB}, c_{AC} and c_{BC} are 3, 1, *and* 1, *respectively, based on which, the correlation coefficients of the object set pairs are calculated.* □

For each object $o \in O$ retrieved from a sequence, OSA examines it against all the objects in the sliding window W. Suppose there are n object sets, an object in W can be associated with at most $n-1$ objects. The complexity of examining an object is $O(\omega)$ and that of OSA is $O(\omega \cdot |S|)$ which is n^2 times faster than MSA. However, OSA needs maintain $O(n^2)$ counters and a window with $O(n \cdot \omega)$ slots which incurs a higher space requirement.

4.3 Index-Based Algorithm (IBA)

Since correlation query retrieves object set pairs whose correlation coefficients are higher than a given threshold based on Definition 2, evaluating all the object set pairs is unneeded especially when most of them do not provide higher coefficients. Motivated by this observation, we propose Index-Based Algorithm (IBA). IBA preserves multiple indices, each of which corresponds to one object set. Each index maintains the positions of objects (in the sequence) belonging to the corresponding object set in an ascending order. For instance, for object set A in our running example, the index maintains $\langle 1, 3, 4, 7\rangle$, i.e., a shorter sequence. The index can be prepared off line and its small construction cost that involves only one sequence scan can be amortized by multiple correlation queries with different ω's. Also, statistics collected during index construction is useful to speed up the search.

Given two indices, the correlation coefficient of two corresponding object sets X and Y can be determined by a merge-like matching function. Initially, two pointers p_X and p_Y point to the head of both indices. Follow steps in a *comparison*, *match*, and *slide* strategy. In the comparison step, two positions pointed by p_X and p_Y are compared and the smaller one in the sequence is taken to compare against the buffer W, which keeps ω recently examined positions and corresponding object sets that contribute these position entries. If a match is found, the counter c_{XY} is increased by one, and both matched positions become unavailable for later match. Otherwise, the position is inserted into the buffer. Finally, the pointer located at the examined position slides to the next one and the same steps repeat. If one of indices reaches its end, another index is iteratively fetched. It continues until both indices are completely scanned. We use Example 5 to illustrate this matching.

Example 5. *The trace of IBA matching function (for object sets A and C, based on our running example) is depicted in Figure 5. An object with underline represents the one having smaller position, i.e., the examined object. In the indices, the positions of objects are stored. For illustration, we show the objects.*

A	C	W	c_{AC}
$\langle init \rangle$	$\langle init \rangle$	$(\perp,\perp)$	0
$\underline{a_1}$	c_8	$(\perp,a_1)$	0
$\underline{a_3}$	c_8	$(\perp,a_3)$	0
$\underline{a_4}$	c_8	(a_3,a_4)	0
$\underline{a_7}$	c_8	$(\perp,a_7)$	0
–	$\underline{c_8}$	$(\cancel{a_7},\cancel{c_8})$	1
–	$\underline{c_9}$	$(\cancel{c_8},c_9)$	1
–	$\underline{c_{12}}$	(c_9,c_{12})	1

Fig. 5. Trace of IBA for object sets A and C

First, all the four objects from A, i.e, a_1, a_3, a_4 and a_7, are retrieved as all of them are smaller than c_8, the head object of set C. Then, the index for A reaches its end and c_8, the head object of C is retrieved. It matches a_7 in W and c_{AC} is increased to 1. Thereafter, objects c_9 and c_{12} are examined and the end of set C is reached, indicating the completion of this matching function. Since c_{AC} (i.e., $|AC|$) equals 1 and $|A|$ and $|C|$ are 4 and 3, respectively, the correlation coefficient of sets A and C $\phi(A,C) = 1/\sqrt{4 \cdot 3} = 0.29$. □

This matching function outperforms MSA because it only scans objects belonging to the targeted object sets but not the entire sequence as MSA does. It reduces the number of scanned objects from $O(|S|)$ to $O(|X| + |Y|)$, with X and Y indicating the examined object sets. However, it may still suffer from multiple scans of indices. Actually, the performance of IBA can be significantly improved when several optimization techniques are applied. In what follows, we first discuss three optimization techniques, namely, *candidate screening*, *group matching* and *early termination* and then explain how to integrate them into IBA to further boost up the search performance.

Candidate Screening. Candidate screening attempts to filter out object set pairs with their correlation coefficient definitely lower than a given correlation threshold, so the examination of those can be saved. Based on the cardinality and distribution of each object set, two coefficient values can be estimated respectively. In the following, we detail the two correlation coefficient estimations.

- **Estimation based on cardinalities.** As $|X|$ and $|Y|$ are the cardinalities of X and Y, respectively and they can be accounted during index building, the upper bound of the correlation coefficient between X and Y is $\frac{\min(|X|,|Y|)}{\sqrt{|X|\cdot|Y|}}$. For instance, the maximum correlation coefficient between A and D in our example is $\frac{\min(4,2)}{\sqrt{4\cdot 2}} = 0.45$.
- **Estimation based on distributions.** The cardinality-based estimation is straightforward, but it is nothing related to ω. In fact, the number of close objects is highly dependent on ω and the distance between close objects. During the index construction for each object set, we account 1) the smallest and the largest positions of objects inside the object set to get the distance range; and 2) the distance between any two adjacent objects. For any two object sets, if their distance ranges are more than ω apart, they are guaranteed not correlated. Thus, the estimated coefficient should be zero. For instance, the ranges of A and D in our running example are $(1, 7)$ and $(10, 12)$. Consequently, the ranges of A and D are disjoint and their estimated coefficient is, of course, zero.

  If two object sets have their ranges overlap, their coefficient can be estimated based on the probability of finding close object pairs, as detailed in the following. Assuming distances between adjacent objects in an object set X follows normal distribution, we collect the mean (μ_X) and standard deviation (σ_X) of all the distances between adjacent objects during index construction. Other possible distributions will be studied in our future work. Consider A from our example. After building the index, $|A|$, μ_A and σ_A are collected as 4, 1.67 (i.e.,$\frac{2+1+2}{3}$) and 0.58, respectively.

  We estimate the probability that the distance between objects of two object sets is not greater than ω, denoted by p. So, p is the probability that objects are close enough to match. Let $\delta_{X,Y}$ be the expected distance between objects in X and Y, and p can be estimated by $P(|\delta_{X,Y}| \leq \omega) = P(-\omega \leq \delta_{X,Y} \leq \omega)$, i.e., the probability that $\delta_{X,Y}$ lies within the range $[-\omega, \omega]$. To obtain p, we first obtain the standard normal variable Z based on Central Limit Theorem [11], i.e.,

$$Z = \frac{(\mu_X - \mu_Y) - \delta_{X,Y}}{\sqrt{\sigma_X^2/|X| + \sigma_Y^2/|Y|}}$$

where the value of Z follows normal distribution. We estimate p as $P(z_{lower} \leq Z \leq z_{upper})$ (i.e., $P(-\infty \leq Z \leq z_{upper}) - P(-\infty \leq Z \leq z_{lower})$), in which z_{lower} and z_{upper} are the lower and upper limits, respectively. To resolve this probability, z_{lower} and z_{upper} are computed as $z_{lower} = \frac{(\mu_X - \mu_Y) - \omega}{\sqrt{\sigma_X^2/|X| + \sigma_Y^2/|Y|}}$, and $z_{upper} = \frac{(\mu_X - \mu_Y) + \omega}{\sqrt{\sigma_X^2/|X| + \sigma_Y^2/|Y|}}$. Finally, the estimated maximum correlation coefficient is determined as $p \cdot \frac{\mathsf{min}(|X|,|Y|)}{\sqrt{|X| \cdot |Y|}}$.

Our approach first conducts cardinality-based estimation that is lightweight and discard those object set pairs with their estimations smaller than the given threshold. For those object set pairs passing the first estimation, distribution-based estimation is conducted and compared. Finally, the indices of those object set pairs passing both tests are examined with matching functions.

Group Matching. Instead of pairwise matching, matching among a group of object sets is preferred, thus avoiding the multiple index accesses if an object set is founded to be correlated to more than one object set simultaneously. The idea of group matching is pretty similar to OSA by maintaining several counters. The only difference is that multiple indices, rather than a single sequence, are traversed at the same time.

Early Termination. Early termination determines approximate the correlation coefficient of object set pairs without completely traversing the indices, thereby improving the response of the search. We maintain c_X, c_Y and c_{XY} to keep track of the numbers of examined objects in X, Y, and matched objects, respectively. In addition, we keep ω_X and ω_Y to bookkeep the number of buffered objects of X and Y that are still available (i.e., not yet matched). During matching, we estimate both the maximal correlation coefficient $max\phi(X,Y)$ and the minimal correlation coefficient $min\phi(X,Y)$.

The maximal coefficient $max\phi(X,Y)$ can be obtained if all remaining unexamined objects can be matched and calculated as $\frac{c_{XY} + \mathsf{min}(|X| - c_X + \omega_X, |Y| - c_Y + \omega_Y)}{\sqrt{|X||Y|}}$ at any point of time. Consider Example 5. Behind object c_8, there is no more object from A and two objects from C pending for the examination, with an empty buffer. Since the current c_{AC} is one, we can approximate the maximal correlation coefficient $max\phi(A,C)$ is $\frac{1+\mathsf{min}(4-4+0,3-1+0)}{\sqrt{4 \cdot 3}} = \frac{1}{\sqrt{4 \cdot 3}} = 0.29$. Since the maximum value of the coefficient is below the given threshold ($t = 0.5$), it is safe to skip the remaining objects (i.e., c_9 and c_{12}) from examination and assures that object set A and C are not correlated.

The minimal correlation coefficient, $min\phi(X,Y)$ can be determined if all the remaining unexamined objects do not match. It is expressed as $\frac{c_{XY}}{\sqrt{|X||Y|}}$. Once an object set pair with minimal coefficient larger than the given threshold, it is guaranteed to be one of the answer sets. Back to Example 5 and suppose $t = 0.2$. After the examination of object c_8, c_{AC} is one and there might not be any close object pair. Therefore, the minimal value of coefficient can be derived according

Algorithm IBA
input: a sequence S; a set of object sets $\mathcal{O}$,
distance bound ω; correlation threshold t;
output: a result set of object set pairs R;
Begin
1. **foreach** $(X, Y) \in \mathcal{O} \times \mathcal{O}$ **do**
2. **if** (X,Y) pass *candidate screen* **then**
3. start from heads of I_X and I_Y;
4. **repeat**
5. read o with the smallest position from I_X and I_Y;
6. increase c_X if $o \in X$ (or c_Y if $o \in Y$) by 1;
7. compare o with W;
8. **if** match **then** increase c_{XY} by 1.
9. add o to W;
10. compute $max\phi$ and $min\phi$;
11. **if** $max\phi \le t$ **then goto** 14;
12. **if** $min\phi > t$ **then** $R \leftarrow R \cup \{(X, Y)\}$; **goto** 14;
13. **until** I_X and I_Y end;
14. **if** $\frac{c_{XY}}{\sqrt{c_X \cdot c_Y}} > t$ **then** $R \leftarrow R \cup \{(X, Y)\}$;
15. **endforeach**
16. **return** R;

End

Fig. 6. The pseudo-code of IBA

to $\frac{c_{AC}}{\sqrt{|A| \cdot |C|}}$, i.e., $min\phi(A, C) = \frac{1}{\sqrt{4 \cdot 3}} = 0.29$. Thus, it can be safely included as an answer set.

Putting all the techniques together, Figure 6 lists the pseudo-code of IBA. IBA first prepares a pool of candidate object set pairs. Then, it studies all the individuals with candidate screening and discards those uncorrelated based on the two estimated coefficients (line 2). The remainders are then examined through group matching. Here, the figure shows the matching function (line 5-9) for sake of simplicity and I_X and I_Y are the indices of X and Y, respectively. During the match, we validate if early termination applies to stop the matching without examining the rest of the indices (line 10-12). Finally IBA outputs the result object set pairs if their correlation coefficients (line 14) (or their minimal correlation coefficients obtained while the match is early terminated (line 12)) are greater than the correlation threshold of the query.

Let $1/f$ be a fraction of candidates passing the candidate screening. IBA examines n^2/f candidates with $f \in [1, n^2]$. As each matching function incurs $O(\omega \cdot |S|/n)$ comparisons, the complexity of IBA is $O(n \cdot \omega \cdot |S|/f)$. The performance of IBA depends on f that is affected by distance bound and correlation threshold. So, for a small ω or a large correlation coefficient, f will become large. When $f > n$, IBA will achieve better performance than OSA. To construct the index, a sequence needs to be scanned once and the cost of $O(|S|)$ is amortized by correlation queries.

5 Variants of Correlation Query

In this section, we discuss several variants of our correlation query, namely, constrained correlation query, position correlation query and correlation spectrum query, and discuss the extensions of our algorithms to support them.

Constrained Correlation Query. In our model, if multiple objects are available for matching, the farthest one within a window is picked to maximize the counts and thus the correlation coefficient. However, the matching in some cases is not arbitrary. For instance, in document analysis, a word is usually semantically related with closest one; in event causality analysis, one cause event must occur right before its consequence. Therefore, the presence order have to be considered in identifying close object pairs. Constrained correlation query takes additional matching constraints into consideration. Our proposed algorithms can be easily adjusted by incorporating matching rules, like matching the closest one. When an examined object from a sequence is compared with buffered objects, the matching rules are applied to find a right candidate to match.

Position Correlation Query. For some applications, it is also interesting to know the correlation of objects with respect to their positions in a sequence. For example, a company may be interested to explore the correlation of their products sold to certain days and event analysts want to identify what events are likely to happen at certain times. Specific to temporal data, this is also referred to as temporal autocorrelation. Putting the search into a generalized framework, position correlation query explores the correlation of objects to their positions in a sequence. This query can be extended to determine object periodicity in a sequence by specifying regular interval. To support this variant, our algorithms can be extended by buffering specific sequence positions rather than examined objects. The other parts of our algorithms remain the same to count the number of close objects and to determine correlation coefficients.

Correlation Spectrum Query. Correlation coefficients increase together with the number of close objects which is in turn controlled by ω. In some applications, we might suspect that two object sets are correlated but are not so certain about the setting of a distance bound which can produce a high correlation coefficient. A straightforward approach is to obtain the coefficient for each possible ω, which varies from 1 up to the length of the entire sequence. Correlation spectrum query returns the coefficients between two object sets according to a range of ω but not a single one. The proposed algorithms can be extended by keeping a large number of counters and a very large buffer. However, it may not be space and time efficient. We shall study this in our future direction.

6 Performance Evaluation

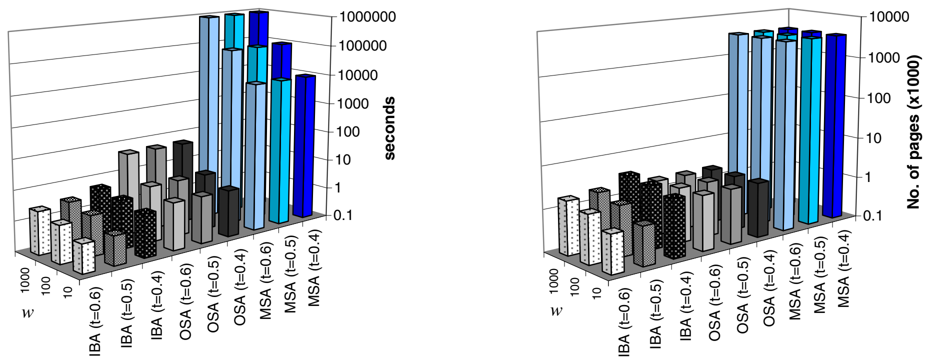
This section evaluates the performance of our three proposed algorithms, namely, Multi-Scan Algorithm (MSA), One-Scan Algorithm (OSA) and Index-Based Algorithm (IBA) for correlation query. We implemented them in GNU C++ and conducted experiments on Linux computers with Intel CPU 3.2GHz. We evaluate our algorithms based on synthetic and realistic data sequences with each sequence stored in one file. Synthetic data sequences are characterized by the sequence length (i.e., $|S|$), the number of object sets (i.e., n) and the variations of object set cardinalities (controlled by a factor s). The sequence length varies from 1M (2^{20} objects) to 5M with 2M as the default unless specified otherwise. The number of object sets (n) is ranged from 20 to 100 in step of 20 with 60 as the default. The cardinalities of object set are controlled by a skewness factor s. In generating synthetic sequence, the probability of objects in a sequence mapped to object sets follows Zipf distribution with s controlling the skewness of the distribution. The value of s varies from 1.5 to 3 in step of 0.5. This affects the cardinalities of object sets and the distributions of objects of an object set in a sequence. As a large s is set, both object set cardinalities and distributions vary a lot and only a few object sets would produce higher correlation coefficients.

We also use two realistic data sequences, i.e., EARTHQUAKE [2] and APRS [1]. EARTHQUAKE is an earthquake log. It remarks times, geographical coordinates and earthquake magnitudes. This log contains 446k records ordered according to time. We classify each entry based on coordinates into 100 equal-sized rectangular geographical regions. For EARTHQUAKE, $|S| = 446k$ and $n = 100$. Correlation query is evaluated on this earthquake log to search which pairs of geographical regions usually experienced earthquake at the same time (according to the setting of ω). APRS is a message log about radio base station broadcasting messages in United States. It includes times and names of base stations that broadcast. The log consists of 188k records related to 1000 base stations collected on Aug 23 2001, and it is ordered based on time. For APRS, $|S| = 188k$ and $n = 1000$. In this log, it only records base stations who broadcast messages but no information about their correspondents. Correlation query is used to find pairs of communicating base stations based on an observation that two communicating base stations would have multiple message exchanges within small time intervals, determined by ω.

Correlation query is evaluated based on two parameters, namely, distance bound (ω) and correlation threshold (t). The settings of ω is varied from 10, 100, to 1000 and t is varied among 0.4, 0.5, and 0.6. Two performance metrics are measured, namely, *elapsed time* and *I/O cost*. The elapsed time is the duration of time, in terms of seconds, from the time when an algorithm starts to the time when all the results are returned. The I/O cost measures the number of pages accessed from an underlying file storing the sequence. The page size is 4KB. The results to be present are the averages of 100 runs for each experiment setting.

6.1 Evaluation on Synthetic Data

The first set of experiments is based on synthetic data sequence. We evaluate all the factors, namely, ω, t, n, $|S|$ and s. We first evaluate the impact of ω on the search performance. The larger the ω is, the more the objects are considered to be close and hence the larger the resulted correlation coefficients are. Figure 7(a) and Figure 7(b) depict the results in terms of elapsed time and number of pages



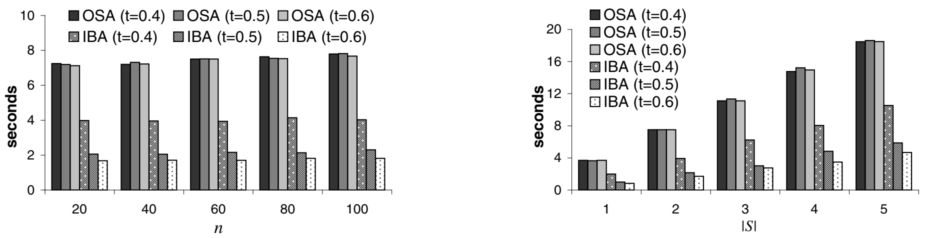
(a) Elapsed time (b) I/O cost

Fig. 7. Impact of ω

accessed for various ω while $|S|$, n and s are fixed at $2M$, 60 and 2.0, respectively. From Figure 7(a), it can be observed that an increase of ω results in longer elapsed time. For both MSA and OSA, the size of the buffer is increased as ω grows thus increasing the lookup cost. Among all, MSA incurs the longest elapsed time, several orders of magnitude longer than OSA and IBA for same settings because of its multiple scans. On the other hand, IBA performs the best and at least 10 times faster than OSA. From the figure, we can see OSA and MSA are invariant to the correlation threshold setting (t from 0.4 to 0.6) but IBA performs better when a larger t is set. This is because the proposed optimization techniques become more effective when t is larger.

In Figure 7(b), observations similar to Figure 7(a) are made that MSA is the worst among all candidates. Both OSA and MSA incur constant I/O costs, due to a fixed number of scans. The performance of IBA varies depending on the number of object set pairs being investigated. When t is smaller (e.g., $t = 0.4$) or ω is larger (e.g., $\omega = 1000$), IBA becomes less competitive than OSA in terms of number of page accesses. This is because the optimization techniques proposed to speed up the performance of IBA do not take effect for a longer distance bound or a larger correlation threshold, without mentioning that IBA still suffers from multiple scans compared with OSA. However, the measurement of counts for correlation coefficient is CPU intensive. IBA, although accessing a little more pages, incurs less overheads in matching objects to measure the coefficient and hence its cost is payed off. As previously shown, IBA takes shorter elapsed time. Since MSA is identified as the weakest candidate, we omit it from the following discussion. Besides, we focus our remaining evaluation on the elapsed time.

Then, we evaluate the factor of n, the number of object sets. The immediate effect of n is on the size of a candidate pool and the number of candidates in matching for IBA. Figure 8(a) plots the results in terms of elapsed time against n. The other factors such as $|S|$, s and w are fixed at 2M, 2.0 and 100, respectively. For IBA, the index construction time is 6.2 seconds for all n evaluated and the



(a) Impact of n (b) Impact of $|S|$

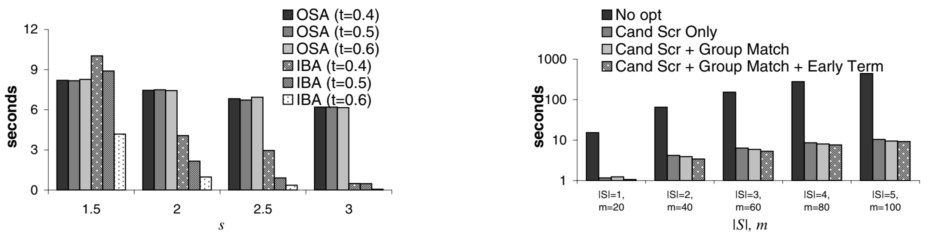
Fig. 8. Impacts of n and $|S|$

indices are used for queries with various t. The performance of OSA is consistent to our analysis that it is invariant to n. On the other hand, IBA is more or less stable to n. Although the number of object set pairs grows as n does, the average sizes of indices are reduced due to fixed $|S|$. Further, most of the pairs are filtered out when the threshold t is set to be high. As a result, we can observe a significant difference between IBA and OSA especially when t is set to 0.6.

Next, we evaluate the impact of $|S|$, the length of sequence. Figure 8(b) shows the results in terms of elapsed time versus the length of sequence, $|S|$. The other factors such as n, s, and ω are set to 60, 2 and 100, respectively. Obviously, a longer sequence results in a longer elapsed time. OSA is invariant to t as explained before and its running time is linear proportional to $|S|$. IBA again runs much faster than OSA for all $|S|$ evaluated. From this, we can conclude that for a long sequence, IBA is superior to OSA, particularly when a larger t is specified. For $|S| = 1M, 2M, 3M, 4M$ and $5M$, the index construction times for IBA are 3.1, 6.2, 9.4, 12.4, 15.1 seconds, respectively.

Further, we examine the impact of s, the skewness parameter for object set cardinality variation. If the object cardinalities are very different, the correlation coefficients of object set pairs would not be high due to a number of unmatched objects. In this evaluation, we vary s among 1.5, 2, 2.5 and 3. When s is set to 3, the produced sequence has the most significant variation in the cardinalities of object sets, i.e., the most skewed sequence with regard to cardinalities.

The results are displayed in Figure 9(a). Here, the performance of OSA is improved together with the increase of s. This is because when s is large, certain object sets dominate the entire sequence and thus the buffer. As a results, the majority of the objects in the sequence belong to a small number of object sets, and the comparison between objects from the same set, which is expected to occur very frequently, can be saved. On the other hand, IBA performs well when s is set to 2 or above. For these settings, the object set cardinalities are skewed and most of the object set pairs that are identified not correlated will be eliminated at the beginning. However, when s is at 1.5, object sets are in similar sizes and hence the estimation based on cardinalities and distribution is not effective in candidate screening. Many object set pairs have to be examined



(a) Impact of s

(b) Evaluation of IBA ($\omega = 100$)

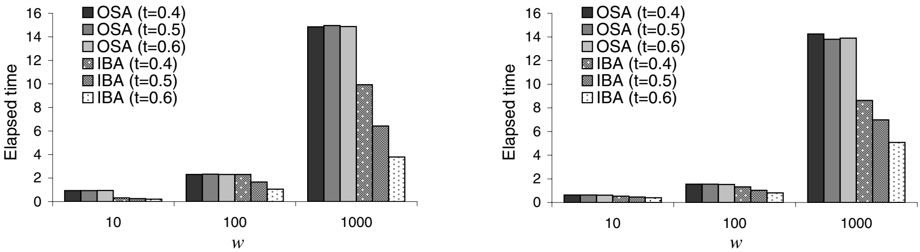
Fig. 9. Elapsed time for various s and Evaluation of IBA optimization

in detail, causing a longer elapsed time. However, this cardinality variation can be detected during the IBA index construction. If s is small, OSA is preferred. Otherwise, IBA is more efficient especially when a larger threshold (t) is used.

Evaluated upon all the factors in synthetic data sequences, IBA is shown to perform the best. Now we investigate the effectiveness of proposed techniques to improve IBA. Recall that the three proposed techniques are candidate screening (labeled as Cand Scr), group matching (Group Matching) and Early Termination (Early Term). Instead of trying every possible combination of proposed techniques, we incrementally enable those techniques against IBA with no technique applied (No opt) and evaluate the performance in terms of elapsed time. In this experiment, we fix ω at 100 and t at 0.5. The results are shown in Figure 9(b), from which we can observe that candidate screening is the most effective approach that reduces the elapsed time by screening out irrelevant candidates. Group Matching and Early Termination can further slightly reduce the elapsed time.

6.2 Evaluation of Real Data

In this subsection, we evaluate the performance of OSA and IBA on real datasets. We vary both ω and t in our evaluation. This experiment tests the practicality of our algorithms in real situations. The results in terms of elapsed time for EARTHQUAKE and APRS sequences are shown in Figure 10(a) and 10(b), respectively. For EARTHQUAKE, ω is expressed as days, we evaluate 10 days, 100 days and 1000 days. For APRS, ω is expressed as 10 sec, 100 sec and 1000 sec. The results are consistent with those obtained from synthetic data. When ω is set to a small value (say, 10), both IBA and OSA can quickly determine the results since most of objects are not close and the buffer size is small. While ω is increased, IBA can save more elapsed time than OSA. As we explained above, this improvement is contributed by candidate screening technique which approximates the potential correlation coefficient to filter those unqualified candidates out of the detailed examination. From the result, IBA can be concluded as the best efficient search for correlation query.



(a) EARTHQUAKE (b) APRS

Fig. 10. Evaluation on real datasets

7 Conclusion

Sequence is widely used by various applications. In a sequence, objects that are often closely located are likely to be correlated to each other. In this paper, we identify a new query, namely *correlation query*, to search for object set pairs based on two parameters: 1) *distance bound* (ω) and 2) *correlation threshold* (t). The distance bound determines whether two objects are close in a sequence. Based on the number of close objects, we measure the strength of object correlation by cosine metric as the correlation coefficient. The larger the coefficient is, the stronger the correlation between corresponding object set pairs is interpreted. A correlation query then returns those object set pairs having corresponding correlation coefficient higher than the given correlation threshold. Three search algorithms, namely, Multi-Scan Algorithm (MSA), One-Scan Algorithm (OSA) and Index-Based Algorithm (IBA), are proposed in this paper to efficiently process correlation query. We conducted an extensive set of experiments to evaluate the performance of different algorithms. IBA, together with three optimization techniques, outperforming the other two for both real and synthetic sequences, is the most efficient algorithm to this correlation query.

Acknowledgement

This study was supported in part and monitored by the Advanced Research and Development Activity (ARDA) and the Department of Defense. The views and conclusions contained in this document are those of the author(s) and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of the National Geospatial-Intelligence Agency or the U.S. Government. The work by Ken Lee and Wang-Chien Lee is also supported in part by the National Science Foundation under Grant no. IIS-0328881, IIS-0534343 and CNS-0626709.

References

1. APRS: Automatic Position Reporting System. [web], `http://aprs.net/`
2. U.S. Geological Survey Earthquake Hizards Program. [web], `http://earthquake.usgs.gov/region/neic/`
3. Agrawal, R., Imielinski, T., Swami, A.N.: Mining Association Rules between Sets of Items in Large Databases. In: Proceedings of the 1993 ACM SIGMOD International Conference on Management of Data, Washington, D.C, May 26-28, pp. 207–216 (1993)
4. Agrawal, R., Srikant, R.: Mining Sequential Patterns. In: Proceedings of the 11th International Conference on Data Engineering (ICDE), Taipei, Taiwan, March 6-10, pp. 3–14 (1995)
5. Han, J., Kamber, M.: Data Mining - Concepts and Techniques. Elsevier, Amsterdam (2006)
6. Li, Y., Chung, S.M.: Text Document Clustering Based on Frequent Word Sequences. In: Proceedings of the 2005 ACM International Conference on Information and Knowledge Management (CIKM), Bremen, Germany, October 31-November 5, pp. 293–294 (2005)
7. Mamoulis, N., Yiu, M.L.: Non-contiguous Sequence Pattern Queries. In: Bertino, E., Christodoulakis, S., Plexousakis, D., Christophides, V., Koubarakis, M., Böhm, K., Ferrari, E. (eds.) EDBT 2004. LNCS, vol. 2992, pp. 783–800. Springer, Heidelberg (2004)
8. Mannila, H., Toivonen, H., Verkamo, A.I.: Discovery of Frequent Episodes in Event Sequences. Data Mining and Knowledge Discovery 1(3), 259–289 (1997)
9. Papadimitriou, S., Sun, J., Yu, P.S.: Local Correlation Tracking in Time Series. In: Proceedings of the 6th IEEE International Conference on Data Mining (ICDM), Hong Kong, China, December 18-22, 2006, pp. 456–465 (2006)
10. Tan, P.-N., Kumar, V., Srivastava, J.: Selecting the Right Interestingness Measure for Association Patterns. In: Proceedings of the 2002 ACM SIGKDD International Conference on Knowledge Discovery, Alberta, Canada, July 23-26, 1994, pp. 32–41 (1994)
11. Walpole, R.E., Raymond H, M., Myers, S.L.: Probability and Statistics for Engineers and Scientists. Prentice Hall, Englewood Cliffs (1997)
12. Wang, H., Perng, C.-S., Fan, W., Park, S., Yu, P.S.: Indexing Weighted-Sequences in Large Databases. In: Proceedings of the 19th International Conference on Data Engineering (ICDE), Bangalore, India, March 5-8, 2003, pp. 63–74 (2003)
13. Xiong, H., Shekhar, S., Tan, P.-N., Kumar, V.: TAPER: A Two-Step Approach for All-Strong-Pairs Correlation Query in Large Databases. IEEE Transactions on Knowledge and Data Engineering (TKDE) 18(4), 493–508 (2006)
14. Zhang, P., Huang, Y., Shekhar, S., Kumar, V.: Correlation Analysis of Spatial Time Series Datasets: A Filter-and-Refine Approach. In: Whang, K.-Y., Jeon, J., Shim, K., Srivastava, J. (eds.) PAKDD 2003. LNCS (LNAI), vol. 2637, pp. 532–544. Springer, Heidelberg (2003)