BULGARIAN ACADEMY OF SCIENCES

CYBERNETICS AND INFORMATION TECHNOLOGIES • Volume **15**, No 6
Special Issue on Logistics, Informatics and Service Science

Sofia • 2015 Print ISSN: 1311-9702; Online ISSN: 1314-4081
DOI: 10.1515/cait-2015-0081

Mining Similar Traces of Entities on Web

Xinyan Huang[1,3], *Xinjun Wang*[1,2], *Hui Li*[1,2]

[1]*Shandong University, Num 1500, SunHua Road in High Tech Industrial Development Zone, Ji'nan, China*
[2]*Dareway Software Co., Ltd, Num 1500, SunHua Road in High Tech Industrial Development Zone, Ji'nan, China*
[3] *Shandong University of Finance and Economics, Num 7366, East Erhuan Road in Lixia Zone, Ji'nan, China*
Emails: 20063462@sdufe.edu.cn wxj@sdu.edu.cn lih@sdu.edu.cn

Abstract: *Events about entities have been widely collected on Web, allowing us to analyze how peer entities interact and learn the relationships that exist among the entities. In this paper we investigate similar traces that have not been adequately studied so far. Intuitively, peer entities tend to have similar traces. The challenges in mining similar traces are:* (*1*) *the occurring time lags of traces are usually unknown and varying;* (*2*) *the existence of large-scale events of entities and complexity of the model representing all the events. In this paper we propose a simple, but practical method that addresses all these challenges. Firstly, sliding windows are adopted to filter out the significant events and then find the candidate topic sequences. Secondly, dynamic programming is employed to mine similar candidate topic sequences of entities. Finally, an efficient method is proposed to mine all the similar traces of entities. It is able to mine similar traces of peer entities with high accuracy. We conduct comprehensive experiments on synthetic datasets to demonstrate the efficiency of the method proposed.*

Keywords: *Significant event, similar trace, candidate topic.*

1. Introduction

An event is something that happens at a specific time. Nowadays, an increasing number of events are reported on web every day. However, these events are usually scattered and redundant, so that we cannot receive any relevant information.

Among all the problems considered by the people, the behaviour trajectories of the entities (an enterprise or a person), are given more attention. In this paper the behaviour trajectory of an entity is named a trace, which can be loosely defined as a topic sequence. Intuitively, peer entities tend to have similar traces. Similar traces can provide an insightful and concise explanation over the business strategies of peer entities and implicit relationships among them. At the same time, similar traces provide good references to other peer entities.

Unfortunately, mining similar traces from long and noisy history data is not an easy task, due to the following challenges:

Challenge 1. Considering so big scale of events and relatively complex relationships between them, it is difficult and complicated to design the right model representing the events. So a model in place, that is, a big graph is employed, which is named an event relationship graph.

Challenge 2. A similar trace is represented as a sub-graph (topic) sequence. The issue of mining similar traces becomes more difficult, because the general methods of sub-graph mining are not fit for our problem.

Challenge 3. Time gaps of similar traces are usually unknown and varying. For example, trace A is similar to trace B, just as Fig. 1 shows, topic z'_1 is two weeks after topic z_1; however, topic z'_2 is only one week after topic z_2. Behaviours of entities are usually affected by competitive environment or other factors.

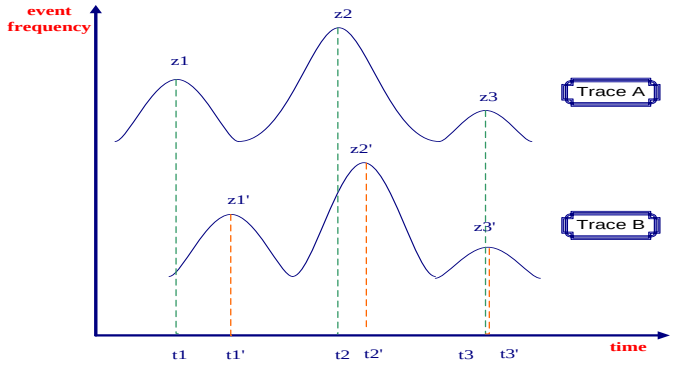

Fig. 1. Two similar traces

In this paper the problem of mining similar traces is solved in the following steps:

Firstly, sliding windows are adopted to filter out the significant events and find the candidate topic sequences. Secondly, dynamic programming is applied to mine similar candidate topic sequences. Finally, an efficient method is proposed to mine all the similar traces of entities.

In our paper we address a significant problem of mining similar traces. In the algorithm proposed – Similar Traces, high efficiency is achieved through filtering out the significant events and candidate topics. At the same time, by the second step, lots of redundant patterns' production can be avoided, which can help in reducing the heavy cost of computation.

The rest of the paper is organized as follows. In Section 2 some related works are discussed. Section 3 gives the problem formulation. Sections 4, 5, 6 and 7 provide an outline of our algorithm to mine similar traces. We report our experimental results in Section 8, and conclude the present study in Section 9.

2. Related works

At present some studies [3, 11] about how to organize the events and events relationships have been investigated. However, less studies aim to mine some meaningful information from the events data of entities [13].

Yang, Shi and Wei [4] introduce the concept of an event evolution graph, in which the events are organized according to the time order, which can help to efficiently browse the whole event evolution process.

Ji, Grishman and Chen [5] propose to identify the "centroid entities" which are frequently involved in events and then link the events, involving the same centroid entity along a time line. However, no more studies are done based on this work.

Li, Wu and Crofoot [10] investigate the relationship among moving objects and proposes a simple, but practical method to mine the relationship from movement data of the objects.

Zhijun Yin et al. [1] study the problem of latent periodic topic analysis from time stamped documents and proposes a model, called LPTA (Latent Periodic Topic Analysis) that exploits the periodicity of terms, as well as term co-occurrences.

Giannotti et al. [7] propose to find the trajectory patterns from the location traces of the moving objects and study their movement behaviors.

However, given the overwhelming scale of events, limited efforts have been focused on the problem in these studies.

Liu et al. [2] propose a "temporal skeletonization" approach to actively reduce the representation of sequences to uncover significant, hidden temporal structures. However, their temporal structures are relatively coarse grained, while our similar traces are sub-graph sequences which can accurately reflect the entities behaviors.

Because the scale of events is very big and similar traces are sub-graph sequences, methods of pure sub-graph mining [12, 14] will lead to poor efficiency and they are not fit for this issue.

So we propose an efficient method, fit for the problem. Firstly, sliding windows are adopted to filter out the significant events and find the candidate topic sequences. Secondly, dynamic programming is adopted to mine similar candidate topic sequences. Finally, an efficient method is proposed to mine all similar traces of entities.

In our algorithm, high efficiency is achieved through filtering out the significant events and candidate topics. At the same time, through the step of mining similar candidate topic sequences, the production of lots of redundant patterns can be avoided.

3. Problem formulation

The work presented is based on the assumption that all the event relationship graphs of entities have been established already. Our goal is to mine all similar traces from all event relationship graphs.

We formally define the notion related with mining of similar traces as follows:

Definition 3.1. Event. An event is something that happens at some specific time, and often at some specific place [11]. In our paper an event of an entity is defined as a model E with six attributes, represented as follows:

E ⟨subject, activity, {object}, time, {location}, frequency⟩.

Among the six attributes, subject, activity and time elements are required. The frequency points to the occurrence frequency of the event, which comes from original pages.

Definition 3.2. Event relationship. An event relationship points to the dependent relationship from one event to another. Only three kinds of relationships are considered in our former work [11], which are causal relationship, part-of relationship and following relationship.

Definition 3.3. Event relationship graph. An event relationship graph $G = (V, E)$ is to link all the events V of an entity according to the relationships E between them. A fragment of an event relationship graph is shown in Fig. 2, in which R_c represents a causal relationship, R_f represents a following relationship and R_p represents a part-of relationship.

Definition 3.4. Topic. A topic means an event plus directly related events [6]. In this paper, a topic is denoted as a directed graph $G^* = (V^*, E^*)$, in which all the events V^* are related according to their relationship E^*.

Definition 3.5. Trace. A trace is denoted as a topic sequence, which presents the behaviour trajectory of the entity. For example, a trace TC (G_1, G_2, G_3) means that this trace includes topic G_1, topic G_2 and topic G_3.

Definition 3.6. Similar trace. A similar trace is also denoted as a topic sequence TC*(G_1*, G_2*, G_3*), the occurrence frequency in all the event relationship graphs being beyond σ (σ is the minimum support that we set).

In this paper the problem of mining periodic traces is to mine all similar traces from all event relationship graphs of the entities. We will solve the problem in the following sections:

4. Find all the event classes

A fragment of an event relationship graph is shown in Fig. 2 [11], in which each event is represented by its activity. In this section a clustering method is employed and all the events with similar activity are clustered into a group and get an identical label, such as A, B, etc. A group is named an event class.

In this section our main goal is to find all the event classes EC = {ec_1, ec_2....., ec_D}, ec_i is an event class represented as ec_i ={e_2, e_6, e_j, ..., e_n}, where all the events in ec_i have similar activity.

Finally, the event relationship graph is turned into a labelled directed graph, in which three kinds of edges lie, labelled 1, 2, 3, etc.

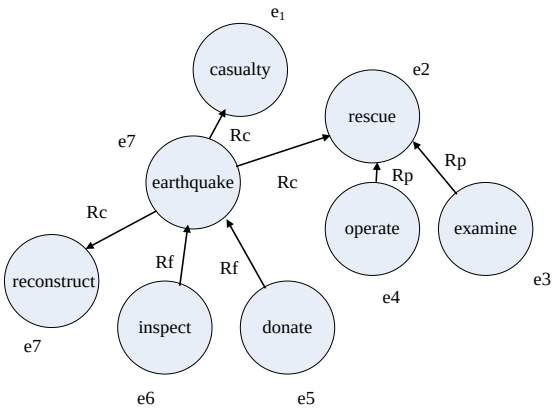

Fig. 2. A fragment of an event relationship graph

5. Significant events identification

In this paper a significant event is the event which has more frequencies than both the events before it and after it in a time window. Taking into account that a vast number of events is generated each second, it becomes a crucial task to identify the significant events. Because there is no explicit knowledge about the current or future events, we propose to handle the vast number of events [16] only by means of finding the significant events first and then increase it according to the topic.

Our approach is to exploit sliding windows [8] to identify the significant events from events with timestamp information. Next we describe the identification of significant events by statistical analysis of the events frequency. Once a significant event has been detected, we keep track of the co-occurring events with the significant event, which mainly describe the topic.

We describe next the process of significant events identification. The first step is to partition all the events data of an entity E= (e_1, e_2, e_3, ...) into fixed sized windows (w_1, w_2, w_3, ...). For each extracted event, we get its frequency as an IDF [8] value (idf(e)). In addition, we calculate the percentage of the shift of the IDF value (sidf(e)) from one event to the next one in a window. For further evaluation, we also calculate the average IDF value (avg(idf(e))) for all events in the window. If the IDF value (idf(e)) of an event is lower than the average value (avg(idf(e))), the event is filtered out. To find the most significant event e_i is to find the event with a local max IDF value. In this way, both faster and slower increasing events can be identified. In addition, to gain the ultimate goal of mining the topic sequences, we also find the second most significant events e_i' and e_i'' with the second and the third largest frequency, as Fig. 3 shows.

Then every one of the most significant events and the second most significant events are organized as a whole, which is regarded as a topic candidate. For example, all the significant events are organized as (e_1', e_1, e_1''), (e_2', e_2, e_2''),

(e_3', e_3, e_3''). Then a clustering method is employed. These topic candidates with a similar event at least are clustered into a group and get an identical topic label, such as A′, B′, etc. In addition, add the edges to each candidate topic if the events in it are related according to the corresponding event relationship graph.

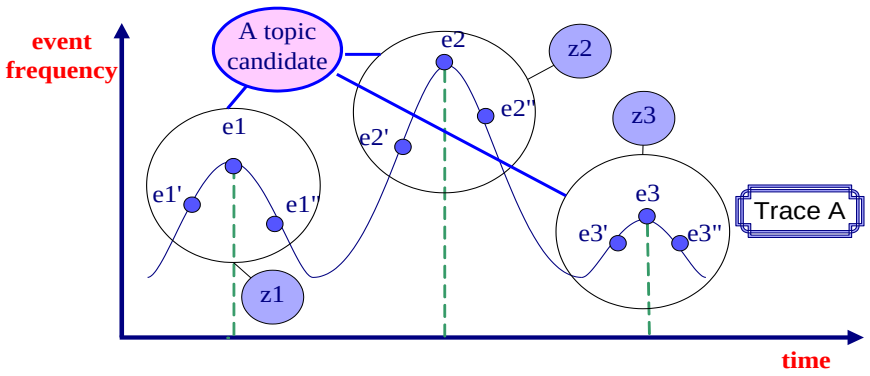

Fig. 3. Finding significant events and topic sequences

These candidate topic sequences, such as B′D′F′A′C′, are passed on to the phase of mining similar topic sequences, which is described in Section 6.

6. Mining the similar candidate topic sequences

To mine the similar candidate topic sequences, a straightforward solution could be mapping the problem onto a Local Sequence Alignment (LSA) problem. The classic LSA problem aims to identify similar subsequences in two symbolic sequences, and can be efficiently solved, using a well known dynamic programming algorithm called Smith-Waterman algorithm [9, 10]. Now we introduce the algorithm of mining the similar candidate topic sequences.

Given two candidate topic sequences $S_1 = a_0a_1...a_{m-1}$ and $S_2 = b_0b_1...b_{n-1}$, over the alphabet $\sum$ (all the topic labels), we define the matching score between any pair (a_i, b_j), where a_i, $b_j \in \sum$, as follows.

We use dynamic programming to compute the optimal alignment scores of all subsequences $S'_1=a_0a_2...a_i$, $(0 \le i \le m-1)$, and $S'_2=b_1b_2...b_j$, $0 \le j \le n-1$, and store the scores in a matrix H, as given

$$(1) \qquad H[i, j] = \begin{cases} 0 \quad \text{if} \quad i = 0 \quad \text{or} \quad j = 0, \\ H[i-1, j-1] \quad \text{if} \quad i, j > 0 \text{ and } a_i = b_j, \\ \max(H[i-1, j], H[i, j-1]) \quad \text{if} \quad i, j > 0 \text{ and } a_i \neq b_j, \end{cases}$$

where $H[i, j]$ is the length of the longest common subsequence A and B.

After obtaining the matrix H, we can get the optimally aligned subsequences between S_1 and S_2. Finally, $H[m-1, n-1]$ is the cell with the highest value in matrix H. By tracing back the values of matrix H, we can identify the longest aligned subsequence TC, as Algorithm 1 shows.

Algorithm 1. Mining the similar candidate topic sequences algorithm

Input: Two candidate topic sequences $S_1 = a_0a_1...a_{m-1}$ and $S_2 = b_0b_1...b_{n-1}$, matrix H

Output: A similar topic sequence TC

Step 1. Initialize TC ←Φ

Step 2. i←m–1, j←n–1, k←H[m–1, n–1]

Step 3. Build(k, i, j) /*build the longest aligned subsequences*/

Step 4. if (i=0 or j=0)

Step 5. return

Step 6. endif

Step 7. if (H[i][j]=H[i–1][j])

Step 8. Build(k, i–1, j)

Step 9. elseif (H[i][j]=H[i][j–1])

Step 10. Build(k, i, j–1)

Step 11. else

Step 12. $TC_i = a_{i-1}$

Step 13. Build(k–1, i–1, j–1)

Step 14. endif

Step 15. end

Step 16. Return TC

For all entities, Smith-Waterman algorithm is adopted by pairs. Finally, the longest aligned subsequences are mined.

7. Mining the similar traces

In this section we will describe the algorithm of mining similar traces, SimilarTraces, which could mine all the similar traces from the similar candidate topic sequences gained from Section 6.

A similar candidate topic sequence gained from Section 6 is represented as TC (A′, B′, D′, E′), for example. Each label tc_i, such as A′, B′, etc., in the similar candidate topic sequence, represents a graph with a support beyond the threshold σ. Each graph that the label presents, is to be mined separately [15]. Then the work of mining similar traces is completed.

For example, tc_i labeled A′ with its embedding and patterns, is given in Fig. 4.

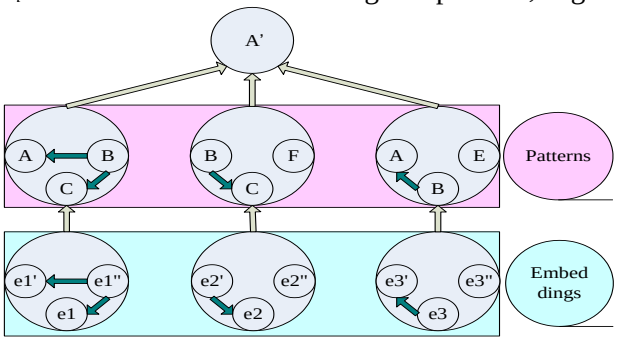

Fig. 4. tc_i labeled A′ with its embeddings and patterns

The main algorithm of mining similar traces is shown in Algorithm 1.

Algorithm 2. The similar traces mining algorithm

Input: a similar topic sequence TC = {tc_1, tc_2, …, tc_d}, event relationship graphs $G_1...G_m$ of entities, all the labels $L\{L_1, L_2, L_3,\ldots, L_i,\ldots, L_f\}$ appear in $G_1...G_m$, support threshold σ

Output: similar traces *S*

Step 1. Initialize $S \leftarrow \Phi$

Step 2. For i = 1 to d

Step 3. $T \leftarrow$ Embed(tc_i) /*find all the embeddings of tc_i*/

Step 4. $P \leftarrow$ Pattern(tc_i) /*find all the patterns of tc_i*/

Step 5. $L \leftarrow$ Label(T) /*find all the labels of T*/

Step 6. Do

Step 7. la $\leftarrow$ Max(L) /*find the label with max frequency */

Step 8. $E \leftarrow$Event(la) /*find all the events in T which label equals $L'[i]$ */

Step 9. $n \leftarrow$ Count(E, la′)
/*to find all the frequent labels that the events in E are related to */

Step 10. if ($n >= \sigma$)

Step 11. Merge(P, la′) /*add the frequent label la′ to P*/

Step 12. $L=L$ la′ /*add the label la′ to L*/

Step 13. Update(T) /*add the events to T that label la′ represents*/

Step 14. endif

Step 15. end

Step 16. $p \leftarrow$Biggest(P) /*find the biggest pattern in P*/

Step 18. Check(P, σ) /*Trim off these vertices and edges with frequency less than σ in P*/

Step 19. $S=S, p$;

Step 20. end

Step 21. Return S

In Algorithm 2 tc_i represents a label of a topic candidate. Embed(tc_i) (Step 3) is to find find all the embeddings of tc_i. Pattern(tc_i)(Step 4) is to find all the patterns of tc_i. Label(T)(Step 5) is to find all the labels of T. Max(L)(Step 7) is to find the label with max frequency. Event(la)(Step 8) is to find all the events in T the labels of which are equal to L′[i].

The function Count(E, la') (Step 9) is employed to find all the frequent labels that the events in E are related to. The function Merge(P, la′) (Step 9) is employed to add the frequent label la' to P. Update(T)(Step 13) is to add the events to T that label la′ represents. Check(P, σ)(Step 18) is to trim off these vertices and edges with frequency less than σ in P.

According to the problem of mining the similar traces, our algorithm is effective and efficient due to the following reasons:

1. Significant events help to greatly reduce the heavy cost to deal with so many events.

We only filter out the events with higher frequency than the events around them, which helps to greatly reduce the heavy cost of dealing with so many events.

2. Candidate topic sequences help to simplify the complexity of mining similar traces.

Through mining candidate topic sequences, we can gain the original similar traces deviating from the existing pattern-growth way, which greatly simplifies the mining of similar traces and improves the efficiency.

8. Experiments

In this section we have performed extensive experiments to evaluate the performance of our algorithm SimilarTraces on synthetic graphs. All experiments were done on a 2.8 GHz Intel Pentium IV PC with 1 GB main memory, in Windows 7 operating system. Our algorithm is implemented in Python 2.7.

SimilarTraces is experimented on synthetic graphs which are generated by Erdős-Rényi random network model. Erdős-Rényi model is a well-known model to generate random graphs. The graphs are constructed by function $G(n, p)$. However, the graphs generated by Erdős-Rényi model are undirected graphs, which are different from our event relationship graphs that are directed graphs. So probability of 0.5 is added to Erdős-Rényi model to decide the direction of the edge. Besides, the labels of vertices and the labels of edges are randomly distributed under the condition that any two adjacent vertices' labels cannot be identical.

In the experiment, in order to calculate the recall and precision ratio, a set of similar traces is injected into the graphs, which is to inject a set of subgraph sequences into graphs. Nowadays, despite lots of studies in graphs mining, few algorithms are capable of sub-graphs mining in big graphs because the number of frequent sub-graphs grows exponentially. In fact, lots of the patterns mined have no value. So our goal is to find some meaningful pattern sequences from graphs, that is to find similar frequent sub-graph sequences from graphs.

Data	$\|V\|$	l_V	l_e	d	n	$\|V_S\|$	E_s
1	800	130	3	4	5	8	1
2	800	130	3	4	5	15	1
3	1600	350	3	4	10	15	1
4	1600	350	4	4	15	15	1

Fig. 5. Data sets

In the experiment five peer entities are considered. We generated four different data sets (labelled Data 1 and 4) with varied parameter settings, referring to [7]. The description of the data sets is given in Fig. 5, which is for each graph. The details of the parameters are given as follows. |V| is the number of vertices of graphs; l_V is the number of vertices' labels and l_e is the number of edges' labels; d is the average degree. $|V_S|$ is the number of vertices of each injected pattern; n is the number of patterns injected; e_S is the number of instances of each pattern injected into each graph.

Figs 6-9 show the efficiency of our algorithm for four data sets in Fig. 5. The minimum support threshold is set to 2.

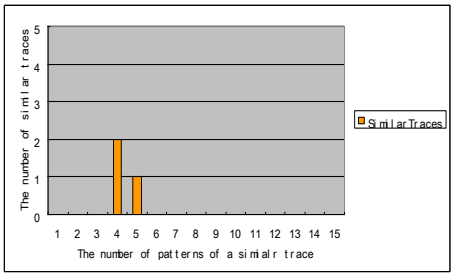

Fig. 6. Data 1

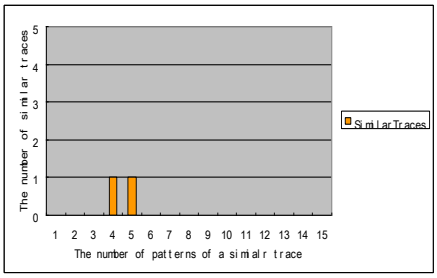

Fig. 7. Data 2

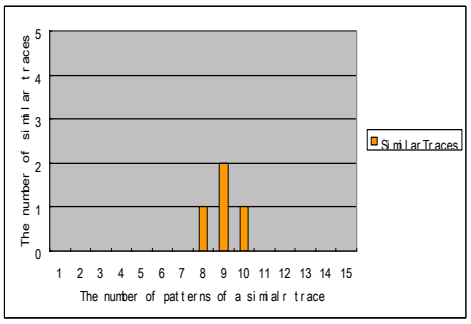

Fig. 8. Data 3

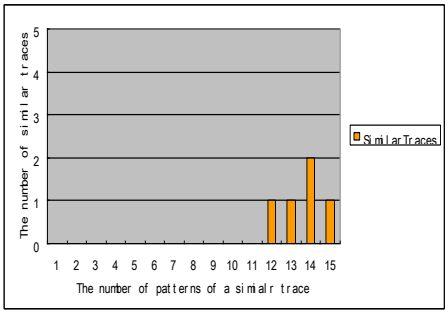

Fig. 9. Data 4

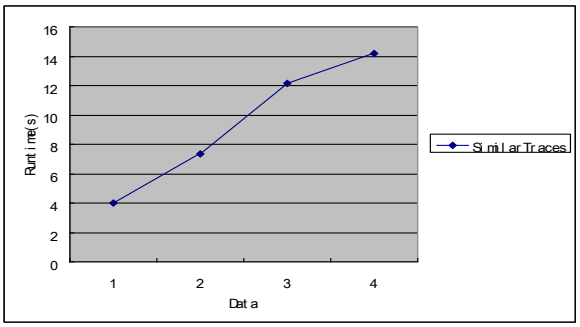

Fig. 10. Similar traces runtime

In Figs 5-9 we can see that our algorithm – SimilarTraces could mine most similar traces injected into graphs. From Data 1 up to Data 4, with a bigger graph, more and bigger patterns, SimilarTraces works well.

Fig. 10 shows that with a bigger graph, the runtime of SimilarTraces does not grow sharply.

9. Conclusion

In this paper we address a significant and difficult problem: mining similar traces of peer entities. We propose a novel and efficient framework to solve the aforementioned problem. High efficiency is achieved through filtering out the significant events and candidate topics. At the same time, through the steps of

mining similar candidate topic sequences, the production of many redundant patterns can be avoided, which can help in reducing the heavy cost of computation. In addition, our algorithm could mine similar traces with high accuracy. The experiments demonstrate efficiency, as well as scalability of our algorithm.

***Acknowledgements*:** The research work is supported by the National Natural Science Foundation of China under Grant No 61303005, Science and Technology Development Plan Project of Shandong Province No 2014GGX101019 and No 2015GGX101007, and the Fundamental Research Funds of Shandong University Nos 2014JC025, 2015JC031 and 2015JC031.

References

1. Yin, Z., L. Cao, J. Han et al. LPTA: A Probabilistic Model for Latent Periodic Topic Analysis. – In: Proc. of 11th International Conference on Data Mining (ICDM), 2011 IEEE, 2011, pp. 904-913.
2. Liu, C., K. Zhang, H. Xiong et al. Temporal Skeletonization on Sequential Data: Patterns, Categorization, and Visualization. – In: Proc. of 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, ACM, 2014, pp. 1336-1345.
3. Liu, Z., M. Huang, W. Zhou. Research on Event-Oriented Ontology Model. 2009, 36(11), pp. 191-195.
4. Yang, C. C., X. Shi, C.-P. Wei. Discovering Event Evolution Graphs from News Corpora. – IEEE Transactivitys on Systems, Man, and Cybernetics, Part A: Systems and Humans, Vol. **39**, 2009, No 4, pp. 850-863.
5. Ji, H., R. Grishman, Z. Chen. Cross-Document Event Extractivity and Tracking: Task, Evaluation, Techniques and Challenges. – In: Proc. of International Conference RANLP, Borovets, Bulgaria, 2009, pp. 166-172.
6. Liu, W., D. Wang, W. Xu et al. A Sub-Topic Partition Method Based on Event Network. – In: Proc. of 7th International Conference on Internet and Web Applications and Services, Stuttgart, Germany, 2012, pp. 194-199.
7. Giannotti, F., M. Nanni, F. Pinelli, D. Pedreschi. Trajectory Pattern Mining. – In: Proc. of SIGKDD, 2007.
8. Weiler, A., M. Grossniklaus, M. H. Scholl. Event Identification and Tracking in Social Media Streaming Data. – In: Proc. of EDBT/ICDT Workshops, 2014, pp. 282-287.
9. Smith, T. F., M. S. Waterman. Comparison of Biosequences. – Advances in Applied Mathematics, Vol. **2**, 1981, No 4, pp. 482-489.
10. Li, Z., F. Wu, M. C. Crofoot. Mining Following Relationships in Movement Data. – In: Proc. of 13th International Conference on Data Mining (ICDM), 2013, IEEE, pp. 458-467.
11. Huang, X., X. Wang, Y. Zhang et al. Mining Periodic Traces of an Entity on Web. – International Journal of Computers Communications & Control, Vol. **10**, 2015, No 5, pp. 654-666.
12. Lin, G., Q. Gui-min, Z. Xiao-Feng. An Overview of Algorithms for Mining Frequent Patterns in Graph Data. – ACTA Electronica Sinica, Vol. **36**, 2008, No 8, pp. 1603-1609.
13. Han, J., H. Cheng, D. Xin, X. Yan. Frequent Pattern Mining: Current Status and Future Directions. – Data Mining and Knowledge Discovery, Vol. **15**, 2007, No 1, pp. 55-86.
14. Wörlein, M., T. Meinl, I. Fischer et al. A Quantitative Comparison of the Subgraph Miners MoFa, gSpan, FFSM, and Gaston. Berlin, Heidelberg, Springer, 2005.
15. Holder, L. B., D. J. Cook, S. Djoko. Substucture Discovery in the SUBDUE System. – In: Proc. of KDD Workshop, 1994, pp. 169-180.
16. Zhang, J., J. Tang, J. Li. Expert Finding in a Social Network. – In: Proc. of DASFAA'07, 2007, pp. 1066-1069.