An Approach To Build Sequence Database From Web Log Data For Webpage Access Prediction

Nguyen Thon Da †, Tan Hanh †† and Pham Hoang Duy †††

Faculty of Information Systems, University of Economics and Law, VNU-HCM, VN

Posts and Telecommunications Institute of Technology, VN

Posts and Telecommunications Institute of Technology, VN

Summary
Sequential data mining is one of important topics in data mining. One of its important application is to predict a next element in a data sequence or to discover sequential rules. Many algorithms were published to tackle these problems such as sequential rule mining and sequence prediction. This paper aims to present an approach to build a sequence database from a Web log data. This step accounts for first phase of sequential data mining, titled data pre-processing phase, which transforms sequential data in Web log into sequential relational database. This sequential relational database is used as the input data of sequential data mining and sequence prediction. Besides, we present a parallel algorithm to efficiently build sequential database from Weblog files in order to predict Webpage access.

Key words:
Sequence, Sequence Database, Sequential Data Mining, Web Log Data.

1. Introduction

Nowadays, huge amounts of temporal information are stored in databases (e.g. e-learning records, customer data, biological data, patient hospital records and stock market data). Discovering temporal relationships in such databases is important in various domains, as it provides a better understanding of the data, and sets a basis for making predictions. On second thought, mining temporal relationships also provides a better understanding of the relations between events, and sets a basis for the prediction of events. For example, in international trade, one could be interested in discovering relations between the appreciations of currencies to make trade decisions [1]. A sequence is a general concept that exists in various domains. For example, in the domain of bioinformatics, protein sequences, microarray data and DNA fragments are sequences. Examples from other domains are sequences of webpages visited by users, sequences of transactions made by customers in a store, sequences of weather observations, educational data or medical record data. Similarly, the authors of the research [2] built sequence databases for Sequential Pattern Mining [3-6], Sequential Rule Mining [1, 7-9] and Sequence Prediction [10-15].

Moreover, meaningful applications of sequential data are many real-life applications such as webpage prefetching and product recommendation. Besides, an important role of sequence database is the input data for sequence prediction: Given a set of training sequences (called sequence database), the problem of sequence prediction consists in finding the next element of a target sequence by only observing its previous items [11].

In this paper, we present how to build sequence databases (datasets) for sequence prediction on webpage log data. Firstly, the concepts and notations of event, data sequence, sequence database are introduced as follows:

Let I = {i1, i2,.., im} be a set of m distinct items comprising the alphabet. An event is a non-empty set of items (without loss of generality, we assume that items of an event are sorted in lexicographic order). A sequence is an ordered list of events. An event is denoted as (i1i2,..., ik), where ij is an item. A sequence α is denoted as (α1 → α2 → ··· → αq), where αi is an event. A sequence with k items ($k = \sum_j |\alpha j|$) is called a k-sequence. SD is a set of sequences S={s1, s2,…sm} and a set of items I={i1, i2,…in} occurring in these sequences, where each sequence is assigned a unique SID (Sequence ID). For instance, Table 1 depicts a sequence database containing five sequences respectively having SIDs: seq1, seq2, seq3, seq4 and seq5. In this example, each single letter represents an item. Item(s) between curly brackets represent an itemset, ie an event. For instance, the sequence seq4 means that items e, b and f occurred at the same time, and were followed successively by a, h and g.

Table 1: An example of sequence database

SID	***SEQUENCES***
seq1	{a, b, e}, {c, f}, {g}
seq2	{a, c}, {b}, {a, g, f}
seq3	{b}, {a, f}, {e}
seq4	{e, b, f}, {a}, {h}, {g}
seq5	{g}, {e, h, f, c}

There are two types of sequential data that are commonly used in data mining time-series and sequences. A time-

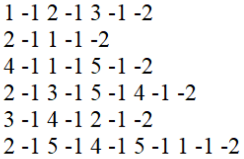
Manuscript received February 5, 2018
Manuscript revised February 20, 2018

series is an ordered list of numbers, while a sequence is an ordered list of nominal values (symbols). Both time-series and sequences are used in many domains. For instance, time-series are often used to represent data such as stock prices, temperature readings, and electricity consumption readings, while sequences are used to represent data such as sentences in texts (sequences of words), sequences of items purchased by customers in retail stores, and sequences of webpages visited by users [16]. In the scope of this paper, we consider sequences having one item in every itemset. Table 2 depicts a sequence database containing six sequences respectively and every sequence containing itemsets that has one unique item.

Table 2: A special case of sequence database

SID	***SEQUENCES***
seq1	{1}, {2}, {3}
seq2	{2}, {1}
seq3	{4}, {1}, {5}
seq4	{2}, {3}, {5}, {4}
seq5	{3}, {4}, {2}
seq6	{2}, {5}, {4}, {5}, {1}

According to the paper [2], the sequence database in Table 2 could be presented as Figure 1.



```
1 -1 2 -1 3 -1 -2
2 -1 1 -1 -2
4 -1 1 -1 5 -1 -2
2 -1 3 -1 5 -1 4 -1 -2
3 -1 4 -1 2 -1 -2
2 -1 5 -1 4 -1 5 -1 1 -1 -2
```

Fig. 1 A dataset (sequence database) in SPMF format.

The figure 1 is a dataset where each line represents a sequence from a sequence database. Each item from a sequence is a positive integer and items from the same itemset within a sequence are separated by single spaces. Note that it is assumed that items within a same itemset are sorted according to a total order and that no item can appear twice in the same itemset. The value "-1" indicates the end of an itemset. The value "-2" indicates the end of a sequence (it appears at the end of each line).

This paper presents how to convert Web log files into a sequence database automatically. The sequential data in a sequence database is used to an input data for predicting the next item of a data sequence. For example, a customer buy product A, then buy product B, and lastly buy product D, we will conclude/predict which product the customer will buy in the future. A similar example, when a user visit a Website, the user will visit separately link 1, link 2, link 3 … in time order. And the given question is “Which next link users could visit next?” Back to designing sequence database, we explain a few ideas as follows:

With regard to Web logs, we consider every link as an itemset (a special items with an item). We also could consider every sequence as series of actions of visiting links. For example, in TABLE 2, a user plays a role as seq1 visiting link {1}, then visiting link {2}, and link {3} is visited in the last time by the user. Similarly, a user with sequence seq6 visited links {2}, {5}, {4}, {5}, {1} by time order. Issues need to address here are how to convert data from a Web logs into a sequence database. We will present more details in next sections.

This paper is organized as follows. In section 2, we formally present related work. In section 3, we present an approach to collect data from Web log files, explain how its sequence database are built. In section 4, we describe an experimental study. Finally, in section 5, we present our conclusions.

2. Web Log Data

Analyzing users' Web log data and extracting their interests of Web-watching behaviors are important and challenging research topics of Web usage mining [17]. Web usage mining is the task of discovering the behavior of the users while they are accessing through the Web. The user access log files provide useful information about a web server. The aim of understanding the navigation preferences of the visitors is to enhance the quality of electronic commerce services (e-commerce), to personalize the Web portals or to improve the Web structure and Web server performance [18]. An important research in Web mining is discovering log files. They contain information about User Name, IP Address, Time Stamp, Access Request, number of Bytes Transferred, Result Status, URL that Referred and User Agent. A log file can be located in three different places: Web Servers, Web proxy Servers, Client browsers [19]. In the scope of this paper, we consider one of type of Web Server Logs. That is Access log: The server access log records all requests that are processed by the server. The location and content of the access log are controlled by the Custom Log directive. The Custom Log directive is used to log requests to the server. [19]. An example of access log is presented as follows:

```
1.53.85.12 - - [21/Aug/2017:01:22:57 +0200] "GET /fileadmin/inees.org/files/logs/access.log.5 HTTP/1.1" 200 3285724 "http://www.inees.org/fileadmin/inees.org/files/logs/" "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/60.0.3112.101 Safari/537.36"
157.55.39.23 - - [21/Aug/2017:01:23:41 +0200] "GET /robots.txt HTTP/1.1" 404 430 "-" "Mozilla/5.0 (compatible;bingbot/2.0; +http://www.bing.com/bingbot.htm)"
```

Based on the properties of Web Log Data, we introduce how to shorten the size of Web Log Data in the next section in order to build sequence databases.

3. Proposed Approach

In this section we present three main parts. In the first part, we introduce how to present a sequence database from Web logs file. In the next two parts, we propose an approach for building database sequence from Web log files.

3.1 Build database sequence from Web Log Data

To build database sequence, we collected Web log files from a Website. In this paper, we focus on two kinds of common Web Server. We consider four fields we use to design a sequence database: c-ip (Log the IP address of the client that made the request), cs(Referer) (the site that the user last visited. This site provided a link to the current site.), date (Log the date on which the activity occurred) and time (Log the time in Coordinated Universal Time (UTC), at which the activity occurred).

Through Log Parser Studio (see https://goo.gl/a2J1ak), we retrieve data from a folder containing Weblog files. Then, we create a table like Table 3. This table contains three fields: User_IP (from c-ip field of Weblog files), Link (from cs-uri-stem field of Weblog files), Action_Time (from date, time field of Weblog files).

*Phases to design a sequence database from Weblog files is presented as follows:

3.1.1 Phase 1

Using the log parser tool (https://goo.gl/a2J1ak) , we sort the table by User_IP, then by Action_Time (by time ascending).

Table 3: Table created from Web Log Data

User_IP	***Link***	***Action_Time***
176.9.34.172	Link_visited_1	17:14:11, 12-May-2017
176.9.34.172	Link_visited_4	05:17:21, 18-May-2017
176.9.34.172	Link_visited_5	12:14:12, 20-May-2017
182.92.18.13	Link_visited_3	11:14:16, 12-Apr-2017
182.92.18.13	Link_visited_2	18:04:23, 14-Apr-2017
182.92.18.13	Link_visited_5	21:12:28, 15-Apr-2016
182.92.18.13	Link_visited_6	09:14:23, 17-Apr-2017
170.23.11.67	Link_visited_2	17:14:19, 05-Jun-2017
170.23.11.67	Link_visited_3	06:17:25, 17-Jun-2017
170.23.11.67	Link_visited_1	08:14:06, 23-Jun-2017
177.80.22.38	Link_visited_7	08:14:25, 18-May-2017
177.80.22.38	Link_visited_4	07:14:16, 19-May-2017

3.1.2 Phase 2

Split every group of User_IPs to form a separate sequence, every sequence contains visited links presenting by time ascending. With above example, we have the following sequence database:

Sequence 1: Link_visited_1 -1 Link_visited_4 -1 Link_visited_5 -1 -2

Sequence 2: Link_visited_3 -1 Link_visited_2 -1 Link_visited_5 -1 Link_visited_6 -1 -2

Sequence 3: Link_visited_2 -1 Link_visited_3 -1 Link_visited_1 -1 -2

Sequence 4: Link_visited_7 -1 Link_visited_4 -1 -2

3.2 An approach to build sequence database from Web Log Files

The pseudo code of the non-parallel approach is presented as follows:

Input: A folder containing Web log files of a Website

Output: A list of sequences (a database sequence)

Step 1: Open the connection to connect to Web Log Files (Using the Log Parser library)

Step 2: Execute the query to get distinct field *User_IP* and field *Link_visited* from Web Log Files.

Step 3: Perform the approach (presented by the Pseudo Code)

```
N ← Size of lines containing in log files;
Arr_User_IP ← Array containing User IPs in log files
Arr_Link ← Array containing Links in log files
for i = 0 to N-1 do
    Arr_User_IP(i) ← values of the field User_IP
    Arr_Link(i) ← values of the field Link_visited
end for
Arr_Distinct_User_IP ←Array to store different User IPs
Arr_Distinct_Link← Array to store different Link_visited
count ← Number of visited links.
count ← i
for k = 0 to count do
    for l = 0 to Count(Arr_Distinct_Link) do
        if Arr_Link(k) ← Arr_Distinct_Link(l) then
            Arr_Link(k) ← Arr_Link(k) + “ -1 “
        // “ –1 “ : use to distinguish a link from another link
    end for
end for
for j = 0 to count – 1 do
    if Arr_User_IP(j) < > Arr_User_IP(j + 1) then
    Arr_Link(j) ← Arr_Link (j) + “ –2 \r\n“
// “-2”: To distinguish a line from another line
end for
List ← null: Array to store sequences
for j = 0 to count do
```

```
26. //Choose every sequence containing 3 links or more
27.     if (Number_of_Links ≥ 3)
28.         Add Arr_Link(j) to List
29. end for
30. Output: A list of sequences (A database sequence)
```

The main idea of the above approach is described as follows

Firstly, the input data is a set of many Web log files contained in a folder. Secondly, the approach is performed. Finally, we will obtain a sequence database. The explanation of the approach is presented as follows:

* From Line 1 to Line 9: Find an array containing different User IPs called Arr_Distinct_User_IP and an array containing different visited links (called Arr_Distinct_Link).

* From Line 10 to Line 18: Indicates that an array containing distinct visited links where a visited link is separated from each other by the symbol "-1". Every visited link is encoded by order number in the list of distinct visited links.

* From Line 19 to Line 23: With every different User IP, there is a group of distinct visited links. Every group is separated from each other by the symbol "-2". These groups are sequences in the sequence database.

* From Line 24 to Line 29: Gradually collected sequences to create the database sequence.

3.3 An parallel approach to build sequence database from Web Log Files

Like the section 3.2, we propose an parallel approach for building sequence database. The pseudo code of the parallel approach is presented as follows:

Input: A folder containing Web log files of a Website

Output: A list of sequences (a database sequence)

Step 1: Open the connection to connect to Web Log Files (Using the Log Parser library)

Step 2: Execute the query to get distinct field *User_IP* and field *Link_visited* from Web Log Files.

Step 3: Perform the approach (presented by the Pseudo Code)

```
1. N ← Size of lines containing in log files;
2. Arr_User_IP ← Array containing User IPs in log files
3. Arr_Link ← Array containing Links in log files
4. for i = 0 to N-1 do
5.     Arr_User_IP(i) ← values of the field User_IP
6.     Arr_Link(i) ← values of the field Link_visited
7. end for
8. Arr_Distinct_User_IP ←Array to store different User IPs
9. Arr_Distinct_Link← Array to store different Link_visited
10. count ← Number of visited links.
11. count ← i
12. (Parallel) for k = 0 to count do
13. (Parallel) for l = 0 to Count(Arr_Distinct_Link) do
14          if Arr_Link(k) ← Arr_Distinct_Link(l) then
15.         Arr_Link(k) ← Arr_Link(k) + " -1 "
16.       // " –1 " : use to distinguish a link from another link
17. end (Parallel) for
18. end (Parallel) for
19. (Parallel) for j = 0 to count – 1 do
20.     if Arr_User_IP(j) < > Arr_User_IP(j + 1) then
21.     Arr_Link(j) ← Arr_Link (j) + " –2 \r\n"
22. // "-2": To distinguish a line from another line
23. (Parallel) end for
24. List ← null: Array to store sequences
25. for j = 0 to count do
26. //Choose every sequence containing 3 links or more
27.     if (Number_of_Links ≥ 3)
28.         Add Arr_Link(j) to List
29. end for
30. Output: A list of sequences (A database sequence)
```

Explanation the pseudo code (Parallel):

* From *Line 1* to *Line 9*: This is similar to the preferred approach.

* From *Line 10* to *Line 23*: This is also similar to the preferred approach. However, For Loops are performed in a parallel manner.

* From *Line 24* to *Line 29*: This is similar to the preferred approach.

4. Experimental Results

We have performed several experiments to compare the performance of the non-parallel approach and the parallel approach for build sequence database. In section 4.1 through 4.4, we will present this in detail.

4.1 Hardware configuration

We use a PC with the following hardware configuration:
RAM: 32 GB (31.6 GB usable); Intel(R) Core(TM) i7-4800MQ CPU @ 2.70GHz.

4.2 Software configuration

Operation System: 64-bit Windows 10 Education;
Programming Environment: C# 2013, Log Parser Studio

4.3 Data

Datasets (Web Log Data) presented in this paper are collected from 4 Websites as folows:
Website 1: periwinklelecottages.com
Website 2: palmviewsanibel.com
Website 3: devqa.robotec.co.il

Website 4: inees.org
Preferred websites have a few of information illustrated in Table 5.

Table 5: Information from Web Log Data

	Web site 1	Web site 2	Web site 3	Web site 4
Number of transactions	121836	85683	2527429	593367
File size (MB)	97449	74814	629	119
Number of Unique User IPs	61015	40901	7405	1188
Number of Unique Links	4267	3535	5467	451

4.4 Experimental results

After we run the preferred non-parallel and parallel algorithms, we obtained the results depicted in Table 4 and Figure 2.

Table 4: Execution time of the proposed approach

	Web Site 1	Web site 2	Web site 3	Web site 4
Non-Parallel *(miliseconds)*	121836	85683	12382	3508
Parallel *(miliseconds)*	97449	74814	9893	3312
Number of built sequences	12211	9678	312	330

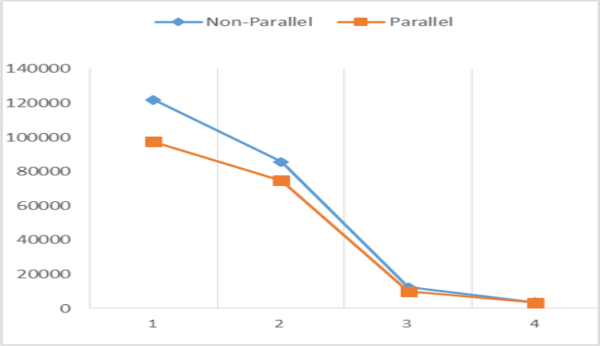


Fig. 2 Compare non-parallel algorithm and parallel algorithm in terms of execution for building sequence databases.

According to Figure 2, we realize that when the larger data size is, the less execution time is when we run the parallel algorithm to build sequence databases (see Fig. 2).

5. Conclusion

This paper has provided a detailed method of building sequence databases based on Web log Data. Moreover, the sequence database created could use as the data input of various data mining algorithms, especially in sequential rule mining and sequence prediction. In addition, the approach shows that the parallel algorithm presented is much better than non-parallel algorithm in terms of execution time.
In the future, we aim to further improve the execution time of our approach. We believe that better results can be achieved by performing in a network environment. Therefore, we also plan to design a distributed system for getting much larger datasets. Besides, some promising areas of research that we intend to find out more intensively are the design of parallel, distributed, multi-core, and GPU-based algorithms.
Furthermore, an important research opportunity is to apply built sequential databases in new applications. One of the most interesting and promising possibility is to utilize built sequence databases for input data in emerging research fields such as the internet of things, improved prediction models and sensor networks.

Acknowledgments

We are grateful to Full Professor Philippe Fournier Viger (Director of Center of Innovative Industrial Design, Harbin Institute of Technology (Shenzhen)) for his helpful comments and suggestions in order that we could finish this paper.

References

[1] P. Fournier-Viger, R. Nkambou, and V. S.-M. Tseng, "RuleGrowth: mining sequential rules common to several sequences by pattern-growth," in Proceedings of the 2011 ACM symposium on applied computing, 2011, pp. 956-961: ACM.
[2] P. Fournier-Viger, A. Gomariz, T. Gueniche, A. Soltani, C.-W. Wu, and V. S. Tseng, "SPMF: a Java open-source pattern mining library," The Journal of Machine Learning Research, vol. 15, no. 1, pp. 3389-3393, 2014.
[3] R. Agrawal and R. Srikant, "Mining sequential patterns," in Data Engineering, 1995. Proceedings of the Eleventh International Conference on, 1995, pp. 3-14: IEEE.
[4] J. Ayres, J. Flannick, J. Gehrke, and T. Yiu, "Sequential pattern mining using a bitmap representation," in Proceedings of the eighth ACM SIGKDD international conference on Knowledge discovery and data mining, 2002, pp. 429-435: ACM.
[5] D.-Y. Chiu, Y.-H. Wu, and A. L. Chen, "An efficient algorithm for mining frequent sequences by a new strategy without support counting," in Data Engineering, 2004. Proceedings. 20th International Conference on, 2004, pp. 375-386: IEEE.

[6] J. Pei et al., "Mining sequential patterns by pattern-growth: The prefixspan approach," IEEE Transactions on knowledge and data engineering, vol. 16, no. 11, pp. 1424-1440, 2004.

[7] P. Fournier-Viger, T. Gueniche, S. Zida, and V. S. Tseng, "ERMiner: sequential rule mining using equivalence classes," in International Symposium on Intelligent Data Analysis, 2014, pp. 108-119: Springer.

[8] P. Fournier-Viger, U. Faghihi, R. Nkambou, and E. M. Nguifo, "CMRules: Mining sequential rules common to several sequences," Knowledge-Based Systems, vol. 25, no. 1, pp. 63-76, 2012.

[9] P. Fournier-Viger, C.-W. Wu, V. S. Tseng, L. Cao, and R. Nkambou, "Mining partially-ordered sequential rules common to multiple sequences," IEEE Transactions on Knowledge and Data Engineering, vol. 27, no. 8, pp. 2203-2216, 2015.

[10] T. Gueniche, P. Fournier-Viger, R. Raman, and V. S. Tseng, "CPT+: Decreasing the time/space complexity of the Compact Prediction Tree," in Pacific-Asia Conference on Knowledge Discovery and Data Mining, 2015, pp. 625-636: Springer.

[11] T. Gueniche, P. Fournier-Viger, and V. S. Tseng, "Compact Prediction Tree: A Lossless Model for Accurate Sequence Prediction," in ADMA (2), 2013, pp. 177-188.

[12] V. N. Padmanabhan and J. C. Mogul, "Using predictive prefetching to improve world wide web latency," ACM SIGCOMM Computer Communication Review, vol. 26, no. 3, pp. 22-36, 1996.

[13] J. Pitkow and P. Pirolli, "Mininglongestrepeatin g subsequencestopredict worldwidewebsurfing," in Proc. USENIX Symp. On Internet Technologies and Systems, 1999, p. 1.

[14] P. Laird and R. Saul, "Discrete sequence prediction and its applications," Machine learning, vol. 15, no. 1, pp. 43-68, 1994.

[15] J. Ziv and A. Lempel, "Compression of individual sequences via variable-rate coding," IEEE transactions on Information Theory, vol. 24, no. 5, pp. 530-536, 1978.

[16] P. Fournier-Viger, J. C.-W. Lin, R. U. Kiran, Y. S. Koh, and R. Thomas, "A survey of sequential pattern mining," Data Science and Pattern Recognition, vol. 1, no. 1, pp. 54-77, 2017.

[17] T. Murata and K. Saito, "Extracting users' interests from web log data," in Web Intelligence, 2006. WI 2006. IEEE/WIC/ACM International Conference on, 2006, pp. 343-346: IEEE.

[18] R. Iváncsy and I. Vajk, "Frequent pattern mining in web log data," Acta Polytechnica Hungarica, vol. 3, no. 1, pp. 77-90, 2006.

[19] L. J. Grace, V. Maheswari, and D. Nagamalai, "Web log data analysis and mining," Advanced Computing, pp. 459-469, 2011.

Nguyen Thon Da received Master degree in Computer Science from the University of Technology, VNU-HCM in 2013. In November 2016, he was accepted as a Ph.D Student in Information Systems at Posts and Telecommunications Institute of Technology, Vietnam. He is now working as IT employee and an assistant teacher at Faculty of Information Systems, University of Economics and Law, VNUHCM. His research interests include data mining, pattern mining, sequence analysis and prediction.

Tan Hanh received the PhD degree from Grenoble Institute of Technology, France. Currently, he is vice president of Posts and Telecommunications Institute of Technology. His research interests are machine learning, information retrieval, and data mining.

Pham Hoang Duy strongly connects with IT-related research and education. From 2005 to 2009, he was working on his PhD research project tackling the problem of knowledge representation and defeasible reasoning in multi-agent systems at the University of Queensland in Australia. His research and teaching interests are data-mining techniques and their applications, especially in the domain of computer systems’ security.