Algorithm for Mining Sequential Pattern in Time Series Data

Chong Zhu , Xiangli Zhang , Jingguo Sun , Bin Huang
Guilin University of Electronic Technology, Guilin, 541004, China
E-mail: zhuch@mails.guet.edu.cn

Abstract

Mining sequential pattern in time series data is broadly used in a variety of areas in order to make a prediction, and an appropriate model should be established before the prediction can be done, therefore, the way how to mine out time series pattern from time series database becomes extremely important. Based on data of the time series database, this paper presents a new frequent time series pattern mining algorithm, which constructs a tree-projection at first, then uses priority depth strategy to traversal the tree-projection in order to mine out all the longest frequent patterns, the paper has descripted the new algorithm with pseudo code in detail. Experimental results demostrate that this algorithm has mined out the frequent series, which meets the real-time restraints successfully. Furthermore, under the same condition and different support situation, the new algorithm has obtained the same ruleset as the traditional AprioriAll method but more effective performance.

1. Introduction

Time series is a group of ordered time-varying values or events, mathematically, when a random process is sampled at a series of instants: $t_1, t_2, \cdots$, t_n (t is a independent variable,also, $t_1 < t_2 < \cdots <$ t_n), we get a set of sequential values: $X_{t1}, X_{t2}, \cdots$ X_{tn}, which is a discret digital time sequence, i.e., a sample of the random process $X(t)$. All the samples of the random process are stored in a time series database, which is broadly used in a variety of areas, such as data analysis in science experiments,fluctuating stock price prediction.,etc. An appropriate model should be established before the prediction can be done, therefore, the way how to mine out time series pattern from time series database becomes extremely important.

Time series is often influenced by other factors, such as finacial time series. Finacial time series consist of a number of sequential different values at different time points. That figuring out the trend of the fluctuating price in time，which is always influenced by a variety of information，is a emphasis and also a difficulty in time series analysis. Frequently changed finacial data can be used to compare the effectivities of different trading systems in terms of making a price prediction and studying the price fluctuating of an specific stock. So far, there are quite a lot of reports about studies in data mining of time series. Referrence literature[1] presented Apriori algorithm--a heuristic search strategy used to search the frequent events sets in series of events database; Apriori algorithm was used in events series frequent patterns analysis, the frequent patterns mining algorithm of events series was presented in reference literature[2]; furthermore, the strategy mentioned above was used in time series association rules analysis and [3] presented a method to partion the time series by a window with a fixed length. However, the method is not so good because it's not easy to determine the length of the window and the calculation is complicated.

Based on the time series database, This dissertation presents a new frequent time series patterns searching algorithm, which constructs a tree-projection at first, and then, it traversals the tree-projection using priority depth strategy to mine out all the longest frequent patterns. Experimental results demonstrated that this algorithm has mined out the frequent series successfully, which satisfies the restraints. Under the same conditions but different support situation, the same ruleset as the traditional AprioriAll method can be obtained and more effective when using the algorithm presented in this paper, the new method can meet the real-time requirements.

978-0-7695-3501-2/09 $25.00 © 2009 IEEE
DOI 10.1109/CMC.2009.208

2. Data pre-processing

Nonstationarity is one of the most conspicuous characteristics of the time series. Strictly, nonstationarity means random process's static characteristics doesn't change according to time. However, we often mean general unstability which implies that both the expected value and the covariance are independent of the time start point when we say nonstationarity. Stationarity is the basical condition of many time series model. However, time series usually behaves unstably because it is impossible to maintain its expected value and covariance unchanged due to the time-varying factors such as economical, political, culture environments that influence the market.

For example, finacial time series is a nonstationarity data sequence, but it changes according to its inherent rules. It is very common to predict by working out its varying mode. Specifically, the prices of primary products that wave slightly most of the time may fluctuate randomly due to the policies or abnormal weather such as a snow storm, a continuous drought and so forth. This fluctuation can be viewed as noise signal. Our purpose of data mining is to find out the inherent changing rules of series. The effect of data mining can be influenced grievously because the noise will weaken the rules and probably result in some fake rules. It is necessary to preprocess the time series, like smoothing[4], clustering[5][6] and so on to suppress random noise before data mining.

3. Algorithm design

Definition 1

A sequence just like $S = \{A_{t1}, A_{t2}, A_{t3}, \cdots A_{tn}\}$ ($t1 < t2 < t3 \cdots < tn$) is called a time sequence, A stands for the observing property and ti ($1 \le i \le n$) stands for the observing instant respectively.

Definition 2

The number of items included in a sequence like $S = \{A_{t1}, A_{t2}, A_{t3}, \cdots A_{tn}\}$($t1 < t2 < t3 \cdots < tn$) is called sequence length, and a sequence with a length of l will be marked as l-sequence.

Definition 3

The times that the sequence: $S_1 = \{A_{t1}, A_{t2}, A_{t3}, \cdots A_{tm}\}$ ($t1 < t2 < t3 \ldots < tm$) appears in the time series database $S = \{A_{t1}, A_{t2}, A_{t3}, \cdots A_{tn}\}$ ($t1 < t2 < t3 \cdots < tn$) is called support factor, marked as ξ, and ($1 \le m \le n$).

Definition 4

We call the sequence $S_1 = \{A_{t1}, A_{t2}, A_{t3}, \cdots A_{tm}\}$ with a support factor not lower than the given threshold ξ is a frequent series, marked as $Support(S_1)$.

Question description: a given time series database and the minimum threshold of support factor, frequent time series patterns searching is to find out the longest sequence in the time series database that appears at least n times, n equal to the threshold of support factor.

3.1. Tree-projection constructing

For example, assuming $S = \{A_{t1}, A_{t2}, A_{t3}, \ldots A_{tn}\}$ ($t1 < t2 < t3 \cdots < tn$) is a time sequence, and $A_{ti} \in \{I_1, I_2\}$, sequence $\{I_1, I_2, I_1, I_2, I_1, I_1, I_2, I_2, I_1, I_1\}$ is one sample of S, the window length w =3, and the threshold ξ=2.

Tree-projection constructing: firstly, we create the root node of the tree and mark it as "null". Next take w records from the beginning of the sequence to create a branch, <(I_1 :1),(I_2 :1),(I_1 :1)> which contains 3 nodes: (I_1:1) is a child node of the root node, (I_2:1), which has a child node (I_1:1) is a child node of (I_1:1). Each node has two items: the value and the count value. The value I_1 appears twice, but we can't put them in one node because the appearance time of the nodes should also be considered during time sequence data mining. Then the time window shifts one item backward, get the subsequence $\{I_2, I_1, I_2\}$. We create a child node (I_2:1) of root node which is the second branch of the tree-projection, <(I_2:1),(I_1:1),(I_2:1)>. Next the time window shifts one item backward again to extract another subsequence $\{I_1, I_2, I_1\}$. We find I_1 when searching the childnodes of the root node, so the count value of this node increases by 1, and then, we come to its childnodes, when I_2 is found the according count value should be increased by 1, and the count value of the child node I_2 of I_1 increased by 1, likewise until the last subsequence $\{I_1\}$, finally, we get the tree-projection shown in figure 1.

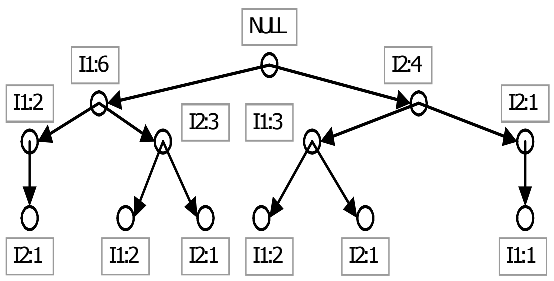

Figure 1. Tree-projection

3.2. Frequent sequential patterns mining

We start searching frequent patterns from the root node, traversaling the tree-projection using priority depth strategy after tree-projection constructing. First, get the childnode (I_1:6) of the root node, since the count value of this node is bigger than the support factor ξ, we continue to the childnodes of (I_1:6), get the first childnode (I_2:3) of (I_1:6). Again, node (I_2:3) appears more than ξ times, so we continue to its childnodes. The count value of its first node (I_1 : 2) which has no child node is 2. Therefore, subsequence $\{I_1\ I_2\ I_1\}$ is added into frequent sequences set, and then, back to node (I_2:3), discovering the count value of its second node (I_1:2) is lower than the threshold, we go backward to (I_1:6) and start searching from its second childnode (I_1:2), the count value is equal to the threshold, continue. But the childnode (I_2:1) of (I_1:2) appears only once, then put $\{I_1\ I_1\}$ into frequent sequences set and backward to the root node. So far, frequent patterns searching in the first subtree has been finished. Then we search the second subtree in the same way and find the frequent sequence $\{I_2\ I_1\ I_1\}$, so we get the frequent sequences set: $\{\{I_1\ I_2\ I_1\},\{I_1\ I_1\},\{I_2\ I_1\ I_1\}\}$. As we can see, the length of the sequence in the set is 2 at least and 3 at most. Searching each 2-sequence in 3-sequences, we find $\{I_1\ I_1\}$ is a subsequence of $\{I_2\ I_1\ I_1\}$, so remove $\{I_1\ I_1\}$ from the set. Since the length of the frequent patterns that we found is equal to the length of the window, it is possible there are still longest frequent patterns that haven't been discovered yet. We get this 4-sequence $\{I_1\ I_2\ I_1\ I_1\}$ by connecting frequent 3-sequences which appears once in the original sequence and it is not a frequent sequence. Data mining ends here now and we get a frequent sequences set $\{\{I_1\ I_2\ I_1\},\{I_2\ I_1\ I_1\}\}$.

4. Algorithm description

Input: time sequence $S=\{A_{t1},A_{t2},A_{t3},\dots A_{tn}\}$ ($t1<t2<t3\ \dots<tn$), A stands for the observing property and ti ($1\le i\le n$) stands for observing instant respectively; Minimum support factor is ξ, threshold time window length is w.

Output: the longest frequent series set.

```
(1) Tree-projection constructing
  (a)built_tree:
    Create the root node , mark as “null”.
      1) Get m items of the sequence into
         Sub_list at a time;
      2) if the number of items is less than 2,
         break;
      3) else insert_tree(Sub_list).
  (b) insert_tree(Sub_list):
  Assuming now_node is root node,X[i] is the i-
  th item in the Sub_list,the initial value of i is 0;
      1) If now_node has a childnode whose
         value is X[i], we assuming this
         childnode: node, the count value
         increased by 1, now_node=node;
      2) else create a new childnode node for
         now_node initialized as X[i], and the
         count value is 1, now_node=node;
      3) i++, if(i<Sub_list.length) return 1,
         else tree-projection constructing
         completed.
(2) Frequent pattern mining
  (a) search(node):
   If the count value is not lower than $\xi$, then
   callsearch(childnode) for all the childnodes of
   node;
  If there are no childnodes or all the childnodes'
  count value is lower than the threshold $\xi$, add
  this subsequence into the frequent sequences
  set, return;
   Return;
  (b) Call search(root), then we will get the
  original frequent series set. Next, analyze the
  set and remove the fake longest frequent
  sequence. If the length of the longest frequent
```

sequence is equal to the time window and more than 1, we should connect the longest frequent sequences in the set to get candidate sequences, then search each candidate sequence in the original sequence just in the same way as the AprioriAll algorithm. If there are at most 1 longest frequent sequence whose length is no less than the time window, the algorithm ends.

5. Experimental results

We extract the data of the cabbage price during the period of February, 2004 to January, 2008 in An'hui province from website www.sounong.net. And the price resolution is 0.01 yuan. The cabbage price in this period fluctuates from 0.7 to 2.1 per kilogram.

Operating environment: PC, Pentium® D915 CPU, 1024M RAM, Win2k operating system, Java should be imployed to implement the algorithm. The sample number is 1144, support factor threshold is 3, and window length is 77.78ms of Time elapsed before we get 112 frequent sequences that meet the restraints, referring to table 1 for the details.

Table 1. Frequent patterns searching results

Length	Amount
L=5	1 entry, eg:(0.91,0.92,0.93,0.94,0.94).,etc
L=4	3 entry, eg:(1.09,1.07,1.06,1.06).,etc
L=3	17 entry, eg:(1.06,1.05,1.05).,etc
L=2	91 entry, eg:(0.85,0.83),(0.85,0.84).,etc

We compared the algorithm in this paper with the traditional AprioriAll algorithm under the same conditions, the results indicate that we get the same rules set as AprioriAll but more effective performance, please refer to figure 2,

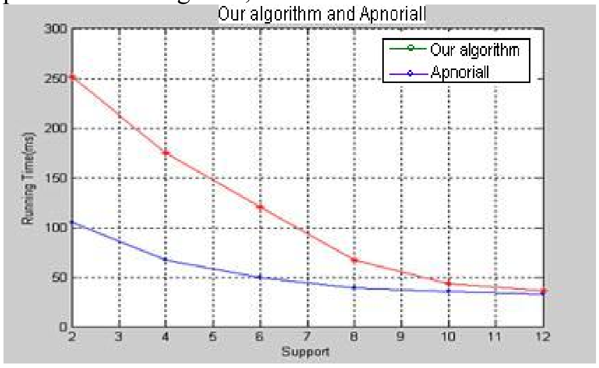

Figure 2. Time consuming

According to figure 2, our algorithm is apparently more effective than tranditonal algorithm AprioriAll, the most time consuming is 107ms which can totally meet the real-time requirements of time sequence prediction.

6. Conclusions

Based on data of the time series database, this dissertation presents a new frequent time series patterns finding algorithm, Experimental results demonstrated that this algorithm has mined out the frequent series successfully, which satisfies the real-time restraintes. And, under the same condition and different support situation, it has obtained the same ruleset as the traditional AprioriAll method gets but more effective performance. We are going to the next stage of multi-sequences study, trying to find out the associations between the sequences, and then figure out the reference value of the frequent patterns of multi-sequences in helping us to see the relationships among all the time sequences more clearly in order to make more reasonable decisions[7,8,9].

7. References

[1] Agrawal R, Ramakrishnans, Fast algorithms for mining association rules in large databases[A]. *Proceedings of the Twentieth International Conference on Very Large Databases[C]*, Santiago:ACM Press, 1994, pp. 487-499.

[2] Manila H, Toivonen H, Verkamo A I, *Discovery frequent episodes in sequences[A]*. Proc of KDD95[C], 1995.

[3] Das G, Lin K, Mannila H,et al, Rule discovery from time series[A], *Proceedings of Fourth Annual Conference on Knowledge Discovery and Data Mining[C]*, NewYork:AAAI Press,1998, pp. 16-22, Montreal:AAAI Press, 1995, pp. 210-215.

[4] C.O.Kwok, O. Etzioni, and D.S. Weld, Scaling Question Answering to the Web [J]. *ACM Trans. Information Systems*, 2001, 19 (3):242-262.

[5] Shi Zhongzhi, *Knowledge& discoveries*, Tsinghua press, Beijing, 2002.

[6] S.D.Whitehead, Auto-FAQ: An experiment in cyberspace leveraging. *In Proceedings of the Second International WWW Conference*, 1995, volume 1:25-38.

[7] Wang Xiaoye, *A study of similarity and trend prediction in data mining of time series[D]*. P.H.D Tianjin: Tianjin Univesity, 2003.

[8] Huang he, A study of a quick mining patterns in time series[J], *Computer engineering and application*, 2003, 39(21):192-194.

[9] S.Oyama, T.Kokubo, T. Ishida, Domain-Specific Web Search with Keyword Spices[J], *IEEE Trans. Knowledge and Data Eng*, 2004, 16(1):17-27.

8. Acknowledgements

The authors would like to thank the projects (No.0832246 and No.0832084) supported by Science Foundation of Guangxi Province of China and the project:Innovation Project of Guangxi Graduate Education (No.2008105950810M420) in supporting this work.