An Efficient Curve Extraction Algorithm for Massive Data

Zhongwen Li[1, a], Long Li[2], Guoxin Li[3], Yan Li[1]

[1]College of Engineering and Design, Hunan Normal University, Changsha, 410081, China

[2]China Patent Technology Development Company, Beijing, 100088, China

[3]Computer Lab. of China University of Political Science and Law, Beijing, 100088, China

[a]lee_zw@163.com

Keywords: curve extraction, time-series, database query, time-series database, curve drawing

Abstract. An efficient curve extraction algorithm for massive time-series data based on extreme value of periods is proposed. By dividing into periods, then calculate the number of sampling points within a time period according to extreme value of the each time period's data. Uniformly take the sampling points through a database query mechanism, the experimental results show that compared with traditional algorithms, the number of points needed for sampling can be assigned, and the drawn curve has a good approximation of the original curve in the case that the number of access points has been pre-determined; it is able to greatly shorten the time of curve query and drawing.

Introduction

Database that consists of time-series data is called time-series databases. When the data is stored, how to query and analyze the existing data becomes difficult and important.Current researches on time-series data are basically focused on similarity search [1,2], time-series segmentation and pattern discovery[3], and time-series prediction[4], etc., where a lot of research results are achieved. But as a research field with promising application prospects [5], research on time-series visualization is relatively less. Now there are some researches, and in addition, corresponding visualization tools are developed, such as time series on spirals [6], time searcher [7], vizTree [8], timeseries bitmaps [9] and so on. Domestic research in this field is less.

In order to facilitate analysis, a more commonly used method is to retrieve data from the database, and show the data in form of curves, enabling data analysts to intuitively analyze trends of retrieved data and local data. However, due to the high time density of data, retrieving so much data all at once for curve drawing is not possible: on the one hand, retrieving data from the database takes a lot of time, and such a long time waiting for response is unacceptable for the user; on the other hand, such a huge data amount will consume a lot of memory space of analysis software, and drawing all the data is not feasible when the amount of data becomes larger.

At present, there is no better solution to one time off curve query and display of data of such large amount.

Algorithm description

The basic idea of the proposed method in this paper is: to divide the whole time interval that needs query analysis into segments, acquire different number of data points respectively in each segment according to a certain strategy, the data point collection obtained by each section is treated as a point collection of trend of change of the whole time interval.

Point extraction and method execution are discussed separately below.

Basic Assumption. If total number of points needed for extraction is *pcount*, which can be set by the user or automatically by the program according to the user's display screen resolution. The total time is divided into *n* sections.

Number of Points in each Time Section. According to the general statistical law, for any two time periods with the same span, the one with bigger extreme value difference will accommodate

more data change information, namely, hidden with more curve changes, therefore, more data points should be collected; and for the time period with smaller extreme value difference, data change smoothly in its range, and less data points may be selected. Therefore, taking more points from the section with larger changes and fewer points from that with gentle changes can reflect the general data changes in a better way when the overall number of point is fixed.

In order to describe the degree of data change, in this paper we use absolute distance di between values to reflect value change in i time section, i.e.:

$$d_i=max_i-min_i. \quad (1)$$

Where max_i is the maximum data of the *i* period, min_i is the minimum data of the *i* time period.

According to Eq. 1, the sum of absolute distances of all segments is

$$sum=\sum_{i=1}^{i=n} d_i \quad (2)$$

The number of points taken in section *i* is determined according to the ratio of absolute distance d_i of each time period in the sum of absolute distances, i.e.

$$p_i=d_i/sum\times pcount \quad (3)$$

For Eq. 3, the determination of p_i is applicable for when di is not 0, when di is 0, indicating that the time section has no value change, then take p_i =2, so seen from the two circumstances, the value of p_i is:

$$\begin{cases} d_i/sum\times pcount & \text{when } d_i \text{ is not } 0 \\ p_i=2 & \text{when } d_i=0 \end{cases} \quad (4)$$

In the above Eq. 1, Eq. 2, Eq. 3, Eq. 4, the range of *i* is $0<i\le n$. When calculating p_i according to Eq. 3 the results are round up to integer.

Number of Segment. Let the total data number of the time region that needs query and analysis be total, when the extracted number *pcount* is fixed, it obvious that the more segments are divided the more likely that the extracted points can reflect the change trend of original data, therefore we have

$$n=total/pcount \quad (5)$$

However, due to the fact that the more *n* is, the more system resources are consumed. Therefore, according to experience the value of *n* should generally not beyond 50.

Extraction Algorithm

The pseudo-code of the extraction algorithm is as Fig.1.

```
Process name：Extraction Algorithm
Input 1： pcount (means the number of data points that need to be extracted)
Input 2: timeSE (means time period that the user wants to query)
Output：tendency curve of time-series
Code：
    // Obtain the total number of data total from the database in the time period that the user wants to query
    total=getAllPointCount(timeSE);
   if(total<=pcount)
          drawAll(); // directly query all the time-series, draw curve for the user to check and analyze
  else {      // otherwise get point by follow strategy
          n=getSegmentCount(total, pcount); // divide the time interval into n sections by equal data amount according to
equation (5)
          scount= total/n；//get point number of each time period
       // query start and end time of each segment, then store them in data structure segTimeSet
          segTimeSet=querySegmentTimeSet(scount, timeSE);
        //query the maximun and minimun value of each segment, then store them in data structure minMaxSet
          minMaxSet=queryMinMaxSet(segTimeSet);
          drawMinMax(minMaxSet); // draw curve with the data structure minMaxSet
          for(i=0; i<n; i++){
                  pi=getPointCountOfSegment(pcount,minMaxSet); //get the number of points of each segment
          drawThreadi.start(); //according the number of points and the time of segment period, start thread for curve drawing
        }
    }
END
```

Fig.1. pseudo-code of the extraction algorithm

Experiment and Analysis

Presently there is a test data management application system for spacecraft. The main functions provided by the application system including the storage, query and analysis of spacecraft mass test data. As for the application system, fields of main data tables in its database are time, [parameters 1], [parameter 2], [parameter 3]..., wherein the time is the primary key, with accuracy of millisecond, and the time column stored in the database is long integer data type. The database system is Oracle9i database. The data query and analysis subsystem of the application system makes query and curve drawing on test data on basis of the extraction algorithm in this paper.

Upon finishing curving drawing of the present experiment, the curve displayed in curve display view is as shown in Fig.2. The total number of data points drawn is 3010, because p_i is calculated by round up, the number of total points taken will only approximate to *pcount*. By contrasting, Fig.3 shows a curve drawn without using this method. We can see that they look quite the same, with much less data points, Fig.2 gives perfect character of Fig.3, and by using the algorithm hereby proposed, its time consumption is only 5 seconds, far less than 572 seconds for full data extraction. When using the Douglas-Poiker method or the vertical distance limit value method, which requires data process, its time consumption will be more than 572 seconds.

From the above experiment we can see that the algorithm proposed here is better both time wise and space wise than other extraction algorithms, and with good curve character extraction effect.

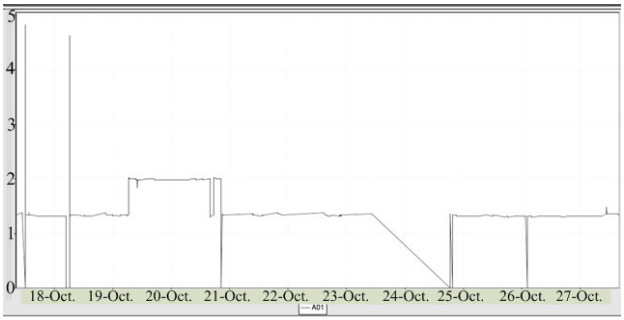

Fig.2. Graph of sectional extreme values curve

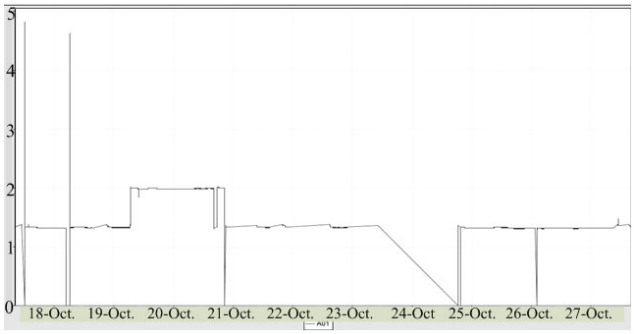

Fig.3. Graph of all data points query curve

Conclusions

This paper presents an efficient curve extraction algorithm for massive time-series data based on segmented extreme value. First the time range for query and analysis is to be segmented, and then decide the number of sampling points of the time period according to absolute distance of data of each time period and the total number of sampling points, to achieve uniformly sampling of points through the query mechanism of database itself. The experimental results show that the algorithm proposed in this paper is better both time wise and space wise than other extraction algorithms, and with good curve character extraction effect.

Acknowledgements

The work was supported by Hunan Provincial Natural Science Foundation of China (No. 13JJ6029), the Program for Excellent Talents in Hunan Normal University (No. ET13108), the Youth Innovation Fund of the Third China High Resolution Earth Observation Conference, and HUNAN Universities Innovation platform Project(No. 13K032).

References

[1] Yang Y, Papadopoulos S , Papadias D, Kollios G. Authenticated indexing for outsourced spatial databases. VLDB Journal, DOI 101 1007/ s0077820082011322 , 2008 : 1-18
[2] Bueno R , Traina A J M , Traina Jr C. Genetic algorithms for approximate similarity queries. Data and Knowledge Engineering , 2007 , 62 (3) : 459-482
[3] Qisong Chen, Xiaowei Chen, Yun Wu. Optimization Algorithm with Kernel PCA to Support Vector Machines for Time Series Prediction. Journal of computers. 2010, Vol 5, No 3 : 380-387.
[4] [12] Claudio Ribeiro, Ronaldo Goldschmidt, Ricardo Choren,A Reuse-based Environment to Build Ensembles for Time Series Forecasting. Journal of Software, 2012, Vol 7, No 11 (2012), 2450-2459.
[5] YU J, HUNTER J, REITER E, et al. Recognising visual patterns to communicate gas turbine time-series data[C]. App lications and Innovations in Intelligent Systems. London: Springer, 2002:105-108.
[6] Weber M., Alexa M., Muller W. Visualizing Time-Series on Spirals[C]//IEEE Symposium on Information Visualization. California:IEEE Computer .
[7] Society Technical Committee on Visualization and Graphics. 2001. Hochheiser H., Shneiderman B., Dynamic Query Tools for Time Series Data Sets: Timebox Widgets for Interactive Exploration. Information Visualization, 2004, 3(1): 1-18.
[8] Jessica Lin, Eamonn Keogh, Stefano Lonardi. Visualizing and discovering non-trivial patterns in large time series databases. Information Visualization, 2005. 4(2): 61-82.
[9] Nitin Kumar, Nishanth Lolla. Eamonn Keogh Time-series bitmaps: a practical visualization tool for working with large time series databases[C]//SSIAM 2005 Data Mining Conference.Newport Beach:SIAM, 2005.