基于蜡烛图的海量时序数据可视化方法

作者 1[1]，作者 2[2]

（1.北京理工大学计算机学院，北京 100081；
2.中国中国兵器工业第 201 研究所，北京 100081）

摘　要：时序数据是按时间次序观测到的数据集合，高频采样的时序数据造成大量的数据堆积，未经处理的原始数据由于数量巨大，通常难以观测和利用。本文提出一种针对时序大数据的可视化方法。该方法对输入的时序数据首先进行高效线性压缩，将时序数据划分为多个数据区间，计算每个区间用于表征数据特性的统计数据，并采用蜡烛图和折线图对数据序列进行可视化，所提出的方法计算复杂度低，可视化效果直观。本文进一步探讨了使用分布式计算系统 Spark 对海量的时序数据进行预处理和区间计算，大幅提高数据的处理效率。

关键词：时序数据；数据可视化；Spark
中图分类号：V211　　**文献标识码**：A

A visualization method for time-series data base on candlestick charts

A[1], A[2]

(1. Department of Electronic Engineering, School of Information Science and Technology, Beijing Institute of Technology, Beijing 100081, China
2. Department of Electronic Engineering, School of Information Science and Technology, Beijing Institute of Technology, Beijing 100081,China)

Abstract: Time-series data is a collection of data obtained by time sequence. A visualization system for huge amounts of time-series data is designed and implemented in this paper. According to the dimensions of time-series data and points number after compressing each dimension data, the data were divided into several intervals. For each interval, the effective data were extracted and calculated from all data, and were used for getting the characteristics of the interval data. Due to the large amount of data, Spark was chosen to calculate. After extracting the character data of each interval, we chose the candlestick charts in order to display the data better. Through the candlestick charts, we can easily understand the data change.

基金项目：XXX
作者简介：作者 1（1963—），男，教授，工学博士，E-mail: xxx@xx.com

Key word: Time-series data; Visualization; Spark

1 引言

时序数据是指按时间次序排列观测得到的数据集合[1]。时序数据存在于社会生活中的各个领域，如科学研究中的天文观测、气象数据，金融经济中的股票交易信息等。目前随着物联网和工业智能不断发展，通过传感器采集到的时序数据的数量和维度不断增加。庞大的数据量使得这些数据的加工和处理面临着巨大的挑战。一方面庞大的数据量对于应用系统是一个难以承受的负担；另一方面，用户很难从中找到有用的关键信息。

数据可视化将晦涩的数字通过适当图形形式展示出来，通常直观易于观察，对于发现数据的特征和进一步理解系统的行为有更大的帮助。但是，数据可视化不是单纯的将庞大的数据进行显示，这是因为数据量巨大使得显示变得臃肿不堪，用户难以找到数据中的关键信息；此外，庞大的数据量对于任何一个可视化系统来说都是一个难以承受的负担，通常会导致系统崩溃或者数据可视化中有效信息的不可视。

考虑到现有应用多建立在关系数据库的基础之上，本文提出一种基于蜡烛图的海量时序数据可视化方法，该方法对输入的时序数据首先进行高效线性压缩，将时序数据划分为多个数据区间，计算每个区间用于表征数据特性的统计数据，并采用蜡烛图和折线图对数据序列进行可视化。所提出的方法不仅能够对原始数据进行大幅压缩，并且在压缩数据的基础之上可进一步按照比例分层压缩显示。在较好保留原始数据特征的基础之上，提供了适用于不同场景和需求的可视化方法。该方法首次将蜡烛图引入到工业时序数据的可视化技术中来，并且为了提高海量时序数据的预处理效率，给出了基于 Spark 的分布式并行区间压缩方法。

本文首先介绍了时序数据可视化的相关工作，接着给出了时序数据预处理，即数据进行区间划分的基本方法，然后给出了按照指定的比例对数据区间进行可视化的基本方法，最后基于 Spark 探讨了对海量时序数据进行高效预处理的方法。

2 相关工作

时间本身充满了多变性，它可以是一个时间点，也可以是一个时间间隔，可以是线性的，也可以是周期性的。再加上时序数据单维度和多维度的区别，使得时序数据的可视化也呈现了多样性。很多情况下，时序数据的可视化被量身定制。SimVis[2]能够有效的对模拟数据可视化，新闻报道数据可以使用 ThemeRiver[3]分析。

时序数据可视化也可以通用化，时间被看作一个维度或映射到某个空间中。一般可视化框架有 XmdvTool[4]和 Visage[5]等标准可视化技术。而且平行坐标，复杂图标，都可以用来可视化时序数据，对于数据量比较少的数据，这些框架足以胜任，但是海量时序数据的可视化仍然具有较大的挑战性。

根据时间的特征，可以将时序数据的可视化分为线性时间可视化，分支时间可视化和周期时间可视化。ThemeRiver[3]是线性时间可视化的典型例子，可以通过类似于河流的图像有效的显示时序数据的特征，但是数据量太过庞大会导致数据的显示比较拥挤，难以突出数据的有效特征。对于周期性时间可视化，Shen[6]等人利用圆环图设计了一种有效的周期性数据可视化的工具，能有效的显示周期性数据的特征。分支时间可视化的用处也相对广泛，例如，Lifelines[7]可以可视化病人病例和历史档案。Liu 等人利用 Storyflow[8]描述电影的情节发展和人物的关系演变。

无论是时序数据的通用可视化方法还是根据时间特性和数据特性对时序数据的可视化方法，都不能够有效的解决数据量庞大带来的可视化拥挤的问题。

为了解决数据量庞大导致的可视化问题，本文选取蜡烛图作为时序数据的可视化方法。蜡烛图(candlestick chart)又称 K 线图。蜡烛图反映了一段时间内金融产品价格波动趋势。一张完整的蜡烛图是由多个单位时间下的 K 线合成，蜡烛图的横轴表示时间，纵轴则是表示价格或者点数。在股市系统中，它是以每个分析周期的开盘价、最高价、最低价和收盘价绘制而成。通常使用蜡烛图及其相应的技术来对一些数据进行分析和趋势的预测。本文在蜡烛图的基础之上增加了平均值和标准差来描述一个区间内数据的特征，能够较好的表达当前数据的规律及趋势。

3 时序数据预处理

时序数据的预处理是指将时序数据转化为数据区间列表的过程。本文定义一个数据区间为 $R=<v_s, v_e, v_{max}, v_{min}, v_{avg}, v_{var}>$，其中 v_s 和 v_e 分别表示区间内第一个数据值和最后一个数据值,v_{max}，v_{min}，v_{avg} 和 v_{var} 分别表示区间内数据点的最大值、最小值、平均值和标准差。

对于给定的分割区间大小 H，时序数据预处理的目的是对 H 个连续的数据点进行线性扫描的过程中，计算这些数据点的统计特征，并将这些数据点压缩为一个数据区间。之所以选择区间开始值、结束值、最大值、最小值、平均值和标准差是因为这些统计数据反映了一个区间内数据的基本特征，并且这些统计值在区间合并过程中具有良好的特性，即可以根据两个相邻区间的统计特征值计算获得这两个区间合并之后的数据区间的统计特征值。

时序数据预处理的算法如下所示：

算法：数据预处理

输入：分割区间大小 H，原始数据序列 S

输出：区间列表 *R*

```
BEGIN
    counter = 0;
    FOR d in S
        IF (counter % H == 0) {
            创建一个新的区间 I = <d, d, d, d, d, 0>;
        }ELSE{
            num = (counter % H + 1);
            I=<I.v_s, d, MAX(I.v_max,d),MIN(I.v_min, d),(I.v_avg*num+d)/(num+1), l.var+d*d>;
            IF(counter % H == H-1) {
                I = <I.v_s, d, MAX(I.v_max,d), MIN(I.v_min,d), (I.v_avg * num + d)/(num + 1),
                  sqrt((l.var-l.v_avg*l.v_avg)/H)>;
                RL.push(I);
            } //END-IF
        } //END IF-ELSE
    }//END FOR
    RETURN RL;
END;
```

如表 1 所示，D_1、D_2、D_3 和 D_4 为同一系统采集的时序数据，以 D_4 为例，设置区间间隔为 800；顺序扫描时序数据，保存起始数据点 *Start*(D_4)，终止数据点 *End*(D_4)，并计算 D_4 列数据中每个区间 $H_i(i=1,2,...,n)$的最大值 *Max*(D_4)，最小值 *Min*(D_4)，平均值 *Average*(D_4)，标准差 *Variance*(D_4)，处理输出结果如下表 2 所示。

表 1 原始时序数据 D_1–D_4

时序	D_1	D_2	D_3	D_4

1	25651	1024	0	-50
…	…	…	…	…
800	25651	0	0	25
801	25651	0	0	25
…	…	…	…	…
1600	25634	0	880	24
1601	25634	0	880	24
…	…	…	…	…
65255	29235	0	739	78

经过预处理之后的数据由于大幅压缩，且每个数据区间的描述属性字段固定，因此可以在关系数据库中存储和查询。

表 2 数据列 D_4 处理输出结果

区间	$Mzx(D_4)$	$Min(D_4)$	$Start(D_4)$	$End(D_4)$	$Average(D_4)$	$Variance(D_4)$
1	25	-50	-50	25	12.95	26.95
…	…	…	…	…	…	…
20	38	32	32	38	34.25	1.90
…	…	…	…	…	…	…
40	50	48	50	48	48.97	0.66
…	…	…	…	…	…	…
81	89	85	87	85	87.21	1.11

3.3 时序数据可视化

经过数据预处理后，数据会被分隔为多个大小均为 H 的数据区间并存储在关系数据库中。之后，时序数据可视化都是基于这些数据区间完成的。然而考虑到原始数据量巨大，因此即使压缩为数据区间之后，数据区间的个数仍然较多。例如当 H=100 时，百万条数据经过预处理之后获得上万个数据区间，如果全部显示则会导致数据点过于密集；如果将 H 设置为特别大的值，例如 H=10000，则获得的区间数量只有 100，但是数据的很多局部特性丢失。因此，本文提出对已有区间进行合并运算，并使用合并之后的区间进行可视化，合并的粒度可以根据用户的需要决定。

定义数据预处理之后的初始区间数据为元区间数据，区间值为 H，每个区间的定义为 $R=<v_s, v_e, v_{max}, v_{min}, v_{avg}, v_{var}>$，。设数据可视化阶段设定的区间大小为 kH，则需要由元区间数据生成以 kH 为区间值的数据区间。定义以 kH 为区间值的区间数据为 $R'=<\alpha_s, \alpha_e, \alpha_{max}, \alpha_{min}, \alpha_{avg}, \alpha_{var}>$。需要合并的元区间中第 i 个元区间的表示方法为 $R_i=<v_{is}, v_{ie}, v_{imax}, v_{imin}, v_{iavg}, v_{ivar}>$，则 R'中每个值的计算方法如下。

$$\alpha_s = v_{1s} \tag{1}$$

$$\alpha_e = v_{ke} \tag{2}$$

$$\alpha_{max} = \max(v_{1max}, v_{2max} \dots v_{kmax}) \tag{3}$$

$$\alpha_{min} = \min(v_{1min}, v_{2min} \dots v_{kmin}) \tag{4}$$

$$\alpha_{avg} = \frac{1}{k}\sum_{j=1}^{k} v_{javg} \tag{5}$$

$$\alpha_{var}^2 = \frac{\sum_{j=1}^{k} v_{javg}^2}{k} + \frac{\sum_{j=1}^{k} v_{jvar}^2}{k} - \frac{(\sum_{j=1}^{k} v_{javg})^2}{k^2} \tag{6}$$

上面的公式给出了根据元区间数据计算以 kH 为区间值的数据区间特征值的方法。对于 R'中开始值，结束值，最大值，最小值，平均值的求解过程都比较清晰易懂。对于标准差的计算表达式可以通过简单的推导得出，通过将α_{var}^2与合并的 k 个元区间的方差之和做差，消去原始数据，即可得到 R'中标准差的计算表达式。

4 基于 Spark 的可视化系统

4.1 可视化系统设计

考虑到原始时序数据量巨大，处理时间较长，为了提高数据预处理的效率，本文基于分布式处理框架 Spark 构建相应的可视化系统。如图 1 所示，该系统分为三部分，数据存储、数据计算和数据可视化。

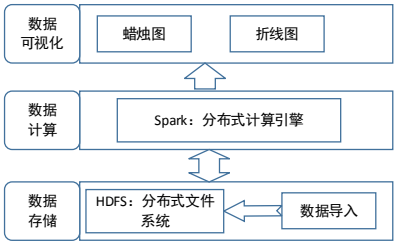

图 1 可视化方法结构图

4.1.1 数据存储

数据存储的功能主要是保存用户上传的数据，并且与计算引擎进行数据交换。由于本文的方法是面向海量时序数据的，所以采用分布式计算架构完成原始数据的存储和预处理工作。本文选择 Hadoop 分布式文件系统(HDFS[9]) 作为数据存储系统。

4.1.2 数据计算

数据计算的主要功能是计算能够表征区间特性的特征数据，比如平均值，最大值，最小值，标准差等，计算过程并不复杂，但是由于数据量比较庞大，所以必须保证计算引擎有足够的计算能力和容错性。由于 Spark[10]具有丰富的算子、良好的容错性、高效的计算能力，现在被越来越多的商业和研究机构使用，所以在数据处理中采用 Spark 作为计算引擎。

4.1.3 数据可视化

数据可视化的功能主要是将数据计算层处理过后的数据使用各种图形表示出来。系统中使用蜡烛图表示某单维时序数据列在某一段区间内的最大值、最小值、初始值、结束值的变化，同时使用折线图将数据列区间的平均值、标准差的变化展示出来。蜡烛图中的最高价对应区间内数据的最大值，最低价对应最小值，开市价对应初始值，收市价对应结束值。目前可以使用编程方法或者一般图形绘制方法绘制蜡烛图，折线图。最后我们选择使用 Matlab 绘制图形。

4.2 可视化方法实现

本系统中 Spark 运行在一个拥有 8 个计算节点的集群中，集群的内存总容量为 28GB，CPU 总量为 22 个，运行的 Spark 版本为 1.4.1，Spark 运行模式为 Standalone。另外，系统的存储环境是分布式文件系统 HDFS，其默认数据备份为 3。Spark 有很好的编程语言接口，其

应用程序支持 Java、Scala、Python、R 等主流编程语言。我们选择 Java 开发我们的应用程序。系统中所需的应用程序设计如图 2。

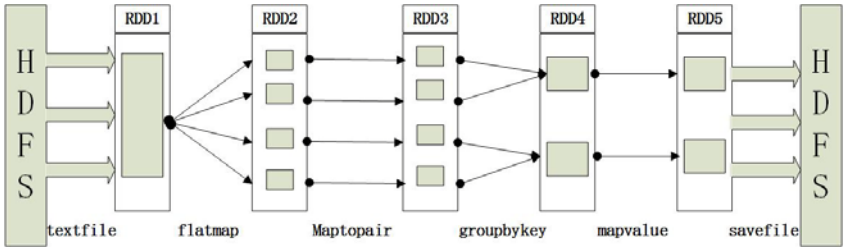

图 2Spark 处理数据流程图

如图 2，在 Spark 中，数据处理经过 6 个过程：首先从 HDFS 读取数据形成 RDD1；RDD1 按行分隔形成多个 RDD2，每个 RDD2 表示一行数据；RDD2 经过 maptopair 处理，转化为 Key－value 型数据 RDD3，Key 为 RDD2 每行第一个数字（即行标号）整除区间值之后取整的结果，Value 为 RDD2；将 RDD3 中相同 Key 的数据进行聚合形成 RDD4，RDD4 为 value 型数据，每个 RDD4 表示一个区间数据集；最后对每个 RDD4 内的数据进行处理，提取最大值、最小值、平均值、标准差、区间开始值、区间结束值、区间标号等形成 RDD5，每个 RDD5 表示一个区间的特征值；将所有 RDD4 处理完之后，将得到的 RDD5 存储到 HDFS 中。

5 实验分析

为了实现状态监测、故障预警和诊断，车辆传动装置内部安装了大量传感器。在传动装置车载试验过程中，这些传感器采集的数据将持续不断的发送到上位机电脑中。以车辆转向加速试验为例，采集到的数据包括档位及方向、发动机转速、发动机回水温度、风扇转速等多达 40 个参数变量。以每秒 10 次的采样速率计算，装置持续运转一个小时会收集到 72000 条数据记录。我们以这些数据作为实验的基本，利用我们的系统对数据进行可视化。

针对不同维度，我们进行不同区间的划分，进而通过对比评估本系统的性能。我们选取两种波动性区别比较大的数据，如图 3 与图 7 所示为原始数据相应的折线图。图 3 表示的数据具有如下特点：数据整体会出现剧烈的波动，且波动幅度大。图 7 展示的数据比较平滑，没有急剧的波动，整体呈现上升的趋势，但是偶尔存在数值下降的区间。如图 4-6 和图 8-10，蓝色线是蜡烛图，蜡烛图中每个区间包含的数据包括区间最大值，最小值，起始值，结束值。红色线表示区间平均值折线图。黑色线表示区间标准差折线图。图 11 与图 12 分别为图 6 与图 10 的一部分显示，能够更清楚的看到蜡烛图的构成。

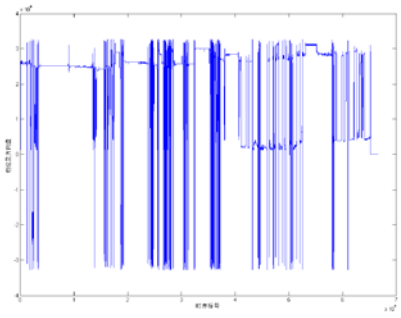

图 3 档位及方向盘数据

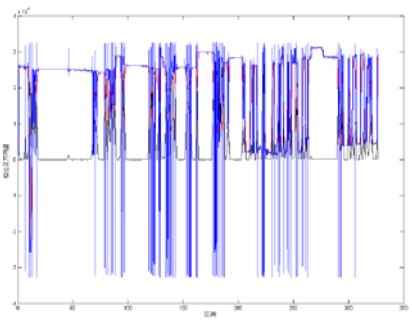

图 4 区间值为 200 的可视化结果

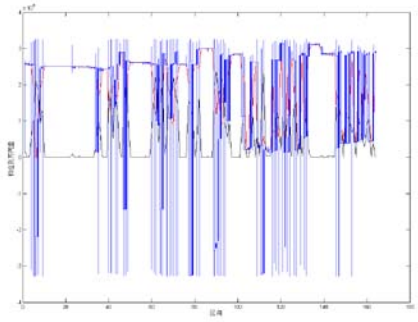

图 5 区间值为 400 的可视化结果

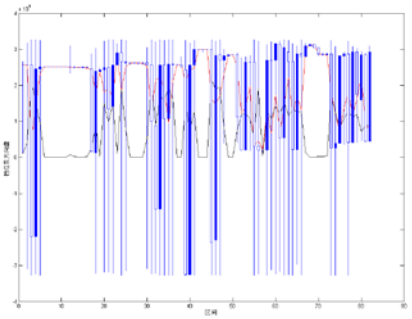

图 6 区间值为 800 的可视化结果

利用本系统对图 3 的数据进行可视化，图 4，图 5，图 6 分别对应区间值分别为 200，400，800。区间值表示多少个连续的值组成一段区间。根据图 4-6，可以发现蜡烛图能够很好的展示数据的波动性，对比原图中局部数据的剧烈变化，图 4-6 都能够比较好的还原出原始数据的波动，且数据的整体特性保存的比较好。另外，随着数据压缩比的增加（区间增大），数据的局部剧烈波动明显减少。

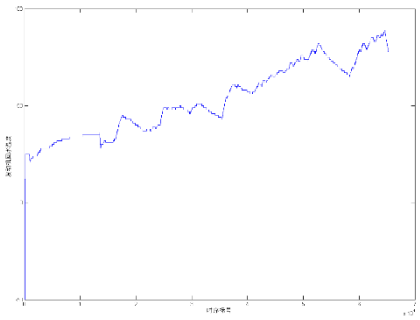

图 7 发动机回水温度

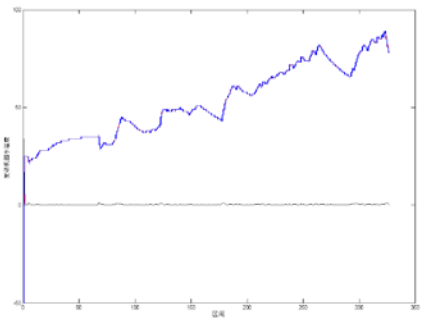

图 8 区间值为 200 的可视化结果

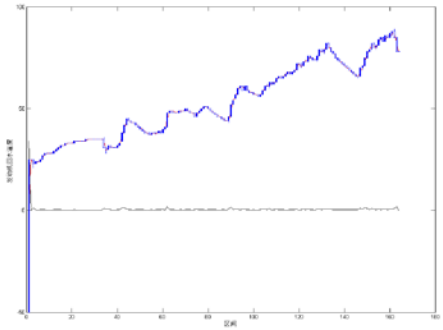

图 9 区间值为 400 的可视化结果

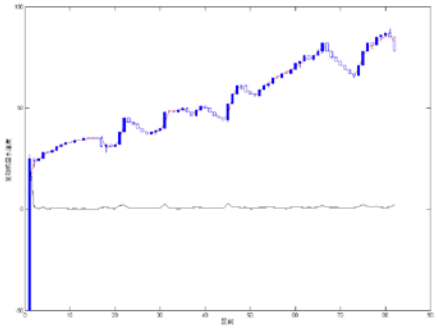

图 10 区间值为 800 的可视化结果

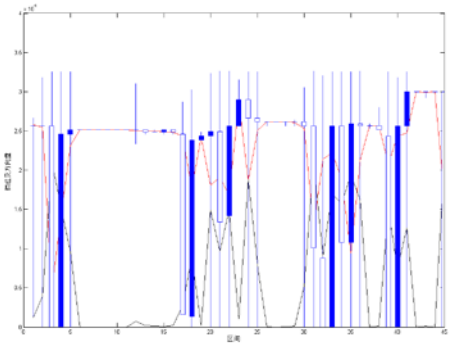

图 11 蜡烛图（图 6）

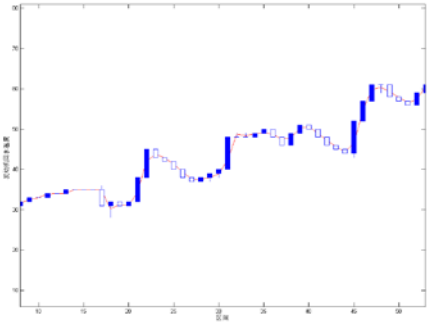

图 12 蜡烛图（图 10）

图 7 的呈现的数据整体稳定增长，偶尔会有区间下降。图 8，9，10 分别对应区间值为 200，400，800 的数据可视化结果，可以发现，三个可视化结果同样呈现增长趋势，且局部区间的下降也能够被还原，数据的整体特性保存的非常好。随着数据压缩比的增加，数据的局部特性有所遗失，但是整体波动特性并没有影响。

根据实验结果可以发现，本文提出的基于蜡烛图的时序数据可视化方法较好地保留了数据的整体特性。随着数据压缩比的增加，数据局部特性有所遗失，但是整体的波动性还是能够得到很好的体现。

6 总结

本论文提出一个基于蜡烛图的海量时序数据可视化方法。对于大数据时代下的海量时序数据，能够进行快速的处理。每一个数据区间提取的统计数据能够真实反映该区间的数据特征，将计算得到的数据通过蜡烛图，折线图等反馈给用户，使用户能够直观的获取数据的特征。本系统对大数据有良好的处理能力，可视化效果明显，能直观的反映数据各个阶段的波动性，对数据原有信息保存比较完整。

参考文献：

[1]吴璟. 时序数据的特征提取及其应用的研究[D]. 中国石油大学(北京), 2009.

[2] Gemmell J, Bell G, Lueder R. MyLifeBits: a personal database for everything[J]. Communications of the ACM, 2006, 49(1): 88-95.

[3] Havre S, Hetzler E, Whitney P, et al. Themeriver: Visualizing thematic changes in large document collections[J]. Visualization and Computer Graphics, IEEE Transactions on, 2002, 8(1): 9-20.

[4] Ward M O. Xmdvtool: Integrating multiple methods for visualizing multivariate data[C]//Proceedings of the Conference on Visualization'94. IEEE Computer Society Press, 1994: 326-333.

[5] Kolojejchick J, Roth S F, Lucas P. Information appliances and tools in Visage[J]. Computer Graphics and Applications, IEEE, 1997, 17(4): 32-41.

[6] Shen Z, Ma K L. Mobivis: A visualization system for exploring mobile data[C]//Visualization Symposium, 2008. PacificVIS'08. IEEE Pacific. IEEE, 2008: 175-182.

[7] Plaisant C, Mushlin R, Snyder A, et al. LifeLines: using visualization to enhance navigation and analysis of patient records[C]//Proceedings of the AMIA Symposium. American Medical Informatics Association, 1998: 76.

[8] Liu S, Wu Y, Wei E, et al. Storyflow: Tracking the evolution of stories[J]. Visualization and Computer Graphics, IEEE Transactions on, 2013, 19(12): 2436-2445.
[9] Borthakur D. HDFS architecture guide[J]. HADOOP APACHE PROJECT http://hadoop. apache. org/common/docs/current/hdfs design. pdf, 2008.
[10]Zaharia M, Chowdhury M, Franklin M J, et al. Spark: cluster computing with working sets[C]//Proceedings of the 2nd USENIX conference on Hot topics in cloud computing. 2010, 10: 10.