基于斜率提取边缘点的时间序列分段线性表示方法*)

詹艳艳　徐荣聪　陈晓云

(福州大学数学与计算机科学学院　福州 350002)

摘　要　本文引入解析几何中的斜率，提出了一种新颖的基于斜率提取边缘点的时间序列分段线性表示方法 SEEP。对于斜率变化范围比较集中的时间序列，SEEP 表示方法有着非常好的效果，与以往的分段线性表示方法相比，SEEP 表示方法与原始时间序列之间的拟合误差更小，而且要小很多；对于斜率变化范围比较大的时间序列，SEEP 表示方法与原始时间序列之间的拟合误差，和以往的分段线性表示方法相比，也相差不大，并且 SEEP 表示方法计算简单，易于实现。算法的时间复杂度仅为 $O(n)$。

关键词　斜率，时间序列，分段线性表示，压缩率，拟合误差

Time Series Piecewise Linear Representation Based on Slope Extract Edge Point

ZHAN Yan-Yan　XU Rong-Cong　CHEN Xiao-Yun

(Department of Mathematics and Computer Science, Fuzhou University, Fuzhou 350002)

Abstract　In this paper, we introduce the slope which is widely used in analytic geometry, and bring forward a new Piecewise Linear Representation of Time Series which is based on Slope Extract Edge Point (SEEP). For the Time Series whose slope change range is relatively concentrate, SEEP representation has very good effects, compared with several other Piecewise Linear Representations, it's fitting error with originality Time Series can be much smaller; for other Time Series, whose slope change range is relatively large, it's fitting error with originality Time Series can not discrepant very much. This algorithm can be easy calculated and carried out, and it's time complexity is only $O(n)$.

Keywords　Slope, Time series, Piecewise linear representation(PLR), Compression ratio, Fitting error

1　引言

时间序列是指按照时间先后顺序排列的各个观测记录的有序集合，广泛存在于商业、经济、科学工程和社会科学等领域。随着时间的推移，时间序列通常包含大量的数据。如何对这些时间序列数据进行统计和分析，从中发现一些有价值的信息和知识，一直是用户感兴趣的问题。但是由于时间序列数据的海量和复杂的特点，直接在时间序列上进行数据挖掘，不但在储存和计算上要花费高昂代价，而且可能会影响算法的准确性和可靠性。

因此，许多研究者提出了时间序列的模式表示方法，其目的是刻画时间序列的主要形态而忽略那些微小的细节。时间序列的模式表示有三方面的好处，其一是对时间序列进行压缩，换来更小的存储和计算代价；其二是只保留时间序列的主要形态，去除了细节干扰，更能反映时间序列的自身特征，有利于提高数据挖掘的效率和准确性；其三是很多应用领域关心的是时间序列一段时间内的变化模式和规律，而不是时间序列中单个序列点的值，模式表示更符合这类领域的特点。

在时间序列的各种模式表示方法中，分段线性表示(Piecewise Linear Representation, PLR)方法相对而言更加简单直观，具有时间多解析的特点，而且多数 PLR 表示方法支持时间序列的动态增量更新。时间序列的 PLR 表示方法已经在下列一些领域得到应用[1]：

(a) 支持快速相似性搜索；

(b) 支持时间序列新的距离度量，包括模糊查找，加权序列，DTW 距离，信息反馈等等；

(c) 同时支持文本和数据序列；

(d) 支持新的聚类、分类算法；

(e) 支持奇异点检测。

2　相关定义[3,7]

定义 1(时间序列)　时间序列是由记录值和记录时间组成的元素的有序集合，记为 $X=\langle x_1=(v_1,t_1),x_2=(v_2,t_2),\cdots,x_n=(v_n,t_n)\rangle$，元素 $x_i=(v_i,t_i)$ 表示时间序列在 t_i 时刻的记录值为 v_i，记录时间 t_i 是严格增加的($i<j\Leftrightarrow t_i<t_j$)。

一般情况下，时间序列的采样间隔时间 $\Delta t=t_{i-1}-t_i$ 相等，可以看作 $t_1=0,\Delta t=1$，此时将时间序列 $X=\{x_1=(v_1,t_1),x_2=(v_2,t_2),\cdots,x_n=(v_n,t_n)\}$ 简记为 $X=\{x_1,x_2,\cdots,x_n\}$。X 的长度是它的势，记为 $|X|$。用 $x(i)$ 代表时间序列 $X=\langle x_1,x_2,\cdots,x_n\rangle$ 中的第 i 个元素 x_i，用 X^E 表示时间序列 $X=\langle x_1,x_2,\cdots,x_n\rangle$ 的 SEEP 表示。对于广义时间序列，记录值 v_i 可以是多种类型，包括离散符号、结构数据、多媒体数据等等。本文只考虑狭义时间序列，即 v_i 为实数值的情形。

时间序列的模式是指时间序列的某种变化特征，它可以是时间序列在一段时间的均值或者方差，也可以是时间序列离散化后的符号，甚至是时间序列的傅立叶变换系数。通过提取时间序列的模式，将时间序列变换到模式空间，就得到了时间序列的模式表示。以下给出时间序列模式表示的正式定义：

定义 2(时间序列的模式表示)　设有时间序列 $X=\langle x_1,$

*)本文获国家自然科学基金(60573076)、省自然科学基金(Z051503)和校科技发展基金(2004-XQ-17)资助。**詹艳艳**　硕士生，主要研究领域为机器学习、数据挖掘；**徐荣聪**　博士，副教授，硕士生导师，主要研究领域为计算机数学等；**陈晓云**　博士，副教授，硕士生导师，主要研究领域为数据挖掘、机器学习、模式识别。

$x_2,\cdots,x_n\rangle$，它可以用模式表示如下：

$$X(t)=f(w)+e(t) \tag{2.1}$$

其中，w是时间序列的模式，$f(w)$是时间序列的模式表示。$e(t)$是时间序列与它的模式表示之间的误差。

时间序列的模式可以是时间序列的全局特征，也可以是与时间相关的局部特征。这里我们着重讨论一种特殊的模式表示。将时间序列沿时间轴分成多个子段，将每个子段的两个端点坐标组合在一起定义为时间序列的模式，模式函数定义为连接两个端点的直线段，那么就得到了时间序列的一种模式表示方法，称为时间序列的分段线性表示。具体定义如下：

定义 3(时间序列的分段线性表示) 设有时间序列 $X=\langle x_1,x_2,\cdots,x_n\rangle$，它的分段模式表示为：

$$X(t)=\begin{cases} f_1(t,w_1)+e_1(t), t\in[1,t_1] \\ f_2(t,w_2)+e_2(t), t\in[t_1,t_2] \\ \cdots\cdots \\ f_k(t,w_k)+e_k(t), t\in[t_{k-1},n] \end{cases} \tag{2.2}$$

其中，w_i 表示时间区间$[w_{i-1},w_i]$的两个端点的坐标，$f_i(t,w_i)$表示连接模式 w_i 两端点的线性函数，$e_k(t)$是一段时间内时间序列与它的模式表示之间的误差。

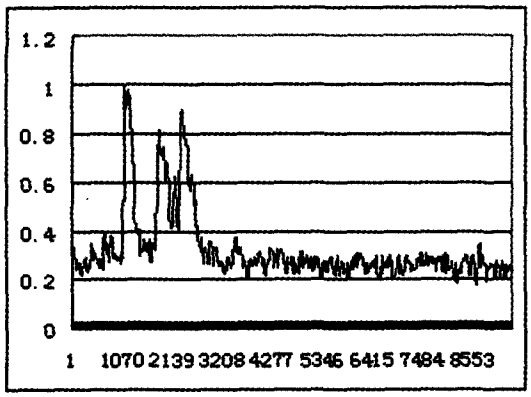


图 1(a)原始时间序列，长度为 9382

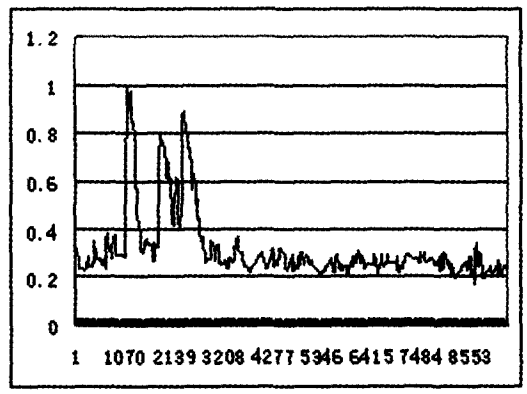


图 1(b)时间序列的分段线性表示，线段数目为 257，即处理后的时间序列长度为 258

图 1 时间序列及其模式表示

时间序列的模式表示能够压缩数据，但是保留了时间序列的主要形态，忽略了其中的一些微观细节。如图 1 所示，图 1(a)是一条具有 9382 个采样点的原始时间序列；图 1(b)是该时间序列的一个分段线性表示，只需保存 258 个线段端点，压缩率达到 97.3%，但是仍然很好地保留了原时间序列的主要形态，与原始序列的拟合误差仅为 1.63。

定义 4(时间序列分段线性表示的拟合误差) 设时间序列 $X=\langle x_1,x_2,\cdots,x_n\rangle$，通过线性分段算法得到时间序列的 PLR 表示为 $L(X)=\langle L(x_{i_1},x_{i_2}),L(x_{i_2},x_{i_3},\cdots,L(x_{i_{k-1}},x_{i_k}))$，其中 $L(\cdot,\cdot)$ 表示连接两点的直线段。将 $L(X)$ 经过线性插值后得到的时间序列记为 $X^c=\langle x_1^c,x_2^c,\cdots,x_n^c\rangle$，那么该 PLR 表示与原始时间序列之间的拟合误差定义为：

$$E=\sqrt{\sum_{i=1}^{n}(x_i-x_i^c)^2} \tag{2.3}$$

定义 5(时间序列分段线性表示的压缩率) 设时间序列 $X=\langle x_1,x_2,\cdots,x_n\rangle$，通过线性分段算法得到的时间序列为 $X=\langle x'_1,x'_2,\cdots,x'_n\rangle$，其中 $x'_1=x_1,x'_n=x_n$。那么该 PLR 算法的压缩率为$(1-\frac{n'}{n})\times 100\%$。

3 基于斜率提取边缘点的时间序列分段线性表示方法

在本文中，我们提出了一种基于斜率提取边缘点的时间序列分段线性表示方法。通过借鉴解析几何中两点确定一直线时该直线性状的两个描述值：斜率和截距，把斜率和时间序列的特点结合起来，提出了一种基于斜率提取边缘点(即分段点)的方法，该方法首先计算连接同一点(除线段的两个端点外)的左右两条线段的斜率，然后通过斜率的变化幅度来确定该点是否是边缘点，最后将这些边缘点依次用线段相连接，就得到了时间序列的一种分段线性表示，简称为时间序列的 SEEP 表示(Time Series Piecewise Linear Representation based on Slope Extract Edge Point)。

3.1 时间序列的 SEEP 表示

时间序列的 SEEP 表示是一种分段线性表示方法，根据斜率的变化率从时间序列上检测时间序列的边缘点(即分段点)，用连接这些边缘点之间的直线段序列来表示原始时间序列。下面给出时间序列 SEEP 表示的具体定义：

设时间序列 $X=\langle x_1,x_2,\cdots,x_n\rangle$，其边缘点集合为，$\{x_{i_1},x_{i_2},\cdots,x_{i_k}\}$其中 $1\leqslant i_1<i_2<\cdots<i_k\leqslant n$。那么该时间序列的 SEEP 表示为

$$X^E=\begin{cases} L(x_{i_1},x_{i_2}), t\in[i_1,i_2] \\ L(x_{i_2},x_{i_3}), t\in[i_2,i_3] \\ \cdots\cdots \\ L(x_{i_{k-1}},x_{i_k}), t\in[i_{k-1},i_k] \end{cases} \tag{3.1}$$

其中 $L(x,y)$ 表示连接边缘点 x 和 y 之间的线性函数。在不引起混淆的情况下，公式(3.1)也可以简单表示为

$$X^E=\langle L(x_{i_1},x_{i_2}),L(x_{i_2},x_{i_3}),,\cdots,L(x_{i_{k-1}},x_{i_k})\rangle \tag{3.2}$$

3.2 边缘点的确定

为了得到时间序列的 SEEP 表示，最主要的问题是如何寻找时间序列的边缘点？下面首先给出边缘点的定义：

将时间序列 $X=\langle x_1,x_2,\cdots,x_n\rangle$看作按时间顺序排列的 n 个序列点的集合。如果时间序列中的某个点是边缘点，那么在局部范围内，位于该点两端的时间序列将呈现不同的变化趋势。如图 2 所示的时间序列中，时间序列在序列点 A 左边呈现上升趋势，右边则呈现下降趋势，因此认为 A 是一个边缘点；而序列点 B 左边呈现下降趋势，右边则呈现平坦趋势，也被认为是一个边缘点。

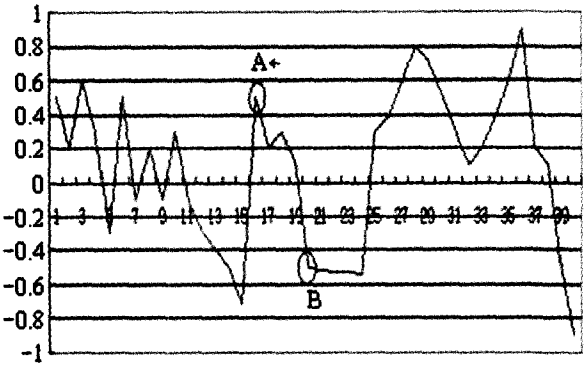


图 2　时间序列的边缘点

那么，如何确定边缘点呢？简单地说，就是要计算时间序列中的某一点 $x(i)$ 与其左边的邻点 $x(i-1)$ 确定的线段的斜率 tg1，以及 $x(i)$ 与其右边的邻点 $x(i+1)$ 确定的线段的斜率 tg2，当斜率的变化率 $|tg2-tg1|$ 大于等于参数 d 时（d 为本算法中要求输入的一个参数，表示斜率的变化率所应满足的最小值），此点即为边缘点，将其加入处理后的时间序列中，否则，此点不是边缘点，计算 $x(i+1)$ 与其左邻点 $x(i-1)$ 确定的线段的斜率 tg1，以及 $x(i+1)$ 与其右邻点 $x(i+2)$ 确定的线段的斜率 tg2，依次类推。

这里有一个难点，就是当 $x(i)$ 的左右两个邻点位于 $x(i)$ 的同侧或者是异侧的时候，边缘点的分析方法略有不同（这里所说的同侧和异侧，是指做一条经过 $x(i)$，并且平行于 x 轴的水平轴，$x(i)$ 的左右两个邻点位于该水平轴的同侧或者异侧）。

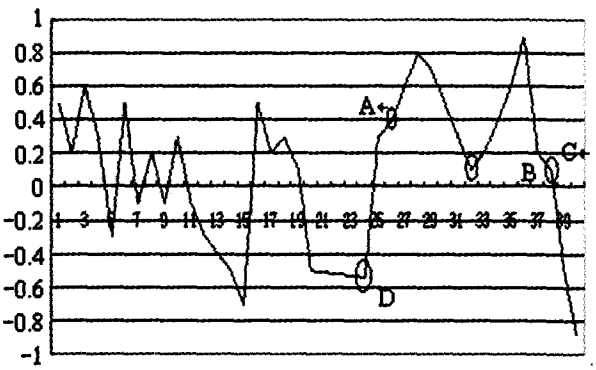


图 3　时间序列边缘点的确定

如图 3 所示，点 A 与其临近的左右两个邻点所确定的线段，虽然本身斜率都比较大，但它们之间的斜率变化率却很小。所以，一般情况下，此点就不是边缘点。点 C 与其临近的左右两个邻点所确定的线段的斜率变化率显然比较大。所以，一般情况下，此点是边缘点。以上这两种都是当 $x(i)$ 的左右两个邻点位于 $x(i)$ 的异侧的情况。当 $x(i)$ 的左右两个邻点位于 $x(i)$ 的同侧的时候，如点 B 与其临近的左右两个邻点所确定的线段，虽然它们斜率的绝对值变化率不大，但这两条线段本身的斜率比较大。所以，显然，一般情况下，此点是边缘点。点 D 的左右两个邻点，虽然位于其同侧，但显然，点 D 与其左边邻点所确定的线段斜率趋向于 0，而点 D 与其右边邻点所确定的线段斜率比较大，所以一般情况下，点 D 是边缘点。因此，当 $x(i)$ 的左右两个邻点位于 $x(i)$ 的同侧时，只要 $x(i)$ 与其左右两个邻点所确定的两条线段中有一条的斜率大于 d，此点即为边缘点。

这里所说的一般情况，是因为 d 是一个人为输入的参数。如果为 d 赋以一些极端的值，则上面所述的点 A 也有可能是边缘点，B、C、D 也有可能不是边缘点。下一节将给出基于斜率提取边缘点的时间序列分段线性表示的具体算法 SEEP。

3.3　时间序列的 SEEP 表示算法

算法名称：基于斜率提取边缘点的时间序列分段线性表示算法 SEEP

算法输入：时间序列 $X=\langle x_1,x_2,\cdots,x_n\rangle$，参数 n,d（其中 n 为原始时间序列的长度，d 为斜率的变化率）

算法输出：时间序列的 SEEP 表示

具体步骤：

1) $ii=0;jj=1;kk=2;$

2) $xx=\{(x_1,1)\}$; //第一个序列点是边缘点

3) for($i=1;i<n-1;i++$)

4) if($(x[kk]-x[jj])*(x[jj]-x[ii])<0$) //$x[kk]$ 和 $x[ii]$ 位于 $x[jj]$ 同侧

if(fabs($(x[jj]-x[ii])/((jj-ii))$)$>=d$)

then $xx=xx+\{(x_i,i)\}$;//序列点 x_i 加入边缘点集合 xx

$ii=i;jj=i+1;kk=i+2;$

else $jj=i+1;kk=i+2;$

5) else //$x[kk]$ 和 $x[ii]$ 位于 $x[jj]$ 异侧

if(fabs($(x[kk]-x[jj])/(kk-jj)-(x[jj]-x[ii])/(jj-ii)$)$>=d$)

then $xx=xx+\{(x_i,i)\}$;//序列点 x_i 加入边缘点集合 xx

$ii=i;jj=i+1;kk=i+2;$

else $jj=i+1;kk=i+2;$

6) $xx=xx+\{(x_n,n)\}$; //最后一个序列点是边缘点

7) output $L(X)=\{L(x_{i_1},x_{i_2}),L(x_{i_2},x_{i_3}),\cdots,L(x_{i_{k-1}},x_{i_k})|(x_{i_m},m)\in xx\}$;

3.4　算法分析

SEEP 算法只需扫描序列一次，算法的时间复杂度仅为 $O(n)$，其中 n 是时间序列的长度。SEEP 算法的一个优点是可以直接用于时间序列的在线分段。当时间序列动态增长时，以前的边缘点结果集依然可以保留，只需对新的时间序列数据应用 SEEP 算法即可。

4　实验及结果分析

4.1　实验数据

在实验中，我们采用了 Keogh 等人[2]提供的来自不同领域的 8 条时间序列的实际数据集（本文简称为 KData）来比较算法的性能，具有较好的广泛性和代表性。各序列如表 1 所示。

表 1　KData 数据集描述

序列名称	序列长度	序列名称	序列长度
Burst	9382	Ocean	4096
Chaotic	1800	Powerplant	2400
Leleccum	4320	Speech	1021
Earthquake	4097	Tide	8746

4.2　实验方法

本文选择以下四种主要的时间序列分段线性表示作为比

较对象。

①基于重要点的 PLR 表示[3]：Pratt 和 Fink 提出了基于重要点的分段方法，重要点被定义为在局部范围内的极值点，并且与端点的比值超过参数 R。将重要点用线段连接，就得到了基于重要点的时间序列分段线性表示，这里称为 PLR_PF 表示。通过选择不同的参数 R，可以获得精细粒度不同的分段线性表示。PLR_PF 表示的输入参数是 R。

②基于 PAA 的分段线性表示算法[4,5]：Keogh 和 Yi 等人分别独立提出了时间序列的 PAA 表示方法，用等宽度窗口分割时间序列，每个窗口内的时间序列用窗口平均值来表示，就得到了时间序列的一种分段线性表示，这里称为 PLR_KY 表示，它的输入参数为窗口大小，记为 w。

③基于特征点的 PLR 表示[6]：Xiao 等人提取时间序列的特征点作为时间序列的分段点，通过连接这些特征点，得到时间序列的分段线性表示，这里称为 PLR_XFH 表示，它的输入参数为 v，表示特征区间大小。

④基于时态边缘算子的 TEO 表示[7]：肖辉等人提出使用时态边缘算子提取时间序列的分段点，然后通过连接这些边缘点，得到时间序列的分段线性表示，这里称为 TEO 表示，它的输入参数有两个，参数 u 表示时态边缘算子的检测窗口大小，参数 d 表示最小子段长度。

本文提出的时间序列的 SEEP 表示方法需要两个输入参数，参数 n 表示输入的原始时间序列的长度，参数 d 表示斜率需要满足的最小变化率。由于数据集中各时间序列来自不同领域，序列值相差很大，为便于对比，在采用线性分段算法之前首先对时间序列做规范化处理，将序列值规范化到[0，1]之间。规范化公式如下：

$$norm(x_i)=\frac{x_i-\min(X)}{\max(X)-\min(X)} \qquad (4.1)$$

对于算法性能的评价指标，我们主要考察四种 PLR 表示方法与原时间序列之间的拟合误差。由于五种时间序列的分段线性表示方法的输入参数各不相同，为了公平起见，因此我们比较在选取的分段点数量相同的存储空间即压缩率相同的情况下，分别采用五种 PLR 表示方法与原始时间序列之间的拟合误差。采用公式(2.3)来计算拟合误差。拟合误差越小，表示算法性能越好。

4.3 实验结果及分析

(1)五种线性分段算法在数据集 KData 上的性能比较：

由于 PAA 分段算法的自身特点，它的压缩率只能为(1−1/w)＊100%，其中参数 w 是正整数。这里我们取 $w=5$，即压缩率为 80%。实验结果如表 2 所示。

表 2 压缩率为 80%时几种 PLR 表示的拟合误差

算法 / 数据集	PLR_PF 表示	PLR_XFH 表示	PLR_KY 表示	TEO 表示	SEEP 表示
Burst	0.89	0.9	0.32	0.16	**0.07**
Chaotic	1.63	1.63	1.76	1.22	**0.49**
Earthquake	**2.10**	2.22	3.48	2.17	3.50
Leleccum	0.88	0.98	**0.50**	0.54	0.91
Ocean	0.38	0.31	0.31	**0.19**	0.51
Powerplant	2.23	1.11	1.07	1.03	**0.59**
Speech	1.52	3.22	2.39	**1.42**	1.72
Tide	**2.29**	2.48	3.22	2.37	3.35

在表 2 中，粗体数据表示拟合误差最小。从表 2 中可以看出，在 8 条时间序列中，基于重要点的 PLR_PF 表示在其中的 2 条时间序列上拟合误差最小；基于 PAA 的 PLR_KY 表示在其中的 1 条时间序列上拟合误差最小；基于边缘算子的 TEO 表示在其中的 2 条时间序列上拟合误差最小，我们提出的 SEEP 表示在另外 3 条时间序列上拟合误差最小，并且远远小于其他几种算法的拟合误差。

实验结果表明：对于斜率变化范围比较集中时间序列(如图 4(a)所示)，SEEP 表示方法有着非常好的效果，与以往的分段线性表示方法相比，SEEP 表示方法与原始时间序列之间的拟合误差更小，而且要小很多；对于斜率变化范围比较大的时间序列(如图 4(b)所示)，SEEP 表示方法与原始时间序列之间的拟合误差，和以往的分段线性表示方法相比，也相差不大。

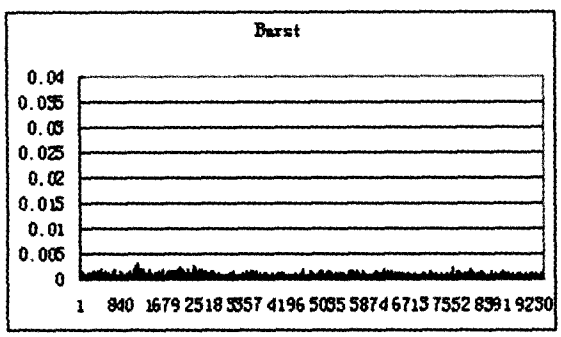


图 4(a)斜率变化范围比较集中

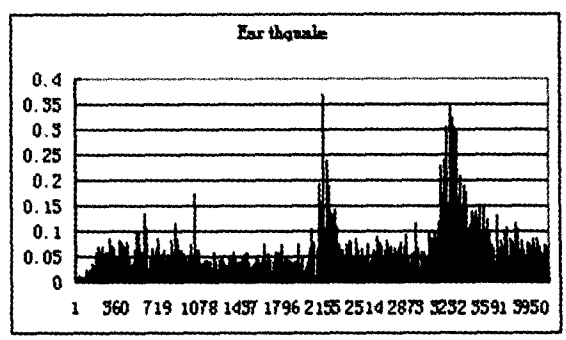


图 4(b)斜率变化范围比较大

图 4 斜率变化范围

(2)比较在上一个实验中结果比较好的三条时间序列在不同压缩率下的拟合误差和斜率变化率 d 的变化情况，实验结果如图 5 所示。

从图 5 中可以看出压缩率与拟合误差及斜率变化率之间的关系，图 5(a)说明拟合误差随着压缩率增加而增加，但在压缩率小于 75%的时候拟合误差的增长极其缓慢，当压缩率大于 75%以后，拟合误差才出现显著增长。图 5(b)则表明 SEEP 算法的压缩率随着斜率变化率的增大而增加。

结论 由于时间序列的海量和复杂的数据特点，直接在时间序列上进行数据挖掘不但在储存和计算上要花费高昂代价，而且可能会影响算法的准确性和可靠性。

本文借鉴了解析几何中斜率的基本思想，将线段的斜率与时间序列的特点结合起来，根据斜率的变化幅度选取一些时间序列的边缘点，将这些边缘点依次用线段连接，就得到了时间序列的一种分段线性表示，称为时间序列的 SEEP 表示。

(下转第 161 页)

多冗余的工作。例如规则(2),知识粒子{3,8},即$a \leqslant bad$,满足原则4.2,因此我们不将推导其前件条件强于$a \leqslant bad$、后件形如$Cl^{\leqslant}$的规则,但是我们可以继续推导后件形如$Cl^{\geqslant}$的规则。

通过上面的规则推导,我们很容易看到规则集合覆盖了除对象4和7以外的所有对象,并且在某种程度上很简化。当然,也可能存在某些对象被几条规则覆盖。例如,对象9被规则(3)和规则(5)覆盖,但它们的含义是不同的。而且标准a与决策属性d语义相关,规则也准确地反映了这一点。至于如何进一步简化这些规则,从而基于要求提炼出更加精炼和完备的规则集,本文不做深入研究。

除此而外,根据计算近似质量的公式,让我们来计算上述例子的$\lambda_P(Cl)$,此处$P=\{a\}$。我们来比较前面章节中提到的公式哪个更合理。

表3 Cl_t的下近似和上近似

Cl_t	$\underline{P}(Cl)_t$	$\overline{P}(Cl)_t$	$Bnp(Cl_t)$
$Cl_1^{\leqslant}=\{1,3,8\}$	{3,8}	{1,3,6,8,9}	{1,6,9}
$Cl_2^{\leqslant}=\{1,3,6,8,9\}$	{1,3,6,8,9}	{1,3,6,8,9}	Φ
$Cl_2^{\geqslant}=\{2,5,6,9\}$	{2,5}	{1,2,5,6,9}	{1,6,9}
$Cl_3^{\geqslant}=\{2,5\}$	{2,5}	{2,5}	Φ

根据公式(1),我们有

$$\lambda_P(Cl)=\frac{|(U-(\bigcup_{t\in T}Bn_P(Cl_t^{\leqslant})))|}{|U|}=\frac{|\{1,2,3,5,6,8,9\}-\{1,6,9\}|}{9}=\frac{4}{9}$$

根据公式(2),我们有

$$\lambda_P(Cl)=\frac{|\bigcup_{t=1}^{n-1}(\underline{P}(Cl_t^{\leqslant}))\cup\bigcup_{t=2}^{n}(\underline{P}(Cl_t^{\geqslant}))|}{|U|}=\frac{|\{1,3,6,8,9\}\cup\{2,5\}|}{9}=\frac{7}{9}$$

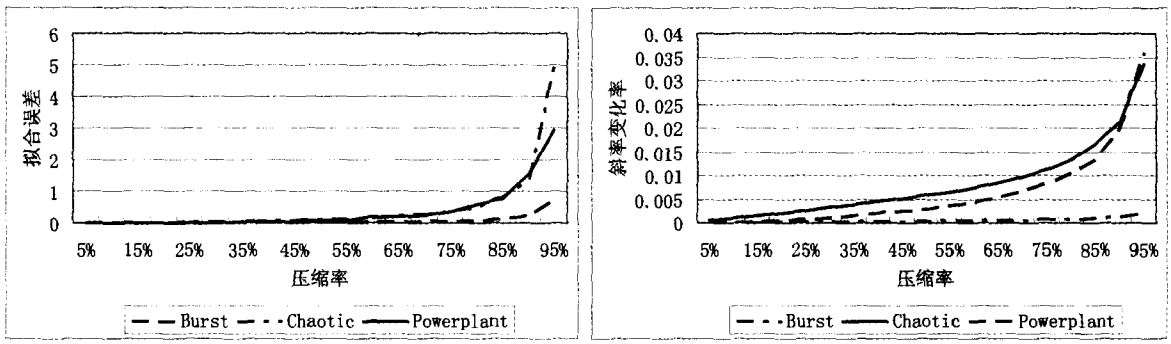
通过表3,我们能看到公式(1)夸大了可疑对象的数目。显然,$Bn_{\{a\}}(Cl_1^{\leqslant})=\{1,6,9\}$,但是$\{1,6,9\}\subseteq\underline{P}(Cl_2^{\leqslant})$,因此这些对象被确定规则所捕捉,而公式(2)保证对象集{1,6,9}不再是可疑对象。

结论 本文基于经典粗集理论,研究探讨了含序信息粗集方法。本文定义了标准、向上合并、向下合并、有序决策表等概念,并通过标准和属性共同刻画决策类的有序合并。本文提出了根据得到的知识粒子形成规则的四条原则,最后通过例子阐述上述概念和思想。

本文只是初步研究了含序粗集方法,该方法的许多方面和细节还在进一步研究和讨论中,未来需要做的工作还有许多,如:

(下转第163页)

(上接第142页)



(a) 在不同压缩率下的拟合误差变化情况　　(b) 在不同压缩率下的斜率变化率变化情况

图5 在不同压缩率下的拟合误差和斜率变化率的变化情况

时间序列的SEEP表示简单直观,具有很强的数据压缩能力和一定的除噪能力,能突出时间序列的模式变化特征。

在来自不同领域的时间序列数据集上的实验表明:对于斜率变化范围比较集中时间序列,SEEP表示方法有着非常好的效果,与以往的分段线性表示方法相比,SEEP表示方法与原始时间序列之间的拟合误差更小,而且要小很多,并且当压缩率小于75%的时候拟合误差的增长极其缓慢;对于斜率变化范围比较大的时间序列,SEEP表示方法与原始时间序列之间的拟合误差,和以往的分段线性表示方法相比,也相差不大。而且SEEP表示方法计算简单,易于实现,并且由于算法的时间复杂度仅为$O(n)$,因此即使在数据量比较大的时间序列中仍然具有很高的执行效率。

致谢 感谢Keogh等人提供的来自于不同领域的实验数据集,使本文可以顺利完成。

参考文献

1 Keogh E. Fast similarity search in the presence of longitudinal scaling in time series databases [C]. In: Proceedings of the IEEE 9th International Conference on Tools with Artificial Intelligence, Washington: IEEE Computer Society, 1997. 578～584

2 Keogh E, Folias T. The UCR Time Series Data Mining Archive [EB/OL]. http://www.cs.ucr.edu/～eamonn/TSDMA/index.html. Irvine, CA: University of California, Department of Information and Computer Science, 2002

3 Prat K B, Fink E. Search for patterns in compressed time series [J]. International Journal of Image and Graphics, 2002, 2(1): 89～106

4 Keogh E J, Chakrabarti K, Pazzani M J. Sharad Mehrotra. Dimensionality Reduction for Fast Similarity Search in Large Time Series Databases [J]. Knowl. Inf. Syst, 2001, 3(3): 263～286

5 Yi B K, Faloustsos C. Fast Time Sequence Indexing for Arbitrary Lp Norms [C]. In: Proceedings of the 26th International Conference on Very Large Data Bases, San Francisco: Morgan Kaufmann Publishers Inc, 2000. 385～394

6 Xiao Hui, Feng Xiao-Fei, Hu Yun-Fu. A new segmented time warping distance for data mining in time series database [C]. In: Proceedings of 2004 International Conference on Machine Learning and Cybernetics, Shanghai, China, 2004. 1277～1281

7 肖辉. 时间序列的相似性查询与异常检测: [博士论文]. 上海: 复旦大学, 2005