ISSN 1000-9825, CODEN RUXUEW
Journal of Software, Vol.19, No.9, September 2008, pp.2349−2361
DOI: 10.3724/SP.J.1001.2008.02349

E-mail: jos@iscas.ac.cn
http://www.jos.org.cn
Tel/Fax: +86-10-62562563

用基于移动均值的索引实现时间序列相似查询*

林子雨[1,2+], 杨冬青[1,2], 王腾蛟[1,2]

[1](北京大学 高可信软件技术教育部重点实验室,北京 100871)

[2](北京大学 信息科学技术学院,北京 100871)

Similarity Search of Time Series with Moving Average Based Indexing

LIN Zi-Yu[1,2+], YANG Dong-Qing[1,2], WANG Teng-Jiao[1,2]

[1](Key Laboratory of High Confidence Software Technologies for the Ministry of Education, Peking University, Beijing 100871, China)

[2](School of Electronics Engineering and Computer Science, Peking University, Beijing 100871, China)

+ Corresponding author: E-mail: cainiu@263.net

Lin ZY, Yang DQ, Wang TJ. Similarity search of time series with moving average based indexing. *Journal of Software*, 2008,19(9):2349−2361. http://www.jos.org.cn/1000-9825/19/2349.htm

Abstract: In this paper, a method called MABI (moving average based indexing) is proposed to effectively deal with the issue of ε-search query in subsequence matching. Two important theorems, distance reduction theorem and DRR(distance reduction rate) relation theorem, are proposed here to be as the basis of MABI. DRR relation theorem has strong capability in "pruning" those unqualified candidate sequences so as to achieve of fast similarity search. Furthermore, by modifying BATON* introduced by Jagadish, *et al*., a multi-way balanced tree structure is introduced, to construct the index from time series, which significantly speeds up the similarity search. Extensive experiments over a stock exchange dataset show that MABI can achieve desirable performance.

Key words: similarity search; subsequence matching; moving average; time series database

摘 要: 提出了基于移动均值的索引来解决子序列匹配中的"ε-查询"问题;提出并证明了基于移动均值的缩距定理和缩距比关系定理,后者具有很好的"裁减"能力,可以在相似查询时淘汰大部分不符合条件的候选时间序列,从而达到快速相似查找的目的;引入了由 Jagadish 等人提出的 BATON*-树,并在此基础上适当修改,建立了 MABI 索引,极大地加快了相似查询过程;最后,在一个股票交易数据集上进行了实验,证明了 MABI 索引的良好性能.

关键词: 相似查询;子序列匹配;移动均值;时间序列数据库

中图法分类号: TP311 **文献标识码**: A

相似查询是时间序列研究领域的热点问题,其任务是从时间序列数据库中找到那些与给定的查询序列具有相似变化模式的序列或子序列.相似查询包括两种类型:全序列匹配和子序列匹配.顾名思义,对于一个查询序列 ***Q***,完全匹配方法要求在数据库中找到一个时间序列 ***S***,使得 ***S*** 和 ***Q*** 的长度相等并且满足所采用的相似判

* Supported by the National Natural Science Foundation of China under Grant No.60473051(国家自然科学基金); The National High-Tech Research and Development Plan of China under Grand Nos.2007AA01Z191, 2006AA01Z230 (国家高技术研究与发展计划(863))

Received 2007-04-14; Accepted 2007-12-24

定标准,而子序列匹配则只需要 $\boldsymbol{S}$ 中的一个子序列与 $\boldsymbol{Q}$ 匹配即可.

时间序列数据库非常庞大,不仅要记录历史数据,而且每时每刻都有新的数据产生并添加到数据库中.比如 MACHCO 计划,这个天文学数据库包含 0.5TB 以上的数据,并且每天都往数据库中输入新的数据[1].从如此庞大的数据库中搜寻满足特定要求的数据(常常是一个时间序列片段),就必须有功能强大且行之有效的数据压缩、索引和检索机制来加快查询速度.在这些研究中,较为成熟的是时间序列的压缩技术.这里的压缩技术并不是通常意义上的、为了减少数据所占空间而使用的数据压缩技术(即对数据重新进行二进制编码,比如文件压缩软件 WINZIP 就采用这种方式压缩数据).针对时间序列的数据压缩,必须能够保证在经过数据压缩处理后仍然可以对处理后的数据进行索引和查询.目前比较常用的时间序列数据压缩方法包括离散傅里叶变换(discrete Fourier transform,简称 DFT)[2]、符号映射(symbolic mappings)[3]、奇异值分解(singular value decomposition,简称 SVD)、离散小波变换(discrete wavelet transform,简称 DWT)[4]和分段线性表示 PLR(piecewise linear representation)[5−8]等等,它们都是通过降低维数(把一个长度为 n 的时间序列映射到一个 N 维空间,其中 $N<n$)来达到数据压缩的目的;然后,利用 R*-树对 N 维空间中的目标向量建立索引,就可以实现时间序列的快速查询.但是,在高维空间中,类似 R*-树这样的层次索引结构通常无法取得令人满意的性能.在某些情况下,即使采用了降维(dimensionality reduction)技术,相似查询工作仍然需要访问大量索引.

本文提出了采用基于移动均值的索引来有效解决时间序列子序列匹配的“ε-查询”问题[9].在ε-查询中,两个相似的时间序列之间的距离需要满足 $\boldsymbol{D}(S_1,S_2)<\varepsilon$,并且需要用户明确给出$\varepsilon$的值.本文提出并证明了与移动均值相关的两个重要定理,即缩距定理和缩距比关系定理.缩距比关系定理具有很强的裁剪能力,可以淘汰大部分不符合条件的候选时间序列,缩小相似查询的搜索范围.同时,为了加快搜索速度,我们还引入了由 Jagadish 等人于 2006 年提出的 BATON*-树索引结构,经过适当修改以后,构建了基于移动均值的索引—— MABI(moving average based indexing)索引.与类似 R*-树的层次结构索引不同的是,MABI 索引结构只对“缩距比”(见下面第 1 节定义)进行索引,在很大程度上降低了搜索的代价.我们在真实的股票数据上进行了大量的实验.实验结果表明,缩距比关系定理具有出色的淘汰不符合条件的候选时间序列的能力;同时,实验也表明 MABI 索引具有良好的性能.

本文第 1 节介绍相关的定义,并给出两个重要的定理及其证明,即缩距定理和缩距比关系定理.第 2 节详细介绍 MABI 索引结构及其构建方法,并给出如何在索引树中进行查询的方法.第 3 节给出性能分析与讨论.实验设计和结果将在第 4 节进行阐述.第 5 节介绍相关工作.最后给出结论.

1 定义和定理

定义 1(移动均值[10]).给定一个长度为 n 的时间序列 $\boldsymbol{X}=[x_i]$(其中 $i=0,1,\ldots,n-1$)和移动均值系数 k,序列 $\boldsymbol{X}$ 的 k 阶移动均值 $MV_k(\boldsymbol{X})=(x_k[j])(0\le j<\mathrm{Len}(\boldsymbol{X})-k+1)$是通过如下变换实现的:

$$x_k[j]=\frac{x[j]+x[j+1]+\ldots+x[j+k-1]}{k}=\frac{\sum_{l=j}^{j+k-1}x[l]}{k}.$$

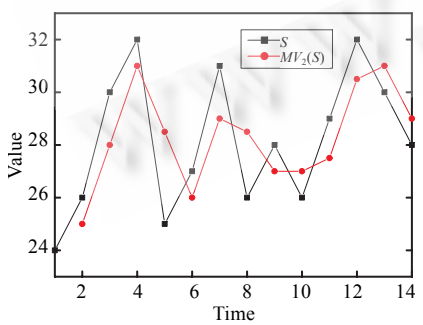


Fig.1 A sequence S and its 2-moving average $MV_2(S)$

图 1 序列 S 及其的 2 阶移动均值 $MV_2(S)$

移动均值(moving average)在股票分析中有着广泛的应用,它最初是用来消除股票记录中的短期波动,从而揭示股票的行情走向,后来被应用到时间序列分析中,用来消除时间序列中的噪音,减少序列中突然出现的高峰和低谷,从而使序列变得更加“平滑”.

例如,有一个时间序列 $\boldsymbol{S}$={24,26,30,32,25,27,31,26,28,26,29,32,30,28},滑动窗口的大小 k=2,根据定义 1 就可以计算得到 $\boldsymbol{S}$ 的 2 阶移动均值 $MV_2(\boldsymbol{S})$={25,28,31,28.5,26,29,28.5,27,27,27.5,30.5,31,29}.图 1 显示了时间序列 $\boldsymbol{S}$ 及其 2 阶移动均值 $MV_2(\boldsymbol{S})$.

定义 2(两个时间序列相似). 对于两个时间序列,$\boldsymbol{X}=\{x_0,x_1,\ldots,x_{n-1}\}$,$\boldsymbol{Y}=\{y_0,y_1,\ldots,y_{n-1}\}$,如果两者之间的距离满足 $\boldsymbol{D}(\boldsymbol{X},\boldsymbol{Y})<\varepsilon$(其中$\varepsilon$是人为设定的一个门槛值),则 $\boldsymbol{X}$ 和 $\boldsymbol{Y}$ 是两个相似的时间序列.

在实际应用中,通常使用欧氏距离(Euclidean distance)来衡量两个时间序列之间的距离.对于两个时间序列,$\boldsymbol{X}=\{x_0,x_1,\ldots,x_{n-1}\}$和 $\boldsymbol{Y}=\{y_0,y_1,\ldots,y_{n-1}\}$,两者之间的欧氏距离 $\boldsymbol{D}(\boldsymbol{X},\boldsymbol{Y})$的计算方法如下:

$$\boldsymbol{D}(\boldsymbol{X},\boldsymbol{Y})=\sqrt{\sum_{i=0}^{n-1}(x_i-y_i)^2}\ .$$

定理 1(缩距定理).对于两个时间序列 $\boldsymbol{X}=\{x_0,x_1,\ldots,x_{n-1}\}$和 $\boldsymbol{Y}=\{y_0,y_1,\ldots,y_{n-1}\}$,它们的 2 阶移动均值分别为 $MV_2(\boldsymbol{X})$和 $MV_2(\boldsymbol{Y})$,则一定满足如下公式:

$$\boldsymbol{D}(MV_2(\boldsymbol{X}),MV_2(\boldsymbol{Y}))\leq\boldsymbol{D}(\boldsymbol{X},\boldsymbol{Y}).$$

上面公式中,当且仅当 $\boldsymbol{X}$ 和 $\boldsymbol{Y}$ 是相同的等值序列时,$\boldsymbol{D}(MV_2(\boldsymbol{X}),MV_2(\boldsymbol{Y}))=\boldsymbol{D}(\boldsymbol{X},\boldsymbol{Y})$关系成立;除此以外,都有 $\boldsymbol{D}(MV_2(\boldsymbol{X}),MV_2(\boldsymbol{Y}))<\boldsymbol{D}(\boldsymbol{X},\boldsymbol{Y})$成立.为了使本文的结构更加流畅,我们把缩距定理的详细证明过程放在附录中.

推论 1. 对于两个相似的时间序列,它们的 2 阶移动均值也一定相似.

证明:设有两个时间序列 $\boldsymbol{X}=\{x_0,x_1,\ldots,x_{n-1}\}$和 $\boldsymbol{Y}=\{y_0,y_1,\ldots,y_{n-1}\}$,它们的 2 阶移动均值分别为 $MV_2(\boldsymbol{X})$和 $MV_2(\boldsymbol{Y})$.根据定义 2,$\boldsymbol{X}$ 和 $\boldsymbol{Y}$ 相似,则一定满足 $\boldsymbol{D}(\boldsymbol{X},\boldsymbol{Y})<\varepsilon$;又根据定理 1,$\boldsymbol{D}(MV_2(\boldsymbol{X}),MV_2(\boldsymbol{Y}))\leq\boldsymbol{D}(\boldsymbol{X},\boldsymbol{Y})$,有 $\boldsymbol{D}(MV_2(\boldsymbol{X}),MV_2(\boldsymbol{Y}))<\varepsilon$成立;所以,根据定义 2,$MV_2(\boldsymbol{X})$和 $MV_2(\boldsymbol{Y})$也相似. □

定义 3(缩距比). 设有一个时间序列 $\boldsymbol{S}$ 和它的 2 阶移动均值 $MV_2(\boldsymbol{S})$,它们与水平坐标轴之间的欧氏距离分别为 Ds 和 D_M,则 $\boldsymbol{S}$ 和 $MV_2(\boldsymbol{S})$之间的缩距比被定义为 $r=(Ds-D_M)/Ds$.

图 2 给出了序列 $\boldsymbol{S}$ 和 Ds 的形象表示.现在假设有两个时间序列 $\boldsymbol{X}$ 和 $\boldsymbol{Y}$,它们与水平坐标轴之间的欧氏距离用 D_X 和 D_Y 来表示,它们的 2 阶移动均值 $MV_2(\boldsymbol{X})$和 $MV_2(\boldsymbol{Y})$与水平坐标轴之间的欧氏距离分别用 D_{M1} 和 D_{M2} 来表示,相应的,缩距比分别用 r_1 和 r_2 来表示,判定两个时间序列相似的门槛值用ε来表示,则有下面的定理成立:

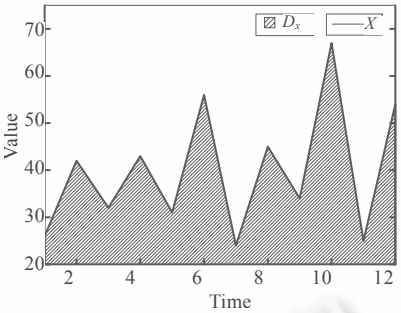


Fig.2 A sequence X and its D_X

图 2 序列 X 和 D_X

定理 2(缩距比关系定理). 对于两个相似的时间序列 $\boldsymbol{X}$ 和 $\boldsymbol{Y}$,当 $D_X>\varepsilon$时,则必满足以下公式:

$$\frac{D_X\times r_1-2\varepsilon}{D_X+\varepsilon}<r_2<\frac{D_X\times r_1+2\varepsilon}{D_X-\varepsilon}.$$

证明:因为 $\boldsymbol{X}$ 和 $\boldsymbol{Y}$ 是相似的时间序列,所以根据定义 2,必满足:$\boldsymbol{D}(\boldsymbol{X},\boldsymbol{Y})<\varepsilon$;根据三角不等定理[11],则有:$D_X-D_Y<\boldsymbol{D}(\boldsymbol{X},\boldsymbol{Y})$或 $D_Y-D_X<\boldsymbol{D}(\boldsymbol{X},\boldsymbol{Y})$,即$|D_X-D_Y|<\boldsymbol{D}(\boldsymbol{X},\boldsymbol{Y})$,所以有:$|D_X-D_Y|<\varepsilon$;根据推论 1,$MV_2(\boldsymbol{X})$和 $MV_2(\boldsymbol{Y})$也相似,则也满足:$|D_{M1}-D_{M2}|<\varepsilon$;则有:$|D_X-D_Y|+|D_{M1}-D_{M2}|<2\varepsilon$.再根据数学中关于两个数和的绝对值的相关知识,我们可以有:$|(D_X-D_Y)-(D_{M1}-D_{M2})|\leq|D_X-D_Y|+|D_{M1}-D_{M2}|$;由此,我们又可以有:$|(D_X-D_Y)-(D_{M1}-D_{M2})|<2\varepsilon$;则$|(D_X-D_{M1})-(D_Y-D_{M2})|<2\varepsilon$;用缩距比来表示就是$|D_X\times r_1-D_Y\times r_2|<2\varepsilon$;前面已经证得$|D_X-D_Y|<\varepsilon$;根据不等式的知识,我们最终可以推得:

$$\frac{D_X\times r_1-2\varepsilon}{D_X+\varepsilon}<r_2<\frac{D_X\times r_1+2\varepsilon}{D_X-\varepsilon}.$$ □

在实际应用中,对于一个时间序列 $\boldsymbol{X}$,$D_X>\varepsilon$几乎总是成立的,因为当时间序列 $\boldsymbol{X}$ 满足 $D_X\leq\varepsilon$ 时,寻找与 $\boldsymbol{X}$ 相似的时间序列通常没有实际意义.另外,为了表述得简洁和方便,在下文的论述中,我们将用 *range* 来表示缩距比区间:$\left[\frac{D_X\times r_1-2\varepsilon}{D_X+\varepsilon},\frac{D_X\times r_1+2\varepsilon}{D_X-\varepsilon}\right]$.

2 MABI 索引机制

基于移动均值的索引(moving average based indexing,简称 MABI)充分利用了前面已经证明的缩距比关系定理,可以快速淘汰大部分不符合条件的候选对象,缩小了搜索的范围,取得了很好的查询性能.在这部分内容中,我们将先介绍 MABI 索引的结构,然后阐述如何构建索引以及如何从索引中进行相似查询.

2.1 MABI索引结构

定义 4(平衡树[12]).一棵多叉树 T 是平衡树,当且仅当对于树中的任何结点 t,t 的任意两棵子树的高差都不大于 1.

MABI 索引结构的设计借鉴了 Jagadish 等人于 2006 年提出的 BATON*-树[12]索引结构,并作了适当修改,从而支持时间序列的索引.BATON*-树是一种多叉平衡树,它是为建立有效的 P2P 中继网络而提出的一种索引结构,既可以支持精确查询,还可以支持域查询.BATON*-树的结构如图 3 所示,具体描述如下:

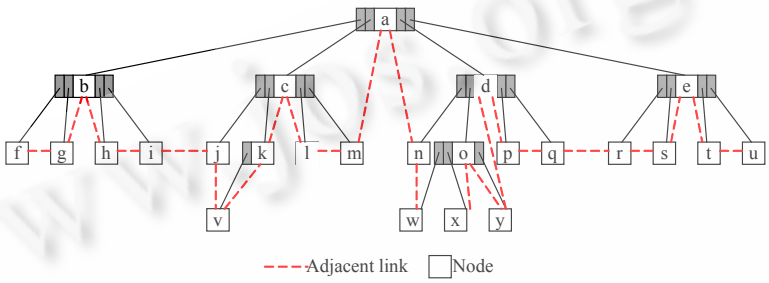


Fig.3 BATON*-tree structure

图 3 BATON*-树结构

- BATON*-树中的每个结点可以有 m 个(m>2)子结点.一个结点连接到其他结点的方式包括:指向父结点的父指针、指向子结点的子指针、指向相邻结点的邻居指针以及邻居路由表.一个父结点不仅要维护到子结点的指针,还要记录子结点所管理的值域.
- BATON*-树中每个结点 t 都维护一个“邻居路由表”,该表中记录了一些指向其他结点的指针.
- 对于 BATON*-树的某个结点 t,如果它的出度为 m,那么该结点所存储的值域要比它的前$\lceil m/2 \rceil$个子结点所存储的值域大,同时要比它的后$\lfloor m/2 \rfloor$个子结点所存储的值域小.

鉴于 BATON*-树的良好性能,及其对搜索代价和更新代价的有效控制,本文的 MABI 索引在设计中借鉴了该树结构.MABI 索引结构如图 4 所示,具体描述如下:

- MABI索引结构中的每个结点可以有 m 个(m>2)子结点.一个结点连接到其他结点的方式包括:指向父结点的父指针、指向子结点的子指针和指向相邻结点的邻居指针.
- MABI 索引结构中的每个结点都存储一个缩距比区间和指向一个信息表 T 的指针.信息表 T 包含 3 个字段:*DRR*,*DS* 和 *Entry*.*DRR* 字段存储一个子序列 $\boldsymbol{S}$ 的缩距比,*DS* 字段存储子序列 $\boldsymbol{S}$ 和水平坐标轴之间的欧氏距离,*Entry* 字段存储子序列 $\boldsymbol{S}$ 在其所属的全段时间序列中的入口位置.根据子序列 $\boldsymbol{S}$ 的入口位置和子序列的长度,就可以得到整个子序列 $\boldsymbol{S}$ 的所有元素.如果某个子序列 $\boldsymbol{S}$ 的缩距比属于该缩距比区间,则把该子序列 $\boldsymbol{S}$ 的相关信息存储到该结点的信息表 T 中.
- 对于MABI索引结构的某个结点 t,如果它的出度为 m,那么该结点所存储的缩距比区间的值比它的前$\lceil m/2 \rceil$个子结点所存储的缩距比区间的值要大,同时比它的后$\lfloor m/2 \rfloor$个子结点所存储的缩距比区间的值要小.例如,在图 4 中,结点 o 所存储的缩距比区间的值比子结点 w 和 x 所存储的缩距比区间的值要大,同时,比子结点 y 所存储的缩距比区间的值要小.
- 对于 MABI 索引结构的某个结点 t,该结点所存储的缩距比区间的值比它的左邻居结点所存储的缩距比要大,同时,比它的右邻居结点所存储的缩距比区间的值要小.也就是说,如果沿着邻居指针对

MABI 索引结构进行遍历,所得到的一系列缩距比区间的值是按照升序排列的.

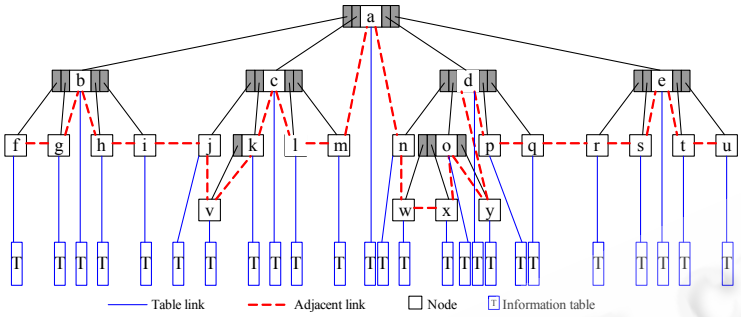


Fig.4 MABI index structure

图 4 MABI 索引结构

MABI 索引结构与一般意义上的多路树存在着一定的区别,在前者中,每个结点只管理一个值区间和 m 个链接,而后者一般管理 $m-1$ 个值区间和 m 个链接.MABI 索引结构也不同于 B-树,对于 B-树而言,每个内部结点都必须有 $m/2$ 到 m 个子结点,而对于 MABI 索引结构而言,除了叶子结点的直接父亲结点以外,所有的内部结点都必须有 m 个子结点[12].

2.2 索引的创建

MABI 索引结构只对缩距比区间进行索引,索引的建立过程按如下步骤进行:

步骤 1:使用 PAA 方法[5]对时间序列 $\boldsymbol{X}$ 执行降维操作,从而得到序列 $\boldsymbol{X}^*$;

步骤 2:在序列 $\boldsymbol{X}^*$的始端放置一个宽度为 w 的窗口,得到一个包含 w 个数据点的子序列 $\boldsymbol{S}$,利用算法 1 把子序列 $\boldsymbol{S}$ 插入到 MABI 索引结构中;

步骤 3:以每步前进一个数据点的方式沿着序列 $\boldsymbol{X}^*$移动窗口,从而把序列 $\boldsymbol{X}^*$的其他子序列插入到 MABI 索引结构中,完成索引的建立过程.

采用 PAA 方法[5]对时间序列 $\boldsymbol{X}$ 执行降维操作,因为该方法不仅简单易懂,而且在性能上也可以与其他较复杂的方法相媲美.在 PAA 方法中,一个长度为 n 的时间序列 $\boldsymbol{Y}=\{y_1,y_2,\ldots,y_n\}$可以被表示成一个 N 维空间向量 $\boldsymbol{C}=\{c_1,c_2,\ldots,c_N\}$(假设 n 可以被 N 整除),其中 $\boldsymbol{C}$ 的第 i 个元素为

$$c_i=\frac{N}{n}\sum_{j=\frac{n}{N}(i-1)+1}^{\frac{n}{N}i}y_j .$$

算法 1. 把子序列 $\boldsymbol{S}$ 插入到 MABI 索引结构.

输入: 1:子序列 $\boldsymbol{S}$

2: MABI 索引结构的根结点 *root*

begin

计算子序列 $\boldsymbol{S}$ 的 D_S 和 r 的值;

为子序列 $\boldsymbol{S}$ 创建一个信息表记录 t;

$t.DRR\leftarrow r$;

$t.DS\leftarrow D_S$;

$t.Entry\leftarrow$子序列 $\boldsymbol{S}$ 在全段时间序列 $\boldsymbol{X}^*$中的入口位置;

$insertPoint\leftarrow LocateInsertPoint(root,r)$;

把信息表记录 t 加入到结点 *insertPoint* 的信息表 T;

if 结点 *insertPoint* 过载 then *SplitNode*(*root*,*insertPoint*); end

```
    if 索引结构失去平衡 then
      BalanceTree(root,insertPoint,n);
    end
  end
```

在MABI索引结构中,每个结点的信息表 T 中只能包含一定数量的记录(每条记录代表一个子序列),当某个结点 n 的信息表 T 中包含过多的记录,出现结点"过载"时,结点 n 的缩距比区间就被分裂成多个子区间,同时,为结点 n 生成多个子结点,被分裂后的多个区间分别由结点 n 以及这些新生成的子结点管理.区间分裂的原则是,分裂后的各个子区间内要包含大致相等数量的记录.当生成新结点时,MABI 树结构可能会失去平衡,因此,需要对树结构进行调整,使其重新到达平衡状态.结点分裂和树的平衡算法可以参见文献[12],这里不再赘述.

2.3 相似查询

假设 $\boldsymbol{Q}$ 是使用 PAA 方法得到的一段查询序列,根据算法 2,我们就可以从 MABI 索引结构中找出那些与 $\boldsymbol{Q}$ 相似的子序列.见算法 2,相似查询过程主要包括如下几个步骤:

步骤 1:使用缩距比关系定理,淘汰大部分不符合条件的候选对象;

步骤 2:使用三角不等定理[11],继续从剩余的候选对象中淘汰一部分不符合条件的候选对象;

步骤 3:使用时间序列相似定义(见定义 2),继续从剩余的候选对象中淘汰所有不符合条件的候选对象.

算法 2. 查找相似的子序列.

```
输入 : 1:长度为 w 的查询序列 Q
       2:MABI 索引结构的根结点 root
       3:时间序列相似门槛值ε
输出: 1:与 Q 相似的所有子序列的集合 S
begin
    初始化一个新的队列 q;
    //步骤 1
    range←CalculateRange(Q,ε);
    lowerNode←GetTheFirstNodeLessThanLowerRangeBound(range,root);
    upperNode←GetTheFirstNodeMoreThanUpperRangeBound(range,root);
    n←lowerNode;
    //步骤 2
    while RightAdjacentNode(n)≠upperNode do
      n←RightAdjacentNode(n);      T←n.tableLink;
      for each t∈T do
        if |D_Q−t.DS| <ε then q.push(n); end
      end
    end
    //步骤 3
    while q.IsEmpty()=FALSE do
      n←q.pop();      t←n.tableLink;
      Z←在时间序列 X* 中的入口位置为 t.Entry 并且长度为 w 的子序列;
      if CalculateEuclideanDistance(Q,Z) <ε then  put Z into S;  end
    end
    return S;
end
```

设置3个步骤进行相似查询的原因在于,对于两个序列 $\boldsymbol{S_1}$ 和 $\boldsymbol{S_2}$,步骤2中所使用的 $\left|D_{S_1}-D_{S_2}\right|<\varepsilon$ 这个条件,要比步骤1中所使用的缩距比关系定理这个条件更加严格.这一点从缩距比关系定理的证明过程很容易得出,这里不再给予证明.同样,步骤3中所使用的约束条件 $D(\boldsymbol{S_1},\boldsymbol{S_2})<\varepsilon$ 也要比步骤2中所使用的约束条件 $\left|D_{S_1}-D_{S_2}\right|<\varepsilon$ 更加严格.步骤2中的约束条件 $\left|D_{S_1}-D_{S_2}\right|<\varepsilon$ 能够起到淘汰不符合要求的候选对象的作用,原因在于,如果两个序列满足 $\left|D_{S_1}-D_{S_2}\right|\geq\varepsilon$,则根据三角不等定理,$D(\boldsymbol{S_1},\boldsymbol{S_2})>\left|D_{S_1}-D_{S_2}\right|\geq\varepsilon$,那么,这两个序列 $\boldsymbol{S_1}$ 和 $\boldsymbol{S_2}$ 就不可能相似.另外,通过使用MABI索引结构中的邻居指针,在算法2中,我们就可以快速地从 *lowerNode* 这个结点遍历到 *upperNode* 这个结点,极大地加快了相似查询的速度.

3 性能分析与讨论

本节进行与性能相关的分析与讨论,包括MABI索引的性能、采用平衡树的必要性以及MABI索引与其他方法的比较.

3.1 MABI索引的性能

我们的MABI索引以BATON*-树为基础,而采用BATON*-树建立索引时,可以把搜索代价降低到 $\boldsymbol{O}(\log_m N)$,其中 m 表示多叉平衡树的扇出系数,N 表示树中的结点数目.在采用树结构建立索引时,在空间限制范围内,m 值越大,搜索代价就越低.但是,m 值增大产生的直接后果就是,更新代价会以 $(m/\log_2 m)^2$ 倍增长.因此,更新代价的增长又使得 m 的值不能过大,必须综合考虑搜索代价和更新代价,从而确定一个合理的 m 值.针对这个问题,文献[12]提出了有效的解决方案,可以使更新代价的增加与 m 呈线性关系,这里不再深入讨论.

3.2 采用平衡树的必要性

MABI索引采用平衡树的原因在于,缩距比区间分布的不均匀性,也就是说,缩距比不是均匀地分布在[0,1]上,而是在[0,1]的某些子区间内的分布密度比较大,在另外一些子区间内的分布密度则比较小.我们对选取的一棵MABI索引树中包含的9 999个缩距比的区间分布情况进行了统计,所有的缩距比都分布在 0.9×10^{-2}~1.8×10^{-2} 之间,表1显示了缩距比的具体分布情况.从表1中可以看出缩距比区间分布的不均匀性,比如,在 1.3×10^{-2}~1.4×10^{-2} 这个范围内包含的缩距比数目占比(占比就是某子区间内的缩距比数目与全部区间内的缩距比数目的百分比)是0.43%,而在 1.6×10^{-2}~1.6×10^{-2} 这个范围内包含的缩距比数目占比却可以达到25%.

Table 1 Distribution of distance reduction rate of the selected sequence

表1 被选取的序列的缩距比分布情况

Range ($\times10^{-2}$)	0~1.0	1.0~1.1	1.1~1.2	1.2~1.3	1.3~1.4	1.4~1.5	1.5~1.6	1.6~1.7	1.7~1.8	1.8~100
Percent (%)	10	5.4	3.2	4	0.43	2.1	4.3	25	12	34

由于树结点管理的缩距比区间内只能包含一定数量的缩距比,达到一定程度以后,就会出现结点“过载”现象,从而导致缩距比区间的分裂和新结点的生成,最终会增加树的深度.当缩距比的区间分布比较均匀时,在最后生成的树中,各个分支的深度都不会有大的差别.但是,在缩距比区间分布不均匀的情况下,在某些子区间内,就会频繁出现过载现象,如果不采用平衡树结构,则会导致树中的某些分支深度过大,某些分支深度又过小,严重影响了查询的性能.为了保证查询性能,采用平衡树结构是非常必要的.

3.3 引入BATON*-树的必要性

MABI索引实质上是根据时间序列的缩距比建立的一维空间内的索引.对于一维空间内的索引,B-树是一种典型的、广泛应用的多路平衡树,但是,MABI索引的设计中,并不是以B-树为基础,而是以BATON*-树为基础,主要原因是BATON*-树是一种同时支持精确查询和域查询的索引结构,具有邻居指针,可以实现快速的域查询,而B-树则没有这个特点.在进行时间序列相似查询时,需要根据当前查询 Q 和缩距比关系定理计算得到一个缩距比区间,然后获得索引树中所有落入缩距比区间内的对象,这种查询属于域查询,BATON*-树的邻居指针可以在域查询中发挥重要作用.在实验部分,我们将证明基于BATON*-树的MABI索引可以取得比B-树索引更好

的性能.

此外,BATON*-树具有更新代价的有效控制解决方案,可以使更新代价的增加与 m 呈线性关系.更新代价是时间序列数据库设计必须要考虑的问题,好的索引结构既要尽量最小化查询代价,也应合理控制更新代价.

3.4 与其他方法的比较

针对时间序列的相似查找,目前比较常用的方法都是通过降低维数来达到数据压缩的目的,然后利用 R*-树对 N 维空间中的目标向量建立索引.但是,在高维空间中,类似 R*-树这样的层次索引结构通常无法取得令人满意的性能.在某些情况下,即使采用了降维技术,相似查询工作仍然需要访问大量索引.

本文在研究这个问题时,采取了与以前方法截然不同的技术路线.我们首先通过缩距比关系定理,把时间序列之间的相似性判定从多维空间转化到一维空间中,然后利用一维空间中的索引技术建立 MABI 索引结构,从而实现时间序列的快速相似查找.一维空间的索引比多维空间的索引具有更好的性能,因此,在时间序列的相似查找问题上,MABI 索引具有比其他索引更好的性能,我们将在实验部分给出实验结果.

另一方面,当采用其他索引(比如 TIP[13]和 STB[14])进行时间序列相似查找时,需要临时计算查询序列 Q 与其他候选对象之间的评估指标(比如欧氏距离),而在 MABI 索引中,所有缩距比都事先计算并存储在信息表中,在候选对象的筛选阶段,由于采用缩距比关系定理来淘汰不符合条件的对象,查询序列 Q 只需要与候选对象之间进行缩距比的比较即可,这时就可以直接使用这些预先存储的缩距比,而不需要临时计算,从而极大地加快了查找速度.

4 实验设计与结果

本节介绍我们对 MABI 索引进行的实验及其结果.实验算法用 C++语言编写.实验的硬件环境是:HP Proliant DL585 服务器,其配置为:4 颗 AMD 皓龙 2.4G CPU(双核)、32GB 内存和 1.20TB 硬盘.实验的软件环境是:Windows Server 2003 操作系统和 ORACLE 10g 数据库管理系统.

我们选取了美国股市 1980 年~2003 年的所有股票交易记录(下载地址:http://www.macd.cn),获得了 1 228 764 条交易记录,每条记录包含交易日期、开盘价、收盘价和成交量等信息.我们选取了收盘价信息,从而得到 1 228 764 个数据点,然后从中截取 10 段时间序列,每段包含 100 000 个数据点.我们先用 PAA 方法把这些时间序列转换成长度为 10 000 的序列,然后让滑动窗口在该序列上移动,边移动边计算窗口内所包含的子序列的缩距比 r 等信息,并利用算法 2 建立 MABI 索引,最终得到的每棵索引树中包含 9 999 个结点.

4.1 实验1:缩距比关系定理淘汰能力

实验的目的在于证明缩距比关系定理淘汰不符合条件的候选对象的能力.我们从前面得到的 10 段序列中任意选取一段长度为 200 的子序列作为查询序列 q,计算得到该序列的缩距比为 1.628×10^{-2}.假设 $\varepsilon=D_q/f$,其中,D_q 是查询序列 q 与水平坐标轴之间的欧氏距离.表 2 显示了当 f 的取值从 1 变化到 5 500 时,缩距比区间 *range* 和缩距比关系定理淘汰能力的变化情况.在表 2 中,淘汰率的定义是被淘汰的结点个数除以 MABI 索引结构中的总结点数.

Table 2 Performance of DRR relation theorem when varying the value of f

表 2 当 f 取不同值时缩距比关系定理的性能

f	1	500	1 000	1 500	2 000	2 500
Pruning amount	0	3 551	5 488	6 029	6 538	6 718
Pruning rate (%)	0	35.51	54.89	60.30	65.39	67.19
range($\times10^{-2}$)	–99.19~100.81	1.226~2.024	1.427~1.826	1.494~1.760	1.527~1.727	1.547~1.707
f	3 000	3 500	4 000	4 500	5 000	5 500
Pruning amount	7 276	8 001	8 440	8 693	8 791	8 915
Pruning rate (%)	72.77	80.02	84.41	86.94	87.92	89.16
range($\times10^{-2}$)	1.561~1.694	1.570~1.685	1.578~1.678	1.583~1.672	1.588~1.668	1.591~1.664

为了进一步说明淘汰率的变化情况,我们在图 5 中给出了当 f 取 1~10 000 之间的不同值时得到的淘汰率曲

线.从图 5 中我们可以看出,f 值越大,缩距比关系定理的淘汰能力越强.比如,当 f=500 时,淘汰率是 35.51%;当 f=10000 时,淘汰率可以高达 95.2%.另外,随着 f 值的增加,其淘汰个数增加率也逐渐降低.也就是说,当 f 比较小时,小幅度增加 f 的值,就可以带来很大程度的淘汰能力的增加;而当 f 增大到一定程度时,即使再大幅度增加 f 的值,其淘汰能力的增加也很有限.比如,当 f 从 1 增加到 500 时,淘汰率从 0 增加到 35.51%;但是,当 f 从 10 000 增加到 15 000 时,淘汰率仅从 95.2%增加到 96.2%.

图 6 显示了当 f 取 1~10 000 之间的不同值时缩距比区间 *range* 的上界和下界的变化情况,从中可以看出,随着 f 取值的增大,缩距比区间 *range* 的上界和下界不断向 1.628×10^{-2}(查询序列 q 的缩距比)逼近,由此导致缩距比区间 *range* 的长度不断减小(如图 7 所示),这也意味着缩距比关系定理的淘汰能力逐渐增强.

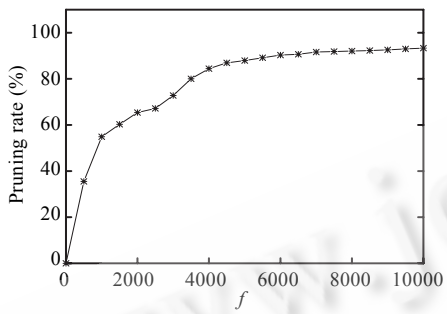


Fig.5 The change of pruning rate
图 5 淘汰率的变化

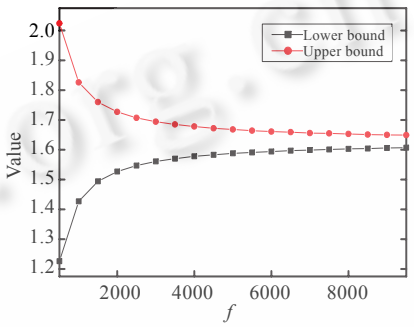


Fig.6 The change of DRR range bound
图 6 缩距比区间边界变化

4.2 实验2:MABI索引与*k-d*树的性能比较

k-d 树[15]是一种支持多关键字的多维二叉查找树,并可用于存储 k 维空间数据点.本实验中,我们让 MABI 索引的扇出系数 m 分别取值为 2,4 和 6,并以性能比(*k-d* 树查询时间/MABI 索引查询时间)作为评估两种方法性能的指标.图 8 显示了当 f 值(该值的含义见实验 1)从 500 增大到 5 000 时,两种方法性能比的变化情况.从实验结果可以看出,在 m=2,m=4 和 m=6 这 3 种情况下,MABI 索引都比 *k-d* 树取得了更好的查询性能,而且,m 值越大,性能比越大.但是,需要指出的是,m 值的增大会导致索引树更新代价的增加,因此,在实际应用中,m 的值并不是越大越好,必须综合考虑搜索代价和更新代价,从而确定一个合理的 m 值,确定 m 值的方法可以参考文献[12].

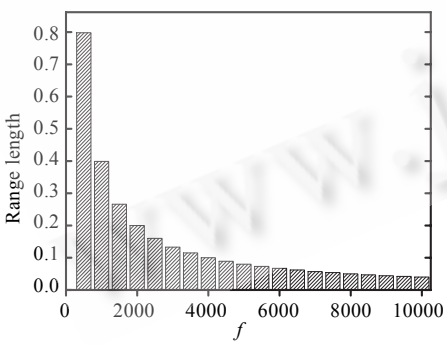


Fig.7 The change of DRR range length
图 7 缩距比区间长度变化

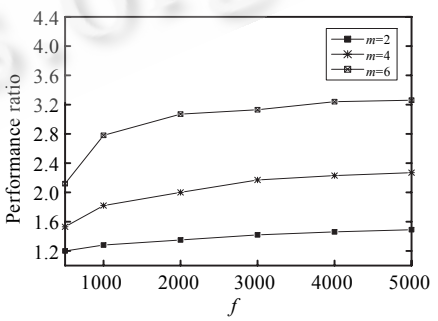


Fig.8 Performance ratio of MABI index and *k-d* tree
图 8 MABI 索引与 *k-d* 树的性能比

4.3 实验3:MABI索引与B-树的性能比较

B-树是被广泛使用的一维空间内的索引结构,本实验中,我们分别用 B-树和 MABI 结构为缩距比建立索引树,然后让 B-树和 MABI 索引的扇出系数分别取值为 2,4 和 6,并以性能比(B-树查询时间/MABI 索引查询时间)

作为评估两种方法性能的指标.图 9 显示了当f值(该值的含义见实验 1)从 500 增大到 5 000 时,两种方法性能比的变化情况.从实验结果可以看出,在 m=2,m=4 和 m=6 这 3 种情况下,MABI 索引都比 B-树取得了更好的查询性能,而且,m 值越大,性能比越大.MABI 索引与 B-树的性能差别的原因在于,相对于 B-树而言,MABI 索引可以更好地支持域查询,后者在结构的设计上使其不仅可以支持精确查询,也可以很好地支持域查询.由于 MABI 索引增加了 B-树所没有的邻居指针,在相似搜索过程中,搜索算法充分发挥了邻居指针的作用,极大地加快了缩距比区间的搜索速度,为相似搜索过程的总体性能带来了较大的提升.

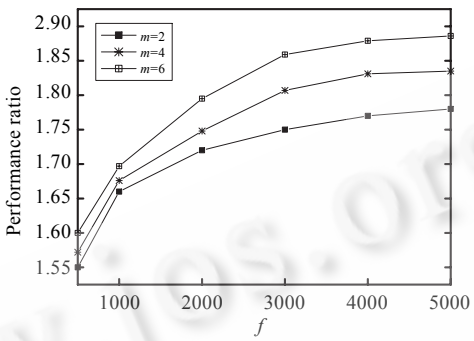


Fig.9 Performance ratio of MABI index and B-tree
图 9 MABI 索引与 B-树的性能比

5 相关工作

时间序列的快速相似查询需要借助于有效的索引机制.R*-树[16]就是一种经常被使用的层次索引结构,除此之外,近年来还诞生了一些新的专门针对时间序列的索引方法.在文献[13]中,作者描述了扩展多维动态索引文件(extended multidimensional dynamic index file,简称 EMDF)方法,它存储和查找那些用多键记录表示的时间序列.EMDF 方法一般用在预选过程中,为了加快模式匹配的速度,它会从所有时间序列模式集合中筛选出一个比较小的候选集,在实际匹配过程中就可以使用这个预先筛选得到的候选集,从而改善了整体性能.但是 EMDF 方法也有一些缺点,比如,它需要维护动态域表,这就要存储许多指针类型变量.通常情况下,时间序列模式的维数较高而且数量大,这样,随着维数和模式数量的增加,所需存储的指针也会相应地急剧膨胀.为了克服 EMDF 的这些缺点,文献[13]对它作了进一步改善,提出了称为“TIP-索引”的新索引方法,取得了更好的性能.TIP-索引包含多个帧标识符块(FID blocks)、一个动态区域表(DRT)和多个层次目录块(HD blocks).就结构而言,TIP-索引和 EMDF 非常相似,二者单个的 FID block 是完全相同的,都由若干个 FID 组成.但是,TIP-索引中只有一个 DRT 来表示所有的维,而 EMDF 却需要多个 DRT 来表示.在插入和搜索方面,TIP-索引要比 EMDF 快许多,而且,TIP-索引的性能在不同类型数据分布情况下比较稳定、一致,在这一点上 EMDF 的表现不尽如人意,它的性能会随着数据分布的变化而改变.

文献[14]提出了一种新的索引方法——STB-索引(shape-to-vector-index),这是第 1 个提出的对分段线性表示进行索引,它使用一个固定长度的窗口在时间序列上一步步移动,得到长度相等的多个时间子序列,然后把具有相似形状的时间子序列放到一个箱子里,对于每个箱子,它可以快速计算给定的查询序列和箱子中包含序列之间的欧氏距离的下界限,这样就可以按照最佳者优先的顺序进行查找,同时也实现了箱间裁剪,即不用详细察看每个箱子的具体内容而直接剔除那些不满足查询条件的箱子,另外,利用优化技术,也可以进行箱内裁剪,不用与箱子中包含的每个子序列进行比较,而直接删除那些不满足条件的子序列,利用箱间裁剪和箱内裁剪两种技术,大大加快了查找速度,提高了性能.前面提到的 TIP-索引和 EMDF 都主要是用在预选过程来找到一个候选集,而并非找到最终符合条件的精确匹配结果,并且每个帧的大小是固定不变的,但 STB-索引则返回一个最终符合条件的时间序列集合,而且提出了在查询序列长度不等时的解决方案.

除了上述几种方法以外,文献[17]提出了使用优点树(vantage-point-tree)进行索引的方法,文献[18]使用最大和最小值进行索引,文献[19]中的索引方法则是基于特征的多层抽象层级结构,文献[20]提出了基于网格的索引,文献[21]也提出一种称为 DDR 的方法建立时间序列的索引.当然还有一些性能较好的索引方法,有些还是针对某一特定领域和应用提出来的,由于篇幅关系,这里不再一一列举.

6 结束语

本文提出了缩距定理和缩距比关系定理,并引入一种新的多叉平衡结构——BATON*-树,在此基础上,我们提出了MABI索引.缩距比关系定理可以淘汰大部分不符合条件的候选对象,缩小搜索范围;基于BATON*-树的MABI索引结构可以根据缩距比对时间序列建立索引,极大地加快了相似查询过程.MABI索引方法不同于现有的其他索引方法,该方法设计简单,只需对缩距比进行索引,避免了因维数增加而导致的查找性能下降的问题.实验结果表明,MABI 索引具有良好的查询性能.在今后的研究工作中,我们将继续在改进索引性能方面做出努力,并研究支持不同长度查询的有效方法.

References:

[1] Michael KN, Huang Z, Hegland M. Data-Mining massive time series astronomical data sets-a case study. In: Wu XD, Ramamohanarao K, Korb KB, eds. Proc. of the 2nd Pacific-Asia Conf. on Knowledge Discovery and Data Mining (PAKDD'98). Melbourne: Springer-Verlag, 1998. 401−402.

[2] Agrawal R, Faloutsos C, Swami A. Efficient similarity search in sequence databases. In: Lomet DB, eds. Proc. of the 4th Conf. on Foundations of Data Organization and Algorithms (FODO'93). Chicago: Springer-Verlag, 1993. 69−84.

[3] Agrawal R, Lin KI, Sawhney HS, Shim K. Fast similarity search in the presence of noise, scaling, and translation in times-series databases. In: Dayal U, Gray PM, Nishio S, eds. Proc. of the 21st Int'l Conf. on Very Large Data Bases (VLDB'95). Zurich: Morgan Kaufmann Publishers, 1995. 490−501.

[4] Chan K, Fu W. Efficient time series matching by wavelets. In: Proc. of the 15th IEEE Int'l Conf. on Data Engineering (ICDE'99). Sydney: IEEE Computer Society, 1999. 126−133.

[5] Keogh EJ, Chakrabarti K, Pazzani MJ, Mehrotra S. Dimensionality reduction for fast similarity search in large time series databases. Journal of Knowledge and Information Systems, 2000,3(3):263−286.

[6] Yi BK, Faloutsos C. Fast time sequence indexing for arbitrary *Lp* norms. In: Abbadi AE, Brodie ML, Chakravarthy S, Dayal U, Kamel N, Schlageter G, Whang KY, eds. Proc. of the 26th Int'l Conf. on Very Large Databases (VLDB 2000). Cairo: Morgan Kaufmann Publishers, 2000. 385−394.

[7] Keogh EJ, Chakrabarti K, Mehrotra S, Pazzani MJ. Locally adaptive dimensionality reduction for indexing large time series databases. In: Aref WG, ed. Proc. of the 2001 ACM SIGMOD Int'l Conf. on Management of Data (SIGMOD 2001). Santa Barbara: ACM, 2001. 151−162.

[8] Keogh EJ. Fast similarity search in the presence of longitudinal scaling in time series databases. In: Proc. of the 9th Int'l Conf. on Tools with Artificial Intelligence (ICTAI'97). Newport Beach: IEEE Computer Society, 1997. 578−584.

[9] Vega-Lopez IF, Moon B. Quantizing time series for efficient subsequence matching In: Hamza MH, ed. Proc. of the 24th IASTED Int'l Conf. on Database and Applications (DBA 2006). Innsbruck: IASTED/ACTA Press, 2006. 209−214.

[10] Kim SW, Yoon J, Park S, Kim TH. Shape-Based retrieval of similar subsequences in time-series databases. In: Proc. of the 2002 ACM Symp. on Applied Computing (SAC 2002). Madrid: ACM, 2002. 438−445.

[11] Shasha D, Wang TL. New techniques for best-match retrieval. ACM Trans. on Information Systems, 1990,8(2):140−158.

[12] Jagadish HV, Ooi BC, Tan KL, Vu QH, Zhang R. Speeding up search in peer-to-peer networks with a multi-way tree structure. In: Chaudhuri S, Hristidis V, Polyzotis N, eds. Proc. of the 2006 ACM SIGMOD Conf. on Management of Data (SIGMOD 2006). Chicago: ACM, 2006. 1−12.

[13] Kim YI, Park YB, Chun JH. A dynamic indexing structure for searching time-series patterns. In: Proc. of the 20th Int'l Computer Software and Applications Conf. (COMPSAC'96). Seoul: IEEE Computer Society, 1996. 270−275.

[14] Keogh EJ, Pazzani MJ. An indexing scheme for fast similarity search in large time series databases. In: Ozsoyoglu ZM, Ozsoyoglu G, Hou WC, eds. Proc. of the 11th Int'l Conf. on Scientific and Statistical Database Management (SSDBM'99). Cleveland: IEEE Computer Society, 1999. 56−67.

[15] Lee SL, Chun SJ, Kim DH, Lee JH, Chung CW. Similarity search for multidimensional data sequences. In: Proc. of the 16th Int'l Conf. on Data Engineering (ICDE 2000). San Diego: IEEE Computer Society, 2000. 599−608.

[16] Beckmann N, Kriegel HP, Schneider R, Seeger B. The R*-tree: An efficient and robust access method for points and rectangles. In: Garcia-Molina H, Jagadish HV, eds. Proc. of the 1990 ACM SIGMOD Conf. on Management of Data (SIGMOD'90). Atlantic City: ACM, 1990. 322−330.

[17] Bozkaya T, Ozsoyoglu M. Indexing large metric spaces for similarity search queries. ACM Trans. on Database Systems, 1999, 24(3):361−404.

[18] Park S, Kim SW, Chu WW. Segment-Based approach for subsequence searches in sequence databases. In: Proc. of the 2001 ACM Symp. on Applied Computing (SAC 2001). Las Vegas: ACM, 2001. 248−252.

[19] Li CS, Yu PS, Castelli V. MALM: A framework for mining sequence database at multiple abstraction levels. In: Gardarin G, French JC, Pissinou N, Makki K, Bouganim L, eds. Proc. of the 1998 ACM 7th Int'l Conf. on Information and Knowledge Management (CIKM'98). Bethesda: ACM, 1998. 267−272.

[20] An JY, Chen HX, Furuse K, Ohbo N, Keogh EJ. Grid-Based indexing for large time series databases. In: Liu JM, Cheung YM, Yin HJ, eds. Proc. of the 4th Int'l Conf. on Intelligent Data Engineering and Automated Learning (IDEAL 2003). Hong Kong: Springer-Verlag, 2003. 614−621.

[21] An JY, Chen YP, Chen HX. DDR: An index method for large time-series datasets. Information Systems, 2005,30(5):333−348.

附录. 定理 1(缩距定理)的证明过程

证明:(1) 当两个时间序列都是相同的等值序列,如两者都是{3,3,3,3,3}时,$\boldsymbol{D}(MV_2(\boldsymbol{X}),MV_2(\boldsymbol{Y}))=\boldsymbol{D}(\boldsymbol{X},\boldsymbol{Y})$显然成立.

(2) 当其他情况时,因为距离值大于 0,我们只需要证明距离的平方满足以下条件即可:

$$\boldsymbol{D}^2(MV_2(\boldsymbol{X}),MV_2(\boldsymbol{Y}))\leq\boldsymbol{D}^2(\boldsymbol{X},\boldsymbol{Y}).$$

根据移动均值的定义:

$$MV_2(\boldsymbol{X})=\left\{\frac{x_0+x_1}{2},\frac{x_1+x_2}{2},\ldots,\frac{x_{n-2}+x_{n-1}}{2}\right\};$$

$$MV_2(\boldsymbol{Y})=\left\{\frac{y_0+y_1}{2},\frac{y_1+y_2}{2},\ldots,\frac{y_{n-2}+y_{n-1}}{2}\right\};$$

所以可得:

$$\boldsymbol{D}^2(MV_2(\boldsymbol{X}),MV_2(\boldsymbol{Y}))=\sum_{i=0}^{n-2}\left(\frac{x_i+x_{i+1}}{2}-\frac{y_i+y_{i+1}}{2}\right)^2=\sum_{i=0}^{n-2}\left(\frac{x_i-y_i}{2}+\frac{x_{i+1}-y_{i+1}}{2}\right)^2=\frac{1}{4}\sum_{i=0}^{n-2}\left[(x_i-y_i)+(x_{i+1}-y_{i+1})\right]^2,$$

又因为 $\boldsymbol{D}^2(\boldsymbol{X},\boldsymbol{Y})=\sum_{i=0}^{n-1}(x_i-y_i)^2$,所以,

$$\boldsymbol{D}^2(\boldsymbol{X},\boldsymbol{Y})-\boldsymbol{D}^2(MV_2(\boldsymbol{X}),MV_2(\boldsymbol{Y}))=\sum_{i=0}^{n-1}(x_i-y_i)^2-\frac{1}{4}\sum_{i=0}^{n-2}\left[(x_i-y_i)+(x_{i+1}-y_{i+1})\right]^2$$

$$=\frac{1}{4}\left[\begin{array}{l}3(x_0-y_0)^2+2(x_1-y_1)^2+\ldots+2(x_{n-2}-y_{n-2})^2+3(x_{n-1}-y_{n-1})^2-2(x_0-y_0)(x_1-y_1)\\-2(x_1-y_1)(x_2-y_2)-\ldots-2(x_{n-2}-y_{n-2})(x_{n-1}-y_{n-1})\end{array}\right]$$

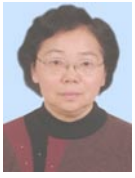
$$=\frac{1}{4}\begin{Bmatrix}2(x_0-y_0)^2+2(x_{n-1}-y_{n-1})^2+\\ \left[(x_0-y_0)^2-2(x_0-y_0)(x_1-y_1)+(x_1-y_1)^2\right]+\\ \left[(x_1-y_1)^2-2(x_1-y_1)(x_2-y_2)+(x_2-y_2)^2\right]+\\ \ldots+\\ \left[(x_{n-2}-y_{n-2})^2-2(x_{n-2}-y_{n-2})(x_{n-1}-y_{n-1})+(x_{n-1}-y_{n-1})^2\right]\end{Bmatrix}$$

$$=\frac{1}{4}\begin{Bmatrix}2(x_0-y_0)^2+2(x_{n-1}-y_{n-1})^2+\\ \left[(x_0-y_0)-(x_1-y_1)\right]^2+\\ \left[(x_1-y_1)-(x_2-y_2)\right]^2+\\ \ldots+\\ \left[(x_{n-2}-y_{n-2})-(x_{n-1}-y_{n-1})\right]^2\end{Bmatrix}$$

>0.

所以可得:$\boldsymbol{D}^2(MV_2(\boldsymbol{X}),MV_2(\boldsymbol{Y}))<\boldsymbol{D}^2(\boldsymbol{X},\boldsymbol{Y})$.证毕. □

林子雨(1978－),男,吉林柳河人,博士生,CCF 学生会员,主要研究领域为数据库,实时主动数据仓库,数据挖掘.



杨冬青(1945－),女,教授,博士生导师,CCF 高级会员,主要研究领域为数据库,数据仓库,Web 数据集成,移动数据挖掘.

王腾蛟(1973－),男,博士,副教授,CCF 高级会员,主要研究领域为数据库,数据仓库,Web 数据集成,数据挖掘.