文章编号:1001 - 9081(2007)05 - 1232 - 03

基于 DCT 的时序数据相似性搜索

崔 振[1],任亚洲[2],王 瑞[3]
(1. 华侨大学 信息科学与工程学院,福建 泉州 362021;
2. 山东政法学院 司法信息系,山东 济南 250014;
3. 厦门大学 数学科学学院,福建 厦门 361005)
(hqucuizhen@126.com)

摘 要:数据的高维度是造成时序数据相似性搜索困难的主要原因。最有效的解决方法是对时序数据进行维归约,然后对压缩后的数据建立空间索引。目前维归约的方法主要是离散傅立叶变换(DFT)和离散小波变换(DWT)。提出了一种新的方法,利用离散余弦变换(DCT)进行维归约,并在此基础上给出了对时序数据进行范围查询和近邻查询的相似性搜索方法。与基于 DFT、DWT 的搜索方法相比,该方法在理论分析和实验结果上都显示出较高的效率。

关键词:时间序列; 离散余弦变换; 范围查询; 近邻查询
中图分类号: TP311.13 **文献标识码**:A

Similarity search over time series data using DCT

CUI Zhen[1], REN Ya-zhou[2], WANG Rui[3]
(1. *College of Information Science and Engineering, Huaqiao University, Quanzhou Fujian* 362021, *China*;
2. *Department of Law and Information, Shandong University Political Science and Law, Jinan Shandong* 250014, *China*;
3. *School of Mathematical Sciences, Xiamen University, Xiamen Fujian* 361005, *China*)

Abstract: High dimensionality is the main difficulty of similarity search over time-series data. The most promising solution involves performing dimensionality reduction on the data, then indexing the reduced data with a spatial method. Recently, two methods of dimensionality reductions have been proposed, DFT and DWT. In this paper we proposed a new method, dimensionality reduction with DCT, and further provided the method of similarity search about range query and nearest neighbor query. Compared with those methods based on DFT and DWT, it is more efficient in theory and experiment.

Key words: time series; Discrete Cosine Transform(DCT); range query; nearest neighbor query

0 引言

时间序列(简称时序)是一组有序的、随时间变化的数值序列,例如股票价格数据、产品销售记录、地区气温等。时间序列数据的挖掘是数据挖掘的一个重要分支,研究它会带来广泛的社会效益。例如在股票市场,通过对股票价格的历史走势分析,可以预测未来的走势;电力部门可以通过对用电量的分析,指导电力分配;医学领域,医生能通过对药物疗效的数据分析,掌握药物的特性等。

时间序列数据的挖掘体现在多个方面,比如从时间序列中发现关联规则,时序数据集的分类以及聚类等。所有这些工作的前提都是要比较时间序列之间的相似性。度量两个序列的相似性,标准的方法是采用欧式距离。两个时间序列

$$\begin{cases}\vec{x} = \{x_0,\cdots,x_{n-1}\}\\ \vec{y} = \{y_0,y_1,\cdots,y_{n-1}\}\end{cases}$$

的欧式距离为

$$D(\vec{x},\vec{y}) = \sqrt{\sum_{i=0}^{n-1} |x_i - y_i|^2}$$

在进行序列的相似性搜索之前还给出一个阈值 ε,如果 $D(\vec{x},\vec{y}) \leqslant \varepsilon$,我们就说 $\vec{x},\vec{y}$ 两者在 ε 内相似。

时序数据相似性搜索面临的一个困难是它的高维性。如果直接对时序数据库中所有序列依次进行扫描,计算与查询序列的距离,计算量是非常大的,并且在查询序列变化时,还要再重新扫描数据库,这对庞大的数据库而言是不现实的。而大部分连续的时间序列中的点不是相互独立的,而是彼此相关的,因此一定存在信息冗余。所以相似性搜索最有效的方法是先对时间序列进行维归约(dimension reduction)以提取序列的特征,然后用空间索引结构建立基于特征空间的索引。这种方法首先由 Agrawal 提出,用离散傅立叶变换(DFT)将时间域上的时间序列数据变换为频率域上的序列数据,用前 k 个系数作为序列的特征,然后针对特征空间建立索引结构[1]。但这种方法限制每个序列的长度相同,Faloutsos 克服了这个困难,通过滑动窗口(sliding window)将时间序列分解为一系列长度相同的子序列,再由子序列抽取特征进行相似性搜索[2]。

近年来,Chan 和 Fu 首次用离散小波变换(DWT)进行维归约,通过保留小波变换后的前 k 个位置的系数,即低频特征,得到时间序列的一个粗略逼近,并通过实验证明了 DWT 的优越性[3]。在此基础上,一些学者做了更深入的研究[4~6]。其中,Wu 等对用 DFT 和 DWT 进行搜索做了全面的比较[6]。

收稿日期:2006 - 11 - 20;**修订日期**:2007 - 01 - 29

作者简介:崔振(1981 -),男,山东菏泽人,助教,硕士,主要研究方向:数据挖掘、软件工程; 任亚洲(1982 -),男,山东菏泽人,助教,硕士,主要研究方向:数据挖掘; 王瑞(1978 -),男, 山东定陶人,硕士研究生,主要研究方向:数据挖掘、偏微分方程.

本文基于欧式距离,从时间序列经 DFT 变换后序列系数的复对称性中得到启发,提出一种新的方法,用离散余弦变换(DCT)将时间序列从时间域变换到频率域,然后抽取 k 个特征并映射到 k 维空间上的点,并通过建立空间索引结构(R-树)来加快搜索速度。这主要是因为 DCT 具有两方面的优势:

1) DCT 变换后的前 k 个系数能量集中和去相关性能都优于 DFT 和 DWT[7],因此抽取特征量的能量与原来的序列更接近,从而加强了搜索时的剪枝处理能力,提高了查询的准确性和速度;

2) DFT 变换后的每个系数都是复数,所以对于 k 个系数要建立 $2k$ 维的空间索引结构;而 DCT 变换后的每个系数是实数,所以空间索引结构的维数保持不变,因此降低了存储空间。

目前,以欧式距离作为相似性评价函数,用 DWT 进行查询是最好的相似性搜索方法,而用 DFT 查询是经典的搜索方法。所以本文用我们提出的方法和基于 DFT、DWT 的相似性搜索方法做了详细的比较,理论分析与实验都显示了 DCT 的优越性。

1　离散余弦变换

离散余弦变换是一种实数域变换,克服了离散傅立叶变换中复数域运算的缺点,符合人类语言及图像信号的特点。因此已应用于数字信号处理、频谱分析等领域中[7]。离散余弦变换有两种定义方法,我们只给出本文中使用的一种定义[7]。

定义 1　设时间序列 $\vec{x}$,经变换后的序列为 $\vec{X}$,则有

$$X_0 = \frac{1}{\sqrt{n}}\sum_{m=0}^{n-1} x_m$$

$$X_k = \sqrt{\frac{2}{n}}\sum_{m=0}^{n-1} x_m \cos\left[\frac{(2m+1)k\pi}{2n}\right]$$

其中,$k = 1,2,\cdots,n-1$。

上述的 DCT 定义是一个正交变换[7],而正交变换保持欧式距离[4],很容易得到下面的能量保持定理和距离保持定理。

定理 1　设 $\vec{X}$ 是序列 $\vec{x}$ 经 DCT 变换后的序列,则有

$$\sum_{i=0}^{n-1} |x_i|^2 = \sum_{k=0}^{n-1} |X_k|^2$$

定理 2　设 $\vec{X}$, $\vec{Y}$ 分别是序列 $\vec{x}$, $\vec{y}$ 经 DCT 变换后的序列,则有

$$D(\vec{x},\vec{y}) = D(\vec{X},\vec{Y})$$

上述定理表明两个序列的欧式距离在时间域和频率域相同,而且很容易推得下面的定理

定理 3　如果仅用 DCT 的前 $k(1 \leqslant k \leqslant n)$ 维,那么在查询中也不会漏掉具有资格的对象,即不会出现误漏现象。

这可能会保留一些没有资格的对象,即出现假警报现象,因为抽取特征的能量是原序列能量的下界。但我们可以通过验证原对象间的距离而排除假警报。

2　相似性搜索

在查询之前,要做一些预处理工作。由于实验中采用了子序列匹配[8]形式组织的数据,所以先用窗口在序列上滑动产生若干个长度相同的子序列。但这可能会出现子序列不相似于它在 y 轴方向上平移后的序列,我们以每个子序列减去它的平均值所产生的序列作为查询对象,执行 DCT(或 DFT, DWT),然后再抽取特征,并建立相应的空间索引 R-树。

在建立索引后,就可以进行两种类型的查询:范围查询和近邻查询。范围查询就是返回与查询序列 q 距离小于等于 ε 的序列。在范围查询时,首先要变换查询序列 q 到相应的特征空间上的点 q_f,然后再进行范围查询。下面给出范围查询算法描述:

```
RangeQuery( nd, qf, ε, resultList)
// nd 为 R-tree 节点指针, qf 为查询序列, ε 阈值
begin
  if ( nd is a leaf node) then
    while( every objects pi in nd)
      if( overlap( pi, qf_rect))
      // qf_rect是以 qf 为中心边长 2 * ε 的 MBR
        find original objects pi' by index;
      //叶子节点中的索引号索引原序列
      if( distance( q, pi')<= ε) //q 为 qf 原对象
      insert( resultList, pi'); // pi'加入 resultList
  else
    while( every children nodes ri of nd)
      if ( distance( ri, qf) <= ε)
        RangeQuery( ri, qf, ε, resultList)
end
```

近邻查询是指在时序数据库中找出与所给的查询序列最相似的 m 个序列。在已有的近邻算法[9]的基础上,我们给出动态改变 ε 的算法。算法如下:

```
NearestNerghbor( nd, qf, resultList)
begin
  if ( nd is a leaf node)
    while( every objects pi indexing in nd)
      if ( overlap( pi, qf_rect))
        find original objects pi' by index;
        dis = distance( q, pi');
        if ( len( resultList) < k)
          insert ( pi', dis) into resultListby ascending ordering of dis
          and update( ε);
        else
          if ( dis < ε)
            insert ( pi', dis) into resultList by ascending ordering
            of dis; delete the tail of resultList; update( ε);
  else
    generate childrenList of nd;
    compute distance between every children ri and qf;
    sort the childrenList by ascending;
    for i = 0 to len( childrenlist) - 1
      if ( distance( ri, qf) <= ε)
        NearestNerghbor( ri, qf, resultList);
      else
        break;
end
```

此算法利用了启发式搜索策略,当查询点 q_f 与节点 r_i 的距离大于 ε 时,则节点以下的路径会被剪去。因为 ε 是当前所搜索到的第 m 个最近点的距离,若 $distance(r_i, q_f) > \varepsilon$,则由

于 $distance(r_i, q_f)$ 是分枝 r_i 中的每个孩子节点的下界，所以分枝 r_i 包含的叶子节点中的每个特征点到 q_f 的距离大于 ε，则由定理 2 和定理 3 可以保证分枝 r_i 包含的原序列到查询序列 q 的距离大于 ε，这就避免了出现误漏现象。

在算法中，采取了两种策略来提高搜索的效率。一种策略是对查询点 q_f 与节点 r_i 的距离进行了排序。我们考虑最坏的情况，假设查询点 q_f 与节点 r_i 的距离按降序排列，那么从 r_1 开始查询所有分枝节点，最坏情况是当查询完 r_i 分枝后 $distance(r_{i+1}, q_f) \leqslant \varepsilon$，则还需要查找 r_{i+1} 分枝，这样就需要扫描整个 *R*- 树和整个时序数据库。而按升序排列，由于提取的特征能量很接近原来的序列，也就是仅仅搜索孩子列表中前几个分枝就可以得到最终的结果（除非建立的索引结构极差），且一旦出现 $distance(r_i, q_f) > \varepsilon$ 时，则孩子列表中 r_i 后面的孩子分枝 $r_j (j > i)$ 就可以剪去，因此提高了搜索的效率。另一种策略是不断的更新阈值 ε。由于 ε 始终为查询过程中 q_f 到第 m 个近邻的上界，所以 ε 的更新不会出现误漏现象。更重要的是 ε 的值不断减少，使得满足 q_f 与节点的距离小于 ε 的分枝数大量减少，即增强了剪枝能力，从而加快了搜索的速度。

3 实验结果

实验中采用 R-树索引方法，并且每个节点的最大分枝数为 8。同时，由于要与 DWT，DFT 进行一系列的比较，所以序列长度被限制为 2 的整数幂次方。

我们测试两种数据集：

1）股票数据：采用从 1980 年 1 月 2 日到 2006 年 9 月 29 日 IBM 公司的每日股票信息（周末除外），含有 6 752 个记录的数据集（网址 http://cn.finance.yahoo.com/q?s=IBM）。我们只考虑收盘价，数据是一个长度为 6 752 的序列，用一个宽度为 n 的滑动窗口产生 6 753 - n 个子序列，以这些序列消除偏移所产生的序列作为实验对象。

2）随机数据：产生长度为 6 000 的序列 x，且 $x_{i+1} = x_i + z_i$，z_i 是属于[-100,100]的随机数。

所做实验的计算机配置为，Windows XP，VC++6.0，CPU 2.0GHz，512MB 内存。

3.1 范围查询

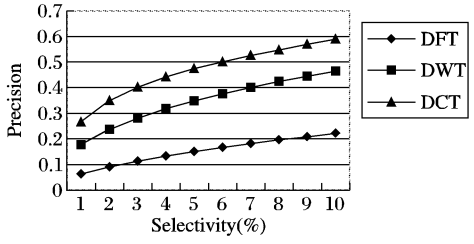

图 1 选取度变化时的准确率比较

实验对象是长度 128（默认值）的 6 625 个股票序列，目标是在 DFT，DWT 和 DCT 之间做准确率（precision）和运行时间比较。所谓准确率就是指在时间域具有资格的序列数与变换后具有资格的序列数的比率。第一个实验，固定特征空间的维数为 6，选取的阈值是使得查询的结果为整个数据库大小的 0.5% ~5%，这个比率也叫选取度。我们随机选取 100 个序列作为查询序列，然后取平均得到实验结果如图 1 和图 2。我们可以看到，随着选取度的增大，准确率也在上升，并且用 DCT 压缩数据后进行查询所得到的准确率要明显高于 DWT 和 DFT，这是因为 DCT 的能量集中性比 DWT 和 DFT 好。图 2表示范围查询的运行时间，DCT 略快于 DWT，因为 DCT 的准确率高，以至于在索引树中要查询的序列数要少，从而节省了查询时间。

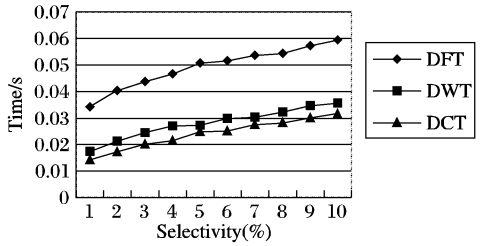

图 2 选取度变化时的运行时间比较

第二个实验，选取度固定为 6%，特征空间维数变化时三者之间的准确率和查询速度比较。实验结果分别如图 3 和图 4。图 3 显示出，随着维数的增加，准确率上升，但趋势变缓，这主要是因为大部分能量集中在最前面的几个系数，再增加一些系数对查询时准确率的影响不大。图 4 表示随着维数的增加，时间变化的趋势。当维数较低时，花费的时间随着维数的增加而迅速减少。但到达一定的程序，甚至会出现缓慢的上升趋势，这明显地体现在 DCT 和 DWT 中，而由于 DFT 的维数是其系数的 2 倍，并且能量的集中率较低，当到达一定的维数后也会出现上升的趋势，只是图中没显示出来。促使这种现象的原因是因维数上升增加的查询时间超过了因准确率上升而减少的那部分时间。

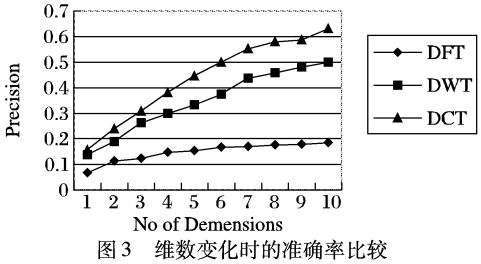

图 3 维数变化时的准确率比较

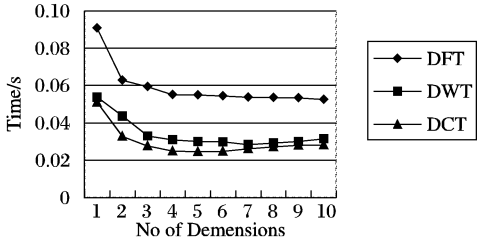

图 4 维数变化时的运行时间比较

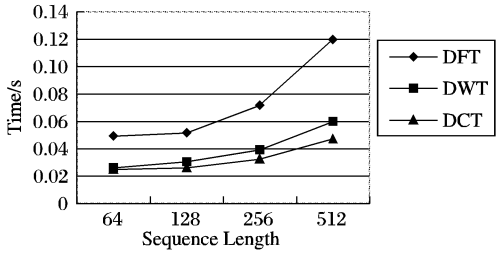

图 5 序列长度变化时的时间比较

第三个实验，考虑序列长度变化的情况，由于要与 DWT 进行比较，我们选取2^i的长度，并且固定选取度和抽取特征

（下转第 1244 页）

文献[6]中的算法得到的;c-opt 版本是用本文中的方法产生的。表格 2 中列出了将优化后的各版本的程序在具有 16 个 CPU 的 SMP 机器上运行的结果(单位 s)。从这些结果我们可以推断出:仅仅使用循环变换优化 l-opt 的经典局部优化策略效果可能不好,而基于数据变换 d-opt 的性能提高了许多,我们的 c-opt 方法相对于 col 方法减少了 42% 的执行时间,可以看出 c-opt 版本的程序运行效率最高。

表 1 实验中用到的程序

程序	程序来源	循环	数组个数和维数
mxm	Spec92	3 个	3,2-D
adi	livermore	5 个	3,2-D;3,1-D
emit	Spec92	2 个	3,3-D;10,1-D
Syr2k	BLAS	2 个	3,2-D
gfunp	Homepack	3 个	1,1-D;5,2-D
trans	Nwchem	3 个	2,2-D

表 2 实验结果

程序	col	row	l-opt	d-opt	c-opt
mxm	220.0	181.5	100.0	112.6	79.8
adi	144.1	134.9	22.8	45.6	22.8
emit	88.64	176.5	100.0	47.1	100.0
Syr2k	215.3	86.3	52.0	77.4	52.0
gfunp	86.05	128.4	73.3	68.0	46.9
trans	181.9	100.0	100.0	48.2	48.2
Average	155.6	134.6	74.68	66.48	58.28

3 结语

本文阐述了如何联合使用循环变换和数据变换两种编译优化技术来改善核外计算程序的性能。在此过程中运用了大量的线性代数知识,具有理论上的可靠性。实验结果证实了本文提出的方法的优点。

参考文献:

[1] 唐剑琪, 方滨兴, 胡铭曾. 面向工作站机群的核外计算模型及实现[J]. 高技术通讯, 2003, 13(7): 20 - 24.

[2] WOLF M, LAM M. A data locality optimizing algorithm[J]. Proceedings of the SIGPLAN 91 Conference on Programming Language Design and Implementation, Toronto, Canada, 1991. 30 - 44.

[3] LI W. Compiling for NUMA parallel machines[D]. Cornell University, Ithaca, New York, 1993.

[4] KANDEMIR M, CHOUDHARY A, RAMANUJAM J, *et al*. A matrix-based approach to the global locality optimization problem[A]. Proceeding PACT'98[C]. 1998.

[5] BIK A, WIJSHOFF H. On a completion method for unimodular matrices. 94 - 14[R]. Leiden University, 1994.

[6] O'BOYLE M, KNIJNENBURG P. Non - singular data transforma - tions: Definition, validity, applications[A]. Proceedings of 6th Workshop on Compilers for Parallel Computers. Aachen Germany, 1996. 287 - 297.

(上接第 1234 页)

数分别为 6% 和 6。由前面的分析很容易得到,序列长度的增加,准确率会下降。而运行时间呈相反的趋势。这里仅给出运行时间的比较结果(图 5)。

3.2 近邻查询

实现前面的近邻算法,随机抽取 100 个序列进行实验后取平均得到图 6 的结果。

随着查询数目 m 的增加,所得到的阈值也会增加,但三者所得到的阈值相同。同时,查询数目的增加也会使时间增加。从图 6 看到,用 DCT 所花费的时间要少于 DWT,更少于 DFT 所花费的时间。

前面所有的实验都是使用现实生活中的数据。我们也对模拟数据进行了实验,结果与前面的相似,由于篇幅的原因,这里不再一一列出。

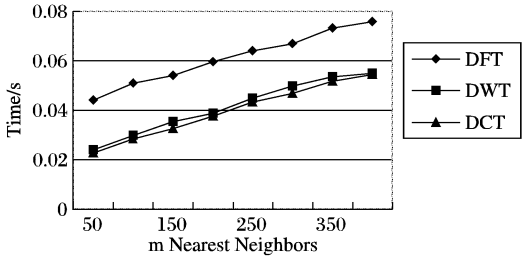

图 6 近邻查询的运行时间比较

4 结语

文中提出了一种更有效的时间序列相似性搜索方法,通过离散余弦变换进行维归约,并在其特征空间上建立空间索引结构来进行相似性搜索。给出了范围查询和动态更新 ε 的近邻查询的算法,并从理论上对算法进行了分析。我们用提出的方法与基于 DFT 和 DWT 的搜索方法进行了详细的比较,实验结果显示出该方法在准确率、时间上都具有很大的优越性。

下一步,我们将提出的方法推广到多变量的时间序列,并进一步应用于序列的分类、聚类等时序数据挖掘中。

参考文献:

[1] AGRAWAL R, FALOUTSOS C, SWAMI A. Efficient similarity search in sequence databases [A]. Proceedings of the 4th Int'l Conference on Foundations of Data Organization and Algorithms [C]. New York: Springer, 1993. 69 - 84.

[2] FALOUTSOS C, RANGANATHAN M, MANOLOPOULOS Y. Fast subsequence matching in time-series databases [A]. Proceedings of the 1994 ACM SIGMOD International Conference on Management of Data [C]. Minneapolis: ACM press, 1994. 419 - 429.

[3] CHAN KP, FU AWC. Efficient time series matching by wavelets [A]. Proceedings of the 15th International Conference on Data Engineering[C]. IEEE, 1999. 126 - 133.

[4] POPIVANOV I, MILLER RJ. Similarity search over time-series data using wavelets [A]. Proceeding of the 18th International Conference on Data Engineering [C]. Washington DC: IEEE Computer Society, 2002. 212 - 221.

[5] VLACHOS M, LIN J, KEOGH E, *et al*. A wavelet-based anytime algorithm for k-means clustering of times series [A]. The 3 SIAM International Conference on Data Mining [C]. San Francisco, CA, 2003.

[6] WU YL, AGRAWAL D, ABBADI AE. A comparison of DFT and DWT based similarity search in time-series databases [A]. Proceedings of the 9th International Conference on Information and Knowledge Management [C]. McLean VA: ACM Press, 2000. 488 - 495.

[7] 吴乐南. 数据压缩 [M]. 北京: 电子工业出版社, 2000.

[8] (加) HAN JW. 数据挖掘: 概念与技术(英文版) [M]. 第 2 版. 北京: 机械工业出版社, 2006.

[9] CHEUNG KL, FU A. Enhanced nearest neighbor search on the R-tree[J]. ACM SIGMOD Record, 1998, 27(3): 16 - 21.