基于 MR 框架的不确定时间序列相似性计算方法

李成为,王 屿,郑迪威
(南京航空航天大学 计算机科学与技术学院,江苏 南京 211106)

摘 要:为了更好地适应大规模不确定时间序列数据的相似性耗时多、计算效率低的问题,基于传统的动态时间规整(DTW)相似性计算算法,在 FastDTW 算法已经进行粗细粒度化剪枝节省部分运算时间的情况下,通过融入 MapReduce 计算框架,提出一种不确定时间序列的相似性计算算法 MR-FastDTW。该算法在 FastDTW 算法执行递归返回阶段时需要计算的递归矩阵,用 MapReduce 的思想分成多个子矩阵。同时对求得的路径周围的子矩阵进行并行计算,最后汇总范围内子矩阵的结果,得出最终路径。实验结果表明,MR-FastDTW 算法解决了 FastDTW 在递归返回段执行到一定程度后计算量大的问题,提高了计算速度和计算准确性;相比于经典的 DTW 及其改进的 FastDTW 算法,具有更高的效率。

关键词:不确定时间序列;相似性计算;动态时间规整;FastDTW;MapReduce
中图分类号:TP301 **文献标识码**:A **文章编号**:1673-629X(2018)10-0027-05
doi:10.3969/j.issn.1673-629X.2018.10.006

A Similarity Computation Method of Uncertain Time Series Based on MR Framework

LI Cheng-wei, WANG Yu, ZHENG Di-wei
(School of Science and Technology, Nanjing University of Aeronautics and Astronautics, Nanjing 211106, China)

Abstract: In order to better adapt to high time-consuming and inefficient computation of large-scale and indefinite time-series data, based on the traditional dynamic time warping (DTW) similarity algorithm, in the case of saving some computing time by FastDTW which has implemented the coarse-grained pruning, we propose a MR-FastDTW algorithm to calculate the similarity of uncertain time series by integrating MapReduce framework. The algorithm needs to calculate the recursive matrix when the FastDTW algorithm performs the recursive return phase, which is divided into several sub-matrices by the idea of MapReduce. At the same time, the sub-matrices around the obtained path are calculated in parallel. Finally, the results of sub-matrices in the range are summarized and the final path is obtained. Experimental shows that the MR-FastDTW algorithm can solve the problem of large amount of computation when FastDTW is executed to a certain extent in the recursion return segment, and improve the calculation speed and accuracy. Compared with the classic DTW and its improved FastDTW algorithm, it has higher efficiency.

Key words: uncertain time series; similarity calculation; dynamic time warping; FastDTW; MapReduce

0 引 言

不确定时间序列(uncertain time series)是按照时间戳先后顺序排列的记录值的序列,在每个时间戳的序列值都是未知的或者不可预料的。不确定时间序列数据广泛应用于定位服务[1]、医疗数据分析[2]、时序数据库[3]、无线传感器网络[4]等等。不确定性的产生是由于数据采集和环境方面的误差[5]、使用了预测技术[6-7],以及对于测量属性的多次读取数据[8]等因素。

在不确定时间序列数据处理的诸多任务和问题中,不确定时间序列相似性计算是其很重要的一个部分[9]。对于不确定时间序列相似性计算方法,主要从两个方向进行研究:一个是参考最初为确定性时间序列数据提出的相似性计算算法来进行改造[10-11];一个是根据不确定时间序列的特性,融合传统的确定性时间序列相似性算法思想,提出专门适用于不确定时间序列的相似性算法[6-7,12-14]。文中主要对第二个方向进行研究。

在传统的确定性时间序列相似性计算方法中,

收稿日期:2017-11-07 **修回日期**:2018-03-16 **网络出版时间**:2018-05-25
基金项目:国家自然科学基金(61370075)
作者简介:李成为(1992-),男,硕士研究生,研究方向为数据与知识工程。

DTW 由于不要求两个序列等长,并且两个序列求差值的点可以一对多或多对一,广泛用于计算各种情况的时间序列相似性。文献[8]针对经典 DTW 算法具有较高计算时间和空间损耗的问题,提出解决流检测问题的 ShortestDTW 算法;文献[14]通过多次在一些超平面上的投影来组织序列,提出了 FastDTW 算法,但该算法需要预先知道不同对象之间的距离,属于一种粗犷的过滤方式,会带来许多不相似的噪音;文献[15]基于 DTW 原理提出了 TD 算法,对时间跨度较大且体现一个连续、完整过程的时间序列数和时间跨度较小、体现状态点的时间序列数据具有一定的计算效果;文献[16]提出一种将特征提取和降维进行融合的 PLA-SDTW 算法,可解决高维时间复杂度的问题。

针对时间序列数据具有较大数据量的特点[17],近年来已有研究工作尝试使用 MapReduce 计算框架[18]计算时间序列的相似性。文中基于传统 DTW 算法,通过融入 MR 计算框架,提出一种不确定时间序列的相似性计算算法。该算法在不确定时间序列计算规模大时,并行计算每个子矩阵的动态时间规整距离,在获得与原 DTW 同等的匹配效果的同时,可使计算时间复杂度大为缩小。

1 基础知识

1.1 时间序列相似性计算

给定一个收集好的时间序列 $C = \{S_1, S_2, \cdots, S_N\}$,其中 N 表示时间序列的数量,查询函数定义为:$RQ(Q, C, \varepsilon) = \{S \mid S \in C \mid \text{distance}(Q, S) \leqslant \varepsilon\}$,其中 ε 是使用者提供的距离阈值。

(1)DTW 距离。

给定两个时间序列 Q 和 C,它们的长度分别为 n 和 m,将这两个时间序列记为:$Q = < x_1, x_2, \cdots, x_n >$,$C = < y_1, y_2, \cdots, y_n >$。它们之间的 DTW 距离的递归定义为:

$$\text{DTW}^2(\varphi, \varphi) = 0$$

$$\text{DTW}^2(Q, \varphi) = \text{DTW}^2(\varphi, C) = \infty$$

$$\text{DTW}^2(Q, C) = \text{dist}^2(\text{First}(Q), \text{First}(C)) + \begin{cases} \text{DTW}^2(Q, \text{Rest}(C)) \\ \text{DTW}^2(\text{Rest}(Q), C) \\ \text{DTW}^2(\text{Rest}(Q), \text{Rest}(C)) \end{cases}$$

其中,φ 表示空时间序列;$\text{First}(Q)$ 表示时间序列 Q 的第一个元素;$\text{Rest}(Q)$ 表示时间序列 Q 除第一个元素外其他元素组成的子序列;$\text{dist}^2(x, y)$ 为两点 x 和 y 的距离,通常采用欧氏距离进行计算。DTW 的计算复杂度为 $O(nm)$,计算代价较高,所以有研究人员对距离进行了变形处理。

(2)FastDTW 距离。

先将两个时间序列粗粒度化,在递归到最底层后寻找最短 DTW 路径,然后每次细粒度扩展 r 个单位(r 为半径),即将路径及其周围的点逐步细粒度化,并再次寻找最短 DTW 路径,最终求出原始序列间的 DTW 距离。

FastDTW 的细粒度化计算过程如图 1 所示。

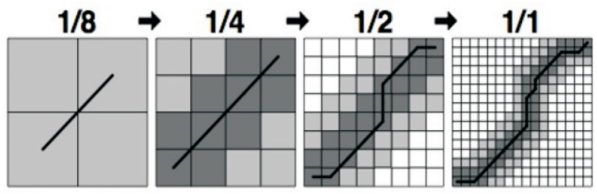

图 1 FastDTW 的细粒度化计算过程

第一幅图表示在递归最底层阶段执行 DTW 算法。第二幅图表示在递归返回阶段,将第一幅图求得的路径经过的方格进行细分,并且向左右,上下,以及左上右下的斜对角方向扩展 r 个单位,重新执行 DTW 算法得到的新路径。以此类推,直到最后求得最终的路径。

1.2 不确定时间序列及其期望距离

不确定时间序列的每个元素 x 都可以表示为 $x = r(x) + \varepsilon(x)$,其中 $r(x)$ 表示该元素的真实值,$\varepsilon(x)$ 表示该元素的误差,是服从某一连续型分布函数或某离散型分布的随机变量。不确定时间序列 T 定义为一系列随机值的集合 $T = < t_1, t_2, \cdots, t_n >$,其中 t_i 是在时间戳 i 对实数值的随机变量建模。

期望距离[19]可以用来计算两个不确定时间序列数据之间的距离,该算法是用所有可能出现的计算结果的均值加上误差值来代表这两个不确定时间序列之间的距离。期望距离的具体定义如下:两个时间序列 X, Y(其中至少有一个是不确定时间序列)的概率分布为 $f(x)$、$f(y)$,则它们的期望距离为:

$$E(X, Y) = \int_{-\infty}^{+\infty}\int_{-\infty}^{+\infty} \text{pointsdis}(x, y) f_{X,Y}(x, y)\,\mathrm{d}x\mathrm{d}y$$

其中,$\text{pointsdis}(x, y)$ 可以用 $\| x - y \|^2$ 表示,即 $x^2 + y^2 - 2|xy|$,可以推导出:

$$E(X, Y) = E(X^2) + E(Y^2) - 2E(X)E(Y) = (E(X) - E(Y))^2 + \text{Var}(X) + \text{Var}(Y)$$

上述公式即表明了期望距离可以用时间序列中各个时间点之间的距离的期望和方差来表示,由此可以大大减少不确定时间序列之间点与点之间距离的计算量。

2 MR-FastDTW 算法

FastDTW 算法通过粗细粒度化的变化,在计算路径时,首先计算递归最底层的路径,并在返回阶段,围

绕着上一阶段得出的路径拓展相应范围的计算矩阵,减少 DTW 暴力求解所造成的无用计算带来的损耗。

算法 1 描述了 FastDTW 的计算过程:

算法 1:FastDTW 算法

Input:长度分别为 $|X|$ 和 $|Y|$ 的两段时间序列 X 和 Y;

阈值 Radius//最粗分辨率的最小值

Output:时间序列 X 和 Y 之间的最小 DTW 距离;X 与 Y 之间的规整路径

1. Integer minTSsize=radius+2
2. IF($|X|\leqslant$minTSsize OR $|Y|\leqslant$minTSsize)
3. RETURN DTW(X , Y)
4. ELSE
5. TimeSeries shrunkX=X. reduceByHalf( )
6. TimeSeries shrunkY=Y. reduceByHalf( )
7. WarpPath lowResPath = FastDTW (shrunkX, shrunkY, radius)
8. SearchWindow window = Exp[andeResWindow (lowResPath, X , Y ,radius)]
9. RETURN DTW(X , Y ,window)

针对 FastDTW 计算方法在递归返回阶段执行到一定程度后,文中采用 MR 计算框架简化相似性计算,提出了 MR-FastDTW 算法,以提升运行速度。

对于一个给定的不确定时间序列,在 FastDTW 算法执到第 1 步和第 12 步时,MR-FastDTW 算法采用了期望距离来执行不确定性数据的相似性计算。在执行 MR 计算框架时,对于递归到一定程度后即第 12 步,对矩阵进行分块处理,分块好的矩阵进行 Map 处理,产生相应的<key,value>值对,然后用 Reduce 对这些生成的键值对进行处理,执行递归细粒度化,并在新的搜索矩阵上面执行上述步骤,以此重复,得到最后的结果。该算法的详细步骤如下:

1.输入采集到的序列 A、待检测的确定时间序列 B。

2:执行 FastDTW 算法,DTW 计算过程使用期望距离来计算。当粗粒度进行到第 n (n >3)次细粒度化时,把细粒度化好的矩阵放入 MR 计算框架中执行计算。

3:执行 MR 计算。

(1)由于 DTW 的计算矩阵是 $n*m$ 矩阵,将序列 X 划分为长度分别为[m/p]的 p 个子序列 $X_0,X_1,\cdots,X_{p-1}$,将序列 Y 划分为长度分别为[n/q]的 q 个子序列 $Y_0,Y_1,\cdots,Y_{q-1}$ 。于是就构造了 $p*q$ 个子矩阵 $\boldsymbol{M}_{f*g}$, $f\in[1,p]$, $g\in[1,q]$。每个子矩阵的规模为[$m*n$]/[$p*q$]。

(2)每个子矩阵求的路径作为 key 值,编号作为 value 值进行排序。

(3)排序好后的值传入 Reduce 部分,进行路径汇总筛选,并规约在一起得出总的动态规约路径。

算法 2 描述了 MR-DTW 的计算过程:

算法 2:MR-DTW 算法

Input:时间序列 X , Y ,规约窗口 window, p , q

Output:MR - DTW (X , Y , window)//需要加上 window 参数

1. INTEGER　n/q , m/p
2. DTW(n_2 , m_2)
3. MAP://MAP 阶段
4. E ((1,2, $\cdots$,n/q), m/p)→HDFS // D_{1*1} 最后一行的值
5. E (n/q ,(1,2, $\cdots$,m/p))→HDFS→HDFS// D_{1*1} 最后一列
6. HDFS[E ((1,2, $\cdots$,n/q), m/p)]→Matrix((n/q , $n/q+1$, $\cdots$, $n/q*2$), m/p)
7. $D(n/q*2,m/p)$
8. //读取 HDFS 中的 D_{1*1} 最后一行,存入 D_{2*1} 的第 0 行
9. HDFS [E (n/q , (1,2,$\cdots$,m/p))] → Matrix (n/q , (m/p , $m/p+1$, $\cdots$, $m/p*2$))
10. $D(n/q*2,m/p*2)$
11. //读取 HDFS 中的 D_{1*1} 最后一列 ,存入 D_{1*2} 的第 0 列
12. E (n/q ,(m/p , $m/p+1$, $\cdots$, $m/p*2$))→HDFS
13. E ((n/q , $n/q+1$, $\cdots$, $n/q*2$), m/p)→HDFS
14. //将第 D_{1*2} 的最后一列和 D_{1*2} 的最后一行存到 HDFS
15. //…并行计算子矩阵,中间重复步骤
16. HDFS[E (($n*(q-2)/q$, $n*(q-2)/q+1$, $n(q-1)/q$), m)]→Matrix(($n(q-1)/q$, $n(q-1)/q+1$, $\cdots$, n), m)
17. HDFS[$E(n,(m*(p-2)/p,m*(p-2)/p+1,\cdots,m(p-1)/p))$]→Matrix(n,($m*(p-1)/p$,$m*(p-1)/p+1$,$\cdots$,m))
18. $D(n,m)$
19. //获取子矩阵 $\boldsymbol{D}_{(f-1)*g}$ 的最后一行以及矩阵 $\boldsymbol{D}_{f*(g-1)}$ 的最后一列,存入到 D_{f*g} 的第 0 行第 0 列,计算 D_{f*g}
20. SORT: //SORT 阶段
21. Sort($D(n/q,m/p)$,$\cdots$,$D(n,m)$)//对 $p*q$ 个矩阵的值排序
22. REDUCE://REDUCE 阶段
23. Combime(key) according to the sort result
24. //根据 sort 结果,筛选链接每个小矩阵的值
25. Return DTW (n,m)

把 MP-DTW 算法与 FastDTW 算法相结合,得到了基于 MR 计算框架的 MR-FastDTW 算法。算法 3 描述了 MR-FastDTW 的计算过程:

算法 3:MR-FastDTW 算法

Input:长度分别为 $|X|$ 和 $|Y|$ 的两段不确定时间序列 X 和 Y ,FastDTW 扩展半径 r ,起始执行并行算法阈值 num

Output:MR-FastDTW 动态弯曲矩阵距离

1. Integer minTSsize=radius+2
2. IF($|X|\leqslant$minTSsize OR $|Y|\leqslant$minTSsize)

3. RETURN DTW(X , Y)

4. ELSE

5. TimeSeries shrunkX = X. reduceByHalf( )

6. TimeSeries shrunkY = Y. reduceByHalf( )

7. WarpPath lowResPath = FastDTW (shrunkX, shrunkY, radius)

8. SearchWindow window = Exp [andeResWindow (lowResPath, X , Y , radius)]

9. IF NUM< N

10. RETURN DTW(X , Y , window)

11. ELSE

12. RETURN MR-DTW(X , Y , window)

3 实验验证

3.1 测试数据集的选择

UCR 数据库是时间序列分析研究中一种被广泛应用的实验测试数据集,文中采用 UCR 数据库中的数据作为输入数据,采取人工加噪的方法,即对时间序列的每个元素加入期望为 0、标准差为 $\varepsilon(x)$ 的误差函数,定义 $0.1\sigma \sim 1.5\sigma$ 作为误差函数的范围,以表达不确定性,其中 σ 表示该数据集自身的标准方差。

从 UCR 数据集中随机抽取 8 组数据作为实验测试数据集。8 组数据的具体构成见表 1。

表 1 由 UCR 数据集构建的不确定时间序列

UCR 数据集	时间序列长度	UCR 数据集	时间序列长度
Synthetic Control	60	Gun-Point	150
OSU Leaf	427	Lightning-2	637
WormsTwoClass	900	CinC_ECG-torso	1 639
InlineSkate	1 882	HandOutLines	2 709

3.2 算法精确度和复杂度的比较

实验采用 4 台计算机通过路由器组建 Hadoop 集群,每台计算机的内存为 4 GB,处理器为 Intel i5-6500,操作系统为 Win7。一台作为主节点,三台作为数据节点。使用表 1 中的数据集人工合成不确定时间序列,对 MR-FastDTW 算法与传统的 DTW 算法和 FastDTW 算法进行比较。

(1)精确度对比。

精确度是对比不同算法之间准确性的标准,即不同算法检测的结果与实际不确定时间序列的结果相比较的准确程度。由于不确定数据存在误差,所以对每组实验数据测十次,并取均值。再用不同的算法单次读取该数据,并与均值进行比较。DTW 算法未对数据进行任何限制处理,所以得到的结果精确度最高;FastDTW 算法采用了粒度化的方法对数据集进行了限制处理,精确度下降,但仍有较高的准确度;MR-FastDTW 算法的精确度与 FastDTW 算法的精确度基本一致,也表现出较高的准确度。精确度比较如图 2 所示。

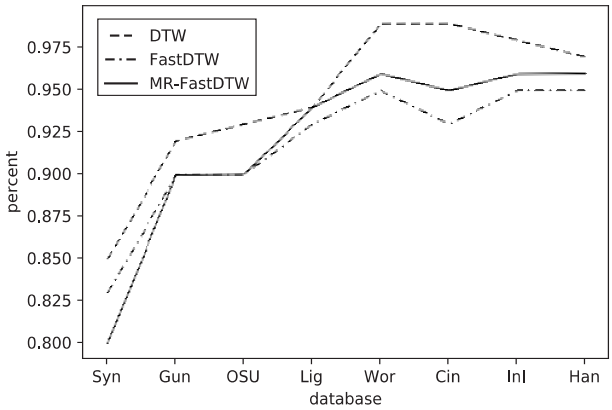

图 2 相似性算法的精确度比较

(2)时间复杂度对比。

时间复杂度主要是不同算法执行同组数据所消耗的时间。时间复杂度的比较如图 3 所示。

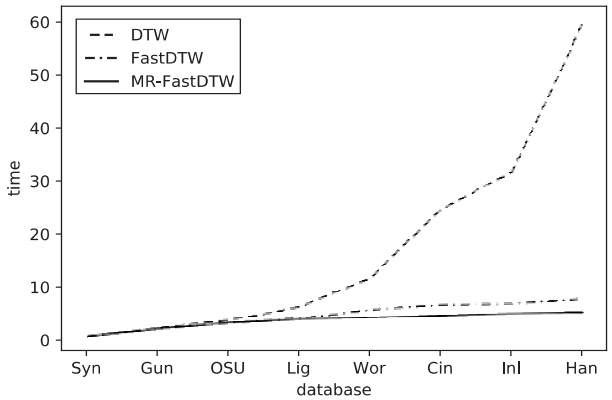

图 3 相似性算法的时间复杂度比较

从图 3 可以看出,当数据量很小时,三种算法没有太大的差异。当参与计算的数据量逐步增加时,DTW 算法其时间消耗随数据规模的增大开始非线性增长,FastDTW 算法和 MR-FastDTW 算法的时间消耗表现平稳。当数据量规模超过 900 时,DTW 算法时间消耗会急剧增长,FastDTW 算法由于是串行计算,时间消耗有所增大,而 MR-FastDTW 算法由于是并行处理,时间消耗最小。

4 结束语

提出了基于 MR 计算框架的 MR-FastDTW 算法,解决了 FastDTW 在递归返回段执行到一定程度后计算量大的问题,在提高计算速度的同时,提高了计算准确性。

需要指出的是,目前 MR-FastDTW 算法在执行过程中需要使用阈值,它控制着递归返回段进行到何种程度时再进行 MapReduce 计算,因此阈值是 MR-FastDTW 算法实现的一个关键点。但是,阈值的取值目前

还没有一个严格的范围标准,当前只是大概取程序在递归返回阶段执行到三分之一时刻的值,这只是经验值,缺乏严格的论证和理论依据。此外,该算法在执行矩阵拆分以及合并计算值时,操作还稍显繁琐。所有这些都将是未来工作中需要改进的地方。

参考文献:

[1] CHENG R, KALASHNIKOV D V, PRABHAKAR S. Querying imprecise data in moving object environments[J]. IEEE Transactions on Knowledge & Data Engineering, 2004, 16(9):1112-1127.

[2] LIAN Xiang, CHEN Lei, YU J X. Pattern Matching over cloaked time series[C]//IEEE international conference on data engineering. Cancun, Mexico: IEEE, 2008: 1462-1464.

[3] SATHE S, JEUNG H, ABERER K. Creating probabilistic databases from imprecise time-series data[C]//Proceedings of the 27th international conference on data engineering. Hannover, Germany: IEEE, 2011: 327-338.

[4] DALLACHIESA M, JACQUES-SILVA G, GEDIK B, et al. Sliding windows over uncertain data streams[J]. Knowledge & Information Systems, 2015, 45(1): 1-32.

[5] CERIOTTI M, CORRÀ M, D'ORAZIO L, et al. Is there light at the ends of the tunnel? Wireless sensor networks for adaptive lighting in road tunnels[C]//International conference on information processing in sensor networks. Chicago, IL, USA: IEEE, 2011: 187-198.

[6] DALLACHIESA M, NUSHI B, MIRYLENKA K, et al. Uncertain time-series similarity: return to the basics[J]. Proceedings of the VLDB Endowment, 2012, 5(11): 1662-1673.

[7] ORANG M, SHIRI N. A probabilistic approach to correlation queries in uncertain time series data[C]/International conference on information and knowledge management. [s. l.]: ACM, 2012: 2229-2233.

[8] LI Guiling, WANG Yuanzhen, LI Min, et al. Similarity match in time series streams under dynamic time warping distance [C]//International conference on computer science and software engineering. Hubei, China: IEEE, 2008: 399-402.

[9] KEOGH E, KASETTY S. On the need for time series data mining benchmarks: a survey and empirical demonstration [J]. Data Mining and Knowledge Discovery, 2003, 7(4): 349-371.

[10] SARANGI S R, MURTHY K. DUST: a generalized notion of similarity between uncertain time series[C]//Proceedings of the 16th ACM SIGKDD international conference on Knowledge discovery and data mining. Washington, DC, USA: ACM, 2010: 383-392.

[11] ORANG M, SHIRI N. An experimental evaluation of similarity measures for uncertain time series[C]//Proceedings of the 18th international database engineering & applications symposium. Porto, Portugal: ACM, 2014: 261-264.

[12] WU W C H, YEH M Y, PEI J. Random error reduction in similarity search on time series: a statistical approach[C]// IEEE 28th international conference on data engineering. Washington, DC, USA: IEEE, 2012: 858-869.

[13] YEH M Y, WU K L, YU P S, et al. PROUD: a probabilistic approach to processing similarity queries over uncertain data streams[C]//International conference on extending database technology: advances in database technology. Saint Petersburg, Russia: ACM, 2009: 684-695.

[14] SALVADOR S, CHAN P. Toward accurate dynamic time warping in linear time and space[J]. Intelligent Data Analysis, 2007, 11(5): 561-580.

[15] 李正欣,张凤鸣,李克武. 基于 DTW 的多元时间序列模式匹配方法[J]. 模式识别与人工智能,2011,24(3):425-430.

[16] 沈静逸. 基于 DTW 和 LMNN 的多维时间序列相似性分析方法[D]. 杭州:浙江大学,2017.

[17] 程艳云,张守超,杨　杨. 基于大数据的时间序列预测研究与应用[J]. 计算机技术与发展,2016,26(6):175-178.

[18] 付　晨,钟　诚,叶　波. MapReduce 并行加速数据流多模式相似性搜索[J]. 计算机应用,2017,37(1):37-41.

[19] 王佳林,王　斌,杨晓春. 面向不确定时间序列的分类方法[J]. 计算机研究与发展,2011,48:31-39.