類似部分区間検索のためのタイムワーピング距離の下限値計算

A Query Adaptive Lower Bound Calculation for Subsequence Search under Time Warping Distance

石川 雅弘 ♥ 吉川 昂伯 ♦
陳 漢雄 ♠ 古瀬 一隆 ♣
大保 信夫 †

Masahiro ISHIKAWA Takanori YOSHIKAWA
Hanxiong CHEN Kazutaka FURUSE
Nobuo OHBO

音声データ，株価などの経済データ，あるいは気温や磁気変動などの自然観測データ等，様々な分野で時系列データが大量に蓄積されるようになってきた．蓄積されたデータを活用するための基本技術の一つとして時系列データに対する類似検索の研究が行なわれているが，同程度の長さの時系列データを多数集めたデータベースを検索対象とし，問合わせとして与えられた時系列データに全長が類似したデータを検索するのが一般的である．しかし，例えば過去の株価の動きの中から最近数週間の値動きに類似した期間を検索するような，長大な時系列データから問合せ時系列に類似した部分区間を検索する研究は少ない．

本稿では，タイムワーピング距離に基づいた類似区間検索を効率的に行なうための，問合せに適応した下限距離計算法を提案し，実験によりその有効性を示す．

In recent years, huge amount of time-series data is accumulated in various domains. To utilize them, similarity search techniques are heavily studied. However, in the most studies, target database consists of large amount of data whose length are almost the same and a query is a time-series data of almost the same length. We focus on the case where the target database is a single very long time-series, and the length of a query is much shorter than that of the database. In this paper, we propose a query-adaptive lower-bound calculation for subsequence search under time-warping distance, and show its efficiency by experimental results.

♥ 正会員 つくば国際大学産業情報学科 mi@tius.ac.jp
♦ 筑波大学大学院システム情報工学研究科 yosikawa@dblab.is.tsukuba.ac.jp
♠ 正会員 筑波大学大学院システム情報工学研究科 chx@cc.tsukuba.ac.jp
♣ 正会員 筑波大学大学院システム情報工学研究科 furuse@cc.tsukuba.ac.jp
† 筑波大学大学院システム情報工学研究科 ohbo@cs.tsukuba.ac.jp

1. はじめに

近年，音声データ，株価データ，気温や磁気などの自然観測データ等，様々な分野において時系列データが大量に蓄積されるようになってきた．ここで時系列データとは，気温など経時変化する値を観測時間順に並べて得られる実数値のシーケンスである．蓄積されたデータを有効に活用するため，時系列データの分類やクラスタリングなどの研究が行なわれているが [1][2]，基本技術の一つとして重要なのが類似検索である．時系列データの類似検索に用いる距離尺度に適したものとして動的タイムワーピング (DTW: Dynamic Time Warping) 距離が用いられるようになってきたが [4][5][6]，計算コストが高いという問題があり，効率的な検索のためには距離計算コストの削減が重要である．また，時系列データに対する類似検索の研究では，同程度の長さの時系列データを多数集めたデータベースを検索対象とし，問合せ時系列データと全長が類似したデータを検索するものが一般的であるが [7][5][8]，本稿で提案する手法が想定するのはデータベース時系列が問合せ時系列に比べ大幅に長い場合であり，検索されるのはデータベース時系列中問合せに類似した部分区間である．

本稿では，このような検索を効率的に行うための DTW 距離の下限値計算の手法を示し，実験によりその有効性を示す．

2. 動的タイムワーピング距離 (DTW 距離)

2.1 DTW 距離の定義

時系列データには，観測間隔の違いや観測機器の故障によるデータの欠損などにより時間軸上のずれが生じる事があり，距離を求める際にはそれを考慮する必要がある．時間軸上のずれを吸収した距離として広く用いられているのが DTW 距離 [3] である．長さ m と n の時系列データ $\boldsymbol{x}, \boldsymbol{y}$ 間の DTW 距離 $D(\boldsymbol{x}, \boldsymbol{y})$ は下式で与えられる．

$$D(\boldsymbol{x}, \boldsymbol{y}) = D_0(x_1, y_1) + \min \left\{ \begin{array}{l} D(\boldsymbol{x}_{2:}, \boldsymbol{y}\ ) \\ D(\boldsymbol{x}\ , \boldsymbol{y}_{2:}) \\ D(\boldsymbol{x}_{2:}, \boldsymbol{y}_{2:}) \end{array} \right\} \quad (1)$$

$$D_0(x, y) = |x - y| \quad (2)$$

ここで $\boldsymbol{x}_{i:}$ は $\boldsymbol{x}$ の第 i 要素から始まる接尾辞となる部分時系列データであり，$\boldsymbol{x}_{i:j}$ は第 i 要素から始まり第 j 要素で終わる部分時系列データである．また，$D_0(x, y)$ は引数の差の絶対値の単調増加関数ならば任意であり，例えば $|x - y|^2$ が用いられる事もある．

2.2 DTW 距離の計算方法

DTW 距離は動的計画法を用いて求められる．その考え方を二つの時系列データ $\boldsymbol{x} = \langle x_1, x_2, x_3, x_4 \rangle = \langle 4, 3, 5, 3 \rangle$ と $\boldsymbol{y} = \langle y_1, y_2, y_3, y_4, y_5, y_6 \rangle = \langle 3, 4, 3, 5, 5, 4 \rangle$ を例に説明する．

まず，二つの時系列データの長さに合わせて 4×6 のテーブルを考える (図 1 参照)．行方向は$\boldsymbol{x}$に，列方向は$\boldsymbol{y}$に対応しており，最下行最左要素を $a_{1,1}$，最上行最右要素を $a_{6,4}$ とする．テーブルが空の状態から計算を開始し，全ての要素の値が求められた時点で計算が終了する．始めに $a_{1,1}$ に $D_0(x_1, y_1)$ を格納し，$a_{1,2}$〜$a_{1,4}$ には $a_{i,j} = a_{1,j-1} + D_0(x_1, y_j)$ を，$a_{2,1}$〜$a_{6,1}$ には $a_{i,j} = a_{i-1,1} + D_0(x_i, y_1)$ を格納する．左，下，左下の三つの接する要素がすでに求められた他の各要素には，下式

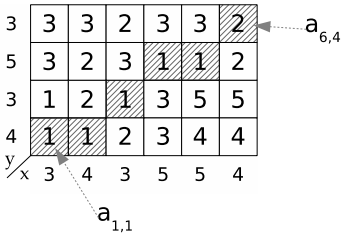

図 1　累積距離テーブル

Fig. 1　Accumulated Distance Table

に従って求めた値を格納する．

$$a_{i,j} = D_0(x_i, y_j) + \min \left\{ \begin{array}{l} a_{i-1,j} \\ a_{i,j-1} \\ a_{i-1,j-1} \end{array} \right\} \quad (3)$$

全ての要素を求めた時，最後に求められた最上最右要素 $a_{6,4}$ の値が DTW 距離となる．$a_{1,1}$ から $a_{6.4}$ までを，右上に向かって辿るパス (ワーピングパス) は多数存在するが，図中網掛けで示したのは $a_{6.4}$ から逆算して式 (3) 第二項で選択された要素を辿ったパスであり，DTW 距離を与える時の$\boldsymbol{x}$と$\boldsymbol{y}$の要素の対応を示している．この時の対応を図 2 に示す．$a_{1,1}$ から $a_{6,4}$ まで

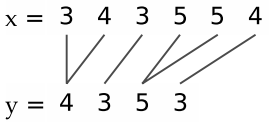

図 2　ワーピングパスが与える要素の対応

Fig. 2　Element Correspondence by the Warping Path

辿るワーピングパスは他にも存在するが，DTW 距離とは，全ての可能なパスのうち (すなわち全ての可能な対応付けのうち)，$a_{6,4}$ に最小の値を与えるパスから得られる距離のことである．

2.3　差分テーブルと DTW 距離

ここで差分テーブルと呼ぶテーブルを導入する．差分テーブルとは，累積距離テーブルと同サイズのテーブルであり，その (i, j) 要素 $t_{i,j}$ の値は $D_0(x_i, y_j)$ である．図 3 に例を示す．解

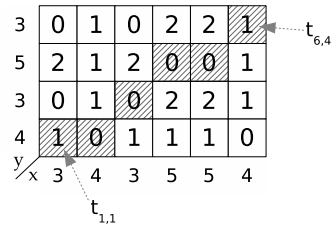

図 3　差分テーブル

Fig. 3　Difference Table

を与えるワーピングパスが与えられた時，パス上の要素の値を合計する事で DTW 距離を求める事ができる．図から，解を与えるパスは差分テーブル上では値の小さな要素上を通る傾向にある事が分かる．なぜなら，全ての可能なパスの中で値の合計が最小となるものが解を与えるからである．

3.　問合せに適応した DTW 距離の下限値

3.1　検索グラフ

図 4 は，図 3 の差分テーブルで値が 1 以下の要素のみを残したものである．この時 $t_{1,1}$ と $t_{6,4}$ を結ぶパスは二つしか残って

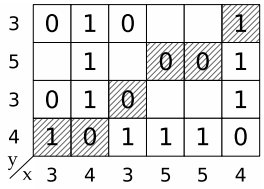

図 4　1 以下の要素のみを残した差分テーブル

Fig. 4　Difference Table with Smaller Values (less than two)

いないが，その一つが解を与えている事が分かる．常にこの例のように解となるパスが残る保証はないが，値の小さな要素だけに注目する事で可能なパスの数を削減し，低コストで距離を求められる可能性がある．

図 5 は，データベース$\boldsymbol{d}$と問合せ$\boldsymbol{q}$による差分テーブルであり，4 以下の要素にのみ値を記入してある．以下では値の記入さ

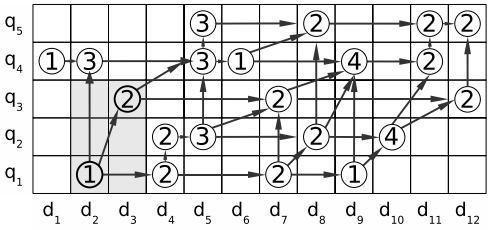

図 5　マークされた要素を結ぶ事で得られる検索グラフ

Fig. 5　Search Graph built from Marked Elements

れた要素をマークされた要素と呼ぶ．マークされた二つの要素 $t_{i,j}, t_{k,l}$ 間に以下の条件が成り立つ場合に有向辺 $\langle t_{i,j},\, t_{k,l} \rangle$ を張る事で得られるグラフを検索グラフと呼ぶ．

1. $i \leq k$ かつ $j \leq l$ (ただし $t_{i,j} \neq t_{k,l}$)．
2. $i \leq u$ かつ $j \leq v$ かつ $u \leq k$ かつ $v \leq l$ なる $t_{u,v}$ が存在しない (ただし $t_{i,j} \neq t_{u,v}$ かつ $t_{u,v} \neq t_{k,l}$)．

間に非マーク要素を挟む場合，有向辺上の始点から終点に至るパスの厳密な距離を求める事はできないが，その下限値は求める事ができる．

3.2　有向辺の下限距離

$t_{2,1}$ から $t_{3,3}$ に至るパスを考える．両要素により形成される矩形区間 (図 5 参照) でマークされているのは $t_{2,1}$ と $t_{3,,3}$ のみであり，それぞれの値は 1 と 2 であるため，パスがこの二つの要素を通る以上，最低でも値の合計は 3 である．ここで $t_{2,1}$ から $t_{3,3}$ に至るまでに取り得るパスは $[t_{2,1} \rightarrow t_{3,1} \rightarrow t_{3,2} \rightarrow t_{3,3}]$, $[t_{2,1} \rightarrow t_{3,2} \rightarrow t_{3,3}]$, $[t_{2,1} \rightarrow t_{2,2} \rightarrow t_{3,3}]$, $[t_{2,1} \rightarrow t_{2,2} \rightarrow t_{2,3} \rightarrow t_{3,3}]$ の 4 つである．どのパスでも q_2 に対応する要素を必ず一つは通過する必要があり，かつそれだけで $t_{3,3}$ に到達できる．従って距離の下限値は $7 (= 3 + 4)$ である．なぜなら，マーク要素の上限値は 4 であるため，非マーク要素の下限値は 4 だからである．このように，閾値以下の要素のみをマークする事でマーク要素間のパスの与える距離の下限値を求める事ができる．このようにして求めた下限値は，類似区間検索などの際に一定距離を越える区間をフィルタアウトする事で検索効率改善に利用できる．

以下では，一つの有向辺の下限距離の求め方について，矩形の形状ごとに解説する．

3.2.1 正方形の場合

二つのマーク要素をそれぞれ t_a, t_b とし，形成される矩形が正方形の場合の例を図 6 に示す．ここで要素 t_a, t_b の値をそれぞれ a, b とする．また，閾値は各 q_i 毎に異なる値とし，それぞれを ε_i で表す．

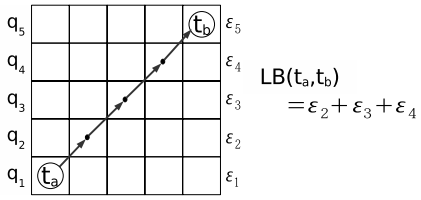

図 6 正方形の場合の下限距離計算
Fig. 6 Lower Bound (square case)

t_a と t_b を結ぶパスは，少なくとも q_2〜q_4 のそれぞれに対応する要素を一つずつ通過する必要があり，かつそれだけで t_b に達する事ができる．したがって，この有向辺の距離の下限は $LB(t_a, t_b) = \varepsilon_2 + \varepsilon_3 + \varepsilon_4$ である．

3.2.2 縦長の長方形の場合

図 7 に示したのは，縦長の長方形になる例である．この時，

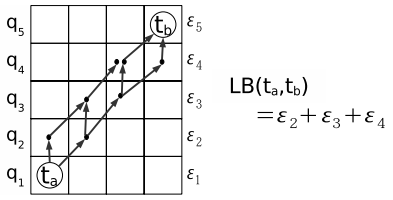

図 7 縦長の長方形の場合の下限距離計算
Fig. 7 Lower Bound (vertical case)

t_a から t_b に達するには，少なくとも q_2〜q_4 に対応する要素を一つずつ通過する必要があり，かつそれだけで t_b に達する事もできる．よって下限距離は $LB(t_a, t_b) = \varepsilon_2 + \varepsilon_3 + \varepsilon_4$ である．

3.2.3 横長の長方形の場合

図 8 は横長の長方形の場合である．t_a から t_b に達するに

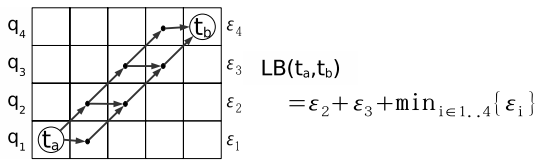

図 8 横長の場合の下限距離計算
Fig. 8 Lower Bound (horizontal case)

は，少なくとも q_2, q_3 に対応する要素を一つずつ通過する必要がある．t_b に到達するにはさらに，横長の分だけ q_1〜q_4 のどこかを余計に通過する必要がある．下限値を与えるのは，その中で最小閾値の要素を通過した時である．よって下限値は $LB(t_a, t_b) = \varepsilon_2 + \varepsilon_3 + min_{k \in 1..4}\{\varepsilon_k\}$ である．

3.2.4 一般の場合

以上の三つのケースをまとめると，一般に一つの有向辺の距離の下限値は下式で与えられる．ここで n, m はそれぞれ矩形の縦幅と横幅である．

$$LB(t_a, t_b) = \begin{cases} \sum_{i=2}^{n-1} \varepsilon_i & (n \geq m) \\ \sum_{i=2}^{n-1} \varepsilon_i + (m-n)\min\{\varepsilon_k\} & (n < m) \end{cases} \quad (4)$$

3.2.5 多段の場合

最後に，有向辺を複数辿った場合の下限距離について説明する．図 9 は，既に t_a から t_b に辿った後に，さらに t_c へ辿るところを表している．$t_a \rightarrow t_b$ 間，$t_b \rightarrow t_c$ 間の下限距離はそれぞ

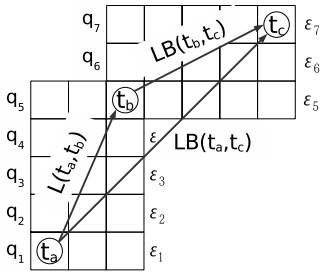

図 9 多段の場合の下限距離
Fig. 9 Lower Bound (Multiple Step)

れ既に $L(t_a, t_b)$，$LB(t_b, t_c)$ として求まっているとする．この時 $t_a \rightarrow t_c$ 間の下限距離 $L(t_a, t_c)$ は下式で与えられる．

$$L(t_a, t_c) = \min \left\{ \begin{array}{l} L(t_a, t_b) + b + LB(t_b, t_c) \\ LB(t_a, t_c) \end{array} \right\} \quad (5)$$

$LB(t_a, t_c)$ は，t_a, t_c 間にマーク要素がないと考えて求めた下限距離である．

4. 実験

4.1 類似区間検索の概要

検索グラフと下限距離による効率化を検証するために，類似検索実験を行なった．行なった類似検索は，問合せ q と整数 k が与えられた時，一本の長大な時系列データから k 個の最も q に類似した部分区間を検索する k-最近傍検索である．検索処理の概要は以下の通りである．

1. マーク付け $\cdots$ 値の小さな差分要素をマークする．
2. 解候補の選択 $\cdots$ k 個の部分区間を解候補として選択し，q とそれらとの実 DTW 距離を計算する．
3. 検索グラフの探索 $\cdots$ 検索グラフを探索しながら区間の下限距離を求める．その時，k 番目の解候補の実距離よりも下限距離が大きな区間をフィルタアウトする事で候補となる区間を絞り込む．

4.2 実験データ

実験には非周期性のランダムウォークデータと，周期性の Cylinder, Bell, Funnel[9] の合成データを用いた．図 10 に各データの例を示す．問合せは各データから切り取った部分時系

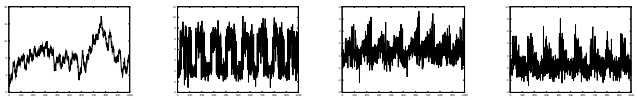

図 10 合成データの例
Fig. 10 Example of Synthetic Dataset

列とし，各データに対して 10-最近傍検索を 20 回行ない，その平均で評価した．

比較対象としたのは，問合せ $\boldsymbol{q}$ とデータベースの接尾辞 $\boldsymbol{d}_{1:}, \boldsymbol{d}_{2:}, \boldsymbol{d}_{3:}$ との DTW 距離を累積距離テーブルを用いて順次計算する方法である．$D(\boldsymbol{q}, \boldsymbol{d}_{i:})$ を計算する際には $\boldsymbol{d}_{i:k}$ なる

部分区間との距離も求まるため，全ての部分区間をチェックする事ができる．ただし，$\boldsymbol{d}_{i:}$ の末尾まで計算するのは無駄が大きいため，累積距離テーブルを埋める際に k 番目の候補解の最大距離よりも大きな値しか求まらなくなった時点で計算を打ち切った．

4.3 実験結果

図 11 は，データ長に対する実行時間の変化を示したものである．時間軸は対数目盛である．Cylinder, Bell, Funnel の三つのデータセットについては，それらの平均を cbf として示した．rw はランダムウォークデータである．マーク要素数は，ランダムウォークデータで全体の 5%，cbf データで 100% である．また正解の再現率は 100% である．

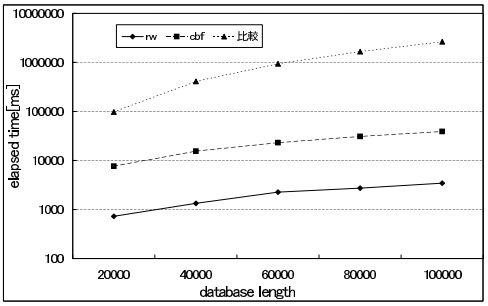

図 11 データベース長に対する検索時間の変化
Fig. 11 Database Length vs. Elapsed Time

グラフからデータベース長に関わらず提案手法の速度が比較手法を大幅に上回っている事がわかる．データベース長が 100,000 の時には，ランダムウォークデータでは約 800 倍の，cbf データでは約 70 倍の速度向上がみられた．両データ共，データベースが長くなるに従い速度向上の比率が高まる事も確認できる．

5. まとめと今後の課題

長大な時系列データを対象とし，問合せ時系列データに類似する部分区間を効率的に検索するために利用できる，タイムワーピング距離の下限値計算手法を示した．また，二種の合成データによる類似部分区間検索の実験により，提案手法の有効性を示した．

現在，提案した下限距離を用いてより高速に類似区間を検索するための，検索グラフの効率的な探索手法の考案と実験を進めている．また，過去に蓄積された時系列データを効率的に扱うには二次記憶上に格納したデータへの I/O コストも考慮に入れる必要があるため，そのための索引手法も検討中である．

距離尺度そのものについては，現在は広く利用されている DTW 距離 を採用しているが，外れ値などへの対応がより柔軟な距離尺度として最長共有部分列に基づく距離への対応も考えたい．

[文献]

[1] Chotirat Ratanamahatana and Eamonn Keogh, “Making Time-Series Classification More Accurate Using Learned”, In proc. of 4th SIAM International Conference on Data Mining, 2004.

[2] Li Junkui, Wang Yuanzhen and Li Xinping, ”LB_HUST: A Symmetrical Boundary Distance for Clustering Time Series”, In proc. of the 9th International Conference of Information Technology, 2006.

[3] Joseph B. Kruskal and Mark Liberman, “The Symmetric Time-Warping Problem: From Continuous to Discrete”, Time Warps, String Edits, and Macromolecules: The Theory and Practice of Sequence Comparison, pp. 125-161 Addison-Wesley, 1983

[4] Eamonn Keogh and Michael Pazzani, “Scaling up Dynamic Time Warping for Datamining Applications”, In proc. of 6th ACM SIGKDD Knowledge Discovery and Data Mining, 2000.

[5] Eamonn Keogh, “Exact Indexing of Dynamic Time Warping”, In proc. of 28th International Conference on VLDB, 2002.

[6] Sang-Wook Kim, Sanghyun Park, et al., “An Index-Based Approach for Similarity Search Supporting Time Warping in Large Sequence Databases”, In proc. of 17th ICDE, 2001.

[7] Yasushi Sakurai, Masatoshi Yoshikawa and Christos Faloutsos ”FTW: Fast Similarity Search under the Time Warping Distance”, In proc. of ACM PODS Conference 2005, pp.326-336, 2005.

[8] 大桃論, 陳漢雄, 古瀬一隆, 大保信夫, 「タイムワーピングに基づく時系列データの類似検索：次元縮小による効率化」, DBSJ Letters Vol.4, No.1, pp.1-4

[9] Hansheng Lei, Venu Govindaraju ”Regression Time Warping for Similarity Measure of Sequence”, In proceedings of the Fourth International Conference on Computer and Information Technology, 2004

石川 雅弘 Masahiro ISHIKAWA
2001 年 筑波大学大学院博士課程工学研究科了．現在つくば国際大学産業情報学科講師．データベースシステムに関する研究に従事．博士 (工学)．日本データベース学会正会員．

吉川 昂伯 Takanori YOSHIKAWA
2005 年 筑波大学情報学類卒業．現在同大大学院システム情報工学研究科在学中．

陳 漢雄 Hanxiong CHEN
1993 年 筑波大学大学院博士課程工学研究科了．現在同大大学院システム情報工学研究科講師．データベースシステムに関する研究に従事．工博．日本データベース学会正会員．

古瀬 一隆 Kazutaka FURUSE
1993 年 筑波大学大学院博士課程工学研究科了．現在同大大学院システム情報工学研究科講師．データベースシステムに関する研究に従事．工博．日本データベース学会正会員．

大保 信夫 Nobuo OHBO
1968 年 東京大学理学部卒．1970 年 同大大学院修士課程了．同年同大理学部助手．1980 年 筑波大学電子・情報工学系講師．1995 年 同大電子・情報工学系教授．現在に至る．データベースシステムに関する研究に従事．理博．