Vol. 45　No. SIG 4(TOD 21)　　情報処理学会論文誌：データベース　　Mar. 2004

ダイナミックタイムワーピングのための類似探索手法

櫻　井　保　志†　　吉　川　正　俊††

本論文ではダイナミックタイムワーピングのための類似探索手法を提案する．気象学，天体物理学，地質学，マルチメディア，経済など，時系列データは数多くの分野で用いられている．それらの中では，時系列データのシーケンスどうしを比較して，その類似性を評価することが頻繁に行われている．従来の研究では，シーケンスの距離基準として主にユークリッド距離が用いられていた．ユークリッド距離関数はシーケンスの各要素を独立して比較するため，長さの異なるシーケンスのペア，もしくはサンプリングレートの異なるシーケンスのペアの距離を比較することは難しい．さらにユークリッド距離関数は，シーケンスに少しでもアウトライアー（異常値）があると，それらに影響を受けることもある．これに対してダイナミックタイムワーピング（DTW; Dynamic Time Warping）は，各々のシーケンスの中で時間軸を柔軟に変化させて距離を算出することができる．このため，近年数多くのアプリケーションで DTW が用いられている．しかし，DTW は動的計画法に基づくアプローチで計算されるため，計算コストが高いことが問題となっている．そこで，DTW に基づく類似検索を高速化するために，DTW 距離を近似する距離関数，およびその関数を用いた索引手法，探索手法を提案する．提案手法は効率的に類似シーケンスを探索することができ，また近似距離関数を用いているものの，探索漏れがないことを保証する．すなわち，探索アルゴリズムはどのような問合せに対しても正確な答えを返す．具体的には，本論文ではまず，探索漏れが発生しないことを保証するための必要十分条件を提案する．そして，その必要十分条件を満足する DTW の近似距離関数について述べる．探索処理では，距離近似によって厳密な距離計算の回数が大幅に低減化する．これは高い探索性能につながる．実験では，既存手法と比べ最大で約 54 倍の性能向上を達成し，提案手法の優位性が明らかとなった．

A Similarity Search Method for Dynamic Time Warping

Yasushi Sakurai† and Masatoshi Yoshikawa††

Time-series data naturally arise in many application domains, such as meteorology, astrophysics, geology, multimedia, economics, etc. There is a frequent need to find similarities between such data sequences. Most of the earlier works on high-speed sequence matching are based on the Euclidean distance function. Since the Euclidean distance function makes all sequence elements independent of each other, this function cannot calculate the distance between sequences of which the length or sampling rate is different. In addition, this function may be sensitive to a few outliers. Recent applications have employed dynamic time warping to overcome these problems. The distance of dynamic time warping is calculated with a dynamic programming algorithm. Although dynamic time warping incurs a heavy CPU cost, it is robust versus noise and scaling for the time axis. To significantly reduce the CPU cost of DTW, we introduce here an approximation technique, an index structure, and a search method. Although our search method utilizes an approximation technique, it is guaranteed to return exact answers, that is, it gives the desired sequences without false dismissals. In particular, we first propose a necessary and sufficient condition for guaranteeing that a distance approximation causes no false dismissals in similarity query processing. We then present a new approximation technique for DTW that satisfies the necessary and sufficient condition. This method prunes a significant number of the search candidates, which leads to a direct reduction in the search cost. Experiments were conducted on real and synthetic sequence data sets. The results reveal that our method is significantly (up to 54 times) faster than the best existing method.

1. まえがき

1.1 ダイナミックタイムワーピングによる時系列データの類似検索

気象学，天体物理学，地質学，マルチメディア，経

† 日本電信電話株式会社 NTT サイバースペース研究所
NTT Cyber Space Laboratories, NTT Corporation
†† 名古屋大学
Nagoya University

済など，時系列データは数多くの分野で用いられている．それらの中では，時系列データのシーケンスどうしを比較して，その類似性を評価することが頻繁に行われている．多くの場合，アプリケーションが扱う時系列データの量は増加し続けているため，時系列データの類似探索を高速化することが求められている．さらに，これらのアプリケーションは，シーケンスのノイズや時間軸の縮尺に類似度が影響されないようなマッチングの仕組みを必要としている．

長いシーケンスの時系列データを扱う場合，もしくは大規模な時系列データベースを構築する場合，その類似探索には多大なコストを要するため，この探索コストを削減する索引手法や探索手法が数多く提案されている．従来，シーケンスマッチングを高速化するための手法は主としてユークリッド距離関数に基づくものが多かった．ユークリッド距離関数はシーケンスの各要素を独立して比較するため，長さの異なるシーケンスのペア，もしくはサンプリングレートの異なるシーケンスのペアの距離を比較することは難しい．さらにユークリッド距離関数は，シーケンスに少しでもアウトライアー（異常値）があると，それらに影響を受けることもある[2]．

近年のアプリケーションは，これらの問題を回避するためにダイナミックタイムワーピング（DTW; Dynamic Time Warping）[5),15)]を用いている[7),13),14)]．DTW は 2 つのシーケンスの距離を最小化するように時間軸を伸長させる変換処理であり，DTW の距離は動的計画法に基づいて計算される．DTW はシーケンスが長くなるに従い多大な計算コストを必要とするが，ユークリッド距離と違い，シーケンスのノイズや時間軸の縮尺に対して頑健である．時系列データアプリケーションにとって，DTW は利用者の意図をより忠実に反映するため有用である．

本研究における目的は，探索漏れを発生させずに DTW に基づく類似探索を高速化することである．これまで，主として音声認識[15)]やバイオインフォマティックス[13)]の分野などで様々な DTW のためのシーケンスマッチング手法が提案されてきた．しかし，これらの多くは精度を犠牲にして速度を向上させるものであった．Keogh は，全体的なパス制約（global constraint）[15)]の条件のもとで，探索処理を高速化する手法を提案している[9)]．全体的なパス制約は，動的計画法において使われている制約の 1 つであり，ワーピングパスがとりうる範囲を限定するものである．Zhu らは，文献 9) の手法の改良手法を提案している[20)]．これらの手法は，ワーピングパスを狭い範囲に限定すると効果を発揮するが，設定するワーピングの幅を広くするにつれて探索性能が低下する．これらの手法は有用であるが，最適なワーピングの幅はデータやアプリケーションに依存する．このため本研究では，狭いワーピング幅だけでなく広いワーピング幅でも DTW の探索を高速化することに焦点を合わせる．Kim らは，探索処理を高速化するために，DTW 距離の近似手法を探索処理に導入している[12)]．この手法は，制約条件を設けずに探索処理を実行することができ，また探索漏れも発生しない．さらに，距離近似のための計算コストは低い．しかし，粗い距離近似であるために，厳密な DTW 距離計算の回数が多くなり，依然として高い探索コストを示している．文献 9)，20) でも，DTW 距離の近似を行っているが，制約条件のワーピング幅を広くすると近似が粗くなり，同様の傾向になる．

1.2 提 案 内 容

本論文では，DTW に基づく類似検索を高速化するために，DTW 距離を近似する距離関数，およびその近似距離関数を用いた索引手法，探索手法を提案する．本論文では，主として以下の内容を提案している．

探索漏れが発生しないことを保証するための必要十分条件

既存の取り組み[3),6),8)〜10),16),20)]では，探索漏れが発生しないことを保証するために下界（lower bound）の性質をとり入れてきた．しかし，この性質は十分条件であるものの，必要条件ではない．本論文では，必要十分条件となる新たな性質を提案する．

厳密な距離が類似距離以下であるときは，
必ずその近似距離も類似距離以下である．

ここで類似距離とは，範囲問合せでは探索範囲を意味する．k 近傍問合せでは，処理を実行している間，候補 k 近傍の厳密な距離をつねに保持している．k 近傍問合せにおいて，類似距離とはその k 近傍距離を意味する．この必要十分条件は，時系列データの類似問合せのためだけのものではなく，多次元データ，文字列，XML など距離近似を行う様々な処理に適用することができる．

DTW のための探索手法

本論文で提案する手法は，下界の性質を持たない．しかし，必要十分条件を満たすために探索漏れがないことを保証する．探索手法は以下のアイデアに基づいている．

(1) 可能な限り低い計算コストで，探索処理に関係のない不必要なワーピングパスを排除する．ワーピングパスのフィルタリングには，k 近傍問合せにおける探索処理途中の k 近傍距離（範

囲問合せにおける探索範囲）を活用する．

(2) 探索処理の中で距離近似の精度を変化させる．まったく類似していないシーケンスのマッチングには粗い近似を，類似したシーケンスには精密な近似を行う．

一般的に，時系列データの検索手法は，長さの異なるシーケンスを扱えることが望ましい．文献 9)，20) の手法と異なり，本論文における提案手法は，長さの等しいシーケンス集合の類似探索だけでなく，異なる長さのシーケンス集合からの探索についても，効率的に処理することができる．

実験では，既存手法と比べ最大で約 54 倍の性能向上を達成し，提案手法の優位性が明らかとなった．また，データ集合サイズが大きくなるほど，もしくはシーケンス長が長くなるほど提案手法の優位性は高まる．これは，大規模で長いシーケンスの時系列データベースにとって，提案手法がより有効であることを示している．

1.3 本論文の構成

本論文の構成は以下のとおりである．2 章では，DTW について述べる．3 章は，提案手法に関する記述である．4 章において，既存手法と提案手法の探索性能を比較した実験結果を提示する．5 章は，結論とまとめである．

2. 関 連 研 究

2.1 ダイナミックタイムワーピング

ダイナミックタイムワーピング（DTW; Dynamic Time Warping）とは，2 つのシーケンスの距離を最小化するように時間軸を伸長させる変換処理である．長さ n_P のシーケンス $P = \{p_1, \ldots, p_{n_P}\}$ と長さ n_Q のシーケンス $Q = \{q_1, \ldots, q_{n_Q}\}$ との距離は，ユークリッド距離関数を用いた場合 $D_{euclid}(P, Q) = \sum_{i=1}^{n} \|p_i - q_i\|$ となる．ここで，$n = n_P = n_Q$ であり，$\|p_i - q_i\|$ は p_i と q_i との 2 乗距離を意味する．これに対して，DTW 距離は以下のように定義される．

$$
\begin{aligned}
&D_{dtw}(P, Q) = f(n_P, n_Q),\\
&f(i, j) = \|p_i - q_j\| + \min \begin{cases} f(i, j-1) \\ f(i-1, j) \\ f(i-1, j-1), \end{cases}\\
&f(0, 0) = 0, f(i, 0) = f(0, j) = \infty\\
&(i = 1, \ldots, n_P; j = 1, \ldots, n_Q).
\end{aligned}
$$

このように，P の各要素と Q の各要素を昇順にマッチングすることによって DTW 距離は得られる．動的計画法を用いることによって DTW 距離が得られるため，計算コストは $O(n_P \cdot n_Q)$ となり，特にシーケンスが長くなるほど多大な計算コストが発生する．

2.2 関 連 研 究

Agrawal らは，時系列データの類似探索のためのアプローチを提案した[1]．シーケンスから特徴ベクトルを抽出し，R*-tree[4]を用いて索引付けを行う．

Keogh らは Adaptive Piecewise Constant Approximation（APCA）に基づくインデックス手法を提案している[10]．APCA は時系列データの次元縮退手法の 1 つであり，時系列データをセグメントと呼ぶ断片に分割して近似する．フーリエ変換，ウェーブレット変換，KL 展開など，これまで数多くの次元縮退手法が提案されてきたが，Keogh らは APCA の近似が優れていることを実験によって示している[10]．

最近のアプリケーションはシーケンスの類似性を計算するために DTW を用いている[7),13),14]．マッチングコストを削減するために，動的計画法に基づいたシーケンスマッチングの高速化手法が，特に音声認識[15]やバイオインフォマティックス[13]などの分野において提案されてきた．しかし，これらの多くは精度を犠牲にして速度を向上させるものであった．

Yi らは，DTW のための近似距離関数を提案している[19]．問合せシーケンスと各データシーケンスとの距離については，まず近似距離関数によって評価し，その後厳密な DTW 距離を求めることによって，探索処理の高速化を図っている．この近似距離関数は，問合せシーケンスの最大値と最小値の範囲とデータシーケンスの各要素との 2 乗距離の和によって計算される．この関数を用いることによって，探索漏れが起こらないことが保証されるものの，データシーケンスと問合せシーケンスが近づくと非常に粗い近似になるという問題点がある．

Kim らは，4 次元ベクトルを用いた近似距離関数を提案している[12]．4 次元ベクトルは，シーケンスの最初の値，最後の値，最大値，最小値によって構成されており，このベクトルを多次元インデックスによって索引付けしている．しかし，文献 9) における実験では，近似が粗いために，探索処理において厳密な距離計算の回数を十分に削減できず，結果として高い探索コストを示している．

Keogh は，Piecewise Aggregate Approximation（PAA）を用いた距離近似手法を提案している[9]．**図 1** に示すように，PAA は，次元を削減するためにシーケンスを同じサイズのセグメントに分割したものである．Zhu らも PAA を用いた近似手法を提案しており，文献 9) の手法に改良を加えている[20]．これらの

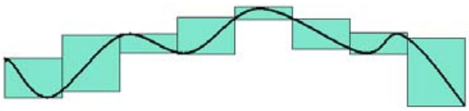


図 1 PAA 表現の例．各セグメントは同じサイズであり，最小値と最大値によって構成されている．この場合，シーケンスは 8 次元に削減されている

Fig. 1 Example of a PAA representation. Each equal-sized segment consists of its lower and upper bounds. In this case, the sequence is reduced to 8 dimensions.

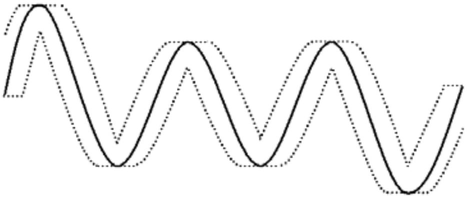


図 2 シーケンス帯の例．帯は上界と下界から構成されており，シーケンスをすべて包囲している

Fig. 2 Example of a sequence envelope. The envelope consists of lower and upper bounds that totally enclose the sequence.

近似手法は全体的なパス制約（global constraint）に基づいている．全体的なパス制約は，動的計画法において使われている制約の 1 つであり，ワーピングパスがとりうる範囲を限定するものである．これらの手法は，ワーピングパスの範囲から問合せシーケンスの帯（**図 2**）を計算し，この帯の PAA 表現とデータシーケンスの PAA 表現とのユークリッド距離を近似距離として用いている．これらの手法は，ワーピングパスを狭い範囲に限定すると効果を発揮するが，設定するワーピングの幅を広くするにつれて探索性能が低下する．これらの手法は有用であるが，最適なワーピングの幅はデータやアプリケーションに依存する．したがって，狭いワーピング範囲だけでなく，広いワーピング範囲でも高い性能を発揮するような探索手法が望まれる．

2.3 PAA を用いた既存手法

長さ n_P のシーケンス $P = \{p_1, \ldots, p_{n_P}\}$ が与えられたとき，P の PAA 表現 A は以下のように定義される[11),18)]．

$$\begin{aligned} A &= \{a_1, \ldots, a_{n_A}\}, \qquad (1)\\ a_i &= \{a_i^L, a_i^U\},\\ a_i^L &= \min(p_{l\cdot(i-1)+1} : p_{l\cdot i}),\\ a_i^U &= \max(p_{l\cdot(i-1)+1} : p_{l\cdot i}). \end{aligned}$$

すなわち，A は，P をセグメントと呼ぶ断片に分割し，最小値と最大値によって作成したものである．セグメント a_i に関して，a_i^L と a_i^U は，P における $p_{l\cdot(i-1)+1}$ から $p_{l\cdot i}$ までの範囲の中で，それぞれ最小値，最大値を示している．

n_A は PAA 表現 A のセグメント数であり，$n_A < n_P$ である．セグメントの基準長を l とすると，A のセグメント数は $n_A = \lceil n_P/l \rceil$ である．セグメント a_i の長さ a_i^R は，以下のとおりである．

$$a_i^R = \begin{cases} l & (1 \le i < n_A) \\ n_P - l \cdot (n_A - 1) & (i = n_A). \end{cases}$$

例 1 長さ $n_P = 16$ のシーケンス P と長さ $n_Q = 16$ のシーケンス Q が与えられているとする．$l = 4$ とするとき，P の PAA 表現 A，および Q の PAA 表現 B は以下のとおりである．

$$\begin{aligned} &P = \{9, 9, 8, 8, 5, 3, 4, 2, 1, 2, 0, 2, 6, 8, 9, 9\},\\ &A \ : \ a_1 = \{8, 9\}, a_2 = \{2, 5\}, a_3 = \{0, 2\},\\ &\quad a_4 = \{6, 9\},\\ &Q = \{7, 7, 7, 6, 4, 2, 0, 2, 4, 4, 5, 4, 4, 5, 5, 5\},\\ &B \ : \ b_1 = \{6, 7\}, b_2 = \{0, 4\}, b_3 = \{4, 5\},\\ &\quad b_4 = \{4, 5\}. \end{aligned}$$

ここで，$n_A = 4$, $n_B = 4$ である． □

文献 9) と 20) における近似手法は，ともに PAA 表現を用いており，その近似距離は，問合せシーケンスの帯の PAA 表現とデータシーケンスの PAA 表現とのユークリッド距離として定義されている．しかし両者は，帯の PAA 表現の計算方法が異なる．文献 20) において Zhu らは，彼らの提案手法の方が文献 9) の手法よりも優れていることを示しているため，本論文では主として彼らの手法に焦点を合わせる．

ワーピングパスの範囲を w とする．長さ n_Q のシーケンス Q が与えられているとき，Q の帯は以下のように定義される．

$$\begin{aligned} &Q^L = \{q_1^L, \ldots, q_{n_Q}^L\}, \quad q_i^L = \min(q_{i-w} : q_{i+w}),\\ &Q^U = \{q_1^U, \ldots, q_{n_Q}^U\}, \quad q_i^U = \max(q_{i-w} : q_{i+w}),\\ &\qquad (i = 1, \cdots, n_Q). \end{aligned}$$

ここで Q^L と Q^U は，帯の下界と上界である[9),20)]．Keogh の手法では，帯の PAA 表現は式 (1) によって計算される．Zhu らの手法では，以下のように帯の PAA 表現を求める．

$$E = \{e_1, \ldots, e_{\lceil n_Q/l \rceil}\}, \qquad e_i = \{e_i^L, e_i^U\},$$

$$e_i^L = \frac{1}{l} \sum_{j=(i-1)\cdot l+1}^{i\cdot l} q_j^L, \quad e_i^U = \frac{1}{l} \sum_{j=(i-1)\cdot l+1}^{i\cdot l} q_j^U.$$

彼らの手法では，各セグメントは帯の下界もしくは上界の平均である．シーケンス P の PAA 表現 $A = \{a_1, \ldots, a_{n_A}\}$ が与えられたとき，P と Q の

近似距離は，A と E のユークリッド距離として定義される．

$$D_{lb-paa}(A,E)=\sum_{i=1}^{n_A} a_i^R \cdot \|(a_i^L : a_i^U)-(e_i^L : e_i^U)\|.$$

ここで，$n_A = \lceil n_Q/l \rceil$，そして $n_P = n_Q$ である．$\|(a_i^L : a_i^U) - (e_i^L : e_i^U)\|$ は，範囲 $(a_i^L : a_i^U)$ と範囲 $(e_i^L : e_i^U)$ との 2 乗距離を意味する．

例 2　例 1 で示したシーケンス P と Q を考える．もし，ワーピングの範囲を $w = 3$（すなわち，シーケンス長 n_Q の 18.75%）とするとき，Q の帯は以下のとおりである．

$$Q^L = \{6,4,2,0,0,0,0,0,0,0,2,4,4,4,4,4\},$$
$$Q^U = \{7,7,7,7,7,7,6,5,5,5,5,5,5,5,5,5\}.$$

文献 20) の近似手法を用いることによって，帯の PAA 表現を以下のように得ることができる．

$$E : e_1 = \{3,7\}, e_2 = \{0,6.25\},$$
$$e_3 = \{1.5,5\}, e_4 = \{4,5\}.$$

近似距離は，A と E のユークリッド距離であるため，$D_{lb-paa}(A,E) = 8$．一方，もしワーピングの範囲を $w = 15$（すなわち，範囲に制限がない場合）とするとき，帯の PAA 表現は

$$E : e_1=\{0,7\}, e_2=\{0,7\}, e_3=\{0,7\}, e_4=\{0,7\}$$

であるため，$D_{lb-paa}(A,E) = 4$．　□

例 2 に示したように，ワーピングの範囲が拡大するに従って帯が広くなり，その結果，近似距離が小さくなる．

3. 提案手法

2.1 節において述べたように，厳密な DTW 距離の計算コストは動的計画法に基づいて行われるため，シーケンス長が増えるに従い多大な計算コストが発生する．さらに，時系列データ集合のサイズが大きくなるにつれて，問合せの探索コストは増大する．類似検索を高速化するため，計算コストの低く，精度の高い近似距離関数が必要となる．

本論文では，k 近傍問合せに焦点を合わせて説明しているが，提案手法は範囲問合せにも有効である．本手法は，k 近傍問合せの場合 k 近傍距離を用いて処理を行っているが，範囲問合せでは探索範囲を示す距離を用いて処理を行う．

3.1 探索漏れが発生しないことを保証するための必要十分条件

従来の近似手法は，探索漏れが発生しないことを保証するため下界（lower bounding）の特性を用いてきた[3),6),8)∼10),16),20)]．我々が提案する必要十分条件と比較するため，ここで下界の性質を示す．

性質 1　（下界）

$D_{exact}(P,Q)$ をシーケンス P と Q の厳密な距離とし，$D_{approx}(P,Q)$ を P と Q の近似距離とする．距離関数は下式を満たす．

$$D_{approx}(P,Q) \le D_{exact}(P,Q).$$

□

性質 1 に基づく類似問合せ処理のためのアルゴリズムは，近似距離が k 近傍距離を上回るようなシーケンスは除外し，近似距離が k 近傍距離以下であるシーケンスについては厳密な距離を計算する．このため探索漏れが発生しない．しかし，性質 1 は探索漏れが発生しないことを保証するための必要条件ではない．なぜなら，厳密な距離が k 近傍距離を超えるとき，下界の性質は必要ないためである．

次の新しい性質は，類似問合せ処理において，探索漏れが発生しないことを保証するための必要十分条件となる．

性質 2　（必要十分条件）

シーケンス P と Q，および類似距離 $D_{similar}$ が与えられたとき，距離関数は下式を満たす．

$$If \quad D_{exact}(P,Q) \le D_{similar}$$
$$then \quad D_{approx}(P,Q) \le D_{similar}.$$

□

ここで，$D_{similar}$ は，k 近傍問合せにおいては k 近傍距離，範囲探索においては探索範囲を意味する．

補題 1　性質 2 は類似問合せ処理において，探索漏れが発生しないことを保証するための必要十分条件である．

証明： まず，以下のような条件を考える．

C_1：距離近似は，探索漏れが発生しないことを保証する．

C_2：距離近似は，性質 2 を持つ．

データシーケンス P と問合せシーケンス Q が与えられたとき，$D_{approx}(P,Q) > D_{similar}$ である場合にのみ P は除外される．類似問合せ処理は，$D_{exact}(P,Q) \le D_{similar}$ となる P を除外した場合にのみ探索漏れを起こす．したがって近似は，$D_{exact}(P,Q) \le D_{similar}$ である場合に $D_{approx}(P,Q) \le D_{similar}$ を満たすような距離を与えなければならない．これは $C_1 \Rightarrow C_2$ と $C_1 \Leftarrow C_2$ につながる．よって，補題が成り立つ．

□

ここで提案した必要十分条件は，シーケンスの類似探索のみに限定されるものではない．多次元データ，文字列，XML など距離近似を行う様々な処理に適用

することができる．

本論文において提案する近似手法は，性質 1 を持たないものの，性質 2 を満たす．したがって，探索漏れがないことを保証する．

3.2 基本的なアイデア

DTW 距離関数とユークリッド距離関数の違いは，前者がワーピングパスとその幅を持っている点である．すなわち，ただ 1 つのパスを計算するユークリッド距離関数と異なり，DTW 距離関数はすべてのワーピングパスの距離を計算しなければならないため，多くの計算コストを要する．そこで我々のアプローチは，以下のようなアイデアに基づいて距離計算のコストを低減させている．

(1) もし，ワーピングの範囲の中に，計算する必要のない部分を数多く検出することができれば，計算コストの低減に有効である．類似探索処理の途中で得られる k 近傍距離（問合せシーケンスと候補シーケンスとの距離）と比較して，この k 近傍距離よりも大きい距離値をもたらすことが明らかなワーピングパスは，距離計算の対象から除外する．

(2) まず，DTW の近似距離を粗く高速に計算する．もし，現時点での k 近傍距離よりも近似距離が大きければ，そのシーケンスは問合せシーケンスと類似していないと見なして安全に除外する．k 近傍距離よりも近似距離が小さければ，そのシーケンスについては，より精密な近似距離を求めて k 近傍距離と比較する．

アイデア (1) は，探索処理途中の k 近傍距離と比較することによって，不要なワーピングパスを取り除き，ワーピングの範囲を制限するものである．このアイデアは性質 2 に基づいている．制限されたワーピングの範囲が狭くなるほど，効率的で精度の高い近似が可能となる．アイデア (2) は，類似していないシーケンスについては粗い近似を行って除外し，問合せシーケンスと類似しているものほど，より精密な近似を行うというものである．これは，インターレース GIF 方式の画像を連想させるかもしれない．インターレース GIF 方式の画像では，最初はぼんやりした画像が現れ，ダウンロードが進むと次第に画像が鮮明になってくる．ダウンロードの途中でも画像のおおよその内容が分かる．これと同様に，k 近傍距離よりも小さければ，徐々に精密な近似距離を求めていく．

本論文では，不必要なワーピングパスを削除してワーピングの範囲を削減するために，動的計画法と PAA を組み合わせた距離関数を新たに提案する．そして，性質 2 を満たす近似手法を提案する．近似距離は，ウェーブレット係数と削減されたワーピング範囲から計算される．この近似手法をとり入れた探索アルゴリズムでは，問合せシーケンスと類似していないシーケンスについては粗い近似距離を高速に計算し，k 近傍距離よりも大きいことを確認して安全に除外する．類似しているシーケンスについては，さらに精密な近似を行い，最終的な探索結果を得ることができる．

3.3 PAA 表現のための DTW

動的計画法と PAA を組み合わせた新たな距離関数について述べる．シーケンス P の PAA 表現 $A=\{a_1,\ldots,a_{n_A}\}$ とシーケンス Q の PAA 表現 $B=\{b_1,\ldots,b_{n_B}\}$ が与えられており，$a_i=\{a_i^L,a_i^U\}$，$b_j=\{b_j^L,b_j^U\}$ とする．また，セグメント a_i，b_j の長さは各々 a_i^R，b_j^R とする．ここで，下界距離を出力する新たな距離関数を提案する．

$$
\begin{aligned}
&D_{dp-paa}(A,B)=g(n_A,n_B),\\
&g(i,j)=g_{seg}(i,j)+\min\begin{cases} g(i,j-1)\\ g(i-1,j)\\ g(i-1,j-1),\end{cases} \qquad (2)\\
&g_{seg}(i,j)=\min(a_i^R,b_j^R)\cdot\|(a_i^L:a_i^U)-(b_j^L:b_j^U)\|,\\
&g(0,0)=0, g(i,0)=g(0,j)=\infty.
\end{aligned}
$$

補題 2 P と P の PAA 表現 A，Q と Q の PAA 表現 B が与えられたとき，$D_{dp-paa}(A,B)\le D_{dtw}(P,Q)$ が成り立つ．

証明：

$g_{seg}(1,1)=\min(a_1^R,b_1^R)\cdot\|(a_1^L:a_1^U)-(b_1^L:b_1^U)\|$,

であるため，下式が成り立つ．

$$g(1,1)\le f(x,b_1^R)\ \ (1\le x\le a_1^R),$$
$$g(1,1)\le f(a_1^R,y)\ \ (1\le y\le b_1^R).$$

さらに，

$g_{seg}(i,j)=\min(a_i^R,b_j^R)\cdot\|(a_i^L:a_i^U)-(b_j^L:b_j^U)\|$,

であるため，下式が成り立つ．

$$
\begin{aligned}
&\min\{g(i-1,j-1),\ g(i-1,j)\}\le f(x,y)\\
&\qquad (x=\sum_{k=1}^{i-1}a_k^R,\ \ \sum_{k=1}^{j-1}b_k^R<y\le\sum_{k=1}^{j}b_k^R),\\
&\min\{g(i-1,j-1),\ g(i,j-1)\}\le f(x,y)\\
&\qquad (\sum_{k=1}^{i-1}a_k^R<x\le\sum_{k=1}^{i}a_k^R,\ \ y=\sum_{k=1}^{j-1}b_k^R).
\end{aligned}
$$

よって，下式が成り立つ．

$$g(i,j)\le f(x,y)\quad (x=\sum_{k=1}^{i}a_k^R,\ \ y=\sum_{k=1}^{j}b_k^R).$$

したがって，$g(n_A,n_B)\le f(n_P,n_Q)$ であるため，補

```
Procedure improvedDP(PAA A, PAA B, distance D_{k-nn})
  for i = 1 to n_A do
    begin(i) := 0;
    end(i) := n_B;
  enddo
  for i = 1 to n_A do
    for j = begin(i) to end(i) do
      compute the distance g(i, j)
      if j > end(i) and g(i, j) > D_{k-nn}
        and i ≠ n_A then
        end(i) := j;
        break;
      endif
    endfor
    if no segment which satisfies begin(i) ≤ j ≤ end(i)
      and g(i, j) ≤ D_{k-nn} exists then
      return g(i, end(i));
    else
      begin(i) :=
        min_{begin(i)≤j≤end(i)}{j|g(i, j) ≤ D_{k-nn}};
      end(i) :=
        max_{begin(i)≤j≤end(i)}{j|g(i, j) ≤ D_{k-nn}};
      if i ≠ n_A and begin(i + 1) < begin(i) then
        begin(i + 1) := begin(i);
    endif
  endfor
  return g(n_A, n_B);
```

図 3 k 近傍距離を用いた DTW 距離計算アルゴリズム

Fig. 3 A DTW distance calculation algorithm using k-nearest neighbor distance.

題が成り立つ． □

次に，$D_{dp-paa}(A,B)$ を計算するためのアルゴリズムについて述べる．3.2 節でも述べたように，探索処理の途中では k 近傍距離 D_{k-nn} を得ることができる．式 (2) においては，D_{k-nn} よりも大きい値をとることが明らかな $g(i,j)$ は計算する必要がない．**図 3** は，動的計画法のアプローチを改良し，k 近傍距離を用いて効率的に DTW 距離を計算するアルゴリズムである．

例 3 **図 4** (a) は，このアルゴリズムの動きを例示したものである．A と B の値は，例 1 で算出したものを用いている．図 4 (a) において，枡の中の数値は $g(i,j)$ を表している．$d_{k-nn} = 22$ とすると，$g(1,2) = 68$ であるため，$g(1,3)$ と $g(1,4)$ の計算は省略することができる．また，$g(3,1) = 72$ であるため，$g(4,1)$ の計算は省かれている． □

このアルゴリズムは，PAA 表現の DTW 距離だけでなく，シーケンスの DTW 距離を計算する場合にも適用し，計算コストの削減が可能となる．我々の手法は，シーケンス P と Q の距離 $D_{dtw}(P,Q)$ の計算にも，図 3 のアルゴリズムを導入する．また，アルゴリズムは全体的なパス制約（global constraint）にも対応できる．その際には，アルゴリズムの最初で設定している $begin(i)$ と $end(i)$ の初期値を，制約条件に合わせて変更する．

3.4 DTW 距離の近似

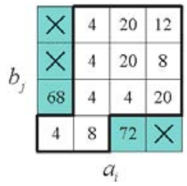
より精密に DTW 距離を近似するために，本節で述べる近似手法は，ウェーブレット係数と 3.3 節で述べた距離関数を用いる．距離関数によって不要なワーピングパスを排除し，ワーピングの範囲を削減する．削減したワーピングの範囲に基づき，ウェーブレット係数を用いて精度の高い近似を行う．

本節では，説明を単純にするために，長さが 2 のべき乗であるシーケンスについて述べているが，近似手法は任意の長さに対応することができる．

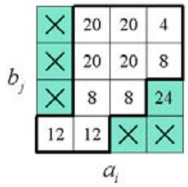
3.4.1 ウェーブレット係数

$P = \{p_1, \ldots, p_{n_P}\}$ を長さ n_P のシーケンスとする．ハール基底に基づく $r(0 \le r \le \log_2 n_P)$ レベルのウェーブレット変換 W_r は以下のように得られる[17]．

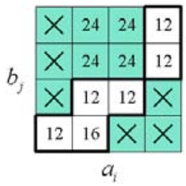
$$
\begin{aligned}
W_r &= \begin{cases} P & (r=0) \\ \{\Phi_r, \Phi'_r\} & (0 < r \le \log_2 n_P), \end{cases} \\
\Phi_r &= \{\phi_{r,1}, \ldots, \phi_{r,n_\Phi}\}, \\
\Phi'_r &= \{\phi'_{r,1}, \ldots, \phi'_{r,n_\Phi}\}, \\
\phi_{r,i} &= \begin{cases} p_i & (r=0) \\ (\phi_{r-1,2i-1} + \phi_{r-1,2i})/\sqrt{2} & (0 < r \le \log_2 n_P), \end{cases} \\
\phi'_{r,i} &= (\phi_{r-1,2i-1} - \phi_{r-1,2i})/\sqrt{2} \quad (0 < r \le \log_2 n_P).
\end{aligned}
\tag{3}
$$



(a) $g(i,j)$



(b) $g'(i,j)$



(c) $g_{point}(i,j)$

図 4　PAA 表現を用いた DTW 距離計算の例

Fig. 4　Example of a DTW distance calculation using PAA representations.

ここで，$n_\Phi = n_P/2^r$ である．我々の手法は W_r の中で，Φ_r の係数のみを用いる．文献 17) より，

$$\sum_{i=1}^{n_\Phi} \phi_{r,i}^2 \le \sum_{i=1}^{n_P} p_i^2. \quad (4)$$

が成り立つ．

例 4　例 1 で示したシーケンス P と Q を考える．$r=1$ とするとき，P のウェーブレット係数 Φ_1 と Q のウェーブレット係数 Ψ_1 は以下のとおりである．

$$\Phi_1 = \{9\sqrt{2}, 8\sqrt{2}, 4\sqrt{2}, 3\sqrt{2}, 3\sqrt{2}/2, \sqrt{2}, 7\sqrt{2}, 9\sqrt{2}\},$$
$$\Psi_1 = \{7\sqrt{2}, 13\sqrt{2}/2, 3\sqrt{2}, \sqrt{2}, 4\sqrt{2}, 9\sqrt{2}/2, 9\sqrt{2}/2, 5\sqrt{2}\}.$$

ここで，$n_\Phi = 8$，$n_\Psi = 8$ である．□

3.4.2　ワーピング範囲の削減

ここで，再び図 4 (a) を例として考える．図中で，もし格子点 $(3,3)$ を通過するワーピングパスすべてが k 近傍距離よりも大きければ，格子点 $(3,3)$ は DTW 距離の近似を行う際に考慮する必要がない．このように，近似に不必要な格子点を調べることにより，不要なワーピングパスを削除し，ワーピングの範囲を削減する．

今ここで，$g(n_A, n_B) = g'(1,1)$ となるような逆方向の DTW 距離関数を考える．

$$g'(i,j) = g_{seg}(i,j) + \min \begin{cases} g'(i, j+1) \\ g'(i+1, j) \\ g'(i+1, j+1), \end{cases} \quad (5)$$
$$g'(n_A+1, n_B+1) = 0,$$
$$g'(i, n_A+1) = g'(n_B+1, j) = \infty.$$

式 (2) において示した順方向の距離関数 $g(i,j)$ は格子点 $(1,1)$ から格子点 (i,j) までのワーピングパスの中で，最小距離を表す．式 (5) において示した逆方向の距離関数 $g'(i,j)$ は格子点 (n_A, n_B) から格子点 (i,j) までのワーピングパスの中で，最小の距離を示している．したがって，$g_{point}(i,j)$ は，格子点 (i,j) を通過するワーピングパスの中で，最小距離を意味する．

$$g_{point}(i,j) = g(i,j) + g'(i,j) - g_{seg}(i,j)$$

もし以下の条件を満たすとき，格子点 (i,j) は k 近傍距離に影響を与える可能性がないため，DTW 距離の近似の対象から除外できる．

$$g_{point}(i,j) > D_{k-nn}. \quad (6)$$

式 (6) により，ワーピングパスの範囲は，PAA 表現における $begin(i)$ から $end(i)$ まで $(i = 1, \ldots, n_A)$ の格子点の範囲に削減することができる．

$$begin(i) = \min_{0 \le j \le n_B} \{j | g_{point}(i,j) \le D_{k-nn}\}, \quad (7)$$
$$end(i) = \max_{0 \le j \le n_B} \{j | g_{point}(i,j) \le D_{k-nn}\}.$$

したがって，以下のような PAA の帯を得ることができる．

$$E = \{e_1, \ldots, e_{n_A}\}, \quad e_i = \{e_i^L, e_i^U\}, \quad (8)$$
$$e_i^L = \min(b_{begin(i)}^L : b_{end(i)}^L),$$
$$e_i^U = \max(b_{begin(i)}^U : b_{end(i)}^U),$$

ここで，$\min(b_{begin(i)}^L : b_{end(i)}^L)$ は B のセグメント集合 $\{b_{begin(i)}, \ldots, b_{end(i)}\}$ の最小値を示している．同様に，$\max(b_{begin(i)}^U : b_{end(i)}^U)$ は，そのセグメント集合の最大値である．

例 5　図 4 (a) は $g(i,j)$ の値を，図 4 (b) は $g'(i,j)$ の値を，図 4 (c) は $g_{point}(i,j)$ の値を示している．$D_{k-nn} = 22$ とすると，式 (7) によって，$begin(1) = 1$，$end(1) = 1$，$begin(2) = 1$，$end(2) = 2$，$begin(3) = 2$，$end(3) = 2$，$begin(4) = 3$，$end(4) = 4$ を得ることができる．図 4 (c) は削減されたワーピング範囲を示している．そこから，以下のような帯の PAA 表現を得ることができる．

$$E : e_1 = \{6,7\}, e_2 = \{0,7\}, e_3 = \{0,4\}, e_4 = \{4,5\}.$$

□

3.4.3 距離近似

式 (3) において示したように P のウェーブレット係数を Φ_r とする．式 (8) において示したようにシーケンス Q の帯の PAA 表現を E とする．DTW の近似距離を以下のように得ることができる．

$$D_{wave-paa}(\Phi_r, E) = \sum_{i=1}^{n_\Phi} \|\phi_{r,i} - 2^{r/2} \cdot (e_j^L : e_j^U)\|, \quad (9)$$
$$j = \lceil i \cdot 2^r / l \rceil.$$

本手法は，PAA 表現と 3.4.1 項で述べたウェーブレット係数を併用する．そこで，ウェーブレット係数のレベルを r とすると，PAA 表現の基準長 l は

$$l \bmod 2^r = 0$$

となるように選択する．

補題 3 シーケンス P の PAA 表現を A，Q の PAA 表現を B，B の帯を E，P の r レベルのウェーブレット係数を Φ_r とする．このとき，以下の不等式が成り立つ．

$$If \quad D_{dtw}(P,Q) \le D_{k-nn}$$
$$then \quad D_{wave-paa}(\Phi_r, E) \le D_{k-nn}.$$

証明： 補題 2 と式 (7) により，もし $D_{dtw}(P,Q) \le D_{k-nn}$ である場合に，$D_{dtw}(P,Q)$ を出力するワーピングパスを E は必ず包囲している．したがって，以下の不等式が成り立つ．

$$D_{dtw}(P,Q) \ge \sum_{i=1}^{n_P} \|p_i - (e_i^L : e_i^U)\|$$

式 (4) より以下の不等式が成り立つ．

$$\sum_{i=1}^{n_P} \|p_i - (e_i^L : e_i^U)\| \ge \sum_{i=1}^{n_\Phi} \|\phi_{r,i} - 2^{r/2} \cdot (e_j^L : e_j^U)\|,$$
$$j = \lceil i \cdot 2^r / l \rceil.$$

よって，補題が成立する． □

本手法では，帯 E が最適なワーピングパスを包囲していない可能性がある．したがって，性質 1 を満たしていない．しかし D_{k-nn} を出力するワーピングパスが存在する場合，E はそのパスを必ず包囲している．よって，性質 2 を満たす．

例 6 P のウェーブレット係数を Φ_r，B の帯を E とする．P と Q の近似距離は以下のとおりである．

$$\begin{aligned} & D_{wave-paa}(\Phi_r, E) \\ & = \|9\sqrt{2} - (6\sqrt{2} : 7\sqrt{2})\| + \|8\sqrt{2} - (6\sqrt{2} : 7\sqrt{2})\| \\ & \quad + \|4\sqrt{2} - (0 : 7\sqrt{2})\| + \|3\sqrt{2} - (0 : 7\sqrt{2})\| \\ & \quad + \|3\sqrt{2}/2 - (0 : 4\sqrt{2})\| + \|\sqrt{2}/2 - (0 : 4\sqrt{2})\| \\ & \quad + \|7\sqrt{2} - (4\sqrt{2} : 5\sqrt{2})\| + \|9\sqrt{2} - (4\sqrt{2} : 5\sqrt{2})\| \\ & = 50. \end{aligned}$$

□

3.5 索引構造

索引として，我々はシーケンシャルな構造を提案する．索引は単純であり，以下のような特徴量データの配列である．

$$F(P) = \{n_P, v_P, Set(A), Set(\Phi)\},$$
$$Set(A) = \{A_{l_h}, \ldots A_{l_2}, A_{l_1}\},$$
$$Set(\Phi) = \{\Phi_{r_h}, \ldots \Phi_{r_2}, \Phi_{r_1}\}.$$

P の特徴量データ $F(P)$ は，P の長さ n_P，P の分散値 v_P ($v_P = \sum_{i=1}^{n_P} \|p_i - \bar{p}\,|$)，PAA 表現の集合 $Set(A)$，ウェーブレット係数の集合 $Set(\Phi)$ から構成されている．$Set(A)$ は h 種類の PAA 表現を含んでいる．l_i は PAA 表現 A_{l_i} を作成するための基準長であり，以下のような大小関係になっている．

$$1 < l_1 < l_2 < \ldots < l_{h-1} < l_h < n_P.$$

すなわち，A_{l_h} を用いた近似は最も粗く，A_{l_1} を用いた近似は最も精密な近似となる．同様に，$Set(\Phi)$ も h 種類のウェーブレット係数を含んでいる．

3.6 探索処理

範囲問合せと k 近傍問合せは，時系列データを用いたアプリケーションにとって有用である．本節で述べる探索アルゴリズムは両方の問合せを効率的に支援することができる．本論文では k 近傍探索について述べるが，距離近似手法，索引構造，探索アルゴリズムはどのような範囲問合せについても適用することができる．

本論文において提案する探索アルゴリズムは以下のような 2 つの特徴を持っている．

(1) 段階的に精度を向上させる距離近似
厳密な DTW 距離の計算コストに比べて，近似手法の計算コストは低い．また，PAA 表現の基準長が長くなるほど近似の計算コストは低くなる．すなわち，PAA 表現の基準長 l_i $(1 \le i \le h)$，ウェーブレット変換のレベル r_i を考えたとき，l_i と r_i による距離近似 $dist(l_i, r_i)$ と計算コスト $cost(l_i, r_i)$ は以下のとおりである．

$$cost(l_h, r_h) < \ldots < cost(l_1, r_1) < cost(exact)$$
$$dist(l_h, r_h) \ge \ldots \ge dist(l_1, r_1)$$

そこで，探索アルゴリズムは最初に，l_h と r_h を用いて距離の近似を行う．もし，その時点での k 近傍距離よりも近似距離が大きければ，厳密な距離計算を行わずに，安全にシーケンスを除外することができる．l_h と r_h の近似距離が

k 近傍距離よりも小さければ，l_{h-1} と r_{h-1} を用いて，より精密な近似距離を求めて k 近傍距離と比較する．最後に，l_1 と r_1 を用いても k 近傍距離を上回る近似距離を得られないときは，高い計算コストを払って厳密な距離計算を行う．

(2) ユークリッド距離による候補 k 近傍の収集
我々の探索手法は，探索処理実行途中の k 近傍距離，すなわち候補 k 近傍距離に基づいて距離計算コストの低減化を図っている．探索処理の初期の段階から，候補 k 近傍距離が最終的な k 近傍距離と近いほど，近似手法の効率と精度は増す．そこで，すべてのシーケンスについてウェーブレット係数からユークリッド距離を算出し，ユークリッド距離が小さい k 個のシーケンスを収集し，これらを候補 k 近傍シーケンスとする．候補 k 近傍シーケンスから厳密な DTW 距離を計算し，候補 k 近傍距離とする．候補 k 近傍距離を計算した後，この距離を用いて DTW 距離の近似を行っていく．

図 5 は 3.5 節で述べた索引構造を利用した探索アルゴリズムである．まず，問合せシーケンスの分散，ウェーブレット係数，PAA 表現を計算する．次に，ウェーブレット係数を用いて候補 k 近傍シーケンスを収集し，候補 k 近傍シーケンスの厳密な DTW 距離を計算し，候補 k 近傍距離を得る．その後，PAA 表現の基準長とウェーブレット変換のレベルを段階的に小さくしながら距離近似を行う．もし，P の分散値が Q の分散値よりも大きければ，ウェーブレット係数 Φ_r と B の帯から近似距離 $D_{wave-paa}(\Phi_r, E_B)$ を計算する．そうでなければ，近似距離 $D_{wave-paa}(\Psi_r, E_A)$ を計算する．候補 k 近傍距離よりも近似距離が大きければシーケンスを安全に除外する．なぜなら，そのシーケンスは最終的な k 近傍シーケンスに入る可能性がないためである．

我々の近似手法は，シーケンス長が異なるシーケンスのペアについても，その距離を近似することができる．また，探索アルゴリズムは問合せシーケンスの長さとデータベースに格納された各シーケンスの長さを考慮して，最も類似したシーケンスを見つけることができる．ただし，図 5 のアルゴリズムでは，ウェーブレット係数のユークリッド距離を計算している．我々はシーケンスの長さが異なる場合のために，以下のような定義を与える．

シーケンス P のウェーブレット係数 $\Phi_r = \{\phi_{r,1}, \ldots, \phi_{r,n_\Phi}\}$，シーケンス Q のウェーブレット

```
Procedure search(sequence Q, integer k)
  calculate v_Q; // v_Q is the variance of Q
  calculate Set(Ψ); // Set(Ψ) = {Ψ_{r_h}, ..., Ψ_{r_1}}
    // Set(Ψ) is the set of wavelet coefficients of Q
  calculate Set(B); // Set(B) = {B_{l_h}, ..., B_{l_1}}
    // Set(B) is the set of PAA representations of Q
  for i = 1 to database_size do
    D_euclid[i] = ||Φ_{r_1}, Ψ_{r_1}||
    // Φ_{r_1} is the r_1-level wavelet coefficients of P
    // P is the i-th sequence
    if NNL_euclid[k].dist > D_euclid[i] then
      add i and D_euclid[i] to NNL_euclid
      // NNL is the nearest neighbor list
  enddo
  for each P ∈ NNL_euclid do
    add P and D_dtw(P,Q) to NNL_dtw
  for i = 1 to database_size do
    if D_euclid[i] ≥ NNL_euclid[k].dist then
      for (l = l_h, r = r_h) to (l_1, r_1) do
        if v_P > v_Q then
          if D_paa(A_l, B_l) > NNL_dtw[k].dist then
            break;
          else
            if D_wave-paa(Φ_r, E_B) > NNL_dtw[k].dist
              then
              break;
            // E_B is the envelope of B_l
        else
          if D_paa(B_l, A_l) > NNL_dtw[k].dist then
            break;
          else
            if D_wave-paa(Ψ_r, E_A) > NNL_dtw[k].dist
              then
              break;
            // E_A is the envelope of A_l
        if D_dtw(P,Q) ≤ NNL_dtw[k].dist then
          add P and D_dtw(P,Q) to NNL_dtw
      enddo
    endif
  enddo
  return NNL_dtw;
```

図 5 k 近傍探索アルゴリズム
Fig. 5 k-nearest neighbor search algorithm.

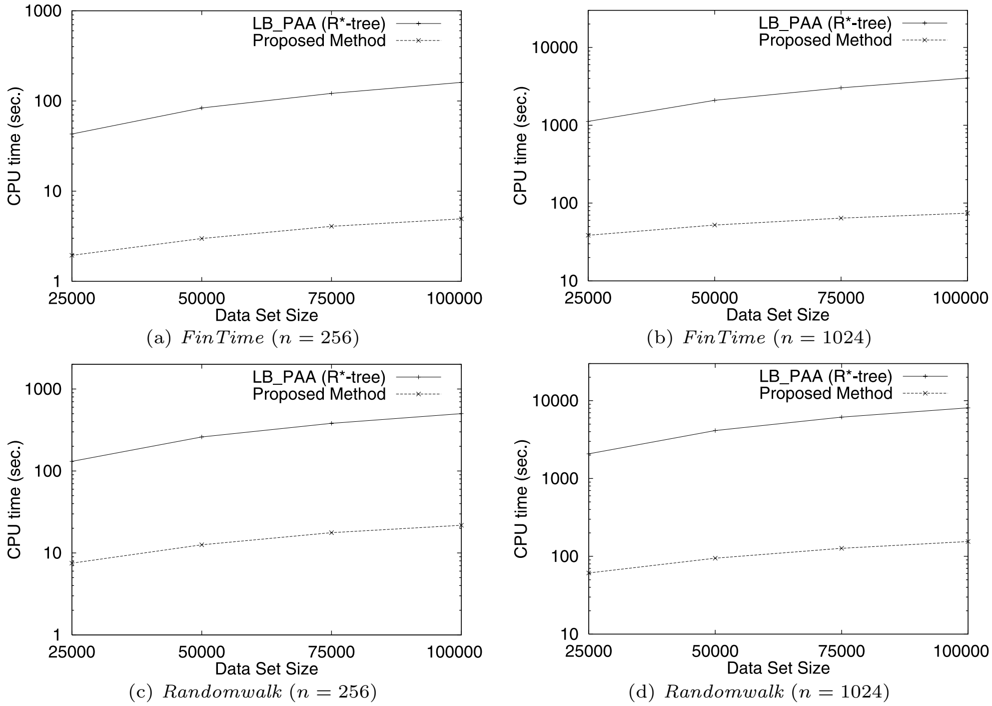
係数 $\Psi_r = \{\psi_{r,1}, \ldots, \psi_{r,n_\Psi}\}$ が与えられている．$n_\Phi < n_\Psi$ のとき，$\Phi'_r = \{\phi'_{r,1}, \ldots, \phi'_{r,n_\Psi}\}$ と Ψ_r とのユークリッド距離 $\|\Phi'_r, \Psi_r\|$ を以下のように定義する．

$$\|\Phi'_r, \Psi_r\| = \sum_{i=1}^{n_\Psi} (\phi'_{r,i} - \psi_{r,i})^2,$$

$$\phi'_{r,i} = \begin{cases} \phi_{r,j} & (j = \lceil j \rceil) \\ (\lceil j \rceil - j) \cdot \phi_{r,\lfloor j \rfloor} + (j - \lfloor j \rfloor) \cdot \phi_{r,\lceil j \rceil} & \\ & (otherwise), \end{cases}$$

$$j = (i-1) \cdot \frac{n_\Phi - 1}{n_\Psi - 1} + 1.$$

逆に $n_\Phi > n_\Psi$ のとき，$\|\Phi_r, \Psi'_r\|$ の計算が必要となる．したがって，図 5 のユークリッド距離計算には下式を導入することによって，長さの異なるシーケンス集合の類似探索が可能となる．



(a) *FinTime* ($n = 256$) (b) *FinTime* ($n = 1024$)
(c) *Randomwalk* ($n = 256$) (d) *Randomwalk* ($n = 1024$)

図 6 探索に要する CPU 時間
Fig. 6 CPU time for searching.

$$D_{euclid}[i] = \begin{cases} \|\Phi'_r, \Psi_r\| & (n_\Phi < n_\Psi) \\ \|\Phi_r, \Psi_r\| & (n_\Phi = n_\Psi) \\ \|\Phi_r, \Psi'_r\| & (n_\Phi > n_\Psi). \end{cases}$$

例 7 長さ $n_Q = 10$ のシーケンス $Q = \{7,7,7,6,4,2,0,2,4,4\}$ と，長さ $n_P = 6$ のシーケンス $P = \{9,9,8,8,5,3\}$ が与えられている．$r = 1$ とするとき，Q のウェーブレット係数 Ψ_1，長さを補正した P のウェーブレット係数 Φ'_1 は各々以下のようになる．

$$\Psi_1 = \{7\sqrt{2}, 13\sqrt{2}/2, 3\sqrt{2}, \sqrt{2}, 4\sqrt{2}\},$$
$$\Phi_1 = \{9\sqrt{2}, 8\sqrt{2}, 4\sqrt{2}\},$$
$$\Phi'_1 = \{9\sqrt{2}, 17\sqrt{2}/2, 8\sqrt{2}, 6\sqrt{2}, 4\sqrt{2}\}.$$

Φ'_1 と Q のウェーブレット係数 Ψ_1 とのユークリッド距離は $\|\Phi'_1, \Psi_1\| = 116$ となる． □

4. 性能評価

提案手法の効果を検証するために，提案手法と既存手法を実装し，比較実験を行った．比較対象として，文献 20) において提案されている既存手法を用いる．既存手法には，文献 20) に従って，索引構造として R*-tree [4] を用いる．本論文でも，この手法を文献 20) と同様に LB_PAA と称する．LB_PAA ではシーケンスの分割数を 16 に設定した．CPU 時間は SUN UltraSPARC-II 450 MHz において計測した．1 回の問合せにおいて探索するシーケンスの数は 20 であり，すなわち 20 近傍探索 ($k = 20$) である．すべての実験結果は 100 回の問合せ試行の平均をとっている．データ集合のサイズは 10 万件である．本論文では以下のような 2 種類のデータ集合を用いて実験を行った．

(1) *FinTime*
金融時系列データのベンチマーク，*FinTime*を利用した．10 万の有価証券に対する売買データを作成し，日々の終値を実験に用いた．

(2) *Randomwalk*
文献 1)，18) に従い，以下のようなランダムウォークモデルから 10 万シーケンスを作成した．

$$p_i = p_{i-1} + v_i$$

ここで，シーケンスの先頭要素 p_0 は，範囲 (0 : 10) からランダムに生成された値であり，v_i は分散を 1 とする正規分布に基づいて生成された値である．

提案手法については，長さ 1024 のシーケンスの場合，基準長が $l_1 = 4$，$l_2 = 16$，$l_3 = 64$，$l_4 = 256$ である 4 種類の PAA 表現，およびレベルが $r_1 = 0$，$r_2 = 2$，$r_3 = 4$，$r_4 = 6$ である 4 種類のウェーブレッ

ト係数を用いた．長さ 256 のシーケンスについては 3 種類の PAA 表現（l_1，l_2，l_3）と 3 種類のウェーブレット係数（r_1，r_2，r_3）を用いた．

4.1 探索性能

*FinTime*および *Randomwalk*それぞれについて索引を構築した．データ集合サイズは，25,000 から 100,000 まで変化させた．我々は探索性能を CPU 時間に基づいて評価する．なぜなら，DTW による探索では，CPU 時間がディスクアクセスに要する時間を大幅に上回り，探索コストは主として CPU 時間に依存するためである．

図 6 は，シーケンス長 $n = 256$ と $n = 1024$ に関する CPU 時間による探索性能の比較である．図 6 を含め，CPU 時間に関する図はすべて y 軸が対数目盛になっている．**図 7** は，$n = 1024$ の *FinTime*に関するシーケンスアクセス数を示している．図 6 は，すべてのデータ集合について，我々の手法が探索コストを大幅に削減していることを示している．図 7 に示しているように，提案手法は非常に少ないシーケンスアクセス数を示している．すなわち，厳密な DTW 距離の計算回数を大幅に低減させており，この結果 CPU 時間の低減につながっている．データ集合サイズが大きくなるほど，もしくはシーケンス長が長くなるほど提案手法の優位性は高まる．これは，大規模で長いシーケンスの時系列データベースにとって，提案手法がより有効であることを示している．具体的に実験では，提案手法は既存手法と比べ，最大で *FinTime*を用いた場合で約 54 倍，*Randomwalk*では約 51 倍の性能向上を達成した．

4.2 ワーピング範囲の変化に対する探索性能

近似手法 LB_PAA では，全体的なパス制約を導入することによってワーピングの幅が縮小し，効率的な探索を行っている．提案手法も制約を与えることによって，効率をより高めることができる．我々は全体的なパス制約を導入し，ワーピングの幅を変化させたときの探索性能を比較した．全体的なパス制約として，迫江と千葉による制約，Sakoe-Chiba Band [15)]を用いる．ワーピングの幅は，シーケンス長の 10%から 100%まで変化させている．近似距離計算だけでなく，厳密な DTW 距離計算についても，この制約に則って実行した．**図 8** は 10 万件のデータ集合を用いた際の LB_PAA と提案手法との比較である．両手法とも，ワーピングの幅が縮小するに従い CPU 時間が少なくなり，より効率的になっている．ただ，提案手法はいずれの幅においても既存手法と比べ，探索コストを大幅に低減させている．

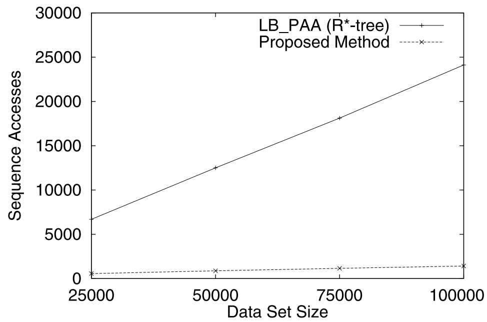


図 7 シーケンスアクセス数
Fig. 7 Sequence accesses.

4.3 異なる長さのシーケンス集合に対する探索性能

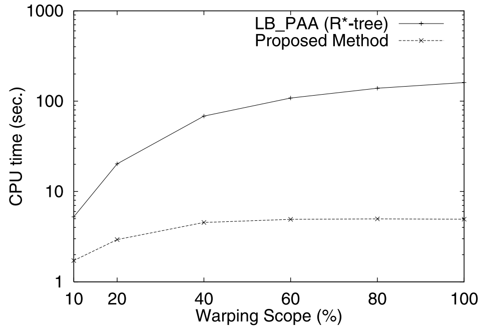
提案手法は，異なる長さのシーケンス集合においても効率的に探索処理を行うことができる．本実験では，$Random(1024, 16)$，$Random(1024, 32)$，$Random(1024, 64)$，$Random(1024, 128)$ という 4 つのデータ集合を用いた．ここで，$Random(n_{ave}, n_{diff})$ はランダム関数であり，n_{ave} はデータ集合におけるシーケンス長の平均である．n_{diff} は，様々な長さのシーケンスを含むデータ集合の中で，シーケンス長の最大値と最小値の差を意味する．すべてのデータ集合のサイズは 10 万件であり，$n_{ave} = 1024$ である．

図 9 は，*FinTime*を用いた実験結果を示している．提案手法の探索時間と索引を用いずに全数探索を実施した場合の時間を比較している．明らかに，すべてのデータ集合において提案手法が有効であることが分かる．提案手法は n_{diff} が大きくなっても，優れた探索性能を示している．

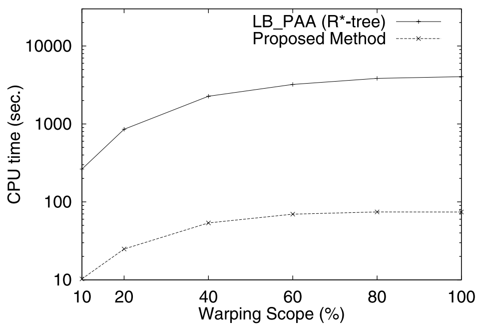
5. むすび

本論文では，DTW に基づく類似探索を高速化するための手法について述べた．まず，類似問合せ処理において距離近似が探索漏れを起こさないことを保証するための必要十分条件を提案した．従来の類似探索手法は，下界の性質を用いて探索漏れが発生しないことを保証していたが，これは必要条件ではなかった．本論文では必要十分条件となる新たな性質を示し，この性質に則った近似手法を提案した．

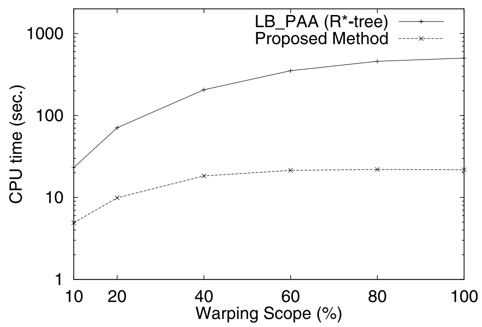
近似手法は効率的に不要なワーピングパスを取り除き，ワーピングの範囲を削減した後，近似距離を求める．この近似手法を利用した探索アルゴリズムは，シーケンスの DTW 距離を効率的かつ精密に近似し，類似したシーケンスを高速に収集することができる．提案手法は，特にデータ集合サイズが大きくなるほど，もしくはシーケンスが長くなるほど効率的になる．さらに提案手法は，シーケンス長が統一されたデータ集



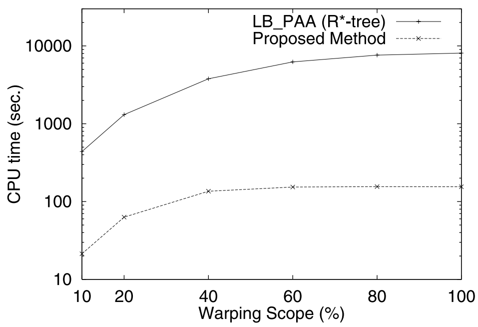
(a) *FinTime* ($n = 256$)



(b) *FinTime* ($n = 1024$)



(c) *Randomwalk* ($n = 256$)



(d) *Randomwalk* ($n = 1024$)

図 8　ワーピングの幅を変化させたときの CPU 時間
Fig. 8　CPU time vs. width of warping.

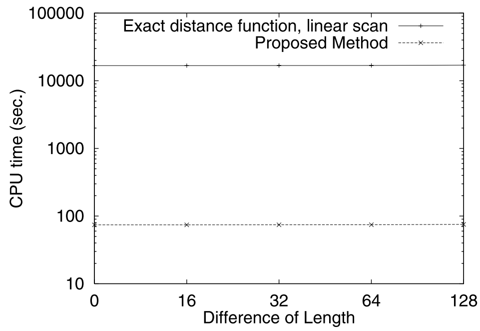


図 9　異なるシーケンス長のデータ集合に対する探索時間
Fig. 9　CPU time vs. difference of sequence length.

合だけでなく，長さの異なるシーケンスを含むデータ集合を扱うことができる．実験では，既存手法と比べ最大で約 54 倍の性能向上を達成し，提案手法の優位性が明らかとなった．

参 考 文 献

1) Agrawal, R., Faloutsos, C. and Swami, A.N.: Efficient Similarity Search In Sequence Databases, *Proc. 4th Conference on Foundations of Data Organization and Algorithms (FODO)*, pp.69–84 (Feb. 1993).

2) Agrawal, R., Lin, K.-I., Sawhney, H.S. and Shim, K.: Fast Similarity Search in the Presence of Noise, Scaling and Translation in Time-Series Databases, *Proc. VLDB*, pp.490–501 (Sept. 1995).

3) Ankerst, M., Braunmüller, B., Kriegel, H.-P. and Seidl, T.: Improving Adaptable Similarity Query Processing by Using Approximations, *Proc. VLDB*, pp.206–217 (Aug. 1998).

4) Beckmann, N., Kriegel, H.-P., Schneider, R. and Seeger, B.: The R*-tree: An Efficient and Robust Access Method for Points and Rectangles, *Proc. ACM SIGMOD*, pp.322–331 (May 1990).

5) Berndt, D.J. and Clifford, J.: Finding Patterns in Time Series: A Dynamic Programming Approach, *Advances in Knowledge Discovery and Data Mining*, pp.229–248, AAAI/MIT (1996).

6) Guha, S., Jagadish, H.V., Koudas, N., Srivastava, D. and Yu, T.: Approximate XML joins, *Proc. ACM SIGMOD*, pp.287–298 (June 2002).

7) Jang, J.-S.R. and Lee, H.-R.: Hierarchical Filtering Method for Content-based Music Re-

trieval via Acoustic Input, *Proc. ACM Multimedia*, pp.401–410 (Sept./Oct. 2001).
8) Kahveci, T. and Singh, A.K. An Efficient Index Structure for String Databases, *Proc. VLDB*, pp.351–360 (Sept. 2001).
9) Keogh, E.J.: Exact Indexing of Dynamic Time Warping, *Proc. VLDB*, pp.406–417 (Aug. 2002).
10) Keogh, E.J., Chakrabarti, K., Mehrotra, S. and Pazzani, M.J.: Locally Adaptive Dimensionality Reduction for Indexing Large Time Series Databases, *Proc. ACM SIGMOD*, pp.151–162 (May 2001).
11) Keogh, E.J., Chakrabarti, K., Pazzani, M.J. and Mehrotra, S.: Dimensionality Reduction for Fast Similarity Search in Large Time Series Databases, *Journal of Knowledge and Information Systems*, pp.263–286 (2000).
12) Kim, S.-W., Park, S. and Chu, W.W.: An Index-Based Approach for Similarity Search Supporting Time Warping in Large Sequence Databases, *Proc. ICDE*, pp.607–614 (April 2001).
13) Mount, D.W.: *Bioinfomatics: Sequence and Genome Analysis*, Cold Spring Harbor, New York (2000).
14) Otsuka, K., Horikoshi, T., Suzuki, S. and Kojima, H.: Memory-Based Forecasting for Weather Image Patterns, *Proc. 17th Conference on Artificial Intelligence (AAAI)*, pp.330–336 (July 2000).
15) Rabinar, L. and Juang, B.-H.: *Fundamentals of Speech Recognition*, Englewood Cliffs, N.J. (1993).
16) Sakurai, Y., Yoshikawa, M., Kataoka, R. and Uemura, S.: Similarity Search for Adaptive Ellipsoid Queries Using Spatial Transformation, *Proc. VLDB*, pp.231–240 (Sept. 2001).
17) Wickerhauser, M.V.: *Adapted Wavelet Analysis from Theory to Software*, A K Peters Ltd, Massachusetts (1994).
18) Yi, B.-K. and Faloutsos, C.: Fast Time Sequence Indexing for Arbitrary Lp Norms, *Proc. VLDB*, pp.385–394 (Sept. 2000).
19) Yi, B.-K., Jagadish, H.V. and Faloutsos, C.: Efficient Retrieval of Similar Time Sequences Under Time Warping, *Proc. ICDE*, pp.201–208 (Feb. 1998).
20) Zhu, Y. and Shasha, D.: Warping Indexes with Envelope Transforms for Query by Humming, *Proc. ACM SIGMOD*, pp.181–192 (June 2003).

(平成 15 年 9 月 20 日受付)
(平成 16 年 1 月 7 日採録)

（担当編集委員　片山 紀生）

櫻井　保志（正会員）

1991 年同志社大学工学部電気工学科卒業．同年日本電信電話株式会社入社．1996 年奈良先端科学技術大学院大学情報科学研究科博士前期課程修了．1999 年同大学院博士後期課程修了．工学博士．現在，NTT サイバースペース研究所に所属．2004 年よりカーネギーメロン大学客員研究員（2005 年までを予定）．索引技術，情報検索に関する研究開発に従事．

吉川　正俊（正会員）

1980 年京都大学工学部情報工学科卒業．1985 年同大学院工学研究科博士後期課程修了．工学博士．同年京都産業大学計算機科学研究所講師．同大学工学部助教授を経て 1993 年より奈良先端科学技術大学院大学情報科学研究科助教授，2002 年より名古屋大学情報連携基盤センター教授，現在に至る．1989 年～1990 年南カリフォルニア大学客員研究員，1996 年～1997 年ウォータルー大学客員准教授．XML データベース，多次元空間索引等の研究に従事．電子情報通信学会，ACM，IEEE Computer Society 各会員．日本データベース学会理事．