社団法人 情報処理学会 研究報告
IPSJ SIG Technical Report

2008－DBS－146 (47)
2008／9／23

ダイナミックプログラミングに基づくストリームマッチング

豊田 真智子† 櫻井 保志†† 市川 俊一†

† 日本電信電話株式会社 NTT 情報流通プラットフォーム研究所
〒 180–8585 東京都武蔵野市緑町 3–9–11
†† 日本電信電話株式会社 NTT コミュニケーション科学基礎研究所
〒 619–0237 京都府相楽郡精華町光台 2-4
E-mail: †{toyoda.machiko,ichikawa.toshikazu}@lab.ntt.co.jp, ††yasushi.sakurai@acm.org

あらまし データストリームはさまざまな分野で発生するデータであり，これらのデータの処理が重要となっている．本稿では，ダイナミックプログラミングに基づいて，データストリームから類似する部分シーケンスのペアを検出する問題に焦点を当てる．ストリームのサンプリングレートの違いや周期の変化を考慮した場合，時間スケールを扱う距離尺度が必要となる．従来，時間スケールを扱う距離尺度としてダイナミックタイムワーピング (DTW: Dynamic Time Warping) が用いられてきたが，主に蓄積されたデータを対象としているため，ストリーム処理に適用した場合に大幅な計算コストを必要とする．本稿では，データストリームにおける類似度を測定するために，新たな距離尺度を提案し，それを用いた効率的なアルゴリズムを示す．実データと人工データを用いた実験では，従来手法と比較して，大幅な性能向上を達成していることが確認された．
キーワード データストリーム，ダイナミックプログラミング，タイムワーピング，シーケンスマッチング

Stream Matching based on Dynamic Programming

Machiko TOYODA†, Yasushi SAKURAI††, and Toshikazu ICHIKAWA†

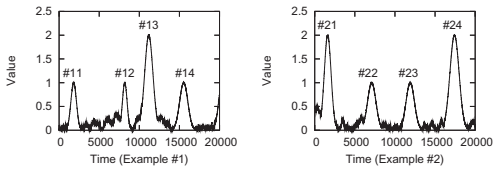
† NTT Information Sharing Platform laboratories, NTT Corpration
9–11, Midori-cho 3-Chome Musashino-shi, Tokyo, 180–8585 Japan
†† NTT Communication Science Laboratories, NTT Corporation
2–4, Hikaridai, Seika-cho, Soraku-gun, Kyoto, 619–0237 Japan
E-mail: †{toyoda.machiko,ichikawa.toshikazu}@lab.ntt.co.jp, ††yasushi.sakurai@acm.org

Abstract Data stream are generated in several domain, and the processing is becoming increasingly important. In this paper, we focus on the problem of finding pairs of similar subsequences from data streams based on dynamic programming. Consider the difference of streams sampling rates and the variation of time period, a similarity measure which addresses time-scaling is important. DTW is a popular similarity measure and can consider time-scaling. However, since it have used for stored data set, it requires drastic time for data streams. In this paper, we propose a similarity method to measure similarity on data streams, and show an efficient algorithm with the similarity method. Our experiments on real and synthetic data sets demonstrated that our method is dramatically faster than the existing method.
Key words Data stream, Dynamic programming, Time warping, Sequence matching

1. まえがき

データストリームは金融やセンサネットワーク管理，製造，ネットワークモニタリングなどのさまざまな分野において発生し，データストリームの処理やマイニングがますます重要となってきている．データストリームのサイズは無限長であり，連続的にオンラインでデータが到着するため，すべてのデータを保存することは困難である．さらにデータストリームは，1：ストリームのサンプリングレートはアプリケーションに応じて決定されるためストリーム間で異なる場合があり，それらの時間周期が変化する．2：データは時間の経過と共に発達する可能性があるため，学習セットや事前定義モデルをデータストリームに適用することが難しい．という特徴を持っている．

データストリームの管理は，データベース技術を拡張した技術の 1 つとして位置づけられる．類似シーケンス検出は，連続的にデータストリームを監視するために要求される技術である．理想的には，任意のデータストリームに対する類似シーケンス



(a) シーケンス #1　(b) シーケンス #2

図 1　Cross-similarity

検出は次のような要求を満たすことが望ましい．

（1） Time-scaling：従来，類似する部分シーケンスを検出する研究は，固定の部分シーケンス長を想定しているものが多い．この制限は，順応性のある部分シーケンスマッチング処理には適していない．例えば，センサネットワークにおいては，同じ動作や振舞いを表す周期は動作の違いによって異なる場合がある．そのため，固定した周期に注目することは，類似する部分シーケンスペアを見逃す可能性がある．

（2） Data-adaptive：類似部分シーケンスはそのデータ自身から直接検出されることが望ましい．事前にモデルを定義した場合には，新たなパターンの存在に気づかない可能性がある．

本稿では，データストリームから類似する部分シーケンスペアやそれらの周期性を検出する問題に焦点を当てる．我々が解決したい問題は，以下の通りである．

［問題 1］（Cross-similarity） 2 つのデータストリームが与えられたとき，類似する部分シーケンスのペアやそれらの周期性を DTW 距離に基づいて検出する．

この問題を図 1 を用いて説明する．シーケンス# 1 には 3 つの小さなスパイク (# 11, # 12, # 14) が，シーケンス# 2 には 2 つの小さなスパイク (# 22, # 23) が存在する．これらスパイクの振幅はほぼ同じであるが，周期は異なっている．一方，シーケンス# 1 と# 2 には 3 つの大きなスパイク (# 13, # 21, # 24) が存在し，これらの周期も異なっている．直感的に cross-similarity は 2 つのシーケンス間の部分的な類似を意味する．例えば，部分シーケンスペア# 11 と# 22，# 11 と# 23，# 13 と# 21，# 13 と# 24 はシーケンス# 1 と# 2 の cross-similarity となる．ここで，DTW 距離とはシーケンス間の類似度を判断するための距離尺度のことであり，3.1 節で詳しく述べる．

問題の詳細な定義は，3.2 節で述べるが，Naive な手法は cross-similarity を検出するために $O(nm^2 + n^2m)$ の時間とメモリ空間を必要とする．本稿では，このようなコストを劇的に低減できることを示す．

本稿の貢献は以下の通りである．

- データストリームにおいて cross-similarity を検出するための手法を提案する．本手法により，類似する部分シーケンスのペアを特定するだけでなく，それらの間の周期性も発見される．また，本手法は高速かつ低メモリでデータストリームを処理することが可能である．
- 実データと人工データを使った実験を行い，提案手法が cross-similarity を的確に検出していることを示す．また，理論的な分析の通り，提案手法が従来手法に比べて大幅に性能を改善することを示す．

2. 関 連 研 究

データベースコミュニティでは，長年に亘って時系列データベースに対する類似クエリの問題が研究されてきており，これらの問題をデータストリームに拡張した研究が多く取り組まれている [1] [4] [5] [8] [12]．例えば，Zhou らは DTW 距離に基づき，オンラインで部分シーケンスを検出するための効率的な手法を提案している [11]．櫻井らは DTW 距離に基づいて効率的に複数の数値ストリームをモニタする SPRING を提案している [7]．Tang らは，ある周期で繰り返し現れるが，連続する周期間には小さな違いが存在するという *pseudo periodical* ストリームにおけるストリームの変化を検出している [10]．Papadimitriou らはデータストリームにおける主なトレンドを簡潔に表す，最適ローカルパターンを発見する手法を提案している [6]

しかしながら，上記に挙げた文献において，冒頭で述べた問題を扱っている研究はない．

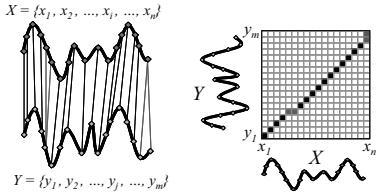


図 2　DTW を用いたシーケンスマッチング

3. 問 題 定 義

本章では，シーケンス間の類似判定手法であるダイナミックタイムワーピング (DTW) について述べ，問題を客観的に定義する．

3.1 ダイナミックタイムワーピング：DTW

ダイナミックタイムワーピング (DTW) は時間スケールを考慮した有用な距離判定手法の 1 つである．DTW は，2 つのシーケンス間の距離を最小化するように時間軸方向に伸長を行い，各要素同士をマッチングさせた計算により距離値を求め，類似度を判定する．この距離値は DTW 距離と呼ばれ，最適にシーケンス長を調整した後の距離の合計値で表される．この値が小さいほど 2 つのシーケンスは類似度が高く，0 の場合は完全に一致していることを意味する．動的計画法により，図 2 の右図のようなマトリックスを用いて計算され，色付けされたパスは DTW により対応付けられた要素を示しており，2 つのシーケンス間の最適なワーピングパスを表している．よりフォーマルには，以下の通りである．

長さ n のシーケンス $X = (x_1, x_2, ..., x_n)$ と長さ m のシーケンス $Y = (y_1, y_2, ..., y_m)$ を考える．これらの DTW 距離 $D(X,Y)$ は次のように定義される．

$$D(X,Y) = f(n,m)$$

$$f(i,j) = \|x_i - y_j\| + min \begin{cases} f(i,j-1) \\ f(i-1,j) \\ f(i-1,j-1) \end{cases} \quad (1)$$

$$f(0,0) = 0, \quad f(i,0) = f(0,j) = \infty$$

$$(i = 1, ..., n;\ j = 1, ..., m)$$

ここで，$\|x_i - y_j\| = (x_i - y_j)^2$ は 2 つの数値の距離を表す．この距離は，L1 距離 ($\|x_i - y_j\| = |x_i - y_j|$) など他の距離を用いてもかまわない．提案するアルゴリズムは，この選択とは完全に独立したものである．

3.2 Cross-similarity

データストリーム X は $x_1, x_2, ..., x_i, ..., x_n, ...$ の値，データストリーム Y は $y_1, y_2, ..., y_j, ..., y_m, ...$ から成る半無限長のシーケンスである．ここで x_n，y_m は最新の値であり，時間と共に n，m は増加する．$X[i_s : i_e]$ は i_s から始まり i_e で終わる X の部分シーケンスを意味し，$Y[j_s : j_e]$ は j_s から始まり j_e で終わる Y の部分シーケンスを意味する．L_w は $X[i_s : i_e]$ と $Y[j_s : j_e]$ 間の最適なワーピングパスであり，$L_x \leqq L_w$, $L_y \leqq L_w$, $L_w < L_x + L_y$ を満たす．我々が解決したい問題は，以下のように定義される．

［定義 1］（Cross-similarity） 2 つのストリーム X と Y, 閾値 ε, 類似する部分シーケンスの最小の長さ ζ が与えられたとき，DTW 距離に基づいて次の条件を満たす類似部分シーケンスのペア $X[i_s : i_e]$ と $Y[j_s : j_e]$ を検出する．

（1） $D(X[i_s : i_e], Y[j_s : j_e]) \leqq \varepsilon L_w$

（2） $L_x \geqq \zeta$ and $L_y \geqq \zeta$

我々のアプローチが従来の時系列データ処理と異なる点は，我々のアルゴリズムが DTW 距離に基づいてデータストリーム間の部分的な類似性を見つけるように設計されていることである．このため，我々のアルゴリズムはサンプリングレートが異なる，または，時間周期が変化するようなシーケンスに対してロバストである．本稿では，低メモリと高速処理が要求されるデータストリーム処理としてこの問題を解決する．

4. 提 案 手 法

本章では，データストリームにおける cross-similarity のための新たな手法を提案する．まず，DTW を用いた cross-similarity のための手法を示し，提案手法について述べる．

4.1 Naive

Cross-similarity 問題のための最も素直な方法は，考えられる X の部分シーケンスと Y の部分シーケンスそれぞれについて，すべての DTW 距離を計算することである．すなわち，計算で使用されるタイムワーピングマトリックスの数はシーケンスの伸張と共に大幅に増加する．本論文ではこの方法を Naive な手法と呼ぶ．

X 方向の i と Y 方向の j から始まる X と Y のタイムワーピングマトリックスにおいて，要素 (k, l) の距離値 $d_{i,j}(k, l)$ を考える．X と Y 間の DTW 距離は次のように計算される．

$$
\begin{aligned}
&D(X[i_s : i_e], Y[j_s : j_e]) \\
&\qquad = d_{i_s,j_s}(i_e - i_s + 1, j_e - j_s + 1) \\
&d_{i,j}(k,l) = \|x_{i+k-1} - y_{j+l-1}\| + min \begin{cases} d_{i,j}(k, l-1) \\ d_{i,j}(k-1, l) \\ d_{i,j}(k-1, l-1) \end{cases} \\
&d_{i,j}(0,0) = 0, \quad d_{i,j}(k,0) = d_{i,j}(0,l) = \infty \\
&\quad (i = 1, ..., n;\ k = 1, ..., n-i+1; \\
&\quad\ j = 1, ..., m;\ l = 1, ..., m-j+1)
\end{aligned}
\tag{2}
$$

部分シーケンス $X[i_s : i_e]$ と $Y[j_s : j_e]$ の類似度評価には，平均距離 d' を用いる．

$$
d' = \frac{d_{i_s,j_s}(i_e - i_s + 1, j_e - j_s + 1)}{L_w} \tag{3}
$$

ここで，L_w は $X[i_s : i_e]$ と $Y[j_s : j_e]$ 間の最適なワーピングパスである．DTW は固定長のクエリシーケンスに類似するシーケンスを検出するための手法であるため，Naive な手法のアルゴリズムは，$O(nm)$ のタイムワーピングマトリックスを用いて $O(nm^2 + n^2m)$ 個の値を更新する必要がある．また，アルゴリズムはタイムワーピング行列の中の 2 列 (すなわち，現在の列と直前の列) のみを用いて距離計算を行い，メモリ使用量は $O(nm^2 + n^2m)$ となる．

4.2 提案手法の概要

前節で述べたように，Naive な手法では $O(nm)$ のタイムワーピングマトリックスを必要とするため，データストリーム処理に適していない．提案手法は，cross-similarity を検出するために 1 つのマトリックスしか必要とせず，(1) cross-similarity を評価するための新しいスコアリング関数，(2) スコアリング関数を用いたストリーム処理アルゴリズム，の 2 つから成り立つものである．以降で提案手法の詳細を述べる．

スコアリング関数は cross-similarity のための動的計画法に基づいた類似判定手法である．部分シーケンス間の類似度は，2 つのシーケンスを適切に伸張した後にスコアとして計算される．このスコアは，累積スコアとして計算され，”最大累積スコア”を求めることによって最終的なマッチング (すなわち cross-similarity) を獲得する．スコアリングマトリックスのセル (i, j) には，X の i 番目の要素と Y の j 番目の要素のベストスコア $s(i, j)$ の値が含まれる．ここで，もしスコア $s(i, j)$ がマイナス値となった場合には，この値が 0 で置き換えられる，”ゼロリセット”を導入する．このアプローチは，バイオインフォマティックス分野において提案されているものである (例えば，Smith-Waterman アルゴリズムが挙げられる [9])．バイオインフォマティクス分野では，記号シーケンスが扱われているが，我々のスコアリング関数は数値シーケンスを扱うものである．$s(i, j)$ がゼロで置き換えられるということは，(i, j) で終了する X と Y の部分シーケンスがもはや cross-similarity の定義 (1) を満たさないことを意味しており，ここから新たなマッチングが開始される．0 が続く区間は，部分シーケンスペアがまったく似ていないことを表している．

［知見 1］ ゼロリセットを導入することにより，部分的に類似するシーケンスペアを検出することができる．

次のステップとして，スコアリングマトリックスにおいて類似する部分シーケンス $X[i_s : i_e]$ と $Y[j_s : j_e]$ の位置，特に，開始点である i_s と j_s の位置を特定する．蓄積されたシーケンスの場合，類似部分シーケンスの終了位置からワーピングパスを逆にたどることによりこれらの開始点を容易に特定することができる．ストリーム処理において開始点を見失わないようにするため，スコアリングマトリックスにおける開始点の情報 $p(i, j)$ を保持する．これは，スコアリングマトリックスのセル (i, j) には，X の i 番目の要素と Y の j 番目の要素のベストスコアである $s(i, j)$ の値と，$s(i, j)$ に対応する開始点 $p(i, j)$ が保存されることを意味する．開始点 $p(i, j)$ は座標値として表される．例えば，部分シーケンス $X[i_s : i_e]$ と $Y[j_s : j_e]$ における開始点 $p(i_e, j_e)$ は (i_s, j_s) となる．スコアリングマトリックスの例は後ほど説明する (図 4)．

［知見 2］ スコアリングマトリックスには，類似する部分シーケンスペアの累積スコアと各部分シーケンスの開始点が保持される．これにより，ストリーム処理で類似部分シーケンスペアを特定することができる．

最後に，スコアリング関数の結果を散布図を用いて可視化する．この散布図は，類似部分シーケンスペア $X[i_s : i_e]$ と $Y[j_s : j_e]$ のマッチング要素 i_e，j_e を特定し，X と Y 間の cross-similarity が反映される．散布図における横軸は X の要

素を，縦軸は Y の要素を表す．具体的には，部分シーケンスペア $X[i_s:i_e]$ と $Y[j_s:j_e]$ が類似している場合，図における位置 (i_e, j_e) が描画される．

［知見 3］ 散布図を用いることにより，cross-similarity と周期性の存在を特定することができる．

4.3 アルゴリズム

データストリーム $X = (x_1, ..., x_i, ..., x_n, ...)$ と $Y = (y_1, ..., y_j, ..., y_m, ...)$ が与えられたとき，部分シーケンスペア $X[i_s:i_e]$ と $Y[j_s:j_e]$ のスコア $S(X[i_s:i_e], Y[j_s:j_e])$ は次のように計算される．

$$
\begin{aligned}
&S(X[i_s:i_e], Y[j_s:j_e]) = s(i_e, j_e)\\
&s(i,j) = max \begin{cases} 0 \\ 2\varepsilon - ||x_i - y_j|| + s_{best} \end{cases}\\
&s_{best} = max \begin{cases} s(i, j-1) \\ s(i-1, j) \\ s(i-1, j-1) \end{cases}\\
&s(0,0) = s(i,0) = s(0,j) = 0
\end{aligned}
\tag{4}
$$

$X[i_s:i_e]$ と $Y[j_s:j_e]$ の類似度評価には，平均スコア s' を用いる．

$$
s' = \frac{s(i_e, j_e)}{L_w} \tag{5}
$$

セル (i,j) の開始点 $p(i,j)$ は次のように計算される．

$$
p(i,j) = \begin{cases} p(i, j-1) & (s_{best} = s(i, j-1)) \\ p(i-1, j) & (s_{best} = s(i-1, j)) \\ p(i-1, j-1) & (s_{best} = s(i-1, j-1)) \\ (i,j) & (s_{best} = 0) \end{cases} \tag{6}
$$

$S(X[i_s:i_e], Y[j_s:j_e])$ の開始点は以下のように求める．

$$
(i_s, j_s) = p(i_e, j_e) \tag{7}
$$

図 3 はスコアリング関数を用いたアルゴリズムを示している．新たなデータが到着するとき，インクリメンタルにスコア $s(i,j)$ を更新し，$s(i,j)$ に対応する開始点 $p(i,j)$ を決定する．部分シーケンスペア $X[i_s:i_e]$ と $Y[j_s:j_e]$ の特定には，式 (4)(5)(6)(7) を用いる．もし条件 $s' \geqq \varepsilon$，$L_x \geqq \zeta$，$L_y \geqq \zeta$ が満たされた場合，最適な部分シーケンスペアであるとしてそのペアを報告する．この図では，時刻 i で x_i を受け取った場合のスコアの計算を表している．時刻 j で y_j を受け取った場合のスコアの計算も，このアルゴリズムによって同様に計算される．

［補題 1］ データストリーム X と Y が与えられたとき，$X[i_s:i_e]$ と $Y[j_s:j_e]$ が定義 1 の最初の条件を満たすなら，$s(i_e, j_e) \geqq \varepsilon L_w$ が成り立つ．

［証明 1］ 定義 1 の最初の条件は，

$$
d_{i_s, j_s}(i_e - i_s + 1, j_e - j_s + 1) \leqq \varepsilon L_w
$$

となる．この式を変形すると，以下の式が成り立つ．

$$
2\varepsilon L_w - d_{i_s, j_s}(i_e - i_s + 1, j_e - j_s + 1) \geqq \varepsilon L_w.
$$

$X[i_s:i_e]$ と $Y[j_s:j_e]$ は条件を満たすので，式 (4) から，

Algorithm
Input: a new value x_i at time-tick i
Output: qualifying subsequence pairs and distances

```
1.  for j = 1 to m do
2.    Compute s(n,j) by Equation (4);
3.    Compute s' by Equation (5);
4.    Compute p(n,j) by Equation (6);
5.    if s' ≥ ε ∧ L_x ≥ ζ ∧ L_y ≥ ζ then
6.      (i_s, j_s) := p(n,j);  i_e := n;  j_e := j;
7.      Report (i_s, i_e, j_s, j_e, s');
8.    end if
9.  end for
```

図 3 cross-similarity のための提案アルゴリズム

7	5	6 (1, 7)	0 (1, 7)	4.25 (1, 4)	0 (1, 4)	3.5 (1, 4)	3.86 (1, 4)	2.13 (1, 4)
6	8	1 (1, 6)	0 (1, 4)	4 (1, 4)	3.5 (1, 4)	3.2 (1, 4)	2.17 (1, 4)	0 (1, 4)
5	13	0 (1, 4)	5 (1, 4)	0 (1, 4)	0.6 (2, 1)	0 (2, 1)	0 (2, 1)	0 (2, 1)
4	4	5 (1, 2)	0 (1, 4)	1.5 (2, 1)	0 (2, 1)	4.5 (2, 1)	4.6 (2, 1)	3.33 (2, 1)
3	9	0 (1, 2)	1 (1, 2)	2.67 (2, 1)	4 (2, 1)	2.6 (2, 1)	1.5 (4, 1)	0 (4, 1)
2	6	5 (1, 2)	0 (1, 2)	5.5 (2, 1)	0.33 (2, 1)	5.5 (4, 1)	5.33 (4, 1)	0 (4, 1)
1	11	0 (1, 1)	5 (2, 1)	0 (2, 1)	5 (4, 1)	0 (4, 1)	0 (6, 1)	0 (7, 1)
j	y_j / x_i	5	12	6	10	6	5	1
	i	1	2	3	4	5	6	7

図 4 スコアリング関数の例

$$
s(i_e, j_e) = 2\varepsilon L_w - d_{i_s, j_s}(i_e - i_s + 1, j_e - j_s + 1).
$$

を得る．以上より，

$$
s(i_e, j_e) \geqq \varepsilon L_w
$$

が成り立つ． □

図 4 は提案アルゴリズムの詳細を示している．$X = (5, 12, 6, 10, 6, 5, 1),$，$Y = (11, 6, 9, 4, 13, 8, 5),$，$\varepsilon = 3$，$\zeta = 4$ とする．セル (i,j) には，スコア $s(i,j)$ と開始点 $p(i,j)$ が保存されるが，説明を簡単にするため，この図ではスコア $s(i,j)$ の代わりに平均スコア s' の値が記されている．色付けされたセルは，類似する部分シーケンスペアを表す．例えば，セル $(6,4)$ は部分シーケンス $X[2:6]$ と $Y[1:4]$ の平均スコアが 4.6 であり，これらのシーケンスペアが条件を満たす，すなわち，類似していることを意味している．また散布図においては，これらの類似部分シーケンスペアの終了位置（すなわち，色付けされたセルの位置）が散布図の座標値に対応付けられて描画される．

4.4 計 算 量

X を伸張している長さ n のシーケンス，Y を伸張している長さ m のシーケンスとする．

［補題 2］ Naive な手法は，$O(nm^2 + n^2m)$ のメモリ量と $O(nm^2 + n^2m)$ の計算時間を要する．

［証明 2］ Naive な手法は $O(nm)$ 個のタイムワーピングマトリックスを用い，時刻 i において x_i を受け取った場合には $O(nm^2)$ 個の値を，時刻 j において y_j を受け取った場合には $O(n^2m)$ 個の値を類似する部分シーケンスペア特定のために更新する必要がある．そのため，$O(nm^2 + n^2m)$ の計算時間を要する．Naive な手法は，各マトリックス毎に m 個の配列 2 列と n 個の配列 2 列を保持するので，全体として $O(nm^2 + n^2m)$ のメモリ使用量を要する． □

［補題 3］ 提案手法は，$O(m+n)$ のメモリ使用量と $O(m+n)$ の計算時間を要する．

［証明 3］ 提案手法は単一のスコアマトリックスを用い，時刻 i において x_i を受け取った場合には $O(m)$ 個の値を，時刻 j において y_j を受け取った場合には $O(n)$ 個の値を類似する部分シーケンスペア特定のために更新する．そのため，$O(m+n)$ の計算時間と $O(m+n)$ のメモリ使用量を要する． □

5. 評価実験

提案手法の有効性を検証するため，実データと人工データを用いた実験を行った．実験で使用したマシンは，2GB のメモリと Intel Pentium4 3GHz の CPU を搭載した Linux マシンである．提案手法に対する比較手法には，Naive な手法と SPRING を用いた．SPRING は櫻井ら [7] によって提案され，データストリームから固定長のクエリシーケンスに類似する部分シーケンスを検出するための DTW に基づく手法である．SPRING は cross-similarity の検出を意図したものではないが，1 章で述べた問題を解決するための手法として適用することが可能である．SPRING は $O(m)$ 個のマトリックスを用い，時刻 i において x_i を受け取った場合には $O(m^2)$ 個の値を，時刻 j において y_j を受け取った場合には $O(nm)$ 個の値を更新するため，cross-similarity のための SPRING のアルゴリズムは，$O(m^2+nm)$ のメモリ使用量と $O(m^2+nm)$ の時間を要する．

5.1 cross-similarity の検出

cross-similarity の検出精度について，実データ 2 つと人工データ 1 つのデータセットを使用して評価した．

Spikes:

Spikes は図 5 (a)，(b) で示される人工データセットであり，大小のスパイクから構成される．スパイクとスパイクの間のデータは，ランダムウォーク関数を用いて異なる長さで生成し，各スパイクの周期も元なっている．*Spikes* を用いた実験結果は，図 5 (c)，(d) のようになった(注1)．これらの結果から，提案手法が大小のスパイクを完全に検出している様子が確認される．各スパイクの周期の違いは描画の長さの違いとして現れており，幅の広いスパイクは描画長が長く，幅の狭いスパイクは描画長が短い．

Temperature:

Temperature は温度センサから約 10 日間収集されたデータであり，観測値は摂氏で表されている．センサからは約 1 分ごとに計測値が送信されており，データセットは多くの時刻で計測値が欠けている．今回の実験で設定した ζ は 1800 であり，これは約半日に相当する値である．*Temperature #1* と *Temperature #2* では，天候によって約 18 度から 32 度までの大きな温度変化が観測されている．欠損値があるにもかかわらず，提案手法はこれらのパターンの検出に成功している (図 6 (c))．

Mocap:

多重ストリームへの拡張として，実データセット *Mocap* を用いた．*Mocap* は被験者が動作を行っている間の動作情報を記録することによって生成された実データセットである (動作例：walking, running, kicking など)．人体に取り付けたマーカーの各部位の角速度が，1 秒間に 120 回というサンプリング周期で測定されており，CMU のモーションキャプチャデータベースから取得したデータである(注2)．今回実験には，一番シンプルな手足のマーカーから取得された 8 個のデータを 8 次元データとして使用した．このデータには，walking モーションが複数回行われており，これらのデータの長さはそれぞれ異なっている．データの詳細は図 7 の通りである．今回の実験では，ζ を 240 に設定した．これは，約 2 秒の時間に相当する値である．

結果は図 8 のようになった．この図から，提案手法が walking モーションを完璧に検出していることが確認される．

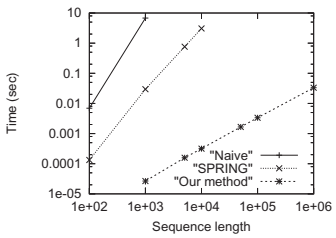


図 9 データシーケンス長に対する cross-similarity のための計算時間

5.2 性　　能

提案手法の効率性の評価と 4.4 節で述べた計算量を検証するための実験を行った．実験には，複数のサイン波から構成される人工データを用い，シーケンス長を変化させた場合の計算時間について，提案手法，Naive，SPRING の 3 手法を比較した．結果は，各時刻においてスコアリングマトリックスを更新する時間と cross-similarity を検出する時間を測定した．

我々が期待した通り，提案手法は Naive と SPRING よりも高速に cross-similarity を特定していることがわかる (図 9)．この傾向は，4.4 節における理論的な議論と合致しており，計算時間を劇的に削減することが示された．

6. ま と め

本稿では，データストリームにおける cross-similarity の問題を扱い，この問題を解決する手法を提案した．提案手法は 2 つのアイディアに基づいている．スコアリング関数：データストリームから，効率的に類似部分シーケンスペアを検出する．ストリームアルゴリズム：cross-similarity が可視化され，周期性を発見することができる．提案手法は cross-similarity の検出に単一のマトリックスしか使用しない．実データと人工データを用いて提案手法の検証を行った結果，Naive な手法に比べて大幅に性能を改善し，高速に処理できることが確認された．

(注1)：提案手法では、cross-similarity が 1 枚の散布図として描画されるが，本実験では結果をわかりやすくするために，大小それぞれのスパイクの cross-similarity を別々の散布図として描画した．

(注2)：http://mocap.cs.cmu.edu/

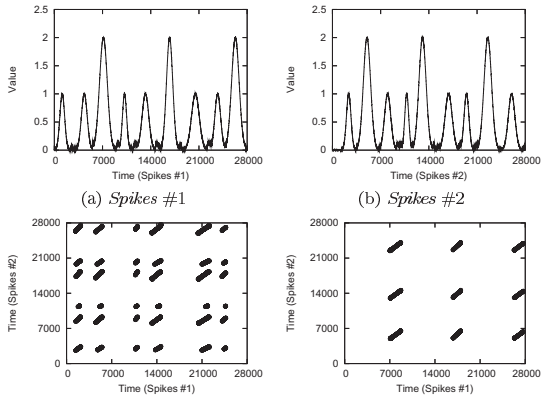
文　　献

[1] L. Gao, and X. S. Wang: “Continually Evaluating Similarity-Based Pattern Queries on a Streaming Time Series, ” *Proceedings of the 2002 ACM SIGMOD International Conference on Management of Data*, pp.370-381, Madison, Wisconsin, USA, June 2002.

[2] J.-S. R. Jang and H.-R. Lee: “Hierarchical Filtering Method for Content-based Music Retrieval via Acoustic Input, ” *Proceedings of the ninth ACM International Conference on Multimedia*, pp.401-410, Ottawa, Canada, September-October 2001.

[3] H. Kawasaki, T. Yatabe, K. Ikeuchi, and M. Sakauchi: “Automatic Modeling of a 3D City Map from Real-World Video, ” *Proceedings of the seventh ACM International Conference on Multimedia (Part 1)*, pp.11-18, Orlando, Florida, USA, October-November 1999.

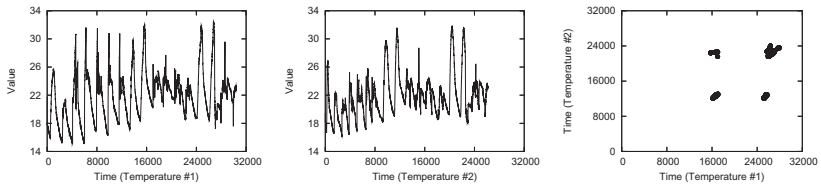
[4] S. Papadimitriou, A. Brockwell, and C. Faloutsos: “Adaptive, Hands-Off Stream Mining, ” *Proceedings of the 29th International Conference on Very Large Data Bases (VLDB 2003)*, pp.560-571, Berlin, Germany, September



(a) *Spikes* #1　(b) *Spikes* #2

(c) Cross-similarity (small spikes)　(d) Cross-similarity (large spikes)

図 5　*Spikes* を用いた cross-similarity の発見



(a) *Temperature* #1　(b) *Temperature* #2　(c) Cross-similarity

図 6　*Temperature* を用いた cross-similarity の発見

(a) *Mocap* #1

Sec.	Time-tick	Motions
0 - 8	0 - 960	walking
8 - 16	960 - 1920	running
16 - 20	1920 - 2400	jumping
20 - 29	2400 - 3480	walking
29 - 38	3480 - 4560	kicking
38 - 44	4560 - 5280	left-leg jumping
45 - 50	5400 - 6000	right-leg jumping
52 - 59	6240 - 7080	stretching
59 - 70	7080 - 8400	walking

(b) *Mocap* #2

Sec.	Time-tick	Motions
0 - 3	0 - 360	walking
4 - 9	480 - 1080	jumping
10 - 14	1200 - 1680	walking
16 - 20	1920 - 2400	punches
20 - 25	2400 - 3000	walking
26 - 33	3120 - 3960	kicking
33 - 38	3960 - 4560	punches

図 7　*Mocap* データセットの詳細

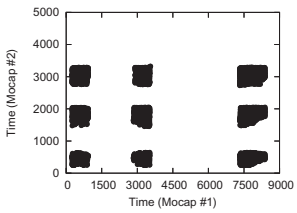


図 8　*Mocap* を用いた cross-similarity の発見

2003.

[5] S. Papadimitriou, J. Sun, and C. Faloutsos: "Streaming Pattern Discovery in Multiple Time-Series, " *Proceedings of the 31th International Conference on Very Large Data Bases (VLDB 2005)*, pp.697-708, Trondheim, Norway, August-September 2005.

[6] S. Papadimitriou, and P.S. Yu: "Optimal Multi-scale Patterns in Time Series Streams," *Proceedings of the 2006 ACM SIGMOD International Conference on Management of Data*, pp.647-658, Chicago, IL, USA, June 2006.

[7] Y. Sakurai, C. Faloutsos, and M. Yamamuro: "Stream Monitoring under the Time Warping Distance, " *Proceedings of IEEE 23rd International Conference on Data Engineering (ICDE 2007)*, pp.1046-1055, Istanbul, Turkey, April 2007.

[8] Y. Sakurai, S. Papadimitriou, and, C. Faloutsos: "Braid: Stream Mining through Group Lag Correlations," *Proceedings of the 2005 ACM SIGMOD International Conference on Management of Data*, pp.599-610, Baltimore, Maryland, June 2005.

[9] T. F. Smith, and M. S. Waterman: "Identification of Common Molecular Subsequences, " Journal of Molecular Biology 147, pp.195-197, 1981.

[10] L. Tang, B. Cui, H. Li, G. Miao, D. Yang, and X. Zhou: "Effective Variation Management for Pseudo Periodical Streams," *Proceedings of the 2007 ACM SIGMOD International Conference on Management of Data*, pp.257-268, Beijing, China, June 2007.

[11] M. Zhou, and M. H. Wong: "Efficient Online Subsequence Searching in Data Streams under Dynamic Time Warping Distance," *Proceedings of IEEE 24th International Conference on Data Engineering (ICDE2008)*, pp.686-695, Cancun, Mexico, April 2008.

[12] Y. Zhu, and D. Shasha: "StatStream: Statistical Monitoring of Thousands of Data Streams in Real Time," *Proceedings of the 28th International Conference on Very Large Data Bases (VLDB 2002)*, pp.358-369, Hong Kong, China, August 2002.