D-032

2 次元軌跡データに対する高速なパターン照合アルゴリズム
Faster pattern matching algorithm for 2-dimentional trajectory data

山本 雅大 * **栗田 和宏** * **笹川 裕人** * **有村 博紀** *
Masahiro Yamamoto Kazuhiro Kurita Hirohito Sasakawa Hiroki Arimura

1. はじめに

本論文では，2 次元平面上の軌跡データに対するパターン照合を考察する．初めに，L_∞ 距離に関する軌跡パターン照合問題を定式化し，次に，その問題に対する効率よい照合アルゴリズム Shift-AndTrajMatch を与える．提案アルゴリズムは，位置同定索引と，ビット並列法を組み合わせる事で，長さ m の軌跡パターンと長さ n の軌跡テキストに対し，$O(m \log m + \lceil \frac{m^2}{w} \rceil)$ 前処理時間と，$O(\lceil \frac{m^2}{w} \rceil)$ 領域を用いて，$O(n(\log m + \lceil \frac{m}{w} \rceil))$ 実行時間で軌跡パターン照合を行うことを示す．ここで w はワード長である．さらに，際限なく到着する軌跡データに対して限定されたメモリで照合を行えるストリーム型アルゴリズムである．最後に，今後の課題を述べる．

2. 準備

本節では，軌跡パターン照合に関する定義を与える．以下に含まれない用語については，アルゴリズム一般に関しては教科書 [3] を，文字列照合に関しては教科書 [5]，計算幾何学に関しては教科書 [2] を参照されたい．

2.1 基本的な定義

文献 [4] にしたがって，軌跡データを定義する．$\mathbb{R}^2$ を **2 次元平面**とし，$\mathbb{R}^2$ の**点**を $p = (x, y) \in \mathbb{R}^2$ とする．その **x 座標**と **y 座標**を，それぞれ，$x = p.x$ と $y = p.y$ と書く．点 p, q に対して，その 2 点間の L_∞ **距離**（以下では単に**距離**という）を $d(p, q)$ と書き，

$$d(p, q) = d_\infty(p, q) = max\{|p.x - q.x|, |p.y - q.y|\}$$

と定義する．$\mathbb{R}^2$ 上の長さ n の**点列**を $S = s_1 \dots s_n$ と書く．ここで，任意の $1 \le i \le n$ に対して，$S[i] = s_i \in \mathbb{R}^2$ である．

2.2 軌跡パターン照合問題

r を非負の実数とする．**軌跡テキスト**（もしくは**テキスト**）$T = t_1 \dots t_n$ と，**軌跡パターン**（もしくは**パターン**）$P = p_1 \dots t_m$ は，それぞれ $\mathbb{R}^2$ 上の長さ n と，m の点列である．軌跡パターン P が軌跡テキスト T に**最大距離 r で出現する**とは，ある位置 $1 \le k \le n$ が存在して，すべての $1 \le i \le m$ に対して，

$$d_\infty(p_i, t_{k+i-1}) \le r$$

が成立することをいう．このとき，位置 k を P の T における**出現位置**という．ここで，本稿で考察する軌跡パターン照合問題を以下のように定義する．

定義 1 軌跡パターン照合問題とは，軌跡テキスト T と軌跡パターン P に対し，P の T 中の出現位置を全て求める問題である．

図 1 に軌跡パターン照合問題の例を示す．ここで，$p_1 \dots p_6$ は，$t_2 \dots t_7$ に出現している．

*北海道大学， Hokkaido University

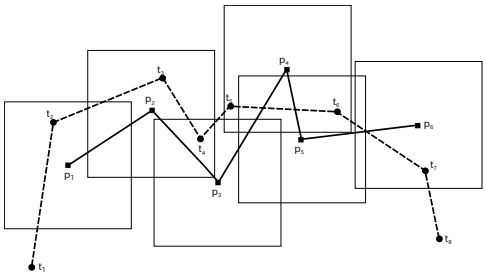

図 1: 軌跡パターン照合問題の例

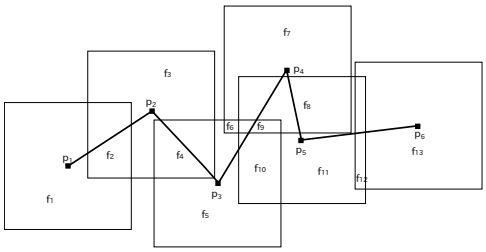

図 2: **配置** (arrangement)$\mathcal{A}(P)$ と面 f_k

3. アルゴリズム

本節では，軌跡パターン照合問題に対する効率良い照合アルゴリズムを与える．本節で提案するアルゴリズム Shift-AndTrajMatch は，ビット並列手法に基づく文字列照合アルゴリズム Shift-And 法 [1, 6] を軌跡パターン照合へと拡張したものである．

提案アルゴリズムを Algorithm 1 に示す．提案アルゴリズムは，パターンを表す非決定性有限オートマトン (NFA) をテキストの文字を読みながら遷移させることで照合を行うアルゴリズムである．このアルゴリズムでは，前処理として，入力パターンの NFA を表現する**ビットマスク集合** B を作成する．これは，Shift-And 法における 1 次元データに対するビットマスクを，2 次元平面上の領域に拡張したものである．探索処理では，ビットマスク集合 B を用い，ビット並列手法により効率良く探索処理を行う．以下で詳細を述べる．

前処理: 入力パターン $P = p_1 \dots p_m$ に対し，それらを中心とする半径 r の正方形の全体 $\mathcal{D}(P) = \{D_1, \dots, D_m\}$ を考える．ここに，各 $i = 1, \dots, m$ に対して，$D_i \in \mathcal{D}(P)$ は点 p_i を中心とする半径 r の正方形を表す．次に，$\mathcal{D}(P)$ の平面上への**配置** (arrangement)$\mathcal{A}(P)$ を考える（図 2 参照）．配置において，線分と線分の交点を**頂点** (vertex) と呼び，頂点と頂点を結ぶ線分を**辺** (edge)，辺で囲まれた領域を**面** (face) と呼ぶ．その際，以下の補題が成立する．

補題 1 P を 2 次元平面上の m 個の点からなる軌跡パターンとし，$\mathcal{A}(P)$ を P で決まる配置とする．このとき，

Copyright © 2014 by
The Institute of Electronics, Information and Communication Engineers and
Information Processing Society of Japan All rights reserved.

$\mathcal{A}(P)$ の頂点，辺，面の数はそれぞれ $O(m^2)$ 個である．

配置 $\mathcal{A}(P)$ により構成される面の集合を $\mathcal{F}(P)$ で表し，$\mathcal{F}(P) = \{f_1, \ldots, f_K\}$ $(K = O(m^2))$ とする．

次に，面の集合 $S := \mathcal{F}(P)$ を，次の演算をサポートする**位置同定** (point location) 索引 $\mathcal{PL}$ に格納する:

- **索引構築演算** $ConstructIndex(P)$: 全ての正方形の配置 $\mathcal{A}(P)$ を格納する．
- **位置同定演算** $PointLocation(p)$: 任意の点 $p \in \mathbb{R}^2$ を入力として，それを含む面 $f_k \in \mathcal{S}$ に対して，f_k を含む正方形の集合のビットマスク表現を返す．

配置中の面の数が $O(m^2)$ 個であるので，既存の一般的な位置同定索引 [2] を用いると，$\mathcal{PL}$ は，少なくとも $O(m^2)$ 領域を必要とする．これに対して，次の補題が示せる．

補題 2 二次元平面上の m 個の正方形の配置に対して，それを $O(\lceil \frac{m^2}{w} \rceil)$ 語の領域と $O(m \log m + \lceil \frac{m^2}{w} \rceil)$ 構築時間を用いて，位置同定演算を $O(\log m + \lceil \frac{m}{w} \rceil)$ 時間で実現する索引構造が存在する．

証明 基本的なアイデアは，m 個の正方形の配置 $\mathcal{A}(P)$ を x 軸と y 軸に射影して得られる区間の集合 $Seg_x(P)$ と $Seg_y(P)$ に分解し，各面をそれらの区間の直積の直和で表す事である．索引構築演算では，全正方形の両端点の x 座標を，$x_1 < \cdots < x_h (h \leq 2m)$ のように $O(m \log m)$ 時間でソートし，区間の集合 $Seg_x(P) = \{ I_x = [x_i, x_{i+1}) \mid i = 1, \ldots, h \}$ を計算する．ここに，$x_0 = -\infty, x_{h+1} = \infty$ である．次に，各区間 $s_x \in Seg_x(P)$ について，s_x を含む正方形の集合 $SegList_x(s_x)$ を，長さ m のビットマスク表現 $B_x[s_x] \in \{0,1\}^m$ で記録し，ビットマスク配列 B_x を構築する．同様に y 軸上の区間の集合を $Seg_y(P)$ と，ビットマスク配列 B_y を構築する．このとき領域は，$Seg_x(P), Seg_x(P)$ に $O(m)$ 語を，B_x, B_y の保持に $O(\lceil \frac{m^2}{w} \rceil)$ 語をそれぞれ用いる．位置同定演算 $PointLocation$ は，入力点の x 座標と y 座標を含む区間 s_x と s_y を，それぞれ，$Seg_x(P)$ と $Seg_y(P)$ 上の二分探索で求める．得られた 2 つのビットマスク $B_x[s_x], B_y[s_y]$ のビット AND をとる事により $O(\log m + \lceil \frac{m}{w} \rceil)$ 時間で実行する．以上より示された． □

探索処理: テキスト $T = t_1 ... t_n$ の点を 1 つずつ読み，その点を含む面に対応するビットマスク B を $PointLocation$ によって求める．次に，求めたビットマスクを用い，7 行目のビット並列手法に基づく演算によって NFA の遷移を計算することで照合を行う．

4. 理論的解析

本節では，提案アルゴリズムの前処理時間と，実行時間，領域を与える．ここで $w = \Theta(\log n)$ はワード長である．また，ワード内のビット演算は定数時間で実行可能と仮定する [1,6].

まず，領域に関しては，配置 $\mathcal{A}(P)$ に対する区間の保持に $O(m)$ 語と，位置同定索引の保持に $O(\lceil \frac{m^2}{w} \rceil)$ 語を要する．よって合計で $O(\lceil \frac{m^2}{w} \rceil)$ 語かかる．前処理時間は，入力パターン P に対する位置同定索引の構築に $O(m \log m + \lceil \frac{m^2}{w} \rceil)$ 時間かかる．

Algorithm 1 Shift-AndTrajMatch

```
   Preprocessing
 1: P ← p_1 ... p_m
 2: T ← t_1 ... t_n
 3: PL ← ConstructIndex(P)
   Searching
 4: S ← 0^m                                  ▷ 状態集合マスク
 5: for pos ∈ 1...n do
 6:     B ← PL.PointLocation(t_pos)
 7:     S ← ((S << 1) | 0^{m-1}1) & B
 8:     if S & 10^{m-1} ≠ 0^m then
 9:         report an occurrence at pos − m + 1
10:     end if
11: end for
```

次に実行時間について考える．テキストを 1 文字読むごとに位置同定演算と，7 行目のビット並列演算を実行するので，合計で $O(\log m + \lceil \frac{m}{w} \rceil)$ 時間かかる．よってテキスト全体で $O(n(\log m + \lceil \frac{m}{w} \rceil))$ 時間かかる．以上の議論より，本稿の結果が示される．

定理 1 長さ m の軌跡パターン P と長さ n の軌跡テキスト T，非負実数 r に対して，アルゴリズム *Shift-AndTrajMatch* は $O(m \log m + \lceil \frac{m^2}{w} \rceil)$ 前処理時間と $O(\lceil \frac{m^2}{w} \rceil)$ 語を用いて，$O(n(\log m + \lceil \frac{m}{w} \rceil))$ 時間で，軌跡パターン照合問題を解く．ここで w はワード長である．

提案アルゴリズムは，パターン長が $m \leq w$ の時には，$O(n \log m)$ 時間で軌跡パターン照合を行う事が出来る．

5. 今後の課題

本論文では，軌跡データに対するパターン照合問題を考察した．提案手法の Shift-AndTrajMatch では，位置同定索引とビット並列手法を組み合わせる事により，効率の良い照合を実現した．今後の課題としては，ブロック分割手法による位置同定索引の領域削減及び，計算量の改善可能性の検討が挙げられる．また，提案手法を実装し，計算機実験を行う事も重要な課題である．

参考文献

[1] Ricardo Baeza-Yates and Gaston H. Gonnet. A new approach to text searching. *Commun. ACM*, 35(10):74–82, October 1992.

[2] Mark de Berg, Otfried Cheong, Marc van Kreveld, and Mark Overmars. *Computational Geometry: Algorithms and Applications*. Springer-Verlag, Santa Clara, CA, USA, 3rd ed. edition, 2008.

[3] Thomas H. Cormen, Clifford Stein, Ronald L. Rivest, and Charles E. Leiserson. *Introduction to Algorithms*. McGraw-Hill Higher Education, 2nd edition, 2001.

[4] Thomas Eiter and Heikki Mannila. Computing discrete Fréchet distance. Technical Report Technical Report CD-TR 94/64, Christian Doppler Laboratory for Expert Systems, TU Vienna, Austria, 1994.

[5] Gonzalo Navarro and Mathieu Raffinot. *Flexible Pattern Matching in Strings: Practical On-line Search Algorithms for Texts and Biological Sequences*. Cambridge University Press, New York, NY, USA, 2002.

[6] Sun Wu and Udi Manber. Fast text searching: Allowing errors. *Commun. ACM*, 35(10):83–91, October 1992.

Copyright © 2014 by
The Institute of Electronics, Information and Communication Engineers and
Information Processing Society of Japan All rights reserved.