D-019

ビット並列手法に基づく大規模連続ストリームパターン照合
Massive Continuous Stream Pattern Matching Based on Bit-Parallel Method

斉藤 智哉 * 喜田 拓也 * 有村 博紀 *
Tomoya Saito Takuya Kida Hiroki Arimura

概要：本稿では，k 変数ストリームのための効率よいパターン照合アルゴリズム K-BPS を提案する．

1. はじめに

新しい型の大規模データ応用として，連続ストリーム処理が注目されている．論文 [4] では，良く知られた文字列照合手法であるビット並列手法に基づく高速なストリームパターン照合手法 BPS を与えた．本稿では，これを k 変数ストリーム上の変数分離形のパターンへ拡張し，K-BPS (Bit-Parallel on k-Streams) を提案する．合成データを用いた計算機実験では，KMP 手法と BM 手法 [2] に基づく既存手法 L2R [1,3] と R2L [1] と比較して，K-BPS は 1.5 ～ 5.0 倍程度高速で，性能も安定していた．ビット並列手法は，将来のビット幅の増大が性能向上に結びつき，正規表現照合も可能であるなどの利点を持つなど，連続ストリーム処理に有望と思われる．

2. 準備

ストリームとパターン: $\mathbb{N}$ で非負整数全体を表わす．集合 A に対し，A^* で要素の有限列を表わす．$\Delta = [1..c] \subseteq \mathbb{N}$ を値の全体集合とし，正整数 $k \geq 1$ に対し，k 個の変数の集合を $\mathcal{X} = \{ X_i \mid i = 1, \ldots, k \}$ とおく．

Δ 上の k 変数ストリームとは，長さ $n \geq 0$ の順序列 $S = (R_1, \ldots, R_n) \in (\Delta^k)^*$ である．ここに，各時点 $1 \leq i \leq n$ に対して，$R_i = (R_i[1], \ldots, R_i[k]) \in \Delta^k$ は値の k 項組であり，k レコードと呼ぶ．$\mathcal{R} = \Delta^k$ で k レコード全体の集合を表わす．例として，表 1 の左側に，$k = 3$ の場合の k 変数ストリーム S_1 を示す．ここに，S の v 行 p 列は変数 X_v の位置 p での値を示す．

真偽値を $\{0,1\}$ で表わす．$\mathcal{X}$ 上の数値述語（述語）は，$X \, op \, Y$ $(X, Y \in \mathcal{X} \cup \Delta, op \in \{=, <, \leq\})$ の形の不等式に論理演算 $\wedge, \vee, \neg$ を有限回適用して得られる論理式 ϕ である．述語 ϕ に対して，Δ^k 上の二値関数 $\mathrm{f}_\phi : \Delta^k \to \{0,1\}$ を次のように定める：(i) 与えられた k レコード $R \in \Delta^k$ 上で，各 $1 \leq v \leq k$ に対して，$X_v = R[j]$ と定める．(ii) 述語 ϕ の真偽値 $\mathrm{f}_\phi(R) \in \{0,1\}$ を，通常のブール論理によって定める．以後，$\mathcal{P} = \mathcal{P}(\mathcal{X})$ で $\mathcal{X}$ 上の述語の全体集合を表わす．

$\mathcal{P}$ 上の**単純時系列パターン（単純パターン）**は，長さ m の述語列 $P = \phi_1 \cdots \phi_m \in \mathcal{P}^*$ である．単純パターン P が**変数分離形**であるとは，P のすべての述語が $\phi = \psi_1 \wedge \cdots \wedge \psi_k$ の形で，任意の $1 \leq i \leq k$ に対して，部分述語 ψ_i が X_i 以外の変数を含まないことをいう．図 1 に，変数分離形の単純パターンの例 P_1 を示す．

$P_1 = \phi_1 \cdot \phi_2 \cdot \phi_3 \cdot \phi_4$ **where**
$\phi_1 = (X_1 \geq 2 \wedge X_1 \leq 3)$ **and**
$\phi_2 = (X_1 \geq 2 \wedge X_3 \geq 2 \wedge X_3 \leq 3)$ **and**
$\phi_3 = (X_1 \geq 3 \wedge X_2 \geq 2)$ **and**
$\phi_4 = (X_2 \leq 3 \wedge X_3 \leq 2)$.

図 1: 変数分離形の単純時系列パターンの例

*北海道大学大学院情報科学研究科, Hokkaido University

表 1: k 変数ストリーム S_1 と P_1 のビットマスク配列 B

k 変数ストリーム S_1								
p	1	2	3	4	5	6	7	8
X_1	1	3	2	4	3	2	3	1
X_2	1	4	2	3	2	1	3	3
X_3	4	4	3	3	2	2	1	1

ビットマスク配列 B				
	1	2	3	4
X_1	0000	1101	1111	0111
X_2	1101	1101	1111	1110
X_3	1011	1111	1110	1010

パターン照合問題：$\mathcal{P}$ 上の長さ m のパターン $P = \phi_1 \cdots \phi_m$ が，長さ n のストリーム $S = (R_1, \ldots, R_n) \in (\Delta^k)^*$ 中で**位置 p に出現する**とは，任意の $1 \leq i \leq m$ に対して，述語 ϕ_i がレコード R_{p+i-1} 上で真であること，すなわち，$\mathrm{f}_{\phi_i}(R_{p+i-1}) = 1$ が成立することをいう．正整数 k に対して，**k ストリーム上の時系列パターン照合問題**とは，サイズ $m \geq 0$ の Δ^k 上のパターン P と長さ $n \geq 0$ の k ストリーム S が入力として与えられたとき，S における P のすべての出現位置を検出する問題である．例えば，表 1 の S_1 上の P_1 の出現は $\{2,3,5\}$ である．

3. 提案アルゴリズム

手法の概要：図 2 に，ビット並列手法に基づく提案アルゴリズム K-BPS の構成を示す．K-BPS では，前処理で与えられたパターン P を解析して，手続き BuildBitmask でビットマスク配列 B を構築する．次に K-BPS は，実行時には手続き ScanStream で入力ストリームを左から右に走査しながら，B を用いて P のパターン照合を行う非決定性有限オートマトン (NFA) $\mathcal{A}$ を高速に模倣する．

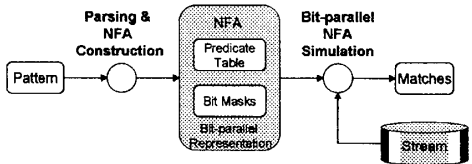

図 2: ビット並列パターン照合の概略

仮想的な NFA $\mathcal{A}$ は，与えられた長さ $m \geq 0$ の単純パターン $P = \phi_1 \cdots \phi_m$ に対して，図 3 のような直線状の NFA $\mathcal{A}$ として定められる．ここに，各遷移の有向辺のラベル $\langle i \rangle$ は，述語 ϕ_i を表わす．

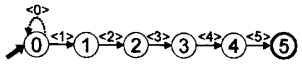

図 3: 単純パターン P_1 に対する NFA M_{P_1}

前処理：論文 [4] の手法を，Δ^k 上の k ストリームに単純に適用すると，$O(c^k)$ 語の領域のビットマスクを要する．そのため，変数分離形のパターンに対する高速な手法を与える．レコード R 上で真となる述語の集合を高速に検出するために，図 4 の手続き BuildBitmask を用いて，次の性質をもつビットマスク配列 $B[1..k][1..c] \in \{0,1\}^m$ を構築する：$(1 \leq \forall v \leq k)(\forall x \in \Delta)(1 \leq \forall i \leq m)$ $B[v][x][i] = \mathrm{f}_{\psi_v^i}(x)$．ここに，任意の $1 \leq i \leq m$ に対して，ϕ の i 番目

の述語を $\phi_i = \psi_1^i \wedge \cdots \wedge \psi_k^i$ とおく．表 1 の右側に，パターン P_1 に対するビットマスク配列の例 B を示す．B の v 行 x 列は X_v に対する値 x でのビットマスクを示す．

BuildBitmask(Δ, P)
入力：領域 $\Delta = \{1, \cdots, c\}$ とパターン $P = \phi_1 \cdots \phi_m$.
出力：ビットマスク配列 $B[1..k][1..c] \in \{0,1\}^m$.

```
1: for v ← 1,...,k do
2:   for x ← 1,...,c do
3:     B[v][x] ← 0^m;
4:     for i ← 1,...,m do
5:       if φ_i(x)=1 then
6:         B[v][x] ← B[v][x] | (1 << i − 1);
7: return B;
```

図 4: ビットマスクの構築手続き BuildBitmask

実行処理：実行時には，NFA $\mathcal{A}$ は初期状態 0 から出発し，時点 p でレコード R_p 上で真な述語 $\langle i \rangle = \phi_i$ をラベルとする遷移を非決定的に実行することを繰り返し，最終状態 m で出現位置 $p - m + 1$ を出力して，照合を模倣する．図 5 の手続き ScanStream は，状態集合 $S \subseteq \{1, \ldots, m\}$ を長さ m のビットマスク $M \in \{0,1\}^m$ で表現し，配列 B を用いたビット並列計算で高速に照合を行う．図 5 の 5 行目は一般に $O(\frac{m}{w})$ 時間で，さらに $m \leq w$ のとき $O(1)$ 時間で NFA の遷移を行う．

ScanStream(S, B)
入力：k ストリーム $S = R_1 \cdots R_n$，マスク配列 $B[1..k][1..c]$.
出力：パターン $P = \phi_1 \cdots \phi_m$ の S 上の全ての出現.

```
1: I ← 0^{m−1}1; F ← 10^{m−1}; D ← I;
2: for p ← 1,...,n do
3:   M ← 1^m;
4:   for v ← 1,...,k do M ← M & B[v][R_p[v]];
5:   D ← ((D << 1) | I) & M;
6:   if (D & F) ≠ 0^m then 出現 p − m + 1 を出力する;
7: end for
```

図 5: NFA の模倣手続き ScanStream

計算モデルとして，レジスタのビット幅 w で，数値演算と論理演算を定数時間で実行する RAM [2] を考える．

定理 1 図 5 の手続き ScanStream は，k ストリーム上のパターン照合問題を $O(\frac{kmn}{w})$ 時間と $O(\frac{kmc}{w})$ 語の領域で解く．$m \leq w$ のとき $O(kn)$ 時間と $O(kc)$ 語で解く．

4. 実験

アルゴリズムの性能を評価するために，提案アルゴリズムの三つの実装 BPS_bin, BPS_lin, BPS_tab （[4] の k 変数版）と既存アルゴリズムを用いて，合成データ上で実験を行った．BPS_tab は，前節の図 4 と図 5 のビットマスクを直接表引きで探索する実装であり，BPS_lin と BPS_bin は，線形探索と二分探索を用いた実装である．

データ．$\Delta^k = [1, c]^k$ 上の一様分布で独立に生成した n 個の k レコードからなるストリームデータ S と，ランダムパターン $P = \phi_1 \cdots \phi_m$ を用いた．ここにパラメータ $0 \leq \varepsilon \leq c$ に対して，独立なランダム整数 $a_j^i \in [1, c - \varepsilon]$ をとり，$\phi_i = \alpha_1^i \wedge \cdots \wedge \alpha_k^i$ $(1 \leq i \leq m)$ かつ $\alpha_j^i = (X_j > a_j^i) \wedge (X_j \leq a_j^i + \varepsilon)$ ととった．とくに指定しない限り，$k = 3$ とし，$c = 100, n = 10^7, m = 3$ とした．

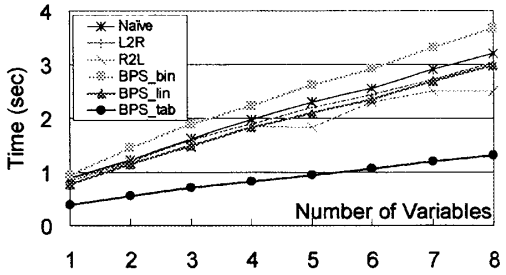

図 6: 変数の数 k に対する計算時間

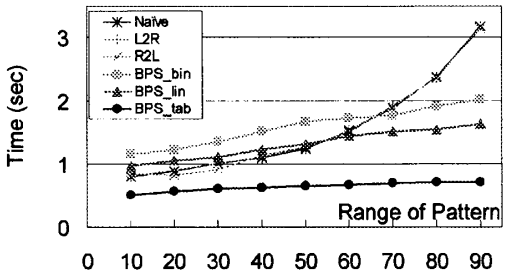

図 7: パターン幅 ε に対する計算時間

実験 1：**データサイズに対する規模耐性**．定理 1 に予想されるように，アルゴリズムの計算時間は，入力サイズに比例した．$k = 8$ 変数で，長さ 10 メガ個のストリームに対して 1.3(sec) 程度と実用的な性能を示した．BPS_tab で，領域 $\Delta = [1, c]$ のビット幅は $c = 2^{18}$ まで動作することを確認した．

実験 2：**k に対する計算時間**．図 6 に，k を 1 から 8 まで変化させたときの各アルゴリズムの計算時間を示す．次元 k によらず述語が空間 Δ^k を満たす体積を一定割合 $\gamma = 0.25$ とするため，$\varepsilon = \gamma^{1/k}$ とした．全アルゴリズムの計算時間は k に比例し，とくに，BPS_tab は他のアルゴリズムと比較して 2.5 倍程度高速だった．

実験 3：**パターン幅に対する計算時間**．パターンに対する依存性を知るために，図 7 に，$k = 3$ とし，ε を 10 から 90 まで変化させたときの各アルゴリズムの計算時間を示す．3 つの既存手法の計算時間は ε が大きくなるにつれ増加した．一方，3 つの BPS アルゴリズムは比較的安定して高速だった．

5. まとめ

本稿では，多変数数値ストリームのための時系列パターン照合を考察した．とくに，変数分離形の単純時系列パターンのクラスに対して，ビット並列手法を用いた高速なアルゴリズム K-BPS を与えた．

参考文献

[1] L. Harada, Pattern Matching over Multi-attribute Data Streams, Proc. SPIRE'02, LNCS 2476, 187–193, 2002.

[2] G. Navarro and M. Raffinot, *Flexible Pattern Matching in Strings*, Cambridge Univ. Press, 2002.

[3] R. Sadri, C. Zaniolo and A. M. Zarkesh, Jafar Adibi, Optimization of sequence queries in database systems, In *PODS'01*, ACM, 2001.

[4] T. Saito, T. Kida, H. Arimura, An efficient algorithm for complex pattern matching over continuous data streams based on bit-parallel method, In *Proc. SWOD'07*, IEEE, 2007.