# 使われるデータベース技術を目指して

藤原　靖宏

NTT ソフトウェアイノベーションセンタ　〒180-8585 東京都武蔵野市緑町 3-9-11

E-mail:　fujiwara.yasuhiro@lab.ntt.co.jp

**あらまし**　deim 2013 に参加する博士課程後期の方や博士課程後期に進学希望の方などが研究活動を行っていくうえで自分の研究に対する思いをしっかり持つことが重要である．研究に対する思いは人それぞれであるが，本講演では人に使われる技術を作ることの重要性について，今までの研究テーマを紹介していきながら伝えていく．

**キーワード**　研究紹介，博士課程，招待講演

## 1. はじめに

deim 2013 に参加する博士課程後期の方や博士課程後期に進学希望の方などが研究活動を行っていくうえで自分の研究に対する思いをしっかり持つことが重要である．Ph.D.とはDoctor of Philosophyであり，研究に対する思い（哲学）は人それぞれが持つべきものである．本講演では研究していく中で，人に使われる技術を作るという思いを持つことの重要性について，今までの研究テーマを紹介していきながら伝えていく．本論文では特に本講演で紹介する技術について説明し，これら技術を研究することになった経緯などは講演の中で明らかにしていく．

本論文の構成は以下の通りである．2章で提案手法DAPSSについて説明する．3章で提案手法SPIRALを説明する．4章で提案手法Staggered Decodingを説明する．5章で簡潔に本論文をまとめる．

## 2. DAPSS

### 2.1. 研究の背景

現在データストリームを利用したアプリケーションへの注目が金融，環境，モバイル，ウェブアプリケーション，製造等の分野で集まっており，またその分野の研究も盛んである[4][6][12]．本研究では特に複数のシーケンスから構成されるデータストリームを対象とする．一般的にデータストリームは高いビットレートで長い期間にわたって流入するため，処理するべきデータ量は膨大になる．そのためデータストリームの処理に対しては，限られたメモリ量でかつ高速に行うことが要求される[2][3][8]．

多くの既存研究では上記の要求を満たすため，データは処理後に廃棄され,また精度は犠牲にされてきた．しかしデータを廃棄することはデータストリームが処理後の分析などに用いられることがありえるため好ましくない．そしてまた精度を犠牲にすることもアプリケーションの応用を考えたときに好ましくない．そのため本研究ではデータストリーム処理の要求として先に挙げたものの他に，データストリームの処理結果が厳密に正確であるという要求を加える．

本研究では流入する複数のシーケンスのうち，探索時刻から任意の長さの問い合わせシーケンスに対して類似したシーケンスの組み合わせを探索する問題を対象とする.この問題は多くの分野で応用が可能である．

データストリームのデータ量は膨大であるため，従来この問題を解くには多くの計算量とメモリ量が必要であった．本研究では DAPSS(DAta stream Processing for Store and Search) を提案する[12]．DAPSS はこの問題を高速，正確，省メモリで処理できる．

DAPSS では処理結果の厳密性を達成するため，データシーケンスの探索処理においてメモリ内に格納したデータシーケンスの特徴量とディスクに格納したオリジナルのデータシーケンスを用いる．

本研究では DAPSS について人工データを用いて性能評価を行った．検証した結果ナイーブな手法と比較して高速に処理が行えることを確認した．

### 2.2. 問題設定

本研究では流入する $m$ 個のシーケンスのうち，探索時刻からユーザが希望する任意の長さの問い合わせシーケンスに対して類似したシーケンスの組み合わせを探索する問題を扱う[12]．この問題の解をシーケンス $X=(x_1,x_2,\cdots,x_n)$ と $Y=(y_1,y_2,\cdots,y_n)$ のユークリッド距離 $D(X,Y)$ を用いて以下のように定義する．

**問題**　問い合わせシーケンスの長さ $l$ と閾値 $\varepsilon$ が与えられたとき，類似したシーケンスを検知する問題の解は以下の条件を満たすシーケンスの組み合わせ $X_l$ と $Y_l$ のすべての集合とする．

$$D(X_l,Y_l)=\sqrt{\sum_{t=n-l+1}^{n}(x_t-y_t)^2}\le\varepsilon \quad (1)$$

この問題をナイーブに解く場合はすべてのシーケンスをメモリ上に保持しておいて，すべてのシーケンスの組み合わせについて距離計算を行う．ナイーブな手法の問題点として，多くのメモリ量が必要になることと，多くの計算量が必要になることが挙げられる．

### 2.3. 提案手法

#### 2.3.1. DAPSS で用いる手法

**可逆圧縮**　可逆圧縮は高速にシーケンスを圧縮するための手法である．可逆圧縮によりディスクにアクセスする IO コストを低減できる．

**PSA**　PSA(Piecewise Statistical Approximation) はシーケンスの平均と標準偏差を特徴量として用いてシーケンスの距離を高速に近似計算するための手法である．PSA では設定する距離関数によりシーケンス間の距離の下限値または上限値を計算できるので，ほとんどディクスにアクセスせずに類似シーケンスを求めることができる．

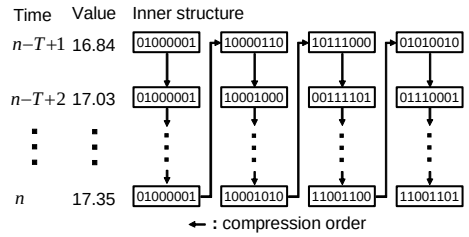



**図 1 可逆圧縮**

**マトリクス** マトリクスは多次元空間内の複数の基準点からの距離を用いて類似シーケンスを絞り込むための手法である．マトリクスにより膨大な組み合わせの中から類似データシーケンスの候補を高速に絞り込むことができる．

### 2.3.2. 可逆圧縮

**前処理** 本手法の前処理ではデータを分割して並び替える．図１に示すように計算機の内部においてはシーケンスのデータ値は複数バイトの集合として表現される.図１において は保存するシーケンスの長さである．ひとつのデータ値においては隣り合うバイトは似ていないため，データ値をそのまま圧縮してもそれほど効果は期待できない．しかしシーケンスにおいてはデータ値が漸次的に変化する特徴があるため，複数データ値の符号部と仮数部は似ている特徴がある．そのため時間ごとにシーケンスをバイト単位に分割し，並び替えたバイト列に対して圧縮を行う．

**圧縮・格納方法** 符号化にはランレングス符号化を用いる.符号化には他にハフマン符号化[9]や LZ77 符号化[11]などがあるが，ランレングス符号化はワンパスで実行できるので高速処理に向いているためである．

更新処理においてはシーケンスごとにシーケンシャルファイルの形で格納する．これは探索処理においてデータシーケンスを参照するときに高速なシーケンシャルアクセスができるようにするためである．

### 2.3.3. PSA

**特徴量** PSA ではシーケンスを特定の長さのセグメントに分割し，セグメントにおける平均と標準偏差を特徴量として距離の近似計算を行う．PSA を以下のように定義する．

**定義1（PSA）** $s_i$をシーケンス $X$（長さ $n$）を $N$ 個の特定の長さのセグメントに分割したときの $i$ 番目のセグメントとする．$l_i$ を $s_i$ の長さとし，$\phi_i$ を $s_i$ の平均とし，$\sigma_i$ を $s_i$ の標準偏差としたとき，シーケンス $X$ の PSA における特徴量を $\hat{X}=\left(\langle l_1,\phi_1,\sigma_1\rangle,\langle l_2,\phi_2,\sigma_2\rangle,\cdots,\langle l_N,\phi_N,\sigma_N\rangle\right)$ と 3 つの係数のタプルとして定義する．

ここで $n=\sum_{i=1}^{N} l_i$ である．シーケンス $X$ においてセグメント $s_i$ の開始点を $p_i(1\le i\le n)$ とすると，

$\phi_i=1/l_i\cdot\sum_{j=p_i}^{p_{i+1}-1}x_j$，$\sigma_i=\sqrt{1/l_i\cdot\sum_{j=p_i}^{p_{i+1}-1}x_j^2-\phi_i^2}$ と計算する．

**性質** PSA では設定する距離関数により距離の下限値と上限値を計算できるので，探索漏れも過剰探索も発生しない．

探索漏れが発生しないことを保証する補助定理として lower bounding lemma [1]が知られている．lower bounding lemma とは近似後のシーケンスの距離を $L(\hat{X}_l,\hat{Y}_l)$ としたときに，$L(\hat{X}_l,\hat{Y}_l)\le D(X_l,Y_l)$ が成り立てば探索漏れが発生しないという補助定理である．すなわち距離の下限値を用いることにより探索漏れなくシーケンスの類似判断ができる．

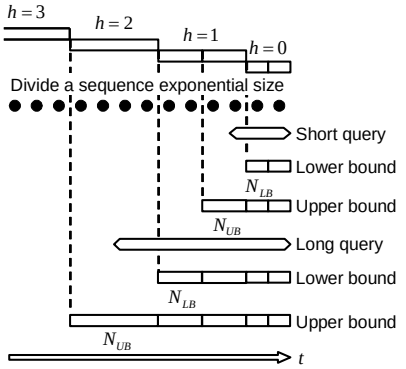



**図 2 シーケンスの分割**

また過剰探索が発生しないことを保証する補助定理として新たに upper bounding lemma を示す．

**補助定理1（upper bounding lemma）** *近似後のシーケンスの距離を $U(\hat{X}_l,\hat{Y}_l)$ としたときに，$U(\hat{X}_l,\hat{Y}_l)\ge D(X_l,Y_l)$ の条件が成り立つことが過剰探索が発生しないことの十分条件である.*

すなわち距離の上限値を用いることにより過剰探索なくシーケンスの類似判断ができる．

**下限値と上限値** PSA では図2に示す通り，下限値は問い合わせシーケンスより短いシーケンスから計算し，上限値は問い合わせシーケンスより長いシーケンスから計算する．シーケンスは複数のセグメントを集約して構成する．下限値の計算に用いるセグメントのなかで最も番号の小さいものを $N_{LB}$，上限値の計算に用いるセグメントのなかで最も番号の小さいものを $N_{UB}$ とすると，$N_{LB}=\min\left(j\,|\,\sum_{i=j}^{N}l_i\le l\right)$，$N_{UB}=\max\left(j\,|\,\sum_{i=j}^{N}l_i\ge l\right)$ として求める．

下限値を計算するときの距離関数 $L(\hat{X}_l,\hat{Y}_l)$ と上限値を計算するときの $U(\hat{X}_l,\hat{Y}_l)$ は以下のように定義する．

**定義2（距離関数 lower bound）** $L(\hat{X}_l,\hat{Y}_l)$ を以下のように定義する．

$$L\left(\hat{X}_l,\hat{Y}_l\right)=\sqrt{\sum_{i=N_{LB}}^{N}l_i\left\{(\phi_i^X-\phi_i^Y)^2+(\sigma_i^X-\sigma_i^Y)^2\right\}}\quad(2)$$

**定義3（距離関数 upper bound）** $U(\hat{X}_l,\hat{Y}_l)$ を以下のように定義する．

$$U\left(\hat{X}_l,\hat{Y}_l\right)=\sqrt{\sum_{i=N_{UB}}^{N}l_i\left\{(\phi_i^X-\phi_i^Y)^2+(\sigma_i^X+\sigma_i^Y)^2\right\}}\quad(3)$$

**セグメントの長さ** セグメントの長さを決定するために処理時刻からの経過時間によってセグメントのレベルわけを行う．図 2 に示すようにセグメント $s_i$ の長さ $l_i$ はレベル $h$ に依存し，$h$ が大きくなるに従って $l_i$ も大きくなる．レベル $h$ のセグメントの長さは $2^h$ である．このようにするのは任意の長さの問い合わせシーケンスに対応するためである．すなわち問い合わせシーケ

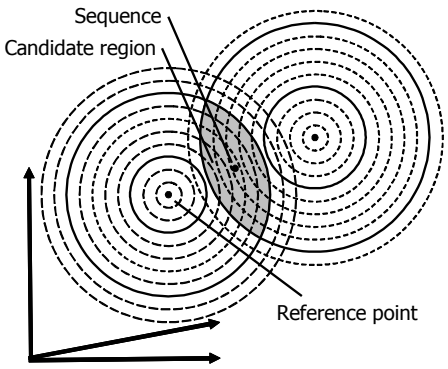


図 3 マトリクス

ンスが短い場合も長い場合も PSA で下限値と上限値を計算したときの相対的な誤差は同等になる．

特徴量の更新は各レベル $h$ のセグメントの個数が $C$ より大きくなるときに結合することでインクリメンタルに行う．$C=2$ であるときの具体的な更新方法を説明する．$c_i$ をレベル $i$ におけるセグメントの個数とする．更新前のセグメントは $c_0=2$，$c_1=2$，$c_2=1$ である．ここでシーケンスデータ $x_{n+1}$ が流入してくると $c_0=3$ となる．$C=2$ であるので，レベル 0 のはじめの 2 つのセグメントを結合してレベル 1 のセグメントを作成する．すると $c_1=3$ となるので，同様に処理する．結果として更新後のセグメントは $c_0=1$，$c_1=1$，$c_2=2$ となる．セグメント $s_i$ と $s_{i+1}$ を結合して $s_i'$ するときは，$l_i'=2l_i$，$\phi_i'=(\phi_i+\phi_{i+1})/2$，$\sigma_i'=\sqrt{(\sigma_i^2+\sigma_{i+1}^2)/2+(\phi_i-\phi_{i+1})^2/4}$ のように更新する．

### 2.3.4. マトリクス

**データ構造** マトリクスは近似後の多次元空間を同心超球構造に分割し，シーケンスに対して領域番号を付与する手法である．本手法では図 3 に示すように，PSA で近似後の多次元空間上に複数の基準点 $O_i(1\le i\le K)$ を設定する．各シーケンスには領域番号 $R(\hat{X}_{\cdot l},O_i)$ を付与する．領域情報はシーケンスと基準点の距離を刻み幅 $\varepsilon/\omega(\omega>0)$ で分割したものであり，$R(\hat{X}_{\cdot l},O_i)=\lfloor\omega\cdot L(\hat{X}_{\cdot l},O_i)/\varepsilon\rfloor$ として計算する．マトリクスのデータ構造は基準点 $O_i$ とシーケンス $\hat{X}_{\cdot l}$ の行列として表現される．

**探索処理** 探索処理ではまずマトリクスの要素の計算をする．はじめに基準点をランダムに決定する．そして各シーケンスと基準点の距離を計算してマトリクスの要素を計算する．次に探索処理ではマトリクスを用いて類似シーケンスの候補を探索する．具体的にはデータシーケンスごとにすべての基準点において $|R(\hat{X}_{\cdot l},O_i)-R(\hat{Y}_{\cdot l},O_i)|<\omega+1$ が成り立つか調べる．ひとつの基準点においてにでも成立しなければ非類似とする．本手法では探索漏れなく類似するシーケンスを求めることができる．

### 2.3.5. 処理概要

**更新処理** 更新処理ではデータを受信する度に PSA における特徴量を更新する．シーケンスデータは一定時間 $T$ ごとに可逆圧縮を用いてディスクに圧縮格納する．

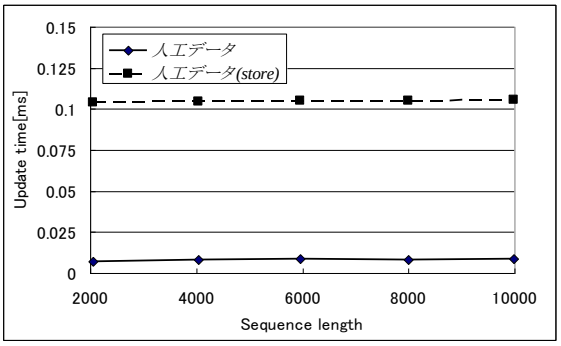


図 4 更新時間

**探索処理** 探索処理ではその都度マトリクスを構築する．次にマトリクスを用いて類似シーケンスの候補を絞り込む．絞り込まれたシーケンスの組み合わせに対して PSA の特徴量から距離の下限値と上限値を計算して類似判断をする．下限値と上限値を用いて類似判断ができない場合，すなわち距離の下限値が閾値以下でありかつ距離の上限値が閾値より大きい場合はディスクに格納されたシーケンスを解凍参照し，正確な距離を計算し類似判断を行う．

### 2.3.6 メモリ使用量

DAPSS におけるメモリ量は $O(m\cdot\log(n)+l)$ になる．これはナイーブな手法におけるメモリ使用量 $O(m\cdot n)$ と比較して省メモリであることがわかる．

## 2.4. 評価実験

実験には人工データを用いた．また実験におけるパラメータは可逆圧縮において保存するシーケンスの長さ $T=64$，PSA におけるセグメントのキャパシティ $C=25$，マトリクスにおける同心超球数 $K=10$，分割幅 $\omega=10$ とした．また閾値 $\varepsilon$ は問い合わせシーケンスの長さ $l$ によって変化させ，$\varepsilon=0.02\cdot l$ とした．実験は CPU が Pentium4 の 3.2GHz，メインメモリが 1GB のマシンで行った．

### 2.4.1. 更新時間

図 4 を見ると更新時間はシーケンスの長さ $n$ に対してほとんど変わらないことがわかる．これはシーケンスが長くなっても PSA の特徴量の更新において結合するセグメントの個数がほとんど変わらないためである．また図 4 より格納処理を行うと大幅に更新時間がかかることがわかる．これはディクスのアクセスには非常に時間がかかるためである．しかしディスクのアクセスは一定時間に一度のみしか行わないので問題にならない．

### 2.4.2 探索時間

**シーケンスの長さを変えた場合** 探索時間の実験結果を図5に示す．なお実験ではシーケンスの数 $m=500$ とした．

結果を見ると DAPSS はナイーブな手法と比較して 15～55 倍と高速な探索が行えることがわかる．ナイーブな手法では探索時間はシーケンスの長さ $n$ が増加すると $O(n)$ で増加する．しかし DAPSS ではシーケンスの長さが増加しても $O\log(n)$ 個の特徴量を用いて計

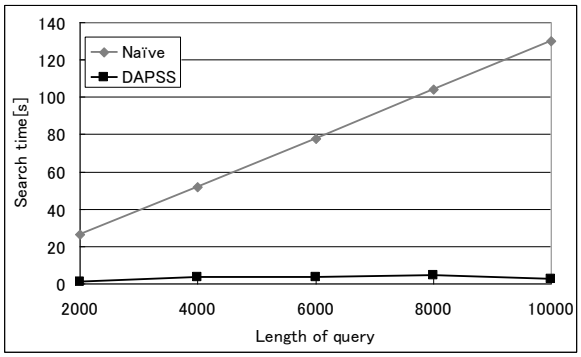



図 5 シーケンス長を変化させたときの探索時間

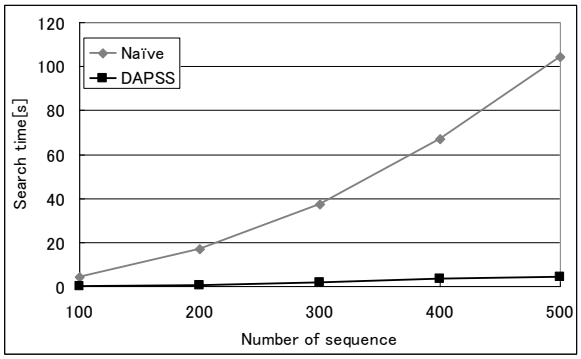



図 6 シーケンス数を変化させたときの探索時間

算を行うため，シーケンスの長さの増加の影響は抑えられ，探索時間の大幅な低減を達成している．

**シーケンスの数を変えた場合** 実験結果を図 6 に示す．なお実験では問い合わせシーケンスの長さ $l=8000$ とした．

結果を見ると DAPSS はナイーブな手法と比較して 20～45 倍と高速な探索が行えることがわかる．ナイーブな手法では探索時間はシーケンスの数 $m$ が増加すると $O(m^2)$ で増加する．しかし DAPSS ではマトリクスによって類似シーケンスの候補を絞り込んでいるためシーケンスの数の増加の影響は抑えられ，探索時間の大幅な低減を達成している．

## 3. SPIRAL

### 3.1. 研究概要

本研究では以下の問題に取り組んだ[15].

Given: 隠れマルコフモデルの集合，
データストリームの長さ $n$ のサブシーケンス $X=(x_1,\cdots,x_n)$

Find: $X$ に対して最も尤度の高いモデル

なおここで $x_n$ は最も新しいシーケンスの値とする．

本問題の応用例としては交通量の監視[13]やコンピュータの不正検出が挙げられる[14].

提案手法には以下のような特長がある

・高速：大規模なデータに対して高速に探索

・正確：最も尤度の高いモデルを正確に探索

### 3.2. 前準備

HMM は以下の要素で構成される．表 1 に主な記号とその定義を示す．

初期状態確率：$\alpha=\{\alpha_i\}$，時刻 1 において状態が $u_i$ である確率 $(i=1,\cdots,m)$.

| | |
|---|---|
| $x_t$ | シーケンス $X$ における時刻 $t$ の値 |
| $u_i$ | 隠れマルコフモデルにおける $i$ 番目の値 |
| $n$ | シーケンス $X$ の長さ |
| $m$ | 状態の個数 |
| $\alpha=\{\alpha_i\}$ | 状態 $u_i$ の初期状態確率 |
| $\beta=\{\beta_{ij}\}$ | 状態 $u_i$ から $u_j$ への状態遷移確率 |
| $\gamma=\{\gamma_i\}$ | 状態 $u_i$ におけるシンボル $v_j$ のシンボル出力確率 |
| $\Phi$ | 正確な尤度 |
| $\Psi$ | 近似尤度 |

表 1. 主な記号とその定義

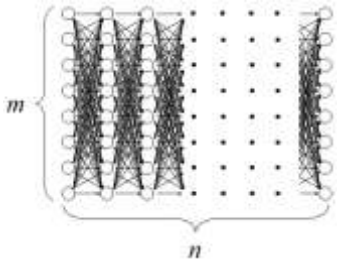



図 7. トレリス構造

状態遷移確率：$\beta=\{\beta_{ij}\}$，時刻が 1 つ進んだときに状態 $u_i$ から状態 $u_j$ への遷移する確率．

シンボル出力確率：$\gamma=\{\gamma_i(v_j)\}$，状態 $u_i$ においてシンボル $v_j$ を出力する確率 $(j=1,\cdots,s)$.

シーケンス $X$ の尤度 $\Phi$ は以下のように Viterbi アルゴリズムにより計算される．

$$\Phi=\max_{1\le i\le m}(\phi_{in})\quad\begin{cases}\phi_{it}=\max(\phi_{j(t-1)}\cdot\beta_{ji})\cdot\gamma_i(x_t),(2\le t\le n)\\ \phi_{i1}=\alpha_i\cdot\gamma_i(x_1),(t=1)\end{cases}$$

Viterbi アルゴリズムでは確率を図 7 のように状態を縦軸に，時間を横軸に並べたときに構成されるトレリス構造から計算する．Viterbi アルゴリズムの計算量 $O(nm^2)$ となる．

### 3.3. 提案手法

提案手法は図 8 の様に以下の 3 つのアイデアによって構成される．なお紙幅の関係からすべての証明は省略したが，手法の詳細等は文献[15]に述べられている．

#### 3.3.1. トレリス構造の縮退

トレリス構造が大きい場合 Viterbi アルゴリズムは高い計算コストを必要とする．そのため状態をクラスタリングし，結合することによって状態の数を削減する．この状態数の削減により，トレリス構造は縮退され，高速に近似尤度を計算することが可能になる．トレリス構造を縮退させるために粒度 $g$ が与えられたとき，$m$ 個の状態は $m/g$ 個に削減される．この結果，尤度の計算量は $O(nm^2/g^2)$ に低減化される．

状態 $u_i$ を以下の特徴量 $F_i$ を用いて k-means 法によりクラスタリングする．

$$F_i=(\alpha_i,\beta_{i1},\cdots,\beta_{im},\beta_{1i},\cdots,\beta_{mi},\gamma_i(v_1),\cdots,\gamma_i(v_s))$$

クラスタリングしてできた新しい状態の確率は各要素のうち最大の確率を用い定義する．例えば状態 $u_i$ と $u_j$ が同じクラスタになった場合，新しい初期状態確

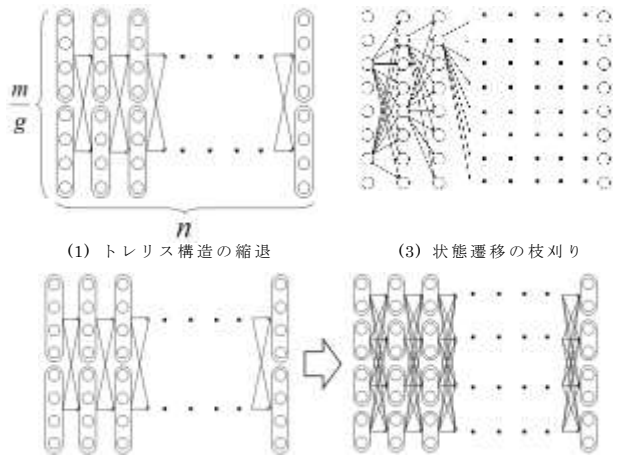



**図 8. 提案手法の概要**

率は最大の確率値から $\alpha' = \max(\alpha_i, \alpha_j)$ となり,状態遷移確率は $\beta'_{ii} = \max(\beta_{ii}, \beta_{ij}, \beta_{ji}, \beta_{jj})$ , $\beta'_{ik} = \max(\beta_{ik}, \beta_{jk})$ , $\beta'_{ki} = \max(\beta_{ki}, \beta_{kj})$ ,(ただここで $k \neq i, j$ とする)となり,同様にシンボル出力確率は $\gamma' = \max(\gamma_i(v_k), \gamma_j(v_k))$ となる.

近似尤度 $\Psi$ は結合後の状態数 $m'(=m/g)$ と確率 $\alpha'$, $\beta', \gamma'$ を用いての以下のように計算する.

$$\Psi = \max_{1 \le i \le m'}(\varphi_{in}) \quad \begin{cases} \varphi_{it} = \max(\varphi_{j(t-1)} \cdot \beta'_{ji}) \cdot \gamma'_i(x_t), (2 \le t \le n) \\ \varphi_{i1} = \alpha'_i \cdot \gamma'_i(x_1), (t=1) \end{cases}$$

近似尤度 $\Psi$ は正確な尤度 $\Phi$ の上限値となる.

**[定理 1]** *状態を結合後のモデルとシーケンスが与えられたとき, $\Psi \ge \Phi$ の関係が成立する.*

### 3.3.2. 複数の近似尤度

トレリス構造を縮退することで計算される近似尤度により高速な探索が可能になるが, 近似計算には近似精度と計算時間のトレードオフが存在する. すなわちトレリス構造を過度に縮退させると近似尤度の計算コストは小さくなるが, 近似尤度の上限値は大きくなってしまう. そのため探索処理において徐々にトレリス構造のサイズを大きくしていき, 近似尤度の精度を上げていく.

提案手法では $h+1$ 個の粒度を用いる. レベル $i(0 \le i \le h)$ の近似においては粒度 $g_i = 2^i$ を用いて, $m$ 個の状態を $\lfloor m/g_i \rfloor$ 個に結合する. 最も粗い粒度は $g_h$ であり, $g_0$ はオリジナルのモデルである. レベルが上がるにつれて $g_i$ は指数的に小さくなり, 近似尤度の精度は上がっていく.

### 3.3.3. 状態遷移の枝刈り

トレリス構造を縮退することにより高速な探索が可能になるが,正確な探索結果は近似尤度からは求めることができない.そのため近似尤度で求めた候補に対して厳密な尤度を求める.しかし厳密な尤度を計算するには大きなコストが必要なため, 提案手法ではトレリス構造において不要な計算を省いて尤度計算を高速に打ち切る.

状態 $u_i$ における時刻 $t$ の尤度計算において遷移を枝刈りするための推定値 $e_{it}$ を以下のように導入する.

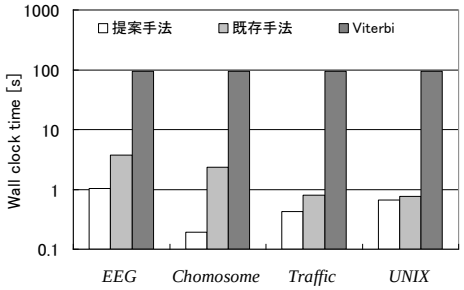



**図 9. データストリームのモニタリング処理時間**

$$e_{it} = \begin{cases} \phi_{it} \cdot (\beta_{\max})^{n-t} \cdot \prod_{j=t+1}^{n} \gamma_{\max}(x_j), (1 \le t \le n-1) \\ \phi_{in}, (t=n) \end{cases}$$

ここで $\beta_{\max}$ と $\gamma_{\max}$ は以下のように状態遷移確率とシンボル出力確率の最大値である.

$$\beta_{\max} = \max_{ij}(\beta_{ij}), \quad \gamma_{\max} = \max_{i}(\gamma_i(v))$$

推定値 $e_{it}$ はトレリス構造においてパスが時刻 $t$ で状態 $u_i$ を通るとしたときの尤度の上限値となる.

**[定理 2]** *時刻 $t$ で状態 $u_i$ を通過するパスに対しては, 時刻 $n$ で状態 $u_j$ の尤度に対して $e_{it} \ge \phi_{jn}$ の関係が成り立つ.*

### 3.3.4. 探索アルゴリズム

データストリームは時々刻々と変化していくが, 一つの値が新たに加わってもその変化の割合はそれほど多くはないことに着目した探索を行う.

まずに前時刻の解を候補として正確な尤度を計算する. そして前時刻に計算した粒度より粒度を 1 段階荒くし近似尤度を計算する. もしモデルが枝刈りできなければ 1 段階粒度を細かくして近似尤度を計算する. この処理を繰り返して最も正確な尤度が高いモデルを解として出力する.

## 3.4. 理論的解析

提案手法によって得られる解について定理 1 及び定理 2 から以下のことが成り立つ.

**[定理 3]** *提案手法はモデルを探索するときに解の厳密性を保証する.*

また Viterbi アルゴリズム においては以下の定理が成り立つ.

**[定理 4]** *Viterbi アルゴリズムは $O(m^2 + ms)$ のメモリ量と $O(nm^2)$ の計算時間を要する.*

また提案手法は複数の近似尤度を用いるが, それらの粒度は指数的に減少するため以下の定理が成り立つ.

**[定理 5]** *提案手法は $O(m^2 + ms)$ のメモリ量と少なくとも $O(n)$ の, 多くとも $O(nm^2)$ の計算時間を要する.*

## 3.5. 評価実験

実験は CPU が Intel Quad Core の 3.33GHz , メインメモリが 32GB のマシンで行った. 実験では以下の標準的な 4 つのデータセットを用いた.

・EEG : このデータセットは EEG とアルコール依存症の関係を調べるために行われた大規模な実験で得られたものであり, UCI のウェブサイト[16]からダウンロードすることができる.

・Chromosome : 人間の第 2,18,21,22 番目の染色体の

データであり，NCBI のウェブサイト[17]から得ることができる．

・Traffic：このデータセットは UCI のウェブサイト[4]から得たフリーウェイ交通量の測定値である．

・UNIX：このデータセットは UNIX の操作履歴であり，UCI のウェブサイト[16]から得ることができる．

本実験では提案手法と既存研究[18]において述べられている手法，および Viterbi アルゴリズムと比較を行った．それぞれのアルゴリズムの実験結果を「提案手法」,「既存手法」,「Viterbi」として図 9 に示す．実験において状態数を 100 とし，モデルの数を 10,000 とした．

本研究における提案手法はその他の手法に対して優位であり，特に Viterbi アルゴリズムに対しては４９０倍高速であることが確認された．

## 4. Staggered Decoding

### 4.1. 研究の背景

品詞タグ付与など，自然言語処理における基礎解析の多くは系列ラベリング問題[19]として定式化することが可能である．従来，系列ラベリングにおける復号化には，ビタビアルゴリズムが適用されてきた．しかし，ビタビアルゴリズムはラベル数の 2 乗に比例した計算時間を要するため，ラベル数が大きなタスクに適用された場合，極めて非効率となる．この問題に対処するため，ビタビアルゴリズムよりも高速で，なおかつ厳密解を保証できる復号化アルゴリズム Staggered Decoding を提案する．

### 4.2. 提案アルゴリズム

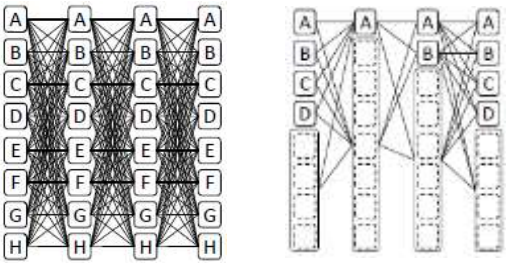
系列ラベリングとは，入力トークンの系列 $\mathrm{x} = \{x_1, x_2, \cdots, x_N\}$が与えられたとき，それに最適なラベル列$\mathrm{y} = \{y_1, y_2, \cdots, y_N\}$を予測する問題である．系列ラベリング問題は，モデル学習と復号化(最適なラベル系列の探索)に分けて考えることが出来るが，本研究では後者を議論の対象とする．

以下本節では，系列ラベリングのモデルとして隠れマルコフモデル(HMM)を考える．ただし提案手法自体は HMM に依存するものではなく，他のモデルに対しても同様に適用することが可能である．

まず，提案アルゴリズムの核となる縮退ラティスという概念を導入する．系列ラベリングにおける復号化は，ラティスの最適経路を探索する問題と捉えることが出来る(図 10-1)．ここで，ラティスの同一列上の複数ノードを 1 つにまとめ上げることによって，元のラティスをより簡潔な構造に変換することが出来る(図 10-2)．これを縮退ラティスと呼ぶ．

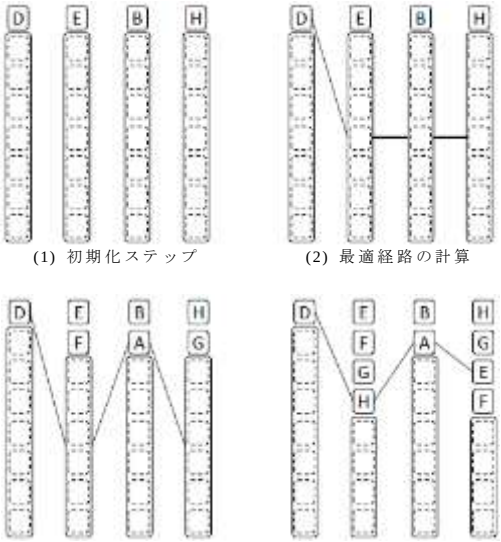
縮退ラティス構築の際に，まとめ上げられずに残ったラベル(=ノード)を活性ラベル，まとめ上げられたラベルを非活性ラベルと呼ぶ．そして，非活性ラベルをまとめ上げて新しく生成されたラベルを縮退ラベルと呼ぶ．縮退ラベルの出力確率と遷移確率は以下に示すように設定する．

$$\mathrm{p(x|z)} = \max_{\mathrm{y'} \in \mathrm{I(z)}} \mathrm{p(x|y')}$$
$$\mathrm{p(z|y)} = \max_{\mathrm{y'} \in \mathrm{I(z)}} \mathrm{p(y'|y)},$$
$$\mathrm{p(y|z)} = \max_{\mathrm{y'} \in \mathrm{I(z)}} \mathrm{p(y|y')}$$
$$\mathrm{p(z|z')} = \max_{\mathrm{y} \in \mathrm{I(z)}, \mathrm{y'} \in \mathrm{I(z')}} \mathrm{p(y|y')}$$



(1) ラティスの例　(2) 縮退ラティスの例

**図 10. 縮退ラティス**



(1) 初期化ステップ　(2) 最適経路の計算

(3) 新たに発見された最適経路　(4) 活性ラベルのみの最適経路

**図 10. Staggered Decoding**

ただしxは入力トークン，yは通常のラベル，zとz′は縮退ラベルを表す．またI(z)は縮退ラベルzに対応する非活性ラベルの集合を表す．

このように確率を設定することにより，縮退ラベルを通る経路のスコアは，それに対応する非活性ラベルを通る経路のスコアに対する上限となる．したがって，縮退ラティスの最適経路が縮退ラベルを全く含まなければ，それは元のラティスの最適経路と一致する．

提案アルゴリズム Staggered Decoding は以下の 3 ステップから構成される．

**初期化ステップ** 入力$\mathrm{x} = \{x_1, x_2, \cdots, x_N\}$に対して，全ての列に活性ラベルが 1 つだけ存在するような縮退ラティスを作成する(図 11-1)．n行目の活性ラベルには$\mathrm{p(y|}x_n\mathrm{)}$を最大化するラベルを選択する．p(y|x)の値は訓練事例を用いて推定する．

**探索ステップ** ビタビアルゴリズムを用いて最適経路を探索する(図 11-2)．もし最適経路が縮退ラベルを含まなければ，それは元のラティスの最適経路に等しいので処理を終了する．そうでなければ，次のステップへ進む．なお，探索の際には枝刈りを行うことが可能であるが誌面の都合上省略する．

**拡張ステップ** 探索ステップで得た最適経路のn番目のラベルが縮退ラベルであれば，その列の活性ラベルの数を倍にする．新しい活性ラベルは$\mathrm{p(y|}x_n\mathrm{)}$の大きな順に選択する．そして再び探索ステップを実行する(図 11-3)．この処理は，探索ステップで縮退ラベルを含まない最適経路が発見されるまで続ける(図 11-4)．

表 2. HMM での復号化速度(文数/秒)

| | 品詞タグ付与 | 結合タグ付与 | スーパータギング |
|---|---|---|---|
| ビタビ | 4600 | 100 | 1.3 |
| 提案法 | **39,000** | **4200** | **570** |

表 3. パーセプトロンでの復号化速度(文数/秒)

| | 品詞タグ付与 | 結合タグ付与 | スーパータギング |
|---|---|---|---|
| ビタビ | 4000 | 100 | 1.3 |
| 提案法 | **23,000** | **5100** | **450** |

### 4.3. 実験結果

提案アルゴリズムの有効性を，品詞タグ付け，品詞タグ付けと基本句同定の結合タスク(結合タグ付与と呼ぶ), スーパータギングの 3 タスクにおいて検証した. 結合タグ付与においては，品詞タグと基本句タグ(BIO 形式)を結合したものをラベルとした. データはそれぞれ，Penn Treebank(PTB)コーパス，CoNLL2000 コーパス，PTB コーパスを HPSG 形式に変換したものを用いた[20].

表 2,3 に, HMM とパーセプトロンにおける提案手法の復号化速度(文数/秒)，及びビタビアルゴリズムとの結果を示す．いずれの問題設定においても復号化速度が大きく向上しているが．特にラベル数の多いタスクほど速度向上が顕著であった.

## 5. まとめ

研究活動を行っていくうえで自分の研究に対する思いをしっかり持つことが重要であるが，本講演では人に使われる技術を作ること重要性について述べる．講演では今までの研究テーマ紹介を通してその重要性を述べるが，本論文では特に本講演で紹介する技術について説明した.

## 参 考 文 献


[1] R.Agrawal, C. Faloutsos and A. N. Swami.: “Efficient Similarity Search In Sequence Databases”, In FODO, 1993.

[2] B. Babcock, S. Babu, M. Datar, R. Motwani, J. Widom.: ” Models and Issues in Data Stream Systems”, In PODS, 2002.

[3] S. Babu, J. Widom.: “Continuous Queries over Data Streams”, SIGMOD Record, 2001

[4] H. Balakrishnan, M. Balazinska, D. Carney, U. Cetintemel, M. Cherniack, C. Convey, E. F. Galvez, J, Salz, M. Stonebraker, N. Tatbul, R. Tibbetts, S. B. Zdonik.: ”Retrospective on Aurora”, VLDB J., 2004.

[5] A. Bulut , A. K. Singh.: “A Unified Framework for Monitoring Data Streams in Real Time”, In ICDE, 2005

[6] J. Chen, D. J. DeWitt, F. Tian, Y. Wang.: “NiagaraCQ: A Scalable Continuous Query System for Internet Databases”. In SIGMOD, 2000.

[7] C. Faloutsos, M. Ranganathan and Y. Manolopoulos.: “Fast Subsequence Matching in Time-Series Databases”, In SIGMOD, 1994.

[8] L. Golab, M. Tamer Ozsu.: “Issues in data stream management”. SIGMOD Record, 2003.

[9] D.A.Huffman.: “A method for the construction of minimum redundancy codes”. In IRE, 1952.

[10] Y. Zhu, D. Shasha.: “StatStream: Statistical Monitoring of Thousands of Data Streams in Real Time”. In VLDB, 2002.

[11] J. Ziv, A. Lempel.: “A Universal Algorithm for Sequential Data Compression”. IEEE Transactions on Information Theory, 1977.

[12] Y. Fujiwara, Y. Sakurai, and M. Yamamuro.: “DAPSS: Exact Subsequence Matching for Data Streams”. DASFAA 2006.

[13] P. Bickel et al., “Traffic flow on a freeway network”, In Workshop on Nonlinear Estimation and Classification.

[14] T. Lane, “Hidden Markov Models for human/ computer interface modeling, IJCAI-99 Workshop.

[15] Y. Fujiwara, Y. Sakurai, M. Yamamuro, “Fast likelihood search for Hidden Markov Models”, ACM Transactions on Knowledge Discovery from Data (TKDD), 2009.

[16] http://archive.ics.uci.edu/

[17] http://www.ncbi.nlm.nih.gov

[18] Y. Fujiwara, Y. Sakurai, and M. Yamamuro, “SPIRAL: Efficient and exact model identification for Hidden Markov Models”, KDD 2008.

[19] John Lafferty, Andrew McCallum, and Fernand Pereira, “Conditional random fields: Probabilistic models for segmenting and labeling sequence data”, ICML2001.

[20] Takuya Matsuzaki, Yusuke Miyao, and Jun'ichi Tsujii, “Efficient HPSG parsing with supertagging and CFG-filtering”, IJCAI2007.