Modified PrefixSpan 法を用いた頻出正規パターンの抽出をめざして

Mining of Sequential Patterns with Variable Wildcard Regions using *Modified PrefixSpan* Method

塔野 薫隆† 北上 始‡
田村 慶一‡ 森 康真‡
黒木 進‡

Shigetaka TONO Hajime KITAKAMI
Keiichi TAMURA Yasuma MORI
Susumu KUROKI

著者らは，ある配列データベースから可変長ワイルドカード領域が含まれている頻出パターンを抽出するために，新 *Modified PrefixSpan* 法を提案する．この新しい方式は，既に提案した *Modified PrefixSpan* 法に，最大誤差数と呼ばれる入力パラメータを追加することによって開発されている．新方式の有効性を調べるために，PROSITE から Zinc Finger データセットを取り出し，可変長ワイルドカード領域が含まれる頻出パターンを抽出する能力の評価を行った．その結果，本提案方式が従来方式に比べて 9 倍の優れた抽出能力を持っている事を確認した．

In order to extract frequent patterns with a "variable wildcard region" from a sequence database, we propose the new *Modified PrefixSpan* method. The new method is enabled to develop by adding a new input parameter, called a maximum error count, to the former *Modified PrefixSpan* method. We verify this new method with a Zinc Finger dataset that is included in PROSITE. The results show that this new method has 9 times superior capacity for extraction of frequent patterns.

1. はじめに

配列データベースから頻出パターンを抽出する方法を用いると，多くの問題に答えることができる．この方法は，DNA やアミノ酸の配列データだけではなく，顧客の購買履歴，Web アクセス履歴，科学的な実験データ，病気の治療履歴，自然災害の履歴などを分析するのに有用であるといわれている．我々は，これまでに分子生物学の分野におけるモチーフ抽出の問題に着目してきた．現在，さまざまなモチーフが見つかっているが，各モチーフはあるアミノ酸配列データベース中に存在する特別な配列パターンであり，多様な蛋白質が持つ機能の 1 つに関係し，生物の進化の過程で保存されてきたものであると考えられている．

本論文では，モチーフ発見を支援するために，アミノ酸の配列データベースからさまざまなワイルドカード領域をもつ頻出パターンの抽出方法を提案する．ワイルドカード領域には，固定長と可変長のワイルドカード領域の 2 種類があるが，著者らは *PrefixSpan* 法[1]を拡張し，両者を含む頻出パターンを抽出する．拡張された *PrefixSpan* 法の評価を行うために，Zinc Finger モチーフを含むデータセットを PROSITE[2]から取り出し，そのデータセットを配列データベースとして利用し，固定長ワイルドカード領域を抽出する処理と可変長ワイルドカード領域を抽出する処理の性能比較を行う．

2. 関連研究

PrefixSpan 法は，配列データベースから頻出パターンを抽出するアルゴリズムとして知られているが，抽出過程では配列の軸位置から最左端位置までの広い範囲を参照しなければならないため，多大な計算時間を要し，無意味な頻出パターンが数多く抽出されてしまうという問題がある．また，ワイルドカード x が明記された頻出パターンを抽出する機能を持たないため，正規表現で表されるモチーフを表現できないという問題がある．例えば，頻出パターンの抽出過程において，〈AC〉,〈A*x*C〉と〈A*xx*C〉は，単に〈AC〉と解釈されてしまうので，モチーフ抽出問題には不向きである．

これらの問題を解決するために，*PrefixSpan* 法を改良した *Modified PrefixSpan* 法[3]では，頻出パターン中に複数存在する固定長ワイルドカード領域の最大長の設定を可能にし，頻出パターンの抽出過程において，〈AC〉,〈A*x*C〉と〈A*xx*C〉を異なるパターンとして区別する機能を持たせた．

しかしながら，PROSITE に登録されているモチーフには，固定長ワイルドカード領域の他に可変長ワイルドカード領域をもつモチーフが数多く存在するにもかかわらず，可変長ワイルドカード領域を自動的に抽出する研究が行われていなかった．本論文では，*Modified PrefixSpan* 法を拡張し，可変長ワイルドカード領域を含む頻出パターンを抽出する方法について提案している．

3. 問題の形式

配列データベース $S=\{S_1, S_2, S_3, \cdots, S_n\}$ の各配列 S_i は，あるアルファベット Σ 上で定義されるとする($1 \leqq i \leqq n$)．例えば，蛋白質の場合は，以下の 20 文字のアルファベット Σ_{protein} 上で定義される配列である．

Σ_{protein}={A,C,D,E,F,G,H,I,K,L,M,N,P,Q,R,S,T,V,W,Y}

Π を Σ 上で定義された曖昧文字の集合，X をワイルドカード文字列の集合，{*}を任意長のワイルドカードを表す文字列を要素とする集合とする．PROSITE のパターンは，($\Sigma \cup \Pi \cup X \cup$ {*}) 上の正規表現として書かれている[4][5]．我々は抽出するパターン表記として，PROSITE のパターン表記を用いる．

ここでは，k 個のアルファベット文字要素をもつパターンを k-パターンと呼ぶ．k-パターンが利用者によって与えられた最小支持数 *min_sup* 以上の配列にマッチするとき，そのパターンを k-頻出パターンと呼び，pat^k で表現する．配列データベース S に含まれる k-パターン pat^k は，以下の正規表現とする．

$pat^k = \langle A_1 - x(i_1, j_1) - A_2 - x(i_2, j_2) - \cdots - x(i_{k-1}, j_{k-1}) - A_k \rangle$

† 学生会員 広島市立大学大学院情報科学研究科
tono@db.its.hiroshima-cu.ac.jp
‡ 正会員 広島市立大学情報科学部
{kitakami,ktamura,mori,kuroki}@its.hiroshima-cu.ac.jp

ここで，<>内のハイフン(-)は，隣同士がお互いに連続していることを表現するための記号であり，その記号を省略することがある．A_iをアルファベットΣ上の文字要素または単に要素と呼ぶ．また，文字要素A_iが，1文字で表現されるとき単一要素，2文字以上で表現(たとえば[ILVF]など)されるとき曖昧要素と呼ぶ．$x(i,j)$の領域において，$0 \leqq i \leqq j$のとき，その領域を可変長ワイルドカード領域と呼ぶ．$i=j$のとき，それを固定長ワイルドカード領域と呼び，この領域を$x(i)$で簡略表現することがある．以下では，固定長ワイルドカード領域だけを含むパターンを基本パターンと呼ぶ．また，$x(0,\infty)$は，任意長のワイルドカード文字列(*)と同じ意味である．

本論文では，配列データベースSから可変長ワイルドカード領域が含まれる頻出パターンの集合Pを抽出する方法に着目する．これによって抽出された各頻出パターンの形式は複数の基本パターンを1つにまとめられた形式を持つ．例えば抽出された頻出パターンの形式が<F-x(2,5)-A>であれば，これは，<F-x(2)-A>,<F-x(3)-A>,<F-x(4)-A>,<F-x(5)-A>の4つの基本パターンが含まれている事を表す．頻出パターンPを抽出するためには，最小支持数min_sup，最大誤差数ε_{max}，最大ワイルドカード数max_wcを必要とする．支持数cnt_rを持つk-頻出パターンpat_r^kを$\langle pat_r^k \rangle : cnt_r$で表現し，$m$個の$k$-頻出パターンが配列データベース$S$から抽出されたとすると，それらの集まりとする集合$P_k$は以下のように表現される．

$P_k = \{\langle pat_1^k \rangle : cnt_1, \langle pat_2^k \rangle : cnt_2, \cdots, \langle pat_m^k \rangle : cnt_m\}$

Sから抽出される頻出パターンの最大要素数をqとすると，Pは$\{P_1, P_2, \cdots, P_q\}$で表現される．$(k+1)$-パターンの最右端文字は，$k$-頻出パターンに対して，$\Sigma$上の1文字を追加することにより構成され得る．その1文字は，S_iにおいて，k-頻出パターンpat_r^kの最右端文字よりも右側の位置に存在する可能性がある．その右側の位置を表現するために，以下の$PDB(pat_r^k)$を導入する．

$PDB(pat_r^k) = \{(i,j) \mid S_i$において，$pat_r^k$の最右端文字の右隣位置を$j$とする$\}$．ただし，$1 \leqq j \leqq \| S_i \|$である．以後，$PDB(pat_r^k)$を$pat_r^k$の射影データベース(Projected Database)と呼ぶ．

4. Modified PrefixSpan 法

*PrefixSpan*法を拡張した*Modified PrefixSpan*法は，固定長ワイルドカード領域が含まれる頻出パターンを抽出する方法である．

表1に示される配列データベースSから固定長ワイルドカード領域が含まれる頻出パターンの抽出方法を示そう．ただし，最小支持数min_supおよび最大ワイルドカード数max_wcを各々3とする．また，表1は，$S=\{S_1, S_2, S_3, S_4, S_5\}$であり，$S_1$=FKYAKWL, S_2=SFVKTA, S_3=ALR, S_4=MSKPL, S_5=FSKFLMAW であることを示している．配列データベースSから頻出パターンを抽出する処理は，k-頻出パターンから$(k+1)$-頻出パターンを作成することによって行われる($k \geqq 1$)．この処理をk=1とk=2の場合について考えてみよう．

k=1のとき，配列データベース中に存在するパターンは，<A>:4,<F>:3,<K>:4,<L>:4,<M>:2,<P>:1,<R>:1,<S>:3,<W>:2,<V>:1,<Y>:1である．これらにより，最小支持数を満たす1-頻出パターンを選び出すと，その集合は図1に表されているように，P_1={<A>:4,<F>:3,<K>:4,<L>:4,<S>:3}になる．例えば，1-頻出パターン<A>:4,<F>:3,<K>:4の射影データベースは，各々，PDB(<A>)={(1,5),(2,*end*),(3,2),(5,8)}，PDB(<F>)={(1,2),(2,3),(5,2),(5,5)}，PDB(<K>)={(1,3),(1,6),(2,5),(4,4),(5,4)}で表現される．PDB(<A>)内の(1,5)は，パターン<A>が1番目の配列S_1の4文字目にあり，S_1には<A>から始まる2-パターンの最右端文字は5文字目以降に存在する可能性があることを意味する．また，PDB(<A>)内の(2,*end*)は<A>が2番目の配列S_2の最右端文字にあり，S_2には<A>から始まる2-パターンの最右端文字は存在しないことを意味する．

表1 配列データベース
Table 1 Sequence Database

ID	Sequence
1	FKYAKWL
2	SFVKTA
3	ALR
4	MSKPL
5	FSKFLMAW

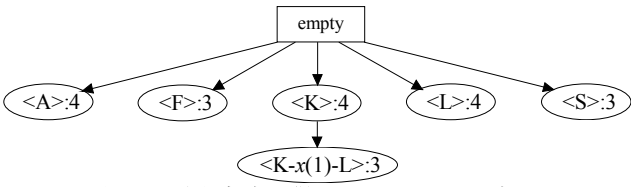

図1 従来方式で得られたマイニング木
Fig.1 Mining Tree generated by the Modified PrefixSpan Method

k=2のとき，2-頻出パターンの集合P_2はP_1中の各1-頻出パターンから構成することができる．例えば，P_1中の<A>の次にくる1文字は，<A>の射影データベースPDB(<A>)={(1,5),(2,*end*),(3,2),(5,8)}を用いて選択され得る．それらを集めた集合は$\{S_1[5], S_3[2], S_5[8]\}$={<K>,<L>,<W>}になるが，その中で$P_1$に含まれていない文字は頻出ではない．従って，2-頻出パターンの最右端文字は<K>,<L>の2つの文字に絞られるが，残念なことに<A-K>:1,<A-L>:1である．また，1～3文字のワイルドカードを含む文字列<A-x(1)>,<A-x(2)>,<A-x(3)>のどれを考えても，その次にくる1文字をさがし2-頻出パターンを構成することができない．以上から，<A>から始まる2-頻出パターンは存在しない(図1)．

次に，P_1中の<K>から始まる2-頻出パターン(ワイルドカード数がゼロの場合)について考えてみよう．<A>の場合と同様に，<K>の次にくる1文字は<K>の射影データベースPDB(<K>)を用いて選択される．それらの中でP_1に含まれる文字は$\{S_2[5], S_5[4]\}$={<T>,<F>}である．従って，<K-T>:1,<K-F>:1となるので，ワイルドカード数をゼロとする限り，2-頻出パターンは存在しない．次に，1文字のワイルドカードを含む2-頻出パターンについて考えてみよう．<K-x(1)>の次にくる1文字はPDB(<K-x(1)>)={(1,3+1),(1,6+1),(2,5+1),(4,4+1),(5,4+1)}を計算することにより選択される．それらの中でP_1に含まれる文字は，$\{S_1[4], S_1[7], S_2[6], S_4[5], S_5[5]\}$={<A>,<L>,<A>,<L>,<L>}である．これにより，<K-x(1)-A>:2,<K-x(1)-L>:3を構成することができる．しかし，2文字および3文字のワイルドカードを含む文字列<K-x(2)>,<K-x(3)>のどちらを考えても，もはや2-頻出パターンは存在しない．以上により，<K>から始まる2-頻出パターンは，<K-x(1)-L>:3だけとなる(図1)．3-頻出パターンを構成するために必要な<K-x(1)-L>の射影データベースPDB(<K-x(1)-L>)は{(1,*end*),(4,*end*),(5,6)}となるので，この2-頻出パターンから3-頻出パターンを構成することができない．

5. 新 Modified PrefixSpan 法

新 *Modified PrefixSpan* 法の処理では，前章で説明した入力パラメータの他に，最大誤差数 ε_{max} を必要とする．最大誤差数がゼロのとき，従来の *Modified PrefixSpan* 法と同じである．また，*k*-頻出パターン⟨pat^k⟩から可変長ワイルドカード領域が含まれる(*k*+1)-パターンを抽出するときに，同じ最右端文字 α をもつ以下のパターンが数多く見つかる．

⟨pat^k-$x(i, i+\varepsilon_1)-\alpha$⟩:$cnt_{\varepsilon 1}$, ⟨$pat^k$-$x(i, i+\varepsilon_2)-\alpha$⟩:$cnt_{\varepsilon 2}$, …

$\varepsilon_1 \leqq \varepsilon_2 (\leqq \varepsilon_{max})$ のとき，明らかに，$cnt_{\varepsilon 1} \leqq cnt_{\varepsilon 2}$ であり，⟨pat^k-$x(i, i+\varepsilon_1)-\alpha$⟩:$cnt_{\varepsilon 1}$ は，⟨pat^k-$x(i, i+\varepsilon_2)-\alpha$⟩:$cnt_{\varepsilon 2}$ に含まれる表現である．従って，このような同じ最右端文字のパターンに対しては，我々は，最も一般的な形式のパターンだけを候補パターンとして選び出している．これにより，冗長性のない(*k*+1)-頻出パターンを抽出している．

以下，表 1 で示される配列データベースの例を用いて，可変長ワイルドカード領域が含まれる頻出パターンを抽出する処理について考えてみよう．ただし，最小支持数 *min_sup*，最大ワイルドカード数 *max_wc*，最大誤差数 ε_{max} を各々3 とする．配列データベース *S* から頻出パターンを抽出する処理は，前章と同じであり，*k*-頻出パターンから(*k*+1)-頻出パターンを作成することによって行う(*k*≧1)．*k*=1 に対応する 1-頻出パターンの構成方法は，図 2 に示されるように，従来の *Modified PrefixSpan* 法と同じである．*k*=2 に対して，従来の方法では，⟨F⟩から始まる 2-頻出パターンは抽出できなかったが，新方式ではそれが可能になっている．

以下では，2-頻出パターン *pat*=⟨F-*x*(0,3)-K⟩に着目し，この 2-頻出パターンから 3-頻出パターンを生成する方法について考えてみよう．図 3 に示されるように，ワイルドカード数 *wc* が 0 と 1 に対応する 3-頻出パターンが生成されるが，ワイルドカード数 *wc* が 2 と 3 に対応する 3-頻出パターンは生成されない．

例えば，2-頻出パターンを *pat*=⟨F-*x*(0,3)-K⟩とし，その 2-頻出パターンからワイルドカード数 *wc* が 1 の 3-頻出パターンを見つけてみよう．このために，誤差 ε を最小値 0 から最大値の 3 まで変化させる．即ち，0≦ ε ≦3 とし，3-頻出パターン⟨*pat*-$x(1, 1+\varepsilon)-\alpha$⟩:*cnt* の最右端文字 α および支持数 *cnt* を計算する方法について考える．この 3-パターンは，4 つの基本パターン⟨*pat*-$x(1, 1)-\alpha$⟩, ⟨*pat*-$x(2, 2)-\alpha$⟩, ⟨*pat*-$x(3, 3)-\alpha$⟩, ⟨*pat*-$x(4, 4)-\alpha$⟩をまとめた表現である．各基本パターンの支持数は，誤差 ε を 0 から 3 まで順に変化させながら，誤差 ε ごとに，同じ α をもつ 3-パターンが異なるタプルに何回出現するかを正確に数え上げることで計算できる．この様子を図 3 の(a)と(b)に示す．

図 3 で，*pat*=⟨F-*x*(0,3)-K⟩から始まる 3-パターンの最右端文字 α が⟨A⟩の場合について考えてみよう．図 3 の(a)で誤差

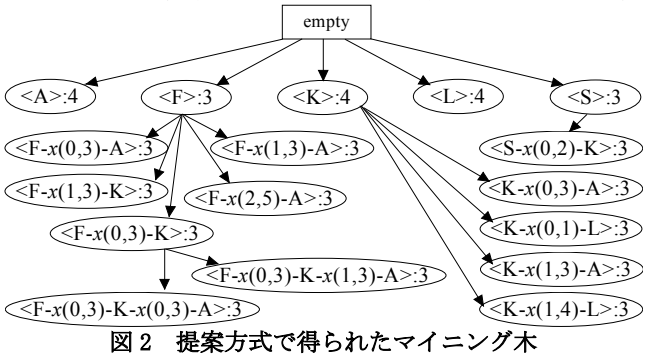

図 2 提案方式で得られたマイニング木

Fig.2 Mining Tree generated by the new method

(a)

ID \ ε	0	1	2	3
1	*pat*-*x*-A *pat*-*x*-L	*pat*-*xx*-K	*pat*-*xxx*-W	*pat*-*xxxx*-L
2	*pat*-*x*-A	–	–	–
3	–	–	–	–
4	–	–	–	–
5	*pat*-*x*-L	*pat*-*xx*-M	*pat*-*xxx*-A	*pat*-*xxxx*-W

(b)

α \ ε	0	1	2	3
⟨A⟩	⟨*pat*-*x*(1,1)-A⟩:2	–	⟨*pat*-*x*(1,3)-A⟩:3	–
⟨F⟩	–	–	–	–
⟨T⟩	–	–	–	–
⟨K⟩	–	⟨*pat*-*x*(1,2)-K⟩:1	–	–
⟨L⟩	⟨*pat*-*x*(1,1)-L⟩:2	–	–	⟨*pat*-*x*(1,4)-L⟩:2
⟨M⟩	–	⟨*pat*-*x*(1,2)-M⟩:1	–	–
⟨W⟩	–	–	⟨*pat*-*x*(1,3)-W⟩:1	⟨*pat*-*x*(1,4)-W⟩:2

図 3 ⟨*pat*-$x(1, 1+\varepsilon)-\alpha$⟩:*cnt* 形式の 3-頻出パターンの計算 (0≦ $\varepsilon \leqq \varepsilon_{max}$)

Fig.3 Extraction of 3-Frequent Patterns having ⟨*pat*-$x(1, 1+\varepsilon)-\alpha$⟩:*cnt* Expression (0≦ $\varepsilon \leqq \varepsilon_{max}$)

ε が 0 のとき，3-パターン⟨*pat*-*x*-A⟩は，配列データベース中に 2 箇所存在するが，それらは，お互いに異なるタプル(1 番目と 2 番目)に含まれているので，図 3 の(b)に示されるように⟨*pat*-*x*(1,1)-A⟩:2 となる．図 3 の(a)で誤差 ε が 1 のとき，最右端文字が⟨A⟩であるパターン⟨*pat*-*xx*-A⟩はどのタプルにも存在しないが，誤差 ε が 2 のとき，⟨*pat*-*xxx*-A⟩はそれらのタプルとは異なる 5 番目のタプルに含まれている．従って，図 3 の(b)に示されるように，ここまでの計算では，⟨*pat*-*x*(1,1)-A⟩:2 かつ⟨*pat*-*xxx*-A⟩:1 であるので，⟨*pat*-*x*(1,3)-A⟩の支持数は 3 になる．最後に，誤差 ε が 3 のときは，最右端文字が⟨A⟩で終わる 3-パターン⟨*pat*-*xxxx*-A⟩は存在しない．以上から，*pat*=⟨F-*x*(0,3)-K⟩から始まる 3-パターンは，⟨*pat*-*x*(1,1)-A⟩:2, ⟨*pat*-*x*(1,3)-A⟩:3 の 2 つ存在することになるが，後者は前者を包含するので，前者を除外する．従って，ワイルドカード数を 1 に固定したとき，*pat*=⟨F-*x*(0,3)-K⟩から始まる 3-頻出パターンは，⟨F-*x*(0,3)-K-*x*(1,3)-A⟩:3 となる．

ワイルドカード数を 1 に保持したままで，他の候補パターンについて考えると，図 3 の(b)に示されるように 6 個の 3-パターンが見つかる．そして冗長なパターンとして，⟨*pat*-*x*(1,1)-L⟩:2 および⟨*pat*-*x*(1,3)-W⟩:1 が除外され，4 個の 3-パターンが残る．しかし，4 個のいずれも最小支持数を満たさないので，他の 3-頻出パターンは存在しない．

6. 計算結果と性能評価

PROSITE から Zinc Finger モチーフを含むデータセットを選択するために，アクセッション番号として PS00028 を用いた．このデータセットは 744 件の配列が含まれている．Zinc Finger モチーフは⟨C-*x*(2,4)-C-*x*(3)-[LIVMFYWC]-*x*(8)-H-*x*(3,5)-H⟩の形式で知られている．この形式は，72 種類の基本パターンを一つにまとめたものである．しかし，評価のために使われたデータセットには，その内の 34 種類の基本パターンだけが存在する．また，その 34 種類の基本パターンの内，配列データベース内で最も支持率の高いパターンの形式は⟨C-*x*(2)-C-*x*(3)-F-*x*(8)-H-*x*(3)-H⟩である．このパターンは，配列データベース内に 3299 箇所あり，支持率は 82%である．また，最も支持率の低いパターンは⟨C-*x*(2)-C-*x*(3)-M-*x*(8)-H-*x*(4)-H⟩である．このパターンの支持率は 0.13%である．

この配列データベースから頻出パターンを抽出するために，入力データとして *max_wc* を 8 にして，従来方式では ε_{max} を 0，提案方式では ε_{max} を 2 として入力し，表 2 でその計算結果が表されている．提案方式の計算では *min_sup* は 90%から 50%まで下げていって各支持率によって得られた基本パタ

表 2 Zinc Finger の計算処理
Table 2 Zinc Finger Evaluation

比較項目＼最小支持率		90%	80%	70%	60%	50%
従来方式	正解の数(件)	0	1	1	1	1
	計算時間(sec)	2.23	3.46	6.7	15.25	49.89
	頻出パターン数(件)	81	277	1,108	3,998	19,104
提案方式	正解の数(件)	9	9	-	-	-
	計算時間(sec)	22.9625	128.616	-	-	-
	頻出パターン数(件)	1,042	9,669	-	-	-

ーンの種類数や頻出パターン数や計算時間を表している．また，提案方式では *min_sup* を 90%と 80%にして得られた基本パターンの種類数や頻出パターン数や計算時間を表している．

従来方式では，最小支持率が 80%のとき，Zinc Finger モチーフの一部分として，82%の支持率をもつ頻出パターン<C-*x*(2)-C-*x*(3)-F-*x*(8)-H-*x*(3)-H>が頻出パターンの集合に含まれていた．それは，1 件の正解に相当する．最小支持率を 50%まで下げたが，他の正解は抽出されなかった．一般に最小支持率を下げると，計算に必要なメモリー領域が増える．我々の計算機環境では，最小支持率を 50%未満にすると，計算の途中でメモリー領域が不足したため，途中で計算を打ち切った．これに対して，提案方式では，最小支持率が 90%のとき，<C-*x*(2,4)-C-*x*(3,5)-F-*x*(8,10)-H-*x*(3,5)-H>:93%ほか 17 件の頻出パターンが正解を含んでいた．それらを基本パターンとして整理すると，9 件の正解(正解総数の 26%)に相当することがわかった．最小支持率を 80%未満に設定すると，提案方式では計算の途中でメモリー領域が不足する．提案方式では最小支持率が 90%のときの計算時間は 23.0 秒であったが，従来方式でこれと同じ程度の計算時間に対応する最小支持率は 60%～50%である．このとき，従来方式では，正解が 1 件だけしか抽出されていないことがわかる．以上から，提案方式は従来の方式に比べて，正解の数が 9 倍多かったと言える．

7. まとめ

従来の *Modified PrefixSpan* 法は，最小支持数と最大ワイルドカード数の 2 つの設定パラメータに対して，固定長ワイルドカード領域を含む頻出パターンだけを抽出していたので，可変長ワイルドカード領域を含む頻出パターンの一部分になっている基本パターンを見落としてしまうという問題がある．本論文では，この問題を解決するために，*Modified PrefixSpan* 法に可変長ワイルドカード領域を含む頻出パターンを抽出する機能を持たせる方法を提案した．具体的には，従来方法に最大ワイルドカード数に対する最大誤差数のパラメータを新たに導入した．提案方法の有効性を確かめるために，PROSITE から Zinc Finger データセットを選び，計算結果を従来方法と比較した．そこでは，提案方法と従来方法の計算終了時間がお互いにほぼ同じになるような最小支持率を各々選択し，各々の方法で抽出された頻出パターン集合を調べた．各々の頻出パターン集合の中を調べた結果，提案方法が従来方法に比べて，正解としての基本パターンが 9 倍多く抽出されていることがわかった．提案方法に限らず従来方法では，一般に，最小支持数を下げれば，計算時間や計算に必要なメモリー容量が増える．今回，提案方法で抽出された基本パターンの中には，設定された最小支持数よりも小さな支持数が含まれている．極端な例になるが，Zinc Finger データセットの場合，抽出された可変長ワイルドカード領域を含む頻出パターンの中に，支持数が 1 の基本パターン(<C-*x*(3)-C-*x*(3)-F-*x*(8)-H-*x*(5)-H>)が含まれていた．これを従来方法で最小支持数を 1 と設定して計算をすると，計算の途中で組合わせ爆発が生じる．即ち，n 本の配列の平均配列長を m とすると，時間計算量は $O(n2^m)$ となってしまうので，従来方法を用いてこの基本パターンを見つけるのは現実的ではない．

今後の課題として，3 章で説明した「曖昧要素」を含む頻出パターンを抽出する方法の研究があげられる．

[謝辞]

本研究の一部は，広島市立大学特定研究費(一般研究費(コード番号：3106)の支援により行われた．

[文献]

[1] Jian Pei, Jiawei Han, Behzad Mortazavi-Asl, and Helen Pinto: PrefixSpan: Mining Sequential Patterns Efficiently by Prefix-Projected Pattern Growth, Proc. of International Conference on Data Engineering (ICDE 2001), pp.215-224, IEEE Computer Society Press, 2001.
[2] PROSITE: http://kr.expasy.org/prosite
[3] Hajime Kitakami, Tomoki Kanbara, Yasuma Mori, Susumu Kuroki, and Yukiko Yamazaki: Modified PrefixSpan Method for Motif Discovery in Sequence Databases, Proc. of the 7th Pacific Rim International Conference on Artificial Intelligence (PRICAI2002), pp.482-491, Springer-Verlag, August 2002.
[4] Inge Jonassen, John F. Collins, and Desmond G. Higgins: Finding Flexible Patterns in Unaligned Protein Sequences, Protein Science, pp.1587-1595, Cambridge University Press, 1995.
[5] D. Gusfield: Algorithms on Strings, Trees, and Sequences, Cambridge University Press, 1997.

塔野 薫隆 Shigetaka TONO
広島市立大学大学院情報科学研究科研究生．2003 ボリビア国・カトリック大学サンタクルス校システム工学部卒業．日本データベース学会学生会員．

北上 始 Hajime KITAKAMI
広島市立大学情報科学部教授．1976 東北大学大学院工学研究科博士前期課程修了．博士(工学)．データベースシステムの研究・開発に従事．情報処理学会一般情報処理教育委員会委員および CE 研究会運営委員，日本データベース学会 BI 研究グループ運営委員など．

田村 慶一 Keiichi TAMURA
広島市立大学情報科学部助手．2000 九州大学大学院システム情報科学府修士課程修了．データベース並列処理の研究に従事．日本データベース学会，情報処理学会 各会員．

森 康真 Yasuma MORI
広島市立大学情報科学部助手．1994 北陸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．データベースシステムの研究・開発に従事．日本データベース学会会員．

黒木 進 Susumu KUROKI
広島市立大学情報科学部助教授．1990 東京大学大学院工学系研究科修士課程修了．博士(工学)．空間データベースの研究に従事．日本データベース学会，情報処理学会，電子情報通信学会，ACM，IEEE 各会員．