段階的一般化法によるミスマッチクラスタからの最小汎化パターン抽出

木村　浩明† 荒木康太郎‡ 田村　慶一‡ 澤田　祐介‡ 北上　始‡

†広島市立大学情報科学部 〒731-3194 広島市安佐南区大塚東三丁目 4 番 1 号
‡広島市立大学院情報科学研究科 〒731-3194 広島市安佐南区大塚東三丁目 4 番 1 号
E-mail: {kimura, kotaro, ktamura, sawada, kitakami}@db.its.hiroshima-cu.ac.jp

あらまし 配列データベースに対して曖昧文字表現を含む頻出配列パターンを抽出する方法を提案する．この提案手法は，ミスマッチクラスタに対する冪集合に含まれる集合要素について，小さなサイズの集合要素から順に列挙し，列挙木を探索するボトムアップアプローチをとっている．この探索過程において，列挙された各集合に対する最汎パターンの計算，不要な部分列挙木の枝刈り，冗長なパターンの除去などを実施することにより，ミスマッチクラスタに対する最小汎化パターン集合を抽出する．提案手法の有効性を確認するために，4 種類のデータセットを用いて実験を行ったので，その実験結果についても報告する．
キーワード テキストマイニング，バイオインフォマティクス，情報抽出

A Step-wise Generalization Method for Extracting Sequential Patterns With Least Minimum Generalization from Mismatch Clusters

Hiroaki KIMURA, Kotaro ARAKI, Keiichi TAMURA,

Yusuke SAWADA, and Hajime KITAKAMI

Faculty of Information Sciences, Hiroshima City University 3-4-1 Ozuka-Higashi, Asa-Minami-Ku, Hiroshima, 731-3194 Japan

Graduate school of Information Sciences, Hiroshima City University 3-4-1 Ozuka-Higashi,

Asa-Minami-Ku, Hiroshima, 731-3194 Japan

E-mail: †{kimura,kotaro,ktamura,sawada,kitakami}@db.its.hiroshima-cu.ac.jp

Abstract We propose a method for extracting frequent sequence patterns including the expression of ambiguous characters from sequence databases. The proposed method is bottom-up approach that sequentially enumerate element included in power set to the mismatch cluster from the small size, and search for the enumeration tree. At the stage of the search process, extract sequential patterns with least minimum generalization from mismatch clusters by computing a most general pattern from each enumerated set, pruning some needless parts of enumeration tree, and removing redundant patterns. We report experimental results about experiment by using some data sets to confirm the usefulness of the proposed method.
Keywords Text Mining, Bioinformatics, Extracting of Information

1. はじめに

曖昧検索は，配列データベースから類似する部分文字列の検索をさし，DNAやアミノ酸モチーフの抽出，肺の損傷が類似した進行特性をもつ患者の発見などへの応用として期待されている．モチーフとは，PROSITE[1]やPfam[2]などで見られる生物学的に重要な機能をもつ特徴的なパターンである．今までに，数多くの曖昧検索の研究[3~8]が行われてきたが，従来の研究では，長さkの基準文字列と誤差半径r以内にあるk-部分文字列を全て求めるだけにとどまっている．一般に，曖昧検索では，非常に多くの類似した部分文字列の集合が検索結果として得られる．我々は，この集合をミスマッチクラスタと呼ぶ．以下では，ユーザがこのミスマッチクラスタを閲覧し，その全体像を把握することは極めて困難であるという問題に着目する．

著者らは，この問題を解決するために，反復精密化法とドメイン分割法を既に提案してきた[9,10]．反復精

密化法はミスマッチクラスタから最汎パターンを計算し，最汎パターンを用いてミスマッチクラスタに対する最小汎化された曖昧配列パターンを抽出する方法である．しかしながら，曖昧配列パターン集合を抽出するには，$O(2^{lk})$の汎化パターン集合からミスマッチクラスタを最小被覆する汎化パターンを見つけ出さなければならないため，膨大な計算時間を要する．ただし，kは曖昧配列パターン上の曖昧文字部位の数，lは各曖昧文字の候補となり得る文字数とする．この計算時間の短縮には，ユーザの背景知識に基づくドメイン分割法を反復精密化法に併用することが効果的であったが，ユーザの背景知識が得られない状況では，ドメイン分割法を利用できないという問題がある．

本論文では，ユーザの背景知識が得られていないという前提に立ち，ミスマッチクラスタから曖昧配列パターン集合を効率的に抽出するために,以下の2点の特徴的な処理から成る段階的一般化法を提案する．

(1)ミスマッチクラスタの部分集合について，小さなサイズの部分集合から順に，部分集合に対する最汎パターンを列挙する中で，不要な部分列挙木を除去する．不要な部分列挙木の検出判定は，各最汎パターンに対してパターン切除操作を適用することによって実施される．

(2)作成された列挙木に含まれるすべてのリーフノードに対して，冗長性を除去し，これによって得られるリーフノードの集まりを解集合(ミスマッチクラスタに対する最小汎化パターン集合)とする．

また，段階的一般化法の有効性を確認するために，PROSITEに含まれるいくつかのデータセットを用いて，段階的一般化法と反復精密化法との性能比較を行う．

2. 関連研究

データの集合に対して汎化を行う処理は，述語論理に基づく帰納論理プログラミング[12,13]，Quinlan の決定木学習[14]，概念階層を用いたデータベースの属性指向帰納法[15]，PrefixSpan 法に基づく可変長配列パターン抽出法[16]などに見られる．

帰納論理プログラミングでは，正や負のファクト集合を導出することが可能なホーン節ルールを抽出するために，最も一般的な仮説から探索を始め仮説の特殊化を繰り返し進めることにより解を見つけ出すトップダウン探索[12]やファクトと背景知識を組み合わせて特殊な仮説を段階的に生成することにより解を見つけ出すボトムアップ探索[13]などがある．これらの探索において，2 種類の操作が仮説に適用されている．その 1 つは，仮説の一般化であり，それは，リテラルを変数に置き換えたり,節の条件部からアトムを除去したり,節により表現されている仮説集合に新しい節を追加したりすることによって実施されている．他の 1 つは，仮説の特殊化であり，それは，変数をリテラルに置き換えたり，節中の 2 つの変数を単一化したり，節の条件部にアトムを追加したり，仮説集合から節を除去することにより達成されている．

決定木学習では，ある分類について表現された複雑なデータの集まりに対して，分類に参考となる属性を用いて分類木を作成することにより，余分な属性を削除した簡単なデータ表現を作成する方法が採られている．

属性指向帰納法では，属性値に関する概念階層木を用いて，データベースリレーションの属性値を一般化する方法が採用されている．

PrefixSpan 法は，述語論理に基礎を置く帰納論理プログラミングの分野とは異なり購買データのバスケット解析から生まれた計算手法である[17]．PrefixSpan 法に基づく可変長配列パターン抽出法では，可変長ワイルドカード領域を有する頻出配列パターンを抽出する方法であるが，そこでは，配列データ上のアルファベット文字をワイルドカードに置き換えることにより一般化する方法が採用されている．

本論文では，PrefixSpan に基づく曖昧配列パターン抽出に関係しており，解くべき問題は，ミスマッチクラスタに対して最小被覆となる曖昧配列パターンを抽出することである．このとき，曖昧配列パターン上に存在する各曖昧文字領域を表現するドメインとよばれる文字集合を見つけ出さねばならない．この文字集合は，抽出される曖昧文字ドメイン間の直積により導かれる文字列集合がミスマッチクラスタ内に存在しなければならないという制約を満足している．

著者らは，既にトップダウン探索に基づく反復精密化法[9,10]を提案し，この制約を満足する曖昧配列パターン集合の抽出に成功している．本論文では，ボトムアップ探索に基づく段階的な一般化の方法を提案している．この方法は，段階的に構築される最汎パターンに対してパターン切除操作を適用することにより不要な部分列挙木を見つけだせるので，曖昧配列パターン集合を効率的に抽出することができる．

3. 用語と記号の定義

配列データベース $DB=\{t_1,t_2,\ldots,t_m\}$において，各要素を$(sid,s_{sid})$と表現する($m$ は要素数, sid は配列識別子)．配列データベースの配列識別子の集合は $\Omega=\{1,2,3,\ldots,m\}$と表現する．各 s_{sid} は，あるアルファベット Σ 上で定義される文字列であり，配列識別子の値として sid を持つ配列データと呼ぶ．配列データ s_{sid} の先頭から j 番目の文字は，$s_{sid}[j]$を用いて表現する．

3.1. ミスマッチクラスタ

曖昧検索では，ユーザが長さkを持つある部分文字列(k-部分文字列)と許容誤差dを検索条件として与える．曖昧検索(類似性検索)により，配列データベース$DB=\{t_1,t_2,...,t_m\}$から取得される部分文字列の集合をミスマッチクラスタMISと呼ぶ．以後，このミスマッチクラスタMISを次の形式で表現する．

$$MIS=\{ <inst_1>, <inst_2>, ・・・, <inst_n> \} \quad (1)$$

3.2. 曖昧配列パターン

Σ_iをアルファベットΣの部分集合とすると，k-曖昧配列パターン(k個の曖昧文字を並べた曖昧配列パターン)は以下の形式で表現する．

$$<pat^k> = <\Sigma_1\text{-}x(i_1,j_1)\text{-}\Sigma_2\text{-}x(i_2,j_2)\text{-} ...\text{-}\Sigma_{k-1}\text{-}x(i_{k-1}j_{k-1})\text{-}\Sigma_k>:cnt \quad (2)$$

式(2)中の，$\Sigma_i \subseteq \Sigma$が存在する場所を曖昧文字領域と呼ぶ．また，$|\Sigma_i| \geqq 2$のとき，集合$\Sigma_i$は曖昧文字ドメインと呼ばれ，$\Sigma_i$内に存在する任意の1文字の配置が許されることを示している．曖昧文字ドメインが1個以上存在するとき，式(2)をk-曖昧配列パターンと呼ぶ．ハイフン“-”は左右の文字の接続を意味するが，以後たびたび省略されることがある．$x(i,j)$は，ワイルドカード領域と呼ばれ，ワイルドカード数がi個からj個までの範囲内であることを示している．$i<j$のとき，$x(i,j)$を可変長ワイルドカード領域と呼び，$i=j$のとき，$x(i,j)$は$x(i)$と書き，$x(i)$は，固定長ワイルドカード領域と呼ぶ．$x(0)$は空文字を意味する．

例えば，以下の*Kringle*モチーフについて考えてみよう．

$$<[FY]\text{-}C\text{-}[RH]\text{-}[NS]\text{-}x(7,8)\text{-}[WY]\text{-}C> \quad (3)$$

上記の$[FY]$, C, $[RH]$, $[NS]$, $[WY]$, C は，それぞれ曖昧ドメイン $\Sigma_1=\{F,Y\}$，$\Sigma_2=\{C\}$，$\Sigma_3=\{R,H\}$，$\Sigma_4=\{N,S\}$，$\Sigma_5=\{W,Y\}$，$\Sigma_6=\{C\}$により定義されているとみなすことができる．

式(2)の最右端にある記号cntは，たびたび省略されることがあるが，そのパターンの支持数を意味する．パターンの支持数とは，そのパターンが出現する異なる配列データの数をさす．

以下では，曖昧表現に関する本質を浮き彫りにするために，ワイルドカード領域の部分の記述を省略し，k-曖昧配列パターンを以下のように表現する．

$$<pat^k>=<\Sigma_1\text{-}\Sigma_2\text{-}...\text{-}\Sigma_{k-1}\text{-}\Sigma_k> \quad (4)$$

特に，$\Sigma_i=\{\alpha_i\}$のとき，式(2)を以下のように表現する．

$$<pat^k>=<\alpha_1\text{-}\ \alpha_2\ \text{-}\ ...\ \text{-}\ \alpha_{k-1}\ \text{-}\ \alpha_k> \quad (5)$$

ただし，$\alpha_i \in \Sigma$とする．

3.3. インスタンスを導出する関数

k-曖昧配列パターン$<\Sigma_1\Sigma_2...\Sigma_{k-1}\Sigma_k>$からインスタンスのすべて(部分文字列の集合)を導出する関数を$EVAL(<\Sigma_1\Sigma_2...\Sigma_{k-1}\Sigma_k>)$と書くことにすると，以下の関係式が成立する．

$$EVAL(<\Sigma_1\Sigma_2 ...\Sigma_{i-1}(\Sigma_i - \{\alpha_i\})\Sigma_{i+1} ...\Sigma_k>) = EVAL(<\Sigma_1\Sigma_2 ...\Sigma_{i-1}\Sigma_i\Sigma_{i+1} ...\Sigma_k>) - EVAL(<\Sigma_1\Sigma_2 ...\Sigma_{i-1}\{\alpha_i\}\Sigma_{i+1}...\Sigma_k>) \quad (6)$$

以後，ある k-曖昧配列パターンが関数 $EVAL$ によって，複数の k-インスタンスから成る集合を導出することができるとき，その k-曖昧配列パターンを k-汎化パターン(あるいは単に汎化パターン)と呼ぶ．

あるk-汎化パターン集合P^k に含まれる2つのk-汎化パターンを$<pat_1^k>$,$<pat_2^k>$としよう．$EVAL(<pat_1^k>) \supseteq EVAL(<pat_2^k>)$が成立するとき，$<pat_1^k>$は$<pat_2^k>$を概念的に包括するという．別の言い方では，$<pat_2^k>$は$<pat_1^k>$に冗長であるともいう．

3.4. 最汎パターン

ある k-インスタンス(長さ k の部分文字列)の集合を I^k としよう．$1 \leqq j \leqq k$ に対して，アルファベット Σ_j を以下のように定義する．

$$\Sigma_j = \{ inst[j] \mid inst \in I^k \} \quad (7)$$

このとき，$<\Sigma_1\Sigma_2...\Sigma_{i-1}\Sigma_i...\Sigma_k>$を k-インスタンス集合 I^k に対する最汎パターン$<mgpat^k>$と呼ぶ．

また，ある k-汎化パターンの集合を P^k とするとき，$1 \leqq j \leqq k$ に対して，アルファベット Σ_j を以下のように定義する．

$$\Sigma_j = \{string[j] \mid string \in EVAL(<pat^k>), <pat^k> \in P^k \} \quad (8)$$

このとき，$<\Sigma_1\Sigma_2...\Sigma_{i-1}\Sigma_i...\Sigma_k>$を$k$-汎化パターン集合$P^k$に対する最汎パターン$<mgpat^k>$と呼ぶ．

3.5. 正と負のオブジェクト

長さ k をもつ MIS の最汎パターン$<mgpat^k>$および長さ k を持つある汎化パターン$<pat^k>$について考えてみよう．MIS の要素を正のインスタンスと呼び，$EVAL(<mgpat^k>) - MIS$ の要素を負のインスタンスと呼ぶ．$EVAL(<pat^k>)$が MIS に含まれれば，汎化パターン$<pat^k>$を正の汎化パターンと呼び，$EVAL(<pat^k>)$が $EVAL(<mgpat^k>) - MIS$ に含まれれば，汎化パターン$<pat^k>$を負の汎化パターンと呼ぶ．

4. 反復精密化法

$<pat^k>$は $EVAL(<pat^k>)$を最小被覆する汎化パターンであるので，$<pat^k>$を $EVAL(<pat^k>)$に対する最小汎化パターンと呼ぶ．反復精密化法は，2 段階で処理が進められる．最初に，負の最小汎化パターン集合を計算する．次に，正の最小汎化パターン集合を計算する．この正の最小汎化パターン集合が，ミスマッチクラスタ MIS に対する最小汎化パターンとなる．それぞれの段階では，パターン切除操作を用いた最小被覆パターン集合の計算が中心であるので，先ずこの集合を計算する方法について述べる．

4.1. パターン切除操作

k-汎化パターン$<pat^k>=<\Sigma_1\Sigma_2...\Sigma_{i-1}\Sigma_i...\Sigma_k>$と k-イン

スタンス集合 I^k={<$inst_1^k$>, <$inst_2^k$>,...,<$inst_n^k$>}があるとしよう．$EVAL(<pat^k>)-I^k$ を最小被覆する k-汎化パターンの集合を $COVS(<pat^k>, I^k)$で表現する．特に，<$inst^k$>=<α_1- α_2 - ... - α_{k-1} - α_k>に対して I^k={<$inst^k$>}のとき，$EVAL(<pat^k>)-${<$inst^k$>}を最小被覆する k-汎化パターンの集合 $COVS(<pat^k>,{<inst^k>})$は，k-汎化パターン<pat^k>から<$inst^k$>をパターン切除する操作 $PCUT$ により計算可能である． $PCUT$ は以下の通りである．

$$PCUT(<pat^k>,<inst^k>) = \{ <\Sigma_1\Sigma_2...\Sigma_{i-1}(\Sigma_i-\{\alpha_i\})...\Sigma_k> \mid 1\leqq i\leqq k \} \quad (9)$$

$\alpha_i \in \Sigma_i$であれば，$PCUT$ (<pat^k>, <$inst^k$>)は，最大 k 個の汎化パターンを生成し，$\alpha_i \notin \Sigma_i$ であれば，$PCUT$ (<pat^k>,<$inst^k$>)={<pat^k>}となる$(1\leqq i\leqq k)$.

I^k={<$inst_1^k$>, <$inst_2^k$>}のとき，$EVAL(<pat^k>)-I^k$ を最小被覆する k-汎化パターンの集合 $COVS(<pat^k>, I^k)$は，以下のように計算可能である．

$$COVS(<pat^k>,\{<inst_1^k>, <inst_2^k>\}) = \{ pat \mid pat\in PCUT(x,<inst_2^k>), x\in PCUT(<pat^k>,<inst_1^k>)\}. \quad (10)$$

$|I^k|\geq 2$ のとき，I^k ={<$inst^k$>}$\cup SubI^k$, $|SubI^k|\geq 1$ の関係が成立するとしよう．このとき，式(10)を以下の再帰的な計算式に拡張することが可能になる．

$$COVS(<pat^k>, I^k) = \{ pat \mid pat\in COVS(x, SubI^k), x\in PCUT(<pat^k>,<inst^k>) \},$$

ただし，$COVS(<pat^k>,\{<inst^k>\}) = PCUT(<pat^k>, <inst^k>).$ (11)

式(9)において，k-インスタンス<$inst^k$>=<α_1- α_2 - ... - α_{k-1} - α_k>を k-汎化パターン<$\Gamma_1\Gamma_2$ ... Γ_k>に置き換えてみよう．パターン切除操作 $PCUT$ を以下のように拡張することができる．ただし，$\Gamma_i\subseteq\Sigma$ とする$(1\leqq i, j\leqq k)$.

$$PCUT(<\Sigma_1\Sigma_2...\Sigma_k>,<\Gamma_1\Gamma_2...\Gamma_k>) = \{ <\Sigma_1\Sigma_2...\Sigma_{i-1}(\Sigma_i-\Gamma_i)...\Sigma_k> \mid 1\leqq i\leqq k \} \quad (12)$$

Γ_i Σ_i かつ $\Gamma_i\cap\Sigma_i$ φ であれば，$PCUT$ (<$\Sigma_1\Sigma_2...\Sigma_k$>, <$\Gamma_1\Gamma_2...\Gamma_k$>)は，$k$ 個の汎化パターンを要素として返し，$\Gamma_i\cap\Sigma_i=\varphi$ であれば，$PCUT$ (<$\Sigma_1\Sigma_2...\Sigma_k$>,<$\Gamma_1\Gamma_2...\Gamma_k$>)は{<$\Sigma_1\Sigma_2...\Sigma_k$>}を返す$(1\leqq i\leqq k)$．特に $\Gamma_i=\Sigma_i$ であれば，$PCUT$ (<$\Sigma_1\Sigma_2...\Sigma_k$>, <$\Gamma_1\Gamma_2...\Gamma_k$>)は空集合 φ を返す．

4.2. 反復精密化法

以下では，反復精密化法により，ミスマッチクラスタ MIS に対する最小汎化パターンの計算例を示す．ミスマッチクラスタ MIS を{<ABF>, <AEC>, <AEF>, <DBF>, <DEC>, <DEF> }とすると，この最汎パターンは<[AD][BE][CF]>となる．式(11)を用いて，負の最小汎化パターン集合の計算例を示すと，後述の(1)～(6)の処理になる．引き続き，正の最小汎化パターン集合の計算例を示すと，後述の(7)の処理になる．

負の最小汎化パターン集合 $COVS$(<[AD][BE][CF]>, MIS)の抽出処理を以下に示す．

(1) 最汎パターン<[AD][BE][CF]>から MIS の第一要素<ABF>をパターン切除した結果は以下のとおりである．

$PCUT$(<[AD][BE][CF]>,<ABF>)= {<D[BE][CF]>,<[AD]E[CF]>,<[AD][BE]C>}

(2) <[AD]E[CF]>と<[AD][BE]C>から MIS の第二要素<AEC>をパターン切除した結果は，それぞれ以下のとおりである．

$PCUT$(<[AD]E[CF]>,<AEC>)={<DE[CF]>,<[AD]EF>}

$PCUT$(<[AD][BE]C>,<AEC>)={<D[BE]C>,<[AD]BC>}

しかしながら，<DE[CF]>と<D[BE]C>は，上記(1)で得られた<D[BE][CF]>に含まれるので，削除する．

(3) <[AD]EF>から MIS の第三要素<AEF>をパターン切除した結果は空集合である．

(4) <D[BE][CF]> から MIS の第四要素<DBF>をパターン除去した結果は以下のとおりである．

$PCUT$(<D[BE][CF]>,<DBF>)={<DE[CF]>,<D[BE]C>}

(5) <DE[CF]>と<D[BE]C>から MIS の第五要素<DEC>をパターン切除した結果はどちらも空集合である．

(6) 残った汎化パターン<[AD]BC>には MIS の第五要素<DEF>が含まれないので，パターン切除の計算は不要である．

以上により，{<[AD]BC>}が負の最小汎化パターン集合となる．前述の(1)～(6)の処理に対応する探索過程を図 1 に示す．

次に，正の最小汎化パターン集合 $COVS$(<[AD][BE][CF]>,{<[AD]BC>})を計算してみよう．

(7) 最汎パターン<[AD][BE][CF]>から負の最小汎化パターン<[AD]BC>をパターン切除した結果は以下のとおりである．

$PCUT$(<[AD][BE][CF]>,<[AD]BC>)= {<[AD]E[CF]>,<[AD][BE]F>}

即ち，$COVS$(<[AD][BE][CF]>,{<[AD]BC})= {<[AD]E[CF]>, <[AD][BE]F>}となる．従って，<[AD]E[CF]> と <[AD][BE]F> が MIS={<ABF>,<AEC>, <AEF>,<DBF>,<DEC>,<DEF>}に対する最小汎化パターンとなる．

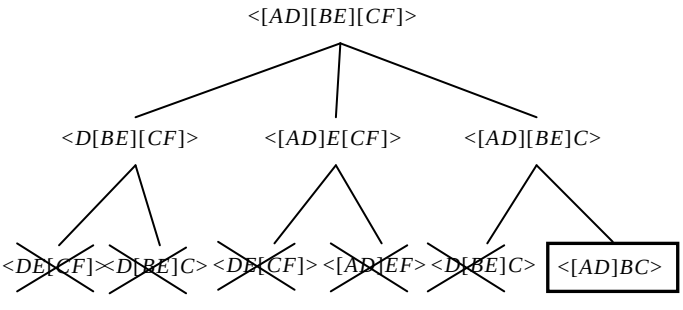

図 1. ミスマッチクラスタ MIS に対する負の最小汎化パターンの計算例

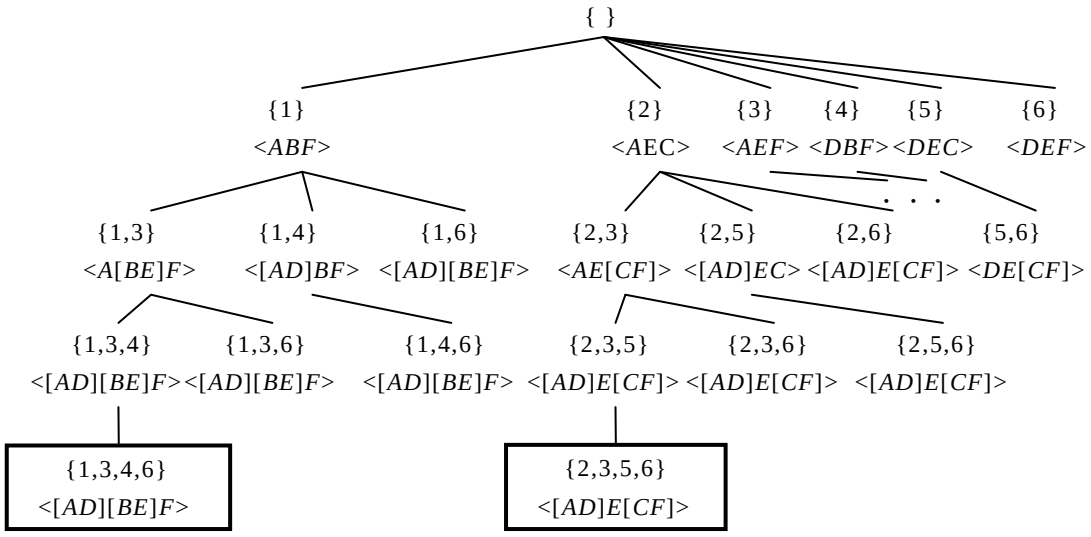

図 2. 列挙木による探索処理の例

5. 段階的一般化法

我々が新しく提案する段階的一般化法は，ミスマッチクラスタ *MIS* に含まれる要素を少しずつ組み合わせることにより最小汎化パターンを探索するボトムアップアプローチである．ミスマッチクラスタ *MIS* の冪集合を *P*(*MIS*)で表現するとき，包含関係により半順序付けられた *P*(*MIS*)は半順序集合であり，ハッセ図を用いて表現可能である．このハッセ図の一部は，サイズの小さな集合 $S \in P(MIS)$から順に列挙する列挙木と同一である．ただし，この列挙木の作成においては深さ優先に進められるが，親ノードから子ノードを列挙するときだけは，幅優先に実施する．

5.1. 列挙木を用いた抽出例

図 2 には，*MIS*={<*ABF*>, <*AEC*>, <*AEF*>, <*DBF*>, <*DEC*>, <*DEF*> }に対して最小被覆をもつ汎化パターンを抽出するための列挙木の一部分が描かれている．この列挙木では，*MIS* の各要素に 1 から 6 までの識別番号を付与する．即ち，*MIS* = { (1,<*ABF*>), (2,<*AEC*>), (3,<*AEF*>), (4,<*DBF*>), (5,<*DEC*>), (6,<*DEF*>)}とする．また，列挙木の各ノードはその識別番号の集合 *S* を用いて表現する．深さ 2 以降の各ノードには，列挙された識別番号集合 *S* とその集合 S に対する最汎パターンの計算結果が併記されている．親ノードから子ノードを列挙するときは，親ノードの識別番号集合 S_1 に含まれるどの識別番号よりも大きい識別番号をもつ識別番号集合 S_2 をその親の兄弟ノードから探し，集合和 $S_1 \cup S_2$ をとることによりその親の子ノードを生成する．例えば，親ノードに対する識別番号集合{ 1, 3 }とその親の兄弟ノードに対する識別番号集合{ 1, 4 }の集合和をとると，その親の子ノードに対する識別番号集合{ 1, 3, 4 }が生成される．その親の子ノードに対する最汎パターンを計算すると<[*AD*][*BE*]*F*>が得られる．

列挙木による探索処理において，あるノードで計算された最汎パターン<*pat*>が *MIS* 以外の要素を概念的に包括するとき，そのノードから深さ方向の探索を打ち切ることができる．即ち，もし，以下の条件式が成立すれば，深さ方向探索における枝刈りが可能となる．

$$EVAL(<pat>) - MIS \quad \phi \tag{13}$$

例えば，親ノードの識別番号集合{ 1 }とその親の兄弟ノードの識別番号集合{ 2 }の集合和をとると，その親の子ノードの識別番号集合{ 1, 2 }が生成される．このとき，子ノード{ 1, 2 }から計算される最汎パターンは <*A*[*BE*][*CF*]> となる．式 (13) により *EVAL*(<*A*[*BE*][*CF*]>)－*MIS* ={ <*ABC*> } ϕとなるため，この最汎パターン<*A*[*BE*][*CF*]>をもつノードから深さ方向の探索を打ち切ることができる．

列挙木による探索処理が終了すると，各リーフに対する最汎パターンの集合が解集合となるが，冗長なパターンは解集合から削除すべきである．例えば，リーフの{ 1, 3, 6 }，{ 1, 4, 6 }，{ 5, 6 }，{ 6 }などは，リーフの{ 1, 3, 4, 6 }に冗長であるので，それらは解集合から削除されるべきである．図 2 の例では，{<[*AD*][*BE*]*F*>, <[*AD*]*E*[*CF*]> }が解集合である．即ち，この集合が *MIS* に対する最小汎化パターン集合である．

5.2. 段階的一般化法の処理手順

ミスマッチクラスタ *MIS*={ (1, <$inst_1$>), (2, <$inst_2$>),・・・,(*n*, <$inst_n$>) }から最小汎化パターン集合を抽出するとしよう．この最小汎化パターン集合を抽出するための処理手順は以下のとおりである．

(1)ミスマッチクラスタ *MIS* に含まれる識別番号を

Stack に格納する．すなわち，*Stack* := { {1}, {2},・・・,{n}}とする．ただし，スタック領域に格納されている番号は昇順に取り出せるように管理する．候補解を格納する領域 *Candidate* を *Candidate* := { }とする．

(2)*Stack* が ϕ になるまで，*Stack* から集合 *S* を 1 つずつ取り出し，以下の処理を行う．

・*S* を親ノードとするとき，その兄弟ノードを用いて，子ノード *C* を全て列挙する．

・列挙された各子ノード *C* について，以下の処理を行う．

- 子ノード *C* の最汎パターン<*pat*>を計算する．
- もし，*EVAL*(<*pat*>)－*MIS* ϕ ならば，子ノード *C* を捨てる．
  そうでなければ，子ノード *C* を *Stack* に格納する．ただし，*Stack* に追加される子ノードは列挙された順に取り出せるように管理する．
  また，*S*:={*C*,<*pat*>}とし，*Candidate* に非冗長な子ノード *C* をもつ *S* だけを *Candidate* に追加する．

(3)*Candidate* から冗長なパターン<*pat*>をもつ要素を除去する．

(4)*Candidate* を最小汎化パターン集合として出力する．

5.3. 枝刈り条件の高速計算

式(13)において，*EVAL*(<*pat*>)の計算結果は大規模になる傾向があるため，その式を直接用いた判定計算は非現実的である．我々は，親ノード，兄弟ノード，子ノードで計算された各最汎パターンを<pat_p>, <pat_s>, <pat_c>とし，以下の式を用いて枝刈り条件の判定を行っている．

$$\{\, pat \mid pat \in PCUT(x, \langle pat_s \rangle),\ x \in PCUT(\langle pat_c \rangle, \langle pat_p \rangle) - MIS \neq \phi \,\} \qquad (14)$$

図 2 の抽出例を用いて，式(11)の計算例を示してみよう．親ノードに対する識別番号集合{ 1, 3 }，その親の兄弟ノードに対する識別番号集合{ 1, 4 }，その親の子ノードに対する識別番号集合{ 1, 3, 4 }は，最汎パターンとしてそれぞれ<pat_p>=<*A*[*BE*]*F*>, <pat_s>=<[*AD*]*BF*>, <pat_c>=<[*AD*][*BE*]*F*>を与える．子ノードに対する識別番号集合{ 1, 3, 4 }において，*EVAL*(<[*AD*][*BE*]*F*>)－*MIS* が空集合となるか否かの判定は，以下のように行う．式(11)の *PCUT* (<pat_c>, <pat_p>)により，子ノードに対する識別番号集合{ 1, 3, 4 }の最汎パターン<pat_c>=<[*AD*][*BE*]*F*>から親ノードに対する識別番号集合{ 1, 3 }の最汎パターン<pat_p>=<*A*[*BE*][*CF*]>をパターン切除した結果は，{<D[BE]F>}となる．次に，式(11)の *PCUT*(*x*,<pat_s>) により，*x*=<*D*[*BE*]*F*>から兄弟ノードに対する識別番号集合{ 1, 4 }の最汎パターン<pat_s>=<[*AD*]*BF*>をパターン切除した結果は，{<*DEF*>}となる．これは *MIS* に含まれるので，式(11)を満たさない．従って，この場合は，枝刈り条件を満たさないので，識別番号集合{ 1, 3, 4 }をもつ子ノードから深さ方向の探索を進めることができる．親ノードに対する識別番号集合{ 1 }，兄弟ノードに対する識別番号集合{ 2 }，子ノードに対する識別番号集合{ 1, 2 }について考えてみよう．親ノードと兄弟ノードの部分文字列は各々<*ABF*>と<*AEC*>である．子ノードで計算される最汎パターンは<*A*[*BE*][*CF*]>である．式(11)の *PCUT*(<pat_c>,<pat_p>) により，*PCUT*(<*A*[*BE*][*CF*]>, <*ABF*>)={<*AE*[*CF*]>, <*A*[*BE*]*C*>}となり，ここで得られた 2 つの要素に対して，式(11)の *PCUT*(*x*, <pat_s>) を計算すると，*PCUT*(<*AE*[*CF*]>,<*AEC*>)∪*PCUT*(<*A*[*BE*]*C*>, <*AEC*>)= {<*AEF*>,<*ABC*>}が得られる．<*ABC*>は *MIS* に含まれていないので，*EVAL*(<*A*[*BE*][*CF*]>)－*MIS* ϕとなることがわかる．従って，この場合は識別番号集合{ 1, 2 }をもつ子ノードから深さ方向の探索が打ち切られる．

6. 実験評価

表 1 に実験評価に用いられた 4 種類のデータセットの特徴を示す．最初に，4 種類のデータセット Cytochrome C (登録番号:PS00814)，Zinc Finger (登録番号:PS00028)，Leucine (登録番号:PS00717)，Kringle (登録番号 :PS00021)のそれぞれに対して，サフィックス木[11]を用いて曖昧パターン検索を行う．次に，反復精密化法，段階的一般化法の 2 種類の手法を用いて，出力として得られたミスマッチクラスタに対して，最小汎化を行う．これにより，段階的一般化法の有効性を確認する．このために利用した計算機環境は，Intel Pentium(R) 4-2.53GHz，メモリー：1.5GB，SWAP メモリー：2GB，HDD：80GB，OS：Fedora 7 である．

表 1. データセット詳細

データセット	データ件数	総長(bytes)	モチーフ表現数
Cytochrome C	29	4325	120
Leucine	27	13088	192
Zinc Finger	467	245595	72
Kringle	88	58453	32

表1に示される4種類のデータセットに含まれるモチーフはそれぞれ以下の通りである．

(1) Cytochrome C :

<*C*-*x*(2)-[*STAQ*]-*x*-[*STAMV*]-*C*-[*STA*]-*T*-*C*-[*HR*]>

(2) Leucine :

<*P*-[*LIVM*]-*x*(2)-[*LIVM*]-*A*-*x*(2)-[*LIVMFT*]-*x*(2)-[*HS*]-*x*-*S*-*T*-[*LIVM*]-*S*-*R*>

表 2. 曖昧検索，汎化処理の結果

データセット	検索キーと許容誤差半径	ミスマッチクラスタ数(個)	最小汎化パターン数(個)
CytochromeC	<C－x(2)－A－x－A－C－A－T－C－R> 4	29	7
Leucine	<P－L－x(2)－L－A－x(2)－L－x(2)－H－x－S－T－L－S－R> 5	31	14
ZincFinger	<C－x(2,4)－C－x(3)－L－x(8)－H－x(3,5)－H> 1	2729	38
Kringle	<Y－C－R－N－x(7,8)－W－C> 4	3552	905

表 3. 段階的一般化法と反復精密化法の計算時間

データセット	反復精密化法による計算時間(sec):A	段階的一般化法による計算時間(sec):B	性能比(A/B)
CytochromeC	0.041	0.003	13.66
Leucine	761.0	0.044	17295
Zinc Finger	0.816	2.229	0.366
Kringle	—	82.20	—

(3) Zinc Finger:

<*C*-*x*(2,4)-*C*-*x*(3)-[*LIVMFYWC*]-*x*(8)-*H*-*x*(3,5)-*H*>

(4) Kringle:

< [*FY*]-*C*-[*RH*]-[*NS*]-*x*(7,8)-[*WY*]-*C*>

また，表 1 のモチーフ表現数はモチーフに含まれるインスタンス数を表している．曖昧検索における検索キーは各データセットのモチーフの一部分を用い，モチーフが可変長ワイルドカード領域を含む場合は，その一部分が全て網羅された固定長ワイルドカード領域が含まれる部分文字列の集合に変換し，その集合を検索キーとして実験を行う．

6.1. 実験結果

実験に用いた曖昧検索条件(検索キーと許容誤差半径)に対する曖昧検索結果，汎化処理の結果，汎化処理に要した計算時間などを表 2，表 3 に示す．なお，2 つの手法の出力結果が等しく，またここで得られた汎化パターン集合は曖昧パターン検索によって得られたミスマッチクラスタ内の全ての要素を被覆していることをプログラムで確認している．計算時間を比較すると，提案手法である段階的一般化法は反復精密化法に比べて，Cytochrome C では約 14 倍の高速化，Leucine では約 17300 倍の高速化に成功したといえる．さらに，Kringle の汎化処理は，反復精密化法では計算を終えることができなかったが，段階的一般化法を用いることにより最小汎化パターンを高速に計算することに成功した．

Zinc finger では 2 つの可変長ワイルドカード領域をそれぞれ複数の固定長ワイルドカード領域に変換して実験を行ったが，変換された 2 つの固定長ワイルドカード領域どうしの組み合わせ方により実行時間にばらつきがみられた．特に検索キーに *C*-*x*(2)-*C*-*x*(3)-*L*-*x*(8)-*H*-*x*(4)-*H*>を用いたときは，実行時間が反復精密化法の 10 倍程度にしかならなかった．

6.2. 考察

このように，段階的一般化法を用いることにより多くのデータセットにおいて汎化処理の高速化に成功した．一方，Zinc Finger の(一部の)データセットでは，反復精密化法の方が高速であった．これは，曖昧検索の許容誤差半径が 1 であったため，抽出されるミスマッチクラスタが非常に似通ったものであったことが原因であると考える．実際，検索キーとして<*C*-*x*(2)-*C*-*x*(3)-*L*-*x*(8)-*H*-*x*(4)-*H*>を用いたとき，検索結果として返されたミスマッチクラスタ内には，他の場合に比べてかなり類似したインスタンス(部分文字列)が多かった．段階的一般化法ではインスタンスどうしの類似性が高まると *PCUT* による部分列挙木の枝刈りができず，探索ノードが増えるためより多くの時間がかかる．逆に反復精密化法では，初めに生成される最汎パターンの曖昧文字ドメインが小さくなる．以上の理由により，Zinc Finger の(一部の)データセットでは，反復精密化法が段階的一般化法よりも高速になったと思われる．

7. まとめ

本論文では，ミスマッチクラスタに対する汎化処理の計算時間を短縮するため，段階的一般化法を提案した．提案手法の有効性を確かめるため，4 種類のデータセットを用いて実験を行った．その結果，Cytochrome C データセットでは約 14 倍，Leucine データセットでは約 17300 倍の高速化に成功した．さらに，反復精密

化法では計算できなかった Kringle データセットにおいても，最小汎化パターンを高速に計算することができた．

今後の課題としては，アミノ酸の特徴を考慮したドメイン分割法の併用や，可変長ワイルドカード領域への対応の研究が重要であると考えられる．

謝辞

本研究の一部は，日本学術振興会，科学研究費補助金(基盤研究(C)，課題番号：17500097) の支援により行われた．

[参考文献]

[1] http://kr.expasy.org/prosite/.

[2] http://www.sanger.ac.uk/Software/Pfam/

[3] Erick L.L. Sonnhamer, Sean R. Eddy, and Richard Durbin: Pfam: A Comprehensive Database of Proteins, Vol. 28, pp.405-420, 1997

[4] Zvi Galil, Kunsoo Park: An Improved Algorithm for Approximate String Matching, SIAM Journal on Computing, Vo.19, No.6, pp.989-999 (1990).

[5] Sanghyun Park, Wesley W. Chu, Jeehee Yoon, Chihcheng Hsu: Efficient Searches for Similar Subsequences of Different Lengths in Sequence Databases, Proceedings of 16th International Conference on Data Engineering (ICDE2000), IEEE Computer Society Press, pp.23-32 (2000).

[6] Marie-France Sagot: Spelling Approximate Repeated or Common Motifs Using a Suffix Tree, Theoretical Informatics, Third Latin American Symposium (LATIN 1998), pp.374-390 (1998).

[7] Pavel A. Pevzner and Sing-Hoi Sze: Combinatorial approaches to finding subtle signals in DNA sequences, Proceedings of the 8th International Conference on Intelligent Systems for Molecular Biology, pp.269-278 (2000).

[8] Eleazar Eskin and Pavel A. Pevzner: Finding composite regulatory patterns in DNA sequences, Proceedings of the 10th International Conference on Intelligent Systems for Molecular Biology, pp.269-278, pp.354-363 (2002).

[9] 荒木康太郎，田村慶一，加藤智之，北上　始：ミスマッチクラスタに対する最小汎化パターン抽出方式，日本データベース学会論文誌（DBSJ Leters)，Vol.6, No.3, pp.5-8, 2007 年 12 月．

[10] Kotaro Araki, Keiichi Tamura, Tomoyuki Kato, Yasuma Mori, and Hajime Kitakami: Extraction of Ambiguous Sequential Patterns with Least Minimum Generalization from Mismatch Clusters, Proceedings of The Third International Conference on Signal-Image Technology & Internet-Base system (SITIS' 2007): Track Information Management & Retrieval Technologies (IMRT), IEEE Computer Society Press, pp.32-39, December 16-19 in 2007.

[11] Ching-Fung Cheung, Jeffrey Xu Yu, Hongjun Lu: Constructing Suffix Tree for Gigabyte Sequences with Megabyte Memory, IEEE Transaction Knowledge Data Engineering, Vo.17, No.1, pp.90-105 (2005).

[12] Ehud Y. Shapiro, Inductive Inference of Theories from Facts, Computational Logic - Essays in Honor of Alan Robinson, pp.199-254, 1991.

[13] Stephen Muggleton, Inverse Entailment and Progol, New Generation Computing, Vol.13, No.3&4, pp.245-286, 1995.

[14] Quinlan, J. R. C4.5: Programs for Machine Learning, Morgan Kaufmann Publishers, 1993.

[15] Usama M. Fayyad, Gregory Piatetsky-Shapiro, Padhraic Smyth, and Ramasamy Uthurusamy Editors: Advances in Knowledge Discovery and Data Mining, AAAI Press/The MIT Press, 1996.

[16] 加藤 智之，北上 始，森 康真，田村 慶一，黒木 進：極小かつ非冗長な可変長ワイルドカード領域を持つ頻出パターンの抽出，電子情報通信学会論文誌D「データ工学特集号」，Vol.J90-D, No.2, pp.281-291，2007年2月．

[17] Jan 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), IEEE Computer Society Press, p.215-224, 2001.