情 報 学 基 礎 38－5
（1995．7．28）

複数文字列パターンによるアミノ酸配列からのタンパク質モティーフの発見

山口 美千代[1], 篠原 武[1], 藤野 亮一[2], 有村 博紀[1], 有川 節夫[3]

1) 九州工業大学情報工学部
2) (株) 新日本製鉄情報通信システム
3) 九州大学理学部

正則パターンとは定数と互いに異なる変数からなる文字列であり，その正則パターン中の変数を空文字を含む定数文字列で置き換えて得られる定数文字列全体を表わす．本研究では，正例のみから複数のパターンを同時に学習する k 極小多重汎化アルゴリズムを，アミノ酸配列からのタンパク質モチーフ発見に応用する．まず，この k 極小多重汎化アルゴリズムに例の集合から例外となる文字列を積極的に検出する探索戦略を組み込み，次にその有用性を膜貫通領域を対象とした実験で実証する．実験では，この探索戦略を用いた学習アルゴリズムが，従来からの探索戦略に比較して高い精度の仮説を，安定して，より短い時間で見つけることを確認した．

Protein Motif Discovery from Amino Acid Sequences by Sets of Regular Patterns

Michiyo Yamaguchi[1], Takeshi Shinohara[1], Ryouichi Fujino[2], Hiroki Arimura[1], Setsuo Arikara[3]

1) Department of Artificial Intelligence, Kyushu Institute of Technology
2) Nippon Steel Information Systems, Inc.
3) Research Institute of Fundamental Information Science, Kyushu University

A regular pattern is a string consisting of constant symbols and mutually distinct variables, and represents the set of the constant strings obtained by substituting possibly empty constant strings for variables. To the problem of discovering protein motifs from given amino acid sequences, we apply the learning algorithm, called k-minimal multiple generalization (k-mmg), that finds a minimally general collections of at most k regular patterns explaining all the positive examples. We incorporate into the learning algorithm a search heuristics, called minimal covering approach, for detecting exceptions in examples. The experiments on protein data show that our algorithm argumented with the new search heuristics more quickly finds accurate hypotheses from given amino acid sequences than the previous heuristics.

1 序論

ある機能をもつタンパク質に共通して現われる特徴的なパターンをモチーフという．アミノ酸配列からモチーフを抽出する問題は，分子生物学において重要な問題である．機械学習の手法をタンパク質モチーフ発見に応用する試みがおこなわれているが [1, 2, 7]，そうした手法の多くは，正例とともに負例も必要とする．著者らは，k 極小多重汎化アルゴリズムと呼ばれる，正例のみから正則パターンの集合を学習するアルゴリズムを用いる手法を開発し，アミノ酸配列からのモチーフ抽出に応用してきた [3]．この極小多重汎化手法で得られた仮説のうち高精度のものを検証すると，それらの仮説が例外パターンと呼ばれる，変数を含まない定数パターンを含んでいることが珍しくない [3]．これら例外パターンは，正例に含まれる例外的な文字列を表現し，それを正例から排除していると考えられる．そこで本研究では，例外パターンを積極的に発見し，それを正例から排除する探索戦略を学習アルゴリズムに導入する．さらに，この戦略を組み込んだ学習アルゴリズムを膜貫通領域の同定問題に適用し，実験によってこの戦略の効果を実証する．

2 正則パターン

本節では，[9] にしたがって，正則パターンとその拡張言語を定義する．集合 A に対して，$\sharp A$ で A の要素数を表す．**定数記号**の有限集合を $\Sigma = \{a, b, \ldots, A, B, \ldots\}$ とし，**変数**の可算集合を $X = \{x, y, z, x_1, x_2, \ldots\}$ とする．ここで，X と Σ は交わらないとする．Σ^* で，Σ 上の全ての有限文字列全体の集合を表し，Σ^+ で Σ 上の空でない有限文字列全体の集合 $\Sigma^* - \{\varepsilon\}$ を表す．ここに，ε は空文字である．

正則パターン [9] とは，定数記号と変数からなる文字列のことで，各変数は 1 回しか文字列中に出現しないものである．例えば，$aaxabyb$ は正則パターンである．**代入**とは，任意の $a \in \Sigma$ に対して $\theta(a) = a$ を満たすような，正則パターンからそれ自身への準同型写像 θ のことである．パターン p が表す**拡張言語** $L(p)$ とは，p 中の変数に空代入を含む定数文字列を代入して得られる定数文字列全体である [9]．本稿では，拡張言語のみを考えるので，拡張言語のことを単に言語とよぶ．パターンの集合 P に対しては，P の表わす**和言語**を $L(P) = \bigcup_{p \in P} L(p)$ と定義する．

任意の拡張正則パターン言語は，正則パターン p 中が変数の連続した出現を持たないような**標準形**と呼ばれる正則パターンで定義できる [9]．したがって，標準形のパターンのみを本論文では扱うこととし，標準形のパターン全体のなすクラスを $\mathcal{RP}_\varepsilon$ と書く．また，変数の出現数を高々 m 個に制限して得られる部分クラスを $\mathcal{RP}_{\varepsilon,m}$ と書く．さらに，任意の和言語は，$p \prec q$ となる p と q とが存在しないような，**既約形**と呼ばれるパターン集合 P で定義可能である．ある正則パターンのクラスを $\mathcal{D}$ としたとき，高々 k 個の標準形正則パターンの集合で，既約形であるもの全体のクラスを $\mathcal{D}^k$ と書く．以下では，パターンは $p, q, \ldots$ と書き，パターン集合は $P, Q, \ldots$，定数文字列集合は $S, T, \ldots$ と書くことにする．

3 極小多重汎化

正例からの学習においては，正例を説明する極小の概念を求めることがしばしば重要となる．ここで扱う概念は，高々 k 個の正則パターン言語の和によって表わされるものなので，言語の包含関係の意味で極小なものを求める必要がある．本研究で用いる学習アルゴリズムは，与えられた正例を高々 k 個の正則パターンの集合 $\{p_1, \ldots, p_l\}$ $(l \leq k)$ で一般化し，その言語の和 $L(P_1) \cup \cdots \cup L(P_l)$ が全ての正例を含むもののうちで極小なものを求める．例えば，定数記号の集合を $\Sigma = \{A, B, C, \ldots, W, Y\}$ としたとき，7 個の文字列

e_1 WLVNFIIVIMVFILFLVGLYLL
e_2 VALVTITLWFMAWTPYLVINCMGL
e_3 GFLAASALGVVMITAALAGIL
e_4 SKILGLFTLAIMIISCCGNGVVVYI
e_5 MTIKTSIMKILFIWMMAVFWT
e_6 IFYSIFVYYIPLFLICYSYWFIIAAVSA
e_7 GCGSLFGCVSIWSMCMIAFDRYNVIV

からなる集合を $S = \{e_1, \ldots, e_l\}$ とするとき, 集合 $P = \{p_1, \ldots, p_3\}$

p_1 $x_1\mathrm{F}x_2\mathrm{M}x_3\mathrm{LV}x_4\mathrm{L}$
p_2 $x_1\mathrm{FL}x_2\mathrm{V}x_3\mathrm{A}x_4$
p_3 $x_1\mathrm{LF}x_2\mathrm{M}x_3\mathrm{V}x_4$

は S の 3-mmg である. このとき, S 中の各要素が定義する拡張パターン言語 $L(p_1), L(p_2), L(p_3)$ は, それぞれ, S の部分集合 $\{e_1, e_2\}$, $\{e_3, e_6\}$, $\{e_4, e_5, e_7\}$ を覆っている. このような極小なパターン集合 P を k 極小多重汎化 (または k-mmg) という.

正確には, 以下のように極小多重汎化を定義する. 正則パターン間には, 代入によって関係 $\preceq$ が自然に定義できる.

$$p \preceq q \iff (\exists\theta:\ \text{代入}\ )\ p = \theta(q)$$

このとき, $p \preceq q$ と書き, q は p より**一般的**または, p は q より**具体的**であるという. また, $p \preceq q$ かつ $q \npreceq p$ が成立するとき, $p \prec q$ と書く. 定義より明らかに, $p \preceq q$ ならば $L(P) \subseteq L(Q)$ である. Σ が 2 つ以上の記号を含むときには, この逆も成立する. この順序 $\preceq$ は,

$$P \sqsubseteq Q \iff (\forall p \in P, \exists q \in Q)\ p \preceq q$$

によって, 正則パターン集合間の関係 $\sqsubseteq$ に拡張できる. $P \sqsubseteq Q$ が成立するとき, Q は P より**一般的**であるという. パターン集合の言語は $L(P) = \bigcup_{p\in P} L(p)$ であるので, $\preceq$ の場合と同様に $P \sqsubseteq Q$ ならば $L(P) \subseteq L(Q)$ が成立する. この逆は必ずしも成立しないが, 定数記号が十分に多ければ成立する. この逆が成立するとき, パターン集合のクラス $\mathcal{D}^k$ は**コンパクト性をもつ**という.

定義 1 文字列の集合を S とする. S の**極小多重汎化** (k-mmg) とは, $\sharp P \leq k$ かつ, $S \subseteq L(P)$ を満たすパターン集合 P で, $\sqsubseteq$ に関して極小なものをいう.

文献 [3] は, パターン集合のクラス $\mathcal{D}^k$ がコンパクト性を持つときには, 正例の k-mmg を計算することで, $\mathcal{D}^k$ が定義する言語の族が多項式時間仮説更新を用いて正例から極限同定可能であることを示している. このことは, k-mmg 計算を正例からの知識獲得に用いる根拠の一つを与えている.

4 学習アルゴリズム

以下で説明する k-mmg を計算するアルゴリズムは, 最も一般的なパターン集合 $\{x\}$ から始めて, 各パターンを具体化したり, あるいは複数のパターンに分割したりして, 仮説空間を一方向に探索する. そこで用いられるパターンの基本操作 ρ は**精密化演算子**と呼ばれ, 関係 $\preceq$ を段階的に計算するものである.

パターン p に以下のような操作を施して得られるパターン全体を $\rho(p)$ とする.

- p のある変数 x を xay で置き換える. ここに, y は p に現われない変数であり, a は任意の定数記号である.
- p のある変数 x を空語 ε で置き換える.

```
MMG(k, S)
  P := Tighten({x}, S);
  Δk := k;
  while c ( Δk ≥ 2 かつ Δk 分割可能 な p ∈ P が存在する ) do
    Δk 分割可能な p を選ぶ;                (選択点 1)
    ΔS := S − L(P − {p});
    ΔP := Divide(p, Δk, ΔS);
    ΔP := Tighten(ΔP, ΔS);
    P := (P − {p}) ∪ ΔP;
    Δk := k + ♯P − 1;
  endwhile
  Return P;
```

```
Divide(p, k, S)
  ρ(p) の部分集合 P で ♯P ≤ k かつ S に関して
      既約なもの P を選ぶ;                (選択点 2)
  Return P;

Tighten(P, T)
  while 精密化可能な p が存在する do
    精密化可能な p を選ぶ;                (選択点 3)
    Δ := TΔ − L(P − {p});
    p の ΔT に関する精密化 r を選択する;
                                          (選択点 4)
    P := (P − {p}) ∪ {r};
  endwhile
  Return P;
```

図 1: k-mmg アルゴリズム

このとき, 任意の $r \in \rho(p)$ に対して, $r \prec q \prec p$ となるような正則パターン q は存在しない. すなわち, p と r の中間のパターンは存在しない. 逆に, p より真に具体的な任意のパターンは, p に ρ を有限回適用することによって得ることができる. これを, 精密化演算子 ρ の**完全性**という. さらに, $\rho(p)$ は p の長さの多項式時間で計算できる. 正整数を m としたとき, パターン中の変数の個数を m 個以下に制限した場合には, 変数 x を xa または ax で置き換える操作を追加する.

P が S に関して**既約**であるとは, $S \subseteq L(P)$ かつ P の任意の真部分集合 P' に対して, $S \not\subseteq L(P')$ であることをいう. $S \subseteq L(p)$ であるとき, p が S に関して k **分割可能**であるとは, $\sharp P \leq k$ かつ, $S \subseteq L(P)$, $P \sqsubseteq \{p\}$ をみたす S に関して既約な P が存在することをいう. そうした P のことを, p の S に関する k **分割**という. パターン p が S に関して**精密化可能**であるとは, $S \subseteq L(r)$ をみたす $r \in \rho(p)$ が存在することをいう. こうした r のことを, p の S に関する**精密化**という.

図 1 に, k-mmg を計算する学習アルゴリズムを示す. **MMG**(k, S) は S の k-mmg を求める. 補助手続き **Divide**(p, k, S) は, p の S に関する k 分割を求め, 補助手続き **Tighten**(P, S) は, P 中のそれぞれのパターンが $S \subseteq L(P)$ をみたす限り, それらをより具体的なパターンに置き換えていって得られるパターン集合を求める.

定理 2 正の整数を k, m とし, 文字列の有限集合を S とする. 仮説空間を $\mathcal{RP}^k_{\varepsilon,m}$ としたとき, アルゴリズム **MMG** は, S の k-mmg の 1 つを時間 $O(\sharp\Sigma \cdot k^3 m^k l^2 n)$ で計算する. ここに, l は S 中の最長文字列の長さであり, n は S の要素数である.

系 3 仮説空間 $\mathcal{RP}^k_{\varepsilon,m}$ に対して, 与えられた正例の集合 S の k-mmg は, S の例の長さの総和の多項式時間で計算可能である.

5 例外を活用する戦略

学習アルゴリズムが生成した仮説中には, 変数を含まないパターンが含まれることがしばしばある. このような定数文字列は, 具体例として自分自身しか含まないので, 正例を覆う意味では役にたたないように思われる. しかし, これらの定数文字列は, 誤差などに由来する例外的な文字列を正例集合から排除することで, 仮説自体の精度を上げていると考えられる. したがって、例外を優先して探し, これを正例から排除することで, 高い精度の仮説を発見することができると考えられる.

学習アルゴリズムは, 仮説空間のパラメータ k, m に加えて, 許される例外の最大数 e を受けとり, 仮説の探索を開始する. 例外パターンが見つかると、学習アルゴリズムはそれを仮説中に含めず、同時に、その文字列を正例から取り除く。ただし、正例から取り除く例外数は、高々 e 個までとする。

	patterns
k	**2222222
	*122*2221*21212
	*122*2221*22
	*222*1222*
	12112212212112222220010222
	2210220221112021112
	22112111202121222
e	22222222021212211221
	222222222

表 1: 例外パターンを許した場合に, 学習アルゴリズムによって生成された仮説の例. 使用した正例数は 50 で, 仮説空間のパラメータは $k = 7, m = 3$ とし, 仮説中に許される例外数の最大値を $e = 2$ とした. 仮説の精度は 84.0% である.

前節の学習アルゴリズムによる仮説空間の探索においては、以下に挙げる 4 種類の選択点が存在する.

(選択点 1) どのパターンを分割するか.

(選択点 2) どのようにパターンを分割するか.

(選択点 3) どのパターンをさらに具体化するか.

(選択点 4) どのようにパターンを分割するか.

学習アルゴリズムにおいて, これらの選択点でどのように選択するかは, あらかじめ与えられた戦略にしたがって決定される. どのような探索戦略を採用しても, 学習アルゴリズムは正しく k-mmg を計算する. しかし, 計算の効率や発見される仮説の精度は, 使用する探索戦略に大きく依存するので, 文献 [3] では, 以下のような探索戦略を採用した.

- 確率的戦略: 全ての選択点において, 無作為に選択する.
- 極大被覆戦略: 仮説中のパターンが含む正例の数が, 均等になるように仮説を選択する. 具体的には, (選択点 1) と (選択点 3) では, できるだけ多くの正例を被覆するパターンを選択し, 他の選択点では無作為に仮説を選択する.

しかし, これらの戦略, 特に極大被覆戦略では, できるだけ例外を生成しないように仮説を選択することになる. これは, われわれの例外を活用する手法においては, 好ましくない. そこで, 例外パターンを積極的に生成するために次のような戦略を用いる.

- 極小被覆戦略: できるだけ例外パターンを生成するように, 仮説を選択する. 具体的には, (選択点 1) と (選択点 3) では, できるだけ少ない正例を被覆するパターンを選択し, 他の選択点では無作為に仮説を選択する.

6 実験結果

集合 Pos と Neg を, この学習システムの入力として使われる正例と負例の集合とする. 仮説中が含むパターン数の上限 k とパターン中の変数の数の上限 m が入力パラメータとして指定されると, 学習アルゴリズムは Pos と Neg から小さな部分集合を無作為に選択し, この部分集合に k-mmg アルゴリズムを適用し, 仮説として拡張正則パターンの集合 H を作り出す. 仮説の Pos と Neg に対する精度は, $p(n)$ を $Pos(Neg)$ に含まれる例のうち正しく正 (負) であると認識された割合とし, 組 (p, n) で表す. 学習システムは, pos を取り変えながら以上の手順を繰り返し, 最も精度の高いものを最終的

な仮説として出力する．今回は，使用する正例の数を 20 ～ 100 とし，仮説を制限するパラメータを $k = 3\sim7, m = 3, e = 2$ として実験を行なった．

実験では次のデータを用いた．

- 膜タンパク質の膜貫通領域: PIR データベース [8] から，取って来た膜貫通領域に対応するアミノ酸配列 689 個と，それ以外の部分からとった長さ 30 のアミノ酸配列 19256 個を取ってきて，それぞれ正例と負例とした．

また，アミノ酸配列をアミノ酸の親水性指標 [6] にしたがってデータを変換したデータを用いて実験を行なった．変換を用いた理由は，仮説空間を大幅に縮小することができ，しかも，膜貫通領域はアミノ酸の親水性と密接なつながりがあることかが知られているので，より高い精度の仮説の生成に有効であるからである．

6.1 極小被覆戦略と他の戦略の比較

はじめに，極小被覆戦略と他の 2 つの戦略の間で生成された仮説の精度を比較する．仮説生成において例外パターンを許した場合 ($k = 7, e = 2$) を調べる。このとき、図 2 から，仮説の精度の最大値に関しては，極小被覆戦略での値は他の戦略のものと変わらないが，精度の平均値に関しては，極小被覆戦略での値の方が高くなっていることがわかる。例えば、正例数 50 の場合に生成された仮説の平均値は，3% 程度高くなっている．また，例外を許さない場合についても同様の結果が得られた。

6.2 本研究で導入した手法の有効性

次に、本研究で提案した手法に基づいた学習アルゴリズムと文献 [3] の学習アルゴリズムを総合的に比較する．具体的には，例外を許した極小被覆戦略と、対照実験として例外を許さない確率的戦略について実験した。例外を許すことによる最大パターン数の実質的増大を排除するため，仮説空間について，極小被覆戦略ではパラメータを $k = 5, e = 2$ とし，確率的戦略ではパラメータを $k = 7, e = 0$ として，和 $k + e$ が同じ値になるように調整した．

図 3から，最大値に関しては，例外を許した極小被覆戦略によって生成された仮説の精度と，例外を許さない確率的戦略によるものとの間には違いは観察されなかった．しかし，平均値に関しては，例外を許した極小被覆戦略によるものの方が，正例数の全域にわたって精度が高かった．計算時間に関しては，例外を許した極小被覆戦略では 3.8～5.9 秒であり，例外を許さない確率的戦略では 5.3～6.1 秒であった．このように，例外を許した極小被覆戦略の方が，精度の平均と計算時間の両方でわずかではあるがより優位であった．

7 結論

本稿では，k-mmg アルゴリズムに，例外を活用する手法を導入し，その有効性をタンパク質モチーフ発見の実験で検証した．実験の結果，この手法は仮説空間を実質的に拡大し，仮説の精度を向上させる一方で，計算時間を減少させるのに効果があった．これらの結果より，正例からの機械発見におけるの探索戦略として，例外パターンを許す手法と極小被覆戦略の組合せには一定の効果があることが示された。例外の数 e をおさえるための関数として，今回使用したような定数関数だけでなく，正例数 n に対してゆっくりと増加する $\log n$ や $\sqrt{n}$ のような関数を用いることで、より多量のデータからの知識獲得が可能になると思われる。この拡張が今後の課題である。

謝辞

本研究は，一部文部省科学研究費重点領域研究「ゲノム情報」の補助を受けた．

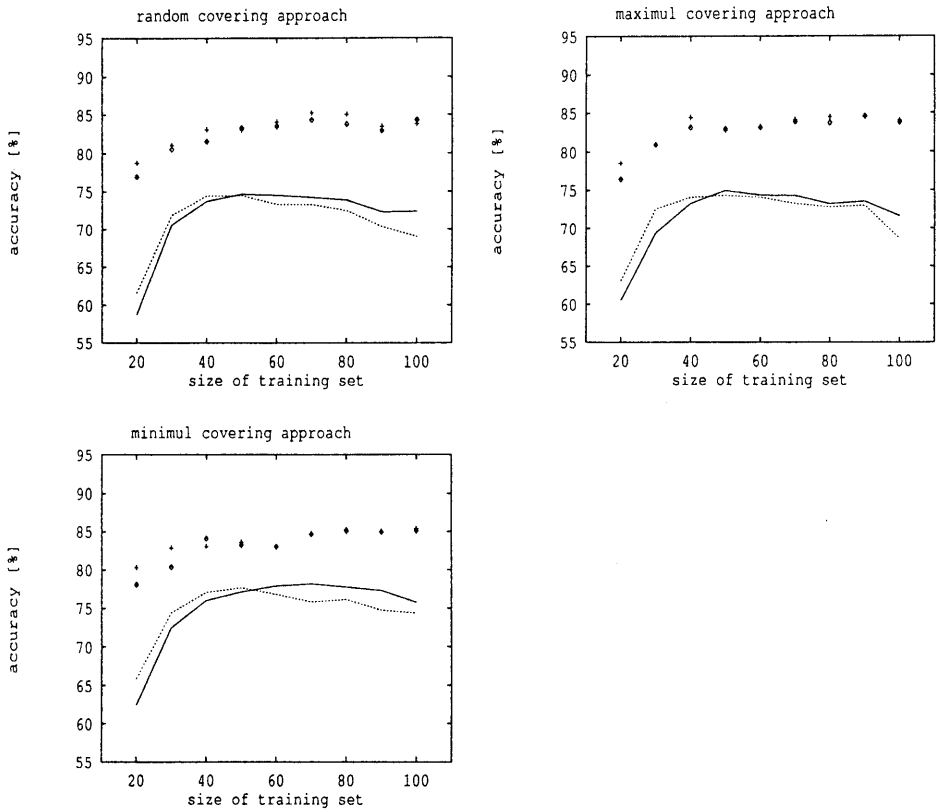

図 2: 例外を許した場合とそうでない場合についての仮説の精度の比較. 左から, 確率的戦略および, 極大被覆戦略, 極小被覆戦略についての結果を示す. 例外を許した場合とそうでない場合のそれぞれについて, 実線 — と点線 ··· は平均値を表し、 ◇ と + は生成された仮説の最大値を表す.

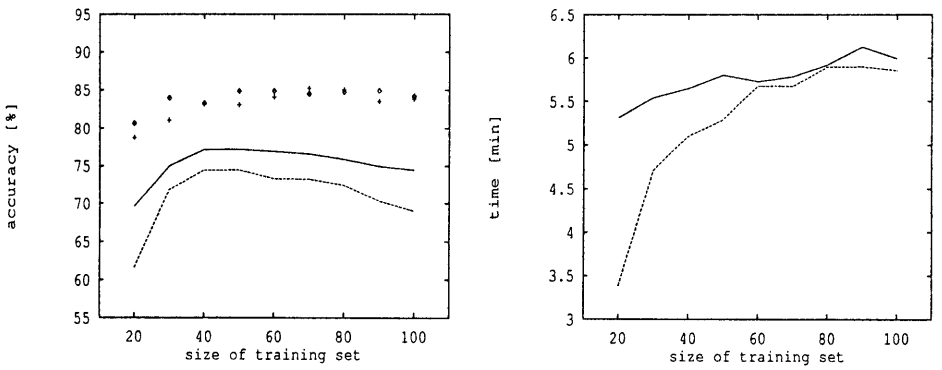

図 3: 例外を許した場合とそうでない場合の仮説の精度と, 計算時間を比較する. 左側のグラフは生成された仮説の精度を示し, 右側のグラフは生成に要した時間を表す. 例外を許した場合とそうでない場合のそれぞれについて, 実線 — と点線 ··· は平均値を表し、 ◇ と + は生成された仮説の最大値を表す.

参考文献

[1] S. Arikawa, S. Kuhara, S. Miyano, A. Shinohara, and T. Shinohara. A learning algorithm for elementary formal systems and its experiments on identification of transmembrane domains. In *Proc. the 25th Hawaii International Conference on System Sciences*, pp. 675–684, 1992.

[2] S. Arikawa, S. Miyano, A. Shinohara, S. Kuhara, Y. Mukouchi, and T. Shinohara. A machine discovery from amino acid sequences by decision trees over regular patterns.

New Generation Computing, 11, pp. 361–375, 1993.

[3] H. Arimura, R. Fujino, T. Shinohara, and S. Arikawa. Protein Motif Discovery from Positive Examples by Minimal Multiple Generalization over Regular Patterns. In *Genome Informatics Workshop*, pp. 39–48, 1994.

[4] H. Arimura, T. Shinohara, S. Otsuki. Finding minimal generalizations for unions of pattern languages and its application to inductive inference from positive data. In *Proc. the 11th STACS*, LNCS 775, Springer-Verlag, pp. 649–660, 1994.

[5] GenBank. GenBank Release Notes. IntelliGenetics Inc., 1991.

[6] J. Kyte, R.F. Doolittle. In *J. Mol. Biol.*, 157, pp. 105-132, 1982.

[7] R. D. King, A. Srinivasan, S. Muggleton, C. Feng, R. A. Lewis, and M. J. E. Sternberg. Drug design using inductive logic programming. In *Proc. the 26th Hawaii International Conference on System Sciences*, pp. 646–655, 1993.

[8] PIR. Protein identification resource. National Biomedical Research Foundation, 1991.

[9] T. Shinohara. Polynomial time inference of extended regular pattern languages. In *RIMS Symposia on Software Science and Engineering*, LNCS 147, pp. 115–127, Springer-Verlag, 1982.