A Machine Discovery from Amino Acid Sequences by Decision Trees over Regular Patterns

Arikawa, Setsuo
Research Institute of Fundamental Information Science Kyushu University

Kuhara, Satoru
Graduate School of Genetic Resources Technology, Kyushu University

Miyano, Satoru
Research Institute of Fundamental Information Science Kyushu University

Mukouchi, Yasuhito

他

出版情報：RIFIS Technical Report. 44, 1991-08-01. Research Institute of Fundamental Information Science, Kyushu University
バージョン：
権利関係：

RIFIS-TR-CS-44

RIFIS Technical Report

A Machine Discovery from Amino Acid Sequences by Decision Trees over Regular Patterns

Setsuo Arikawa
Satoru Kuhara
Satoru Miyano
Yasuhito Mukouchi
Ayumi Shinohara
Takeshi Shinohara

August 1, 1991
Revised: September 25, 1991

Research Institute of Fundamental Information Science
Kyushu University 33
Fukuoka 812, Japan

E-mail: miyano@rifis.sci.kyushu-u.ac.jp Phone: 092 (641)1101 Ex. 4471

A Machine Discovery from Amino Acid Sequences by Decision Trees over Regular Patterns[*]

Setsuo Arikawa[†]
arikawa@rifis.sci.kyushu-u.ac.jp

Satoru Kuhara[‡]
kuhara@kintaro.grt.kyushu-u.ac.jp

Satoru Miyano[†]
miyano@rifis.sci.kyushu-u.ac.jp

Yasuhito Mukouchi[††]
mukouchi@rifis.sci.kyushu-u.ac.jp

Ayumi Shinohara[†]
ayumi@rifis.sci.kyushu-u.ac.jp

Takeshi Shinohara[‡‡]
shino@donald.ai.kyutech.ac.jp

[†] Research Institute of Fundamental Information Science, Kyushu University 33
Fukuoka 812, Japan.

[‡] Graduate School of Genetic Resources Technology, Kyushu University 46
Fukuoka 812, Japan.

[††] Department of Information Systems, Kyushu University 39
Kasuga 816, Japan.

[‡‡] Department of Artificial Intelligence, Kyushu Institute of Technology
Iizuka 820, Japan.

Abstract: This paper describes a machine learning system that discovered a "negative motif", in transmembrane domain identification from amino acid sequences, and reports its experiments on protein data using PIR database. We introduce a decision tree whose node are labeled with regular patterns. As a hypothesis, the system produces such decision tree for a small number of randomly chosen positive and negative examples from PIR. Experiments show that our system finds reasonable hypotheses very successfully. As a theoretical foundation, we show that the class of languages defined by decision trees of depth at most d over k-variable regular patterns is polynomial time learnable in the sense of probably approximately correct (PAC) learning for any fixed $d,\ k \geq 0$.

1 Introduction

Hydrophobic transmembrane domains can be identified with 90% accuracy for all data in PIR database by two consecutive polar amino acids (Arg, Lys, His, Asp, Glu, Gln, Asn) that are not included in the transmembrane domains. This result was discovered by the machine learning system that we developed using a decision procedure called a decision tree over regular

[*] The work is partly supported by Grant-in-Aid for Scientific Research on Priority Areas, "Genome Informatics" from the Ministry of Education, Science and Culture, Japan.

patterns. On each trial, the system randomly chooses, from PIR database [11], a small number of training sequences; transmembrane domain sequences and sequences cut out from the parts other than transmembrane domains. The system has found very simple decision trees over regular patterns which indicate that significant motifs are not inside but outside the sequences of the transmembrane domains. We call such motifs "negative motifs".

This paper describes a machine learning system that discovered negative motifs and reports its experiments on knowledge acquisition from amino acid sequences that reveal the importance of negative data. Former researches for finding motifs have focused only on positive examples and ignored mostly negative examples. The approach by decision tree over regular patterns provides new direction and method for discovering motifs.

A regular pattern [15, 16] is an expression $w_0x_1w_1x_2\cdots x_nw_n$ that defines the sequences containing $w_0, w_1, ..., w_n$ in this order, where each w_i is a sequence of symbols and x_j varies over arbitrary sequences. Regular patterns have been used to describe some features of amino acid sequences and DNA sequences [1, 5]. A decision tree over regular patterns is a tree which describes a decision procedure for determining the class of a given sequence. Each node is labeled with either a class name (1 or 0) or a regular pattern. At a node with a regular pattern, the decision tree tests if the sequence matches the pattern or not. Starting from the root toward a leaf, the decision procedure makes a test at each node and goes down by choosing the left or right branch according to the result of the test. The reached leaf answers the class name of the sequence.

We employ the idea of ID3 algorithm [12] for constructing a decision tree since it is sufficiently fast and experiments show that small enough trees are usually obtained. We also devise a new method for constructing a decision tree over regular patterns using a score function different from that in [12]. Given two sets of positive and negative examples, our machine learning system finds appropriate regular patterns as node attributes during the construction of the decision tree. Hence, unlike ID3, we need not struggle for defining the attributes of a decision tree beforehand. Our system makes a decision tree just from a small number of training sequences, which we also guarantee in the PAC learning theory [18] in Section 5. Therefore it may cope with a diversity of classification problems for proteins and DNA sequences.

A hydropathy plot [4, 7, 14] has been used generally to predict transmembrane domains from primary sequences. With this knowledge, we first transform twenty amino acids to three categories ($*$, $+$, $-$) according to the hydropathy index of Kyte and Doolittle [7]. From randomly chosen 10 positive and 10 negative training examples, our system has successfully produced some small size decision trees over regular patterns which are shown to achieve very high accuracy. The regular patterns appearing in these decision trees indicates that two consecutive polar amino acid residues are important negative motifs for transmembrane domains.

We have also made an experiment on raw sequences without transformation. Our system discovered a small size decision tree just from 20 raw sequences with more than 85% accuracy that show if a sequence contains neither `E` nor `D` (both are polar amino acids) then it is very likely to be a transmembrane domain. From the view point of Artificial Intelligence, it is quite interesting that these residues were found by our machine learning system without any knowledge on the hydropathy index.

A well-known structure around the membrane integrated domain is the signal-anchor structure that consists of two parts, the hydrophobic part of a membrane-spanning sequence and the charged residues around the hydrophobic part [8, 19]. The negative motif of consecutive polar amino acid residues may be closely related to the signal-anchor structure. After knowing the importance of negative motifs, we have found a pattern x_1-x_2-x_3-x_4-x_5-x_6 that gives the sequences containing at least five polar amino acids. The result on the pattern x_1-x_2-x_3-x_4-x_5-x_6 shows that the accuracy is 95.4% for positive and 95.1% for negative examples although it has been believed to be difficult to define transmembrane domains as a simple expression when the view point was focussed on positive examples.

2 Decision Trees over Regular Patterns

Let Σ be a finite alphabet and $X = \{x, y, z, x_1, x_2, \cdots\}$ be a set of variables. We assume that Σ and X are disjoint. A *pattern* is an element of $(\Sigma \cup X)^+$, the set of all nonempty words over $\Sigma \cup X$. For a pattern π, the language $L(\pi)$ is the set of words obtained by substituting each variable in π for a word in Σ^*. We say that a pattern π is *regular* if each variable occurs at most once in π. For example, $xaybza$ is a regular pattern, but xx is not. Obviously, regular patterns define regular languages, but not vice versa. In this paper we consider only regular patterns. A regular pattern containing at most k variables is called a *k-variable regular pattern.*

A *decision tree over regular patterns* is a binary tree such that the leaves are labeled with 0 or 1 and each internal node is labeled with a regular pattern (see Figure 1). For an internal node v, we denote the left and right children of v by left(v) and right(v), respectively. We denote by $\pi(v)$ the regular pattern assigned to the internal node v. For a leaf u, value(u) denotes the value 0 or 1 assigned to u. The depth of a tree T, denoted by depth(T), is the length of the longest path from the root to a leaf.

For a decision tree T over regular patterns, we define a function $f_T : \Sigma^* \rightarrow \{0, 1\}$ as follows. For a word w in Σ^*, we determine a path from the root to a leaf and define the value $f_T(w)$ by the following algorithm:

```
begin /* Input: $w \in \Sigma^*$ */
    $v \leftarrow$ root;
    while $v$ is not a leaf do
        if $w \in L(\pi(v))$ then $v \leftarrow$right($v$) else $v \leftarrow$left($v$);
    $f_T(w) \leftarrow$ value($v$)
end
```

For a decision tree T over regular patterns, we define $L(T) = \{w \in \Sigma^* \mid f_T(w) = 1\}$. It is easy to see that $L(T)$ is also a regular language. But the converse is not true. Let $L = \{a^{2n} \mid n \geq 1\}$. It is straightforward to show that there is no decision tree T over regular patterns with $L = L(T)$. The same holds for the language $\{a^{2n}b \mid n \geq 1\}$.

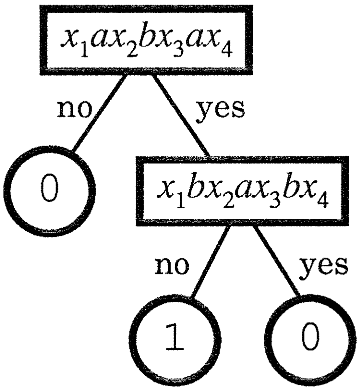


Figure 1: Decision tree over regular patterns defining a language $\{a^m b^n a^l \mid m, n, l \geq 1\}$ over $\Sigma = \{a, b\}$

3 Constructing Decision Trees

This section gives two kinds of algorithms for constructing decision trees over regular patterns that are used in our machine learning system.

The first algorithm employs the idea of Quinlan's ID3 algorithm [12] for constructing decision trees. The ID3 algorithm assumes the attributes of a decision tree in advance. Therefore, we have to determine which regular patterns can be used for attributes of a decision tree. Our algorithm finds appropriate regular patterns for the attributes dynamically during the construction of the decision tree. This is the point where our algorithm differs from ID3. The following recursive algorithm DT1(P, N) sketches our decision tree algorithm:

```
function DT1 ( $P$, $N$ : sets of strings ): node;
  begin
    if $N = \emptyset$ then
      return( CREATE("1", null, null) )
    else if $P = \emptyset$ then
      return( CREATE("0", null, null) )
    else begin
      let $\pi$ be a shortest regular pattern such that $E(\pi, P, N)$ is minimum;
      $P_1 \leftarrow P \cap L(\pi)$;    $P_0 \leftarrow P - P_1$;
      $N_1 \leftarrow N \cap L(\pi)$;    $N_0 \leftarrow N - N_1$;
      return( CREATE($\pi$,DT($P_0, N_0$),DT($P_1, N_1$)) )
      end
  end
```

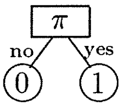
Algorithm 1

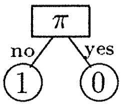
CREATE(π,T_0,T_1) returns a new tree with a root labeled with π whose left and right subtrees are T_0 and T_1, respectively. The cost $E(\pi, P, N)$ is the one defined in [12] by

$$E(\pi, P, N) = \begin{cases} 0 & \text{if } x = 0 \text{ or } y = 0 \\ \frac{p_1+n_1}{|P|+|N|} I(p_1, n_1) + \frac{p_0+n_0}{|P|+|N|} I(p_0, n_0) & \text{otherwise,} \end{cases}$$

where $p_1 = |P \cap L(\pi)|$, $n_1 = |N \cap L(\pi)|$, $p_0 = |P \cap \overline{L(\pi)}|$, $n_0 = |N \cap \overline{L(\pi)}|$, $\overline{L(\pi)} = \Sigma^* - L(\pi)$ and

$$I(x, y) = -\frac{x}{x+y} \log \frac{x}{x+y} - \frac{y}{x+y} \log \frac{y}{x+y}.$$

Now we introduce the second algorithm for constructing decision trees. Let nodes(T) be the number of nodes in T, and $\mathcal{T}(T)$ be the set of trees constructed by replacing a leaf v of T by a subtree 

or 

for some pattern π.

The score function $Score(T, P, N)$ is defined by

$$Score(T, P, N) = \frac{|P \cap L(T)|}{|P|} \cdot \frac{|N \cap \overline{L(T)}|}{|N|}.$$

Then the second algorithm is sketched as follows:

```
function DT2( $P, N$: sets of strings, $MaxNode$: int ) : tree;
    begin
        if $N = \emptyset$ then
            return( CREATE("1", null, null) )
        else if $P = \emptyset$ then
            return( CREATE("0", null, null) )
        else begin
            $T \leftarrow$CREATE("1", null, null);
            while ( nodes($T$) $<$ $MaxNode$ and $Score(T, P, N) < 1$ ) do
                begin
                    find $T_{max} \in \mathcal{T}(T)$ that maximizes $Score(T_{max}, P, N)$;
                    $T \leftarrow T_{max}$
                end
        end
        return ( $T$ )
    end
```

Algorithm 2

Algorithm 2 is slower than Algorithm 1 since all leaves are checked at each phase of a node generation. However, Algorithm 2 constructs decision trees which are finely tuned when the size of decision trees is large. Moreover, it is noise-tolerant, i.e., it allows conflicts between positive and negative training examples.

4 Transmembrane Domain Identification

The problem of transmembrane domain identification is one of the most important protein classification problems and some methods and experiments have been reported. For example, Hartman et al. [6] proposed a method using the hydropathy index for amino acid residues in [7]. The reported success rate is about 75%. Most approaches deal with positive examples, i.e., sequences corresponding to transmembrane domains, and try to find properties common to them.

The sequence in Figure 2 is an amino acid sequence of a membrane protein. There is a tendency to assume that a membrane protein contains several transmembrane domains each of which consists of $20 \sim 30$ amino acid residues. Therefore, if a sequence corresponding to a transmembrane domain is found in an amino acid sequence, it is very likely that the protein is a membrane protein.

```
MDVVNQLVAGGQFRVVKE(PLGFVKVLQWVFAIFAFATCGSY)TGELRLSVECANKTESALNIEVEF
EYPFRLHQVYFDAPSCVKGGTTKIFLVGDYSSSAE(FFVTVAVFAFLYSMGALATYIFL)QNKYREN
NK(GPMMDFLATAVFAFMWLVSSSAWA)KGLSDVKMATDPENIIKEMPMCRQTGNTCKELRDPVTS(
GLNTSVVFGFLNLVLWVGNLWFVF)KETGWAAPFMRAPPGAPEKQPAPGDAYGDAGYGQGPGGYGPQ
DSYGPQGGYQPDYGQPASGGGGYGPQGDYGQQGYGQQGAPTSFSNQM
```

Figure 2: An amino acid sequence which contains four transmembrane domains shown by the parenthesized parts.

Our idea for transmembrane domain identification is to use decision trees over regular patterns for classification. Algorithm 1 and 2 introduced in Section 3 are used to find good decision trees from positive and negative training examples.

A *positive example* is a sequence which is already known to be a transmembrane domain. A *negative example* is a sequence of length around 30 cut out from the parts other than transmembrane domains. From PIR data base [11], our machine learning system chooses randomly a small amount of positive and negative training examples. Then, by using Algorithm 1 or Algorithm 2, the system constructs a decision tree over regular patterns at each trial and produces decision trees with good accuracy.

We have evaluated the performance ratio of a decision tree in the following way. As the total space of positive examples, we use the set POS of all transmembrane domain sequence (689 sequences) from PIR database. The total space NEG of negative examples consists of 19256 negative examples randomly chosen from *all* proteins from PIR. The success rate of a decision tree for positive examples is the percentage of the positive examples from POS recognized as positive (class 1). The success rate for negative examples is counted as the percentage of the negative examples from NEG recognized as negative (class 0).

In order to avoid combinatorial explosion, we restrict regular patterns to the regular patterns of the form $x\alpha y$. In this form, x and y are variables and α is a subword taken from given examples. Given a set P of positive training examples and a set N of negative training examples, we would like to construct a small decision tree over regular patterns which classifies P and N exactly.

The alphabet of amino acid sequences consists of twenty symbols. It has been shown that the use of the hydropathy index for amino acids is very successful [1, 6]. According to the hydropathy index of [7], we transform these twenty symbols to three symbols as shown in Table 1. This transformation reduces the size of a search space drastically small.

Then by this transformation table, the sequence in Figure 2 becomes the following sequence (Figure 3):

Our system can, of course, cope with both raw sequences from twenty symbols and indexed

Amino Acids	Hydropathy Index		New Symbol
A M C F L V I	1.8 ∼ 4.5	→	*
P Y W S T G	−1.6 ∼ −0.4	→	+
R K D E N Q H	−4.5 ∼ −3.2	→	-

Table 1: Transformation rules

```
*-**--***++-*-**--(+*+**-**-+********+*+++)++-*-*+*-**--+-+**-*-*-*
-++*-*--*+*-*++**-++++-****+-++++*-(***+*******++*+***++***)---+---
--(++**-***+******+**+++*+*)-+*+-*-**+-+--**--*+**--++-+*--*--+*++(
+*-++***+**-***+*+-*+***)--+++**+**-*+++*+---+*++-*++-*+++-+++++++-
-++++-+++-+-++-+*++++++++-+-++--+++--+*+++*+--*
```

Figure 3: The sequence obtained by the transformation

sequences from three symbols. We made experiments on both raw and indexed sequences.

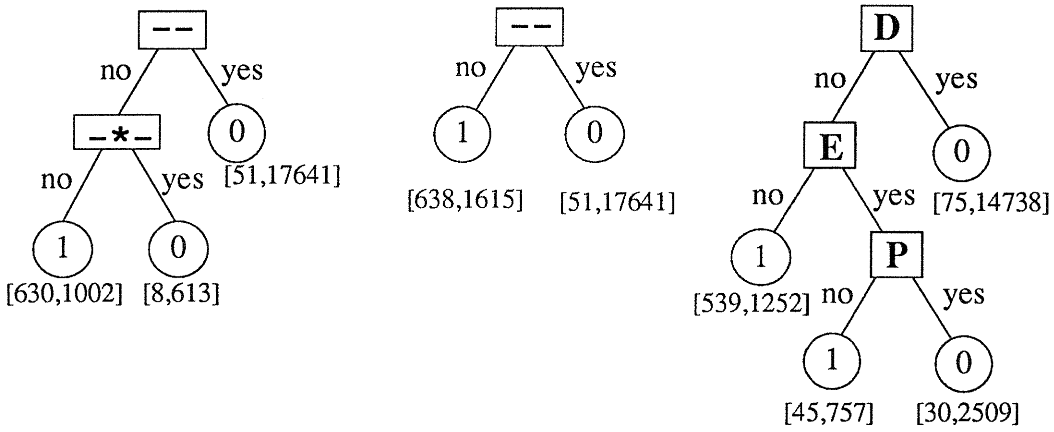
Figure 4 (a), (b) show two of the best decision trees over regular patterns that our machine learning system found from 10 positive and 10 negative training examples. The decision tree (a) recognizes 91.4% of positive examples and 94.8% of negative examples. Even the decision tree of (b) can recognize 92.6% of the positive examples and 91.6% of the negative examples. The negative motif “--” which indicates consecutive polar amino acid residues plays a key role in classification. This may have a close relation to the signal-anchor structure [8, 19] as mentioned in Introduction.

Figure 4 (c) is one of the smallest decision trees discovered by our system just from 10 positive and 10 negative raw sequences that achieve good accuracy. The decision tree (c) also shows the importance of a cluster of polar amino acids in transmembrane domain identification although our machine learning system has no knowledge about the hydropathy.

We examined how the performance of our machine learning system changes with respect to the number of training examples. The training examples are chosen randomly ten times in each case and a point of the graph of Figure 5 is the average of these ten results for each case. Figure 5 shows the results. We may observe the following facts:

1. The hydropathy index of Kyte and Doolittle [7] is very useful. When indexed sequences are used, the system can produce from 40 positive and 40 negative examples a decision tree with only several nodes whose accuracy is more than 90% for the total space in average. On the other hand, for raw sequences the accuracy is not so good but both accuracies approach to the same line as the number of training examples increases.

2. The number of nodes of a decision tree is reasonably small. But when the number of



(a) (91.4%, 94.8%) (b) (92.6%, 91.6%) (c) (84.8%, 89.6%)

Figure 4: The node label, for example, -- is an abbreviation of x_1--x_2 that tests if a given sequence contains the sequence --. The leaf label 1 (resp. 0) is the class name of transmembrane domains (resp. non-transmembrane domains). The total space consists of 689 positive examples and 19256 negative examples. Each of the decision trees (a)-(c) is constructed from 10 positive and 10 negative training sequences. The pair $[p, n]$ attached to a leaf shows the number p of positive examples and the number n of negative examples that have reached to the leaf. The pair $(p\%, n\%)$ means that $p\%$ of 689 positive (resp. $n\%$ of 19256 negative) examples are recognized as transmembrane domains (resp. non-transmembrane domains).

training examples is larger, the number of nodes in a decision tree becomes larger while the accuracy is not improved very much. There may arise the problem of overfitting.

A new discovery obtained from these decision trees is that the motif "--" drastically rejects positive examples. After knowing the negative motif "--", we have examined the decision trees with a single node with the patterns of the form

$$x_1\text{-}x_2\text{-}\cdots\text{-}x_n$$

for $n \geq 3$. The best is the pattern containing "-" five times. The result is quite acceptable as shown in Table 2.

Pattern	*POS* (689)	*NEG* (19256)
x_1-x_2-x_3-x_4-x_5-x_6	657 (95.4 %)	18304 (95.1%)

Table 2: Result for x_1-x_2-x_3-x_4-x_5-x_6

With these decision trees over regular patterns, we have developed a transmembrane domain predictor that reads an amino acid sequence of a protein as an input and predicts symbol by symbol whether each location of a symbol is in a transmembrane domain or not. Experiments on all protein sequences in PIR show that the success rate is 85% $\sim$ 90%.

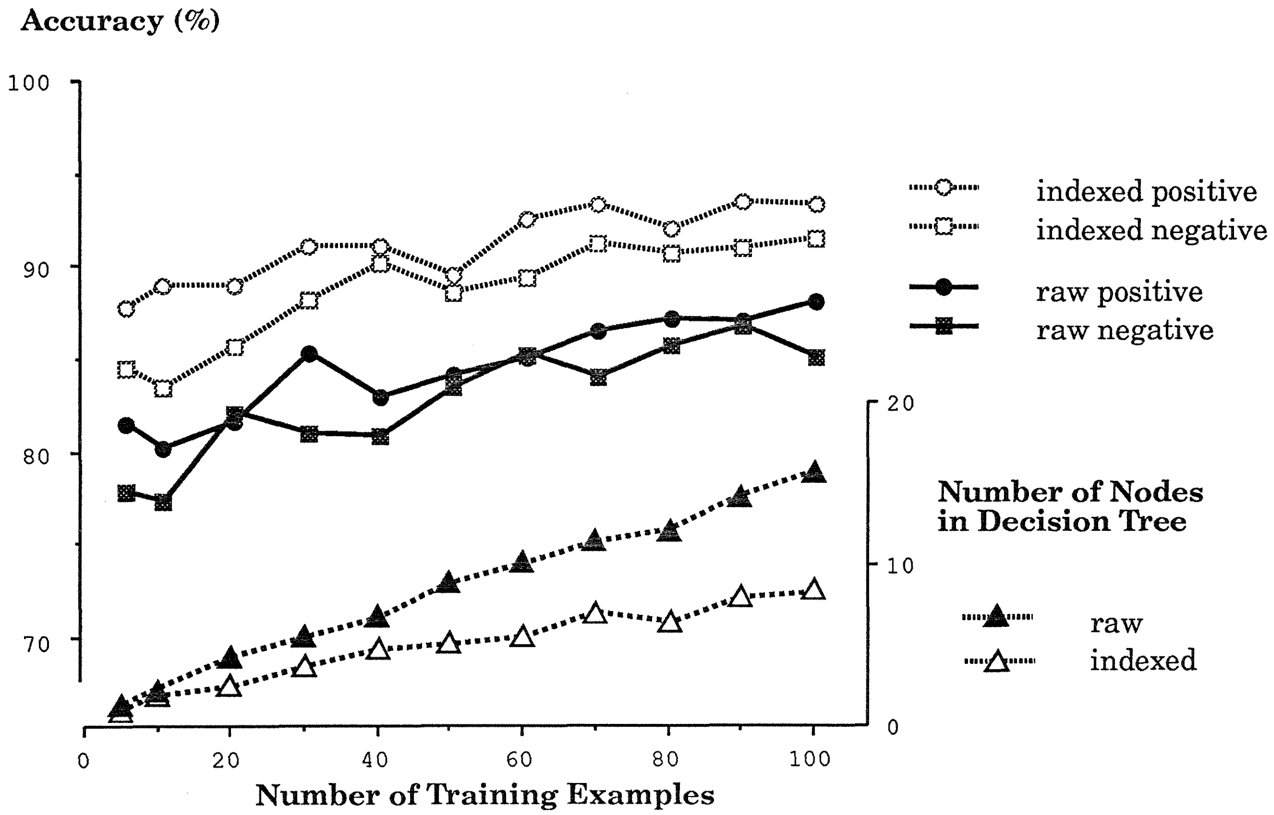


Figure 5: Relations between the number of training examples, accuracy and the number of nodes in a decision tree

5 PAC-Learnable Class

Quinlan's ID3 algorithm is not guaranteed to construct a minimum depth or small size decision tree and it is not known whether it approximates the optimal decision tree with some error ratio. However, it is sufficiently fast and it seems that a small enough decision tree is usually obtained and we have no problem in practical use. The ID3 algorithm has received considerable attentions [13, 17].

This section provides a theoretical foundation from the point of algorithmic learning by showing the following theorem though it is apart from the ID3 algorithm. The proof of the theorem shall be given in the full paper.

For integers $k, d \geq 0$, we consider a decision tree T over k-variable regular patterns whose depth is at most d. We denote by $DTRP(d, k)$ the class of languages defined by decision trees over k-variable regular patterns with depth at most d.

Theorem 1 $DTRP(d, k)$ is polynomial-time learnable for all $d, k \geq 0$.

We need some terminology for the above theorem. When we are concerned with learning, we call a subset of Σ^* a *concept*. A *concept class* $\mathcal{C}$ is a nonempty collection of concepts. For a concept $c \in \mathcal{C}$, a pair $\langle x, c(x) \rangle$ is called an example of c for $x \in \Sigma^*$,where $c(x) = 1$

$(c(x) = 0)$ if x is in c (is not in c). For an alphabet Σ and an integer $n \geq 0$, $\Sigma^{\leq n}$ denotes the set $\{x \in \Sigma^* \mid |x| \leq n\}$.

A concept class $\mathcal{C}$ is said to be *polynomial-time learnable* [2, 10, 18] if there is an algorithm $\mathcal{A}$ which satisfies (1) and (2).

(1) $\mathcal{A}$ takes a sequence of examples as an input and runs in polynomial-time with respect to the length of input.

(2) There exists a polynomial $p(\cdot, \cdot, \cdot)$ such that for any integer $n \geq 0$, any concept $c \in \mathcal{C}$, any real number ε, δ $(0 < \varepsilon, \delta < 1)$, and any probability distribution P on $\Sigma^{\leq n}$, if $\mathcal{A}$ takes $p(n, \frac{1}{\varepsilon}, \frac{1}{\delta})$ examples which are generated randomly according to P, then $\mathcal{A}$ outputs, with probability at least $1 - \delta$, a representation of a hypothesis h with $P(c \oplus h) < \varepsilon$.

Theorem 2 [2, 10] A concept class $\mathcal{C}$ is polynomial-time learnable if and only if the following conditions hold.

(1) $\mathcal{C}$ is of *polynomial dimension*, i.e., there is a polynomial $d(n)$ such that $|\{c \cap \Sigma^{\leq n} \mid c \in \mathcal{C}\}| \leq 2^{d(n)}$ for all $n \geq 0$.

(2) There is a *randomized polynomial-time hypothesis finder* for $\mathcal{C}$ that is an randomized polynomial-time algorithm which produces from a sequence of examples, with probability at least γ for some $\gamma > 0$, a hypothesis which is consistent with the given examples.

Ehrenfeucht and Haussler [3] have considered learning of decision trees of a fixed rank. For learning decision trees over regular patterns, the restriction by rank can be shown to have no sense. Instead, we consider the depth of a decision tree. It is also reasonable to put a restriction on regular patterns. It has been shown that the class of regular pattern languages is not polynomial-time learnable unless $\mathbf{NP} \neq \mathbf{RP}$ [9]. Therefore, unless restrictions such as bound on the number of variables in a regular pattern are given, we may not expect any positive results for polynomial-time learning. By using the equivalence in Theorem 2, we can prove Theorem 1.

Given positive and negative examples, the algorithm in the proof of Theorem 1 finds a minimum depth decision tree which classifies the given data. It runs in polynomial time with respect to the length of input. But it exhausts an enormous amount of time and is not suited for practical use.

6 Conclusion

We have shown that the idea of combining regular patterns and decision trees works quite well for transmembrane domain identification. The experiments also have shown the importance of negative motifs.

A union of regular patterns is regarded as a special form of a decision tree called a decision list. We have reported in [1] that the union of small number of regular patterns can also recognize transmembrane domains with high accuracy. However, the time exhausted in finding hypotheses in [1] is much larger than that reported in this paper.

Our system constructs a decision tree over regular patterns just from strings called positive and negative examples. We need not take care of which attributes to specify as in ID3. Therefore it can be applied to another classification problems for proteins and DNA sequences. We believe that our approach provides a new application of algorithmic learning to Molecular Biology.

We are now in the process of examining our method for predicting the secondary structure of proteins.

References

[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", Technical Report RIFIS-TR-CS-40, Research Institute of Fundamental Information Science, Kyushu University, 1991 (to appear in Proc. 25th Hawaii International Conference on System Sciences).

[2] A. Blumer, A. Ehrenfeucht, D. Haussler and M.K. Warmuth, "Learnability and the Vapnik-Chervonenkis dimension", *JACM*, Vol. 36, pp. 929–965, 1989.

[3] A. Ehrenfeucht and D. Haussler, "Learning decision trees from random examples", *Inform. Comput.*, Vol. 82, pp. 231–246, 1989.

[4] D.M. Engelman, T.A. Steiz and A. Goldman, "Identifying nonpolar transbilayer helices in amino acid sequences of membrane proteins", *Ann. Rev. Biophys. Biophys. Chem.*, Vol. 15, pp. 321–353, 1986.

[5] V. Gusev, N. Chuzhanova, "The algorithms for recognition of the functional sites in genetic texts", *Proc. 1st Workshop on Algorithmic Learning Theory*, pp. 109–119, 1990.

[6] E. Hartmann, T.A. Rapoport and H.F. Lodish, "Predicting the orientation of eukaryotic membrane-spanning proteins", *Proc. Natl. Acad. Sci. U.S.A.*, Vol. 86, pp. 5786–5790, 1989.

[7] J. Kyte and R.F. Doolittle, "A simple method for displaying the hydropathic character of protein", *J. Mol. Biol.*, Vol. 157, pp. 105–132, 1982.

[8] J. Lipp, N. Flint, M.T. Haeuptle and B. Dobberstein, "Structural requirements for membrane assembly of proteins spanning the membrane several times", *J. Cell Biol.*, Vol. 109, pp. 2013–2022, 1989.

[9] S. Miyano, A. Shinohara and T. Shinohara, "Which classes of elementary formal systems are polynomial-time learnable?", Technical Report RIFIS-TR-CS-37, Research Institute of Fundamental Information Science, Kyushu University, 1991 (to appear in Proc. 2nd Algorithmic Learning Theory).

[10] B.K. Natarajan, "On learning sets and functions", *Machine Learning*, Vol. 4, pp. 67–97, 1989.

[11] Protein Identification Resource, National Biomedical Research Foundation.

[12] J.R. Quinlan, "Induction of decision trees", *Machine Learning*, Vol. 1, pp. 81–106, 1986.

[13] J.R. Quinlan and R.L. Rivest, "Inferring decision trees using the minimum description length principle", *Inform. Comput.*, Vol. 80, pp. 227–248, 1989.

[14] J.K.M. Rao and P. Argos, "A conformational preference parameter to predict helices in integral membrane proteins", *Biochim. Biophys. Acta*, Vol. 869, pp. 197–214, 1986.

[15] T. Shinohara, "Polynomial time inference of pattern languages and its applications", *Proc. 7th IBM Symp. Mathematical Foundations of Computer Science*, pp. 191–209, 1982.

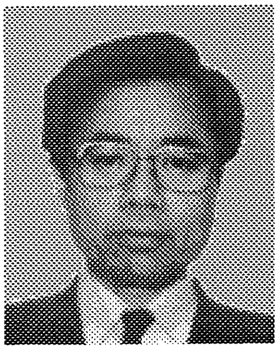
[16] T. Shinohara, "Polynomial time inference of regular pattern languages", *Proc. RIMS Symp. Software Science and Engineering* (Lecture Notes in Computer Science, Vol. 147), pp. 115–127, 1983.

[17] P.E. Utgoff, "Incremental induction of decision tree", *Machine Learning*, Vol. 4, pp. 161–186, 1989.

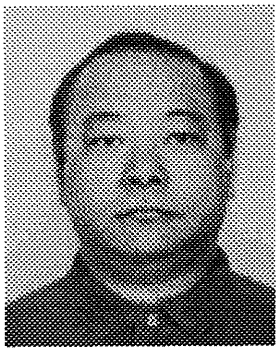
[18] L. Valiant, "A theory of the learnable", *Commun. ACM*, Vol. 27, pp. 1134–1142, 1984.

[19] G. von Heijine, "Transcending the impenetrable: how proteins come to terms with membranes", *Biochim. Biophys. Acta*, Vol. 947, pp. 307–333, 1988.

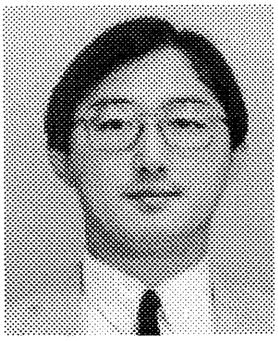
About the Authors



Setsuo Arikawa (有川　節夫) was born in Kagoshima on April 29, 1941. He received the B.S. degree in 1964, the M.S. degree in 1966 and the Dr.Sci. degree in 1969 all in Mathematics from Kyushu University. Presently, he is Professor of Research Institute of Fundamental Inforamtion Science, Kyushu University. His research interests include algorithmic learning theory, logic and inference in AI, and information retrieval systems.



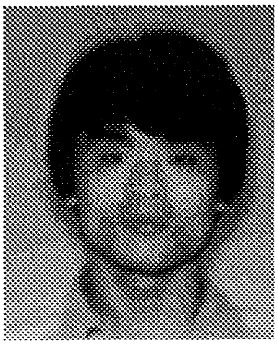
Satoru Kuhara (久原　哲) was born in Fukuoka on April 20, 1950. He recieved the B.A. in 1974, the M.A. degree in 1976 and the Dr. Agr. in 1980 from Kyushu University. Currently, he is an Associate Professor of Graduate School of Genetic Resources Technology, Kyushu University. His present interests include computer analysis of genetic information and protein structure.



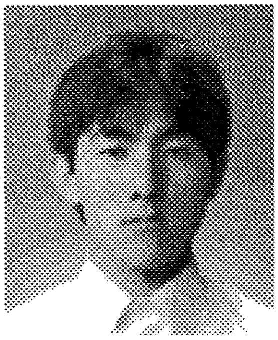
Satoru Miyano (宮野　悟) was born in Oita on December 5, 1954. He received the B.S. in 1977, the M.S. degree in 1979 and the Dr. Sci. in 1984 all in Mathematics from Kyushu University. Presently, he is an Associate Professor of Research Institute of Fundamental Information Science, Kyushu University. His present interests include parallel algorithms, computational complexity and computational learning theory.

Research Institute of Fundamental Information Scinece, Kyushu University 33, Fukuoka 812, Japan.

About the Authors



Yasuhito Mukouchi (向内　康人) was born in Hyougo on September 25, 1963. He received the B.E. and the M.A. from University of Osaka Prefecture in 1987, 1991, respectively. Currently, he is a graduate student at Doctor Course of Department of Information Systems, Kyushu University. His research interests are inductive inference and computational learning theory.



Ayumi Shinohara (篠原　歩) was born in Fukuoka on July 18, 1965. He received the B.S. degree in 1988 in Mathematics and the M.S degree in 1990 in Information Systems from Kyushu University. Presently, he is an Assistant of Research Institute of Fundamental Information Science, Kyushu University. His research interests are computational learning theory and algorithms.

Takeshi Shinohara (篠原　武) was born in Fukuoka on January 23, 1955. He received the B.S. in 1980 from Kyoto University, and the M.S. degree and the Dr. Sci. from Kyushu University in 1982, 1986, respectively. Currently, he is an Associate Professor of Department of Artificial Intelligence, Kyushu Institute of Technology. His present interests include information retrieval, string pattern matching algorithms and computational learning theory.

Research Institute of Fundamental Information Scinece, Kyushu University 33, Fukuoka 812, Japan.