Discovering Patterns and Subfamilies in Biosequences

Alvis Brāzma*
abrazma@cclu.lv
Institute of Mathematics and Computer Science
University of Latvia
29 Rainis Bulevard
LV-1459 Riga, Latvia

Inge Jonassen
inge@ii.uib.no
Department of Informatics
University of Bergen, HIB
N5020 Bergen, Norway

Esko Ukkonen
Esko.Ukkonen@cs.Helsinki.FI
Department of Computer Science
P.O.Box 26 (Teollisuuskatu 23)
FIN-00014 University of Helsinki
Finland

Jaak Vilo
Jaak.Vilo@cs.Helsinki.FI
Department of Computer Science
P.O.Box 26 (Teollisuuskatu 23)
FIN-00014 University of Helsinki
Finland

Abstract

We consider the problem of automatic discovery of patterns and the corresponding subfamilies in a set of biosequences. The sequences are unaligned and may contain noise of unknown level. The patterns are of the type used in PROSITE database. In our approach we discover patterns and the respective subfamilies simultaneously. We develop a theoretically substantiated significance measure for a set of such patterns and an algorithm approximating the best pattern set and the subfamilies. The approach is based on the minimum description length (MDL) principle. We report a computing experiment correctly finding subfamilies in the family of chromo domains and revealing new strong patterns.

Keywords: pattern discovery, sequence motifs, machine learning, protein subfamilies, PROSITE, clustering, algorithms, Bayesian inference, MDL principle

Introduction

The problem that we are considering in this paper is, given a set of biosequences, find simultaneously subsets sharing interesting, biologically relevant common patterns, and the patterns themselves. The sequences are unaligned, and the set may contain noise, i.e., there may be an unknown number of biologically unrelated or faulty sequences. The problem can be viewed from two different perspectives:

1. as finding a set of interesting patterns in a set of unaligned biosequences containing noise of unknown level;
2. as grouping or clustering a set of biosequences into subsets so that each subset shares a distinct common pattern and the noise is sorted out.

A subproblem of this is finding a pattern common to a set of sequences in the presence of an unknown level of noise.

In particular we consider sequences representing proteins and patterns of the type used in PROSITE database (Bairoch 1992), but the method can be applied for any type of sequences and any type of deterministic patterns (motifs). The problem is important, for instance, in finding and characterizing protein families. A protein class may contain subclasses, and while each subclass may have a specific motif, the entire class may have only a very general motif (Wu & Brutlag 1995). Moreover the problem of simultaneous finding patterns and subclasses is an essential part of a more general problem of 'making sense out of biosequential data'.

Our approach is based on

1. finding a definition of an 'interesting set of patterns', effectively leading to a definition of 'significance' or 'fitness' measure of patterns in respect to a set of sequences; and
2. developing an algorithm that, given a set of sequences, finds a set of patterns with high 'fitness' according to the definition.

The most important requirement to the fitness measure is that the subsets defined by the fittest patterns should be close to subfamilies of some biological meaning. We also consider it important that there should be theoretical foundations for the fitness measure, as it can provide a better general understanding of the

* The results were obtained while the author was working at the Department of Computer Science, University of Helsinki.

subject-matter. The algorithm finding the fittest pattern set should have acceptable run-time for real-world data. Additionally we want to prove some performance guarantees to obtain better understanding of the limitations and advantages of the method.

The problem studied in this paper is closely related to and contains as a subproblem the automatic discovery of consensus patterns in biosequences. Such patterns help to understand the relationship between a sequence, structure, and function, as well as to identify new members of the families. Patterns of different types have been used for these purposes. They can be divided into probabilistic patterns, e.g., Hidden Markov Models (Krogh *et al.* 1994) or profiles (Gribskov, McLachlan, & Eisenberg 1987), and deterministic ones. Each of the types has applications where it works better; some comparative analysis is given in (Wu & Brutlag 1995). The most general among the deterministic patterns seems to be the pattern class used in PROSITE database. In biocomputing research this class is becoming a standard for the types of patterns that are within the expressive power of regular languages.

Quite a large number of different algorithms for automatic discovery of patterns in biosequences have been developed (Neuwald & Green 1994; Neuwald, Liu, & Lawrence 1995; Sagot, Viari, & Soldano 1995; Saqi & Sternberg 1994; Smith, Annau, & Chandrasegaran 1990; Smith & Smith 1990), for survey see (Brazma *et al.* 1995). The only algorithm that we are aware of that captures a large subset of the pattern class used in PROSITE database (including gaps of flexible length and flexibly defined groups of amino-acids) is Pratt (Jonassen, Collins, & Higgins 1995; Jonassen 1996). To have real applications to biology, any pattern discovery algorithm should be able to deal with noise. Usually a certain upper bound on the level of noise is assumed. A typical pattern discovery method is based on defining a certain pattern significance measure and looking for the most significant patterns common to at least a given number of strings. Another option is to measure the size of the 'block' in sequences determined by the pattern. A problem in this case is that a small number of very similar sequences may produce a block of a relatively large size. Therefore, a minimal number of sequences sharing a pattern still has to be assumed, or some heuristics applied for filtering out very similar sequences. Ideally, a kind of pattern 'fitness' measure should be used that incorporates both the significance of the pattern as such and its fitness in respect to the given sequences. However, to find a balanced measure is not an easy task (Jonassen, Helgesen, & Higgins 1996). A theoretically substantiated balanced measure can be based on the minimum description length (MDL) principle (Rissanen 1978; Li & Vitanyi 1993; Kilpeläinen, Mannila, & Ukkonen 1995; Brazma, Ukkonen, & Vilo 1995).

Very recently the problem of discovering a set of patterns has also come under attention (Conklin 1995). A Gibbs sampling based algorithm that detects motif-encoding regions (i.e., patterns) in sequences and optimally partitions them into distinct motif models is described in (Neuwald, Liu, & Lawrence 1995). Patterns in this approach are probabilistic. Like in most pattern discovery algorithms, it is assumed that each group in the partition contains at least a given number of sequences. A variation of this method is developed in (Bailey & Elkan 1995). Discovering sets of deterministic patterns has been studied in (T. Shoudai *et al.* 1995), but there an upper bound on the number of subclasses has to be assumed *a priori*. Grouping the sequences and sorting out the noise without any *a priori* assumption on the level of noise and the size of the groups has been studied in (Wu & Brutlag 1995), but there the sequences are assumed to be given prealigned.

In our approach we do not make any such assumptions. Particularly we do not impose any strict bounds on the level of noise, on the number of subfamilies, or on their size. As the pattern class we consider PROSITE patterns. We deduce the pattern *fitness measure* by Bayesian inference from some simple assumptions about the probability distribution of the sequences. This corresponds to the MDL principle meaning in this particular context that the best pattern set is the one that minimizes the sum of

- the length (in bits) of the patterns; and
- the length (in bits) of data when encoded with the help of the patterns.

We have developed an algorithm finding the pattern sets with high fitness according to the defined MDL fitness measure. The algorithm is based on the pattern discovery algorithm Pratt (Jonassen 1996) and a greedy set cover algorithm. The greedy algorithm guarantees the length of the solution to be within the logarithmic factor from the theoretical optimum.

We have carried out a number of computing experiments and obtained encouraging results. In the first experiment we created an artificial "superfamily" by mixing together protein sequences from four different families. Our algorithm restored the families. In the second experiment we used the family of chromo domains. The algorithm correctly found the subfamilies and discovered a distinct, strong pattern in each of them.

The structure of this paper is as follows. In the next section we derive a pattern significance (fitness) measure from some probability distribution assumptions. In the third section we describe the algorithm finding good pattern sets according to this measure and in the fourth section we report the experimental results. Finally, we discuss the implications.

Significance Measure

Let Σ be a finite alphabet. Let $A = \{\alpha_1, \ldots, \alpha_n\}$ be a set of sequences, such that $\alpha_i \in \Sigma^*$. In practice A will be a set of biosequences believed to be biologically related. Our goal is, given the set A, to find disjoint subsets (subfamilies) $B_1, \ldots, B_k$ of A and patterns $\pi_1, \ldots, \pi_k$ such that $A = B_1 \cup \ldots \cup B_k$, and each B_i shares the pattern π_i $(i = 1, \ldots, k)$. The patterns π_i should be good in the sense that they should define a partition into biologically sensible subfamilies B_i and each π_i should ideally describe biologically relevant features shared by the sequences in B_i. Unrelated and noisy sequences should result in singleton sets $B_i = \{\alpha_j\}$.

Note that if the sets B_i were given *a priori*, then the problem would become that of finding interesting patterns common to each set B_i. If, on the other hand, patterns π_i were given, then the problem would essentially boil down to pattern matching. We, however, want to solve both problems simultaneously.

Let us define the class of patterns we are considering.

PROSITE Patterns and the Set Cover

In biocomputing literature the PROSITE patterns are usually introduced by examples. For instance, a pattern `W-x(2)-[LI]-[SAG]-x(4,5)-R` means that the sequence should contain a substring beginning with `W` followed by two arbitrary characters, followed by either `L` or `I`, followed by `S`, or `A`, or `G`, followed by from 4 to 5 arbitrary characters, followed by `R`.

For our purposes we need a more formal definition. Let $\Phi = \{b_1, \ldots, b_s\}$ be an alphabet disjoint from Σ, by which we in fact denote some fixed subsets $K_1, \ldots, K_s$ of Σ. For each b_i we define $L(b_i) = K_i$ $(i = 1, \ldots, s)$. Let us assume that $s \geq m + 1$, $K_1 = \{a_1\}, \ldots, K_m = \{a_m\}$, and $K_s = \Sigma$. We will frequently identify b_i with a_i, for $1 \leq i \leq m$, and we will denote b_s by x.

Let X be the set of all objects of the type $x(t, v)$, where t and v are nonnegative integers, such that $t \leq v$. We define $L(x(t, v))$ be the set of all words over Σ^* of length between t and v (i.e., $L(x(t, v)) = \{\alpha |\ \alpha \in \Sigma^*,\ t \leq |\alpha| \leq v\}$). Thus $x(t, v)$ effectively means the *wildcard* of flexible length from t to v. If $d_1 \ldots d_h$ is a string over alphabet $\Phi \cup X$ then define $L(d_1 \ldots d_k) = \{\delta_1 \ldots \delta_k | \delta_1 \in L(d_1), \ldots, \delta_k \in L(d_k)\}$. A *P-pattern* is a string of the type $\pi = *\phi*$, where ϕ is a string over the alphabet $\Phi \cup X$. We define the *language* of a pattern $*\phi*$ as follows:

$$L(*\phi*) = \{\gamma\beta\delta | \beta \in L(\phi), \gamma \in \Sigma^*, \delta \in \Sigma^*\}.$$

We say that a sequence α *contains* or *matches* a pattern π, if $\alpha \in L(\pi)$. As x is equivalent to $x(1, 1)$, we assume that the wildcards are only in the latter form. The definition of P-patterns covers in practice the most frequently used subclass of patterns that can be expressed in PROSITE pattern notation. If we restrict the characters of ϕ to Σ, then we obtain a subclass of the *substring patterns*. A sequence matches such a pattern if it contains the substring ϕ.

Let $B_1, \ldots, B_k$ be a partition of A (i.e., $B_1, \ldots, B_k$ are disjoint and their union is A) and let $\Pi = \{\pi_1, \ldots, \pi_k\}$ be a set of patterns such that the pattern π_j matches all the sequences of the set $B_j = \{\beta_1^j, \ldots, \beta_{l_j}^j\}$. We call $\Omega = \{(\pi_1, B_1), \ldots, (\pi_k, B_k)\}$ a *cover* of A. We call Π the *pattern set* of Ω. Thus, the notion of the cover includes both the partition and the patterns, and our problem can be formulated as finding the best cover.

The first step is to develop a fitness measure $F(\Omega)$ such that for a given set of biosequences A, the value of $F(\Omega)$ is high when the respective partition $B_1, \ldots, B_k$ corresponds to subfamilies sharing 'interesting' patterns $\pi_1, \ldots, \pi_k$. We will define $F(\Omega)$ so that it reaches maximum for Ω that is the most probable for the given A under certain probability distribution assumptions, and we will show that this definition corresponds to the MDL principle.

First we develop a fitness measure for the substring patterns.

Significance Measure for Substring Patterns

Let us denote the length of a string α by $|\alpha|$. For a set $A = \{\alpha_1, \ldots, \alpha_n\}$ we define $\|A\| = \sum_{i=1}^{n} |\alpha_i|$, and $|A| = n$.

Let us assume that we want to transmit the set of sequences A over a channel. A trivial way would be to transmit $\alpha_1, \ldots, \alpha_n$ one after another. If we abstract from the fact that some delimiters between sequences also have to be transmitted, the message length (in characters) would be $\|A\|$. Now, suppose we know that a substring pattern $*\varepsilon*$ is present in all the sequences, i.e., $\alpha_i = \gamma_i\varepsilon\delta_i$, for some $\gamma_i, \delta_i \in \Sigma^*$. Then we can compress the message by first transmitting ε, and then $\gamma_1, \delta_1, \ldots, \gamma_n, \delta_n$. Similarly, if $\Omega = \{(*\varepsilon_j*, B_j) | j = 1, \ldots, k\}$ is a cover of A, then A can be transmitted using Ω, as:

for $j = 1$ to k do
send ε_j
for $i = 1$ to l_j do
send γ_i^j, send δ_i^j

where $l_j = |B_j|$. The message length obtained by this coding system is $M_s(\Omega) = \|A\| - \sum_{j=1}^{k} (l_j - 1)|\varepsilon_j|$. The MDL principle suggests that the best pattern set Π is the pattern set of a cover Ω that minimizes the message length $M_s(\Omega)$, c.f.(Kilpeläinen, Mannila, & Ukkonen 1995). The second term in the expression of $M_s(\Omega)$ is

$$C_s(\Omega) = \sum_{j=1}^{k} (l_j - 1)|\varepsilon_j|$$

and it can be considered as the *compression* in comparison to $\|A\|$. Minimizing $M_s(\Omega)$ equals maximizing $C_s(\Omega)$ and thus C_s defines a kind of fitness measure for pattern set Π in respect to the trivial message length A. Note that this fitness measure is similar in spirit to the one used in (Wu & Brutlag 1995).

Next let us introduce a slightly different and more efficient coding that takes into account relative frequencies of characters $a_i \in \Sigma$ and leads to a fitness measure substantiated from the point of view of Bayesian inference. Let us assume some probability distribution $P(a_i)$ for the characters $a_i \in \Sigma$, such that $\sum_{i=1}^{m} P(a_i) = 1$. In practice we take $P(a_i)$ proportional to the relative frequency f_i of a_i in a sequence database (or in A).

Let us introduce one more character \$, where $\$ \notin \Sigma$, and assume the probability $P_1(\$) = q$, $(0 < q < 1)$. We normalize probabilities $P(a_i)$ to $P_1(a_i) = (1-q)P(a_i)$, so that $P_1(a_1)+\cdots+P_1(a_m)+P_1(\$) = 1$. Let us assume that patterns π are generated in a random Bernoulli process consecutively outputting letters from $\Sigma \cup \{\$\}$ with probabilities P_1 and \$ signaling the end of the pattern. Let $\pi = \pi'\$$ be such a string, where $\pi' = e_1 \ldots e_h \in \Sigma^*$. We define the *a priori* probability of the pattern π as

$$P_1(\pi) = P_1(e_1) \cdots P_1(e_h) P_1(\$).$$

Let $\Pi = \{\pi_1, \ldots, \pi_k\}$. We define the *a priori* probability of the pattern set Π as

$$P_1(\Pi) = P_1(\pi_1) \cdots P_1(\pi_k) \cdot (1-r)^k r$$

for some $0 < r < 1$. Parameters q and r determine the average length of patterns and how many patterns (in fact, subfamilies) we *a priori* want on average.

Next, let us assume that sequences α are obtained by prefixing and appending to π_i' some random substrings generated in a similar Bernoulli process. More precisely, let $\$_2$ be one more character disjoint from Σ with a probability $P_2(\$_2) = p$. Let us now normalize the probabilities $P(a_i)$ to $P_2(a_i) = (1-p)P(a_i)$. Let some sequences β be generated by outputting characters from $\Sigma \cup \{\$_2\}$ with probabilities P_2 and $\$_2$ signaling the end. Let $\beta = \beta'\$_2$, where $\beta' \in \Sigma^*$. We assume that biosequences are obtained by prefixing and appending sequences β' to π'. The probability p can be estimated from the average length of biosequences in the respective family (more precisely, we use the fact that $\frac{1}{p}+\frac{1}{q}+\frac{1}{p}$ equals the average length of sequences α_i). Frequently p is very small, and sometimes can be assumed to be 0. For more precise definition of this model see (Brazma, Ukkonen, & Vilo 1995). Note that this model in essence is similar to that of (Neuwald, Liu, & Lawrence 1995). The ratio $q = P_1(\$)$ to $p = P_2(\$_2)$ determines the most probable relative length of strings to patterns. By varying this ratio we will slant the fitness of patterns towards longer patterns common to fewer strings or vice versa.

We assume the following coding system. First we send the message '$\pi_1\$\pi_2\$\ldots\pi_k\$$:', i.e., the patterns separated by \$ and some final delimiter ':'. The characters (including the delimiter) are encoded by bit strings of different length (in some prefix coding) depending on probabilities P_1. Next, we send substrings that we have to prefix and append to patterns to obtain A, separated by $\$_2$, and some additional delimiter signaling the switching to the next pattern. The characters are coded by bit-strings of length dependent on P_2.

It is shown in (Brazma, Ukkonen, & Vilo 1995) that for a given A the most probable pattern set Π is the pattern set of a cover Ω minimizing the message length in the described encoding system. This implies the equivalence of Bayesian and MDL approaches for the given model.

Let P_x be a probability distribution over an alphabet Σ'. For $\gamma = g_1 \ldots g_h \in \Sigma'^*$ denote

$$\ell_{P_x}(\gamma) = -\sum_{i=1}^{h} \log P_x(g_i).$$

The value of $\ell_{P_x}(\gamma)$ can be interpreted as the length of γ in bits in some optimal coding in respect to the probability distribution P_x. It has been also shown in (Brazma, Ukkonen, & Vilo 1995) that Ω minimizing the message length $M_s(\Omega)$ is the same that maximizes

$$C_s'(\Omega) = \sum_{j=1}^{k} (u_j' \cdot l_j - w_j'),$$

where $u_i' = \ell_{P_2}(\pi_j) + 2\log p$ and $w_j' = \ell_{P_1}(\pi_j) + \log(1-r)$. Thus $C_s'(\Omega)$ defines a fitness measure for substring

patterns that under the described probability assumptions determines the most probable pattern set and the partitioning for the given set of sequences A.

Next we generalize the model for the class of P-patterns.

Significance Measure for P-patterns

Let $\pi = *d_1 \dots d_h*$ be a P-pattern where the string $d_1 \dots d_h$ is generated in a similar probabilistic process as the substring patterns, except that the alphabet is now $\Phi \cup \{\$\}$, and the probability distribution P_1 is over $\Phi \cup \{\$\}$. For each wildcard character x we generate additionally (with equal probabilities) two random integers t and v, $0 \leq t \leq v \leq g$, to obtain $x(t,v)$. The assumption of an existence of the upper bound constant g is not essential in our model, however, it simplifies the analysis.

The probability distribution P_1 should be regarded as a tool for giving preferences to different characters representing single amino-acids as well as their groups used in the patterns. It may be natural to assign probabilities proportional to the amino-acid frequencies to characters from Σ. Theoretically substantiated probabilities for the group characters can possibly be calculated from substitution matrices. Alternatively we can use some heuristics giving preferences to the groups that we want to use. A simpler heuristics, which we use in the experiments, is to assign the probability 0 to the groups we do not use, and to distribute the probabilities uniformly among all other characters.

The optimal message length for a P-pattern π is

$$M_1(\pi) = \ell_{P_1}(\pi) + \#X(\pi) \cdot 2 \log g,$$

where $\#X(\pi)$ is the number of wildcard characters x in the pattern π.

Next assume that the sequences α_i are generated from patterns in a similar process as in the case of the substring patterns, but additionally the pattern characters b_j with $j > m$ are replaced by $a_i \in K_j$ (where $K_j = L(b_i)$) and gaps $x(t,v)$ are filled. The probability $P(a_i|b_j)$ of a substitution of a_i for b_j is proportional to $P(a_i)$ and can be regarded as the conditional probability of a_i on condition b_j. More precisely, let $P(K_j) = \sum_{a_i \in K_j} P(a_i)$, then $P(a_i|b_j) = \frac{P(a_i)}{P(K_j)}$ if $a_i \in K_j$, and 0 otherwise. The gaps $x(t,v)$ are filled in by randomly choosing an integer y between t and v, and generating a string of length y by outputting y characters according to the probability distribution P.

Let $\hat{\Phi}(\pi) = \langle i_1, \dots, i_k \rangle$ be positions in π that are from Φ (i.e., any character except the type $x(t,v)$). Let us denote by $\pi(h)$ the character $b_j \in \Phi$ in the position h of π. Let us assume that sequences from $B = \{\alpha_1, \dots, \alpha_l\}$ contain a pattern π. It can be shown (see Appendix A) that in this case minimizing the message length (or maximizing the probability) equals maximizing

$$F'(\pi, l) = l \cdot (I(\pi) + 2 \log p + f(\pi)) - M_1(\pi),$$

where

$$I(\pi) = I_1(\pi) + I_2(\pi),$$

$$I_1(\pi) = \sum_{j \in \hat{\Phi}(\pi)} I'(\pi(j)),$$

$$I'(b_h) = -\sum_{i=1}^{m} P_2(a_i) \log P_2(a_i) + \\ + \sum_{a_i \in K_h} P_2(a_i|b_h) \log P_2(a_i|b_h),$$

$$I_2(\pi) = -\sum_{x(t_i,v_i)} \log(v_i - t_i + 1)$$

(the sum is over all characters from X in π), and $f(\pi) = \frac{1}{3} g |\pi| P_1(x) \log(1 - P_2(\$_2))$. $I'(b_h)$ can be interpreted as the decrease in uncertainty provided by the knowledge that the character in the given position has been chosen from a limited variety K_h. The term $2 \log p$ comes from the delimiters $\$_2$. The term $f(\pi)$ is very small and can be ignored in practice.

$I_1(\pi)$ is exactly the same as the first term in the pattern significance measure I of (Jonassen, Collins, & Higgins 1995). The second term in (Jonassen, Collins, & Higgins 1995), which is a heuristic correction for flexibilities, is in our case replaced by a theoretically more substantiated correction I_2, suggesting logarithmic dependence on the gap length, rather than linear.

For a cover $\Omega = \{(\pi_1, B_1), \dots, (\pi_k, B_k)\}$ of A, the total fitness $F(\Omega) = \sum_{j=1}^{k} F'(\pi_j, l_j) - k \log(1 - r)$, where $l_j = |B_j|$, and can be expressed as

$$F(\Omega) = \sum_{j=1}^{k} (u_j \cdot l_j - w_j),$$

where $u_j = I(\pi_j) + 2 \log p + f(\pi_j)$, and $w_j = \ell_{P_1}(\pi_j) + \#X(\pi_j) \cdot 2 \log g - \log(1 - r)$. The dominant term is u_j and it corresponds to the compression gained by using pattern π_j. The term w_j corresponds to the length needed to describe pattern π_j plus delimiters separating groups B_j.

The pattern set Π of Ω maximizing $F(\Omega)$ can be proved to minimize the message length $M(\Omega)$ if encoded as described above, and to be the most probable for the set A under the described probability model. In this way $F(\Omega)$ defines a theoretically substantiated *fitness measure* for a pattern set Π and the corresponding partition $B_1, \dots, B_k$ in respect to the given sequences.

Assuming that P_1 is uniform and ignoring $f(\pi)$, we obtain that $F'(\pi, l)$ can be approximated by $F''(\pi, l) =$

$l \cdot (I(\pi) - c_0) - (c_1|\pi| + c_2 \# X(\pi) + c_3)$ for $c_0 = -2 \log p$, $c_1 = \log |\Phi|$, $c_2 = 2 \log g$, and $c_3 = -\log(1-r) - \log(1-q)$ (where $p = P_2(\$_2)$, $q = P_1(\$)$, r is from the definition of pattern set Π, $|\Phi|$ is the size of the pattern alphabet Φ, and g is the maximal size of wildcards in patterns). As c_3 depends only on q and r, which do not have good theoretical estimates, in practice it should be adjusted experimentally.

MDL Approximation Algorithm

In this section we give a greedy heuristic algorithm that finds the set of patterns defining a partitioning of A. The computation of the absolute best partitioning is NP-hard (hence not believed possible in reasonable time) even when the set of patterns is given, see (Brazma, Ukkonen, & Vilo 1995). Still, we can show that our polynomial-time implementation gives a solution that is guaranteed to be close to the optimal.

By M_{opt} we denote the minimal value of the message length $M(\Omega)$ for the fixed pattern class, say substring or PROSITE patterns. Each set B_i in a cover Ω will be a subset of the set D_i of sequences in A that match the pattern π_i. It is proved in (Brazma, Ukkonen, & Vilo 1995) that the following algorithm `Greedy`, given a set of strings A, finds a cover Ω, such that $M(\Omega) \leq M_{opt} \times \log |A| + O(1)$.

Let $u(\pi_i) = u_i$, and $w(\pi_i) = w_i$ be functions defined in the previous section. Then the algorithm can be described as follows.

```
Greedy(A) return cover of A
U ← A ; Ω ← ∅
while U ≠ ∅ do
      find a pair (π, D) maximizing (u(π)·|D| − w(π)) / |D|,
      where D ⊆ U and π matches all sequences of D
      Ω ← Ω ∪ {(π, D)}
      U ← U − D
end
return Ω
```

The algorithm is a variation of the greedy algorithm for the weighted set cover problem (Chvátal 1979). It follows from the previous section that the gain (i.e., the compression achieved) by selecting pattern π_i is $u(\pi_i) \cdot |D_i| - w(\pi_i)$, where $D_i \subseteq A$ consists of the strings in A that are covered by π_i but not by any pattern that has already been selected in the previous steps. Pattern π_i that gives the maximal relative compression (compression per one string) is selected ("greedily') next. Although the algorithm above is formulated differently than that in (Brazma, Ukkonen, & Vilo 1995), they can be shown to be equivalent. The running time of the new version may be different because we do not generate all patterns from the set A, but instead find only the best pattern for decreasing subsets U of A.

In (Brazma, Ukkonen, & Vilo 1995) we show that for substring patterns, the cover Ω can be computed in time $O(|A| \cdot \|A\|)$. For P-patterns, the running time of algorithm `Greedy` is at most $|A|$ times the time of finding the best pattern π for the set $U \subseteq A$. In practice, to find the P-patterns π we modified an existing algorithm Pratt (Jonassen 1996) so that it maximizes the function $u(\pi) - w(\pi)/|D|$.

Pratt is a tool for pattern discovery in a set of protein sequences (Jonassen, Collins, & Higgins 1995; Jonassen 1996). It is able to discover P-patterns defined in Section 2.2, and allows the user to give a set of parameters defining restrictions on the patterns to be searched for. For example, the user gives a list of the possible amino-acid groups (i.e., the alphabet Φ in our terminology), the maximum pattern length, the maximum length of a wildcard region, and the maximum amount of flexibility to be allowed (i.e., maximum value for v and $v - t$ for wildcard regions $x(t, v)$).

Pratt aims at finding the highest scoring patterns present in at least l of a given set of n sequences according to a fitness measure similar to I in the previous section. Motivated by the results in Section 2, the fitness measure in Pratt was modified to use logarithmic gap penalty, i.e., for every wildcard region $x(t, v)$ in a pattern π we subtract $\log_2(v - t + 1)$ from $I_1(\pi)$.

Pratt optimizes the function I for pre-given l while in our MDL-algorithm we want to maximize the function $E(\pi, l) = \frac{F''(\pi, l)}{l}$. Note that to optimize I for some small value of l (that should give best I) is not sufficient, because there may be a different l that gives worse I-measure but better E-measure. The problem can be overcome if we compute the best pattern according to I-measure for all possible values l. Then for some l we will get the pattern that maximizes E among all possible values l. As very small l values are computationally difficult for Pratt, we use some lower bound l_{min}.

```
F-Pratt (A) return (π, E(π, l))
bestpattern = ε; bestcost = 0
for l = |A| downto l_min do
      π = pattern produced by Pratt(A, l)
      if E(π, l) ≥ bestcost then
            bestcost = E(π, l);
            bestpattern = π;
end
return bestpattern, bestcost
```

This algorithm actually implements the greedy choice of algorithm `Greedy`: "find a pair (π, D) maximizing $\frac{u(\pi) \cdot |D| - w(\pi)}{|D|}$, where $D \subseteq U$." MDL-Pratt is the program implementing the algorithm `Greedy` using `F-Pratt`.

We used a heuristic version of Pratt which is de-

scribed in (Jonassen 1996), therefore formally we are not guaranteed to find the highest-scoring patterns. We want to use quite a general class of patterns, and the non-heuristic version of Pratt would be too time-consuming to be practical. As I and E measures get closer to each other for larger l, another speed-up is to evaluate Pratt not for every possible l, but instead to use some logarithmic scale to skip most of them, especially for larger l. Experimental results indicate that it is possible to skip many of the values l and still obtain almost the same result as when evaluating all l-values.

In comparison with the other existing pattern set discovery algorithms (Neuwald, Liu, & Lawrence 1995; T. Shoudai *et al.* 1995; Wu & Brutlag 1995), MDL-Pratt is able to generate more complicated candidate patterns covering different subsets of the given sequences and uses more powerful fitness measure for their evaluation. This allows MDL-Pratt to use a very simple method, namely the greedy heuristics, for splitting the original set into subfamilies. In (Neuwald, Liu, & Lawrence 1995; T. Shoudai *et al.* 1995) a nontrivial probabilistic algorithm is used for generating candidate patterns and subfamilies. The pattern fitness there is evaluated by using maximum likelihood (ML) principle which, in fact, is a special case of MDL principle (Li & Vitanyi 1993). Basically it means taking the "most informative pattern" common to at least given number of sequences. In our terminology it would essentially mean taking $w_j = 0$ and requiring that the pattern matches at least a given number of sequences. Experiments have shown that combined with this simplified fitness measure the greedy heuristics is not sufficient. The fitness measure in (Wu & Brutlag 1995) can also be regarded as a special case of our MDL measure, and essentially be obtained from our measure by taking $w_j = u_j$. The pattern candidates in (Wu & Brutlag 1995) are generated by simpler algorithm than MDL-Pratt, based on the assumption that the patterns are prealigned. On the other hand (Wu & Brutlag 1995) uses a more complicated algorithm, *beamsearch*, for splitting the sequence set into subfamilies.

Experiments

Here we present two experimental results obtained by MDL-Pratt algorithm described in the previous sections for some real biosequence data. In the first experiment the input for the algorithm was an artificial "superfamily" of biosequences created by mixing together sequences from different protein families. The algorithm correctly splitted them back into the original groups. In the second experiment the algorithm was able to discover subfamilies in a family of chromo domains that are very similar to subfamilies reported by domain experts (Aasland & Stewart 1995). For complete test data and more details see (`http://www.ii.uib.no/~inge/papers/mdl/`).

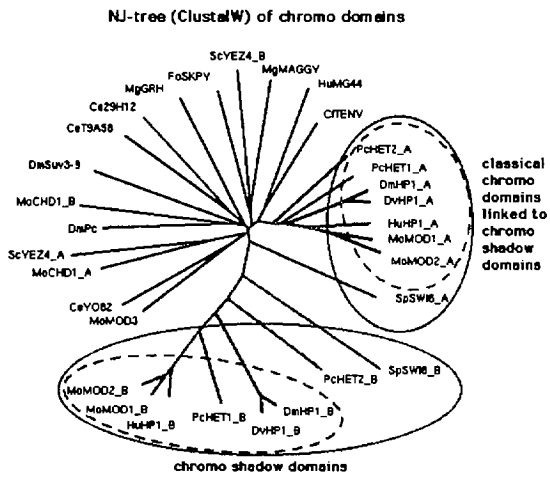

Figure 1: Estimate of the phylogenetic tree for the chromo domain segments included in the analysis. The tree was produced using Clustal W (Thompson, Higgins, & Gibson 1994) which uses the neighborhood joining method (NJ) for estimating phylogenetic trees (Saitou & Nei 1987), and was taken from the 'official' World-Wide-Web pages for (Aasland & Stewart 1995). The dashed ellipses show the two most important subsets identified by MDL-Pratt.

In the first experiment we retrieved the sequences in four PROSITE (release 13.0, January 1996) families using SRS (Etzold & Argos 1993). The families are TP1 (AC PS00541) containing 6, TP2_1 (AC PS00970) containing 5, SAR_1 (AC PS01020) containing 4, and SPASE_II (AC PS 00855) containing 5 sequences. These 20 sequences were input to the MDL-Pratt system with parameters $c_0 = 10$, $c_1 = 10$, $c_2 = 3$, and $c_3 = 10$. It recovered all four families, and one pattern conserved in each of them.

As a more complex experiment we used a set of 31 sequence segments, all believed to contain chromo domains which are involved in chromatin regulation. The 31 segments included two sets identified by Aasland and Stewart: 8 classical chromo domains linked to chromo shadow domains (set 1), and 8 chromo shadow domain segments (set 2) (Aasland & Stewart 1995). Figure 1 shows an estimated phylogenetic tree for the 31 segments, where the subfamilies are shown. The tree was produced using Clustal W (Thompson, Higgins, & Gibson 1994) which uses the neighborhood joining (NJ) method for estimating phylogenetic trees (Saitou & Nei 1987).

The cover produced by MDL-Pratt with $c_0 = 10$,

1. DmHP1_A DvHP1_A HuHP1_A MoMOD1_A MoMOD2_A PcHET1_A PcHET2_A
E-x(0,1)-E-E-[FY]-x-V-E-K-[IV]-[IL]-D-[KR]-R-x(3,4)-G-x-V-x-Y-x-L-K-
W-K-G-[FY]-x-[ED]-x-[HED]-N-T-W-E-P-x(2)-N-x-[ED]-C-x-[ED]-L-[IL].

2. DmHP1_B DvHP1_B HuHP1_B MoMOD1_B MoMOD2_B PcHET1_B
L-x(2,3)-E-[KR]-I-[IL]-G-A-[TS]-D-[TSN]-x-G-[EDR]-L-x-F-L-x(2)-[FW]-
[KE]-x(2)-D-x-A-[ED]-x-V-x-[AS]-x(2)-A-x(2)-K-x-P-x(2)-[IV]-I-x-F-Y-E

3. DmPc MoMOD3 HuMG44 CfTENV FoSKPY MoCHD1_A MoCHD1_B ScYEZ4_B
Y-x(0,2)-L-[IV]-K-W-x(6)-[HE]-x-[TS]-W-E-x(4)-[IL]

Figure 2: The cover obtained when running MDL-Pratt on the 31 chromo domain sequence segments. The remaining 10 sequences were represented by singleton sets in the cover, i.e., no pattern was found between them decreasing the description length for this set of parameters.

$c_1 = 10$, $c_2 = 3$, $c_3 = 10$, is shown in figure 2. The two first sets in this cover correspond to sets 1 and 2. The first set in the cover produced by MDL-Pratt is set 1 defined by Aasland and Stewart, except that the SPSWI6_A segment is not included, and the second set in the cover is set 2 defined by Aasland and Stewart except that PcHET2_B and SpSWI6_B are not included. The reason that the sets identified by MDL-pratt do not include these segments, is probably that the remaining segments share a pattern that is significantly stronger than all patterns (in the class of patterns explored by Pratt) shared by the complete set 1, respectively set 2. This means that for the specific set of sequences our algorithm has been able to automatically discover the subfamilies of chromo domains corresponding to Aasland and Stewart's classification

The algorithm also found a third "subfamily" sharing a common pattern, but this pattern is much weaker. Note also, that the values of c_0, c_1, c_2 and c_3 are very close to the theoretical estimates for the particular sequences and patterns.

In this specific test case, we analyzed segments (and not complete sequences) in order to make comparison with the results in (Aasland & Stewart 1995) easier, and to save computation time. Analyzing sequence segments is relevant for example if one wants to analyze a set of segments found to have local similarities to a query sequence, using a database homology search program, such as BLAST (Altschul *et al.* 1990). Most of the computing time used by MDL-Pratt was spent to produce the set of hypotheses for the greedy set cover algorithm to chose from. The analysis of the 31 sequence segments in the chromo domain test case, took around 10 minutes on a DEC alpha workstation.

These results indicate that MDL-Pratt can be used to discover family- and subfamily-relationships in a set of sequences. It depends on the existence of relatively strong patterns (in the class considered by Pratt) being conserved in the families/subfamilies to be identified.

Discussion

We have introduced a theoretically founded significance (fitness) measure based on MDL principle for a class of PROSITE patterns allowing ambiguous pattern positions and variable length wildcards. We modified the pattern discovery algorithm Pratt accordingly. As far as we know, this is the first explicit application of MDL principle for pattern discovery in biosequences.

The significance measure has been derived from simple assumptions regarding the probability distribution of biosequences. It contains a number of free variables having a direct interpretation in the probability model. The values of most of the variables can be estimated from the frequencies of amino-acids in databases and the average length of sequences. Others can be used as parameters by changing which estimation of multi-level family relationships for the given sequences can be obtained and the main structure of a phylogenetic tree hypothesized. One could first slant the optimum towards weak patterns and big sets in order to to identify high-level families (i.e., main subtrees). Then each of these could be analyzed for sub-families using parameters slanting the optimum towards stronger patterns.

It may seem that there is an excess of the free variables. This is mainly caused by the fact that there is no good theoretical estimates of *a priori* probabilities for the groups of amino-acids. This problem has been recognized in the literature, e.g., (Sagot, Viari, & Soldano 1995; Wu & Brutlag 1995), and some heuristics has usually been applied to selecting the possible groups. Our approach gives a possibility to introduce more flexibility in selecting these groups as, by varying their *a priori* probabilities, we can slant the optimum towards preferring different groups without completely excluding any of them.

Although our approach has mainly a theoretical origin, the algorithm produced very sensible practical results. Applied to a family of chromo domains, it predicted subfamilies very similar to those defined by do-

main experts. Moreover, it revealed that the chosen sequences share a very strong common pattern, while the remaining ones do not contain this pattern.

MDL-Pratt and other pattern-based methods may be important for finding family- and subfamily-relationships in cases where high-quality global multiple alignments are difficult to obtain. Further work can include more elaborate experiments with MDL-Pratt being applied to more sequence sets. The theoretical part of this work may be extended to include MDL based evaluations of covers where patterns may match overlapping sets of sequences or may have multiple hits in sequences, and for other pattern classes. An interesting task from the algorithmic point of view is to develop an algorithm that optimizes the MDL-significance measure directly. Approaches like genetic algorithms may be useful here.

Acknowledgments

The authors wish to thank Rein Aasland for help on the test cases, and for allowing us to use the figure showing the NJ-tree for the chromo-domains (Aasland & Stewart 1995). Alvis Brazma was supported by the Finnish Centre for International Mobility and the reported research was done while the authour was working at the Department of Computer Science of the University of Helsinki. Inge Jonassen was supported by the Norwegian Research Council.

APPENDIX A: Derivation of the Fitness Measure

Let us calculate the optimal message length M_2 for sequences $B = \{\alpha_1, \ldots, \alpha_l\}$ containing a P-pattern π. To send the set $B = \{\alpha_1, \ldots, \alpha_l\}$ we send the prefixes and the suffixes of each string α_i exactly like in the case of substring patterns, but additionally we send the information regarding the positions in π that are not characters from Σ. We will deal separately with characters form Φ and from X. If $\Omega_1 = \{(\pi, B)\}$, then the message length for B can be expressed as

$$M_2(\Omega_1) = \ell_{P_2}(\text{Prefix}(B)) + \ell_{P_2}(\text{Suffix}(B)) + \\ + M_2^1(\Omega_1) + M_2^2(\Omega_1),$$

where the first and the second terms mean the message length corresponding to the sets of prefixes and suffixes, $M_2^1(\Omega_1)$ is the additional information for characters b_i, $i > m$, and $M_2^2(\Omega_1)$ for the characters $x(t, v)$.

Let us begin with $M_2^1(\Omega_1)$. We fix a particular ordering of elements $(a_{h_1}, \ldots, a_{h_d})$ for each given set $K_h = \{a_{h_1}, \ldots, a_{h_d}\}$. For each position in π containing a character $b_i \in \Phi$ we have to send the information specifying the particular character a_{h_j} in this position in the string. To specify this information takes

$$M_2^1(\Omega_1) = -\sum_{j=1}^{s}\sum_{i=1}^{m} N_{i,j}(B) \log P(a_i|b_j)$$

bits, where $N_{i,j}(B)$ is the number of substitutions of a_i for b_j that have been applied to obtain strings of B from the pattern π. Let $\hat{\Phi}(\pi) = \langle i_1, \ldots, i_k \rangle$ be positions in π that are from Φ, and $\pi(h)$ be the character $b_j \in \Phi$ in the position h of π. Then for large $\|B\|$ the last expression tends to

$$-l \cdot \textstyle\sum_{j=1}^{s} \sum_{i=1}^{m} N_j(\pi) P(a_i|b_j) \log P(a_i|b_j) = \\ -l \cdot \textstyle\sum_{h \in \hat{\Phi}(\pi)} \sum_{i=1}^{m} P(a_i|\pi(h)) \log P(a_i|\pi(h)),$$

where $N_j(\pi)$ is the number of letters b_j in the pattern π, and $l = |B|$.

For the characters of $x(t, v)$, we have to send y $(t \le y \le v)$ characters from the basic alphabet Σ, and additionally $\log_2(v - t + 1)$ bits specifying the particular y (note, that in this way we do not have to send a delimiter). Thus

$$M_2^2(\Omega_1) = \ell_P(\text{Gaps}(B)) + \sum_{x(t_i, v_i)} \log(v_i - t_i + 1),$$

where $\text{Gaps}(B)$ is the set of sequences substituted for wild-cards in π to obtain B. The second term represents the information about the length of each substitution sent together with the substitution itself. Thus, the message length corresponding to the most probable pattern π is $M((\pi, B)) = M_1(\pi) + M_2((\pi, B)) - \log r$. (The term $-\log r$ comes from the delimiter separating M_1 and M_2.)

The minimization of $M(\Omega_1)$ is equivalent to maximization of the compression $C(\Omega_1)$ in respect to $\ell_{P_2}(B)$, where $C(\Omega_1) = \ell_{P_2}(B) - (\ell_{P_2}(\text{Prefix}(B)) + \ell_{P_2}(\text{Suffix}(B)) + M_1^1(\Omega_1) + M_2^1(\Omega_1))$. For large l this tends to $F'(\pi, l)$ as defined in the second section of this paper. For more details see (Brazma, Ukkonen, & Vilo 1995).

References

Aasland, R., and Stewart, F. A. 1995. The chromo shadow domain, a second chromo domain in heterchromatin-binding protein 1, HP1. *Nucleic Acids Research* 23:3168–3173.

Altschul, S. F.; Gish, W.; Miller, W.; Myers, E. W.; and Lipman, D. J. 1990. Basic local alignment search tool. *Journal of Molecular Biology* 215:403–410.

Bailey, T., and Elkan, C. 1995. Unsupervised learning of multiple motifs in biopolymers using expectation maximization. *Machine Learning* 21:51–80.

Bairoch, A. 1992. PROSITE: a dictionary of sites and patterns in proteins. *Nucleic Acids Research* 20:2013–2018.

Brazma, A.; Jonassen, I.; Eidhammer, I.; and Gilbert, D. 1995. Approaches to automatic discovery of patterns in biosequences. Technical Report TR-113, Department of Informatics, University of Bergen, Bergen, Norway. (Available at `ftp://ftp.ii.uib.no/pub/bio/papers/survey.ps`).

Brazma, A.; Ukkonen, E.; and Vilo, J. 1995. Finding a good collection of patterns covering a set of sequences. Technical Report C-1995-60, Department of Computer Science, University of Helsinki, P. O. Bo 26, FIN-00014, University of Helsinki. (Available at `ftp://ftp.cs.helsinki.fi/pub/Reports/`).

Chvátal, V. 1979. A greedy heuristic for the set-covering problem. *Math. Oper. Res.* 4:233–235.

Conklin, D. 1995. Machine discovery of protein motifs. *Machine Learning* 21:125–150.

Etzold, T., and Argos, P. 1993. SRS - an indexing and retrieval tool for flat file data libraries. *Comput Appl Biosci* 9:49–57.

Gribskov, M.; McLachlan, M.; and Eisenberg, D. 1987. Profile analysis: detection of distantly related proteins. *Proc. Natl. Acad. Sci. U.S.A* 84:4355–4358.

Jonassen, I.; Collins, J. F.; and Higgins, D. G. 1995. Finding flexible patterns in unaligned protein sequences. *Protein Science* 4(8):1587–1595.

Jonassen, I.; Helgesen, C.; and Higgins, D. 1996. Scoring function for pattern discovery programs taking into account sequence diversity. Technical Report 116, Department of Informatics, University of Bergen, Bergen, Norway.

Jonassen, I. 1996. Efficient discovery of conserved patterns using a pattern graph. Reports in Informatics in preparation, Dept. of Informatics, Unviersity of Bergen. (Available at `ftp://ftp.ii.uib.no/pub/bio/papers/`).

Kilpeläinen, P.; Mannila, H.; and Ukkonen, E. 1995. MDL learning of unions of simple pattern languages from positive examples. In *Proceedings of the 2nd European conference EuroCOLT'95*, 252–260.

Krogh, A.; Brown, M.; Mian, I. S.; Sjoelander, K.; and Haussler, D. 1994. Hidden Markov model in computational biology. Applications to protein modelling. *Journal of Molecular Biology* 235:1501–1531.

Li, M., and Vitanyi, P. 1993. *An introduction to Kolmogorov complexity and its applications.* Texts and monographs in computer science. New York: Springer-Verlag.

Neuwald, A. F., and Green, P. 1994. Detecting patterns in protein sequences. *Journal of Molecular Biology* 239:689–712.

Neuwald, A. F.; Liu, J. S.; and Lawrence, C. E. 1995. Gibbs motif sampling: Detection of bacterial outer membrane protein repeats. *Protein Science* 4:1618–1632.

Rissanen, J. 1978. Modeling by the shortest data description. *Automatica-J.IFAC* 14:465–471.

Sagot, M.-F.; Viari, A.; and Soldano, H. 1995. Multiple sequence comparison: a peptide matching approach. In Galil, Z., and Ukkonen, E., eds., *Proc. of 6th Annual Symposium, CPM (Lecture Notes in Computer Science 937)*, 366–385. Springer.

Saitou, N., and Nei, M. 1987. The neighbour-joining method: a new method for reconstructing phylogenetic trees. *Mol. Biol. Evol.* 4:406–425.

Saqi, M. A. S., and Sternberg, M. J. E. 1994. Identification of sequence motifs from a set of proteins with related function. *Protein Engineering* 7(2):165–171.

Smith, R. F., and Smith, T. F. 1990. Automatic generation of primary sequence patterns from sets of related protein sequences. In *Proc. Natl. Acad. Sci. USA*, 118–122.

Smith, H. O.; Annau, T. M.; and Chandrasegaran, S. 1990. Finding sequence motifs in groups of functionally related proteins. In *Proc. Natl. Acad. Sci. USA*, 826–830.

T. Shoudai, M.; Lape, M.; Miyano, S.; Shinohara, A.; Okazaki, T.; Arikawa, S.; Uchida, T.; Shimozono, S.; Shinohara, T.; and Kuhara, S. 1995. BONSAI garden: parallel knowledge discovery system for amino acid sequences. In C. Rawlings et al., ed., *Proc. of Third International Conference on Intelligent Systems for Molecualar Biology*, 359–366. Menlo Park, California: AAAI Press.

Thompson, J.; Higgins, D. G.; and Gibson, T. J. 1994. Clustal W: improving the sensitivity of progressive multiple sequence alignment through sequence weighting, position-specific gap penalties and weight matrix choice. *Nucleic Acids Research* 22:46.

Wu, T. D., and Brutlag, D. L. 1995. Identificaion of protein motifs using conserved amino acid properties and partitioning techniques. In C. Rawlings et al., ed., *Proc. of Third International Conference on Intelligent Systems for Molecular Biology*, 402–410. Menlo Park, California: AAAI Press.