Generalized Planted (l,d)-Motif Problem with Negative Set[⋆]

Henry C.M. Leung and Francis Y.L. Chin

Department of Computer Science,
The University of Hong Kong,
Pofulam, Hong Kong
{cmleung2,chin}@cs.hku.hk

Abstract. Finding similar patterns (motifs) in a set of sequences is an important problem in Computational Molecular Biology. Pevzner and Sze [18] defined the planted (l,d)-motif problem as trying to find a length-l pattern that occurs in each input sequence with at most d substitutions. When d is large, this problem is difficult to solve because the input sequences do not contain enough information on the motif. In this paper, we propose a generalized planted (l,d)-motif problem which considers as input an additional set of sequences without any substring similar to the motif (negative set) as extra information. We analyze the effects of this negative set on the finding of motifs, and define a set of unsolvable problems and another set of most difficult problems, known as "challenging generalized problems". We develop an algorithm called VANS based on voting and other novel techniques, which can solve the (9,3), (11,4),(15,6) and (20,8)-motif problems which were unsolvable before as well as challenging problems of the planted (l,d)-motif problem such as (9,2), (11,3), (15,5) and (20,7)-motif problems.

1 Introduction

A genome is a sequence consisting of four symbols 'A', 'C', 'G' and 'T'. Along the genome are substrings, called *genes*, which are blueprints of proteins. In order to decode a gene (*gene expression*) to produce the corresponding protein, a molecule called a *transcription factor* binds to a short region (6 - 20 base pairs), called the *binding site*, in the promoter region of the gene. One kind of transcription factor can bind to the binding sites of several genes to cause these genes to coexpress. These binding sites, which should have similar lengths and patterns, can be represented by a pattern called *motif.* The motif discovering problem [14, 18] is predicting the motif given a set of coexpressed genes, i.e., when given a set of sequences T, each of which contains at least one binding site. Pevzner and Sze [18] gave a precise definition of this problem.

Planted (l,d)-Motif Problem: Suppose there is a fixed but unknown string

[⋆] This research is supported in part by an RGC grant HKU 7135/04E

M (the motif) of length l. Given t length-n sequences, each of which contains a planted *d-variant* of M, we want to determine M without a priori knowledge of the positions of the planted d-variants. A *d-variant* (or simply *variant*) is a string derivable from M with at most d symbol substitutions.

Many algorithms [1, 3–5, 8–13, 15–19] have been developed to solve this problem and have predicted some motifs successfully. However, this problem model will fail to find a solution when d is large, because there will be many length-l strings having at least one variant in each input sequence and no algorithm is likely to distinguish the motif from these strings. Buhler and Tompa [4] found the maximum d such that a planted (l,d)-motif problem is still solvable by calculating the expected number $E_t(l,d)$ of length-l strings with at least one variant in each input sequence. When $E_t(l,d)$ is small, say $E_t(l,d) \leq 1$, the problem is theoretically solvable. When $E_t(l,d)$ is large, no algorithm is likely to discover M without extra information. For example, when $t = 20$ and $n = 600$, the planted (9,3), (11,4), (15,6) and (20,8)-motif problems are unsolvable as the values of $E_t(l,d)$ for these problems are huge (2.5×10^5, 3.3×10^6, 1.8×10^8 and 3.1×10^4 respectively).

In biological experiments, besides getting a set of sequences bound by the transcription factor, we may have as a by-product another set of sequences F which are not bound by the transcription factor [2, 6, 7, 20]. We may assume sequences in F contain no d-variant of the motif M. Based on this extra information, we can modify the planted (l,d)-motif problem as follows.

Generalized Planted (l,d)-Motif Problem: Suppose there is a fixed but unknown string M (the motif) of length l. Given t length-n sequences, each of which contains a planted d-variant of M, and f length-n sequences which contains no d-variant of M, we want to determine M without a priori knowledge of the positions of the planted d-variants.

Note that when $f = 0$, the generalized planted (l,d)-motif problem (or simply generalized (l,d)-problem) is reduced to the planted (l,d)-motif problem (or simply (l,d)-problem). The extra information provided by F might make some of the previously unsolvable problems based only on information in T, e.g. (9,3), (11,4), (15,6) and (20,8)-motif problems, solvable.

In this paper, we analyze the information provided by set T and set F (Section 2) and how they are related (Section 3). We define a new set of unsolvable and also another set of "challenging" generalized (l,d)-problems (most difficult solvable problems). In Section 4, we develop an algorithm called VANS (Voting Algorithm with Negative Set) to solve this generalized (l,d)-problem under different situations by employing, in additional to voting, other simple but novel techniques, such as filtering, projection with merging and local search. In particular, VANS can solve those challenging (l,d)-problem, such as (9,2), (11,3), (15,5) and (20,7)-problems, when F is empty. Experimental results (Section 5) show that VANS can solve all theoretically solvable generalized (l,d)-problems when $d \leq 20$ and works well on some real data.

2 Calculation the Expected Value $E_b(l, d)$

Let T be the set of t length-n input sequences, each of which contains a variant of M and let F be the set of f length-n input sequences with no variant of M. Assume the occurrence probabilities of 'A', 'C', 'G' and 'T' are equal. Buhler and Tompa [4] studied the limitation of the (l,d)-problem by calculating the expected number $E_t(l, d)$ of length-l strings with at least one d-variant in each sequence in T. Their calculation is described as follows.

Given a length-l string P and a length-l substring σ in the input sequence, the probability that P and σ are at most d symbol substitution apart is

$$p(l, d) = \sum_{i=0}^{d} C_i^l \left(\frac{3}{4}\right)^i \left(\frac{1}{4}\right)^{l-i}$$

The probability that a length-l string P has at least one d-variant in each sequence in T is

$$\left(1 - (1 - p(l, d))^{n-l+1}\right)^t$$

Consider the 4^l possible length-l strings, the expected number of length-l strings with at least one d-variant in each sequence in T is approximately

$$E_t(l, d) = 4^l \left(1 - (1 - p(l, d))^{n-l+1}\right)^t \tag{1}$$

When $E_t(l, d)$ is much larger than 1, that means there are many random length-l strings which have the same characteristics as motif M on T, i.e. have at least one d-variant in each sequence in T. Under this situation, no algorithm can distinguish the motif M from this set of random length-l strings. On the other hand if $E_t(l, d)$ is smaller than 1, the smaller the value of $E_t(l, d)$, the more plausible that the found pattern is M and not an artifact. Thus $E_t(l, d)$ can be used to estimate the amount of information contained in the set of sequences T; the larger is $E_t(l, d)$, the less is the information and vice versa. Given the parameters t, n and l, we can find the range of d such that the (l,d)-problem is unsolvable, i.e. with $E_t(l, d)$ much larger than 1, e.g. $(9, \geq 3)$, $(11, \geq 4)$, $(15, \geq 6)$ and $(20, \geq 8)$-problems.

Similarly, we can estimate the amount of information contained in F by the expected number $E_f(l, d)$ of length-l strings with no variant in any sequences in F, and also the amount of information of both T and F by the expected number $E_b(l, d)$ of length-l strings with at least one variant in each sequence in T and no variant in the sequences of F. If $E_b(l, d)$ is smaller than 1, the generalized (l,d)-problem is theoretically solvable, otherwise, it is unsolvable.

The probability that a length-l string P has no variant in F is

$$\left((1 - p(l, d))^{n-l+1}\right)^f$$

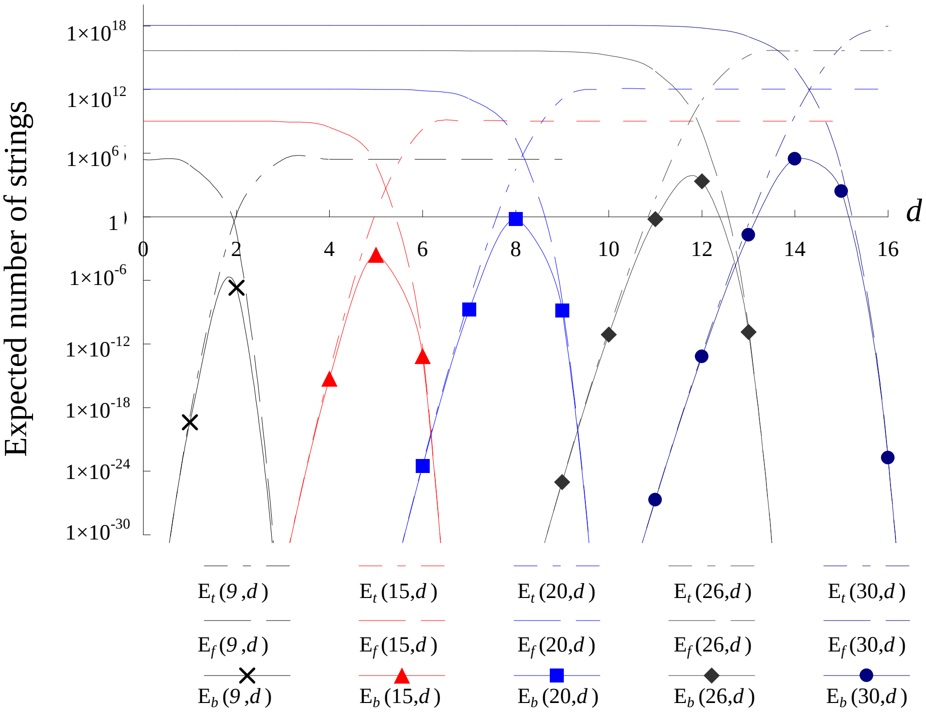

Fig. 1. Expected number of strings against d for different motif length l.

Consider the 4^l possible length-l strings, we have

$$E_f(l,d) = 4^l \left((1-p(l,d))^{n-l+1}\right)^f \tag{2}$$

$$E_b(l,d) = 4^l \left(1-(1-p(l,d))^{n-l+1}\right)^t \left((1-p(l,d))^{n-l+1}\right)^f \tag{3}$$

Figure 1 shows the values of $E_t(l,d)$, $E_f(l,d)$ and $E_b(l,d)$ for different values of l and d when $t = f = 20$ and $n = 600$. We have the following observations which match with our intuition.

1. The (l,d)-problem is easier to solve for a smaller d because T has more information for smaller d. Thus $E_t(l,d)$ increases with d. By the same argument, F has more information for larger d, thus $E_f(l,d)$ decreases with d.
2. The value of $E_b(l,d)$ is always less than $E_t(l,d)$ and $E_f(l,d)$. $E_b(l,d) \approx E_t(l,d)$ when d is small and $E_b(l,d) \approx E_f(l,d)$ when d is large. $E_b(l,d)$ is peaked or the amount of information is the least for some d, $0 < d < l$. It can be shown that $E_b(l,d)$ is maximum when $p(l,d) = p_{thres}$, where

$$p_{thres} = 1 - e^{\frac{\ln f - \ln(t+f)}{n-l+1}}$$

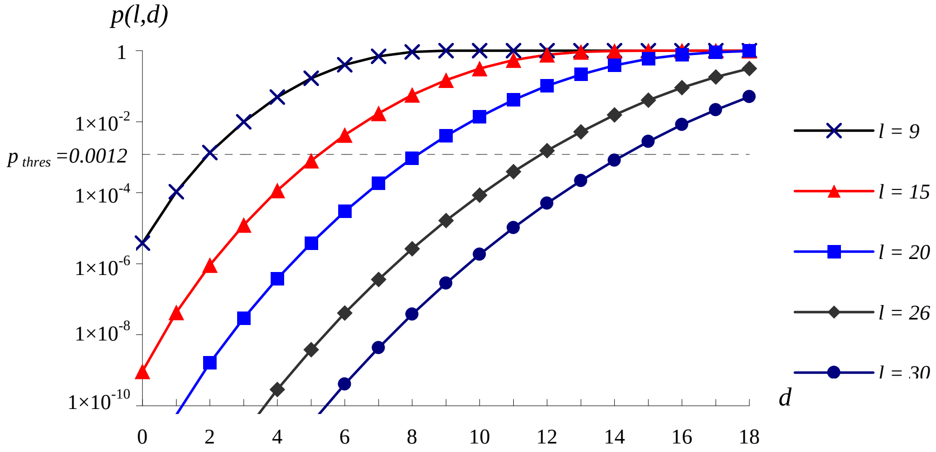

Fig. 2. The value of $p(l, d)$ against d for different motif length l.

Figure 2 shows the value of $p(l, d)$ against d and $p_{thres} = 0.0012$ when $t = f = 20$ and $n = 600$. The intersections between p_{thres} and each curve represent those problems with the least amount of information, e.g. the generalized (9,2), (15,5), (20,8), (26,12) and (30,14)-problems which match the results given in Figure 1.

3. As a result, some previously unsolvable (l,d)-problems, e.g. the (9,3), (11,4) and (15,6)-problem, become solvable after adding the set F. In fact, all (l,d)-problems with $0 \leq d \leq l$ and $l \leq 20$ are solvable. However, when l increases, there are still some generalized (l,d)-problems having a large value of $E_b(l, d)$ (e.g. the generalized (26,12), (30,14) and (30,15)-problems) which means they are theoretically unsolvable.

Buhler and Tompa [4] defined those solvable (l,d)-problems with the largest d as "challenging problems" (i.e. if the (l,d)-problem is a challenging problem, the (l,$d+1$)-problem should be unsolvable). Similarly, we can define a set of "challenging problems" for the generalized planted (l,d)-motif problems. A generalized planted (l,d)-motif problems is *challenging* if it is solvable and either (l,$d-1$) or (l,$d+1$) is unsolvable. For example, the generalized (26,11),(26,13),(30,13) and (30,16)-problems are "challenging problems".

3 Trade off between t and f

Although sequences in both T and F contain information of the motif, the amounts of information in these two sets vary with the values of n and $p(l, d)$. One question we want to know is "If we reduce the number of sequences in T by Δt, how many sequences should we add to F so that the input data retains the same amount of information?" This question can be answered by comparing

Table 1. Trade off between t and f when $n = 600$.

		l			
	R	9	11	15	20
	1	44.16	1118	4.22×10^5	5.35×10^8
	2	0.7547	35.41	1.39×10^4	1.48×10^7
	3	0.0004	0.9763	680.6	6.37×10^5
	4	4.27×10^{-15}	0.0026	40.36	3.74×10^4
d	5	0	5.44×10^{-11}	2.119	2760
	6	0	0	0.0362	237.6
	7	0	0	3.54×10^{-6}	21.46
	8	0	0	4.23×10^{-17}	1.598
	9	0	0	0	0.0463

the value of $E_b(l,d)$ before and after changing the number of input sequences. The amount of information is retained if and only if the new value of $E_b(l,d)$ is no larger than the original value.

$$\begin{aligned}
\frac{4^l\left(1-(1-p(l,d))^{n-l+1}\right)^t\left((1-p(l,d))^{n-l+1}\right)^f}{4^l\left(1-(1-p(l,d))^{n-l+1}\right)^{t-\Delta t}\left((1-p(l,d))^{n-l+1}\right)^{f+\Delta f}} &\geq 1 \\
\Delta t \log\left(1-(1-p(l,d))^{n-l+1}\right) - \Delta f \log\left((1-p(l,d))^{n-l+1}\right) &\geq 1 \\
\Delta f &\geq R\Delta t
\end{aligned}$$

where $R = \frac{\log\left(1-(1-p(l,d))^{n-l+1}\right)}{\log\left((1-p(l,d))^{n-l+1}\right)}$

Since a random sequence containing d-variants of the motif M is independent of another random sequence containing a d-variants of M, the information of M contained in each input sequence is independent of the input size t and f. Therefore, the value of R is independent of the number of sequences t and f in the data set as shown in the above equation.

Table 1 shows the values of R for different values of l and d when $n = 600$. For example, when $l = 20$ and $d = 8$, if we remove one sequence from T, we should add at least $\lceil 1.598 \rceil = 2$ sequences in F to retain the same amount of information in the input data. When d is large, the probability that a random sequence contains a d-variant of the motif M is large while the probability that a random sequence contains no d-variant of the motif M is small, therefore the amount of information in each sequence in T is much less than that in each sequence in F. If we remove a sequence in T, even no sequence is added to F, the total amount of information in the data set remains almost the same(e.g. $l = 9$, $d = 9$). On the other hand, when d is small, the amount of information in each sequence in T is much more than that in each sequence in F. If we remove a sequence in T, many sequences should be added to F in order to retain the same amount of information (e.g. $l = 20$, $d = 1$).

4 Voting Algorithm with Negative Set (VANS)

Our Voting Algorithm with Negative Set (VANS) is based on a simple idea that if a substring σ is a planted d-variant of the motif M, M is also a d-variant of σ. In order to find the motif, each length-l substring in T and F gives one vote to its d-variant. The motif M would receive at least one vote from each sequence in T and no vote from any length-l substring in F. Although the idea used in the Voting algorithms [5] is simple and enumerative, they are so far the fastest algorithms than the other methods based on brute-force [3, 10, 17], finding the maximum clique [15, 18] and heuristic search [1, 4, 8, 9, 11–13] for solving the (l,d)-problem without F. The running times of the brute-force and the clique search algorithms ($O(ntl4^l)$ and $O((nt)^{t+2.376})$ respectively) are much longer than that of the Voting algorithms ($O(ntC^l_d 4^d)$). The brute-force algorithms can only solve the (l,d)-problems with $l \leq 11$ and the clique search algorithms can tackle those problem with small d. Thus, they have difficulties to deal with those challenging (11,3), (15, 5) and (20,7)-problems. Heuristic algorithms can solve the (l,d)-problems for larger l, say $l \leq 20$, but they do not guarantee finding the motifs all the time. The Voting algorithms, [5] on the other hand, can solve not only the challenging (9,2), (11,3), (15,5) and (20,7)-motif problems, but also (30,11) and (40,15)-problems.

As indicated in Figure 1, the generalized (l,d)-problem can be solved for small d and large d when $E_b(l, d)$ is much less than 1. Moreover, since we have shown in Sections 2 and 3, the amount of information in the generalized (l,d)-problem is mainly derived from T when d is small and from F when d is large, our algorithm VANS will first identify a set of candidate motifs by voting from T when d is small and from F when d is large. VANS will then filter out the false candidate motifs by F or T accordingly. Based on the value of d, VANS applies different strategies to solve the generalized (l,d)-problem.

4.1 Voting by sequences in T

Since each length-l substring σ has $C^i_l 3^i$ variants with exactly i substitutions, σ has $\sum_{i=0}^d C^i_i 3^i$ variants. By considering each length-l substring in each sequence in T, the voting process by T takes

$$O\left(nt\sum_{i=0}^{d}(C^l_i 3^i)\right) = O\left(ntC^l_d 4^d\right) \text{ time}$$

The candidate motifs are those variants (length-l strings) which receive at least one vote from each sequence in T. It is shown in [5] that this simple algorithm can solve the challenging (9,2), (11,3), (15,5) and (20,7)-motif problems in time less than a few minutes. The filtering process removes those candidate motifs having variants in F and this filtering step takes $O(nlf)$ time for each candidate motif. Since the expected number of length-l strings with at least one variant in each sequence in T is $E_t(l, d)$, the expected running time is

$$O\left(ntC^l_d 4^d + nlfE_t(l, d)\right)$$

Table 2. Values of d at which voting by T works when $t = f = 20$ and $n = 600$.

l	9	11	15	20	26	30
d	≤ 2	≤ 3	≤ 5	≤ 7	≤ 10	≤ 13
$E_t(l,d)$	≤ 1.6	≤ 4.7	≤ 2.8	$\leq 1.4 \times 10^{-8}$	$\leq 2.1 \times 10^{-11}$	≤ 0.22

This approach works well on T when d or $E_t(l,d)$ is small. Table 2 shows the values of d for which this approach works well.

4.2 Voting by sequences in F

When d increases, both 4^d and $E_t(l,d)$ increase exponentially such that the running time of the voting process becomes unacceptable. Since the amount of information in F is much more than the amount of information in T when d is large, we should focus on F instead of T.

As motif M has no variant in F, M should not be a variant of any length-l substring in F. If each length-l substring in F gives one vote to its variants, we can find a set of candidate motifs which get no vote from any substring in F. The expected number of candidate motifs getting zero vote is $E_f(l,d)$. The filtering process removes those candidate motifs which have a variant in each sequence in T. When d is large, $E_f(l,d) \approx E_b(l,d)$ is small, therefore the expected running time of the filtering process $(O(nltE_f(l,d)))$ should be small too. However, the running time of the voting process by sequences in F, i.e.

$$O\left(nf\sum_{i=0}^{d}(C_i^l 3^i)\right) = O\left(nfC_d^l 4^d\right)$$

might be prohibitively long for large d.

Our approach is to reduce this generalized (l,d)-problem to a smaller generalized (l',d')-problem with $l' < l$, $d' < d$ and d' small enough to be solvable. Let us consider a generalized (l,d)-problem. Since M has no d-variant in F, the length-l' prefix of M and the length-l' suffix of M should not have any d_s-variant in F either, where $d_s = d - (l - l')$. Let

$$d' = \min_{l-d \leq l' \leq l}\{d_s | E_f(l', d_s) \leq 1\}$$

For example, the generalized (20,11)-problem can be reduced to the generalized (14,5)-problem where $E_f(14,5) = 0.0027$. Since $E_f(l',d')$ is small and solvable, the reduced generalized (l', d')-problem is much easier to solve because d' is much smaller. The set of length-l' candidate motifs should contain any length-l' substrings of any length-l candidate motifs for the generalized (l,d)-problem, in particular, the length-l' prefix and length-l' suffix of motif M. If the length-$(2l' - l)$ suffix of a length-l' candidate motif is the same as the length-$(2l' - l)$ prefix of another length-l' candidate motif, we can combine them to form a

Table 3. Values of d for which voting by F works when $t = f = 20$ and $n = 600$.

l	9	11	15	20	26	30
d	≥ 3	≥ 4	≥ 6	≥ 11	≥ 16	≥ 20
d'	1	2	4	5	6	6

length-l candidate motif. It can be shown that any length-l candidate motif for the generalized (l,d)-problem can be formed by combining two candidate length-l' motifs of the generalized (l', d')-problem. Thus the expected number of length-l candidate motifs by merging two candidate (l', d')-motifs is at most $[E_f(l', d') + (l - l')]^2$ and the expected running time is

$$O\left(nfC_{d'}^{l'}4^{d'} + nlt[E_f(l', d') + (l - l')]^2\right)$$

Again this Voting algorithm based on a reduced size generalized (l',d')-problem works very well for large d as long as d' is reasonably small. Table 3 shows the values of d for which this approach works.

4.3 Local search

The voting technique discussed in Section 4.1 and 4.2 works fine for all d when $l \leq 15$. However, when $l > 15$, there are cases that this voting technique will fail. In particular when d is not too small or too large, we cannot solve the generalized (l,d)-problems by voting from T or from F. For example, the generalized (20,9)-problem cannot be solved by voting directly from sequences in T or directly from sequences in F because of the long running time ($c_9^{20}4^9 \approx 4.4 \times 10^{10}$ is a big number). On the other hand, we cannot reduce the generalized (20,9)-problem to another smaller generalized (l',d')-problem with a small value of $E_f(l', d')$.

In order to solve these problems, a local searching method is proposed. The motif M has no d-variant in F and this information in F should be more useful than the information in T for finding M by local search. Assume we have a length-l seed string S. For each length-l neighboring string N, i.e. 1-variant of S, we find the number of d-variants of N occurring in set F. We replace string S by the neighboring string N if N has the least number of d-variants in F and has less d-variants than S in F. We repeat this process several times. If seed S and motif M are within a few symbol substitutions, we may hopefully refine S to M.

The seeds can be generated randomly or selected by voting from T and F. The probabilities that S can be refined to M after k iterations when S and M differ by k symbols for some generalized (l,d)-motif problems were shown in Table 4. It is evident from Table 4 that there is a high probability that we can find motif M from seed S when S and M differ by no more than 5 symbols.

Table 4. Probabilities for refining the seeds successfully. k is the number of symbol substitutions between seed S and motif M.

k	(20,8)	(20,9)	(20,10)
1	0.9895	1	1
2	0.9784	0.9707	0.9394
3	0.9563	0.9282	0.8652
4	0.9074	0.8603	0.7588
5	0.8483	0.4554	0
6	0	0	0

Table 5. VANS' Experimental results on simulated data.

running		l			
time		9	11	15	20
	2	0.4s	2s	201s	9.4s
	3	tends to 0s	9s	240s	10.6s
	4	tends to 0s	1s	382s	11.2s
d	5	0.2s	3s	113.6s	27.1s
	6	-	4s	17m	107.1s
	7	-	-	9s	111.4s
	8	-	-	47s	3.4hr
	9	-	-	-	80m
	10	-	-	-	8.6m
	11	-	-	-	7.7m
	12	-	-	-	-

5 Experimental Results

We have implemented VANS in C++ and tested it on a computer with P4 2.4 GHz CPU and 4GB memory on the simulated and real data. For the simulated data, we picked a length-l motif M randomly and also generated 20 length-600 sequences in F randomly with the 0.25 occurrence probability of 'A', 'C', 'G' and 'T' at each position. Each of these sequences in F would be regenerated if it had a variant of M. Similarly we generated 20 length-600 sequences in T and planted a variant of M at a random position in each of these sequences. For each pair of l and d values, we ran 50 test cases and checked whether our program could discover the motif. Our program discovered the motif in all cases and the average running time is shown in Table 5. Some simulated data is missed (e.g. (9,6), (9,7) and (11,7)) because d is so large that any randomly generated length-600 sequence always contain a variant of any motif.

We have also tested VANS on real biological sequences stored in the public database SCPD. For each set of genes, we chose the 600 bp upstream of the genes as the input sequences in T. We also randomly picked the same number of genes and chose the 600 bp upstream of these genes as the input sequences

Table 6. VANS' Experimental results on real data.

Transcription Factor	Published Motif pattern	Motif Pattern Found
GCR1	CWTCC	CTTCC
GATA	CTTATC	CTTAT
CCBF,SCB,SWI6	CNCGAAA	CGCGAAA
CuRE,MAC1	TTTGCTC	TTTGCTC
GCFAR	CCCGGG	CCCGGG
GCN1	TAATCTAATC	TAATCTAATC

in F. The lengths of the motifs l were the same as those of the published motifs and d was 1. Experimental results are shown in Table 6. VANS could find the motifs for these data sets within one second for each data set.

6 Conclusion

Since the (l,d)-problem has a limitation that no algorithm can discover the motif when d is large, we define the generalized (l,d)-problem which treats those sequences without variants of motif M as additional input. With this extra information, the motif discovering problem with large d becomes theoretically solvable. We also developed the VANS algorithm to solve the generalized (l,d)-problem. Experimental results showed that VANS performed well on most problem instances including the challenging (9,2), (11,3), (15,5), (20,7)-motif problem when F is empty.

The challenging generalized (l,d)-problems for $l > 20$, e.g. (26,11) and (26,13), remain unsolvable because of its long running time. Local search might work if we can reduce the number of seeds by generating "good" seeds efficiently.

References

1. Bailey, T., Charles Elkan, C.: Unsupervised learning of multiple motifs in biopolymers using expectation maximization. Machine Learning. **21** (1995) 51–80
2. Barash, Y., Bejerano, G., Friedman, N.: A Simple Hyper-Geometric Approach for Discovering Putative Transcription Factor Binding Sites. Workshop on Algorithms in Bioinformatics WABI **1** (2001) 278–293
3. Brazma, A., Jonassen, I., Eidhammer, I., Gilbert, D.: Approaches to the automatic discovery of patterns in biosequences. Jour. Comp. Biol. **5** (1998) 279–305
4. Buhler, J., Tompa, M.: Finding motifs using random projections. Research in Computational Molecular Biology RECOMB **1** (2001) 69–76
5. Chin, F., Leung, H.: Voting Algorithms for Discovering Long Motifs. Asia-Pacific Bioinformatics Conference APBC **3** (2005) 261–271
6. Chin, F., Leung, H., Yiu, S.M., Lam, T.W., Rosenfeld, R., Tsang, W.W., Smith, D., Jiang, Y.: Finding Motifs for Insufficient Number of Sequences with Strong Binding to Transcription Factor. Research in Computational Molecular Biology RECOMB **4** (2004) 125–132

7. Chin, F., Leung, H., Yiu, S.M., Rosenfeld, R., Tsang, W.W.: Finding Motifs with Insufficient Number of Strong Binding Sites. Jour. Comp. Biol. (to appear)
8. Fraenkel, Y., Mandel, Y., Friedberg, D., Margalit. H.: Identification of common motifs in unaligned dna sequences: application to Escherichia coli Lrp regulon. Bioinformatics **11** (1995) 379–387
9. Gelfand, M., Koonin, E., Mironov, A.: Prediction of transcription regulatory sites in archaea by a comparative genomic approach. Nucl. Acids Res. **28** (2000) 695–705
10. J. van Helden, B. Andre, Vides, J.C.: Extracting regulatory sites from the upstream region of yeast genes by computational analysis of oligonucleotide frequencies. Journal of Molecular Biology **281(5)** (1998) 827–842
11. Hertz, G.Z., Stormo, G.D.: Identification of consensus patterns in unaligned dna and protein sequences: a large-deviation statistical basis for penalizing gaps. International Conference on Bioinformatics and Genome Research **3** (1995) 201–216
12. Lawrence, C., Altschul, S., Boguski, M., Liu, J., Neuwald, A., Wootton, J.: Detecting subtule sequence signals: a Gibbs sampling strategy for multiple alignment. Science **262** (1993) 208–214
13. Lawrence, C., Reilly, A.: An expectation maximization (em) algorithm for the identification and characterization of common sites in unaligned biopolymer sequences. Proteins: Structure, Function and Genetics **7** (1990) 41–51
14. Leung, H., Chin, F.: Finding Exact Optimal Motif in Matrix Representation by Partitioning. European Conference on Computational Biology ECCB (2005) (to appear)
15. Liang, S.: cWINNOWER Algorithm for Finding Fuzzy DNA Motifs. Computer Society Bioinformatics Conference **2** (2003) 260–265
16. Marsan L., Sagot, M.F.: Algorithms for extracting structured motifs using a suffix tree with an application to promoter and regulatory site consensus identification. Jour. Comp. Biol. **7(3-4)** (2000) 345–362
17. Pesole, G., Prunella, N., Liuni, S., Attimonelli, M., Saccone, C.: Wordup: an efficient algorithm for discovering statistically significant patterns in dna sequences. Nucl. Acids. Res. **20(11)** (1992) 2871–2875
18. Pevzner, P., Sze, S.H.: Combinatorial approaches to finding subtle signals in dna sequences. International Conference on Intelligent Systems for Molecular Biology **8** (2000) 269–278
19. Sagot, M.F.: Spelling approximate repeated or common motifs using a suffix tree. Latin'98: Theoretical informatics, Lecture Notes in Computer Science **1380** (1998) 111–127
20. Sinha, S.: Discriminative motifs. Jour. Comp. Biol. **10** (2003) 599–616
21. Zhu, J., Zhang, M.: SCPD: a promoter database of the yeast Saccha-romyces cerevisiae. http://cgsigma.cshl.org/jian/ Bioinformatics **15** (1999) 563–577