EDAM: AN EFFICIENT CLIQUE DISCOVERY ALGORITHM WITH FREQUENCY TRANSFORMATION FOR FINDING MOTIFS

YIFEI MA GUOREN WANG
YONGGUANG LI AND YUEHAI ZHAO *
College of Information Science and Engineering
Northeastern University, Shenyang, China
E-mail: wanggr@mail.neu.edu.cn

Finding motifs in DNA sequences plays an important role in deciphering transcriptional regulatory mechanisms and drug target identification. In this paper, we propose an efficient algorithm, EDAM, for finding motifs based on frequency transformation and Minimum Bounding Rectangle (MBR) techniques. It works in three phases, *frequency transformation*, *MBR-clique searching* and *motif discovery*. In *frequency transformation*, EDAM divides the sample sequences into a set of substrings by sliding windows, then transforms them to frequency vectors which are stored in MBRs. In *MBR-clique searching*, based on the frequency distance theorems EDAM searches for MBR-cliques used for motif discovery. In *motif discovery*, EDAM discovers larger cliques by extending smaller cliques with their neighbors. To accelerate the clique discovery, we propose a range query facility to avoid unnecessary computations for clique extension. The experimental results illustrate that EDAM well solves the running time bottleneck of the motif discovery problem in large DNA database.

1. Introduction

In the process of gene expression, one or more proteins, called transcription factors have to bind to several specific regions named binding sites. These sites typically have a similar short DNA sequence pattern which is simply referred to *motif*. According to the traits of motif, the motif discovery problem is to find a pattern in sample sequences whose length is l, and in every sample sequence there is a pattern which has no more than d mismatches with this motif pattern [1]. The identification of short sequence motifs, such as transcription factor binding sites, is at the center of the transcriptional regulation understanding. The functional sites are constrained to contain motifs, since their changes will disrupt regulation, which is detrimental to the organism [2, 3].

Several motif-based methods have been proposed to count the total number of motifs rather than sequences, and construct a similar contingency table [4]. Some other methods including Consensus [5], Gibbs Sampler [6] and ANN-Spec [7] for multiple local alignment have been employed to resolve the identification of motifs problem. In many cases where motifs have been experimentally determined, these algorithms have been shown to yield the known motifs, indicating that such methods can discover unknown motifs from a

*This work was supported by the National Natural Science Foundation of China (Grant No. 60273079 and 60473074).

collection of sequences believed to be implanted motifs. Brazma et al. algorithms [8] find and analyze combinations of motif that occur in the upstream regions of genes in the yeast genome. These algorithms can identify all the motifs that satisfy given parameters with respect to a given sample sequences. However, they perform an exhaustive search through all 4^l l-letter patterns and find the high-scoring patterns, thus the algorithms become impractical for $l > 10$. Tompa raised the problem of Brazma, and improved this approach for longer patterns. One way around this problem is to limit the search spaces on the patterns appearing in the sample sequences [9–11].

WINNOWER is an outstanding algorithm for finding motifs in respect that it proposes a clique discovery approach to finding global optimal results [12]. WINNOWER indicates that the motif discovery problem is similar to the clique discovery problem. A clique is a set of nodes in a graph, each of which is connected to the others in this set. The sample sequences are divided into a set of substrings which are represented by nodes. If two substrings are similar, there will be an edge connecting them. Thereby, a motif can be taken as a clique in which different nodes are from different sample sequences. For a set of sample sequences $S = \{s_1, s_2, \ldots, s_q\}$, WINNOWER constructs a graph to find the cliques which represent the motifs in S. For each substring s_{ij} from position j to position $j + l - 1$ in sequence s_i , the algorithm constructs a node representing it. Two node s_{ij} and s_{pq} are connected by an edge, if s_{ij} and s_{pq} are similar ($i \neq p$). A q-clique in a graph is a q-nodes set, in which all the pair nodes are connected. Thereby, (l, d)-motif is a clique with size q in the graph. Since most of edges in the graph cannot make up a clique, called spurious edges, WINNOWER prunes some of these spurious edges to speed up searching. Suppose C is a clique, node n is a neighbor of C if and only if n connects to each node in C. If a clique has at least one neighbor, it is *extendable*. If an edge does not belong to any extendable clique of size q, it is *spurious*. WINNOWER prunes the spurious edges based on the observation that every edge in a q-clique belongs to at least $\binom{q}{k}$ extendable cliques of size k.

Although WINNOWER is a typical algorithm for motif discovery, it still has two main problems. (1) For the case that there are a few motifs in the sample sequences, so only a few cliques and edges in the graph. However, most of running time is spent to compute similarity of pairwise nodes during the construction of the graph. Therefore, most of similarity computations are unnecessary. (2) For the case that numerous motifs exist in the sample sequences, the graph will conclude numerous cliques and edges. In this case, WINNOWER needs huge spaces to record the edges. The space requirement of WINNOWER is often a bottleneck to find motifs in large sample sequences.

In this paper, we present an efficient clique discovery algorithm EDAM based on frequency transformation and MBRs. It works in three phases, *frequency transformation*, *MBR-clique searching* and *motif discovery*. In *frequency transformation*, EDAM divides the sample sequences into a set of substrings by sliding windows, then transforms them to frequency vectors which are stored in MBRs. In *MBR-clique searching*, based on the frequency distance theorems EDAM searches for MBR-cliques used for motif discovery. In *motif discovery*, EDAM discovers larger cliques by extending smaller cliques with their

neighbors. To accelerate the clique discovery, we propose a range query facility to avoid unnecessary computations for clique extension. EDAM has the following advantages over WINNOWER. (1) EDAM avoids a lot of unnecessary similarity computations by MBR-cliques searching, since it only computes the similarity of nodes within the same MBR-clique. (2) Since EDAM uses MBRs to store similar substrings, it saves storage space compared with WINNOWER.

The rest of this paper is organized as follows. Section 2 formally defines the motif discovery problem. Section 3 describes the algorithm EDAM in detail. Section 4 gives an analysis of the time and space complexity of EDAM and WINNOWER. Section 5 shows the experimental results and compares the performance of EDAM with WINNOWER. Finally, Section 6 concludes this paper.

2. Problem Description

Known regulatory motifs are short, sometimes degenerate and appear frequently throughout the sample sequences. Additionally, Protein-binding DNA motifs often contain ambiguous nucleotides, which can have more than one equivalent nucleotide, so the problem is to discover the following motifs in a sample sequences [13].

Definition 1. *Motif discovery.* Given a sample sequences $S = \{s_1, s_2, \ldots, s_q\}$, the motif pattern length l and the maximum hamming distances between the motif occurrences d. Then the (l, d)-motif discovery problem is defined as finding such l-length pattern *m*.

$$(\forall s_i \in S)(\exists sub \in s_i)(Length(sub) = l \wedge hd(m, sub) \leq d) \tag{1}$$

Finding motifs, as WINNOWER demonstrated, is similar to the clique discovery problem. If we choose the hamming distance between a motif and any its occurrence is at most d, $2d$ is the longest acceptable distance between any two occurrences presenting a same motif. Therefore, a clique discovery problem corresponding to (l, d)-motif can be defined as follows.

Definition 2. *Clique discovery.* Given a sample sequences $S = \{s_1, s_2, \ldots, s_q\}$ and a (l,d)-motif discovery problem. Any l-length node set C is called a q-clique if and only if

(1) In C, different substrings come from different sample sequences.
(2) For any pair substrings s_i and s_j $(i \neq j)$ in C, $hd(s_i, s_j) \leq 2d$.

3. EDAM

EDAM is a different algorithm for finding motifs in sample sequences, and it has some advantages over WINNOWER. EDAM avoids a lot of unnecessary similarity computations by MBR-cliques searching, since it only computes the similarity of nodes within the same MBR-clique. Moreover, EDAM uses MBRs to store similar substrings, it saves storage space compared with WINNOWER.

3.1. *Frequency Transformation*

In Frequency transformation, EDAM divides the sample sequences into a series of substrings and transforms these substrings into frequency vectors that are stored in MBRs. Before we explain Frequency transformation, we first introduce frequency vector and MBR.

The frequency vector indicates the number of each kind of nucleotide in the DNA sequences. Since DNA sequences are composed of 4 different nucleotides, they always are treat as strings with the alphabet $\sum$ = {A,C,G,T}. EDAM transforms substrings divided from the sample sequences to a 4-dimensional vectors, and the value in every dimension indicates the number of one kind of nucleotide in the substring [14,15]. For example, given a substring $s = TAGCCGAA$, the frequency vector $f(s) = [3, 2, 2, 1]$.

Definition 3. *Frequency vector.* Given s be a substring and the alphabet $\sum = \{\alpha_1, \alpha_2, \ldots, \alpha_\sigma\}$, f_i indicates the number of i^{th} nucleotide in $\sum$, then the frequency vector: $f(s) = [f_1, f_2, \ldots, f_\sigma]$

Minimum Bounding Rectangle(MBR) represents a subspace in the multidimensional space. Each dimension of MBR has a maximum and a minimum, which bound the subspace. The frequency vectors stored in the MBR are restricted in its subspace. In other words, for each frequency vector $f = [f_1, f_2, \ldots, f_\sigma]$ in a MBR $mbr = [(min_1, max_1), (min_2, max_2), \ldots, (min_\sigma, max_\sigma)]$, the value f_i of any dimension $(1 \leq i \leq \sigma)$ must be in the interval $[min_i, max_i]$. In this way, the similar vectors representing similar substrings definitely are in an identical MBR or adjacent MBRs.

Frequency vector and MBR are two useful definitions for frequency transformation. In frequency transformation, EDAM reads only one sequence s_i of the sample $S = \{s_1, s_2, \ldots, s_q\}$ each time and sets up the MBRs for s_i. These MBRs divide the multidimensional space into different subspace (e.g. the multidimensional space is divided into subspaces by a grid using dichotomy). For each substring s_{ij} from position j to position $j + l - 1$ in sequence s_i, EDAM transforms it to the frequency vector $f(s_{ij})$ and stores $f(s_{ij})$ in the proper MBR.

3.2. *MBR-clique Searching*

Most of the frequency vectors in the MBRs cannot make up any clique, thus, how to avoid finding cliques in these frequency vectors is one of the foundational problems. In this section, we suggest using *MBR-clique searching* to resolve this problem base on the fact that the vectors in a clique are stored in the adjacent MBRs. The similarity of a pair substrings is generally measured by hamming distance, but hamming distance requires to count the number of mismatches, thus, it is difficult to calculate hamming distance by frequency vectors. Here, we suggest using frequency distance as the lower bourn of hamming distance.

Definition 4. *Frequency distance.* The summation of frequency differences (only positive) on every dimension in $\sum = \{\alpha_1, \alpha_2, \ldots, \alpha_\sigma\}$ of the given substrings s_1, s_2. $f_i(s_1)$ $f_i(s_2)$ denotes i^{th} dimension's value of s_1 and s_2 respectively. The frequency distance between s_1

and s_2 is defined as follow.

$$fd(s_1, s_2) = \sum_{i=1}^{\sigma} \begin{cases} f_i(s_1) - f_i(s_2) & \text{if } f_i(s_1) - f_i(s_2) \geq 0 \\ 0 & \text{else} \end{cases} \quad (2)$$

Suppose the hamming distance of a pair substrings s_1, s_2 is d, it means that if s_1 is transformed to s_2, based on that one mismatch needs one substitution, s_1 requires d substitution operations. According to the definition of frequency vector, d substitutions at most make d differences on frequency vectors.

Theorem 1. *Suppose s_1 and s_2 are two substrings. The frequency distance between s_1 and s_2 is a lower bound on their hamming distance.*

$$hd(s_1, s_2) \geq fd(s_1, s_2) \quad (3)$$

Since the clique in EDAM is a set of similar vectors, and these vectors are stored in adjacent MBRs, we estimate the distances between vectors by the distances between vectors and MBRs.

Theorem 2. *Suppose mbr is a MBR, v is a vector, not in mbr, then for any vector m in mbr, the frequency distance between m and v is no more than the minimum frequency distance between v and the bounding of mbr.*

$$fd(m, v) \geq fd(v, mbr) \quad (4)$$

For the vectors are stored in MBRs, we suggest using the MBR distance to estimate the distances between the vectors in them.

Definition 5. *MBR distance.* Suppose mbr_1 and mbr_2 are two MBRs, $min_i(mbr_j)$ and $max_i(mbr_j)$ are the minimum and the maximum of i^{th} dimension in mbr_j. The frequency distance between mbr_1 and mbr_2 is the minimum frequency distance between the their bounds, it is defined as follow.

$$fd(mbr_1, mbr_2) = \sum_{i=1}^{\sigma} \begin{cases} min_i(mbr_1) - max_i(mbr_2) & \text{if } min_i(mbr_1) \geq max_i(mbr_2) \\ min_i(mbr_2) - max_i(mbr_1) & \text{if } min_i(mbr_2) \geq max_i(mbr_1) \\ 0 & \text{else} \end{cases} \quad (5)$$

Theorem 3. *Suppose mbr_1 and mbr_2 are two MBRs, v_1 and v_2 are frequency vectors that are stored in mbr_1 and mbr_2 respectively. The distance between mbr_1 and mbr_2 is the lower bound on the distance between v_1 and v_2.*

$$fd(v_1, v_2) \geq fd(mbr_1, mbr_2) \quad (6)$$

According to the clique definition and Theorem 3, we suggest using MBR-clique searching to record the MBRs which make up cliques, and then finding motifs in these MBR-cliques. A MBR-clique *MC* is a set of MBRs, the frequency distance between each pair of MBRs in MC does not excess the threshold.

Definition 6. *MBR-clique.* Given the sample sequences $S = \{s_1, s_2, \ldots, s_q\}$ and a (l,d)-motif discovery problem. A q-MBR set *MC* is called a MBR-clique if and only if

(1) In MC, different MBRs come from different sample sequences.

(2) For each pair of MBRs mbr_i and mbr_j $(i \neq j)$ in MC, $hd(mbr_i, mbr_j) \leq 2d$.

EDAM only searches for the cliques in MBR-cliques to speed up the discovery. The MBR-clique searching algorithm is illustrated in Algorithm 1. Before searching for MBR-cliques, (step 1) EDAM scans all the MBRs, (step 2 and 3) and filters out the MBRs which is empty. (step 4 and 5) EDAM searches for the MBRs that store the frequency vectors from the first sample sequence s_1, (step 6) initializes them as the 1-MBR-cliques, then extends these 1-MBR-cliques to q-MBR-cliques. (step 7) EDAM discovers motifs in the MBR-cliques.

Algorithm 1 MBRClique()

Input: the MBR set *smbr* that all the MBR in
Output: all the MBR-clique
1: FOR $\forall\ mbr \in smbr$
2: IF *mbr* is empty
3: filter out *mbr* from *smbr*
4: FOR $\forall\ mbr \in smbr$
5: IF $mbr.sequence = 1$
6: extending the 1-MBR-clique MC_1 to a q-MBR-clique C_q
7: searching for the motifs in C_q

For the motif pattern generally is short, the number of MBR is not large compared with the number of frequency vectors, and MBR-clique searching only takes a small part of the total running time for EDAM.

3.3. *Motif Discovery*

In this section, we illustrate the algorithm for finding motifs in the MBR-cliques found by MBR-clique searching. To discover the cliques representing the motifs, we employ a simple idea extending a known $(k-1)$-clique with its neighbor to a k-clique.

The motifs discovery problem implies us that for every sample sequence s_i, there is one and only one vector from s_i in the clique representing a motif. Following this clue, EDAM first finds a known clique $C = \{v_1, v_2, \ldots, v_k\}$ $(k \leq q)$, and every vector v_i $(1 \leq i \leq k)$ in C representing a substring from the sequence s_i, then searches for a neighbor v which is from the sequence s_{k+1} to extend C. Since any single vector makes up a 1-clique, in this way, EDAM can iteratively extend the 1-cliques made up of a vector from s_1 to q-cliques composed of vectors from every sample sequence.

Since the neighbor v must be similar to all the vectors in C, the extension has to calculate totally $e(k-1)$ times hamming distance (there are e neighbors). These calculations for extension cause a running time bottleneck for applications. To resolve this problem EDAM sets a signature on every neighbor ne of $C^{'} = \{v_1, v_2, \ldots, v_{k-1}\}$, if $hd(ne, v_k) \leq d$, ne is also a neighbor of $C = \{v_1, v_2, \ldots, v_{k-1}, v_k\}$. EDAM can set the signature iteratively, because every vector is a neighbor of 0-clique.

Theorem 4. *Cliques combination property.* Given a hamming distance d, and two k-cliques $C_1 = \{v_1, v_2, \ldots, v_{k-1}, v^{'}\}$ and $C_2 = \{v_1, v_2, \ldots, v_{k-1}, v^{''}\}$

$$if \quad hd(v^{'}, v^{''}) \leq d \quad then \quad C_3 = \{v_1, v_2, \ldots, v_{k-1}, v^{'}, v^{''}\} \tag{7}$$

After the neighbors of the k-cliques have been found, we will extend the $(k+1)$-cliques to discovery larger motifs. For only a few of the new discovered $(k+1)$-cliques can be extended to q-cliques, it is necessary to prune the cliques named *spurious cliques* which can not be extended to q-cliques. According to the clique definition, if the $(k+1)$-clique $C_{k+1} = \{v_1, v_2, \ldots, v_{k+1}\}$ can be extended to q-clique $C_q = \{v_1, v_2, \ldots, v_q\}$, the neighbor v_{k+1} of C_k must be similar to every vector $v_i (k+2 \leq i \leq q)$. Thus, we use a range query based on the Theorem 2 to prune some spurious cliques. There are two important parameters v and r in range query, v is the query vector and r is the range radius. A range query $R(v, r)$ is to record the MBRs whose distances to v are within r. To prune some spurious cliques, we set the neighbor v_{k+1} as the query vector and the hamming distance $2d$ as the radius, then propose a rang query $R(v_{k+1}, 2d)$ in the MBR-clique. Based on Theorem 2, if any MBR in the MBR-clique is outside the range query, then C_{k+1} is a spurious clique, thus, EDAM prunes it to avoid unnecessary clique discoveries.

We describe the algorithm on the clique extension for finding motifs illustrated by Algorithm 2. Since every vector is a neighbor of 0-clique,EDAM initializes $vector_1$ from s_1 as the neighbor of 0-clique, and initializes the MBR-clique in which $vector_1$ stored as the query MBR-clique. (step 1-2) For every vector v in the query MBR-clique $mbrClique$, the algorithm calculates the hamming distance between v and the neighbor $vector_k$. If there is no vector in the known clique C_k that comes from same sequence as v does, moreover $hd(v, vector_k) \leq 2d$ and the signature on v has indicated v is a neighbor of C_{k-1}, then it is a neighbor of the clique C_k. Thus, (step 3) the algorithm resets a signature on v. (step 4) After every neighbor has been set signatures, EDAM extends C_k for finding q-cliques. (step 5) if v comes from the sequence next to $vector_k$ does, $C_{k+1} = C_k \cup \{v\}$ makes up a known $(k+1)$-clique. (step 6) If the C_{k+1} is a q-clique, (step 7) all the vectors in C_{k+1} that represent the occurrences of a motif are recorded. (step 8) If C_{k+1} is not spurious, (step 9) the algorithm extends C_{k+1} for further clique discovery. (step 10) After v is extended, if v has been set a signature, (step 11) the algorithm resets the signature on v.

Algorithm 2 searchMotifs()

Input: a known k-clique $C_k = \{vector_1, vector_2, \ldots, vector_k\}$; a neighbor $vector_k$; the query MBR-clique $mbrClique$;
Output: all the motifs

1: FOR $\forall\, v \in mbr$
2: IF $hd(v, vector_k) \leq 2d$ and $v.sequence > vector_k.sequence$ and $v.signature = vector_k.sequence - 1$
3: $v.signature = vector_k.sequence$
4: FOR $\forall\, v \in mbr$
5: IF $v.sequence = vector_k.sequence + 1$ and $v.signature = vector_k.sequence$
6: IF $\{vector_1, \ldots, vector_k, v\}$ has q vectors
7: record all the vectors $\{vector_1, vector_2, \ldots, v\}$, which represents a motif.
8: ELSE IF RangeQuery(v, mbr)=false
9: searchMotifs(v, mbr).
10: IF $v.signature = vector_k.sequence$
11: $v.signature = vector_k.sequence - 1$.

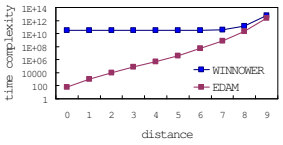
4. Analysis

In this section, we give an analysis of the time and space complexity of EDAM and WINNOWER.

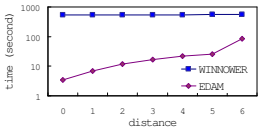
4.1. *Space complexity*

For the sample sequences $S = \{s_1, s_2, \ldots, s_q\}$, there are about $N = \sum_{j=1}^{q} len_j$ subsequences. The spaces for WINNOWER are primarily composed by two parts: nodes and edges in the graph. For WINNOWER constructs a node for each valid subsequence in the sample sequences, it needs $O(N)$ nodes and $p_d O(N^2)$ edges, thereby, the WINNOWER's space complexity is $O(N^2)$. The spaces for EDAM are also composed by two parts: frequency vectors and MBRs. EDAM transforms subsequences divided for the sample sequences into the frequency vectors, thereby, there are $O(N)$ frequency vectors. If the MBR width is w, for every sequence $s_i \in S$, EDAM at most constructs $(l/w)^4$ MBRs. Since $l \ll N$, the EDAM's space complexity is approximately $O(N)$.

4.2. *Time complexity*



(a) The time complexity analysis of motif discovery computed by MATLAB.



(b) The practical performances of EDAM and WINNOWER in an identical sample.

Figure 1. The contrast between the time complexity analysis and the performance of EDAM of motif discovery in 15 4KB length sequences with increasing distance, fixed pattern length $l = 15$.

Given two l-length subsequences s_1 and s_2, the probability p_d that $hd(s_1, s_2) \leq d$ equals $\sum_{i=0}^{d} \binom{l}{i} \left(\frac{1}{4}\right)^{l-i} \left(\frac{3}{4}\right)^{i}$. If the similarity of the vectors in q-clique is completely independent ($d = l$), the probability c_q that q vectors make up a q-clique equals $p_d^{\binom{q}{2}}$. In contrast, the similarity of the vectors in q-clique is completely dependent ($d = 0$), c_q equals p_d^{q-1}. Suppose the length of the j^{th} sample sequence is len_j, the expected number of q-cliques discovered in the sample sequences is in the interval $[p_d^{\binom{q}{2}} \prod_{j=1}^{q} len_j \ , p_d^{q-1} \prod_{j=1}^{q} len_j]$. The running time of WINNOWER is primarily spent in two parts: graph construction and clique discovery. For the graph construction has to compute the hamming distances of every pair nodes, it requires $0.5 \ l \ N^2$ calculations. Additionally, since each node in the graph has $p_d \ N$ edges, WINNOWER requires $\sum_{k=1}^{q}(c_k \ \prod_{j=1}^{k} len_j \ p_d \ N)$ calculations to find all the q-clique. In sum the time complexity of WINNOWER is $0.5 \ l \ N^2 + \sum_{k=1}^{q}(c_k \ \prod_{j=1}^{k} len_j \ p_d \ N)$.

Although EDAM works in three phases, the frequency transformation and MBR-clique searching only take small part of the total running time(primarily because the number of MBR is much smaller than the number of vectors). In motif discovery, each k-clique has potential $p_{d+w}\ len_j$ vectors for further extension. Consequently, the time complexity is $\sum_{k=1}^{q}(c_k\ \prod_{j=1}^{k} len_j\ p_{d+w}\ len_j\ l)$. Figure 4 illustrates the time complexity analysis and the practically performances of WINNOWER and EDAM with the same parameters.

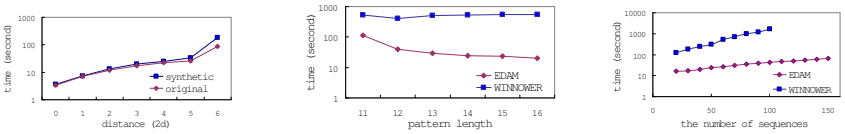
5. Experimental Results

In this section, we illustrate EDAM's efficiency for discovering motifs in the sample sequences. The experiments were performed on a PC with 2.6GHz P4 CPU and 512MB memory, programmed in JAVA. The sample sequences originated from human gene sequences section (chr22) and the MBR width is 2. We do not present the experimental results in terms of some algorithms used the performance coefficients to the known motifs, which mainly measure the algorithm results accuracy. Because EDAM uses the same model (clique) as WINNOWER for finding motifs, and both of the two algorithms can find out all the global optimal results, thus, the performance coefficients of the two algorithms are the same. Beside above problem, running time is another big bottleneck, especially when we discover motifs in large DNA database, so in this paper we compare the running time of the two algorithms instead.

In Figure 1(b) WINNOWER took up a steady running time for all distances, for the running time was mostly for the graph construction. The performances of EDAM occurred a sharp rise as soon as the distance exceeded a percentage of pattern length. The sharp rise reveals that the techniques to avoid necessary computations in EDAM are efficient for short distances, but breaks for the liberal distances. In Figure 2(a) EDAM discovered $(2, 15)$-motifs in two different samples: original sample and synthetic sample. The original sample was from human gene sequences section. The synthetic sample sequences were implanted a serious of rational motifs with randomly distributed background, thus, the number of the results in synthetic sample emerged an outstanding increment over original sample. Due to the effect of the number of results, the running time in the synthetic sample was over in original sample for all the distances.Due to avoiding unnecessary computations, in Figure 2(b) EDAM's merit was obvious for the pattern length increment, whereas WINNOWER did not appear any distinct change. With the increment of the number of sequences in these sample, in Figure 2(c) the running time of WINNOWER for $(2, 15)$-motifs occurred a distinct rise, however, the running time of EDAM rose smoothly (we stopped the tests whose running time were more than one hour). Figure 2 imply that in the tests with a few number of the results or low distance-pattern ratio, EDAM performs better than WINNOWER. It is because EDAM well approximates hamming distance by frequency distance and avoids most of the unnecessary computations. On the other side, the merits are not so distinct.

6. Conclusions and Discussions

Although the motif discovery problem has a long history, it is still far away from being resolved. The well-known algorithm WINNOWER shows better performances than other



(a) EDAM discovered motifs in two different samples: original sample and synthetic sample, the figure illustrates the running time in original sample and synthetic sample.

(b) The performances of EDAM and WINNOWER in an identical sample with fix the distance d and different pattern length l from 11 to 16.

(c) The performances of EDAM and WINNOWER in a series of samples. These samples were composed of fix length sequences, and the number of sequences increased from 30 to 150.

Figure 2. The performances of WINNOWER and EDAM with different parameters.

algorithms, but it still has some shortages. In this paper, we suggest a unique algorithm EDAM using frequency transformation and MBR techniques to solve the running time problem of WINNOWER. The experimental results indicate that EDAM is more efficient than WINNOWER for motif discovery. Although EDAM shows excellent performances, further improvements are still necessary.

References

1. X. Dong, S. Y. Sung, W. Sung and C. L. Tan. Constraint based method for finding motifs in DNA sequences. *In proc. BIBE'04, 2004.*
2. M.Lapidot and Y. Pilpel. Comprehensive quantitative analyses of the effects of promoter sequence elements on mRNA transcription. *Nucleic Acids Research, Vol. 31, No.13,3824C3828,2003*
3. J. Shapiro and D. Brutlag. FoldMiner: Structural motif discovery using an improved superposition algorithm. *Protein Science, 13:278C294,2004.*
4. R .Sharan, I.Ovcharenko, A.Ben-Hur, and R.M. Karp. CREME: a frame work for identifying cis-regulatory modules in human-mouse conserved segments. *Bioinformatics, 19 (Suppl1), I283-I291,2003.*
5. G. Hertz, and G. Stormo. Identifying DNA and protein patterns with statistically significant alignments of multiple sequences. *Bioinformatics 14:563-577, 1999.*
6. M. Jerrum. Large cliques elude the Metropolisprocess. *Random Structures and Algorithms 3(4):347-359,1992*
7. C.T.Workman and G.D. Stormo. ANN-Spec:a method for discovering transcription factor binding sites with improved specificity. *Pac.Symp.Biocomput.,467-478,2000.*
8. A. Brazma, I. Jonassen, I. Eidhammer and D. Gilbert. Approaches to the automatic discovery of patterns in biosequences. *Journal of Computational Biology 5:278-305, 1998.*
9. M. Li, B.Ma and L. Wang. Finding similar regions in many strings. *In proceedings of the 31st ACM Annual Symposium on Theory of computing.473-482, 1999.*
10. X. Liu , D.L. Brutlag and J. S. Liu. BioProspector:discovering conserved DNA motifs in upstream regulatory regions of co-expressed genes. *Pac Symp Biocomput 127-38, 2001.*
11. X. Liu, D.L. Brutlag and J. S. Liu. An algorithm for finding protein-DNA binding sites with applications to chromatin-immunoprecipitation microarray experiments. *Nat biotechnol 835-9, 2002.*
12. P. A. Pevzner and S. H. Sze. Combinatorial approaches to finding subtle signals in DNA sequences. *In Proc. ISMB'00, 2000.*
13. R. V. Satya, A. Mukherjee and U. Ranga. A Pattern Matching Algorithm for Codon Optimization and CpG Motif engineering in DNA Expression Vectors. *In proc. CSB'03, 2003.*
14. M. Garofalakis and A. Kumar. Deterministic Wavelet Thresholding for Maximum-Error Metrics. *In proc. PODS'04, 2004.*
15. T. Kahveci and A. K. Singh. An Efficient Index Structure for String Databases. *In proc. VLDB'01 351-360, 2001.*