A Cluster Refinement Algorithm For Motif Discovery

Gang Li, Tak-Ming Chan, Kwong-Sak Leung and Kin-Hong Lee

***Abstract*—Finding Transcription Factor Binding Sites, i.e., motif discovery, is crucial for understanding the gene regulatory relationship. Motifs are weakly conserved and motif discovery is a NP-hard problem. We propose a new approach called Cluster Refinement Algorithm for Motif Discovery (CRMD). CRMD employs a flexible statistical motif model allowing a variable number of motifs and motif instances. CRMD first uses a novel entropy-based clustering to find complete and good starting candidate motifs from the DNA sequences. CRMD then employs an effective greedy refinement to search for optimal motifs from the candidate motifs. The refinement is fast, and it changes the number of motif instances based on the adaptive thresholds. The performance of CRMD is further enhanced if the problem has one occurrence of motif instance per sequence. Using an appropriate similarity test of motifs, CRMD is also able to find multiple motifs. CRMD has been tested extensively on synthetic and real datasets. The experimental results verify that CRMD usually outperforms four other state-of-the-art algorithms in terms of the qualities of the solutions with competitive computing time. It finds a good balance between finding true motif instances and screening false motif instances, and is robust on problems of various levels of difficulty.**

***Index Terms*—Transcription Factor Binding Site, Motif Discovery**

I. Introduction

Transcription factor binding sites (TFBSs) are crucial in gene regulation, the understanding of which is a central problem in contemporary biology. Finding the pattern of TFBSs in DNA sequences, i.e. motif discovery, is thus important for uncovering the underlying regulatory relationship and understanding the evolutionary mechanism of living organisms. Computational methods provide promising results for further biological validations which alone are expensive and laborious. However, motif discovery is a well-known challenging problem because of the low signal-to-noise ratio due to both weak conservation and short motif widths. Although additional evidence such as expression data and phylogenetic information can be incorporated to help recognizing some noisy sequences without motifs, the fundamental problem of finding TFBSs on the sequence level is still very difficult for computational methods. One major challenge is the difficulty of searching for the global optimum in a high dimensional space. Numerous algorithms, typically consensus-based search algorithms and statistical optimization methods, have been proposed. Consensus search algorithms suffer from the insufficient descriptive power of string patterns and are limited by the motif width and the maximal error they can handle. Statistical methods are significantly affected by their starting points and often trapped in local optima or even non-optimal solutions. Population-based evolutionary algorithms perform a rather time consuming search and evaluate a large number of useless candidate solutions.

In this paper, we propose a new heuristic approach called Cluster Refinement Algorithm for Motif Discovery (CRMD), which manages to locate the local optimal solutions efficiently and effectively and identify the global optimum from a small number of local optima. CRMD employs a flexible statistical model of motif which allows a variable number of motifs and motif instances. First CRMD uses a novel entropy-based clustering method to find a set of complete and good starting candidate motifs from the input sequences. Then it employs a fast refinement method to search for optimal motifs from the candidate motifs. The clustering chooses informative motif candidates of various types, where the probable initial solutions are maintained and those non-informative ones are discarded to reduce the search space significantly. The refinement method incorporates a greedy sampler to obtain the optimal motif instances from the initial candidate motifs, and it returns a variable number of motif instances by removing or adding motif instances adaptively according to the auto-adjusted thresholds. CRMD can be easily extended if prior knowledge, such as One Occurrence Per Sequence (OOPS), is available. Endowed with an appropriate similarity test of motifs, CRMD is also capable of discovering multiple distinct motifs.

In the experiments, CRMD has the best results on most of the 800 synthetic datasets of a comprehensive range of difficulties. The results on the extensive real datasets, including a set of eight selected real datasets, ABS database [1], SCPD database [2], *Escherichia coli* datasets [3] and Tompa's datasets [4], also show that CRMD seldom falls into local optima as MEME [5] and Motif Sampler [6] do, and its performance is even better than or competitive with those of GAME [7] and GALF-P [8]. GAME and GALF-P are time consuming Genetic Algorithm based motif discovery approaches and are supposed to locate the close-to-optimal binding sites. If the OOPS assumption is assumed, the qualities of the results of CRMD can be further improved. For a real multiple motif problem, CRMD locates a significantly larger number of binding sites than MEME and Motif Sampler. In addition, CRMD has shorter running time than most of the other algorithms tested in this paper even if it is implemented in MATLAB and executed on Windows.

The rest of this paper is organized as follows. In Section

The authors are with the Department of Computer Science and Engineering, The Chinese University of Hong Kong, Shatin, New Territories, Hong Kong (email: {gli, tmchan, ksleung, khlee}@cse.cuhk.edu.hk).

Digital Object Indentifier 10.1109/TCBB.2009.25

II, the problem background and the related methods are briefly introduced. In Section III, the problem is formulated mathematically and an objective function of the problem is derived formally. Section IV describes the algorithm of CRMD and its sub-routines in detail. Extensive experimental results are evaluated in Section V. The last section concludes the paper.

II. Background

TFBSs are small nucleotide fragments (usually$\leq$30bp) in the cis-regulatory regions of genes in DNA sequences. They interact with transcription factors (TFs) and affect the transcriptional activity (or gene expression). The cis-regulatory regions are usually upstream to the transcription start sites (TSS) of the genes. TFBSs typically have a width of 5-10 bp, but there are also real cases such as the CRP binding sites with widths up to around 20 bp. In general, the range of widths can be restricted to around 5 bp to 25 bp. Some well-known characterized TFBSs such as the TATA box are proximal to the TSS, but generally there is no prior spatial knowledge of where the TFBSs occur in the regulatory regions.

Computational methods for identifying TFBSs, namely *de novo* motif discovery, have been proposed as an attractive pre-screening procedure and alternative to the expensive and laborious biological experiments such as DNA footprinting [9] and ChIP-chip [10]. The basis is that certain conserved pattern, called the "motif", exists among the TFBSs in the cis-regulatory regions for a set of similarly expressed genes (co-expressed genes), because those genes are probably regulated by the same or similar TFs. Benefitting from the availability of the large amount of sequencing and microarray data, now we can identify co-expressed genes by clustering and then extract their cis-regulatory regions. *de novo* motif discovery methods try to identify the motif, or equivalently the set of TFBS instances of co-expressed genes without prior knowledge about their consensus appearance.

There have been a few excellent surveys of motif discovery algorithms [4][3][11]. Current motif discovery methods can be categorized into enumerative (consensus based) approaches and statistical (matrix based) ones. They either discover the string pattern (the consensus) using combinatorial approaches or identify the profile of the TFBSs, typically the Position Frequency Matrix (PFM), or Position Weight Matrix (PWM), using statistical modelling.

In enumerative approaches for motif discovery, exact string matching methods fail and exhaustive enumeration is also infeasible due to the NP-hardness [12]. Existing consensus based approaches [13], set the constraint that the maximal hamming distance between the consensus and the motif instances, d, is assumed to be known. They try to enumerate all the strings satisfying the constraints in polynomial time. Typical works include the data structure of suffix trees [14][15] and projections [16][17]. However, such approaches cannot meet the requirements of real world problems well because they can only handle short motif widths (in general up to 14) and small d within reasonable computational time. In the real cases, however, the width can be up to 22 (in the CRP dataset tested in this paper). d is also difficult to determine beforehand and it varies case by case. With too small a d, most of the true TFBSs are missed due to the stringent criteria. With too large a d, the computation time becomes intolerable and a large number of false positives will be output. Another major drawback of consensus based approaches is that the discrete consensus of the motif is not accurate enough to represent the weak conservation between different nucleotides.

A more accurate choice is to use the PFM and PWM to represent a motif with continuous frequency or likelihood of each nucleotide appearing at each position within the motif. Some statistical methods such as Expectation Maximization [5][18] and Motif Sampler [19][20] have been proposed and shown some success in TFBS identification. However, since statistical methods sample TFBSs probabilistically, they may take a long time for their solutions to converge and stabilize. Another disadvantage is that they are sensitive to initial settings, and are often trapped in local optima since many of these methods perform local search only, and their results might not even be local optimal if the searching is ineffective. In TFBS identification, the local optima problems become more critical because the weakly conserved TFBSs are typically weak signals surrounded by a large amount of noise.

Genetic Algorithms (GAs) [7][8][21][22][23] have been applied to TFBS identification as well. The advantage of such GA based methods is that they are likely to locate the global optimum in a typically difficult search space. On the other hand, they are stochastic and so they may fail to report consistent results in different runs. They require a large population of solutions and the computation time is typically long. Nevertheless, the results of the state-of-the-art GA-based method provide a close-to-exhaustive-search-based benchmark to evaluate the performance of our algorithm compared to other approaches.

Recently, approaches incorporating multiple evidence besides the DNA sequences have been proposed to improve the prediction accuracy for real motif discovery problems. Recent reviews usually include these integrated approaches [24][25]. The evidence generally comprises of microarray data for the input sequences, phylogenetic footprinting, ChIP-chip and negative sequences previously known to contain no motifs, just to name a few. Multiple evidence also means additional data sources are needed specifically. While these methods gain success in specific cases, the general motif discovery problem remains challenging because it is usually difficult to have these additional information and the search on sequences known to have certain motifs is still difficult. In this work we focus on the motif discovery involving only DNA sequences, and we believe the improvement on DNA sequences alone will certainly further enhance the methods integrated with additional evidence.

III. Objective

Biologically, the TFBS identification problem is to locate the subsequences in the cis-regulatory regions which are bound by a common protein. Up to now, the process of factor binding is still obscure to biologists, let alone the properties of the

binding sites. To cope with this problem with computational methods, we define the problem as an optimization problem of a certain mathematic objective function in the following subsections. The algorithm to maximize the objective function is presented in Section IV.

A. Problem Formulation

Given a set of DNA sequences, we are required to find the binding sites corresponding to the motif instances, and we are also interested in the common string pattern of the motif. To be consistent with the biological observation, we assume no maximal distance between the motif instances, and the number of motif instances in the sequences.

Data Input. We are given a set of sequences $S = \{S_i|i = 1, 2, ..., D\}$ of nucleotides defined on the alphabet $B = \{A, T, G, C\}$. $S_i = (S_i^j|j = 1, 2, ..., l_i)$ is a sequence of nucleotides, where l_i is the length of the sequence.

The motif width is w nucleotides long, which is assumed known throughout the paper. The set of all the w long subsequences contained in S is $\{s_i^{j_i}|i = 1, 2, ..., D, j_i = 1, 2, ..., l_i - w + 1\}$, where j_i is the binding site of a possible motif instance $s_i^{j_i}$ on sequence S_i.

Position Output. We are required to find the Position Indicator Matrix (PIM) $A = \{A_i|i = 1, 2, ..., D\}$ of the motif, where $A_i = \{A_i^j|j = 1, 2, ..., l_i\}$ is the indicator row vector with respect to (w.r.t) a sequence S_i. A_i^j is 1 if position j in S_i is a binding site, and 0 otherwise. We refer to the number of motif instances as $|A| = \sum_{i=1}^{D}\sum_{j=1}^{l_i} A_i^j$.

Induced by A is a set of $|A|$ motif instances denoted as $S(A) = \{S(A)_1, S(A)_2, ..., S(A)_{|A|}\}$, where $S(A)_i = S(A)_i^1 S(A)_i^2 ... S(A)_i^w$ is the ith motif instance in $|A|$. $S(A)$ can also be expanded as $(S(A)^1, S(A)^2, ..., S(A)^w)$, where $S(A)^j = S(A)_1^j S(A)_2^j ... S(A)_{|A|}^j$ is the list of the nucleotides on the jth position in the motif instances.

Consensus Output. We are also required to find the consensus which is a string abstraction of the motif instances. In the absence of a string consensus, we should find the Position Count Matrix (PCM) $N(A)$ of the numbers of different nucleotide bases on the individual positions of the motif instances of A. $N(A) = (N(A)^1, N(A)^2, ..., N(A)^w)$, and $N(A)^j = \{N(A)_b^j|b \in B\}\}$, where $N(A)_b^j = |\{S(A)_i^j|S(A)_i^j = b\}|$.

$N(A)$ can be further normalized by $|A|$, and thus we get the Position Frequency Matrix (PFM) $\hat{N}(A) = \frac{N(A)}{|A|}$, which can be regarded as a virtual consensus, i.e., the relative frequencies of the nucleotide types on the individual positions in the motif instances. Given an A, it is trivial to calculate $N(A)$. On the contrary, it is not straightforward to find the corresponding A from $N(A)$.

Fig. 1 illustrates an artificial motif discovery problem. We use $M(C) = \{M(C)_b|b \in B\}$ to denote the numbers of different nucleotides in the dataset C, where $M(C)$ applies to all the positions in C. Similarly to PFM, $M(S)$ can be normalized as the relative frequencies of the nucleotides in the sequences S, which is denoted as $\theta_0 = \{\theta_{0b} = \frac{M(S)_b}{\Sigma_{b\in B} M(S)_b}|b \in B\}$.

B. Maximum A Posteriori

To solve a motif discovery problem, we need to find the optimal PIM A or PCM $N(A)$ in terms of a certain optimization measure. There are various methods to evaluate a set of candidate motif instances. We adopt the Bayesian analysis to derive the posterior probability of the motif instances, and thus the motif discovery is to find the motif instances of the maximal probability. To make it easy to understand our proposed algorithm, we repeat the major steps of the derivation in [26] herein.

For the likelihood of the motif instances, we assume that the nucleotides in a motif instance are generated independently across positions. Therefore, the motif instances A follow the multinomial distribution $\prod_{j=1}^{w} p(N(A)^j)$, where $p(N(A)^j)$ is the independent probability of generating the nucleotides on the jth position of the motif instances. We further assume that the probabilities of generating the nucleotides on a position of the different motif instances are independent. The joint probability $p(N(A)^j)$ is thus the product of the probabilities of the nucleotides on position j in the sequences respectively, i.e., $p(N(A)^j) = \prod_{b\in B} \theta_{jb}^{N(A)_b^j}$, where θ_{jb} is the latent probability of generating base b in position j, $N(A)_b^j$ is the number of nucleotide b on position j. In a more succinct form, $\prod_{b\in B} \theta_{jb}^{N(A)_b^j}$ can be written as $\theta_j^{N(A)^j}$, where θ_j is the vector of the latent probabilities $\{\theta_{jb}|b \in B\}$ on position j in the motif instances. In summary, the motif instances A follow the multinomial distribution $\prod_{j=1}^{w} \theta_j^{N(A)^j}$.

For the likelihood of the background sequences, we assume that the nucleotides on the sequences excluding the motif instances follow a multinomial distribution $\theta_0^{M(S(A^C))} = \prod_{b\in B} \theta_{0b}^{M(S(A^C))_b}$, where θ_0 is the vector of the probabilities generating the background nucleotides and A^C is the complement of A w.r.t S. In this paper, we assume θ_0 is fixed as the relative frequencies of the bases in S, which is indifferent to the positions of the bases. Similarly, we also assume an independent binomial distribution of the number of motif instances $|A|$, i.e, $p(A|p_0) = p_0^{|A|} \times (1 - p_0)^{L-|A|}$, where $L = \sum_{i=1}^{N}(l_i - w + 1)$ is the total number of the subsequences and p_0 is an abundance ratio to indicate the probability of a position being a binding site.

The PIM A can be viewed as the missing label of the data S, θ is the latent parameters of the distribution model of A, and p_0 is also unknown beforehand. The likelihood of S is the product of the probabilities of the background sequences and the motif instances as follows,

$$p(S|\theta, \theta_0, A, p_0) = p_0^{|A|}(1 - p_0)^{L-|A|}\theta_0^{M(S(A^C))}\prod_{j=1}^{w}\theta_j^{N(A)^j}$$

For Bayesian analysis, we employ a multinomial Dirichlet distribution as the conjugate prior for θ, i.e., $p(\theta|\alpha) \propto \prod_{j=1}^{w}\prod_{b\in B}\theta_{jb}^{\alpha_b - 1}$, where α is a small common prior for all the θ_js. We also prescribe a Dirichlet distribution as the conjugate prior for p_0, i.e. $p(p_0|p_a, p_b) \propto p_0^{p_a - 1}(1 - p_0)^{p_b - 1}$. Therefore, we have the posterior distribution of A, θ and p_0 as follows, where we have used $\theta_0^{M(A^C)} = \frac{\theta_0^{M(S)}}{\theta_0^{M(A)}} \propto \frac{1}{\theta_0^{M(A)}}$.

(a) sequences S	(b) PIM A	(c) instances $S(A)$	(d) PCM $N(A)$	(e) PFM $\hat{N}(A)$
acgtCGATTGCctaag	0000100000000000	CGATTGC		
taTGATCGAtgacgca	0010000000000000	TGATCGA	A:0261107	A: 0.0 0.2 0.6 0.1 0.1 0.0 0.7
cgaCAATTGAgcttac	0001000000000000	CAATTGA	C:8023323	C: 0.8 0.0 0.2 0.3 0.3 0.2 0.3
gCGCTCGAcaagctgt	0100000000000000	CGCTCGA	G:0800080	G: 0.0 0.8 0.0 0.0 0.0 0.8 0.0
cgttTGTCACAgtcta	0000100000000000	TGTCACA	T:2026600	T: 0.2 0.0 0.2 0.6 0.6 0.0 0.0
tcagcCACACCCagct	0000010000000000	CACACCC		
ccagagCGTCTGAttg	0000001000000000	CGTCTGA		
gacttcaCGACTGAgc	0000000100000000	CGACTGA	$M(S)_A$:38 $M(S)_C$:47	$\theta_{0A} = 0.2375$ $\theta_{0C} = 0.2938$
gctgcccatCGATTGA	0000000001000000	CGATTGA	$M(S)_G$:38 $M(S)_T$:37	$\theta_{0G} = 0.2375$ $\theta_{0T} = 0.2313$
ccaggtacCGATTGCa	0000000010000000	CGATTGC		

Fig. 1. An artificial problem of motif discovery. It shows (a) the sequences S, (b) the Position Indicator Matrix A, (c) the motif instances $S(A)$, (d) the Position Count Matrix $N(A)$ and the count of the background nucleotides $\{M(S)_b|b \in B\}$, (e) the Position Frequency Matrix $\hat{N}(A)$ and the background relative frequencies $\{\theta_{0b}|b \in B\}$. In the sequences S, the letters in lower case are the background bases, and the letters in upper case are the motif instances

$$
\begin{aligned}
& p(\theta, A, p_0|S, \theta_0, \alpha, p_a, p_b) \\
&= p(S|\theta, \theta_0, A, p_0)p(A|p_0)p(\theta|\alpha)p(p_0|p_a, p_b) \\
&= \frac{p_0^{|A|+p_a-1}(1-p_0)^{L-|A|+p_b-1}}{\theta_0^{M(S(A))}} \prod_{j=1}^{w} \theta_j^{N(A)^j+\alpha-1}
\end{aligned}
$$

Since we are not interested in θ and p_0, we can integrate them out using the conversion between the beta function and the gamma function[1]. The resulted posterior conditional distribution of A alone is shown in Eq. 1, where we have used $|\alpha| = \sum_{b\in B} \alpha_b$ and $\sum_{b\in B} N(A)_b^j = |A|$.

$$
\begin{aligned}
& p(A|S, \theta_0, \alpha, p_a, p_b) \propto \int p(\theta, A, p_0|S, \theta_0, \alpha) d\theta dp_0 \\
&= \frac{\Gamma(|A|+p_a)\Gamma(L-|A|+p_b)}{\theta_0^{M(S(A))}} \prod_{j=1}^{w} \frac{\prod_{b\in B}\Gamma(N(A)_b^j+\alpha_b)}{\Gamma(|A|+|\alpha|)}
\end{aligned} \tag{1}
$$

The objective of motif discovery can thus be formulated as to maximize the posterior probability of A in Eq. 1. Prior knowledge, such as the abundance of motif instances in the dataset, the background frequencies of the nucleotide types and the probabilities of nucleotides in the motif instances, can be easily incorporated in the model.

IV. Algorithm

Given a set of sequences S and the motif width w, solving for the optimal PIM A in terms of $p(A)$ in Eq. 1 directly is computationally intractable. Under the assumption of exactly one occurrence (of motif instance) per sequence (OOPS), the (w, d) motif discovery problem is already NP-hard [12]. If the OOPS assumption is relinquished, the search space becomes much bigger, and thus the problem is even more difficult. However, as shown in Motif Sampler [6] and MEME [5], given an initial PCM, it is possible to search for the PIM A whose $N(A)$ is the local optimum of the original PCM via an iterative procedure. Therefore, it is likely to obtain the global optimal A among the local optimal A's from a sufficient number of different initial PCMs.

[1]The beta function $B(x, y) = \int_0^1 t^{x-1}(1-t)^{y-1}dt$, the gamma function $\Gamma(z) = \int_0^\infty t^{z-1}e^{-t}dt$, and $B(x, y) = \frac{\Gamma(x)\Gamma(y)}{\Gamma(x+y)}$

Algorithm 1: Main: the main program of CRMD

Input: The sequences S
Output: The best set of motif instances BA
$P \leftarrow -\infty$;
$clusters \leftarrow Cluster(sub(S))$;
foreach $cluster_i \in clusters$ **do**
1 $N(\tilde{A}_{(i)}) \leftarrow D \times \hat{N}(A(cluster_i))$;
$[A_{(i)}, p(A_{(i)})] \leftarrow Refine(N(\tilde{A}_{(i)}), S)$;
if $p(A_{(i)}) > P$ **then**
$P \leftarrow p(A_{(i)})$;
$BA \leftarrow A_{(i)}$;
if *OOPS* **then**
$[BA, P] \leftarrow Adapt(BA)$;

Algorithm 1 is the main program of our algorithm, i.e., the Cluster Refinement Algorithm for Motif Discovery (CRMD). Firstly, all the w long subsequences are extracted from the sequences ($sub(S)$ in Step 1). In each sequence, the subsequences starting positions range from the first possible binding site 1 until the last possible binding site $|S_i| - w + 1$. Secondly, the $Cluster$ procedure partitions the set of all the candidate subsequences $sub(S)$ so as to group the similar subsequences in the same $cluster_i$, whose PIM is $A(cluster_i)$. Each $cluster_i$ is then used to construct a set of D artificial motif instances $\tilde{A}_{(i)}$ whose PFM $\hat{N}(\tilde{A}_{(i)})$ is equal to the PFM $\hat{N}(A(cluster_i))$ of $cluster_i$. Thirdly, the $Refine$ procedure uses the PCM $N(\tilde{A}_{(i)})$ of the artificial motif instances $\tilde{A}_{(i)}$ to search for the local optimal PIM $A_{(i)}$. The best set of motif instances $A_{(i)}$ in terms of $p(A_{(i)})$ is returned as the result. If we know the motif is consistent with OOPS, a post $Adapt$ procedure can be applied to further enhance the best set of motif instances.

Fig. 2 illustrates the execution path of CRMD with the example in Fig. 1. First, the set of all the subsequences s are extracted from the sequences S. The subsequences are then grouped into separate clusters using $Cluster$. In this example, there are altogether 17 clusters. The PFMs of the

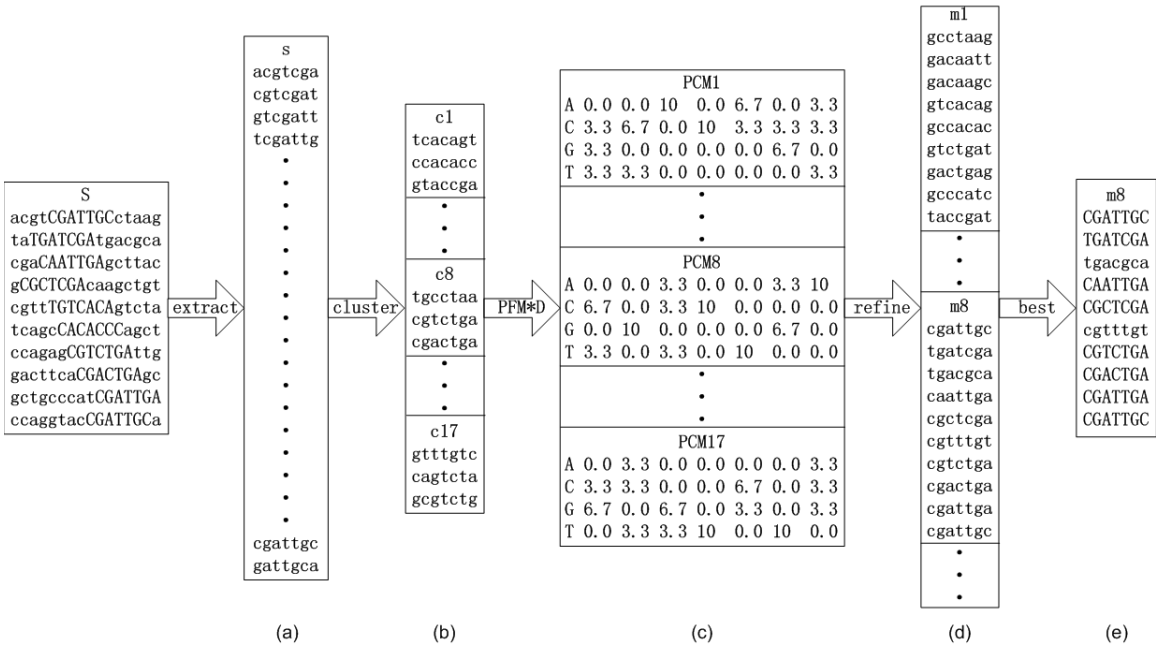


Fig. 2. The execution path of CRMD with the example in Fig. 1. (a) all the subsequences of seven bps are extracted from the sequences S. (b) the subsequences are then grouped into separate clusters. (c) the initial PCMs are calculated as the PFMs of the clusters multiplied with D (10 in this example). (d) the initial PCMs are subsequently refined to find the local optimal set of motif candidates. (e) the best set is returned as the discovered motif instances where the correct instances are in upper cases

clusters are multiplied by $D = 10$ as the PCMs of the artificial sets of motif instances. The artificial PCMs are subsequently used as the initial PCMs to find the local optimal sets of motif candidates in $Refine$. Finally, the best set of the motif instances is returned, in which the correct motif instances are highlighted in upper cases.

In the following subsections, we will give the details of the procedures in the main program of CRMD in Algorithm 1. Subsections A, B and C describe the $Cluster$, $Refine$ and $Adapt$ procedures, respectively. Lastly, subsection D shows how CRMD is extended to handle multiple motif discovery problems.

A. Entropy-based Clustering

The $Cluster$ procedure in Algorithm 1 chooses the initial PCMs for the $Refine$ procedure. A good initial PCM is important for $Refine$ as the resulted local optimum is more likely to be the global optimum than a bad initial PCM. A random PCM usually contains too much noise, and its PFM bears little similarity with the existing subsequences, and so searching from a random PCM infrequently leads to true motif instances. Using an existing subsequence in S as the initial PFM is better than a random one since it is better conserved and it has at least one similar subsequence. However, for a typical motif discovery problem, there are thousands of subsequences, and so using all of them would be expensive. MEME selects some subsequences randomly and perturbs their PFMs somehow as the starting points in its EM algorithm. Nonetheless, there is still no guarantee that the randomly selected subsequences definitely occur in the motif instances.

Our approach of creating and selecting the initial PCMs is by clustering all the subsequences into modest-sized groups. There are four advantages of our clustering. First, clustering all the subsequences guarantees that every subsequence has a large chance to occur in a certain cluster and thus is likely to be considered in the subsequent process. Second, grouping the similar subsequences together exempts us from the costly computation of processing every subsequence later, and in the extreme case the huge number (4^w) of all the potential consensus. Third, clustering similar subsequences into the same group has already accomplished part of the job of maximizing the posterior probability in Eq. 1. Forth, our clustering has an explicit control of the number of the subsequences in a cluster, as it discards small insignificant clusters and partitions large clusters to remove the noise. Compared to other clustering algorithms for motif discovery [18][27], the third and the fourth advantages are very important in finding the motif efficiently and effectively.

Algorithm 2 is the pseudocode of the procedure $Cluster$ in Algorithm 1. Provided with a set of subsequences s, $Cluster$ checks the size of s, i.e., $|s|$, at first. If $|s|$ is smaller than $\frac{D}{4}$, s is discarded. If $|s|$ is larger than $\frac{D}{4}$ and smaller than D, it is returned as a cluster. If $|s|$ is larger than D, $Cluster$ continues to partition s. Step $Pos(s)$ selects the optimal position pos and the optimal nucleotide $base$ to partition s into two sets of the subsequences. The subsequences in the first set s^{pos}_{base} have the nucleotide $b = base$ on position pos, while the subsequences in the other set $s^{pos}_{b \neq base}$ have nucleotides other than $base$ on position pos. Both sets are then recursively clustered in $Clustering(s^{pos}_{base})$ and $Clustering(s^{pos}_{b \neq base})$, respectively. In this way, the set of subsequences s are separated into smaller and smaller clusters by applying $Cluster$ recursively.

Algorithm 2: Cluster: partition the subsequences into separate clusters

Input: The subsequences s
Output: the clusters
$clusters \leftarrow \emptyset$;
if $|s| \leq D$ **then**
  if $|s| \geq \frac{D}{4}$ **then**
    $clusters \leftarrow s$
else
  $[pos, base] \leftarrow Pos(s)$;
  $clusters \leftarrow clusters \cup Cluster(s^{pos}_{base})$;
  $clusters \leftarrow clusters \cup Cluster(s^{pos}_{b \neq base})$;

$Cluster$ keeps a set of subsequences s intact and returns it as a cluster if and only if its size is in the range $[\frac{D}{4}, D]$. If the cluster size is too large, the subsequences in the cluster may have too much diversity which introduces unnecessary noise into the resulted PCM. If the cluster size is too small, the subsequences may constitute no significant motif and thus the cluster is discarded, because we are looking for the binding sites of a common transcription factor bound to sufficient sequences. Even if a motif instance happens to be included in a discarded cluster, it is still possible to recover it from another cluster consisting of other motif instances. In the extreme case, a set of exact $\frac{D}{4}$ subsequences is split and returned as a cluster in each recursion of Algorithm 2. Therefore, all the clusters consist of exact $\frac{D}{4}$ subsequences, and the maximal number of clusters is $\frac{4L}{D}$, where L is the total number of all the subsequences. The actual number of clusters is much smaller than the maximal number because the number of the subsequences in a cluster is usually larger than $\frac{D}{4}$. On the other hand, if the number of the subsequences in a cluster is smaller than $\frac{D}{4}$, the cluster is discarded directly. It is empirically observed that for a typical dataset of thousands of subsequences, the number of clusters is up to only several hundreds.

To choose the optimal position and nucleotide base in step $Pos(s)$ to partition the current set of subsequences s, we adopt the clustering criterion in Eq. 2. For each potential partitioning position pos and nucleotide type $base$, $Pos(s)$ calculates the relative entropy of the subset resulted from partitioning the subsequences s on position pos according to nucleotide type $base$, i.e., $En(s^{pos}_{base})$, and then scales the entropy with the size of the subset, i.e., $|s^{pos}_{base}|$. The position and nucleotide type giving the largest scaled entropy are chosen for partitioning. $|s^{pos}_{base}|$ is considered in finding $Pos(s)$ so that a large cluster is preferred. $En(s^{pos}_{base})$ is the sum of the relative entropies of the subsequences s^{pos}_{base} on all the positions. We use the relative information entropy because we aim to find the set of subsequences which are similar to each other and yet different from the background sequences.

$$Pos(s) = \underset{pos=\{1,2,..,w\}, base \in B}{\operatorname{argmax}} En(s^{pos}_{base})|s^{pos}_{base}|$$

$$En(s^{pos}_{base}) = \sum_{j=1}^{w} \sum_{b \in B} \frac{N(s^{pos}_{base})^j_b}{|s^{pos}_{base}|} log(\frac{N(s^{pos}_{base})^j_b}{|s^{pos}_{base}|} \frac{1}{\theta_{0b}}) \quad (2)$$

Our choice of the clustering criterion in Eq. 2 is deliberate, as it enables $Cluster$ to find the clusters of approximately large posterior probability as defined in Eq. 1. Actually, if we simplify Eq. 1 using the Burnside formula[2] to approximate the gamma function, we may get the log of $p(A)$ as follows,

$$\tilde{L}(A) = log(p(A|S, \theta_0, p_a, p_b, \alpha)) \approx K + f(|A|, p_a, p_b, \alpha) + \sum_{j=1}^{w} \sum_{b \in B} (N(A)^j_b + \alpha_b - 0.5) log \frac{N(A)^j_b + \alpha_b - 0.5}{|A| + |\alpha| - 0.5} \frac{1}{\theta_{0b}}$$

where K is an invariant constant w.r.t all the variables, and $f(|A|, p_a, p_b, \alpha)$ is a function of $|A|$ only. If we substitute A with s^{pos}_{base}, the last term is approximate to $En(s^{pos}_{base})|s^{pos}_{base}|$ in Eq. 2, where $\frac{N(s^{pos}_{base})^j_b}{|s^{pos}_{base}|} \approx \frac{N(A)^j_b + \alpha_b - 0.5}{|A| + |\alpha| - 0.5}$ and $|s^{pos}_{base}| \approx |A| + |\alpha| - 0.5$. Therefore, the set of subsequences s^{pos}_{base} of large $En(s^{pos}_{base})|s^{pos}_{base}|$ is likely to have a large $p(s^{pos}_{base})$.

B. Greedy Refinement

In Algorithm 1 and Fig. 2, the $Refine$ procedure finds a local optimal set of motif instances from an initial cluster. Rather than using the actual subsequences in the cluster, $Refine$ uses the PCM $N(\tilde{A}) = D \times \hat{N}(A(cluster))$ of the initial subsequences $\tilde{A}$ as the seed for further refinement. $\tilde{A}$ is simply a symbol consisting of no actual subsequences since only its PCM is needed in the refinement, and its PFM is equal to $\hat{N}(A(cluster))$. Greedy Refinement subsequently finds a new set of subsequences A, whose $N(A)$ is similar to and yet better conserved than $N(\tilde{A})$.

There are two advantages of our $Refine$ procedure. In Section IV-B1 (Selecting Motif Instances), it uses a fast greedy local search method to find the local optimal motif instances. A greedy heuristic makes the search deterministic, while the Gibbs sampling in Motif Sampler is a random process, and thus $Refine$ converges much faster. In Section IV-B2 (Changing Instance Number), it uses auto-adjusted thresholds to change the number of motif instances adaptively. The search is flexible as it allows a variable number of instances. At the same time, the number is changed by at most one instance each iteration, and thus $Refine$ still converges very fast.

Algorithm 3 shows the overall pseudocode of the $Refine$ procedure. Iteratively, it replaces the old motif candidate instances $A^{(i-1)}$, which is $\tilde{A}$ initially, with the new candidate motif instances $A^{(i)}$. The new candidate motif instances are selected among all the subsequences to maximize the posterior probability $p(A^{(i)})$ based on the old candidate motif instances. NUM denotes the number of motif instances. At the beginning, NUM is set to the number of sequences, i.e., D. In each iteration, after finding NUM candidate motif instances,

[2]Burnside formula $\Gamma(x+1) = x! \approx (x+0.5)^{x+0.5} e^{-x-0.5} \sqrt{2\pi}$

Algorithm 3: Refine: identify the motif instances based on a cluster

Input: the initial $N(\tilde{A})$ and S
Output: The Local Optimal A and $p(A)$
$NUM \leftarrow D$;
$A^{(0)} \leftarrow \emptyset$;
$N(A^{(0)}) \leftarrow N(\tilde{A})$;
for $i \leftarrow 1$ **to** D **do**
  $ratios \leftarrow Ratio(N(A^{(i-1)}), S)$;
1 $A^{(i)} \leftarrow \text{argmax}_{A_i^j}\,(ratios_i^j, NUM)$;
2 **if** *not OOPS* **then**
    $s_1 \leftarrow \text{argmin}_{s_j \in A^{(i)}}\, ratios(s_j)$;
    $T_1 \leftarrow Expect(N(A^{(i)} - \{s_1\}), \theta_0)$;
    if $Ratio(N(A^{(i)} - s_1), s_1) < T_1$ **then**
      $NUM \leftarrow NUM - 1$;
      $A^{(i)} \leftarrow A^{(i)} - \{s_1\}$
    else
      $s_2 \leftarrow \text{argmax}_{s_j \notin A^{(i)}}\, ratios(s_j)$;
      $T_2 \leftarrow Expect(N(A^{(i)}), \hat{N}(A^{(i)}))$;
      if $Ratio(N(A^{(i)}), s_2) > T_2$ **then**
        $NUM \leftarrow NUM + 1$;
        $A^{(i)} \leftarrow A^{(i)} + s_2$
  if $A^{(i)} = A^{(i-1)}$ **then**
    $A \leftarrow A^{(i)}$;
    $p(A) \leftarrow p(A|S, \theta_0, p_a, p_b, \alpha)$;
    return;

$Refine$ tries to remove the least likely candidate motif instance s_1 to increase $p(A^{(i)})$. If it is removed successfully, NUM is decreased. Otherwise $Refine$ tries to add the next most likely subsequence s_2 to increase $p(A^{(i)})$. If it is added successfully, NUM is increased. $Refine$ stops iterating when A remains the same in two consecutive iterations, and finally it returns the last A.

The two main steps in $Refine$, the selection of motif instances and the adaptive changing of the number of motif instances are given below:

1) Selecting Motif Instances: Iteratively, $Refine$ finds a new set of more conserved motif instances $A^{(i)}$ which are similar to the old set of candidate motif instances $A^{(i-1)}$. The similarity of a subsequence A_i^j to existing motif instances A^* is measured by how much the posterior probability $p(A^*)$ increases if A_i^j is added in A^*. Instead of calculating the two probabilities with or without A_i^j and then comparing them, we can calculate the ratio between them directly. $Ratio(N(A^*), A_i^j)$ in Eq. 3 is the strength of a position A_i^j being a binding site based on the current $N(A^*)$. The derivation is similar to Eq. 1, where we have used the equation $\Gamma(n+1) = n\Gamma(n)$ to cancel out the Gamma functions in both numerator and denominator. $N(A^*)_{b^k}^k$ is the number of nucleotide $b^k = S_i^{j+k-1}$ in $S(A^*)^k$, i.e., the same nucleotide type on position k in $S(A^*)$ as the one b^k in the subsequence $S(A_i^j)$.

$$
\begin{aligned}
Ratio(N(A^*), A_i^j) &= \frac{p(A_i^j = 1|A^*, S)}{p(A_i^j = 0|A^*, S)} \\
&= \frac{\int p(A_i^j = 1|\theta, A^*, S, p_0) p(\theta|A^*, S) p(p_0|p_a, p_b)\, d\theta}{\int p(A_i^j = 0|\theta, A^*, S, p_0) p(\theta|A^*, S) p(p_0|p_a, p_b)\, d\theta} \\
&= \frac{1}{\theta_0^{M(S(A_i^j))}} \frac{|A^*| + p_a}{L - |A^*| + p_b - 1} \prod_{k=1}^{w} \frac{N(A^*)_{b^k}^k + \alpha_{b^k}}{|A^*| + |\alpha|} \qquad (3)
\end{aligned}
$$

After calculating the ratios of all the A_i^j based on the old instances $A^{(i-1)}$, $Refine$ selects NUM subsequences of the maximal ratios directly and replaces $A^{(i-1)}$ with $A^{(i)}$ in Step 1. $Refine$ is greedy because it always select the subsequences of the best matches, and so it may get stuck in local optima. This is exactly why CRMD adopts a multi-start approach with $Cluster$ to locate the global optimum out of many local optima. However, being greedy, $Refine$ is fast as $A^{(i)}$ usually converges in less than $\frac{D}{2}$ iterations. On the contrary, Motif Sampler uses Gibbs sampling to iteratively select subsequences with probabilities in proportion to their Bayes factors. As a Markov Chain Monte Carlo method, Gibbs sampling may take an undetermined time before generating samples following the target distribution.

2) Changing Instance Number: An important issue in discovering motif instances is choosing an appropriate number of predicted motif instances. Predicting too many motif instances may lead to many false instances, while predicting too few motif instances may miss many true instances.

To address the issue of the unknown number of motif instances, CRMD changes the number of motif instances NUM adaptively to increase the posterior probability as defined in Eq. 1. More specifically, Algorithm 3 adds or removes a marginal motif instance by comparing its ratio to the thresholds T_1 and T_2, which are calculated adaptively based on the existing motif instances. The number of the predicted motif instances is changed by at most one in an iteration, and so it is fast and easy for the motif instances to converge in the Greedy Refinement.

In detail, after sampling NUM candidate instances, $Refine$ selects the one with the smallest ratio, i.e., $s_1 \leftarrow \text{argmin}_{s_j \in A^{(i)}}\, ratios(s_j)$, and checks if removing it would increase the posterior probability of the rest of the candidate instances $A^{(i)} - \{s_1\}$. $Refine$ calculates the ratio $Ratio(N(A^{(i)} - \{s_1\}), s_1)$. A small ratio means s_1 affects $A^{(i)} - \{s_1\}$ negatively and thus should be removed. Otherwise, $Refine$ checks if the subsequence of the largest ratio in the remaining subsequences, i.e., $s_2 \leftarrow \text{argmax}_{s_j \notin A^{(i)}}\, ratios(s_j)$, would benefit the probability of the current set of motif instances $A^{(i)}$. Similarly, $Refine$ calculates the ratio $Ratio(N(A^{(i)}), s_2)$ and adds s_2 if the ratio indicates that it will increase the probability. NUM is decreased or increased depending on whether a subsequence is removed or added. The order of removing and adding motif instances is irreversible. Due to the possible spurious binding sites, some noise may be included in the current set of motif instances.. Therefore, the noise must be removed first before searching for more motif instances.

In Algorithm 3, $Refine$ compares the ratios with the

thresholds T_1 and T_2 in the two "if" conditions, and it removes or adds the subsequence if the condition is satisfied. It is important to choose appropriate values for the two thresholds T_1 and T_2 since they control the value of NUM directly. Intuitively, both thresholds should be 1 since the ratio is 1 when the posterior probabilities of a set of motif instances with or without the subsequence are equal. However, since the motif is usually weakly conserved, it is possible that a true binding site is mutated somehow and looks quite different from the others, and so removing it (or not adding it) may actually increase the posterior probability of the set of the motif instances. We also have to be prudent to add new motif candidates since a false subsequence may readily increase the posterior probability of a very weakly conserved motif. Therefore, 1 may be inappropriate for the thresholds.

$Refine$ adjusts T_1 and T_2 automatically to account for two concerns. First, because each iteration in Algorithm 3 may have a different set of motif instances $A^{(i)}$, the thresholds are always calculated in accordance with the current $A^{(i)}$. Second, since there is no prior knowledge of the subsequence to be included or excluded, the thresholds should take into account all the possible subsequences of a certain distribution. Therefore, the threshold that $Refine$ uses is the expected ratio of a random subsequence generated from a certain distribution Θ w.r.t the current set of motif instances A, i.e. $E(Ratio(N(A), s)|\Theta)$. A naive yet computationally intensive method to calculate the expectation is to collect the ratios of all the possible subsequences over the current motif instances A and taking their average in proportion to their probabilities of the specified distribution. Fortunately, Eq. 4 shows an analytical formula to calculate the expected ratio efficiently. In Eq. 4, s_i is one of the 4^w possible subsequences generated from the distribution parameterized by Θ with the probability $p(s_i|\Theta)$. b_i^j is the nucleotide base on position j in the subsequence s_i.

$$\begin{aligned} E(Ratio(N(A), s)|\Theta) &= \sum_{i=1}^{4^w} Ratio(N(A), s_i) p(s_i|\Theta) \\ &= \frac{|A| + p_a}{L - |A| + p_b - 1} \sum_{i=1}^{4^w} \prod_{j=1}^{w} \frac{N(A)^j_{b_i^j} + \alpha_{b_i^j}}{(|A| + |\alpha|)\theta_0^{b_i^j}} \Theta^j_{b_i^j} \\ &= \frac{|A| + p_a}{L - |A| + p_b - 1} \prod_{j=1}^{w} \sum_{b \in B} \frac{N(A)^j_b + \alpha_b}{(|A| + |\alpha|)\theta_0^b} \Theta^j_b, \end{aligned} \tag{4}$$

The computation of the part $\sum_{i=1}^{4^w} \prod_{j=1}^{w}$ in Eq. 4 is greatly simplified by using Eq. 5, which reduces $4^w \times w$ variable references (of only $4w$ distinct $x^j_{k^j}$) on the left to $4w$ variable references (without repetition) on the right. The reason is that $\sum_{i=1}^{4^w} \prod_{j=1}^{w}$ in Eq. 4 actually involves only the complete enumeration over the cartesian product of the sets $\{\frac{N(A)^j_b + \alpha_b}{(|A|+|\alpha|)\theta_0^b} \Theta^j_b | b \in B\}$, where $j = 1 \cdots w$. Therefore, we can rewrite $\sum_{i=1}^{4^w} \prod_{j=1}^{w}$ as the left of Eq. 5, where $x^j_{k^j} = \frac{N(A)^j_{k^j} + \alpha_{k^j}}{(|A|+|\alpha|)\theta_0^{k^j}} \Theta^j_{k^j}$, and simplify it as the right of Eq. 5.

$$\overbrace{\sum_{k^1 \in B} \sum_{k^2 \in B} \cdots \sum_{k^w \in B}}^{w} \prod_{j=1}^{w} x^j_{k^j} = \prod_{j=1}^{w} \sum_{k^j \in B} x^j_{k^j} \tag{5}$$

Eq. 4 is consequently used to calculate T_1 and T_2 under different distributions. For removing a motif instance, we use the possible subsequences generated from the background distribution ($\Theta = \theta_0$) to calculate the threshold $T_1 = E(Ratio(N(A), s)|\theta_0)$. If the ratio of the instance in question is smaller than T_1, it is no better than a background subsequence, and consequently it is definitely discarded. For adding a subsequence, $T_2 = E(Ratio(N(A), s)|\hat{N}(A))$ is used, which is the average ratio of the subsequences generated from the current PFM. Derived from the Maximum Likelihood principle, $\hat{N}(A)$ is actually the latent probabilities generating the current motif instances. Therefore, a new subsequence should be definitely added if its ratio is bigger than the average ratio of the motif instances, namely T_2.

C. Post Adaptation

Greedy Refinement allows a variable number of motif instances, and it adjusts the number of motif instances automatically in the searching. If the problem has One Occurrence of motif instance Per Sequence (OOPS), the performance of CRMD can be further enhanced since the search space is greatly reduced.

To take the advantage of the OOPS assumption, we make two modifications to Algorithm 3 of $Refine$. First, in Step 1, the binding sites are selected on the sequences separately. For each sequence, $Refine$ compares the ratios of its subsequences, and then selects the one and the only one of the maximal ratio. Second, the number of motif instances NUM is constantly D, and so the part of changing NUM in the $if - then$ loop is not executed (Step 2). Since NUM is not changed anymore, it becomes easier and faster for $A^{(i)}$ to stabilize.

However, in the case that OOPS is only an approximation, we still need to fine tune the number of motif instances. Considering that the motif is close to OOPS, it is better not to change the number of motif instances NUM inside $Refine$ since the iterative searching may amplify the noise introduced by any additional candidate instance due to the changing NUM.

In Algorithm 1 of the main program, the procedure $Adapt$ further processes the best set of motif instances BA. The original BA is consistent with OOPS, but the true motif may be slightly different from OOPS. $Adapt$ first removes the existing instances in BA whose ratios are smaller than T_1 even though it might remove all the candidate instances on a certain sequence. $Adapt$ then adds new instances not in BA if their ratios are larger than T_2 even though it might find more than one candidate instance on a certain sequence. Here T_1 and T_2 are calculated in the same way as in the procedure $Refine$. It is unnecessary to apply $Adapt$ to every $A_{(i)}$ returned by $Refine$ in Algorithm 1, because $Adapt$ usually does not change the ranking of the posterior probabilities of the sets of motif instances if the problem is inherently OOPS.

D. Multiple Motifs Discovery

CRMD can be extended to solve the multiple motif discovery problem. It is possible for a set of real DNA sequences to contain multiple motifs. The multiple motifs may have various kinds of consensuses, numbers of instances and degrees of conservations. Due to the diversity of the multiple motifs, the signal-to-noise ratios are even lower than that in the single motif discovery problem. Therefore, it is usually more difficult to find multiple motifs than a single motif. Some traditional motif discovery algorithms run their single motif searching procedures multiple times to locate different motifs. After finding a motif, the corresponding subsequences and their neighbors are masked off the sequences so that the overlapping subsequences will not be identified as new motif instances later on. The shortcoming of this masking scheme is that the discovery of subsequent motifs are dependent on the previously predicted motifs. If some spurious instances are included in a motif, the neighboring subsequences which might be true motif instances are masked off. Even if a true binding site is predicted but included in a wrong motif, masking it off too early may corrupt the consensus of the corresponding motif and thus affect the discovery of the motif later.

To avoid the aforementioned drawbacks, CRMD finds multiple motifs without masking simultaneously. As $Refine$ samples a set of motif instances based on each cluster, a straightforward approach for CRMD is to select a few candidate motifs among all the sets of motif instances returned by $Refine$, i.e., $\{A_{(i)}\}$. The selection is performed according to two criteria, namely the posterior probabilities of the possible motifs and the similarities between the selected motifs. Since the motifs are weakly conserved and the clusters from $Cluster$ may have similar consensus, it is possible that the resulted motifs after $Refine$ are similar and predict many common binding sites. Therefore, among a group of similar motifs, only the motif of the highest posterior probability is selected.

In the current implementation of CRMD, we specify the number of motifs M beforehand. When a new motif is returned by $Refine$, it is firstly checked if it is similar to any of the motifs already selected. If so, the new motif replaces the similar motif if the former has a higher probability. If there is no similar motifs already selected, the new motif replaces the selected motif of the lowest probability if the former has a higher probability. The approach ensures that the M output motifs are different from each other, and at the same time they are of as high probability as possible.

To measure the similarity between the PCMs of two motifs, we adopt the homogeneity test using the χ^2 distance in [28]. Basically, it shifts and aligns the two motifs. If their PCMs on all the overlapping positions are statistically generated from the same distribution, the two motifs are deemed similar.

V. Experiments

We have tested CRMD on both synthetic and real DNA datasets. A testing dataset consists of sequences with motif instances already tagged, and hence it can be used for the algorithm performance evaluation. For some datasets, the widths of the motifs were assumed known beforehand and were be used directly in CRMD. For the other datasets with unknown widths, we either used a common fixed width or tried a range of different widths and selected the width giving the best result.

Some researchers use two levels of performance indices to evalute the algorithm [4][3]. On the nucleotide level, it is calculated that how many nucleotides that the predicted instances and the true instances overlap for. On the site level, a predicted instance is correct if it overlaps with the true instance for at least one nucleotide. To combine the performance indices on both levels, we propose that a motif instance A_i^j is correctly recovered if either of its ends is within three bp away from the corresponding end of the true motif instance [7][8]. More formally, we have,

$$A_i^j = 1 \ is \begin{cases} correct & \text{if } |j - j_s| < 3 \text{ or } |j + w - j_e| < 3 \\ incorrect & \text{otherwise} \end{cases} \tag{6}$$

where j_s and j_e are the indices of the starting and ending positions of the closest true motif instance. The three bp tolerance is reasonable since the widths of the tagged motif instances vary around the known width in a real dataset. It is conjectured that the true motif instance should lie somewhere between the two ends of the tagged instances [4]. This criterion of successful prediction is strict and practical since it does tell a biologist where to look for the true binding sites. In contrast to comparing binding sites, comparing the PFM or PWM of the discovered motif and the true motif may be insufficient, because a small difference in PFM or PWM may lead to very different binding sites.

To measure the performance of CRMD and other algorithms, we adopt the metrics of $Precision$, $Recall$ and $F-score$ [7][8] defined as below, where the operator $|\cdot|$ is the cardinality of the set.

$$\begin{aligned} Precision &= \frac{|correct\ motif|}{|motif\ found|} \\ Recall &= \frac{|correct\ motif|}{|true\ motif|} \\ F-score &= 2 \times \frac{Precision * Recall}{Precision + Recall}, \end{aligned}$$

After we find the candidate instances computationally, the results need to be verified in biological experiments. We hope for a high $Precision$ to avoid wasting too much effort on the false motif instances. In the meanwhile, we should miss as few true motif instances as possible, so a high $Recall$ is preferred. However, there is often a tradeoff between $Precision$ and $Recall$ in real problems. Sometimes a high $Recall$ means a large number of candidate instances, which may consist of many false positives. On the contrary, a high $Precision$ can be achieved by retaining only the highly conserved motif instances at the risk of deleting some true weakly conserved motif instances by mistake. Therefore, $F-score$ is introduced to mix $Precision$ and $Recall$.

We have compared CRMD to Motif Sampler [6], MEME [5], GAME [7] and GALF-P [8]. Since Motif Sampler and MEME are sensitive to the initial settings, they were executed in the manner of multi-start with different starting points.

GAME and GALF-P are GA-based, and their results may be inconsistent and affected by the random seeds, so only the average results of GAME and GALF-P in 20 runs are reported. In each run, the total numbers of the sets of motif instances searched by GALF-P and GAME are 3,000,000 and 30,000,000, respectively. With such a large number of sampling, the searching of GALF-P and GAME are relatively exhaustive, and their results are expected to be close-to-optimal.

The following subsections A, B and C give the details of the results for the synthetic single motif, real single motif and real multiple motif discovery problems tested in our experimental evaluations. In the real single motif discovery experiment, we have tested the eight selected datasets in GAME [7] and GALF-P [8], the ABS database [1], the SCPD database [2], the *Escherichia coli* dataset [3] and the Tompa dataset [4].

A. Synthetic Datasets

A total of 800 synthetic datasets with length 300 bp for each sequence were generated with the following eight combinations of scenarios: (1) motif width: short (8 bp) or long (16 bp); (2) number of sequences: small (20) or large (60); (3) motif conservation: high or low. For each combination, 100 datasets are generated randomly and embedded with the instances of a random motif. In the high conservation scenario, on every position of the motif instances, the dominant nucleotide is generated with 0.91 probability (while all other three nucleotides with 0.03 each). In the low conservation scenario, only 60% of the positions in the motif instances are as highly conserved as in the previous high conservation scenario, while the rest 40% of the positions are lowly conserved, where the dominant nucleotide is generated only with probability 0.55 (while all other three nucleotides with 0.15 each) in every instance. To simulate the noisy situation in real data, in each synthetic dataset, the probability of containing no motif instances is 0.1 for each sequence. In the rest of the sequences which contain motif instances, the probability for a sequence to have more than one instance is 0.1. The number of additional instance(s) in such a sequence follows the geometric distribution with $p = 0.5$, i.e., $p(k) = (1-p)^{k-1}p$, and so there are expectedly $\frac{1}{p} = 2$ additional motif instances embedded in the sequence.

Table I shows the results of the five algorithms. For each scenario, the results are averaged over the 100 datasets. CRMD has the highest average $F-scores$ on six out of eight scenarios. In the remaining two scenarios CRMD has the second highest average $F-scores$. CRMD also has the highest average $F-score$ over all the 800 problems of eight different scenarios, which proves that CRMD is relatively robust in a variety of problems. For the easy datasets in the last two scenarios (with long motif width and high conservation), all the algorithms have very good results ($F-scores$ around 0.98), and so there is no big room for the improvement for CRMD. For the other more difficult problems, the results of the algorithms vary in a wider range and the advantage of CRMD is more apparent. It is also interesting to notice that MEME has the highest average $Precision$ in most of the scenarios, while GALF-P has the highest average $Recall$ in most of the scenarios. However, CRMD has both good $Precisions$ and $Recalls$ on most of the datasets, and thus it yields the highest average $F-scores$ due to the good balance between $Precision$ and $Recall$.

TABLE II
THE REAL DATASETS: THE NUMBERS AND THE LENGTHS OF SEQUENCES, THE WIDTH AND THE NUMBERS OF MOTIF INSTANCES

dataset	#sequence	length	width	#instance
CREB	17	350	8	19
CRP	18	105	22	23
ERE	25	200	13	25
E2F	25	200	11	27
MEF2	17	199	7	17
MYOD	17	200	6	21
SRF	20	345	10	36
TBP	95	200	6	95

B. Real Datasets

To investigate the performance of CRMD on real datasets, and how it is compared to other algorithms, we have also tested the algorithms on a wide range of real datasets. Section V-B1 (Eight Selected Datasets) describes a detailed analysis of the results on the eight datasets tested by GAME [7] and GALF-P [8]. Section V-B2 (ABS and SCPD databases) reports the results on the ABS database [1] and the SCPD database [2]. Section V-B3 (E. coli and Tompa datasets) reports the results on the *Escherichia coli* dataset [3] and the Tompa dataset [4].

1) Eight Selected Datasets: Following GAME [7] and GALF-P [8], we have tested eight real datasets, i.e., CREB, CRP, ERE, E2F, MEF2, MYOD, SRF and TBP, and compared the performance with the other four algorithms. These eight datasets consist of the sequences from many different species. The CRP dataset contains TFBSs bound by the cyclic amp receptor protein in *Escherichia Coli* [29][30][31]. The ERE dataset contains the estrogen receptor elements that ER binds, from the sequences of various species [32]. The E2F dataset contains TFBSs of the E2F family from different mammalian species [33][34][35]. The datasets of CREB, MEF2, MYOD, SRF and TBP were chosen by GAME from the ABS database of annotated regulatory binding sites [1]. As shown in Table II, the real datasets have a variety of the numbers of sequences, the lengths of sequences, the widths of motifs and the numbers of motif instances. We adopt the same motif widths as used in [7] and [8]. For a fair comparison, all the algorithms were run with as few prior knowledge as possible, and most of their running options were set to their default values.

Table III compares the results of the four algorithms (GAME, MEME, Motif Sampler and CRMD) on the eight real datasets. Due to the adaptive thresholds adopted in the Greedy Refinement, CRMD is able to choose an appropriate number of motif instances, and thus finds a good balance between $Precision$ and $Recall$, which consequently leads to the highest $F-scores$ on six problems. Compared to Motif Sampler, CRMD is better on seven out of eight datasets in terms of the $F-scores$. Compared to MEME, CRMD loses on two datasets while it wins on all the other datasets. For the MYOD problem in particular, because its motif width is short and the number of the sequences is small, the signal-to-noise ratio is low. Other algorithms were unable to identify

TABLE I
AVERAGE RESULTS FOR THE SYNTHETIC DATASETS EXPERIMENT: WIDTH IS FOR THE MOTIF WIDTH, NUM IS FOR THE NUMBER OF SEQUENCES, CON IS FOR CONSERVATION DEGREE, P IS FOR $Precision$, R IS FOR $Recall$ AND F IS FOR F-SCORE. SAMPLER REFERS TO MOTIF SAMPLER

Scenario			GALF-P			GAME			MEME			Sampler			CRMD		
Width	Num	Con	P	R	F	P	R	F	P	R	F	P	R	F	P	R	F
Short	Small	Low	0.38	0.56	0.44	0.29	0.32	0.30	0.49	0.34	0.39	0.44	0.37	0.40	0.46	0.45	**0.46**
Short	Large	Low	0.52	0.59	**0.55**	0.42	0.32	0.36	0.63	0.33	0.42	0.55	0.41	0.46	0.53	0.53	0.53
Long	Small	Low	0.87	0.91	0.89	0.78	0.87	0.82	0.91	0.86	0.88	0.87	0.89	0.88	0.91	0.88	**0.91**
Long	Large	Low	0.91	0.90	0.91	0.92	0.90	0.90	0.96	0.85	0.90	0.89	0.92	0.91	0.92	0.92	**0.92**
Short	Small	High	0.73	0.90	0.80	0.71	0.80	0.75	0.87	0.84	0.85	0.85	0.85	0.85	0.86	0.83	**0.86**
Short	Large	High	0.81	0.86	0.83	0.83	0.83	0.83	0.91	0.76	0.83	0.87	0.83	**0.85**	0.84	0.83	0.84
Long	Small	High	0.97	0.99	0.98	0.94	0.99	0.97	0.98	0.99	0.98	0.96	1.00	0.98	0.99	0.99	**0.99**
Long	Large	High	0.97	0.97	0.97	0.98	0.99	0.98	0.99	0.98	0.98	0.96	1.00	0.98	0.99	0.99	**0.99**
Average			0.77	0.84	0.80	0.73	0.75	0.74	0.84	0.74	0.78	0.80	0.78	0.79	0.81	0.80	**0.81**

TABLE III
THE RESULTS FOR THE REAL DATASETS ASSUMING NO OOPS: P IS FOR $Precision$, R IS FOR $Recall$ AND F IS FOR F-SCORE. SAMPLER REFERS TO MOTIF SAMPLER

Problem	GAME			MEME			Sampler			CRMD		
	P	R	F	P	R	F	P	R	F	P	R	F
CREB	0.43	0.42	0.42	0.71	0.63	**0.67**	0.71	0.63	**0.67**	0.67	0.63	0.65
CRP	0.79	0.78	0.78	0.89	0.67	0.76	0.94	0.70	0.80	1.00	0.74	**0.85**
ERE	0.52	0.78	0.62	1.00	0.68	**0.81**	0.75	0.72	0.73	0.71	0.80	0.75
E2F	0.79	0.87	0.83	0.82	0.85	0.84	0.88	0.85	0.87	0.83	0.93	**0.88**
MEF2	0.52	0.55	0.53	0.93	0.82	0.88	0.72	0.76	0.74	0.85	1.00	**0.92**
MYOD	0.14	0.14	0.14	0.29	0.19	0.23	0.46	0.29	0.35	0.86	0.90	**0.88**
SRF	0.71	0.86	0.78	0.74	0.89	0.81	0.76	0.86	0.81	0.79	0.86	**0.83**
TBP	0.81	0.74	0.77	0.83	0.69	0.76	0.74	0.67	0.70	0.83	0.89	**0.86**
Average	0.59	0.64	0.61	0.78	0.68	0.72	0.74	0.69	0.71	0.82	0.84	**0.83**

TABLE IV
THE RESULTS FOR THE REAL DATASETS ASSUMING OOPS: P IS FOR $Precision$, R IS FOR $Recall$ AND F IS FOR F-SCORE

Problem	GALF-P			MEME			CRMD		
	P	R	F	P	R	F	P	R	F
CREB	0.70	0.84	0.76	0.71	0.63	0.67	0.73	0.84	**0.78**
CRP	0.99	0.73	**0.84**	0.67	0.52	0.59	0.94	0.74	0.83
ERE	0.82	0.76	**0.79**	0.76	0.76	0.76	0.67	0.80	0.73
E2F	0.77	0.85	**0.81**	0.76	0.70	0.73	0.68	0.85	0.75
MEF2	0.91	0.98	0.95	0.94	0.94	0.94	1.00	1.00	**1.00**
MYOD	0.57	1.00	0.72	0.06	0.05	0.05	0.86	0.90	**0.88**
SRF	0.75	0.89	0.82	0.95	0.53	0.68	0.77	0.92	**0.84**
TBP	0.87	0.87	0.87	0.94	0.94	**0.94**	0.85	0.94	0.89
Average	0.80	0.87	0.82	0.72	0.63	0.67	0.81	0.87	**0.84**

TABLE V
THE RUNNING TIME (SECOND) COMPARISON. CRMD WAS IMPLEMENTED IN MATLAB, GAME WAS IMPLEMENTED IN JAVA, AND THE OTHER ALGORITHMS WERE IMPLEMENTED IN C. CRMD, GAME AND GALF-P WERE EXECUTED IN WINDOWS, WHILE THE OTHER ALGORITHMS WERE EXECUTED IN LINUX

Problem	GALF-P	GAME	MEME	Sampler	CRMD
CREB	42.75	133.00	1.41	16.69	24.77
CRP	98.20	390.05	0.50	13.87	22.55
ERE	83.20	334.20	2.06	34.83	47.48
E2F	86.95	286.65	2.15	33.54	58.67
MEF2	34.40	112.05	1.29	14.83	23.14
MYOD	26.25	91.05	1.45	13.60	24.00
SRF	49.10	224.05	1.69	22.76	82.84
TBP	74.40	768.80	39.05	87.00	599.13

the true motif in MYOD probably because of the marginal win of the fitness of the true motif. On average, CRMD has the highest $Precision$, $Recall$ and $F-score$, which proves that the performance of CRMD is quite stable on the eight problems.

We have also tested the algorithms with the prior knowledge of OOPS. A close investigation of the eight problems reveals that they are more or less consistent with the OOPS assumption. As shown in Table II, except for the problem SRF, the numbers of motif instances are close to the numbers of sequences. We activated the $Adapt$ procedure in CRMD and searched only for OOPS solution in the $Refine$ procedure. We have also run GALF-P and MEME on the eight problems, which are capable of searching for OOPS consistent motifs. GALF-P searches for OOPS solutions only in its GA procedure, and then it shrinks or expands the solutions with a heuristic post processing procedure. MEME has an OOPS running option which enables MEME to search for exactly one instance in each sequence. Table IV shows the results of CRMD, GALF-P and MEME. Mostly, CRMD obtains better results than those obtained without OOPS in Table III. Compared to GALF-P and MEME, CRMD has the highest $F-score$s on four problems, and the second highest $F-score$s on the other four problems. CRMD also has the highest average $F-score$. In particular, on the problem of MYOD, CRMD has a remarkable advantage over the other two algorithms. Even though GALF-P is the best on three problems, its results have some variance since it is GA-based and sensitive to the initial population.

Table V compares the running time of the five algorithms (without OOPS) on the eight real datasets. CRMD was implemented in MATLAB, GAME was implemented in Java, and the other algorithms were implemented in C. CRMD, GAME and GALF-P were executed in Windows, while the other algorithms were executed in Linux. This is not a completely fair comparison, but it conveys a rough idea that how the running time of the algorithms are compared with each other. Even though CRMD was implemented in MATLAB and executed in Windows, its running time is still acceptable. The quality of the solutions of GALF-P is comparable to CRMD, but the running time of GALF-P is longer than CRMD in most problems. Compared to the two non-GA-based algorithms Motif Sampler and MEME, CRMD is comparable to Motif Sampler, and MEME is the fastest, whose solution qualities are worse than those of CRMD though. We believe if we had implemented CRMD in C instead and executed it on Linux, its running speed would be significantly faster. This verifies that CRMD is capable of predicting motif binding sites in real life problems accurately and consistently with a reasonable amount of computation power.

To further evaluate the effectiveness of $Cluster$ in CRMD, Table VI compares the total numbers of all the subsequences, the theoretical maximal numbers of the clusters and the actual

TABLE VI
THE NUMBERS OF THE SUBSEQUENCES, THE THEORETICAL MAXIMAL NUMBERS OF THE CLUSTERS $\frac{2L}{D}$, THE NUMBERS OF THE CLUSTERS AND THE REDUCTIONS OF THE SEEDS FOR $Refine$

Problem	#subsequence	max($\frac{4L}{D}$)	#cluster	reduction
CREB	3544	834	292	92%
CRP	1512	336	118	92%
ERE	4700	752	276	94%
E2F	4750	760	268	94%
MEF2	3293	775	267	92%
MYOD	3315	780	280	92%
SRF	4127	825	292	93%
TBP	18525	780	296	98%

numbers of the clusters. The theoretical maximal number of clusters is $\frac{4L}{D}$ which is already significantly smaller than the total number of all the subsequences L. In the eight testing real datasets, the actual numbers of clusters are even much smaller than the theoretical maximal numbers of the clusters. The reductions over the total numbers of the subsequences are over 90%. This shows that $Cluster$ not only provides good initial PCMs for $Refine$, but also saves a lot of computation time.

It is also interesting to inspect the performance of CRMD without either of $Cluster$ and $Refine$. As regard to $Cluster$, it is empirically observed that with the same number of initial candidates, the performance of using random initialization instead for $Refine$ is worse than that of using $Cluster$. The results of random initialization also vary with the different pseudo-random number seeds. Especially for the problem MYOD, which has short width and small number of sequences, the precision and the recall of the random initialization are zero in the worst case. As regard to $Refine$ (with $Cluster$ still in CRMD), if the thresholds T_1 and T_2 in Algorithm 3 are fixed at 1, the performance of CRMD deteriorates a lot. On the problems MEF2 and MYOD in particular, CRMD cannot find any correct motif instance at all.

2) ABS and SCPD databases: Besides the eight selected datasets, we have tested CRMD, MEME and Motif Sampler on the ABS [1] and the SCPD [2] databases as well. The ABS database has 650 experimental binding sites from 69 transcription factors in human, mouse, rat and chicken genome sequences. We downloaded the sequences and the binding sites from the website of ABS database, and re-grouped the sequences of the same transcription factors together, and thus we got a total of 69 datasets, each of which consists of the sequences bound by a common transcription factor. SCPD is a promoter database of the yeast *Saccharomyces cerevisiae*. It contains 580 experimentally mapped transcription factor binding sites. Because the website provides no FASTA files, we had to collect and organize the sequences and the binding sites manually, and kept only the transcription factors of more than four binding sites, and thus we got a total of 28 datasets.

We assume no prior knowledge of the exact widths for the motifs in the ABS and the SCPD datasets. CRMD and Motif Sampler used the commonly adopted fixed widths of 10 bps and 13 bps in ABS and SCPD, respectively, which are the medians of the widths of the true motif instances in ABS and SCPD, respectively. For MEME, the widths varied between

TABLE VII
THE AVERAGE RESULTS OF MEME, MOTIF SAMPLER AND CRMD ON THE ABS AND THE SCPD DATABASES. P IS FOR $Precision$, R IS FOR $Recall$, F IS FOR $F-score$

database	MEME			Sampler			CRMD		
	P	R	F	P	R	F	P	R	F
ABS	0.10	0.21	0.13	0.15	0.10	0.11	0.18	0.15	**0.16**
SCPD	0.10	0.05	0.05	0.29	0.19	0.23	0.31	0.26	**0.28**

$[6, 26]$ and $[7, 26]$ in ABS and SCPD, respectively, which are actually the ranges of the widths of the true motif instances in ABS and SCPD, respectively. Because we did not use the actual motif width, the tolerance in Eq. 6 was relaxed to six bps. Table VII shows the average results of CRMD, MEME and Motif Sampler on the ABS and the SCPD database. For the ABS dataset, CRMD has higher $Precision$ while lower $Recall$ rate than MEME, and still it has the highest $F-score$. For the SCPD dataset, CRMD has the highest $Precision$, $Recall$ and $F-score$.

3) E. coli and Tompa datasets: We have also tested CRMD on the *Escherichia coli (E. coli)* [3] and the Tompa [4] datasets, which were collected and setup as the benchmark problems for testing motif discovery algorithms. The *E. coli* dataset is of prokaryotic data. It consists of 62 motifs of a variety characteristics, such as the motif width, the number of sites per sequence and the sequence length, etc. The Tompa dataset consists of 56 eukaryotic datasets, covering fly, human, mouse and yeast. The motifs are very weakly conserved in the Tompa dataset, which is by far the most difficult dataset tested in this paper.

The *E. coli* and the Tompa datasets were already tested with other algorithms, including MEME and Motif Sampler, in [3] and [4], respectively. They used different performance evaluation indices other than the $Precision$, $Recall$ and $F-score$. Their performance indices can be categorized on two levels. On the nucleotide level, the performance indices (with the prefix n) are calculated w.r.t. to the number of the nucleotides that the true and the predicted instances overlap. On the site level, the performance indices (with the prefix s) are calculated w.r.t the number of the motif instances that the predicted instances overlaps with the true instances for at least one nucleotide. Suppose on the nucleotide level, we have nTP (true positive) as the number of true motif nucleotides correctly predicted, nTN (true negative) as the number of true background nucleotides not predicted, nFP (false positive) as the number of falsely predicted motif nucleotides and nFN (false negative) as the number of true motif nucleotides not predicted. Eq 7 defines the nucleotide level performance indices. The site level ones are similarly defined by replacing all the "n" with "s" in Eq. 7. The original papers [3][4] have more details on the definitions of their performance indices.

$$\begin{aligned} nSn &= nTP/(nTP+nFN) \\ nSp &= nTP/(nTP+nFP) \\ nPC &= nTP/(nTP+nFP+nFN) \\ nF &= (2\times nSn\times nSp)/(nSn+nSp) \end{aligned} \quad (7)$$

TABLE VIII
THE AVERAGE PERFORMANCE OF MEME, MOTIF SAMPLER AND CRMD ON THE E. COLI DATASETS. EACH ALGORITHM OUTPUTS FIVE MOTIFS, AND THE ONE OF THE BEST nPC AMONG THE FIVE OUTPUTS IS RECORDED AS THE RESULT. THE LAST COLUMN REPORTS THE nPC OF THE TOP-SCORED MOTIF IN TERMS OF THE SCORE FUNCTION USED IN THE INDIVIDUAL ALGORITHM

algorithm	nucleotide level				site level				best
	nPC	nSn	nSP	nF	sPC	sSn	sSp	sF	nPC
MEME	0.158	0.259	0.199	0.225	0.295	0.461	0.436	0.448	0.116
Sampler	0.153	0.179	0.237	0.204	0.302	0.331	0.476	0.390	0.069
CRMD	**0.286**	0.321	0.412	**0.346**	**0.459**	0.531	0.625	**0.558**	**0.221**

TABLE IX
THE AVERAGE RESULTS OF MEME, MOTIF SAMPLER AND CRMD ON THE TOMPA DATASET. $xPPV$ IS $xTP/(xTP+xFP)$ FOR BOTH NUCLEOTIDE AND THE SITE LEVELS, AND $sASP$ IS $(sSn+sPPV)/2$

algorithm	nSn	nPPV	nPC	sSn	sPPV	sASP
MEME	0.067	0.107	0.043	0.111	0.139	**0.125**
Sampler	0.060	0.107	0.040	0.098	0.101	0.100
CRMD	0.091	0.088	**0.047**	0.141	0.108	**0.125**

For the $E.coli$ dataset, each algorithm is required to output five motifs, and the one with the best nPC, is recorded as the result. The widths of the motifs vary from problem to problem, and thus CRMD used 15 as the fixed common width for all the problems, as used by other algorithms in the original paper [3]. Table VIII shows the average results of CRMD, MEME and Motif Sampler on all the datasets, where the results of MEME and Motif Sampler are quoted from the paper [3]. On all the nucleotide level and site level performance indices, CRMD has around 10 percentage better results. More surprisingly as indicated in the last column (best nPC), if CRMD outputs a single motif, its performance is already better than the best of the five outputs of MEME and Motif Sampler.

For the Tompa dataset, the algorithms are permitted to fine tune the parameters and report the best result. Since we did not know the exact widths of the motifs in the datasets, we ran CRMD with a series of widths from 10 to 15. For each width, we output 10 motifs (without the similarity test), and so we had a total of 60 motifs. Among the 60 motifs, we calculated the similarity between each pair of motifs in terms of the χ^2 distance [28], and we selected the motif which has the largest number of similar motifs as the result. Table IX shows the average results of MEME, Motif Sampler and CRMD on the 56 Tompa datasets, where the results of MEME and Motif Sampler are quoted from the paper [4]. Generally, the sensitivity of CRMD is slightly better, and the specificity of CRMD is slightly worse, which result in marginally better performance coefficient. As the motifs in most of the Tompa datasets are very weakly conserved, the algorithms usually predict no correct results on both nucleotide and site levels on those datasets. Therefore, the average performance indices of the three algorithms are pretty low, which shows that the *de novo* motif discovery on real datasets of complex organisms is still difficult for the current algorithms with often more than questionable results.

C. Multiple Motif Dataset

To test the capability of CRMD for multiple motif discovery, we have also tested it on the liver-specific dataset [36], which contains multiple motifs. Biological experiments verified that the liver-specific gene expression is controlled by the combined action of a small set of TFs, primarily HNF-1, HNF-3, HNF-4 and C/EBP. The dataset contains 19 sequences and annotates 60 binding sites belonging to ten motifs. However, three motifs have only one instance each, and three other motifs have only two instances each. These six motifs are supposed to be very difficult to find due to the extreme low signal-to-noise ratio. The rest four motifs have 19, 13, 13 and 11 instances, respectively, among which three motifs have less instances than the sequences. The widths of the motifs vary from 6 to 31, and even the motif instances of the same motif may have different lengths.

TABLE X
THE RESULTS OF MEME, MOTIF SAMPLER AND CRMD ON THE LIVER-SPECIFIC DATASET OF MULTIPLE MOTIFS. EACH PROGRAM IS EXECUTED TWICE WITH FIVE AND TEN OUTPUTS, RESPECTIVELY. P IS FOR $Precision$, R IS FOR $Recall$, F IS FOR $F-score$

#output	GAME			MEME			Sampler			CRMD		
	P	R	F	P	R	F	P	R	F	P	R	F
5	0.27	0.33	0.3	0.31	0.18	0.23	0.34	0.17	0.23	0.46	0.5	0.48
10	0.32	0.6	0.42	0.30	0.23	0.36	0.40	0.18	0.25	0.44	0.78	0.56

We have run GAME, MEME, Motif Sampler and CRMD on the liver-specific dataset, while GALF-P is incapable of handling multiple motif discovery problem. We used the average width 15 in all the experiments. We carried out two sets of experiments. One was with five outputs to account for the four motifs with most TFBSs, and the other was with ten outputs to account for the motifs with fewer TFBSs as well. The prediction tolerance is relaxed to six bps because the motifs are very weakly conserved.

Table X shows the results of the four programs in the two sets of experiments of five outputs and ten outputs, respectively. The $Precision$s in all the experiments are lower than 0.50, and CRMD is the only one whose $Precision$s are higher than or equal to 0.44. The low $Precision$s indicate that there are many false positives in the results. This is expected since we had to output more predicted motifs than the true ones due to the low signal-to-noise ratio. MEME and Motif Sampler have very low $Recall$s in both 5-output and 10-output experiments. When the number of the outputs is increased from 5 to 10, the $Recall$s of GAME and CRMD are increased significantly as more binding sites are correctly predicted. CRMD has the highest $Precision$s and $Recall$s in both experiments, and so its $F-score$s are also the highest. The advantage of the $F-score$s of CRMD over the $F-score$s of GAME, which are the second highest, is more than 10 percentages.

VI. CONCLUSION

In this paper, we have proposed a novel approach for motif discovery, i.e., Cluster Refinement Algorithm for Motif Discovery (CRMD). CRMD uses the Entropy-based Clustering to find good initial motif candidates first, and then it uses the Greedy Refinement to find the local optima of the initial candidates. CRMD searches for motifs by maximizing the posterior probabilities of the motif instances. The posterior

probability allows a variable number of motif instances and it requires little prior knowledge of motifs. The Entropy-based Clustering partitions all the subsequences of DNA sequences into clusters of maximal relatively information entropies, and thus clustering alone has already maximized part of the posterior probability. The number of the clusters is much smaller than the number of all the subsequences, and so the computation cost is significantly reduced. The Greedy Refinement finds the local optimal binding sites given the initial clusters. It selects the motif instances of maximal probabilities deterministically without taking extra time in sampling subsequences probabilistically. It also automatically removes or adds motif instances according to the thresholds which change adaptively following the distribution of the current motif instances. If the prior knowledge of OOPS is available, CRMD can enhance its prediction performance further by searching for OOPS consistent solutions only and adjusting the number of motif instances later on. For multiple motif problem, CRMD measures the similarities among the candidate motifs using the χ^2 homogeneity test, and thus it is able to keep only distinct motifs of high probabilities.

To compare it to other state-of-the-art algorithms, CRMD has been tested extensively on both synthetic and comprehensive real datasets of single and multiple motifs. As observed from the empirical results, CRMD is very competitive, and often the best among the testing algorithms. The synthetic data are generated with a variety of properties and difficulties. CRMD has achieved a good balance between $Precision$ and $Recall$, and thus obtained the highest $F-scores$ on most of the synthetic problems. The real datasets tested are comprehensive. On the eight real datasets selected by GAME [7] and GALF-P [8], CRMD still has the highest $F-scores$ on most of the problems, and its average $Precision$, $Recall$ and $F-score$ are the highest as well. With the OOPS assumption, the performance of CRMD is further enhanced, and its results are better than or comparable to those of the other two algorithms. On other four databases, i.e., the ABS database [1], the SCPD database [2], the *Escherichia coli* dataset [3] and the Tompa dataset [4], CRMD has also achieved the best performance in terms of either of our default metrics or of the nucleotide and site level metrics used in [4] and [3]. For the liver-specific dataset of multiple motifs, CRMD identifies significantly more binding sites than the other multiple motif discovery approaches.

ACKNOWLEDGEMENT

The anonymous reviewers of this paper provided valuable comments and suggestions. Ni Bing in our group helped collect and organize the testing SCPD datasets. This research is partially supported by the grants from the Research Grants Council of the Hong Kong SAR, China (Projects CUHK414107 and CUHK414708).

REFERENCES

[1] E. Blanco, D. Farré, M. M. Albà, X. Messeguer, and R. Guigó, "ABS: a database of annotated regulatory binding sites from orthologous promoters," *Nucleic Acids Research*, vol. 34, no. Database-Issue, pp. 63–67, 2006.

[2] J. Zhu, "SCPD: a promoter database of the yeast Saccharomyces cerevisiae," *Bioinformatics*, vol. 15, no. 7, pp. 607–611, 1999.

[3] J. Hu, B. Li, and D. Kihara, "Limitations and potentials of current motif discovery algorithms," *Nucleic Acids Research*, vol. 33, no. 15, pp. 4899–4913, 2005.

[4] M. Tompa, N. Li, and T. Bailey, "Assessing computational tools for the discovery of transcription factor binding sites," *Nature Biotechnology*, vol. 23, pp. 137–144, 2005.

[5] T. L. Bailey and C. Elkan, "Fitting a mixture model by expectation maximization to discover motifs in biopolymers," in *Proceedings of the Second International Conference on Intelligent Systems for Molecular Biology*, 1994, pp. 28–36.

[6] J. S. Liu, A. F. Neuwald, and C. E. Lawrence, "Bayesian models for multiple local sequence alignment and Gibbs sampling strategies," *J. American Statistical Association*, vol. 90, no. 432, pp. 1156–??, 1995.

[7] Z. Wei and S. T. Jensen, "GAME: detecting cis-regulatory elements using a genetic algorithm," *Bioinformatics*, vol. 22, no. 13, pp. 1577–1584, 2006.

[8] T.-M. Chan, K.-S. Leung, and K.-H. Lee, "Tfbs identification based on genetic algorithm with combined representations and adaptive post-processing," *Journal of Bioinformatics*, vol. 24, p. 341, 2007.

[9] D. J. Galas and A. Schmitz, "DNAse footprinting: a simple method for the detection of protein-DNA binding specificity," *Nucleic Acids Res.*, vol. 5, no. 9, pp. 3157–3170, September 1987.

[10] C. HORAK and M. SNYDER, "ChIP-chip: A genomic approach for identifying transcription factor binding sites," *Methods in enzymology*, vol. 350, pp. 469–483, 2002.

[11] G. Sandve and F. Drablos, "A survey of motif discovery methods in an integrated framework," *Biology Direct*, vol. 1, no. 11, 2006.

[12] M. Li, B. Ma, and L. Wang, "Finding similar regions in many sequences," *Journal of Computer and System Sciences*, vol. 65, pp. 73–96, 2002.

[13] P. Pevzner and S. Sze, "Combinatorial approaches to finding subtle signals in dna sequences," in *Eighth International Conference on Intelligent Systems for Molecular Biology*, 2000, pp. 269–278.

[14] M. F. Sagot, "Spelling approximate repeated or common motifs using a suffix tree," in *LATIN '98: Theoretical Informatics, Lecture Notes in Computer Science*. Springer-Verlag, 1998, pp. 111–127.

[15] P. Bieganski, J. Riedl, J. V. Carlis, and E. Retzel, "Generalized suffix trees for biological sequence data: applications and implementations," in *Proc. of the 27th Hawaii Int. Conf. on Systems Sci.*, 1994, pp. 35–44.

[16] J. Buhler and M. Tompa, "Finding motifs using random projections," in *RECOMB*, 2001, pp. 69–76.

[17] B. Raphael, L. Lung-Tien, and G. Varghese, "A uniform projection method for motif discovery in dna sequences," *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, vol. 1, no. 2, pp. 91–94, 2004.

[18] K. Blekas, D. Fotiadis, and A. Likas, "Greedy mixture learning for multiple motif discovery in biological sequences," *Bioinformatics*, vol. 19, no. 5, pp. 607–617, 2003.

[19] J. S. Liu, A. F. Neuwald, and C. E. Lawrence, "Bayesian models for multiple local sequence alignment and Gibbs sampling strategies," *J. Am. Stat. Assoc.*, vol. 90, no. 432, pp. 1156–1170, November 1995.

[20] C. E. Lawrence, S. F. Altschul, M. S. Boguski, J. S. Liu, A. F. Neuwald, and J. C. Wooton, "Detecting subtle sequence signals: a Gibbs sampling strategy for multiple alignment," *Science*, vol. 262, no. 8, pp. 208–214, October 1993.

[21] G. B. Fogel, D. G. Weekes, G. Varga, E. R. Dow, H. B. Harlow, J. E. Onyia, and C. Su, "Discovery of sequence motifs related to coexpression of genes using evolutionary computation," *Nucleic Acids Res.*, vol. 32, no. 13, pp. 3826–3835, 2004.

[22] M. Lones and A. Tyrrell, "Regulatory Motif Discovery Using a Population Clustering Evolutionary Algorithm," *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, pp. 403–414, 2007.

[23] G. Li, T. Chan, K. Leung, and K. Lee, "An Estimation of Distribution Algorithm for Motif Discovery," in *Evolutionary Computation, 2008. CEC 2008.(IEEE World Congress on Computational Intelligence). IEEE Congress on*, 2008, pp. 2411–2418.

[24] M. K. Das and H.-K. Dai, "A survey of DNA motif finding algorithms," *BMC Bioinformatics*, vol. 8, no. S21, 2007.

[25] S. Hannenhalli, "Eukaryotic transcription factor binding sites vmodeling and integrative search methods," *Bioinformatics*, vol. 24, no. 11, pp. 1325–1331, 2008.

[26] S. Jensen, X. Liu, Q. Zhou, and J. Liu, "Computational discovery of gene regulatory binding motifs: a Bayesian perspective," *Statistical Science*, vol. 19, no. 1, pp. 188–204, 2004.

[27] Z. Qin, L. McCue, W. Thompson, L. Mayerhofer, C. Lawrence, and J. Liu, "Identification of co-regulated genes through Bayesian clustering of predicted regulatory binding sites," *Nature Biotechnology*, vol. 21, pp. 435–439, 2003.

[28] S. Kielbasa, D. Gonze, and H. Herzel, "Measuring similarities between transcription factor binding sites," *BMC Bioinformatics*, vol. 6, no. 1, p. 237, 2005.

[29] G. Stormo, G. Hartzell, *et al.*, "Identifying Protein-Binding Sites from Unaligned DNA Fragments," *Proceedings of the National Academy of Sciences of the United States of America*, vol. 86, no. 4, pp. 1183–1187, 1989.

[30] C. Lawrence and A. Reilly, "An expectation maximization (EM) algorithm for the identification and characterization of common sites in unaligned biopolymer sequences." *Proteins*, vol. 7, no. 1, pp. 41–51, 1990.

[31] J. Liu, "The Collapsed Gibbs Sampler in Bayesian Computations With Applications to a Gene Regulation Problem," *Journal of the American Statistical Association*, vol. 89, no. 427, pp. 958–966, 1994.

[32] C. Klinge, "Estrogen receptor interaction with estrogen response elements." *Nucleic Acids Research*, vol. 29, no. 14, p. 2905, 2001.

[33] A. Kel, O. Kel-Margoulis, P. Farnham, S. Bartley, E. Wingender, and M. Zhang, "Computer-assisted identification of cell cycle-related genes: new targets for E2F transcription factors," *Journal of Molecular Biology*, vol. 309, no. 1, pp. 99–120, 2001.

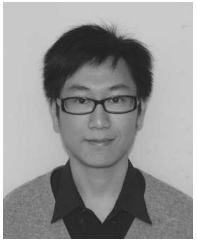
[34] B. Berman, Y. Nibu, B. Pfeiffer, P. Tomancak, S. Celniker, M. Levine, G. Rubin, and M. Eisen, "Exploiting transcription factor binding site clustering to identify cis-regulatory modules involved in pattern formation in the Drosophila genome," *Proceedings of the National Academy of Sciences*, vol. 99, no. 2, pp. 757–762, 2002.

[35] M. Frith, U. Hansen, J. Spouge, and Z. Weng, "Finding functional sequence elements by multiple local alignment." *Nucleic Acids Research*, vol. 32, no. 1, p. 189, 2004.

[36] W. Krivan and W. Wasserman, "A Predictive Model for Regulatory Sequences Directing Liver-Specific Transcription," *Genome Research*, vol. 11, no. 9, p. 1559, 2001.

Gang Li received the B.E. degree in Computer Science from Wuhan University, P. R. China, in 2002. He is currently a Ph.D. candidate in the Chinese University of Hong Kong, Hong Kong SAR.

His research interests include Evolutionary Computation, Independent Component Analysis, and Bioinformatics.



Tak-Ming Chan received his B.Sc. degree in Computer Science from Fudan University, P. R. China, in 2006. He is currently a Ph.D. candidate in the Chinese University of Hong Kong, Hong Kong SAR.

His research interests include Motif Discovery, Evolutionary Computation and Data Mining.

Kwong-Sak Leung (M'77-SM'89) received his BSc (Eng.) and PhD degrees in 1977 and 1980, respectively, from the University of London, Queen Mary College. He worked as a senior engineer on contract R&D at ERA Technology and later joined the Central Electricity Generating Board to work on nuclear power station simulators in England. He joined the Computer Science and Engineering Department at the Chinese University of Hong Kong in 1985, where he is currently Professor of Computer Science & Engineering.

Dr Leung's research interests are in soft computing and bioinformatics including evolutionary computation, parallel computation, probabilistic search, information fusion and data mining, fuzzy data and knowledge engineering. He has authored and co-authored over 200 papers and 2 books in fuzzy logic and evolutionary computation. He has been chair and member of many program and organizing committees of international conferences. He is in the Editorial Board of Fuzzy Sets and Systems and an associate editor of International Journal of Intelligent Automation and Soft Computing. He is a senior member of the IEEE, a chartered engineer, a member of IET and ACM, a fellow of HKIE and a distinguished fellow of HKCS in Hong Kong.

Kin Hong Lee got his degrees in Computer Science from the University of Manchester. Before he joined The Chinese University of Hong Kong in 1984, he had worked for Burroughs machines in Scotland and ICL in Manchester. He is now an Associate Professor of the Computer Science & Engineering Department at the Chinese University of Hong Kong.

His current interests are Computer hardware and Bio-informatics. He has published over 60 papers in these two fields.