Discovery of Non-induced Patterns from Sequences

Andrew K.C. Wong, Dennis Zhuang, Gary C.L. Li, and En-Shiun Annie Lee

Department of System Design University of Waterloo,
200 University Avenue West, Waterloo, Ontario, Canada
akcwong@pami.uwaterloo.ca, dennizeh@gmail.com,
gclli@pami.uwaterloo.ca, ealee@engmail.uwaterloo.ca

Abstract. Discovering patterns from sequence data has significant impact in genomics, proteomics and business. A problem commonly encountered is that the patterns discovered often contain many redundancies resulted from fake significant patterns induced by their strong statistically significant subpatterns. The concept of statistically induced patterns is proposed to capture these redundancies. An algorithm is then developed to efficiently discover non-induced significant patterns from a large sequence dataset. For performance evaluation, two experiments were conducted to demonstrate a) the seriousness of the problem using synthetic data and b) top non-induced significant patterns discovered from Saccharomyces cerevisiae (Yeast) do correspond to the transcription factor binding sites found by the biologists. The experiments confirm the effectiveness of our method in generating a relatively small set of patterns revealing interesting, unknown information inherent in the sequences.

Keywords: Sequence Pattern Discovery, Statistically Induced Patterns, Suffix Tree.

1 Introduction

Sequence data is a very significant type of data in many forms: biological sequence, web click stream, custom purchase history, event sequence, etc. A vast amount of such data from the genomic, proteomic and business arenas has been collected. The discovery of new interesting knowledge from these data has important applications and great value.

Many approaches have been developed to discover patterns from sequences. One common problem encountered is that the quality of the output patterns is overlooked resulting in an overwhelming number of output patterns [1]. To reduce the output size, some methods [1] [2] identify the redundancy among output patterns and discover those irredundant patterns. Others [3] [4] [5] [6] use statistical hypothesis test to extract and rank statistically significant patterns based on how much the frequency of a pattern deviates from the expected one by assuming a background random model. It is hoped that patterns occurring with significantly higher frequency will correspond to the functional units inherent in the sequences. However, some of them are fake or statistically redundant patterns which are considered as significant merely because they contain very strong subpatterns [7]. This problem is exaggerated in dense datasets containing many strong patterns.

T.M.H. Dijkstra et al. (Eds.): PRIB 2010, LNBI 6282, pp. 149–160, 2010.

In this paper, we present the concept of statistically induced patterns to caputure these fake patterns and an efficient algorithm based on generalized suffix tree to discover statistically non-induced patterns. By removing induced patterns, the quality of output patterns can be further improved and the ranking of important functional units can be elevated. Our method is scalable to handle very large sequence data rendering a more compact ouput. Though our method provides a general data mining framework, here we focus specifically on biological data (transcription binding site data).

2 Related Work

Pattern discovery techniques or motif-finding algorithms have evolved in a fast pace in bioinformatics. This is driven by the rapid growth of available DNA and protein sequence database as well as the strong desire to find functional units such as regulation signals embedded in biological sequences. Pattern discovery techniques are developed to reveal such conserved patterns across sequences. In motif finding, two main perspectives are adopted: the probabilistic and the combinatorial. The former uses the profile-based position weight matrix (PWM) to find the location of the motifs in the sequences [8] [9]. Thus, the best motif is the most probable PWM. In the latter, a motif is defined as a consensus that occurs repeatedly in sequences [3] [4] [5] [6] [10]. The problem of producing overwhelming number of patterns is often encountered in the latter approaches.

Extracting statistically significant patterns is one way of shrinking output size. A linear time algorithm is presented in [4] to detect statistically significant patterns (overrepresented δ-significant patterns) which are represented by the internal nodes of the suffix tree. Statistics such as mean and variance are efficiently annotated to each node. Observing that not all nodes in the suffix tree correspond to overrepresented δ-significant pattern, we develop a method to identify those that are not.

Although extracting only statistically significant patterns greatly reduces the output size, often these patterns form a large set of patterns with real motifs mixed with many of their random variations [7]. A background model of order 3 Markov chain and a greedy algorithm have been proposed to separate the artifacts from the real motifs. Our definition of statistically induced patterns is a variation of Blanchette and Sinha's [7] artifact motifs. However, ours uses the Bernoulli scheme instead of markov chain and hence does not require training a complex Markov model. Furthermore, our method in discovering non-induced patterns is more efficent.

3 Methodology

3.1 Preliminary Definitions

Let Σ be a set of distinct elements $\{e_1, e_2, \ldots, e_{|\Sigma|}\}$, called the alphabet, and $|\Sigma|$ be its size. A sequence S over Σ is an ordered list of elements $s_1 s_2 \ldots s_n$. A pattern P is a short sequence $p_1 p_2 \ldots p_m$ over Σ and $|P|$ is its order. We call P a consecutive pattern with no gaps. In general, the input data might come as multiple sequences $S_1, S_2, \ldots, S_N$ with lengths $l_1, l_2, \ldots, l_N$ respectively. Let L be their overall length.

The *number of occurrences* of P in multiple sequences is denoted by k_P. The list of occurrence positions is $L_P = \{\ldots, (i,j), \ldots\}$ where the ordered pair (i,j) is a position denoting that P occurs at position j in sequence i. The *support* of P denoted by q_P in the multiple sequences is the number of sequences in which P occurs at least once. A pattern is said to be *frequent* if its number of occurrences is not less than a specified minimum requirement min_{occ}. Mathematically, that is $k_p \geq min_{occ}$.

3.2 Statistical Model for Input Sequences

A background random model for determining the expected frequency of P is needed to define statistically significant patterns. Without being given specific domain knowledge for the background model, we adopt a simplest model: the Bernoulli scheme. With this scheme, the probability of a pattern P occurring in a position of a random sequence is $pr(P) = \prod_{i=1}^{m} pr(p_i)$, where $p_i \in \Sigma$. Let X_i be a Bernoulli variable that indicates whether P occurs in position i of a random sequence. The total number of possible positions is $T_P = \sum_{i=1}^{n}(l_i - m + 1)$, so the number of occurrences of P is a random variable $X_P = \sum_i X_i$ which follows a binominal distribution. Its expected number of occurrences is $E(X_P) = pr(P) \cdot T_P$.

Definition 1. Statistically significant pattern
To measure how k_P of P deviates from its expected frequency if the given sequences are generated from the random model, we use the *standard residual* [11]

$$z_P = \frac{k_P - E(X_P)}{\sqrt{E(X_P)}}$$

A pattern is *statistically significant or overrespresented* [12] if $z_P \geq t$ where t is the predefined minimum threshold.

Definition 2. Significant representative pattern
As observed in the paper [4] [16], patterns can be clustered into equivalence groups $C_1, C_2, ..., C_K$ such that patterns in the same group C_i have the same list of occurrence positions L_P.

Representative pattern is the pattern in the group C_i that has the highest statistical significance z_P or equivalently has the highest order. *Significant representative pattern* is both statistically significant and representative.

Definition 3. Statistically induced pattern
Let P' be a subpattern of P. The *conditional statistical significance* of P given P' is defined as

$$z_{P|P'} = \frac{k_P - E(X_P|P')}{\sqrt{E(X_P|P')}}$$

where $E(X_P|P') = pr(P|P') \cdot k_{P'} = \frac{pr(P)}{pr(P')} \cdot k_{P'}$.

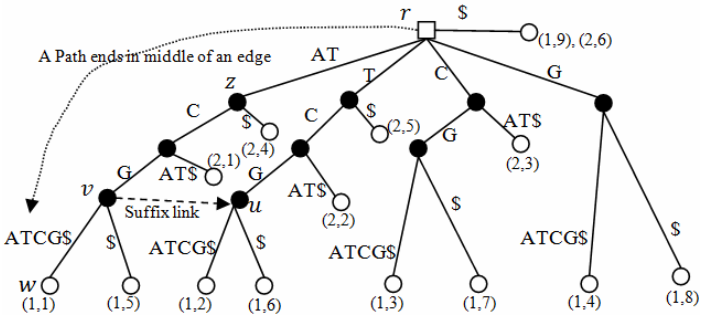

Fig. 1. Generalized suffix tree T for multiple strings $S_1 = \text{ATCGATCG\$}$ and $S_2 = \text{ATCAT\$}$. The square node is the root, the solid circles are the internal nodes and the hollow circles denote the leaf nodes. r, u, v, w and z are the nodes in the suffix tree. Edges are labelled with substrings. The dotted arrow shows the end point "T" for a path ending inside an edge. $(1, 1)$ is a position.

Given a set of significant representative patterns, a pattern P in it is said to be *statistically induced* if there exists a *proper* subpattern P' of P such that $z_{P|P'} < t$. A proper subpattern P' is not statistically induced.

This conditional statistical significance is used to evaluate how strongly the statistical significance of a pattern is attributed by the occurrences of one of its proper subpatterns. Those induced patterns whose significances are due to their proper subpatterns by mere chance are fake patterns. Hence removing them would render a more succinct set of patterns.

3.3 Characterizing Significant Representative Patterns in Generalized Suffix Tree

First, we introduce the generalized suffix tree as the data structure for representing strings. It can be constructed in $O(L)$ time and space. The details of it and its linear time and space construction algorithms can be found in [13]. Here we establish the connection between consecutive patterns and path labels in the suffix tree. Finally, we link significant representative patterns with nodes in the suffix tee.

Generalized Suffix Tree

Given a collection of strings $S_1, S_2, \ldots, S_N$ over Σ, the generalized suffix tree T for these multiple strings is a rooted directed tree with the following properties:

(1) Each leaf node is labelled by a set of positions $\{\ldots, (i, j), \ldots\}$ where (i, j) indicates a suffix of string S_i starting at the position j.
(2) Each internal node has at least two outgoing edges each of which is labelled with a non-empty substring of one of the input string. No two edges out of a node can have the edge-label starting with the same character.

Most often, a termination character $\$ \notin \Sigma$ is appended to each string to ensure that T exists for this set of multiple strings. Fig. 1 gives an example for two input strings.

Consecutive Pattern in Suffix Tree
The label of the path from root ending at node v is the string resulted from concatenation of the substrings that label the edges along that path. The label of a path from root ending inside an edge (v, w), is the label of the path from the root ending at node v, concatenating with the remaining characters of the label of the edge (v, w) down to the end of the path. For convenience, the label of a path ending at node v is represented by $pl(v)$, the path label of v. In Fig. 1, the path label of node u is the substring TCG of S_1. The label of a path ending in the middle of the edge (v, w) with end point indicated as "T" is the substring ATCGAT of S_1. Accordingly, a consecutive pattern that occurs at least once in the input strings has its unique path in suffix tree and is represented by the path label.

Frequent Pattern in Suffix Tree
The number of occurrences of a consecutive pattern is the number of positions found under its path in the suffix tree. For example, positions $\{(1, 1), (1, 5), (2, 1)\}$ are found under the path of pattern ATC. By storing into each node x the number of positions $k(x)$ in the subtree rooted by it, the number of occurrences of a consecutive pattern can be easily obtained by finding the node at or above which its path ends. For example, the number of occurrences of pattern TCG whose path ends at u is given by $k(u) = 2$. Hence frequent patterns are represented as labels of paths that end at or above a node x where $k(x) >= min_{occ}$.

Representative Pattern in Suffix Tree
Note that the paths of representative patterns end at nodes instead of within edges. A pattern with path ending within an edge can be further extended by at least one character to the right without decreasing the number of occurrences and thus by definition cannot be a representative pattern. For example, the pattern A, which ends inside the edge (r, z), has a superpattern AT ending at node z with the same number of occurrences indicated by $k(z) = 4$.

However, it is not a one-to-one mapping; not all nodes correspond to representative patterns. As we might notice that the pattern associated with the path ending at the node u in Fig. 1, $pl(u) = \text{TCG}$, is not a representative pattern because it has a superpattern $pl(v) = \text{ATCG}$ with the same number of occurrences as indicated by $k(u) = k(v) = 2$ which has higher statistical significance. Hence, we need to identify nodes corresponding to representative patterns in the tree. This can be efficiently achieved by utilizing the suffix links. A suffix link of v points to u if $pl(v)$ is an one character left extension of $pl(u)$. The node u is called the suffix node of v because $pl(u)$ is the suffix of $pl(v)$. For example, $pl(v) = \text{ATCG}$ is a string by appending A to the left of $pl(u) = \text{TCG}$. Only one suffix link is shown in Fig. 1.

In summary, a representative pattern corresponds to a node x in a suffix tree that is not a suffix node of the other node. A representative pattern $pl(x)$ is statistically significant if $z_{pl(x)} \geq t$. For example, {ATCG, TCG, CG} forms an equivalence set and the representative pattern is ATCG and hence corresponds to the node v that does not have suffix link pointing to it.

3.4 Discovering Non-induced Patterns

Although significant representative patterns contain fewer redundancies, they could still be too many for human experts to interpret. As noted in **Definition 3**, some significant representative patterns can be statistically induced by others and should be removed. Here we describe an efficent algorithm to find non-induced patterns.

If pattern P is not induced by the proper subpattern P' that has the smallest conditional statistical significance $z_{P|P'}$ among all proper subpatterns of P, where we denotes P' as the valid pattern for P, then P is not statistically induced. In other word, if P is non-induced, then $z_{P|P'} >= t$ for any proper subpattern P' of it, including the one with smallest $z_{P|P'}$. Thus, we develop **Algorithm 1** to efficiently discover non-induced patterns by identifying the valid pattern for each significant representative pattern from the lowest to highest order. Note that each representative pattern corresponds to a node in the suffix tree.

Algorithm 1. Discovery of non-induced patterns

1. Construct a generalized suffix tree T for the input sequences
2. Annotate $k(v)$ the number of positions under each node v of T
3. Extract a set of nodes whose $k(v) \geq min_{occ}$
4. Sort the above nodes in ascending order according to order of $pl(v)$ using counting sort.
5. For each node v
   a) Find the valid node w for v using **Procedure 1**
   b) If v is not a suffix node and $z_{pl(v)} \geq t$ and $pl(v)$ is not induced by $pl(w)$
      Output $pl(v)$
   End if
   End for

Procedure 1. Find valid node for v

1. Let v_S and v_P be the suffix node and parent node of v respectively
2. If $pl(v_S)$ is non-induced
   Let w_1 be v_S
3. Else
   Let w_1 be the valid node of v_S
   End if
4. If $pl(v_P)$ is non-induced
   Let w_2 be v_P
5. Else
   Let w_2 be the valid node of v_P
   End if
6. Pick one node with the smallest conditional statistical significance out of w_1 and w_2 to be the proper node of v

Running time analysis for Algorithm 1. Step 1-3 can be achieved in linear time. Step 4 uses counting sort to sort the nodes according to the path length and can thus be done also in linear time. Steps 5a and 5b take constant time. Step 5 can be done in $O(L)$ time. Therefore, non-induced patterns can be found in linear time.

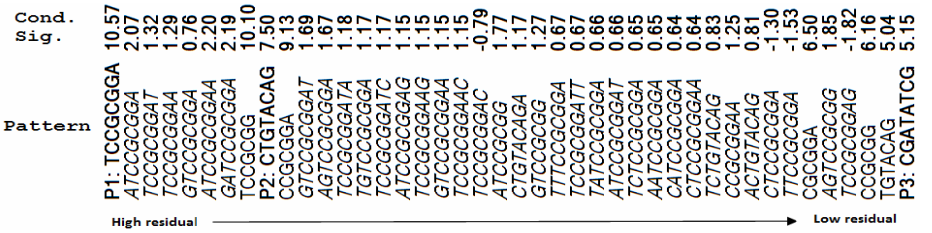

Fig. 2. Patterns reported by Method 1. Patterns ranked higher than P_3 by standard residual are shown. Cond. Sig. is the short form of conditional statistical significance.

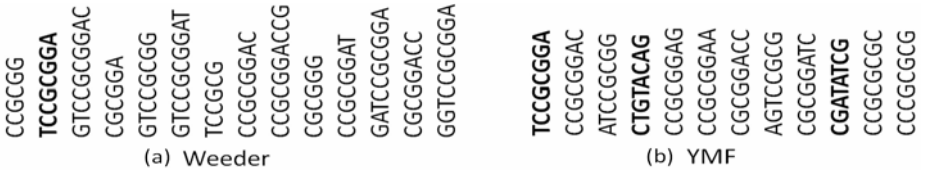

Fig. 3. Highest ranking patterns discovered by Weeder and YMF. The implanted patterns are bolded.

4 Experimental Results

Two sets of experiments were conducted for performance evaluation. In both experiments, min_{occ} is set to 5 and the statistical threshold t is set to 3.

4.1 Experiment on Synthetic Data

To show that our method could remove statistically induced patterns and raise the ranking of the truly significant patterns, 100 random sequences of 1000 bases over DNA alphabet are created. Three strong patterns $P_1 = \mathrm{TCCGCGGA}$, $P_2 = \mathrm{CTGTACAG}$ and $P_3 = \mathrm{CGATATCG}$ are implanted into these sequences such that their standard residuals z_1, z_2 and z_3 are 48, 24, and 12 respectively. We apply Method 1 to discover significant representative patterns and Method 2 to obtain non-induced patterns.

Fig. 2 shows the patterns of P_1, P_2 and P_3 and their superpatterns and subpatterns which are ranked higher than P_3 according to their standard residual. Those patterns in italic font are superpatterns induced by P_1, P_2 and P_3 (indicated by their conditional statistical significance, i.e. less than the prescribed threshold of 3), hence they are removed by method 2. After these induced patterns are removed, the ranking of P_3 according to the standard residual is raised from the 42th to the 7th. The number of patterns reported by method 1 is 527 while the number of patterns reported by method 2 is reduced to 315 with a 40.2% reduction rate. Hence, as anticipated, a more compact set of patterns is obtained by ensuring that patterns are non-induced and the rankings of the real significant patterns are elevated.

Two well known motif finding tools YMF [3] and Weeder [10] are applied to the synthetic data respectively for comparison. For Weeder, medium mode is selected. For YMF, no spacers and degenerate symbols are allowed and the motif length is fixed to 8. Both methods require specifying the organisim from which the input sequences are obtained. Since the synthetic data does not come from any real organisim, we choose an arbitary organism Saccharomyces cerevisiae (SC) for both methods. The highest ranking patterns discovered by Weeder and YMF are shown in Fig. 3. Most discovered patterns are related to the strong implanted pattern P_1 and can be considered as induced patterns. P_1 is among the top 15 patterns from Weeder while YMF discovered all implanted patterns P_1, P_2 and P_3 within top 10 patterns. Note that YMF has the advantage position by knowing the pattern length in advance. Compared to Weeder and YMF, our method is more general: (1) it searches for patterns with arbitrary length while pattern length is restricted from 6 to 10 for the other two; (2) it does not require background information while the others require specifiying the input organisim. In other words, YMF and Weeder are designed more specifically for TF binding site discovery.

4.2 Experiment on Transcription Factor Binding Sites

We next examine the capability of our method in discovering biological functional units, which is the foremost fundamental step towards understanding the complex mechanism of the gene expression regulations. We apply our method to identify transcription factor (TF) binding sites on Yeast using the widely studied SCPD database [14] with many of its TFs known along with their regulated genes. They are from the upstream (promoter) regions of genes regulated by one or more TFs. Each set of genes is called regulon and is associated with one or more TFs. The genes are believed to be co-regulated by specific TFs and the binding sites for them are experimentally determined in the database. Three conditions are imposed when choosing the regulon: (1) the number of genes in it should be at least three, (2) the consensus binding sites are available, and (3) the number of gaps or "don't care" characters in the consensus should be at most two. The condition (3) is imposed since we discover only consecutive patterns in the current stage. There are totally 18 such regulons. For each regulon of the TF(s), the upstream sequences of genes are extracted from position +1 to -800 relative to the ORF (translation start site).

We design a score combining the statistical significance and support to rank the discovered patterns since the former is based only on its number of occurrences and no information of its support is used. However, to find transcription binding sites amongst multiple sequences, the number of supports is important. These genes are regulated by one or more TFs, and thus we expect that each upstream region of the input gene sequence contains one or more binding sites. Hence, patterns with higher support should be considered more important than those with less support. For example, if we have discovered two patterns TTTAAA and CTTCCT with close statistical residual but different support, say 2 and 7, then the latter will be more important and more likely to correspond to binding sites. Hence, a combined score is defined as $score = \frac{\text{Support}}{\text{No. of genes}} \cdot \text{Standard residual}$.

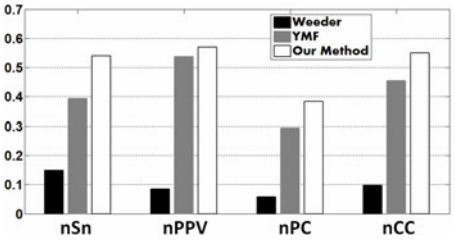

Fig. 4. Combined measures over 18 datasets

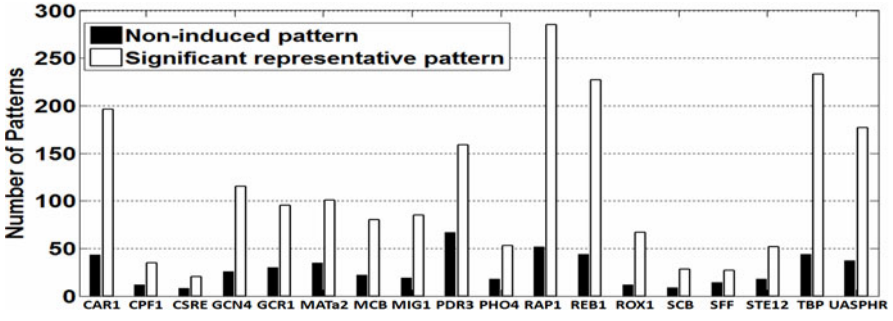

Fig. 5. Number of reported patterns after removing induced patterns among significant representative patterns

In DNA sequences tandem, repeats are common. For example, in a sequence like AAAAAATTTTTT, the pattern AAAA occurs at positions 1, 2 and 3 which overlap multiple times. Hence, a post-processing step is applied to further remove patterns whose occurrences overlap in the original sequences.

We discovered the non-induced patterns for each dataset and compared the result with YMF and Weeder respectively (Table 1). We ranked the discovered patterns according to the combined score and chose the top 15 ones for comparision. For YMF, we used its webserver and obtained 5 best motifs through FindExplanators [7] for motif length from 6 to 8 (all available parameters), resulting a total number of 15 patterns. 0 spacers and 2 degnerate symbols are allowed in the motif definition. For Weeder, we downloaded the standalone platform and used the medium mode. All motifs recommend in the final output are used for comparison. For each motif reported by Weeder, we use only the best occurrences with the percentage threshold greater than 90 as binding site predictions. We use the measures nSn (sensitivity), nPPV (positive preditive value), nPC (performance coefficient) and nCC (correlation coefficient) defined in [15] in comparison.

Among the 18 datasets, our discovered patterns within rank 13 match the consensus binding sites in 14 datasets and 4 of them are ranked top. The patterns in bold do not match the binding sites in the remaining 4 datasets CPF1, CSRE, MATalpha2 and SFF. The reason why our discovered patterns have no match in CPF1, CSRE and SFF is that their consensus binding sites have fewer than 2 occurrences.

As for MATalpha2, the consensus has 6 occurrences, but it has many substitutions. Because our program runs with $min_{occ} = 5$ and is confined to discovering consecutive patterns, these consensuses are not discovered.

The overall performances of Weeder, YMF and our method across 18 datasets are evaluated by the combined measures in [15] and shown in Fig. 4. It indicates that the overall perfromance of our method is better than YMF. Weeder does not perform well comparatively and the reason might be that the percentage threshold 90 is too strict. However, Weeder does not provide a good strategy in choosing this parameter.

Fig. 5 shows the number of reported patterns in terms of non-induced pattern and significant repressentative patterns. After removing induced patterns among significant repressentative patterns, our method produces a relative small set of patterns of which the number of reported patterns ranges from 8 to 67. The result shows that our method is able to retain those patterns associated with conserved functional units in the promoter regions while reducing the number of patterns.

Table 1. Comparison of our method, YMF and Weeder on SPCD datasets (the pattern among the top 15 that achieves the best nSn is used for comparison). IUPAC Nucleotide Code is used.

TF		Motif/Pattern	nSn	nPPV	nPC	nCC	Rank
CAR1	Consensus	AGCCGCCR					
	Weeder	CCTAGCCG	0.23	0.09	0.07	0.14	
	YMF	GCCGCCG	0.7	1	0.7	0.84	
	Our Method	AGCCGCC	0.88	1	0.88	0.94	6
CPF1	Consensus	TCACGTG					
	Weeder	CACGTGGC	0	0	0	-0.01	
	YMF	YCACGWG	1	0.5	0.5	0.71	
	Our Method	**TTC**	**0.29**	**0.01**	**0.01**	**0.04**	**10**
CSRE	Consensus	YCGGAYRRAWGG					
	Weeder	GCGGTCGG	0	0	0	-0.01	
	YMF	CGGATRRA	0.58	0.22	0.19	0.35	
	Our Method	**CCGG**	**0.33**	**0.08**	**0.07**	**0.15**	**1**
GCN4	Consensus	TGANT					
	Weeder	TGACTC	0.07	0.13	0.05	0.08	
	YMF	TGWCTR	0.18	0.51	0.15	0.29	
	Our Method	TGACT	0.34	1	0.34	0.57	13
GCR1	Consensus	CWTCC					
	Weeder	TCTGGCATCC	0.1	0.2	0.07	0.13	
	YMF	TCTYCCY	0.3	0.48	0.23	0.37	
	Our Method	TTCC	0.68	0.39	0.33	0.5	9
MATalpha2	Consensus	CRTGTWWWW					
	Weeder	GGAAATTTAC	0.13	0.14	0.07	0.13	
	YMF	ACGCGT	0	0	0	0	
	Our Method	**GAAAAAAG**	**0**	**0**	**0**	**-0.01**	**1**
MCB	Consensus	WCGCGW					
	Weeder	AGACGCGT	0.19	0.08	0.06	0.1	
	YMF	ACGCGT	0.68	1	0.68	0.82	
	Our Method	ACGCGT	0.68	1	0.68	0.82	1
MIG1	Consensus	CCCCRNNWWWWW					
	Weeder	CCCCAG	0.39	0.1	0.09	0.19	
	YMF	CCCCRS	0.5	0.21	0.18	0.32	
	Our Method	CCCCAG	0.33	0.29	0.18	0.3	2
PDR3	Consensus	TCCGYGGA					
	Weeder	GTCTCCGCGG	0.32	0.14	0.11	0.19	
	YMF	TCCGYGGA	1	1	1	1	

Table 1. (*continued*)

	Our Method	TCCGCGGA	0.64	1	0.64	0.8	1
PHO4	Consensus	CACGTK					
	Weeder	GAAACGTG	0.07	0.02	0.02	0.02	
	YMF	CACGTGSR	0.71	0.75	0.58	0.73	
	Our Method	CACGTG	0.71	1	0.71	0.84	1
RAP1	Consensus	RMACCCA					
	Weeder	AGCACCCA	0.13	0.23	0.09	0.17	
	YMF	CACCCA	0.64	0.86	0.58	0.74	
	Our Method	CACCCA	0.64	0.86	0.58	0.74	8
REB1	Consensus	YYACCCG					
	Weeder	ACCCGC	0.14	0.05	0.04	0.08	
	YMF	TTACCCG	0.7	1	0.7	0.84	
	Our Method	TTACCCG	0.7	1	0.7	0.84	7
ROX1	Consensus	YYNATTGTTY					
	Weeder	CCTATTGT	0.28	0.05	0.04	0.07	
	YMF	TTGTTS	0.48	0.29	0.22	0.35	
	Our Method	ATTGTT	0.6	0.63	0.44	0.6	6
SCB	Consensus	CNCGAAA					
	Weeder	AGTCACGAAA	0.47	0.26	0.2	0.31	
	YMF	CACGAA	0.61	1	0.61	0.78	
	Our Method	CACGAAA	0.71	1	0.71	0.84	1
SFF	Consensus	GTMAACAA					
	Weeder	CTGTTTAG	0.13	0.02	0.02	0.04	
	YMF	TAAWYA	0.38	0.08	0.07	0.17	
	Our Method	**AAAGG**	**0.13**	**0.04**	**0.03**	**0.06**	**2**
STE12	Consensus	TGAAACA					
	Weeder	ATGAAACA	0.2	0.05	0.04	0.07	
	YMF	ACATGS	0.06	0.1	0.04	0.07	
	Our Method	TGAAAC	0.86	0.7	0.63	0.77	3
TBP	Consensus	TATAWAW					
	Weeder	CCGCTG	0	0	0	-0.02	
	YMF	CRCATR	0.01	0.02	0.01	0	
	Our Method	ATATAAA	0.43	0.89	0.41	0.62	13
UASPHR	Consensus	CTTCCT					
	Weeder	TGTCAGCG	0	0	0	-0.01	
	YMF	CCTCGTT	0.14	0.21	0.09	0.17	
	Our Method	CTTCCTC	0.71	0.86	0.64	0.78	9
Average	Weeder		0.16	0.09	0.05	0.09	
	YMF		0.48	0.51	0.37	0.47	
	Our Method		0.54	0.65	0.44	0.56	

5 Conclusion and Future Work

This paper presents an efficient algorithm to discover non-induced patterns from a large sequence data. It uses a generalized suffix tree to assist the identification of significant representative patterns and the removal of the induced patterns whose statistical significance is due to their strong subpatterns. By ensuring that each pattern discovered is non-induced, our method produces a more compact pattern set.

The results from TF binding sites experiment confirm the algorithm's ability to acquire a relatively small set of patterns that reveal interesting, unknown information inherent in the sequences. While the algorithm drastically reduces the number of

patterns, it is still able to retain patterns associated with conserved functional units in the promoter regions without relying on prior knowledge.

Our future work will advance in the following directions: (1) Extending our method to discover patterns with gaps; (2) Discovering distance patterns in protein sequences and relating the discovered patterns to three-dimensional conformation and low sequence similarity.

Acknowledgments. The work was supported by grants from NSERC.

References

1. Rigoutsos, I., Floratos, A.: Combinatorial pattern discovery in biological sequences: the TEIRESIAS algorithm. Bioinformatics 14(1), 55–67 (1998)
2. Parida, L., Rigoutsos, I., Floratos, A., Platt, D., Gao, Y.: Pattern Discovery on Character Sets and Real-valued Data: Linear Bound on Irredundant Motifs and an Efficient Polynomial Time Algorithm. In: Proceedings of the eleventh ACM-SIAM Symposium on Discrete Algorithms, pp. 297–308 (January 2000)
3. Sinha, S., Tompa, M.: Discovery of novel transcription factor binding sites by statistical overrepresentation. Nucleic Acids Research 30(24), 5549–5560 (2002)
4. Apostolico, A., Bock, M., Lonardi, S., Xu, X.: Efficient Detection of Unusual Words. Journal of Computational Biology 7(1/2), 71–94 (2000)
5. Eskin, E., Pevzner, P.: Finding composite regulatory patterns in DNA sequences. Bioinformatics 18(suppl. 1), S354–S363 (2002)
6. Marsan, L., Sagot, M.: Extracting structured motifs using a suffix tree - Algorithms and application to promoter consensus identification. Journal of Computational Biology 7(3/4), 345–362 (2000)
7. Blanchette, M., Sinha, S.: Separating real motifs from their artifacts. Bioinformatics 17(suppl. 1), S30–S38 (2001)
8. Sze, S., Lu, S., Chen, J.: Integrating Sample-Driven and Pattern-Driven Approaches in Motif Finding. In: Algorithms in Bioinformatics: 4th International Workshop, pp. 438–449 (2004)
9. Bailey, T.L., Elkan, C.: Unsupervised learning of multiple motifs in biopolymers using expectation maximization. Machine Learning 21, 51–80 (1995)
10. Pavesi, G., Zambelli, F., Pesole, G.: WeederH: an algorithm for finding conserved regulatory motifs and regions in homologous sequences. BMC Bioinformatics 8, 46 (2007)
11. Haberman, S.: The Analysis of Residuals in Cross-Classified Tables. Biometrics 29, 205–220 (1973)
12. Wong, A., Wang, Y.: High-Order Pattern Discovery from Discrete-Valued Data. IEEE Trans on Knowledge Systems 9(6), 877–893 (1997)
13. Gusfield, D.: Algorithms on Strings, Trees and Sequences: Computer Science and Computational Biology (1997)
14. SCPD database, `http://rulai.cshl.edu/SCPD/`
15. Tompa, M., Li, N., Bailey, T., Church, G., Moor, B., Eskin, E., Favorov, A., Frith, M., Fu, Y., Kent, W., Makeev, V., Mironov, A., Noble, W., Pavesi, G., Pe-sole, G., Regnier, M., Simonis, N., Sinha, S., Thijs, G., Helden, J., Vandenbogaert, M., Weng, Z., Workman, C., Ye, C., Zhu, Z.: Assessing Computational Tools for the Discovery of Transcription Factor Binding Sites. Nature Biotechnology 23(1), 137–144 (2005)
16. Wong, A., Li, G.: Simultaneous Pattern Clustering and Data Grouping. IEEE Trans. Knowledge and Data Engineering 20(7), 911–923 (2008)