JJEE
Volume 5, Number 2, 2019
Pages 77-90
Jordan Journal of Electrical Engineering
ISSN (Print): 2409-9600, ISSN (Online): 2409-9619

# Finding Regulatory Motifs of Genetic Networks Using Cut-Sort Algorithm

Ahmad M. Al-Omari[1a], Mohammed H. Tawalbeh[2b], Abedalmuhdi M. Almomany[3c]

[1]Department of Biomedical Systems and Informatics Engineering, Yarmouk University, Irbid, Jordan
[2]Information Technology & Communications Center, Jordan University of Science and Technology, Irbid, Jordan
[3]Department of Computer Engineering, Yarmouk University, Irbid, Jordan
[a]e-mail: aomari@yu.edu.jo
[b]e-mail: mt@just.edu.jo
[c]e-mail: emomani@yu.edu.jo

*Received: December 29, 2018* *Accepted: February 23, 2019*

***Abstract***— Understanding the targets of regulatory genes has become a challenging problem for bioinformaticians and biologists in systems biology. The main issue in solving this challenge consists in finding motifs that are finding short, recurring patterns in DNA or in amino-acid sequences that presumably have a regulatory function. A motif is considered a signature for a protein family binding to sequence motifs in the genome. The major challenge in finding motifs arises from the fact that most of the time the motifs are not well conserved. To discover such degenerate motifs, aligning multiple sequence motifs becomes a challenge. Usually, a motif discovery algorithm uses some prior information about the motifs to be discovered. In this paper, we present a novel algorithm for finding conserved sequence motifs in DNA without having a priori knowledge about the motifs. However, the algorithm can be used for motifs sequence both in DNA and in proteins. Our algorithm mainly depends on cutting sequences that have conserved motifs into equal fragments, sorting the fragments and then extending in both fragment directions. The algorithm runs in a very short time period. It takes 5.5 seconds for a real data sequence with length N = 28,000 nucleotides to find its identical, degenerate, long and short motifs; it can be easily parallelized by implementing it on General Purpose Graphical Processing Units. The algorithm guarantees to find any globally optimal solution within a short time even for sequences with very long motifs.

***Keywords***— Bioinformatics, Genetic network, GPGPU, Regulation motif, Systems biology.

## I. INTRODUCTION

Motif prediction represents one of the most active research areas in bioinformatics since the late 1980's [1]. Most DNA signals allow some minor variations in their sequence. The problem is central to biological systems because the identification of these motifs allows the construction of connections between genes and regulators through identification of DNA-protein interactions [2]-[4]. Thus, the signal will actually consist of several different possible words, often closely related. Such a collection of words is a repeated pattern which is called a motif that appears as a conserved sequence of nucleotides in a DNA or a conserved sequence of amino acids in a protein. This conserved pattern is usually taken to be a cis-regulatory sequence [3]. For example, finding motifs based on the prediction of a "conserved" sequence in the promoter regions of genes can help us discover genes that are possibly co-regulated by the same transcription factor(s) [3]. In general, these motifs are well defined, i.e., TATA box in the promotor area. Discovering gene regulation mechanisms helps us to understand the development, functioning and evolution of an organism. The importance of motifs has motivated many researchers to develop different tools and algorithms for finding them. Consensus [5] is a greedy multiple alignment that provided a strategy to assess the statistical significance of a given information content score based on large deviation statistic; MEME [6] uses statistical modeling techniques, such as expectation maximization, to automatically choose the best width, number of occurrences, and description for each motif from input of a

Corresponding author's e-mail: aomari@yu.edu.jo

group of DNA or protein sequences. The algorithm in [7], GibbsDNA [8] and AlignAce [9] uses Gibbs sampling where a set of DNA sequences is used to find conserved motifs inside them. This is done by choosing a sequence for sampling, choosing a random motif position for each unchosen sequence, counting residue occurrences for each position for all unchosen sequences, calculating weight for each possible motif position in the chosen sequence, randomly choosing a motif position for the chosen sequence using the weights. The whole process can be repeated using a new chosen sequence for sampling until the convergence to the conserved motifs. The program WINNOWER [10] finds cliques in graphs and uses a word-based approach combined with graph-theoretic methods for motif finding and it builds a graph with vertices corresponding to substrings from the sample sequences and edges corresponding to similar substrings. It was designed to improve the efficiency of existing motif finders using random projections of the input's substrings. SP-STAR [11] is a combinatorial algorithm for signal finding based on a new insight into the design of scoring functions which use the sum of pair scoring. MITRA (Mismatch Tree Algorithm) [12] uses mismatch trees to prune an exhaustive search space. It is based on the suffix tree and its variant.

Motif finding in prokaryotic genomes can be one of the following types: 1) supervised, where the features of a motif sequence are known, for instance, finding a consensus sequence; 2) unsupervised, where little is known about the motif sequence, for example, finding a conserved cis-regulatory region of coregulated genes; and 3) exploratory motif finding, where no prior information is available about the motif sequence [13]. Although many approaches used in previous research needed prior information about motifs [9], [13]-[17], this paper focuses on exploratory motif finding, which does not need any prior information about motifs. Usually some prior information about motifs is needed by a motif searching algorithm. This piece of information could consist of knowing a good alignment of DNA or protein sequences before starting the motif searching process (see [18]). Motifs have such properties as being a palindrome or tandem repeats, knowing something about the motif locations (e.g., conserved regulatory motifs upstream of coregulated genes), or knowing good estimates of algorithm input parameters (see [11]). For example, in [11] such an algorithm runs with a progressively increasing parameter "d" and a fixed parameter "l" until a signal graph becomes non-empty. Not having good initial estimates of "d" and "l" makes the construction of such a graph very difficult and the results poor.

Many methods and algorithms have been proposed by researchers to discover sequence motifs. The most important types of motif finding algorithms include phylogenetic footprinting, integrating algorithms, word-based algorithms, and probabilistic algorithms. Phylogenetic footprinting algorithms [19]-[24] build a global multiple alignments of the orthologous promoter sequences and identify a conserved region in the alignment. Integrating algorithms improves motif finding from genomic sequences by integrating DNA sequence data from coregulated genes and phylogenetic footprinting [16], [17], [25]-[30]. The procedure of these algorithms involves integrating motif overrepresentation and cross-species conservation. Word-based algorithms use optimized data structures of repeated motifs from a DNA sequence to construct suffix trees [31] and then discover the motif models .Probabilistic algorithms [32], [33] calculate the information content score of multiple sequences, count the number of possible alignments and find the p-value to assess the statistical significance of an alignment by determining the expected frequency of an information content score.

Our motivation for this research is to propose an algorithm to increase the speed and the accuracy of finding motifs, which are the main drawbacks of motif finding algorithms [9], [13]-[17]. These issues are critical when we need to identify thousands of motifs in a genome

[34]. Also, this paper proposes a possible parallelization strategy of the algorithm using General Purpose Graphical Processing Units (GPGPUs).

In previous research, we discovered the mechanism of gene regulation for a complex and large genetic network using ensemble methods [35]-[38]. The key idea was identifying an ensemble of models consistent with the data available to predict system behavior over time instead of a model consistent with the data that is based on numerical and parallelization strategies on the GPGPU [39], [40]. The GPGPU is used to solve for 2418 systems of ODEs that have the same mathematical form and different data. However, the research here focuses on gene regulation from the motif perspective, i.e., the DNA binding site of the regulator. The real bioinformatics challenge here is that if several promoters have degenerate motifs as demonstrated in Fig. 1, then different motif sequences might be bound to the same transcription factor to turn a regulation process on or off. Thus, the challenge is that we do not search for an identical sequence of nucleotides in a DNA or identical sequence of amino acids in a protein; rather the search is for conserved sequences that are not identical, have slight changes in their building blocks and are bound to the same transcription factors [2]. This is not simply done by lining up sequences and drawing boxes, although that is how the first motif, TATA box in human, was found. Most of these motifs require some sort of algorithm to find them.

Fig. 1 represents six DNA sequences with different length and degenerate motifs in boxes marked for a particular regulator located in different position of the sequence. The problem is how to find those motifs computationally without any prior knowledge about them, and within a reasonable computational time.

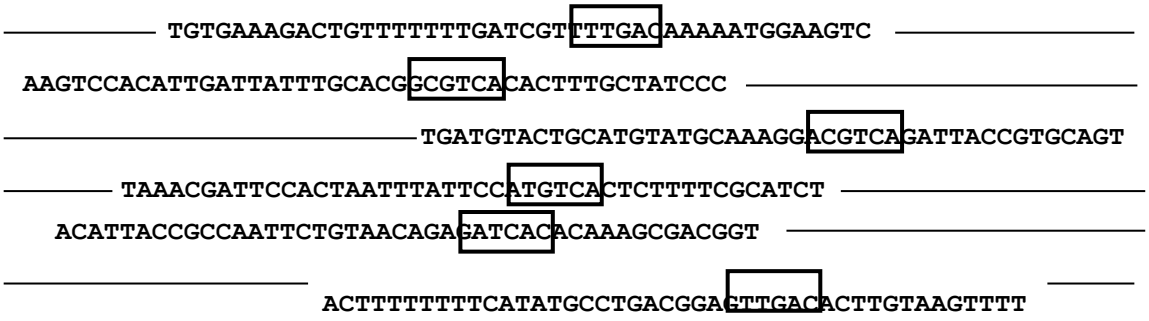


Fig. 1. Six DNA sequences with different length and with degenerate motifs

## II. METHODS

In essence, to locate a motif like the one in Fig. 1, we propose the following approach. If we were to know the length of the "conserved" sequence to be K (K is usually between 5 and 30), then we could go through all possible combinations of K-mers, one K-mer from each sequence, and calculate the information content (IC) of the entire combination. We call a combination a "conserved" site or motif if the IC is high. However, the problem is that there are too many combinations to consider; and sequences and motifs are not aligned as shown in Fig. 2. For example, if we have 20 sequences each having 100 nucleotides, then for K=16 we have $(100\text{-}K+1)^{20} = 85^{20}$ combinations. If a computer makes $10^6$ comparisons per second then we need approximately $1.25X10^{25}$ years of simulation! In the method section, we present a smart solution to find these conserved motifs accurately and quickly.

Given a sample of sequences with unknown pattern (motif) that appears at different unknown positions in each sequence, can the unknown pattern be found? The following Cut-Sort algorithm finds the motif in each sequence accurately and in a very short amount of time. It is

called CUT-SORT algorithm because the core of the algorithm is to CUT sequences into fragments of equal length and then to SORT them. Fig. 3 demonstrates the Pseudocode of the following algorithm and Fig. 4 depicts the algorithm using a flowchart. The whole program code using C++ computer language and the real genes data [41], which were taken from real data of 50 disordered patient gene samples, are available for free on the following link https://sourceforge.net/projects/cut-sort-code-data/files/

The Cut-Sort Algorithm: Given "N" DNA sequences, each sequence "S" with different lengths, finds a motif of size "k" that is conserved among all N sequences. The user enters to the program some sequences with any length; and the program will output the conserved motifs with size k.

1- Divide each sequence "S" into fragments with length L. For example, L could be chosen to be 5<L<30.
2- Save the fragment position that shows where its location is in the original sequence "S".
3- Sort those fragments alphabetically and group them. Now each group has the same fragments with no nucleotide differences. However, those fragments have different sequence locations and positions.
4- Each fragment within a group is now a nucleus for a motif to be found in different sequences "S".
5- Return each nucleus to its original sequence "S" and start extending each nucleus of length "L" from the same group in both directions by adding one nucleotide at a time to the motif nucleus. Compare all added nucleotides in the sequences of the same group after each extension as presented in Fig. 2 below.
6- Stop when (a) there is any nucleotide mismatch if the goal is looking for exact motifs, (b) when the number of nucleotide differences exceeds some threshold T1 in a degenerate motif, (c) when a certain threshold of a motif length T2 is reached.

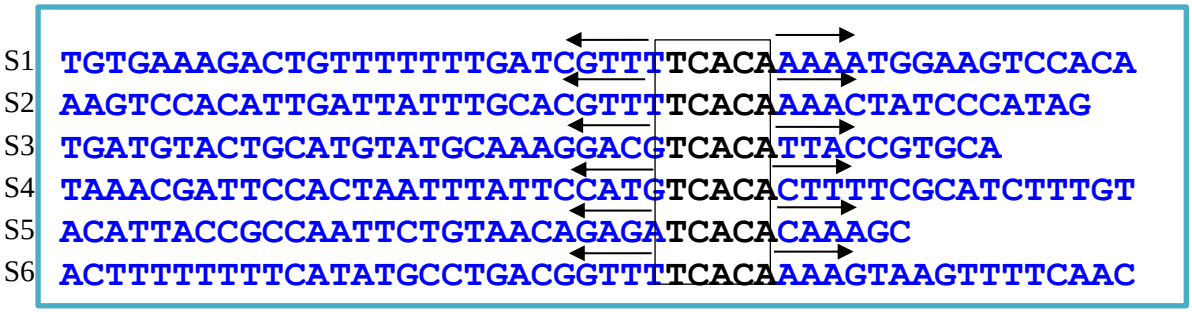


Fig. 2. A group of six nuclei with length equals five nucleotides for each within six DNA sequences

Fig. 2 shows six DNA sequences S={S1,S2,….S6} and N=6, with nucleus length L=5 (TCACA) in one group. The nucleotides are in the black box. The algorithm will extend the arrows for all nucleus one nucleotide at a time in both directions in each single group to find whole motifs. For example, you will find the exact motif GTTT**TCACA**AAA in S1, S2, and S6 when the nucleus TCACA is extended in both directions.

***INPUT:*** *Set of DNA or Protein sequences each one with an unknown pattern at an unknown position* $S\{S_1,S_2,S_3,\ldots,S_N\}$,
***DIVIDE*** $S_i$ *into fragments F with length L and save F positions;* $1 \leq i \leq N$,
***SORT*** *F alphabetically; all similar F are grouped; call each group G,*

***DO***
***FOR EACH*** *F in G extend in both directions, one nucleotide at a time*

***COMPARE*** *extended nucleotides in a group*
***IF*** *Exactly matched*
***THEN*** *extend motifs*
***ELSE IF*** *# nucleotide differences < threshold T1*
***THEN*** *extend motifs*
***ELSE*** *Exit*

***While*** *(Motif length <threshold T2)*

***OUTPUT:*** *Set of discovered conserved regions or motifs*

Fig. 3. Pseudocode of the Cut-Sort algorithm

The algorithm finds all sequence motifs in a given set of DNA or protein sequences without having a priori knowledge about them. It can find identical, degenerate, long and short motifs. Fig. 4 depicts the algorithm flowchart.

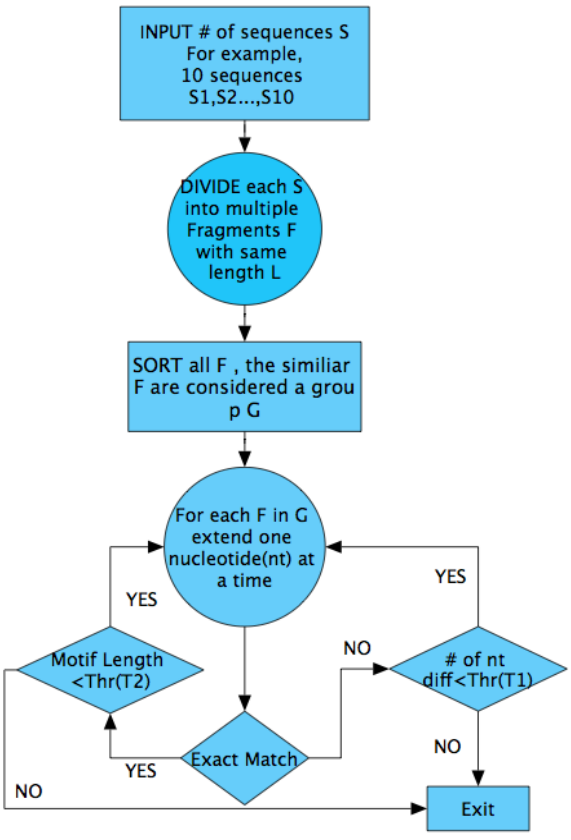


Fig. 4. Cut-Sort algorithm flowchart

The algorithm will keep extending nuclei of the same group, one nucleotide at a time in both directions until the number of nucleotides for the nuclei in the group becomes greater than threshold T1 or the length of the found motifs in the group becomes greater than threshold T2.

## III. RESULTS AND DISCUSSIONS

Predicting motifs is crucial for understanding biological processes and systems biology in particular [3]. It can help to identify proteins, DNA, or RNAs that have a specific property of interest (for example, being phosphorylated by a particular kinase or promoter that has a particular property that is likely to be regulated by a particular transcription factor). It also helps to infer which factors regulate which genes. Model gene expression predicts how the occupancy of that transcription factor would change gene expression [36], [38]. The developed algorithm found short, long, and degenerate motifs from different sequences. The algorithm was able to find very short motifs in different DNA samples and different sample positions as shown in Table 1. It at the same time found very long motifs as it is shown in Table 2. Moreover, Table 3 shows that the algorithm can find not only the identical motifs but also the degenerate motifs with nucleotide mutations.

Table 1 represents short motifs with length 8 nucleotides found in the data. The algorithm finds short motifs in different samples and positions. For example, it found 5 short motifs in 5 different DNA samples (S17, S2, S20, S32, and S46) out of 50 samples. Each motif in those samples found in different positions, (i.e. first motif found in sample#17 in nucleotides position 200, and so on). The minimum and maximum lengths of the motif depend on predetermined thresholds (T2) as presented in the algorithm above.

TABLE 1
SHORT MOTIFS FOUND IN DATA WITH LENGTH EIGHT NUCLEOTIDES

| Short Motifs | Sample # | Start Position in the Sample |
|---|---|---|
| GAGGGCAA | 17 | 200 |
| GAGGGCAA | 2 | 744 |
| GAGGGCAA | 20 | 224 |
| GAGGGCAA | 32 | 216 |
| GAGGGCAA | 46 | 216 |

Table 2 demonstrates long motifs found in the data with length 100 nucleotides. The algorithm finds long motifs in different samples and positions. For example, it found 6 long motifs in 6 different DNA samples (S12, S37, S39, S45, S47, and S49) out of 50 samples used. The minimum and maximum lengths of the motif depend on predetermined thresholds (T2) as shown in the algorithm above.

Table 3 shows degenerate motifs, where motif sequences have slight changes in their building blocks and are still bound to the same transcription factors. The algorithm finds degenerate motifs in different samples and positions. For example, it found 7 degenerate motifs in 7 different DNA samples (S13, S29, S29, S38, S44, S46 and S5) out of 50 samples used. Each motif in those samples is found in different positions, (i.e. the first motif found in sample#13 in nucleotides position 124, and so on). The nucleotide differences can be less or more based on a predetermined threshold (T1) as shown in the algorithm above.

TABLE 2
LONG MOTIFS FOUND IN THE DATA WITH LENGTH 100 NUCLEOTIDES

| Long Motifs | Sample # |
|---|---|
| CCGTGGCACTGGACAACAGTGTGTACCTGTGGAGTGCAAGCTCTGGTGACATCC TGCAGCTTTTGCAAATGGAGCAGCCTGGGGAATATATATCCTCTGT | 12 |
| CCGTGGCACTGGACAACAGTGTGTACCTGTGGAGTGCAAGCTCTGGTGACATCC TGCAGCTTTTGCAAATGGAGCAGCCTGGGGAATATATATCCTCTGT | 37 |
| CCGTGGCACTGGACAACAGTGTGTACCTGTGGAGTGCAAGCTCTGGTGACATCC TGCAGCTTTTGCAAATGGAGCAGCCTGGGGAATATATATCCTCTGT | 39 |
| CCGTGGCACTGGACAACAGTGTGTACCTGTGGAGTGCAAGCTCTGGTGACATCC TGCAGCTTTTGCAAATGGAGCAGCCTGGGGAATATATATCCTCTGT | 45 |
| CCGTGGCACTGGACAACAGTGTGTACCTGTGGAGTGCAAGCTCTGGTGACATCC TGCAGCTTTTGCAAATGGAGCAGCCTGGGGAATATATATCCTCTGT | 47 |
| CCGTGGCACTGGACAACAGTGTGTACCTGTGGAGTGCAAGCTCTGGTGACATCC TGCAGCTTTTGCAAATGGAGCAGCCTGGGGAATATATATCCTCTGT | 49 |

TABLE 3
DEGENERATE MOTIFS FOUND IN THE DATA

| Degenerate Motifs | Sample # | Start Position in the Sample |
|---|---|---|
| CACTCCTGGGATCCCCCCCCCCTTTTTTAAAAG | 13 | 124 |
| CCGGTGTTTTCTCCCCCCCCCCCCTGGGGTTGC | 29 | 252 |
| ATCTGATATTTACCCCCCCCCCCGTGACGTCAC | 29 | 468 |
| GCCCCCCCCCGCCCCCCCCCGAAGACCCGCCTT | 38 | 316 |
| CCCCCCCCCTCCCCCCCCCCAAAAAACGCCTTT | 44 | 308 |
| GTTGCTCCCTATCCCCCCCCTCAGGCAGGAGAA | 46 | 404 |
| TCCACCCCAAAGCCCCCCCCCGCTCCCCCCCGA | 5 | 292 |

With the advent of high-throughput sequencing technology to identify DNA-protein interactions, finding motifs becomes a real challenge because of the thousands of putative binding sites generated [42]. In order to understand some aspects of regulation of gene expression, many computer scientists and biologists invented and developed diverse approaches to find motifs quickly and accurately. The developed algorithms involve combinatorial enumeration, probabilistic modeling, mathematical programming, neural networks and genetic algorithms. Evaluating the performance of all of these approaches has still been a difficult task because there is no obvious understanding of the regulatory mechanisms. Thus, there is no standard to measure the correctness of those tools. However, we have tested the Cut-Sort algorithm on real data for 50 patient samples [41] against the exhaustive search algorithm, which is the most accurate approach. Core i7 CPU@2.4GHz and 12GB RAM is used to show the accuracy and the speed of the Cut-Sort algorithm proposed here. The algorithm has found all motifs in different samples. Fig. 5 shows that the Cut-Sort algorithm needs much less computational time than the exhaustive search when the number of nucleotides is changed. For example, the Cut-Sort algorithm needs 0.945 Sec to find all motifs for the total number of nucleotides equal 9000 while the exhaustive search needs about 1141.03 to find all motifs for the same number of nucleotides. Fig. 6 illustrates the relationship between the nucleus length and time. It is obvious that increasing the nucleus length will decrease the required computational time for the same set of DNA sequences. However, this will generally increase the specificity of the found motifs and decrease their numbers as depicted in Fig. 7.

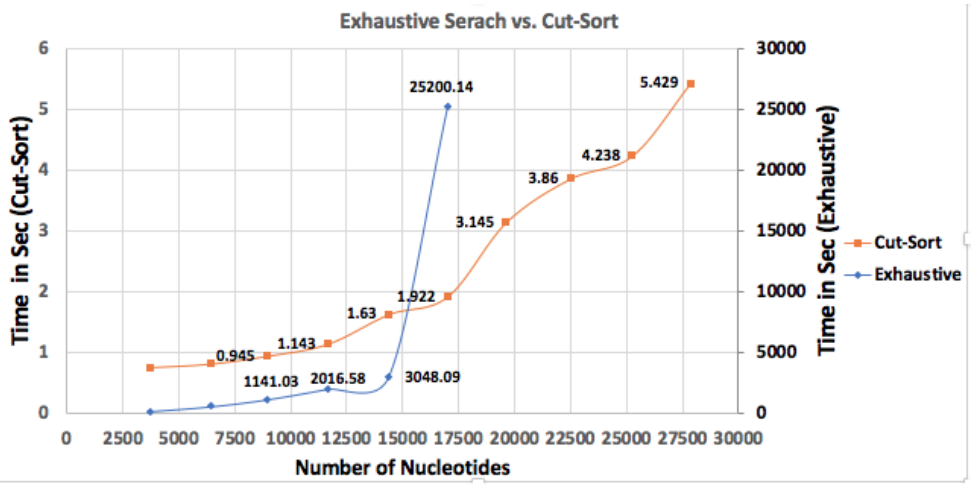


Fig. 5. Relationship between the number of nucleotides and the needed computational time for both the exhaustive search and Cut-Sort algorithm

Fig. 5 demonstrates the relationship between the number of nucleotides and the needed computational time for both the exhaustive search and Cut-Sort algorithm. The time in seconds needed by the Cut-Sort algorithm is substantially less than that in the Exhaustive search algorithm, which finds motifs in different number of nucleotide sequences. The Cut-Sort algorithm found the same set of motifs in 1.922 seconds instead of more than seven hours in the exhaustive search. The numbers that appear on the figures are the time in seconds.

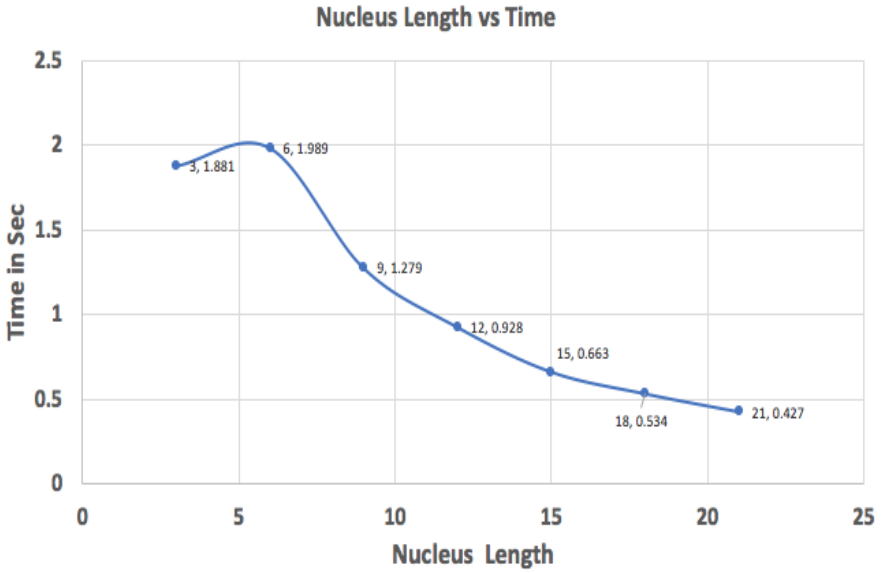


Fig. 6. Relationship between the nucleus length and time

Fig. 6 shows the relationship between the nucleus length and time. The time in seconds needed by the Cut-Sort algorithm decreases as the nucleus length is increased. A longer nucleus length leads to a faster performance by the Cut-Sort algorithm. When the nucleus length was three nucleotides then the algorithm needed 1.1881 Secs to find all motifs corresponding to this nucleus length. When the length of the nucleus was 21 nucleotides, the algorithm needed 0.427 Secs to find all motifs. However, the motifs' sequences and their count are changed when you change the nucleus length as shown in Fig. 7 below. The pair of numbers shown on the figure corresponds to nucleus length and time in seconds.

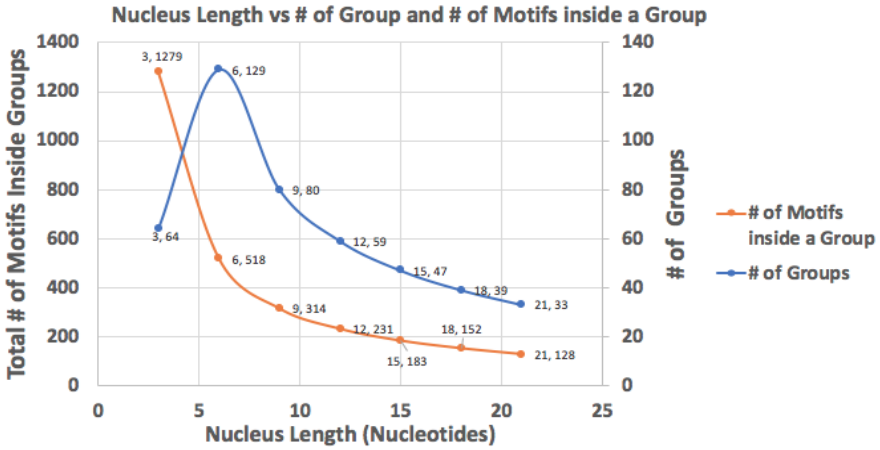


Fig. 7. Nucleus length versus # of group and # of motifs inside a group

The number of candidate motifs inside a group decreases monotonically with Nucleus Length (left vertical axis); and the number of candidate groups ultimately decreases (but non-monotonically) (right vertical axis). The number of motifs inside groups is identified before in Fig. 2. Each group is defined uniquely by its nucleus; and there can be multiple nuclei initially in searching for a motif. The left vertical axis is the number of motifs inside a group. The right vertical axis is the number of groups defined by their nucleus. The numbers that appear on red line correspond to nucleus length and total # of motifs inside all groups, while those appearing on the blue line correspond to nucleus length and # of groups.

In Fig. 7, as the length of the fragment (nucleus) is increased, then the number of candidate groups found for a given set of genes tends to decrease, where each group has the same set of nucleus motifs, and at the same time the number of nucleus motifs inside the group decreases as well. However, as the length of the nucleus increases, the space of the found motifs will decrease so that some short motifs may disappear from the search space. This depends on the required minimum and maximum motif lengths, which are considered user predefined values. Many motif finders require these values; for example, MEME [6] needs to predefine the minimum and the maximum motif length before starting the search process. The irregular increasing in the number of groups when the nucleus length was 6 happened because, with this length, the program found many groups where each group has similar nuclei; this unpredictable behavior depends on the used sequences but the number of groups should generally decrease when the nucleus length increases.

The Cut-Sort algorithm was able to find global optimal solutions. This was one of the drawbacks that appeared in probabilistic algorithms where the competing algorithm may converge to a local optimal solution because the competitor uses some form of local search [9], [16], [17], [33], [43], [44]. On the other hand, Cut-Sort algorithm improved the speed even for very long motifs. The speed problem is clearly the drawback of some word-based approaches that rely on exhaustive enumeration [14], [15], [45]. For example, WINNOWER, which is a word-based approach, requires significant computational resources (time and memory) and becomes very slow when finding subtle motifs in very large samples [11]. In contrast, Cut-Sort algorithm found subtle motifs within a very short time as demonstrated in the result section. As for the probabilistic or machine learning implementations of local search strategies, the shortcoming of these algorithms is that, for subtle motifs, they often converge to local optima that represent random patterns rather than a motif [11]. Li, Ma, & Wang proposed a different combinatorial paradigm with a proven performance that avoids local optima. However, the running time of the algorithm is huge for real scenarios [46]. Of course,

it is obvious that there is a trade-off between avoiding local optima and speed for motif finding algorithms. In comparison, Cut-Sort algorithm found a global optimal solution within a very short time, as illustrated in the result section.

## IV. FUTURE WORK

A potential parallelization scheme for the algorithm: Unlike the other sequential algorithms [25], [47]-[51], which are inherently sequential for solving the motif identification problem, this algorithm can be parallelized on a GPGPU. In this algorithm, detecting motifs in given sequences fits the SIMD (single instruction, multiple data) and warp-level parallelism concepts (where warp size for current NVIDIA GPUs is 32 threads). A common parallelization strategy in this category [40], [52] is (a) to increase the number of warps by increasing the number of thread blocks per streaming multiprocessors on the GPGPU and (b) to decrease the number of threads in a thread block. Side by side, this optimization strategy provides more independent warps from other thread blocks when one warp is stalled. The potential parallel algorithm proposes to assign a thread for each nucleus sequence in a group; and then every thread will be tasked to look for the full motif by extending one nucleotide at a time in both directions of the nucleus. By harnessing the power of GPGPU (NVIDIA Tesla K40), 2880 cores can be utilized in parallel to work on 2880 nuclei at once rather than working on them sequentially. All of the cores will execute the same code but with different data, SIMD principle. To maximize the usage of the GPGPU resources, we can use a warp per block concept; and for a large-scale problem we can extend the same algorithm by using multi-GPGPUs and assigning different groups to each GPGPU.

## V. CONCLUSIONS

A conserved sequence motif is a repeated pattern in DNA or proteins. It is indicative of a biological function such as DNA-binding by a protein. Finding sequence motifs represents one of the most fundamental problems in elucidation of complex biological systems [53]. One of the most important applications is finding genes that are transcriptionally (co-) regulated by the same transcription factor [3]. We developed a novel algorithm that finds degenerate motifs without any prior information about them. The proposed algorithm guarantees to find any globally optimal solutions within a short time even for very long motifs (0.758 sec). Thus, we improved the drawbacks of probabilistic algorithms where the competing algorithm may converge to a locally optimal solution because the competitor uses some form of local search [9], [16], [17], [33], [43], [44]. The Cut-Sort algorithm improved the speed which is the drawback of some word-based approaches that rely on exhaustive enumeration [14], [15], [45]. In the future work, we need to use the proposed parallel algorithm on the GPGPU in order to extract conserved regions within a very short time for large scale genomic problems involved in the reconstruction of genome scale networks using DNA-protein binding sites [36]. The algorithm will be used for characterizing binding sites in particular family of proteins taken from a database [54]. Both problems require scalable solutions as illustrated here.

## ACKNOWLEDGMENTS

This work was supported by the Graduate Scientific Research School at Yarmouk University under Grant number: 35/2017. We would like to thank Prof. Jonathan Arnold from the

University of Georgia-USA, for his invaluable comments that greatly improved the manuscript.

## REFERENCES

[1] S. Aluru, *Handbook of Computational Molecular Biology*, Chapman and Hall/CRC, 2005.

[2] S. Maerkl, and S. Quake, "A systems approach to measuring the binding energy landscapes of transcription factors," *Science*, vol. 315, no. 5809, pp. 233-237, 2007.

[3] C. Harbison, D. Gordon, T. Lee, N. Rinaldi, K. Macisaac, T. Danford, N. Hannett, J. Tagne, D. Reynolds, and J. Yoo, "Transcriptional regulatory code of a eukaryotic genome," *Nature*, vol. 431, no. 7004, pp. 99, 2004.

[4] B. Ren, F. Robert, J. Wyrick, O. Aparicio, E. Jennings, I. Simon, J. Zeitlinger, J. Schreiber, N. Hannett, and E. Kanin, "Genome-wide location and function of DNA binding proteins," *Science*, vol. 290, no. 5500, pp. 2306-2309, 2000.

[5] M. Jackson, T. Nilsson, and P. Peterson, "Identification of a consensus motif for retention of transmembrane proteins in the endoplasmic reticulum," *EMBO Journal*, vol. 9, no. 10, pp. 3153-3162, 1990.

[6] T. Bailey, M. Boden, F. Buske, M. Frith, C. Grant, L. Clementi, J. Ren, W. Li, and W. Noble, "MEME SUITE: tools for motif discovery and searching," *Nucleic Acids Research*, vol. 37, no. suppl_2, pp. W202-W208, 2009.

[7] T. Bailey, N. Williams, C. Misleh, and W. Li, "MEME: discovering and analyzing DNA and protein sequence motifs," *Nucleic Acids Research*, vol. 34, no. suppl_2, pp. W369-W373, 2006.

[8] S. Sze, M. Gelfand, and P. Pevzner, "Finding weak motifs in DNA sequences," *Biocomputing*, pp. 235-246, 2001.

[9] X. Chen, L. Guo, Z. Fan, and T. Jiang, "W-AlignACE: an improved Gibbs sampling algorithm based on more accurate position weight matrices learned from sequence and gene expression/ChIP-chip data," *Bioinformatics*, vol. 24, no. 9, pp. 1121-1128, 2008.

[10] J. Buhler and M. Tompa, "Finding motifs using random projections," *Computational Biology*, vol. 9, no. 2, pp. 225-242, 2002.

[11] P. Pevzner and S. Sze, "Combinatorial Approaches to Finding Subtle Signals in DNA Sequences," *Proceedings of the Eighth International Conference on Intelligent Systems for Molecular Biology*, pp. 269-278, 2000.

[12] E. Eskin and P. Pevzner, "Finding composite regulatory patterns in DNA sequences," *Bioinformatics*, vol. 18, no. suppl_1, pp. S354-S363, 2002.

[13] J. Mrázek, "Finding sequence motifs in prokaryotic genomes-a brief practical guide for a microbiologist," *Briefings in Bioinformatics*, vol. 10, no. 5, pp. 525-536, 2009.

[14] M. Tompa, "An exact method for finding short motifs in sequences, with application to the ribosome binding site problem," *Proceedings of the 8th International Conference on Intelligent Systems for Molecular Biology*, pp. 262–271, 1999.

[15] J. Helden, B. André, and J. Collado-Vides, "Extracting regulatory sites from the upstream region of yeast genes by computational analysis of oligonucleotide frequencies1," *Molecular Biology*, vol. 281, no. 5, pp. 827-842, 1998.

[16] R. Siddharthan, E. Siggia, and E. Van Nimwegen, "PhyloGibbs: a Gibbs sampling motif finder that incorporates phylogeny," *PLoS Computational Biology*, vol. 1, no. 7, pp. e67, 2005.

[17] A. Moses, D. Chiang, and M. Eisen, "Phylogenetic motif detection by expectation-maximization on evolutionary mixtures," *Biocomputing*, pp. 324-335, 2003.

[18] M. Larkin, G. Blackshields, N. Brown, R. Chenna, P. McGettigan, H. McWilliam, F. Valentin, I. Wallace, A. Wilm, and R. Lopez, "Clustal W and Clustal X version 2.0," *Bioinformatics*, vol. 23, no. 21, pp. 2947-2948, 2007.

[19] C. Carmack, L. McCue, L. Newberg, and C. Lawrence, "PhyloScan: identification of transcription factor binding sites using cross-species evidence," *Algorithms for Molecular Biology*, vol. 2, no. 1, pp. 1, 2007.

[20] T. Wang, and G. Stormo, "Identifying the conserved network of cis-regulatory sites of a eukaryotic genome," *Proceedings of the National Academy of Sciences*, vol. 102, no. 48, pp. 17400-17405, 2005.

[21] P. Cliften, P. Sudarsanam, A. Desikan, L. Fulton, B. Fulton, J. Majors, R. Waterston, B. Cohen, and M. Johnston, "Finding functional features in Saccharomyces genomes by phylogenetic footprinting," *Science*, vol. 301, no. 5629, pp. 71-76, 2003.

[22] E. Berezikov, V. Guryev, R. Plasterk, and E. Cuppen, "CONREAL: conserved regulatory elements anchored alignment algorithm for identification of transcription factor binding sites by phylogenetic footprinting," *Genome Research*, vol. 14, no. 1, pp. 170-178, 2004.

[23] M. Blanchette and M. Tompa, "Discovery of regulatory elements by a computational method for phylogenetic footprinting," *Genome Research*, vol. 12, no. 5, pp. 739-748, 2002.

[24] F. Cliften, L. Hillier, L. Fulton, T. Graves, T. Miner, W. Gish, R. Waterston, and M. Johnston, "Surveying Saccharomyces genomes to identify functional elements by comparative DNA sequence analysis," *Genome Research*, vol. 11, no. 7, pp. 1175-1186, 2001.

[25] S. Sinha, M. Blanchette, and M. Tompa, "PhyME: a probabilistic algorithm for finding motifs in sets of orthologous sequences," *BMC Bioinformatics*, vol. 5, no. 1, pp. 170, 2004.

[26] T. Wang, and G. Stormo, "Combining phylogenetic data with co-regulated genes to identify regulatory motifs," *Bioinformatics*, vol. 19, no. 18, pp. 2369-2380, 2003.

[27] A. Prakash, M. Blanchette, S. Sinha, and M. Tompa, "Motif discovery in heterogeneous sequence data," *Biocomputing*, pp. 348-359, 2003.

[28] M. Kellis, N. Patterson, M. Endrizzi, B. Birren, and E. Lander, "Sequencing and comparison of yeast species to identify genes and regulatory elements," *Nature*, vol. 423, no. 6937, pp. 241, 2003.

[29] A. McGuire, J. Hughes, and G. Church, "Conservation of DNA regulatory motifs and discovery of new motifs in microbial genomes," *Genome Research*, vol. 10, no. 6, pp. 744-757, 2000.

[30] M. Gelfand, E. Koonin, and A. Mironov, "Prediction of transcription regulatory sites in Archaea by a comparative genomic approach," *Nucleic Acids Research*, vol. 28, no. 3, pp. 695-705, 2000.

[31] M. Sagot, "Spelling approximate repeated or common motifs using a suffix tree," *Proceedings of in Latin American Symposium on Theoretical Informatics*, pp. 374-390, 1998.

[32] T. Down and T. Hubbard, "NestedMICA: sensitive inference of over-represented motifs in nucleic acid sequence," *Nucleic acids Research*, vol. 33, no. 5, pp. 1445-1453, 2005.

[33] G. Hertz and G. Stormo, "Identifying DNA and protein patterns with statistically significant alignments of multiple sequences," *Bioinformatics*, vol. 15, no. 7, pp. 563-577, 1999.

[34] D. Johnson, A. Mortazavi, R. Myers, and B. Wold, "Genome-wide mapping of in vivo protein-DNA interactions," *Science*, vol. 316, no. 5830, pp. 1497-1502, 2007.

[35] C. Caranica, A. Al-Omari, Z. Deng, J. Griffith, R. Nilsen, L. Mao, J. Arnold, and H. Schüttler, "Ensemble methods for stochastic networks with special reference to the biological clock of Neurospora crassa," *PloS One*, vol. 13, no. 5, pp. e0196435, 2018.

[36] A. Al-Omari, J. Griffith, C. Caranica, T. Taha, H. Schuttler, and J. Arnold, "Discovering regulators in post transcriptional control of the biological clock of Neurospora crassa using variable topology ensemble methods on GPUs," *IEEE Access*, vol. 6, pp. 54582 – 54594, 2018.

[37] Z. Deng, S. Arsenault, C. Caranica, J. Griffith, T. Zhu, A. Al-Omari, H. Schüttler, J. Arnold, and L. Mao, "Synchronizing stochastic circadian oscillators in single cells of Neurospora crassa," *Scientific Reports*, vol. 6, pp. 35828, 2016.

[38] A. Al-Omari, J. Griffith, M. Judge, T. Taha, J. Arnold, and H. Schüttler, "Discovering regulatory network topologies using ensemble methods on GPGPUs with special reference to the biological clock of neurospora crassa," *IEEE Access*, vol. 3, pp. 27-42, 2015.

[39] A. Al-Omari, H. Schuttler, J. Arnold, and T. Taha, "Solving nonlinear systems of first order ordinary differential equations using a galerkin finite element method," *IEEE Access*, vol. 1, pp. 408-417, 2013.

[40] A. Al-Omari, J. Arnold, T. Taha, and H. Schuttler, "Solving large nonlinear systems of first-order ordinary differential equations with hierarchical structure using multi-GPGPUs and an adaptive Runge Kutta ODE solver," *IEEE Access*, vol. 1, pp. 770-777, 2013.

[41] O. Batiha, "Unpublished patient data," Department of Biotechnology and Genetic Eng. Jordan University of Science & Technology, 2018.

[42] Y. Shen, F. Yue, D. McCleary, Z. Ye, L. Edsall, S. Kuan, U. Wagner, J. Dixon, L. Lee, and V. Lobanenkov, "A map of the cis-regulatory sequences in the mouse genome," *Nature*, vol. 488, no. 7409, pp. 116, 2012.

[43] P. Bucher, "Weight matrix descriptions of four eukaryotic RNA polymerase II promoter elements derived from 502 unrelated promoter sequences," *Molecular Biology*, vol. 212, no. 4, pp. 563-578, 1990.

[44] G. Hertz, G. Hartzell III, and G. Stormo, "Identification of consensus patterns in unaligned DNA sequences known to be functionally related," *Bioinformatics*, vol. 6, no. 2, pp. 81-92, 1990.

[45] J. Helden, A. Rios, and J. Collado-Vides, "Discovering regulatory elements in non-coding sequences by analysis of spaced dyads," *Nucleic Acids Research*, vol. 28, no. 8, pp. 1808-1818, 2000.

[46] M. Li, B. Ma, and L. Wang, "Finding similar regions in many sequences," *Computer and System Sciences*, vol. 65, no. 1, pp. 73-96, 2002.

[47] F. Liu, J. Tsai, R. Chen, S. Chen, and S. Shih, "FMGA: finding motifs by genetic algorithm," *Proceedings of the Fourth IEEE Symposium on Bioinformatics and Bioengineering*, pp. 459-466, 2004.

[48] H. Dinh, S. Rajasekaran, and V. Kundeti, "PMS5: an efficient exact algorithm for the (ℓ, d)-motif finding problem," *BMC Bioinformatics*, vol. 12, no. 1, pp. 410, 2011.

[49] U. Keich and P. Pevzner, "Finding motifs in the twilight zone," *Proceedings of the Sixth Annual International Conference on Computational Biology*, pp. 195-204, 2002.

[50] Z. Kashani, H. Ahrabian, E. Elahi, A. Nowzari-Dalini, E. Ansari, S. Asadi, S. Mohammadi, F. Schreiber, and A. Masoudi-Nejad, "Kavosh: a new algorithm for finding network motifs," *BMC Bioinformatics*, vol. 10, no. 1, pp. 318, 2009.

[51] J. Lonardi, and P. Patel, "Finding motifs in time series," *Proceedings of the 2nd Workshop on Temporal Data Mining*, pp. 53-68, 2002.

[52] S. Ryoo, *Program Optimization Strategies for Data-Parallel Many-Core Processors*, University of Illinois at Urbana-Champaign, 2008.

[53] E. Consortium, "Identification and analysis of functional elements in 1% of the human genome by the ENCODE pilot project," *Nature*, vol. 447, no. 7146, pp. 799, 2007.

[54] X. Ma, J. Guo, K. Xiao, and X. Sun, "PRBP: prediction of RNA-binding proteins using a random forest algorithm combined with an RNA-binding residue predictor," *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, vol. 12, no. 6, pp. 1385-1393, 2015.