Contents lists available at Science-Gate

International Journal of Advanced and Applied Sciences

Journal homepage: http://www.science-gate.com/IJAAS.html

# Identification of DNA motif using particle swarm optimization technique

Ahmed Y. Khedr [1, 2, *]

[1]College of Computer Science and Engineering, Hail University, Hail, Saudi Arabia
[2]College of Engineering, Al-Azhar University, Cairo, Egypt

ARTICLE INFO

*Article history:*
Received 20 March 2017
Received in revised form
11 May 2017
Accepted 16 May 2017

*Keywords:*
Soft computing
Swarm
Motif
DNA
Optimization

ABSTRACT

The process of discovering short recurring patterns in DNA called DNA motif. DNA motif is an important part to study the biological cell functions. The main challenging of DNA motif is the running time to identify the motif where it increases with the length of motif and the number of mutations. Particle swarm optimization (PSO) is one of the efficient techniques to find an approximate solution using global optimization technique. We propose a PSO algorithm to find DNA motif. The experimental study on artificial data shows that the running time of the proposed algorithm is faster than recent proposed algorithms. The proposed algorithm is also compared to voting and hybrid algorithms for performance measure. In addition, the accuracy of the proposed algorithm is 90%. Finally, we apply the proposed algorithm on real data includes PDR3, GAL4, MATalpha2, and MCB.



## 1. Introduction

Given $t$ DNA sequences, $S = \{S_1, S_2, S_3, \ldots, S_t\}$ and two integers $l$ and $d$, where $l$ represents the length of the motif and $d$ represents the maximum number of mutations in the motif. Each sequence consists of $n$ nucleotides, $\sum = \{A, T, C, G\}$, long. The problem is to identify all strings $M$ of length $l$ such that $M$ occurs in each of the $t$ sequences with at most $d$ mismatches (Davila et al., 2007).

From the definition, we need to identify a string $M$ that satisfies the following constraint: there exists a string $M_i$ of length $l$ in sequence $i$, $1 \le i \le t$, such that the number of mismatches between $M$ and $M_i$ is less than or equal to $d$. The number of mismatches between two strings of equal length is known as the Hamming distance between them. In this case, the string $M$ is called a motif. Two properties of DNA motif; the first one is the length of motif where it is short, while the second is a recurring pattern. This pattern is used to identify regulatory elements, especially the binding sites in DNA for transcription factors. Different researches in DNA motif can be classified into three approaches based on the type of DNA sequence information. The first approach uses promoter sequences from coregulated genes from a single genome. The second one uses orthologous promoter sequences of a single gene from multiple species. The last approach uses promoter sequences of coregulated genes as well as multiple species.

In our study, we focused on the first approach. All the literature in this approach categorized motif finding algorithms into two groups. The first group contains all methods that find an exact solution. The methods in this group are based on different strategies such as brute-force, voting, sorting, brunch and bound, suffix tree, hybrid and so on (Chin and Leung, 2005; Davila et al., 2006; Dinh et al., 2011; Bandyopadhyay et al., 2014; Abbas et al, 2012). All these techniques depend on finding an exact optimal solution. The main advantage of all methods in this group is that the methods guarantee global optimality. The main disadvantage of all methods in this group is the running time elapsed to find the motif. The second group contains all methods that find an approximate solution. The methods in this group are based on different strategies such as probabilistic, greedy, and soft computing such as genetic, ant colony and swarm (Nazmul et al., 2007; Chengwei and Jianhua, 2009; Li and Wang, 2010; Li and Wang, 2009; Chan et al., 2008; Clerc and Kennedy, 2002; Li and Tompa, 2006; Sze and Zhao, 2006). All these techniques depend on finding local optimal solution. A common advantage of all these methods in this group is that the suitable running time. The main disadvantage of all these algorithms in this group is that the output is not always true.

In this paper, we study how to solve the DNA motif problem by using particle swarm technique. The paper consists of an introduction, three sections and conclusion. In Section 2, we introduce how to define the components of PSO technique. In Section

* Corresponding Author.
Email Address: a.khedr@uoh.edu.sa
https://doi.org/10.21833/ijaas.2017.06.012

3, we present the PSO algorithm for DNA motif. The experimental study for the proposed algorithm is given in Section 4. Finally, the conclusion is given in Section 5.

## 2. Components of PSO for DNA Motif

In this section we study how to solve DNA motif using PSO technique. So, the section includes the representation of the particle, the population of swarm, the initial values of the population, the fitness function and how to update the velocity and position of the particle.

### 2.1. Particle's representation

There are two ways to represent a motif. The first way is consequence representation. The consensus sequence is the calculated order of most frequent residues, either nucleotide or amino acid, found at each position in a sequence alignment.

It represents the results of a multiple sequence alignment in which related sequences are compared to each other and similar sequence motifs are calculated. The second way is the position weight matrix, PWM. A PWM is a matrix of score values that gives a weighted match to any given substring of fixed length. It has 4 x $l$ matrix of real numbers specifying the probability of each base at each position. In this matrix, we have one row for each symbol of the alphabet and one column for each position in the pattern. In the searching techniques, the representation of motif as a consensus sequence is the best choice and may be lead to fast convergence. From definition of DNA motif, the motif $M$ (or its variants) must exist in each sequence $S_i$.

Therefore, the motif $M$ has $t$ positions, one position for each sequence. So, we can represent each particle as vector consists of $t$ components. The $i$-th component represents the start position of the motif $M$ in the sequence $S_i$. Therefore, the particle $P_x$ represents as $Px = (x_{s_1}, x_{s_2}, \dots, x_{s_t})$, where $x_{s_i}$ is the start position of the particle $P_x$ in the sequence $S_i$.

### 2.2. Generation of the initial population

The performance of PSO depends partially on the initial population, especially when the initial population contains one of the particles that near from the optimal solution. From the definition of DNA motif, the motif $M$ must exist in the first sequence. Therefore, we can generate the initial population from the first sequence of DNA. The size of the set of population is $n$-$l$+1. Each particle in this set has a length $l$. Also, the value of each particle in the initial population is $P_x$=($x$, 0, 0, …, 0), where $1 \le x \le n$-$l$+1.

### 2.3. Fitness function

In our problem, the fitness of a particle is related to the total number of mismatches between pairs of the same location nucleotides which is defined for two strings, $x$ and $y$, of length $l$ each as follows (Eqs. 1 and 2).

$$noMisMatch(x, y) = \sum_{j=1}^{l} MisMatch(x_j, y_j) \quad (1)$$

where

$$MisMatch(a, b) = \begin{cases} 1 & if\ a \neq b \\ 0 & otherwise \end{cases} \quad (2)$$

Therefore, the fitness function for a particle $P_x$ can be computed by using the following score function (Eq. 3).

$$fitness(P_x) = \begin{cases} 1 & \text{if } noMisMatch \text{ between } P_x \text{ in } S_1 \text{ and } l-\text{mer} \le 2d \\ 0 & \text{otherwise} \end{cases} \quad (3)$$

### 2.4. Particles and velocities updating

At each iteration, the particles of the current population are evaluated and updating. The velocity value is calculated according to how far an individual's data is from the target. Since in our problem the data is a sequence, therefore the velocity would describe how different the pattern is from the target, and thus, how much it needs to be changed to match the target.

We define the velocity of a particle as the range of allowed number of mismatches between the new and current motif instances as the range [*0 … 2d*].

The velocity of the particle is calculated according to the following rule. If the number of mismatch between the new and current motif instances is in the range then the velocity of the particle is $V_x$=(0, …, $j$, …, 0), where $j$ represents the position of the new motif instance. Otherwise, the number of mismatch between the new and current motif instances is out the range, we select any random motif instance. So, $j$ represents a random position from 1 to $n$-$l$+1.

According to the new velocity of the particle we can update the position of the particle as follows: $P_x$ ($i$+1)= $P_x$ ($i$) + $V_x$.

## 3. Proposed method

The main objective of our method is based on PSO maximizing the fitness function for the particles. The expected maximum fitness of the particle is $t$ because the motif M must appear in the $t$ sequences. Our method starts initially with a set of particles that is generated from the first sequence of DNA. It consists of two main loops; the first loop represents the number of sequences in DNA while the second loop represents the number of particles in the population. In this loop, each particle searches for the best $l$-mer matching with its and then we calculate the fitness of the particle. We also update the velocity and the position of the particle according to the rule. If the value of the *pbest* for the particle is reach to *gbest* then this particle is a motif. Otherwise, repeat the above process. The pseudocode for this method is as follows:

***Method:*** *DNAMotif_PSO*
***Input:*** *(1) t sequences of length n each. The characters of each sequence belong to {A, C, G,T}. (2) The length of motif l. (3) The maximum number of mutations d.*
***Output:*** *Set of candidate motifs.*
***Begin***
*1.Initialize a population of particles with the following:*
*1.1. Create n-l+1 particles from $S_1$.*
*1.2. Set pbestx =1 for each particle Px.*
*1.3. Set gbest=t*
*2.Repeat the following t-1 iterations, 2 ≤ i < t.*
*For each particle, $P_x$, in the population set do*
*2.1. Scan each l-mer in the sequence $S_i$ to find a best match with $P_x$.*
*2.2. Calculate the fitness of $P_x$.*
*2.3. If fitness($P_x$)=1 then*
*$pbest_x = pbest_x + 1$*
*2.4. If $pbest_x = gbest$ then*
*Add the particle to the set of candidate motif.*
*Otherwise*
*Update the velocity and the position of each particle based on the rules of updating*
*3.If the set of candidate motifs is empty then select the particles have pbest nearest to gbest.*
***End.***

The running time for the first step is $O(n)$, the running time for the second step is $O(t * n^2 * l)$, and the running time for the third step is $O(n)$. Therefore, the overall time of the method is $O(t * n^2 * l)$.

## 4. Experimental study

We evaluated the performance of the proposed method according to different measurements such as running time, storage, and accuracy. Also, we measure the performance of the proposed method on artificial and real biological data. The dataset used in measuring the performance of the method was generated artificially by using the same idea used in many research articles (Davila et al., 2007; Chin and Leung, 2005; Dinh et al., 2011; Bandyopadhyay et al., 2014; Abbas et al., 2012). In the artificial data, each input instance consists of $t$=20 random sequences. Each sequence consists of $n$=600 nucleotides, where the set of nucleotides is {'A', 'C', 'G', 'T'} and each nucleotide of the input sequence has the same occurrence of probability. After that we planted randomly in each sequence a motif $M$ or a variant motif of $M$ of length $l$. Also, in our study we apply the proposed method on different instances of $l$ and $d$.

Table 1 shows the results of comparing our proposed method with two previous methods, voting (Chin and Leung, 2005) and hybrid (Abbas et al., 2012). The running time of each method is the average time of twenty instances of type challenging. The challenging instances have been studied are (11, 3), (13, 4), (15, 5), (17, 6), and (19, 7). The result of our method is faster than the two methods. Also, the performance of the method increases with increase the length of motif, $l$.

Also, from our implementation we observe that the storage required by voting and hybrid methods is very large. But our method required less memory to complete the task. Finally, when we compare our method with the exact solution we found that the same accuracy as the exact solution in some cases. The accuracy of the output is approximately equal to 90% in some other cases.

**Table 1:** Comparison between DNAMotif_PSO and other methods

| L | d | Voting Method (Chin and Leung, 2005) | Hybrid Method (Abbas et al., 2012) | DNAMotif_PSO |
|---|---|---|---|---|
| 11 | 3 | 2.542 s | 2.313 s | 1.01 s |
| 13 | 4 | 38.043 s | 31.552 s | 15.53 s |
| 15 | 5 | 7.254 m | 5.967 m | 2.273 m |
| 17 | 6 | 1.469 h | 1.015 h | 0.401 h |
| 19 | 7 | 17.551 h | 10.4 h | 4.4 h |

We test our method on realistic biological data to find known transcriptional regulatory elements upstream of yeast genes. The data are collected from SCPD (Zhu and Zhang, 1999) which contains many transcription factors for yeast and has been used by previous algorithms (Chin and Leung, 2005; Abbas et al., 2012). Experimental results are shown in Table 2, where the nucleotide base codes are given in Table 3.

**Table 2:** Experimental results on biological data

| Transcription Factor | Published Motif | Detected Planted Motif |
|---|---|---|
| PDR3 | TCCGYGGA | TCCGTGGA |
| GAL4 | CGGNNNNNNNNNNCCG | CGGCGGACTCTCATCCG |
| MATalpha2 | CRTGTWWWW | CATGTAATT |
| MCB | WCGCGW | ACGCGT |

**Table 3:** Nucleotide base codes

| Symbol | Nucleotide base(s) |
|---|---|
| N | A or C or G or T |
| R | A or G |
| W | A or T |
| Y | C or T |

## 5. Conclusion

In this paper, we studied one of the important problems in bioinformatics which is DNA motif finding problem. The methodology used to solve this problem is the particle swarm optimization. We discusses how to represent each particle in DNA motif, how to generate the initial population set, how to compute the fitness of each particle, and then how to update the velocity and the position of each new particle. After that, we list the steps of the new proposed method to find DNA motif. The proposed method is implemented on a real machine and compared with two other methods by using artificial

data. The results of comparison show that our proposed method is faster than the compared methods. Also, our method required less memory. Finally, we tested our method on real biological data to prove that our method can applied on real data.

## Acknowledgment

This research was supported by Research Deanship, Hail University, KSA.

## References

Abbas MM, Abouelhoda M, and Bahig HM (2012). A hybrid method for the exact planted (l, d) motif finding problem and its parallelization. BMC Bioinformatics, 13(17). https://doi.org/10.1186/1471-2105-13-S17-S10

Bandyopadhyay S, Sahni S, and Rajasekaran S (2014). PMS6: A fast algorithm for motif discovery. International Journal of Bioinformatics Research and Applications 2, 10(4-5): 369-383.

Chan TM, Leung KS, and Lee KH (2008). TFBS identification based on genetic algorithm with combined representations and adaptive post-processing. Bioinformatics, 24(3): 341–349.

Chengwei L and Jianhua R (2009). A novel swarm intelligence algorithm for finding DNA motifs. International Journal of Computational Biology and Drug Design, 2(4): 323–339.

Chin FY and Leung HC (2005). Voting algorithms for discovering long motifs. In the 3rd Asia Pacific Bioinformatics Conference, Institute for Infocomm Research, Singapore: 261-271.

Clerc M and Kennedy J (2002). The particle swarm-explosion, stability, and convergence in a multidimensional complex space. IEEE Transactions on Antennas and Propagation, 6(1): 58–73.

Davila J, Balla S, and Rajasekaran S (2006). Space and time efficient algorithms for planted motif search. In the International Conference on Computational Science, Springer Berlin Heidelberg, Heidelberg, Germany: 822-829.

Davila J, Balla S, and Rajasekaran S (2007). Fast and practical algorithms for planted (l, d) motif search. IEEE/ACM Transactions on Computational Biology and Bioinformatics, 4(4): 544-552.

Dinh H, Rajasekaran S, and Kundeti V (2011). PMS5: An efficient exact algorithm for the (l, d)-motif finding problem. BMC Bioinformatics, 12(1): 410-420.

Li N and Tompa M (2006). Analysis of computational approaches for motif discovery. Algorithms for Molecular Biology, 1:8. https://doi.org/8-1-7188-1748/10.1186

Li X and Wang D (2009). An improved genetic algorithm for DNA Motif discovery with public domain information. In: Köppen M, Kasabov N, and Coghill G (eds.), Advances in Neuro-Information Processing: 521-528. Springer, Heidelberg, Germany.

Li X and Wang D (2010). Motif discovery using an immune genetic algorithm. Journal of Theoretical Biology, 264(2): 319-325.

Nazmul R, Chowdhury AR, and Tareeq SM (2007). A novel approach of finding planted motif in biological sequences. In the 10th International Conference on Computer and Information Technology, IEEE: 1-5. https://doi.org/10.1109/ICCITECHN.2007.4579353

Sze SH and Zhao X (2006). Improved pattern-driven algorithms for motif finding in DNA sequences. In the Annual Satellite Conference on Systems Biology and Regulatory Genomics, Springer- Verlag Berlin, Heidelberg, Germany: 198-211.

Zhu J and Zhang MQ (1999). SCPD: A promoter database of the yeast saccharomyces cerevisiae. Bioinformatics, 15(7): 607-611.