Genetics and Molecular Research

A discrete artificial bee colony algorithm for detecting transcription factor binding sites in DNA sequences

D. Karaboga[1] and S. Aslan[2]

[1]Department of Computer Engineering, Erciyes University, Kayseri, Turkey
[2]Department of Computer Engineering, Ondokuz Mayis University, Samsun, Turkey

Corresponding author: S. Aslan
E-mail: selcuk.aslan@bil.omu.edu.tr

Genet. Mol. Res. 15 (2): gmr.15028645
Received March 23, 2016
Accepted April 11, 2016
Published April 26, 2016
DOI http://dx.doi.org/10.4238/gmr.15028645

ABSTRACT. The great majority of biological sequences share significant similarity with other sequences as a result of evolutionary processes, and identifying these sequence similarities is one of the most challenging problems in bioinformatics. In this paper, we present a discrete artificial bee colony (ABC) algorithm, which is inspired by the intelligent foraging behavior of real honey bees, for the detection of highly conserved residue patterns or motifs within sequences. Experimental studies on three different data sets showed that the proposed discrete model, by adhering to the fundamental scheme of the ABC algorithm, produced competitive or better results than other metaheuristic motif discovery techniques.

Key words: Bioinformatics; Motif discovery; Binding sites; Artificial bee colony; Discrete optimization

INTRODUCTION

In recent years, the complete genomes of several model organisms, including mouse, chimpanzee, and human, have been sequenced and investigated through the use of technological advances in data processing and retrieval systems (Tompa et al., 2005). However, the annotation of these raw nucleotide sequences, in particular the identification of specific functional sites that are responsible for the regulation of transcriptional and translational processes (e.g., transcription factor binding sites, donor or acceptor sites, and branch points), still requires the refinement of well-known techniques or the development of new solving approaches (Matys et al., 2003; Tompa et al., 2005). Although mutation and selection over millions of years can result in considerable divergence between the genomic sequences of different organisms, sequences derived from the same ancestral genes or important functional sites are more likely to be conserved than other sequences. The identification of these conserved nucleotide patterns or motifs and their comparison with annotated genomic data, referred to as motif discovery, remains one of the fundamental problems of bioinformatics (Matys et al., 2003; Tompa et al., 2005; Das and Dai, 2007).

Based on the type of combinatorial methods employed by the algorithm, motif discovery algorithms can be categorized into two major groups (Tompa et al., 2005; Li and Tompa, 2006; Das and Dai, 2007). In the first group, motifs are found by utilizing exhaustive counting and comparison techniques (Tompa et al., 2005). While exhaustive enumeration methods guarantee that the most appropriate motifs will be found and while the implementation of specialized data structures such as suffix trees increases usability in identifying short motifs, the conservation level of the motifs is important and can directly affect the produced results (Tompa et al., 2005; Li and Tompa, 2006; Das and Dai, 2007). In the second group of algorithms are methods that estimate sequence or motif models by taking advantage of sound maximum-likelihood procedures or Bayesian interfaces. The expectation-maximization (EM) algorithm, introduced by Lawrence and Reilly (1990), and Gibbs sampling, also proposed by Lawrence et al., are effective statistical techniques that are behind popular motif discovery techniques, such as MEME and Gibbs sampler (Tompa et al., 2005).

Evolutionary computation techniques have attracted the attention of researchers and been used to tackle the motif discovery problem in recent years. Liu et al. (2004) introduced a genetic algorithm (GA)-based technique called FMGA. In this study, mutation and crossover operators were specialized for the motif discovery problem. When a mutation operator was applied to an individual in the population, in order to protect highly conserved regions, position weight matrices were used. A similar procedure was also applied to the crossover operation. Special gap arrangement and penalization procedures were employed for increasing the quality of the offspring (Liu et al., 2004). Another GA-based motif discovery technique called MDGA was proposed by Che et al. (2005). Experimental studies for MDGA showed that the GA-based technique produced more accurate motifs compared to Gibbs sampling (Che et al., 2005). A particle swarm optimization (PSO)-based technique for detecting motifs in amino acid sequences was presented by Chang et al. (2004), and these motifs were able to achieve better true positive hits than those reported in PROSITE.

Multiobjective adaptation of these algorithms is another important topic in the study of motif discovery. Kaya (2007) proposed a GA-based multiobjective solving technique in which conflicting arguments, including the number of sequences used to generate a consensus sequence and the lengths and similarity values of motifs, are optimized simultaneously.

Gonzalez-Alvarez et al. (2010) used a differential evolution (DE) algorithm with Pareto tournaments. They also proposed a multiobjective ABC algorithm with genetic operator in order to address the multiobjective optimization of the motif discovery problem and analyzed its performance on different multiobjective adaptations, including multiterm fitness function and ranking and sorting methodologies (Gonzalez-Alvarez et al., 2011a,b). However, before modeling the motif discovery problem as a multiobjective optimization problem and using multiobjective implementations of evolutionary computational techniques that were originally proposed to solve numerical optimization problems, we should correctly adapt the distinctive steps of these techniques to the discrete problems we are hoping to solve. In this study, we propose a new discrete ABC algorithm called consensusABC, which adheres to the individual effect of the candidate generation approach in the mathematical expression used in the employed and onlooker bee phases. In addition to this, a new neighborhood selection strategy based on the similarity values of the consensus sequences is also introduced and combined with the discretized ABC algorithm. Experimental studies on three different data sets extracted from the TRANSFAC database (Matys et al., 2003) showed that the proposed ABC algorithm outperforms other metaheuristic techniques used for comparison.

MATERIAL AND METHODS

Determination of conserved regions in sequences

In long nucleotide sequences, functional regions and coding or non-coding segments can be identified based on nucleotide composition, codon composition, and hexamer usage or frequencies (Mathé et al., 2002; Wang et al., 2004). Among the large variety of such statistical measures, hexamers (six-nucleotide long sequences) and their occurrence periodicities are one of the primary discriminative variables (Mathé et al., 2002; Wang et al., 2004). The characteristics of hexamer frequency indicate an important aspect of the sequences, namely that evolutionary processes protect or conserve some sequential nucleotides more than other regions.

Consider a given set of N different DNA sequences, each of which contains an equal number of nucleotides L, and suppose that we try to find N different *lmers* (l-nucleotide long sequences) in which conservation or similarity in terms of nucleotide types at the same location is higher than or equal to the conservation levels or similarity values belonging to the other possible group of *lmers*. To generate these mentioned *lmers* from the given set of sequences, select random start positions and then define a vector that holds all start positions $p = (p_1,p_2,...,p_N)$ where $1 \leq p_i \leq (L-l+1)$ and $1 \leq i \leq N$. By using start positions stored in the vector p, the nucleotides included in each *lmer* can be organized into a matrix form in which the ith row corresponds to the *lmer* in the ith initial sequence and the jth column of the ith row corresponds to the (p_i+j-1)th nucleotide of the ith sequence in the set of sequences. When this matrix, also called the alignment matrix, is transformed into a more informative form on the basis of the nucleotide frequencies found in each alignment column, a profile or profile matrix can be obtained, in which there are four rows, each representing a different nucleotide type, and l columns. In the profile or profile matrix, an element in the ith row and jth column of the matrix actually holds the number of times nucleotide type i appears in the jth alignment position. By utilizing the residue type and frequency information stored in the alignment and profile matrices, a more compact representation for the discovered motifs, named a consensus sequence, can be introduced. A consensus sequence is compiled by inserting the nucleotide

occurring most often at each position in the alignment and profile matrices. Generation of alignment and profile matrices and consensus sequences is illustrated over four hypothetical nucleotide sequences in Figure 1.

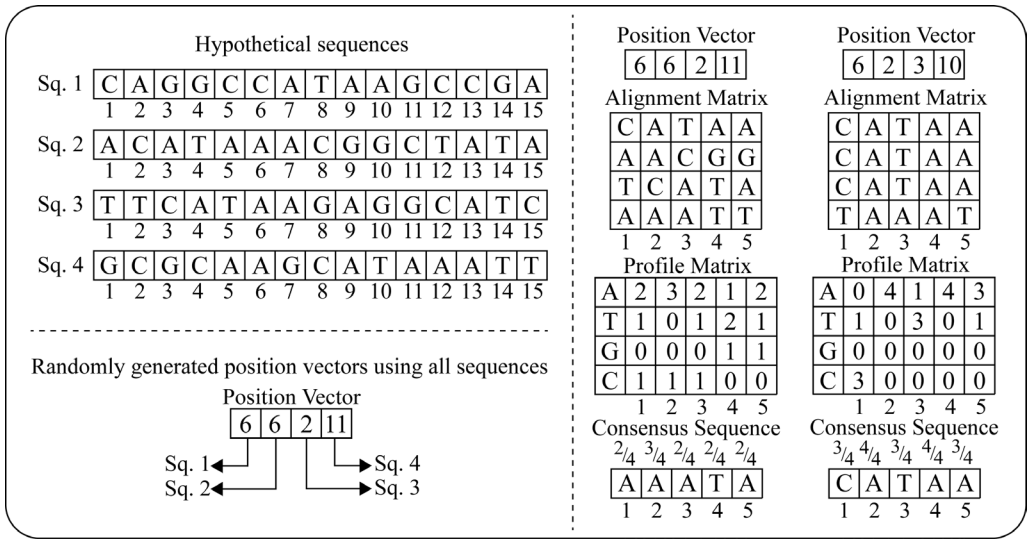

Figure 1. Generation of alignment and profile matrices for four nucleotide motifs.

In the motif discovery process, the representation of the conservation level or similarity value plays a key role in determining the most appropriate start positions. One of the commonly used similarity calculation schemas depends on taking the sum of the maximum values of each column of the alignment matrix and can be formulated as Equation 1 (Lesk, 2002; Jones and Pevzner, 2004; Kaya, 2007; Gonzalez-Alvarez et al., 2010):

$$Sim(P) = \sum_{i=1}^{l} \frac{\max(p(i))}{l \times N} \qquad \text{(Equation 1)}$$

where *max(p(i))* and *l* represent the frequency of the dominant nucleotide of the *i*th column in the profile matrix and the length of the motif, respectively. Another important approach used in similarity value calculation is finding the entropy of the given profile matrix. Given that *P* is a *4×l* profile matrix and *p(i,j)* is the element of *i*th row and *j*th column of *P*, the entropy can be calculated as in Equation 2 (Lesk, 2002; Jones and Pevzner, 2004):

$$Sim(P) = \sum_{j=1}^{l} \sum_{i=1}^{4} \frac{p(i,j)}{N} \log \frac{p(i,j)}{N} \qquad \text{(Equation 2)}$$

Artificial bee colony algorithm

Division of labor and self-organization, which are two remarkable features of intelligent swarms, have led to the emergence of a minimal foraging behavior in honey bee colonies. According to this foraging behavior, bees classified as employed and unemployed foragers minimize energy consumption during the search progress. Employed bees are assigned to exploit a specific nectar source that has been previously explored and are responsible for giving information to unemployed foragers about the quality of the assigned nectar source. The quality of a food source is based on the concentration of its honey, proximity to the hive, and difficulties of extracting nectar. Onlooker bees, which are a subgroup of the unemployed foragers, wait in the hive and attempt to find food sources by means of the information shared by employed bees in the dance area. They watch numerous waggle dances before selecting a food source and the tendency for onlookers to choose a particular food source is directly proportional to the quality of the food sources. If a food source is exploited or abandoned, an employed bee associated with this source becomes a scout bee and searches the environment randomly to find a new food source. Scout bees are another type of unemployed forager and look for new sources without using any information given by the employed bees (Akay and Karaboga, 2012; Karaboga and Aslan, 2015).

The ABC algorithm, a swarm-intelligence based optimization algorithm proposed by Karaboga for solving multi-variable numerical problems, models the foraging behavior of honey bees. The ABC algorithm has been used for optimizing larger sets of constrained or unconstrained numerical and combinatorial problems compared with other swarm-intelligence and evolutionary algorithms (Akay and Karaboga, 2012; Karaboga and Aslan, 2015). When using the ABC algorithm to solve an optimization problem, food sources in the search space correspond to the possible solutions of the problem, and the nectar amount of the food source represents the fitness value of the solution. The main steps of the ABC algorithm, which reflect the cyclical relationship between employed, onlooker, and scout bees, as mentioned above, are summarized in Figure 2.

1: **Initialization:**
2: Send all scout bees to the initial food sources
3: **Repeat**
4: **Employed Bee Phase:**
5: Send all employed bees to new food sources.
6: Apply greedy selection between memorized and newly discovered food sources.
7: **Onlooker Bee Phase:**
8: Calculate probability values of food sources.
9: Send onlooker bees to food sources using probability values.
10: Apply greedy selection between memorized and newly discovered food sources.
11: **Scout Bee Phase:**
12: Memorize best food source found so far.
13: Determine abandoned food source.
14: Send a scout bee for this abandoned food source.
15: **Until maximum cycle number is reached.**

Figure 2. Fundamental steps of the ABC algorithm.

Generating initial food sources

The ABC algorithm begins the optimization progress by randomly generating an initial set of food sources that correspond to the possible solutions. In the ABC algorithm, for a numerical problem that needs to optimize *D* different parameters, the parameter x_{ij}, identified by lower bound x_j^{min} and upper bound x_j^{max}, of the solution vector x_i in the initial food source population in which there are *SN* different solution vectors is formulated as given in Equation 3 (Akay and Karaboga, 2012; Karaboga and Aslan, 2015):

$$x_{ij} = x_j^{\min} + rand(0,1)\left(x_j^{\max} - x_j^{\min}\right) \quad i = 1,2,...,SN \quad and \quad j = 1,2,...,D \quad \text{(Equation 3)}$$

As mentioned previously, the motif discovery problem could be considered a combinatorial optimization problem for which the main objective is to find the appropriate start positions in a manner that maximizes the number of matched residues in each alignment column. Taking the discrete structure of the problem and the changing effect of the evolutionary mechanisms into consideration, specific procedures of the ABC algorithm should be altered to solve the motif discovery problem more effectively. When solving the motif discovery problem with the ABC algorithm, the food sources in the search space correspond to the start points and the nectar amount of the food source is represented by the similarity value of the conserved regions whose start positions are stored in the position vector.

Sending bees to new food sources

In the ABC algorithm, each food source is associated with only one artificial bee, and this bee attempts to produce a new food source using the location information in its memory. If the nectar quality of the new food source is better than a known source, the bee will decide to forget the previous food source information and keep the new food source information in its memory, which could be considered a greedy selection mechanism, for utilization in the next cycle. The mathematical expression using both the employed and onlooker bees to produce a candidate food source in the neighborhood of the memorized food source is given in Equation 4 (Akay and Karaboga, 2012; Bolaji et al., 2013; Karaboga and Aslan, 2015; Celik et al., 2015):

$$v_{ij} = x_{ij} + \phi_{ij}\left(x_{ij} - x_{kj}\right) \quad \text{(Equation 4)}$$

f_{ij} is a random number between *-1* and *1*, $k \in 1, 2, \ldots, SN$, and $j \in 1, 2, \ldots, D$, where *SN* and *D* denote the number of food sources and the dimensions of the solution vectors, respectively, and are randomly chosen indexes. Although the value of *k* is randomly determined, it should be noted that identical values are not assigned to the *k* and *i* indices. v_{ij} is the newly created *j*th parameter for the solution vector v_i, whose parameters have the same value as the solution vector x_i except for the randomly selected *j*th parameter value. However, when solving the motif discovery problem with the ABC algorithm, the mathematical model used to generate candidate solutions should be predisposed by considering the discrete structure and

representation of the food sources. In the proposed model for the discretized ABC algorithm, a neighbor is determined by utilizing the consensus sequences of the profile matrices. If an employed or onlooker bee will be sent to a new food source by referencing the *i*th memorized food source, which also corresponds to the position vector and its alignment and profile matrix counterparts, selection of the *k*th neighbor to generate a candidate position vector is directly related to the similarity between consensus sequences. After deciding on the *k*th food source, the *i*th food source will be updated with a random start position taken from the *k*th food source. In order to understand the basic properties of the proposed model, the creation of the candidate food source around the neighborhood of the *i*th food source with the *k*th food source is examined in Figure 3.

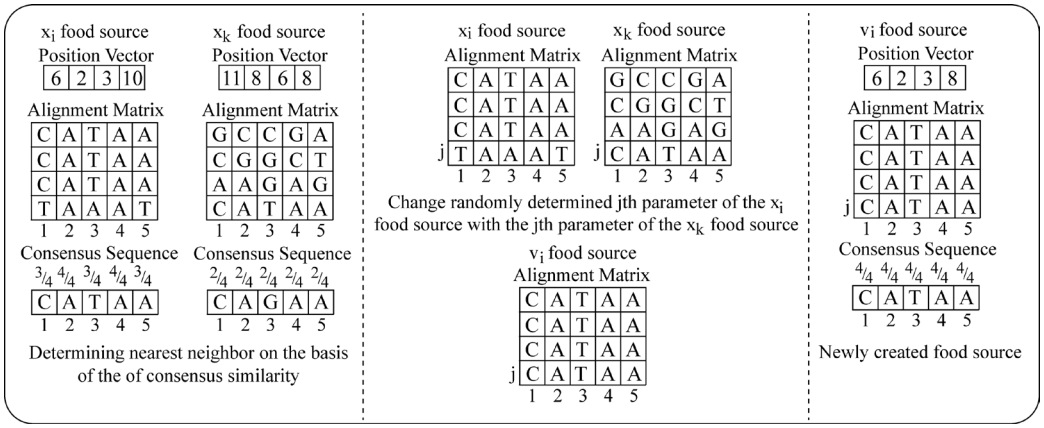

Figure 3. Generation of new food source by employed or onlooker bees.

Choosing or abandoning a food source

The ABC algorithm accommodates a preference for a food source by an onlooker bee with the nectar amount of that food source. After employed bees have shared the information stored in their memory on the dance area, an onlooker bee chooses a food source depending on the probability value associated with that food source. The probability of a food source, which increases with the nectar quality of the source, is calculated as given in Equation 5:

$$p_i = \frac{fitness_i}{\sum_j^{SN} fitness_j} \qquad \text{(Equation 5)}$$

where $fitness_i$ is the fitness value of the solution represented by the food source at position *i*, and *SN* is the number of food sources.

The process undertaken by onlooker bees in choosing a food source is managed with a selection schema similar to the roulette wheel, in which a higher quality food source is more likely to be chosen. As detailed previously, the exploitation process is carried out with

employed and onlooker bees. In a robust search, a good balance between exploitation and exploration processes should be maintained. If a food source cannot be improved through a predetermined number of iterations, the employed bee associated with that food source will become a scout bee and leave the food source to start a random search operation. The number of cycles before abandonment of a source is an important control parameter of the ABC algorithm called the *limit*. In the ABC algorithm, the food source for which the *limit* value has been exceeded most often is abandoned, with one employed bee becoming a scout bee in each cycle (Akay and Karaboga, 2012; Karaboga and Aslan, 2015).

RESULTS

In this section, we conducted experimental studies to analyze the performance of the proposed model. To detect conserved nucleotide groups among sequences with the consensusABC algorithm, we used three real data sets selected from the TRANSFAC database (Matys et al., 2003). The sequences in each data set store information about a functional transcription factor binding site, which plays a vital role in gene expression. The first data set, identified with the code hm03r, contained 10 human sequences of 1500 bp (Matys et al., 2003). The second and third data sets, identified as yst04r and yst08r, contained 7 and 15 yeast sequences, respectively, of 1000 bp each (Matys et al., 2003).

For fair comparison with other metaheuristics, population or colony size and the number of maximum cycles or generations, which are common control parameters and directly determine the total number of evaluations, were chosen to be 200 and 3000, respectively (Kaya, 2007; Gonzalez-Alvarez et al., 2010, 2011a). In order to make a more detailed investigation of the proposed discrete model based on similarity between consensus sequences of the ABC algorithm, we used six different values for the *limit* parameter: 50, 100, 200, 300, 500, and a. For each data set and the selected motif length, 30 independent runs were carried out with different random seeds (Kaya, 2007; Gonzalez-Alvarez et al., 2010, 2011a). The highest and mean similarity values and standard deviations of the 30 runs for each combination of parameters are given in Tables 1-3 for the hm03r, yst04r, and yst08r data sets.

Table 1. Mean and highest similarity values and standard deviations for hm03r data set.

Motif length	*limit* = 50			*limit* = 100			*limit* = 200		
	Mean	Best	Std. Dev.	Mean	Best	Std. Dev.	Mean	Best	Std. Dev.
9	0.84185	0.86666	1.2945e-02	0.84000	0.85555	1.1149e-02	0.82777	0.85555	1.6180e-02
10	0.82600	0.86000	1.3287e-02	0.82400	0.85000	9.6846e-03	0.81600	0.84000	1.3287e-02
11	0.81121	0.83636	1.1861e-02	0.80939	0.82727	1.2515e-02	0.80606	0.83636	1.3363e-02
14	0.77904	0.80714	1.2001e-02	0.77690	0.80000	1.0390e-02	0.77119	0.79285	1.0854e-02
Motif length	*limit* = 300			*limit* = 500			*limit* = i		
	Mean	Best	Std. Dev.	Mean	Best	Std. Dev.	Mean	Best	Std. Dev.
9	0.82370	0.86666	1.9732e-02	0.81111	0.84444	1.4294e-02	0.68037	0.73333	1.8611e-02
10	0.80900	0.85000	1.5833e-02	0.78966	0.82000	1.5421e-02	0.66600	0.73000	2.6986e-02
11	0.79333	0.82727	1.3491e-02	0.77909	0.80909	1.6041e-02	0.66030	0.73636	3.4139e-02
14	0.75833	0.77857	1.1267e-02	0.75214	0.77857	1.6476e-02	0.62738	0.68571	2.0979e-02

From the results given in Tables 1-3, it can be clearly seen that the consensus ABC algorithm produces more efficient results, in terms of mean and highest similarity values, when the *limit* parameter value is set to 50. The similarity values of the consensus ABC algorithm with the support values set to equal the number of sequences in each data set were calculated utilizing Equation 1 (Kaya, 2007; Gonzalez-Alvarez et al., 2010, 2011a) and compared with the MOGOMOD (Kaya, 2007), DEPT (Gonzalez-Alvarez et al., 2010) and MOABC (Gonzalez-

Table 2. Mean and highest similarity values and standard deviations for yst04r data set.

Motif length	*limit* = 50			*limit* = 100			*limit* = 200		
	Mean	Best	Std. Dev.	Mean	Best	Std. Dev.	Mean	Best	Std. Dev.
8	0.92857	0.96428	1.5553e-02	0.92738	0.96428	1.5506e-02	0.91904	0.94642	1.8000e-02
9	0.90423	0.93650	1.4727e-02	0.90740	0.95238	1.3236e-02	0.90105	0.92063	8.0001e-03
13	0.85824	0.87912	1.1668e-02	0.85897	0.87912	9.1638e-03	0.85164	0.87912	1.1083e-02
22	0.77770	0.80519	1.0042e-02	0.77554	0.80519	9.4460e-03	0.77034	0.77922	6.4898e-03
Motif length	*limit* = 300			*limit* = 500			*limit* = i		
	Mean	Best	Std. Dev.	Mean	Best	Std. Dev.	Mean	Best	Std. Dev.
8	0.91666	0.94642	1.8952e-02	0.90178	0.92857	1.7389e-02	0.77797	0.82142	2.9911e-02
9	0.89206	0.92063	1.4678e-02	0.87883	0.90476	1.4727e-02	0.76190	0.87301	3.7976e-02
13	0.84432	0.85714	1.0031e-02	0.83260	0.86813	1.7238e-02	0.73479	0.79120	2.8383e-02
22	0.76839	0.79220	1.1095e-02	0.76255	0.79870	1.1525e-02	0.66385	0.69480	1.9040e-02

Table 3. Mean and highest similarity values and standard deviations for yst08r data set.

Motif length	*limit* = 50			*limit* = 100			*limit* = 200		
	Mean	Best	Std. Dev.	Mean	Best	Std. Dev.	Mean	Best	Std. Dev.
10	0.87636	0.90000	1.2079e-02	0.87787	0.90000	1.0857e-02	0.86969	0.89090	1.0765e-02
11	0.86281	0.89256	1.0086e-02	0.86115	0.89256	1.1151e-02	0.85206	0.86776	1.2152e-02
14	0.81688	0.85064	1.0288e-02	0.81753	0.84415	1.0697e-02	0.81125	0.83116	1.0079e-02
21	0.76277	0.77922	6.8588e-03	0.76305	0.78355	8.3489e-03	0.75800	0.77489	7.3982e-03
Motif length	*limit* = 300			*limit* = 500			*limit* = i		
	Mean	Best	Std. Dev.	Mean	Best	Std. Dev.	Mean	Best	Std. Dev.
10	0.86303	0.89090	1.4110e-02	0.84697	0.88181	1.6206e-02	0.72545	0.79090	3.0796e-02
11	0.84297	0.88429	1.6241e-02	0.83719	0.87603	1.7785e-02	0.71680	0.76033	2.1478e-02
14	0.80389	0.81818	6.4711e-03	0.78831	0.81168	1.0174e-02	0.69913	0.75324	2.1184e-02
21	0.75093	0.77056	7.7717e-03	0.74603	0.76190	8.9033e-03	0.65238	0.69264	2.0967e-02

Alvarez et al., 2011a) algorithms. For the MOGAMOD, DEPT, and MOABC algorithms, each experiment was performed 30 times with the given population and maximum number of cycles and the final Pareto front was created by combining the Pareto fronts found in all 30 independent runs. In other words, the final Pareto front contains the solutions with the highest similarity values. When considering these situations, the highest similarity values of the motifs of different lengths obtained by the consensus ABC algorithm with a *limit* value of 50 were compared to the results from MOGAMOD, DEPT, and MOABC. From a comparison of results (Tables 4-6), it can be seen that the consensus ABC algorithm outperforms all of other algorithms. The neighbor selection strategy based on similarity values between consensus sequences improved the probability of the selection of a food source in which the position vector might contain more suitable elements that can enhance the similarity value. The efficiency of the consensus ABC algorithm can be seen at the level of matched columns in the produced alignment matrices. The alignment matrix produced by the consensus ABC algorithm for the hm03r data set for 10-nucleotide motifs contains at least 3 more matches than that of the DEPT algorithm, at least 5 more matches than that of the MOABC algorithm, and at least 7 more matches than that of the MOGAMOD algorithm where the dominant residue types of the columns are concerned.

Table 4. Comparison of consensus ABC algorithm with MOGAMOD algorithm.

Data set	Motif length	Similarity	
		MOGAMOD (Kaya, 2007)	Consensus ABC
hm03r	9	0.81	**0.86**
	10	0.79	**0.86**
	11	0.74	**0.83**
yst04r	8	0.84	**0.96**
	9	0.80	**0.93**
yst08r	10	0.77	**0.90**
	11	0.80	**0.89**

Table 5. Comparison of consensus ABC algorithm with DEPT algorithm.

Data set	Motif length	Similarity	
		DEPT (Gonzalez-Alvarez et al., 2010)	Consensus ABC
hm03r	9	0.85	**0.86**
	10	0.83	**0.86**
	11	0.81	**0.83**
	14	0.79	**0.80**
yst04r	8	0.96	**0.96**
	9	0.92	**0.93**
	22	**0.80**	**0.80**
yst08r	10	0.89	**0.90**
	11	0.88	**0.89**
	21	0.77	**0.78**

Table 6. Comparison of consensus ABC algorithm with MOABC algorithm.

Data set	Motif length	Similarity	
		MOABC (Gonzalez-Alvarez et al., 2011a)	Consensus ABC
hm03r	9	0.84	**0.86**
	10	0.81	**0.86**
	11	0.80	**0.83**
yst04r	8	0.94	**0.96**
	9	0.90	**0.93**
	13	0.84	**0.87**
yst08r	11	0.85	**0.89**
	14	0.80	**0.85**

Additional assessment of the consensus ABC algorithm was carried out by analyzing the running times observed with the *limit* parameter value of 50. For each of the 30 independent experiments for each data set and motif length, a system equipped with an Intel® i7 2600 CPU at 3.40 GHz with 4 GB of main memory running the Fedora 22 operating system was used, and the total elapsed time until completion of the 3000 cycles was recorded in seconds. Average elapsed times over the 30 runs are shown in Figure 4, where it can be seen that there is a near-linear relationship between the length of the motif and the elapsed run time. In addition, when comparing across data sets, we can see that more time is required to find motifs in data sets containing more sequences or longer sequences.

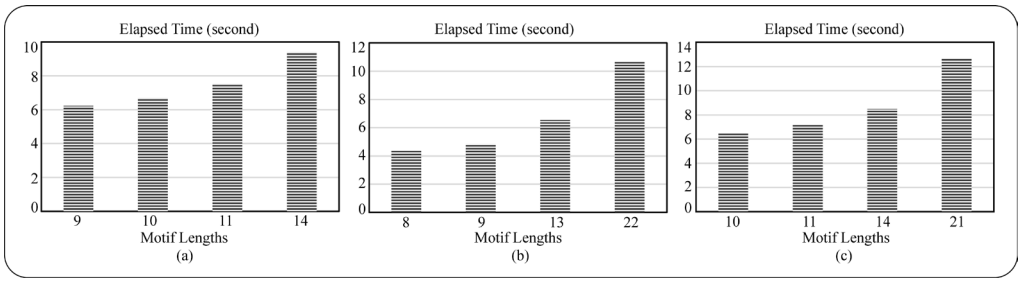

Figure 4. Comparison of the average elapsed times for hm03r (**a**), yst04r (**b**), and yst08r (**c**) data sets.

DISCUSSION

In this paper, we proposed a new discrete ABC algorithm, the consensus ABC algorithm, for solving motif discovery problems. The consensus ABC algorithm has been

tested on three different data sets with different motif lengths and *limit* parameter values. In addition, performance of the consensus ABC algorithm was compared with a GA-based technique (MOGAMOD), a DE-based technique (DEPT), and a genetic operator-based ABC algorithm (MOABC). Comparison of results showed that the consensus ABC algorithm, in which the neighbor selection strategy references the similarity analysis between consensus counterparts of the solutions, provided comparable or better results with appropriate values of the *limit* parameter. Due to the promising results obtained with the consensus ABC algorithm, use of a consensus-based neighbor selection strategy could be more efficient on the multiobjective optimization techniques and parallel implementation on distributed or shared memory architectures.

REFERENCES

Akay B and Karaboga D (2012). Artificial bee colony for large scale problems and engineering design optimization. *J. Intell. Manuf.* 23: 1001-1014. http://dx.doi.org/10.1007/s10845-010-0393-4

Bolaji ALA, Khader AT, Al-Betar MA and Awadallah MA (2013). Artificial bee colony algorithm, its variants and applications: a survey. *J. Theor. Appl. Inf. Technol.* 47: 434-459.

Celik M, Koylu F and Karaboga D (2015). CoABCMiner: an algorithm for cooperative rule classification system based on artificial bee colony. *Int. J. Artif. Intell. Tools* 24: 1-40.

Chang BC, Patnaweera A, Halgamuge SK and Watson HC (2004). Particle swarm optimisation for protein motif discovery. *Genet. Program. Evolvable Mach.* 5: 203-214. http://dx.doi.org/10.1023/B:GENP.0000023688.42515.92

Che D, Song Y and Rashedd K (2005). MDGA: motif discovery using a genetic algorithm. Proceedings of the 7th Conference on Genetic and Evolutionary Computation (GECCO'2005), Washington, 447-452.

Das MK and Dai HK (2007). A survey of DNA motif finding algorithms. *BMC Bioinformatics* 8 (Suppl 7): S21. http://dx.doi.org/10.1186/1471-2105-8-S7-S21

Gonzalez-Alvarez DG, Vega-Rodriguez M, Gomez-Pulido J and Sanchez-Perez JM (2010). Solving the motif discovery problem by using differential evolution with pareto tournaments. IEEE Congress on Evolutionary Computation (CEC-2010), Barcelona, 1-8.

Gonzalez-Alvarez DG, Vega-Rodriguez M, Gomez-Pulido J and Sanchez-Perez JM (2011a). Evolutionary Computation, Machine Learning and Data Mining. Finding motifs in DNA sequences applying a multiobjective artificial bee colony algorithm (Pizzuti C, Ritchie MD and Giacobini M, eds.). Springer Berlin Heidelberg, Heidelberg, 89-100.

Gonzalez-Alvarez DG, Vega-Rodriguez M, Gomez-Pulido J and Sanchez-Perez JM (2011b). Evolutionary Computation, Machine Learning and Data Mining. Comparing Multiobjective Artificial Bee Colony Adaptations for Discovering DNA Motifs (Giacobini M, Vanneschi L and Bush WS, eds.). Springer Berlin Heidelberg, Heidelberg, 110-121.

Jones NC and Pevzner P (2004). An introduction to bioinformatics. MIT Press, Cambridge.

Karaboga D and Aslan S (2015). A new emigrant creation strategy for parallel artificial bee colony algorithm. 9th International Conference on Electrical and Electronics Engineering (ELECO-2015), Bursa, 689-694.

Kaya M (2009). Multi-objective genetic algorithm for motif discovery. *Exp. Syst. Appl.* 36: 1039-1047. http://dx.doi.org/10.1016/j.eswa.2007.11.008

Lawrence CE and Reilly AA (1990). An expectation maximization (EM) algorithm for the identification and characterization of common sites in unaligned biopolymer sequences. *Proteins* 7: 41-51. http://dx.doi.org/10.1002/prot.340070105

Lesk AM (2002). Introduction to bioinformatics. Oxford University Press Incorporation, Oxford.

Li N and Tompa M (2006). Analysis of computational approaches for motif discovery. *Algorithms Mol. Biol.* 1: 8. http://dx.doi.org/10.1186/1748-7188-1-8

Liu FFM, Chen JJP, Chen RM, Chen SN, et al. (2004). FMGA: finding motifs by genetic algorithm. 4th IEEE Symposium on Bioinformatics and Bioengineering (BIBE-2004), 459-466.

Mathé C, Sagot MF, Schiex T and Rouzé P (2002). Current methods of gene prediction, their strengths and weaknesses. *Nucleic Acids Res.* 30: 4103-4117. http://dx.doi.org/10.1093/nar/gkf543

Matys V, Fricke E, Geffers R, Gössling E, et al. (2003). TRANSFAC: transcriptional regulation, from patterns to profiles. *Nucleic Acids Res.* 31: 374-378. http://dx.doi.org/10.1093/nar/gkg108

Tompa M, Li N, Bailey TL, Church GM, et al. (2005). Assessing computational tools for the discovery of transcription factor binding sites. *Nat. Biotechnol.* 23: 137-144. http://dx.doi.org/10.1038/nbt1053

Wang Z, Chen Y and Li Y (2004). A brief review of computational gene prediction methods. *Genomics Proteomics Bioinformatics* 2: 216-221.