*International Journal of Innovative Computing, Information and Control*
Volume **8**, Number **10(B)**, October **2012**

ICIC International ©2012 ISSN 1349-4198
pp. **7329–7339**

# AN ANT COLONY SYSTEM FOR SOLVING DNA SEQUENCE DESIGN PROBLEM IN DNA COMPUTING

Farhaana Yakop[1], Zuwairie Ibrahim[2], Amar Faiz Zainal Abidin[3]
Zulkifli Md. Yusof[3], Mohd Saberi Mohamad[4]
Khairunizam Wan[5] and Junzo Watada[6]

[1]Faculty of Electrical and Electronic Engineering
Universiti Tun Hussein Onn Malaysia
86400 Parit Raja, Batu Pahat, Johor, Malaysia

[2]Faculty of Electrical and Electronic Engineering
Universiti Malaysia Pahang
26600 Pekan, Pahang, Malaysia
zuwairie.ibrahim@gmail.com

[3]Faculty of Electrical Engineering
[4]Faculty of Computer Science and Information Systems
Universiti Teknologi Malaysia
81310 UTM Skudai, Johor, Malaysia

[5]School of Mechatronic
Universiti Malaysia Perlis
02600 Arau, Perlis, Malaysia

[6]Graduate School of Information, Production and Systems
Waseda University
2-7 Hibikino, Wakamatsu, Kita-Kyushu 808-0135, Japan

Received March 2011; revised July 2011

Abstract. *Deoxyribonucleic acid (DNA) is a nucleic acid that contains the genetic information used in the development and functioning of all known existing organisms. In DNA computing, a set of DNA sequences is involved in solving an optimisation problem. The design of those sequences is difficult because of the frequency of DNA sequence mismatch hybridisations. In this paper, an Ant Colony System approach for DNA sequence design is proposed to solve this DNA sequence design problem. A 4-node state transition machine was used in this study as the computation model. During the implementation, each ant was placed randomly at a start node and then moved according to the state transition rule. Once all of the ants completed the tour, the objective function was computed. This process was repeated until the maximum iteration was obtained. Seven ants were used to design seven sequences that were 20 nucleobases in length. The results showed that a set of usable DNA sequences can be produced using this method, which is better than previous approaches using the Genetic Algorithm and Multi-Objective Evolutionary Algorithm.*
**Keywords:** DNA sequence design, DNA computing, Ant colony system

1. **Introduction.** DNA has certain unique properties, such as self-assembly and self-complementary, which makes it capable of saving an enormous amount of data and performing massive parallel reactions. DNA is a polymer of nucleic acids that is assembled from a series of monomers. Monomers, which form the building blocks of nucleic acids, are called nucleotides. Each nucleotide contains a sugar (deoxyribose), a phosphate group,

and one nucleobase. There are four types of nucleobases that can be classified as either purines or pyrimidines. Adenine (A) and guanine (G) nucleobases are classified as purines, and thymine (T) and cytosine (C) nucleobases are classified as pyrimidines. DNA computing is an unconventional computational process that requires data from *in vitro* experiments to be completed successfully [1]. DNA computing has shown potential by solving several mathematical problems, such as graph and statistics problems [2]. *In vitro* experiments are not error-free, and analogously, error can occur in computational experiments. Hence, many researchers have focused on the design of DNA sequences in order to minimise error in DNA computation.

The main objective of DNA sequence design is to prevent mismatch hybridisation among sequences in the data set. Avoidance of mismatch hybridisation ensures that the generated DNA sequences are unique and cannot be hybridised with other sequences. Previous studies have proposed a variety of DNA sequence design approaches [3-8] and applications of DNA sequence design can be seen in several areas [9-11].

2. **DNA Sequence Design Problem.** DNA sequence design requires the following four parameters: $H_{measure}$; *similarity*; *hairpin*; *continuity*. $\mathrm{GC}_{content}$ and *melting temperature* are used as constraints and are defined by the user [12]. In general, the DNA sequence design can be formulated as:

$$\min f_{DNA} = \sum_i \omega_i f_i = f_{Hmeasure} + f_{similarity} + f_{continuity} + f_{hairpin} \tag{1}$$

where $f_i$ is the objective function for each $i \in \{H_{measure}, \textit{similarity}, \textit{hairpin}, \textit{continuity}\}$ and $\omega_i$ is the weight for each $f_i$. In this study, weights are set to one. For convenience, basic notations used for the objectives and constraints formulation are shown in Table 1. In addition, the following notations are used.

$$bp(a,b) = \begin{cases} 1 & a = \bar{b} \\ 0 & \text{otherwise} \end{cases} \tag{2}$$

$$eq(a,b) = \begin{cases} 1 & a = b \\ 0 & \text{otherwise} \end{cases} \tag{3}$$

$$T(i,j) = \begin{cases} i & i > j \\ 0 & \text{otherwise} \end{cases} \tag{4}$$

For a given sequence $x \in \Lambda^*$, the number of non-blank nucleotides is defined as

$$length(x) = \sum_{i=1}^{|x|} n(x_i) \tag{5}$$

Table 1. Basic notations

| Notation | Description |
|---|---|
| $\Lambda$ | {A, C, G, T} |
| $x, y \in \Lambda$ | $x, y$ = {A, C, G, T} |
| $\lvert x \rvert$ | length of $x$ |
| $x_i$ $(1 \le i \le \lvert x \rvert)$ | $i$th nucleotide from 5'end of sequence $x$ |
| $\Sigma$ | A set of $n$ sequences with the same length $l$ |
| $\Sigma_i$ | $i$th member of $\Sigma$ |
| $\bar{a}$ | complementary base of $a$ |
| $l$ | length of sequence |
| $n$ | number of sequences |

where

$$n(a) = \begin{cases} 1 & a \in \Lambda \\ 0 & \text{otherwise} \end{cases} \tag{6}$$

A shift of sequence $x$ by $i$ bases is denoted as follows:

$$shift(x, i) = \begin{cases} (-)^i x_1 \cdots x_{l-1} & i \geq 0 \\ x_{i+1} \cdots x_l (-)^i & i < 0 \end{cases} \tag{7}$$

2.1. **Objective functions.** In this paper, four objective functions are selected for DNA sequences design. The $H_{measure}$ term and similarity objectives are necessary to prevent miss-hybridisation during any experiment using DNA-based technology. Miss-hybridisation reduces the reliability and efficiency of DNA computation. Alternatively, the hairpin and continuity objectives are important in DNA-based technologies to prevent undesired secondary structures that can which produce undesired DNA complexes.

2.1.1. $H_{measure}$. The $H_{measure}$ term computes the number of complementary nucleotides required to prevent cross-hybridisation of two sequences. Cross-hybridisation occurs when bases of two sequences hybridise with their complements at the cross position. $H_{measure}$ is divided into two terms, $H_{dis}$ and $H_{con}$. The $H_{dis}$ term is for the overall complementary and the $H_{con}$ term is the penalty for the continuous complementary region. For $H_{measure}$ calculations, two strands of sequences are placed in parallel. While one sequence is stationary, the reverse complement of the second sequence is shifted from one end to the other. For each position shift of the sequence, the complementarity between the two sequences is calculated. The $H_{measure}$ objective function, $f_{Hmeasure}(\Sigma)$, is given as

$$f_{Hmeasure}(\Sigma) = \sum_{i=1}^{n} \sum_{j=1}^{n} H_{measure}(\Sigma_i, \Sigma_j) \tag{8}$$

where $\Sigma_i$ and $\Sigma_j$ are anti-parallel to one other. This means that the sequences have different direction, such that the first sequence is in the 5'→3' direction and the second sequence is in the 3'→5' direction. For a particular $i$ and $j$, where $i \neq j$, the $H_{measure}$ between two sequences, $x$ and $y$, is calculated using

$$H_{measure}(x, y) = \max_{|i|<l-1} (h_{dis}(x, shift(rev(y), i)) + h_{con}(x, shift(rev(y), i))) \tag{9}$$

where

$$h_{dis}(x, y) = T\left(\sum_{i=1}^{l} bp(x_i, y_i), H_{dis} \times length_{nb}(y)\right) \tag{10}$$

$$h_{con}(x, y) = \sum_{i=1}^{l} T(cbp(x, y, i), H_{con}) \tag{11}$$

$H_{dis}$ is a real value between 0 and 1, and $H_{con}$ is an integer between 1 and $l$. Both values are set by the user and $T$ is determined using Equation (4). In Equation (12),

$$cbp(x, y, i) = \begin{cases} c & \text{if } \exists c,\ s.t. \quad bp(x_i, y) = 0,\ bp(x_{i+j}, y_{i+j}) = 1 \text{ for } 1 \leq j \leq c, \\ & bp(x_{i+c+1}, y_{i+c+1}) = 0 \\ 0 & \text{otherwise} \end{cases} \tag{12}$$

2.1.2. *Similarity.* The *similarity* objective function computes the similarity between two given sequences that are in the same direction. The *similarity* term has to be minimised in order to keep each sequence as unique as possible. The *similarity* fitness function, $f_{similarity}(\Sigma)$, is formulated as

$$f_{similarity}(\Sigma) = \sum_{i=1}^{n} \sum_{j=1, j\neq i}^{n} similarity(\Sigma_i, \Sigma_j) \tag{13}$$

where $\Sigma_i$ and $\Sigma_j$ are parallel to each other. *Similarity* $(x, y)$ also consists of two terms, i.e., $s_{dis}$ and $s_{con}$. The $s_{dis}$ term represents the overall complementary *similarity* calculation and the $s_{con}$ term represents the penalty for the continuous complementary region. The calculation for *similarity* between two sequences is shown in Equation (14).

$$similarity(x, y) = \max_{|i|<l-1} \left(s_{dis}(x, shift(y, i)) + s_{con}(x, shift(y, i))\right) \tag{14}$$

Note that $S_{dis}$ is a real value between 0 and 1, and $S_{con}$ is an integer between 1 and $l$. Both values are set by the user. In Equation (14),

$$s_{dis}(x, y) = T\left(\sum_{i=1}^{l} eq(x_i, y_i), S_{dis} \times length_{nb}(y)\right) \tag{15}$$

$$s_{con}(x, y) = \sum_{i=1}^{l} T(ceq(x, y, i), S_{con}) \tag{16}$$

$$ceq(x, y, i) = \begin{cases} c & \text{if } \exists c,\ s.t. \quad eq(x_i, y_i) = 0,\ eq(x_{i+j}, y_{i+j}) = 1 \\ & \text{for } 1 \leq j \leq c,\ eq(x_{i+c+1}, y_{i+c+1}) = 0 \\ 0 & \text{otherwise} \end{cases} \tag{17}$$

2.1.3. *Hairpin.* The *hairpin* objective function calculates the probability of a single-stranded DNA to form a secondary structure, particularly a hairpin structure. The *hairpin* term can be formulated as

$$f_{hairpin}(\Sigma) = \sum_{i=1}^{n} hairpin(\Sigma_i) \tag{18}$$

where the calculation of hairpin for a sequence is given as

$$hairpin(x) = \sum_{r=R_{\min}}^{l-2*P_{\min}} \sum_{p=P_{\min}}^{l-P_{\min}-r} T\left(\sum_{i=1}^{pinlen(p,r)} bp(x_{p+1-i}, x_{p+r+i}), \frac{pinlen(p, r, i)}{2}\right) \tag{19}$$

with $pinlen(p, r, i) = \min(p + i, l - r - i - p)$, where $p$ is a pair and $r$ is a ring.

2.1.4. *Continuity.* The *continuity* objective function computes the number of continuous bases that are the same in a single-strand of DNA. *Continuity* for a set of sequences, $\Sigma$, is defined as

$$f_{continuity}(\Sigma) = \sum_{i=1}^{n} continuity(\Sigma_i) \tag{20}$$

where

$$continuity(x) = \sum_{1\leq i\leq l} \left(\sum_{a\in \Lambda nb} T(c(a, i), t)^2\right) \tag{21}$$

Here, $t$ is a threshold parameter, which denotes the minimum number of base repetitions in a sequence, $l$ is the length of the sequence, and

$$c(a,i) = \begin{cases} n & \text{if } \exists n,\ s.t. \quad eq(a_i, a_{i+j}) = 1 \text{ for } 1 \le j < n, \\ & eq(a_i, a_{i+n}) = 0 \\ 0 & \text{otherwise} \end{cases} \tag{22}$$

2.2. **Constraints in DNA sequence design.** Generally, it is preferable to have DNA sequences which behave uniformly in fundamental chemical reactions during the *in vitro* computation. Two constraints, namely $\mathrm{GC}_{content}$ and melting temperature, $T_m$, can be employed to ensure uniform chemical characteristics. These constraints are discussed in more as follows.

2.2.1. *$GC_{content}$.* The $\mathrm{GC}_{content}$ term is the percentage of G and C in a sequence. Since GC content can affect the chemical properties of DNA sequences, it is an important constraint in DNA sequence design. The $\mathrm{GC}_{content}$ constraints of a sequence is expressed as

$$\mathrm{GC}_{\min} \le \mathrm{GC}_{\mathrm{content}} = (y\mathrm{G} + z\mathrm{C})/(w\mathrm{A} + x\mathrm{T} + y\mathrm{G} + z\mathrm{C}) \le \mathrm{GC}_{\max} \tag{23}$$

where $w$A, $x$T, $y$G, and $z$C are the numbers of A, T, G, and C in the sequence, respectively, and the $\mathrm{GC}_{\min}$ and $\mathrm{GC}_{\max}$ values are determined by the user.

2.2.2. *Melting temperature.* Melting temperature, $T_m$, is the temperature where half of the double-stranded DNA starts to break into its single-stranded form [13]. $T_m$ can be calculated using the following equation, which is based on Nearest-neighbour formulation:

$$T_{m(\min)} \le T_m(x) = \frac{\Delta H}{\Delta S + R \ln C_T} + 16.6 \log(\mathrm{Na}^+) \le T_{m(\max)} \tag{24}$$

where $\Delta H$ and $\Delta S$ are the enthalpy and entropy changes of the annealing reaction, respectively. The universal gas constant (Boltzmann's constant), $R$, is $\left(\frac{1.987\mathrm{cal}}{\mathrm{mol}^\circ\mathrm{C}}\right)$. $C_T$ is the total oligonucleotide strand concentration. For non-self-complementary molecules, $C_T$ is replaced by $C_T/4$. $\mathrm{Na}^+$ is the salt concentration, which is used for salt adjustment. In this paper, Santa Lucia Unified [14] is used in the calculation of $T_m$.

3. **The Proposed Ant Colony System for DNA Sequence.** Ant colony optimisation (ACO) is a population-based metaheuristics model that can be used to find solutions for optimisation problems [15]. In ACO, a set of software agents, called artificial ants, search for solutions to a given optimisation problem. Ant colony system (ACS) is a variant of ACO algorithms. ACS includes different mechanisms that are related to pheromone communication and the state transition rule.

The computation of the DNA sequence design problem is modelled by a 4-node state machine, as shown in Figure 1 [12]. The 4 nodes represent each of the four DNA bases: A, C, T, and G. Note that a similar computation model has been employed by Ibrahim et al. [8]. In this study, each solution consisted of seven ants, and each ant represented one DNA sequence. The length for each sequence was 20 nucleobases. Algorithm 1 shows the proposed ACS algorithm used in this study. The algorithm begins with the initialisation of $\tau_0$, $\varepsilon$, $\rho$, $H_{con}$, $H_{dis}$, $S_{con}$, and $S_{dis}$. The formulation of $\tau_0$ is shown in Equation (25).

$$\tau_0 = \frac{1}{Q/n} \tag{25}$$

where $Q$ is the sum of objectives calculated for a set of randomly generated DNA sequences and $n$ is the number of sequences. The initialised values of other parameters are shown in Table 2 and Table 3.

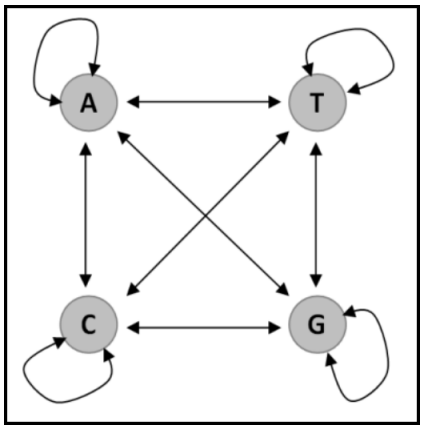


FIGURE 1. A computational model for DNA sequence design

Algorithm 1. Ant colony system for DNA sequence design

*//- - Initialisation*
Initialise parameters $\tau_0$, $\varepsilon$, $\rho$, $H_{con}$, $H_{dis}$, $S_{con}$, $S_{dis}$
Loop /* *each loop is called an iteration*
Loop /* *each loop is called an ant*
Each ant is positioned randomly on the start node
Loop
*//- - State transition rule*
Each ant applies state transition rule to incrementally build a solution
*//- - Local pheromone updating*
Local pheromone updating rules is applied
Until ant build a complete DNA sequence
If $GC_{content}$ and $T_m$ constraints passed then
Proceed with the next ant
Else
Repeat the DNA sequence generation using the current ant
End if
Until all ants have built a complete solution
For each DNA sequences do
Calculate objective functions
Next
If the objective functions of the sequences better than previous then store the sequences as the best found sequences
End if
*//- - Global pheromone updating for the best DNA sequence produced by ants*
A global pheromone updating rule is applied
Until Stopping Condition Meet

During the construction of a new set of DNA sequences, a state transition rule, as shown in Equation (26), was employed to determine the next state (base) for each ant.

$$p_k(r,s)_{\text{ACS}} = \begin{cases} \arg\max\{[\tau(r,S)]\} & \text{if } q \le q_0 \\ \text{random} & \text{otherwise} \end{cases} \tag{26}$$

where $\tau$ is the pheromone information, $s \in S$, and $S = \{\text{A, C, G, T}\}$. The probability, $p_k$, of an ant moving from one state, $r$, to another state, $s$, depends on random variables

TABLE 2. DNA parameters

| Parameter | | Value |
|---|---|---|
| $H_{measure}$ | $H_{con}$ | 6 |
| | $H_{dis}$ | 0.17 |
| *Similarity* | $S_{con}$ | 6 |
| | $S_{dis}$ | 0.17 |
| *Continuity threshold* | $t$ | 2 |
| *Hairpin* | $R_{\min}$ | 6 |
| | $P_{\min}$ | 6 |
| GC% | Min | 15 |
| | Max | 85 |
| $T_m$ | Min | 0 °C |
| | Max | 150 °C |
| $Na^+$ | | 1 Mol |

TABLE 3. ACS parameters

| Parameter | Value |
|---|---|
| $B$ | 1 |
| $Z$ | 0.05 |
| $P$ | 0.1 |
| $q_0$ | 0.5 |
| $N$ | half of ants |
| Number of Sequences = 7 (*no. of ants* − $n$) | |
| Length of DNA Sequences = 20 (*no. of tours*) | |
| Maximum Number of Iteration ($t_{\max}$) = 300 | |

$q$ and $q_0$, where $q$ is a uniformly distributed random variable $[0, 1]$ and $q_0$ is between 0 and 1. In this study, if $q > q_0$, random bases are chosen, as shown in Equation (26). This is due to the observation that if the actual equation for the state transition rule in ACS is used, it will increase the tendency to choose the same base repeatedly.

The sequence generated by ants must satisfy the $GC_{content}$ and melting temperature constraints. For any invalid sequence generated by the ant, the algorithm will require the ant to generate a new solution by random walk. This simple change in the ACS algorithm not only ensures the result obtained is valid but also reduces computation time. An invalid DNA sequence will affect other DNA sequences generated and thus requires additional computation time to generate a new solution. Then, each sequence constructed by ants is subjected to local pheromone updating, shown in Equation (27).

$$\tau(r,s)_{t+1} = (1-\zeta)\cdot\tau(r,s)_t + \tau_0 \tag{27}$$

where $\zeta = [0, 1]$ is the pheromone decay coefficient and $\tau_0$ is the initial value of the pheromone. After, the total fitness is calculated according to Equation (1). The ant with the lowest total fitness performs the global updating, according to Equation (28).

$$\tau(r,s)_{t+1} = (1-\rho)\cdot\tau(r,s)_t + \Delta\tau(r,s) \tag{28}$$

where $\rho$ $(0 \geq \rho \geq 1)$ is the evaporation rate and $\Delta\tau(r,s)$ is the quantity of pheromone laid on edge $(r,s)$ by ant $k$ at iteration $t$, which is given by:

$$\Delta\tau(r,s) = \begin{cases} 1/Q & \text{if } (r,s) \in \text{tour done by ant } k \\ 0 & \text{otherwise} \end{cases} \tag{29}$$

This process is repeated until the maximum number of iterations is reached.

4. **Results and Discussion.** The experiments were conducted for 100 runs, and the experimental results were collected for further analysis. In this experiment, the value of weight $\omega_i$ for each $f_i$ is equal to one and the length of each sequence is fixed to 20 nucleobases. The experimental results obtained from the proposed method were compared with the existing methods, namely ACS with archive [8], Genetic Algorithm (GA) [6], and Multi-Objective Evolutionary Algorithm (MOEA) [7]. Tables 4-7 show the best set of sequences based on the proposed method, ACS with archive [8], GA [6], and MOEA [7], respectively. Figure 2 shows the best results for all methods and the average value of the proposed ACS for 100 runs. In this paper, the melting temperature formulation was calculated based on the Nearest Neighbour (NN) method, with 1 M salt concentration and 10 $n$M DNA concentrations. Table 8 shows the average value for each objective functions for 100 runs.

Smaller values for each objective function as well as the total fitness showed that the proposed ACS provided better results than GA [10] and MOEA [7]. However, the existing ACS with archive [5] performed better than the proposed approach. This is due to the fact that the archive stored the best sequences during the optimisation process.

In terms of computation time, as shown in Table 9, a run averages 15.64s. Additionally, the average iteration number with global convergence for 100 runs is 175 iterations. A large average iteration number with global convergence might suggest that the algorithm has no convergence, while a small value might suggest that the algorithm convergence prematurely. Here, the average iteration with global convergence is 2/3 of the maximum

Table 4. Results obtained based on the proposed ACS

| Sequences | $H_{measure}$ | *Similarity* | *Continuity* | *Hairpin* |
|---|---|---|---|---|
| GGAGTGAGAGAGAGGAAGAG | 25 | 73 | 0 | 0 |
| AGAGAGAATGAGTTCAGATG | 43 | 67 | 0 | 0 |
| CGAGGAGATCCGCGATACCG | 53 | 60 | 0 | 0 |
| AGATGAGGAGCGCAGAGGCG | 39 | 69 | 0 | 0 |
| AGAGCGATGAGAAGAGAGAT | 28 | 72 | 0 | 0 |
| TGAGAGAGAGATGAGAGAGT | 23 | 74 | 0 | 0 |
| AAGAGAAGAGAGAGAGAGAG | 22 | 71 | 0 | 0 |
| Average | 33.285 | 69.428 | 0 | 0 |
| Total | 102.714 | | | |

Table 5. Results obtained based on ACS with archive [8]

| Sequences | $H_{measure}$ | *Similarity* | *Continuity* | *Hairpin* |
|---|---|---|---|---|
| ACGTGTGTCGTGTGTGTGTC | 23 | 77 | 0 | 0 |
| CTCTCCTCTCCTCTCCTCTC | 49 | 35 | 0 | 0 |
| TGTGTGTGTGTGTGTGCGTG | 21 | 68 | 0 | 0 |
| ACGTGTGTGTGTGGTGTGTG | 25 | 74 | 0 | 0 |
| GTTGTGTGTGTCTGTGGTGT | 20 | 81 | 0 | 0 |
| TGATTGATGATGATGATGAT | 39 | 46 | 0 | 0 |
| TTGTGTGTGTGGTCGTGTGT | 21 | 69 | 0 | 0 |
| Average | 28.285 | 64.285 | 0 | 0 |
| Total | 92.571 | | | |

TABLE 6. Results obtained based on GA [6]

| Sequences | $H_{measure}$ | *Similarity* | *Continuity* | *Hairpin* |
|---|---|---|---|---|
| ATAGAGTGGATAGTTCTGGG | 66 | 55 | 9 | 0 |
| CATTGGCGGCGCGTAGGCTT | 62 | 44 | 0 | 0 |
| CTTGTGACCGCTTCTGGGGA | 70 | 60 | 16 | 0 |
| GAAAAAGGACCAAAAGAGAG | 69 | 40 | 41 | 0 |
| GATGGTGCTTAGAGAAGTGG | 61 | 51 | 0 | 0 |
| TGTATCTCGTTTTAACATCC | 74 | 41 | 16 | 4 |
| TTGTAAGCCTACTGCGTGAC | 64 | 47 | 0 | 0 |
| Average | 66.571 | 48.285 | 11.714 | 0.57143 |
| Total | 127.1428571 | | | |

TABLE 7. Results obtained based on MOEA [7]

| Sequences | $H_{measure}$ | *Similarity* | *Continuity* | *Hairpin* |
|---|---|---|---|---|
| CTCTTCATCCACCTCTTCTC | 52 | 60 | 0 | 0 |
| CTCTCATCTCTCCGTTCTTC | 48 | 58 | 0 | 0 |
| TATCCTGTGGTGTCCTTCCT | 56 | 54 | 0 | 0 |
| ATTCTGTTCCGTTGCGTGTC | 59 | 56 | 0 | 0 |
| TCTCTTACGTTGGTTGGCTG | 64 | 51 | 0 | 0 |
| GTATTCCAAGCGTCCGTGTT | 66 | 47 | 0 | 0 |
| AAACCTCCACCAACACACCA | 71 | 41 | 9 | 0 |
| Average | 59.428 | 52.428 | 1.285 | 0 |
| Total | 113.142 | | | |

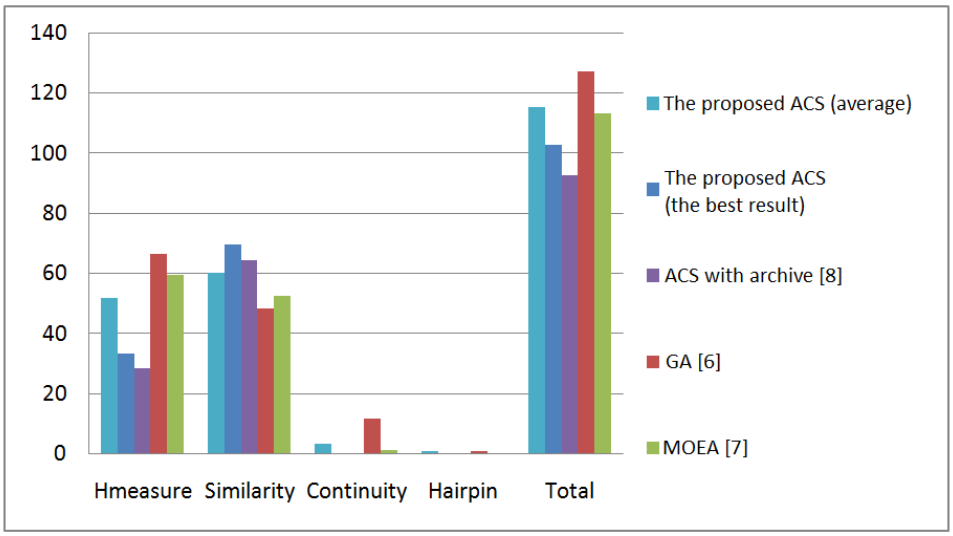


FIGURE 2. The best results for each method and the average value of the proposed ACS obtained based on 100 runs

TABLE 8. Average values for each objective functions for 100 runs

| Objective Functions | $H_{measure}$ | *Similarity* | *Continuity* | *Hairpin* | *Total* |
|---|---|---|---|---|---|
| Average (100 runs) | 51.89 | 60.1 | 3.25 | 0.65 | 115.23 |

iteration value, which is reasonable justification for the algorithm convergence performing properly.

Practical use of this work can be seen in several areas, such as DNA nanotechnology and DNA computing. DNA nanotechnology uses branched DNA structures to create DNA complexes with useful properties. One of the important steps in producing DNA structures is the sequence design, where poor design will produce undesired structures. In DNA computing, a well-designed DNA sequence will ensure the reliability of the DNA computation. Reliability of DNA computing is the ability of the single-stranded DNAs to hybridise correctly and produce identical results, *ceteris paribus*. This can be achieved by creating a large collection DNA sequence pairs that are unique.

Table 9. Computation time for 100 runs

| | |
|---|---|
| The average time taken for 300 iterations using 7 ants | 15.64 s |
| The least iteration number with global convergence | 65 iterations |
| The maximum iteration number with global convergence | 259 iterations |
| The average iteration number with global convergence | 175 iterations |

5. **Conclusions.** This paper presented an Ant Colony System (ACS) algorithm that was implemented to solve a DNA sequence design problem. This problem consists of four objectives ($H_{measure}$, *Similarity*, *Continuity*, and *Hairpin*) and two constraints ($\mathrm{GC}_{content}$ and *melting temperature*). The obtained results obtained were compared to other existing methods, including the ACS with archive, GA, and MOEA methods. It was expected that, even though the proposed approach was applied to solve a DNA sequence design problem, similar approaches can be used to design DNA sequences for any application, such as primer design for polymerase chain reaction and DNA-based nanotechnology. Finally, a multi-objective ant colony optimisation algorithm can be developed to effectively solve the DNA sequence design problem because the problem is a multi-objective optimisation problem.

**Acknowledgment.** The authors also would like to thank Universiti Teknologi Malaysia for supporting this research by UTM GUP research grant (Vote Q.J130000.7123.00H67).

## REFERENCES

[1] L. M. Adleman, Molecular computation of solutions to combinatorial problems, *Science*, vol.266, no.11, pp.1021-1024, 1994.

[2] Y. Tsuboi, Z. Ibrahim and O. Ono, DNA-based semantic memory with linear strands, *International Journal of Innovative Computing, Information and Control*, vol.1, no.4, pp.755-766, 2005.

[3] A. J. Hartemink, D. K. Gifford and J. Khodor, Automated constraint based nucleotide sequence selection for DNA computation, *Proc. of the 4th DIMACS Workshop DNA Based Computer*, pp.227-235, 1998.

[4] A. G. Frutos, A. J. Thiel, A. E. Condon, L. M. Smith and R. M. Corn, DNA computing at surfaces: Four base mismatch word design, *Proc. of the 3rd DIMACS Workshop DNA Based Computer*, pp.238, 1997.

[5] M. Arita and S. Kobayashi, DNA sequence design using templates, *New Generation Computer*, vol.20, pp.263-277, 2002.

[6] R. Deaton, J. Chen, H. Bi and J. A. Rose, A software tool for generating non-crosshybridization libraries of DNA oligonucleotides, *Proc. of the 8th International Workshop on DNA Based Computers*, pp.253-261, 2002.

[7] S. Y. Shin, I. H. Lee, D. Kim and B. T. Zhang, Multiobjective evolutionary optimization of DNA sequences for reliable DNA computing, *IEEE Transactions on Evolutionary Computation*, vol.9, no.2, pp.143-158, 2005.

[8] Z. Ibrahim, T. B. Kurniawan, N. K. Khalid, S. Sudin and M. Khalid, Implementation of ant colony system for DNA sequence optimization, *Proc. of the 14th International Symposium on Artificial Life and Robotics*, pp.293-296, 2009.

[9] J. Watada and R. A. Bakar, DNA computing and its application, *The 8th International Conference on Intelligent Systems Designs and Applications*, pp.288-294, 2008.

[10] R. B. A. Bakar and J. Watada, DNA computing and its applications: Survey, *ICIC Express Letters*, vol.2, no.1, pp.101-108, 2008.

[11] H. Yan, X. Zhang, Z. Shen and N. C. Seeman, A robust DNA mechanical device controlled by hybridization topology, *Nature*, vol.415, pp.62-65, 2002.

[12] T. B. Kurniawan, N. K. Khalid, Z. Ibrahim, M. Khalid and M. Middendorf, An ant colony system for DNA sequence design based on thermodynamics, *Proc. of the 4th IASTED International Conference Advances in Computer Science and Technology*, pp.144-149, 2008.

[13] R. J. Reece, *Analysis of Genes and Genomes*, John & Wiley, West Sussex, 2004.

[14] J. Santalucia, Jr., A unified view of polymer, dumbbell, and oligonucleotide DNA nearest-neighbour thermodynamics, *Proc. of the National Academy Science of the United State of America*, vol.95, pp.1460-1465, 1998.

[15] M. Dorigo and T. Stutzle, *Ant Colony Optimization*, Massachusetts Institute of Technology Press, Massachusetts, 2004.