Received October 22, 2020, accepted November 11, 2020, date of publication November 19, 2020, date of current version December 7, 2020.

Digital Object Identifier 10.1109/ACCESS.2020.3039190

MSGM: A Markov Model Based Similarity Guide Matrix for Optimising Ordered Problems by Balanced-Evolution Genetic Algorithms

RYOMA J. OHIRA[1], (Student Member, IEEE), MD. SAIFUL ISLAM[1], (Member, IEEE), HUMAYUN KAYESH[1], (Student Member, IEEE), AND S. M. RIAZUL ISLAM[2], (Member, IEEE)
[1]School of Information and Communication Technology, Griffith University, Southport, QLD 4222, Australia
[2]Department of Computer Science and Engineering, Sejong University, Seoul 05006, South Korea

Corresponding authors: Ryoma J. Ohira (r.ohira@griffith.edu.au) and Md. Saiful Islam (saiful.islam@griffith.edu.au)

ABSTRACT Where traditional genetic algorithms tend to prematurely converge on local optima, adaptive strategies aim to maintain a healthy level of population diversity by introducing randomness to the population. Often times this is done through adjusting control parameters according to diversity measurements. While these approaches introduce diversity, they do not aid in focusing or directing the search effort. Meanwhile, other works in the literature propose creating individuals designed to improve the population's health and quality but their effectiveness is limited outside of general problems. This article proposes novel sequence-wise approach to designing and editing genotypes for ordered problems. A Markov model based similarity guide matrix (MSGM) is used to determine the relationships between gene nodes in order to produce new genotypes that focus on improving fitness and increasing population diversity. The proposed MSGM based approach is implemented in a balanced-evolution genetic algorithm framework in order to investigate its characteristics with encouraging results demonstrating its effectiveness when solving combinatorial ordered optimisation problems.

INDEX TERMS Adaptive optimisation, balanced-evolution genetic algorithms, Markov model, ordered problems.

I. INTRODUCTION

In the field of biotechnology, genetic engineering is used to directly manipulate an organism's genes. This can be done through isolating and copying genetic material or by artificially synthesising the DNA itself. As genetic algorithms (GAs) are inspired by natural evolution, many studies have investigated concepts borrowed from biotechnology to improve its various operators and strategies. One research problem that stands to gain from this is the ability to direct a GA's search in a direction to improve its diversity, solution quality, or both. While a number of studies propose the creation of individuals from known qualities, these studies are largely limited to general optimisation problems [1]. Many of these approaches are incompatible or inappropriate for optimising ordered problems due to their constraints and their sequential nature of the encoded solution [2]–[4].

Combinatorial optimisation problems are often generalised forms of problems frequently encountered in the fields of operations and manufacturing. The process of solving these problems involves finding an optimal solution in a countably infinite set of possible solutions. Problems such as the *Travelling Salesman Problem* (TSP) and the *Capacitated Vehicle Routing Problem* (CVRP) are generalisations of problems in operations and logistics where the order in which the salesman visits the cities or a vehicle makes deliveries directly impact the cost of the operation [5]. These ordered problems are also often found in operations [6] and robotics [7]. Given a combinatorial ordered optimisation problem with N nodes, there are $N!$ solutions, thus finding the sequences and the relationship between each node in the different combinations is an *NP-hard problem*. Studies into the fitness landscape of problems like the TSP highlight the difficulties in solving these problems [8]–[10].

Given the difficulty of finding the optimal solution, heuristic approaches are often used to find high quality solutions within a reasonable time-frame. Innovations to adaptive GAs have enabled GAs to introduce a sense of intelligence in order to improve the efficiency and effectiveness of the search process. Continual improvements to these techniques

The associate editor coordinating the review of this manuscript and approving it for publication was Radu-Emil Precup.

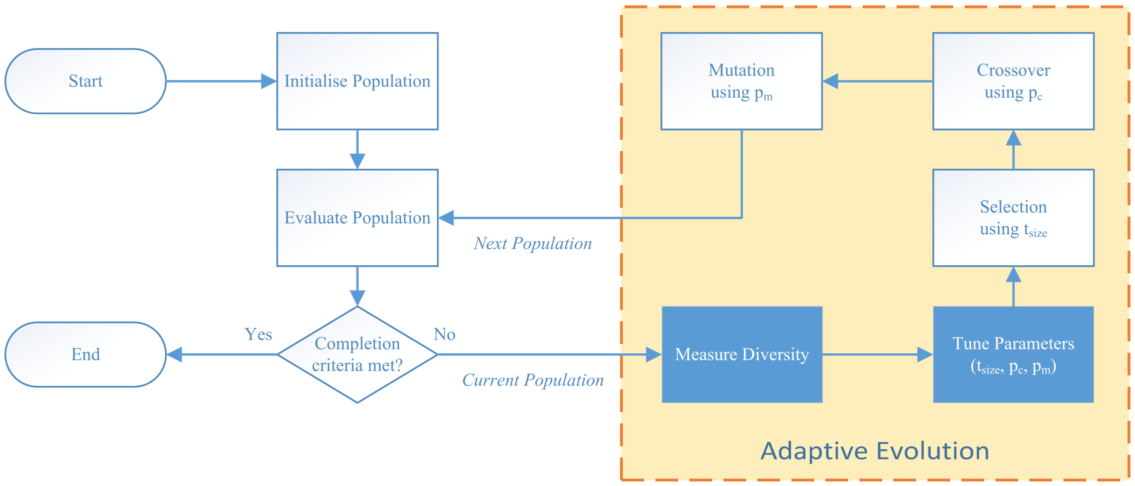


FIGURE 1. Generalised framework for adaptive genetic algorithms, where population is a set of solutions for a problem which evolves through crossover and mutation. Common adaptive strategies include parameter tuning for the tournament size (t_{size}), crossover probability (p_c) and mutation probability (p_m).

enable GAs to further improve their abilities in finding optimal solutions to NP-hard ordered problems. Originally theorised as being a hill-valley landscape, recent studies into the fitness landscape of combinatorial optimisation problems have identified clusters or funnels in the fitness landscape [11], [12]. These clusters of local optima can be found throughout the fitness landscape with the difficulty of finding the global optima being related to the number of clusters in the fitness landscape and the size of the cluster that the global optima resides in [13], [14]. As understanding of the fitness landscape improves, more advanced techniques to direct the search of GAs are introduced. While several frameworks introduce techniques inspired by genetic engineering to help direct an adaptive GA's search pattern, the constraints and the characteristics of ordered problems have not been taken into consideration. In particular, the relationships between nodes, how these relationships contribute to both fitness and diversity, and the sequential nature of the problems themselves. With existing works demonstrating how a Markov model can effectively establish the relationships between gene nodes [15], [16], the following research questions can be raised:

RQ1 How can we use Markov chains to introduce genome editing to direct the search?

RQ2 What balance between fitness and diversity is needed to maintain a healthy level of diversity?

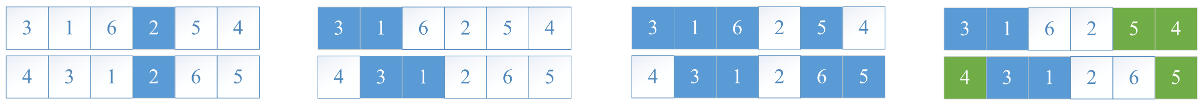
With the success of driving genotype editing through the *similarity guide matrix* (SGM) in existing works [1], RQ1 aims to investigate the appropriateness of a Markov model in comparison to traditional approaches, such as measuring the Hamming distance. By measuring the likelihood of a node proceeding another node, a genotype can be generated according to the likelihood of its sequence rather than the likelihood of gene values appearing at each gene position. However, when considering directing the GA's search and population diversity, a purely diversity focused approach may not be the most appropriate. As the GA should aim to search for the clusters of local optima, RQ2 aims to investigate different strategies to discover and investigate the multiple funnels of the fitness landscape.

In this article, we present a method for generating new genotype solutions that balance between introducing diversity and improving fitness using a *Markov model based similarity guide matrix* (MSGM). The proposed method is implemented into an existing balanced-evolution genetic algorithm (BEGA) framework to demonstrate its effectiveness for optimising ordered problems and compared against the original framework for a range of ordered problems from the TSP and CVRP benchmark instances.

The remainder of this article is organised as follows: Section II highlights the existing works in the literature for adaptive GAs with Section III discussing the balanced-evolution genetic algorithm (BEGA) framework and the limitations of current methods. Section IV describes our proposed approach to adapting the *similarity guide matrix* (SGM) to generate new genotypes for the TSP and CVRP problems. A discussion and analysis of the results are included in Section V with concluding remarks in Section VI.

II. RELATED WORKS

Adaptive GAs aim to prevent premature convergence on sub-optimal solutions by introducing a degree of intelligence to the search process. These approaches often utilise feedback from population diversity measurements to adjust parameters in on online manner [17]. In doing so, adaptive GAs attempt to manage their focus between exploring the solution space and exploiting known solutions according to whether the population has converged too much or has too much diversity. A generalised approach to implementing an adaptive GA is shown in Fig. 1. Adaptive tuning for the tournament size (t_{size}) enables the GA to apply an appropriate amount of selective pressure while the crossover (p_c) and mutation (p_m) probabilities enable adaptive GAs to manage a balance between



(a) Hamming distance (b) Contiguous subsequence (c) Non-contiguous subsequence (d) Broken Pairs

FIGURE 2. Different approaches to measuring the genotypic similarity between a pair of genotypes.

local (exploitation) and global (exploration) search. The works in adaptive GAs can be classified as techniques and strategies for measuring the similarity between individuals, measuring the diversity of the population and mechanisms for controlling the amount of diversity [17]–[19].

A. GENOTYPIC SIMILARITY

Gene-wise measures generally consider the similarities and differences between genotypes according to the absolute positions of the genes. The two most common approaches are the Euclidean and Hamming distances [18], [19]. While several works have applied these to solving ordered problems [20], [21], they do not consider the relationship between genes when measuring population diversity. The limitations of these approaches are further highlighted by works implementing approaches that focus on these relationship between nodes as shown in Fig. 2a.

The broken pairs [22] approach (Fig. 2d) considers the relationships between each pair of neighbouring genes and measures the differences between two genotypes as the number of pairs that have been separated. Numerous works [23], [24] demonstrate the effectiveness and improvements that this approach introduces in comparison to a gene-wise approach. However, the considerations of the sequence-wise nature of the problem is limited to pairs of genes.

In order to consider the relationship between a wider range of genes, Nagata and Kobayashi [25] proposed the use of a Markov model to measure the similarity between genotypes in the population and demonstrates how increasing the scope of the relationship between genes can improve on a GA's ability to maintain diversity. This is further demonstrated with the variable-order Marokov model [15], [16]. Another method for measuring the sequence-wise diversity of a population is the use of the longest common subsequence length (LCS) distance [2]–[4]. Similar to the Hamming distance, it measures the number of gene nodes that share a common non-contiguous subsequence between two genotypes as shown in Fig. 2c. This is more effective than measuring the contiguous subsequence (Fig. 2b) as a non-contiguous measurement would also include contiguous subsequences [2].

While there are different approaches to measuring the similarities and differences between two genotypes, each one has different costs and benefits when applied to ordered problems. How they are applied to measuring population diversity greatly affects their contribution to an adaptive GA's overall performance.

B. POPULATION DIVERSITY MEASUREMENTS

While research into determining the similarities and differences between genotypes is an active and ongoing field, a wide range of strategies for applying these measurements to maintain population diversity exists in the literature [17]. Early works aimed to measure the Hamming best and worst performing solutions [26] in order to minimise the computational costs. However, it was limited in that it could not provide feedback on the state of the population.

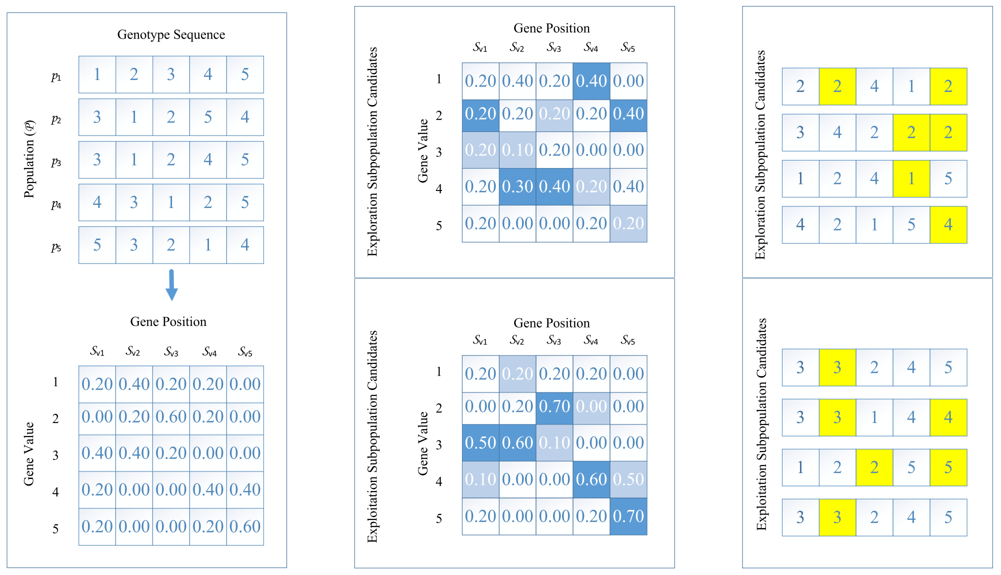
While hardware was a limiting factor to the practicality in measuring the diversity from all individuals in the population, Shimodaira [27] proposed a compromise by measuring the Hamming distance between the best, worst and a selection of the population. While this increased its scope, it was still limited by design with later works [28], [29] measuring the distance between all individuals. Another common strategy is to measure the distance between a reference individual and the population. These methods would compute a solution that represents a central point in the code space. While this often leads to solutions that are not valid within the solution space, they still demonstrate a utility in measuring diversity.

Mc Ginley *et al.* [30] uses a genotype consisting of average values at each position as a reference point to measure the Euclidean distance of the population. As this results in a non-integer value, approaches that measure the Hamming distances [1] or LCS distance [2]–[4] must come up with other methods such as the use of a mode or median genotype. Another method includes measuring the entropy rate of the population [15] that measured the probability of a sequence of nodes occurring in the population. With many strategies available for measuring population diversity, how diversity is introduced into the population is an active and on-going research area.

C. DIVERSITY CONTROL MECHANISMS

Once feedback on the amount of diversity has been received, an adaptive GA must then activate its mechanisms for introducing and controlling diversity. Early methods included re-initialising the population [31] to reboot the population as well as adaptive parameter controls [32]–[34].

Other approaches included adaptive population sizes [35], [36] and fixed populations with dynamic subpopulations [30]. The latter allowed for an adaptive GA to balance its focus between exploration and exploitation without increasing the computational effort. Where these works focused on general, unordered problems, their performance was limited when applied to solving within the constraints of ordered problems [2], [3].



Genotype Sequence
Population ($\mathcal{P}$)
Gene Position
Gene Value
Exploration Subpopulation Candidates
Exploitation Subpopulation Candidates

(a) Building the SGM from the population ($\mathcal{P}$)

(b) Computing the perturbations

(c) Generating candidate individuals

FIGURE 3. BEGA process for generating new candidate individuals from the SGM with duplicated genes in the candidate solutions being highlighted.

Zhang *et al.* [1] proposed a method for generating new individuals that either converged or diverged from the current population in their Balanced-Evolution Genetic Algorithm (BEGA) framework. Using the Hamming distance to measure the population's diversity allows for BEGA to determine how much the population needs to converge or diverge. Positive and negative perturbations of the *similarity guide matrix* (SGM) were used to generate individuals for exploitation and exploration subpopulations. This approach was novel in that it influenced the direction of the search area but was limited in its application due to its gene-wise approach to both measuring and maintaining diversity. While works have demonstrated how adaptive GAs can improve their performance by adapting to sequence-based approaches to diversity maintenance [2], [4], developing sequence-based approaches to directing the search using sequence-based genetic engineering approaches is an open research problem.

III. BEGA FRAMEWORK

The *balanced-evolution genetic algorithm* (BEGA) framework [1] was originally proposed for solving unordered problems such as the Knapsack problem and general mathematical problems. The framework balances the exploration and exploitation responsibilities of its search through two subpopulations. These subpopulations follow similar mechanics to one another but with one crucial difference. Unlike traditional GAs, the BEGA framework does not select two individuals for crossover but creates a temporary population of candidate individuals that are used to crossover with the real population. These candidate individuals are designed specifically to either encourage convergence or introduce diversity depending on which subpopulation they are designed for. This mechanism is controlled by the *similarity guide matrix* (SGM) and the *linear diversity index* (LDI).

Table 1 summarises the symbols used in this article.

TABLE 1. Symbols and their meaning.

Symbol	Meaning
CA	Control amplitude
D_l	Linear diversity measurement
D_p	Genotype distance
D_{sl}	Diversity shift limit
f	Fitness
M	Perturbation matrix
m_d	Minimum distance between individuals
m_s	Multiplier factor for the mutation operator
N	Genotype length
O	Child genotype
$\mathcal{P}$	Population of individuals
$\mathcal{P}_e$	Population of elite individuals
P_r	Transition probability
Q	Candidate genotype
S	Similarity guide matrix
S^M	Markov model based similarity guide matrix
S^F	Fitness based similarity guide matrix
S^B	Balanced similarity guide matrix
S_F	Fitness sum matrix
S_C	Transition count matrix
T	Threshold matrix
X	Genotype

A. SIMILARITY GUIDE MATRIX AND LINEAR DIVERSITY INDEX

The SGM creates a two dimensional matrix that expresses the probability distribution of the gene values at each gene position. This can be seen in Fig. 3a. In the case of an ordered problem such as the TSP, the matrix is a size of N^2 and can be expressed as Eqs. 1-2.

$$S = \{S_{v1}, \ldots, S_{vj}, \ldots, S_{vN}\} \quad 1 \le j \le N \tag{1}$$

$$S_{vj} = [s_{1j}, \ldots, s_{ij}, \ldots, s_{Nj}] \quad 1 \le i \le N \tag{2}$$

The two axis of the SGM (S) corresponds to the gene positions (Eq. 1) and gene values or nodes (Eq. 2). Each element in the matrix (s_{ij}) represents the proportion of individuals in the population that has the node value i in the jth position.

From the SGM, negative and positive perturbations are computed for the exploitation and exploration subpopulations. The differences in these perturbations are demonstrated in Fig. 3b. The *control amplitude* (CA) determines the degree of change in the perturbations. This is calculated using the LDI which is calculated as the average Hamming distance between a mode genotype and the rest of the population. This mode genotype can be computed from the SGM as the gene values at each position with the highest probability. The process for calculating the LDI is demonstrated in Eqs. 3-4.

$$D_l = \sum_{p=1}^{|\mathcal{P}|} \frac{D_p}{|\mathcal{P}|} \tag{3}$$

$$D_p = \frac{\text{Hamdis}(X_p, X_r)}{N} \tag{4}$$

The normalised diversity of each individual (D_p) is calculated as the Hamming distance between individuals in the population (X_p) and the mode geneotype reference point (X_r) normalised against the genotype length (N). The LDI (D_l) is then calculated as the average, normalised Hamming distance of the population ($\mathcal{P}$). The D_l is used in conjunction with the diversity shift limit (D_{sl}) to determine what stage of the evolution BEGA is in and the level of diversity that to be injected into the population. During the first stage where $D_l > D_{sl}$, D_l is used as the CA value as shown in Eq. 5.

$$CA = \begin{cases} D_l & D_l > D_{sl} \\ D_{sl} & D_{sl} \ge D_l \end{cases} \tag{5}$$

However, as BEGA converges on a optima, the population's diversity (D_l) can become a very small value which prevents the GA from being able to effectively maintain diversity. D_{sl} is designed to provide a minimum degree of diversity maintenance even in a maximally converged population with the authors recommending a value of 0.075.

B. COMPUTATION OF PERTURBATIONS

With both S and D_l, the perturbations for the exploration and exploitation subpopulations can be computed. The process for computing the negative perturbation is demonstrated in Algorithm 1. After the control amplitude (C_A) is calculated, the new perturbation matrix (M) must be populated with new values. For each vector (j) in S, the vector has a chance to be inherited directly from S without any changes. This is dependent on the state of the population's search process.

Algorithm 1 Computing Negative Perturbation

```
Input: $S$: SGM, $N$: Genotype length, $CA$: Control
       amplitude
Output: $M$: Perturbation matrix
Initialization: $M[N][N]$;
foreach $j \in 1, \ldots, N$ do
    if $rand() < CA$ then
        $k \leftarrow max(S_{vj})$ ;    // Index of maximum vector
        $l \leftarrow rand(1, N)$ ;     // Random index number
        while $l = k$ do
            $l \leftarrow rand(1, N)$ ;            // $l \neq k$
        $r \leftarrow CA \times rand() \times S_{kj}$ ; // Perturbation value
        $M[k][j] \leftarrow S[k][j] - r$;
        $M[l][j] \leftarrow S[l][j] + r$;
        foreach $i \in 1, \ldots, N$ do
            if $i \neq k$ & $i \neq j$ then
                $M[i][j] \leftarrow S[i][j]$;
    else
        $M[j] \leftarrow S[j]$;
return $M$;
```

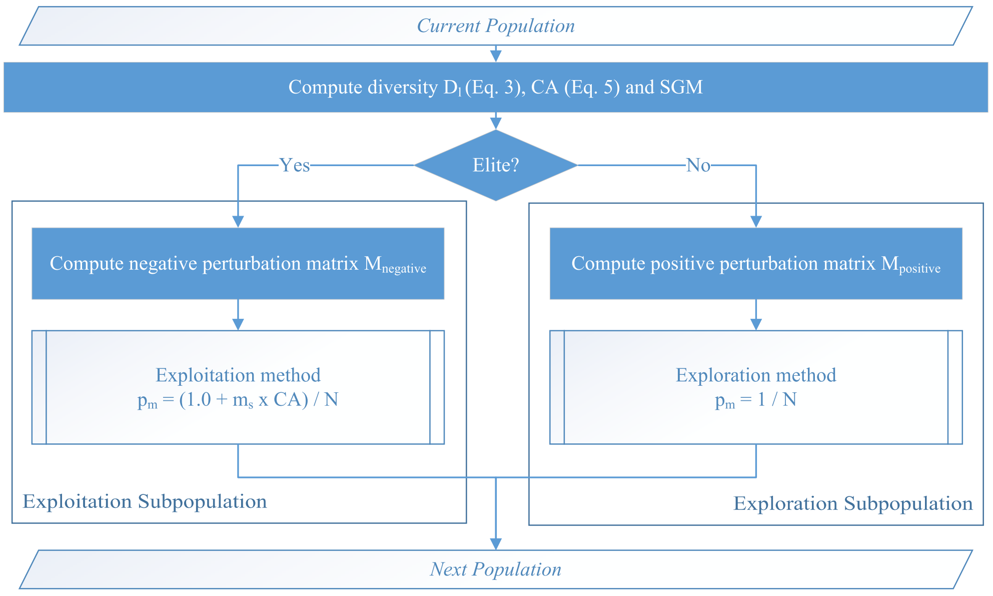
If vector j is selected to undergo changes in its perturbation, then the index (k) of the maximum value (s_{kj}) of the vector S_{vj} is selected. A randomly selected index (l) from S_{vj} is selected as a pair for the the perturbation where $k \neq l$. The negative perturbation of S reduces the maximum S_{vj} value by r and increases the value of the randomly selected member by the same amount with all other members of S_{vj} are directly inherited by M. This spreads the probability distribution for values at each gene position to decrease the likelihood of a common genotype being generated.

The positive perturbation for exploitation works similarly but with k being the index of the minimum value of the vector S_{vj}. Once the positive and negative perturbations of S are computed, candidate individuals can be generated according to the probability of values at each gene position (Fig. 3c).

C. EXPLORATION AND EXPLOITATION

The first step in generating new individuals is to produce a set of candidate individuals for both the exploration and exploitation subpopulations. Candidates for both methods are generated from their respective perturbation matrix and is demonstrated in Alg. 2. M is the SGM with negative or positive perturbations applied. T is the computed threshold matrix that allows for a uniformly random number (rand()) between 0 and 1 to select node i for the jth position in the candidate genotype Q.

The goal of the exploration subpopulation is to inject diversity into the population in order to explore new areas of the fitness landscape. Creating new genotypes for the exploration subpopulation is outlined in Alg. 3. The SGM with negative perturbations (M) is used to generate a new



Current Population
Compute diversity D_I (Eq. 3), CA (Eq. 5) and SGM
Elite?
Yes
No
Compute negative perturbation matrix $M_{negative}$
Compute positive perturbation matrix $M_{positive}$
Exploitation method
$p_m = (1.0 + m_s \times CA) / N$
Exploration method
$p_m = 1 / N$
Exploitation Subpopulation
Exploration Subpopulation
Next Population

FIGURE 4. The BEGA framework for balancing exploration and exploitation searches of the fitness landscape. This component replaces the highlighted "Adaptive Evolution" component of adaptive genetic algorithms illustrated in Fig. 1.

Algorithm 2 Generating New Individuals Based on SGM

```
Input: $M$: Perturbation matrix, $N$: Genotype length
Output: $Q$: Candidate genotype
Initialization: $Q[N]$, $T[N][N]$: Threshold matrix
foreach $i \in 1, \ldots, N$ do
    foreach $j \in 1, \ldots, N$ do
        $T[i][j] \leftarrow \begin{cases} M[i][j] & i = 0 \\ T[i-1][j] + M[i][j] & i \neq 0 \end{cases}$
foreach $i \in 1, \ldots, N$ do
    foreach $j \in 1, \ldots, N$ do
        $Q[j] \leftarrow \begin{cases} 0 & \text{rand}() \leq T[0][j] \\ i & T[i-1][j] < \text{rand}() \leq T[i][j] \end{cases}$
return $Q$;
```

Algorithm 3 Exploration Method

```
Input: $M$: Perturbation matrix, $X$: Reference genotype,
       $CA$: Control amplitude, $m_s$: Mutation multiplier
Output: $O$: New genotype
Initialization: $O[N]$: New genotype
$Q \leftarrow$ NewGenotype($M$);            // Candidate genotype (Alg. 2)
$O \leftarrow$ Crossover($Q, X$);
$p_m \leftarrow (1.0 + m_s \times CA)/N$;    // Mutation probability
$O \leftarrow$ Mutate($O, p_m$);
return $O$;
```

candidate genotype Q. For crossover, the original BEGA framework uses uniform crossover between the candidate genotype and the reference genotype. Uniform mutation is applied to the new genotype with a probability $p_m = (1.0 + m_s \times CA)/N$. The exploitation method is similar to the exploration method except the positive perturbation matrix is used to generate Q and constant mutation probability is used ($p_m = 1/N$).

D. FRAMEWORK: PUTTING IT ALL TOGETHER

The BEGA framework manages two sub-populations in order to balance between exploration and exploitation and is demonstrated in Fig. 4. A sub-population of elite individuals ($\mathcal{P}_e$) is selected with a minimum Hamming distance (m_d) between each individual as the exploitation sub-population. Individuals that are not selected for the exploitation sub-population become part of the exploration sub-population. The positive and negative perturbations of the SGM are computed (Alg. 1) which is then used in the exploration and exploitation methods (Alg. 3).

E. LIMITATIONS

As BEGA relies on a gene-wise approaches for the SGM, LDI, the positive and negative perturbations, and generating the candidate individuals, there are significant limitations when applied to ordered problems such as the TSP. This can immediately be seen in the candidate individuals themselves where the perturbation matrices can result in solutions that do not adhere to the problems' constraints.

While duplicate gene values can be replaced with missing gene values after the individuals are created, this can reduce the effectiveness of the diversity balance mechanisms introduced by BEGA [2]. Another limitation is its focus on absolute gene positions rather than the relationships between gene values and their positions in the sequence. This limits the ability of both the SGM and LDI to express the diversity to a position-by-position approach. Without being able to accurately measure and monitor the state of the population in its evolutionary process, BEGA is unable to effectively determine when diversity is needed or by how much. These two limitations results in BEGA having difficulty both monitoring and maintaining population diversity in an effective manner when solving ordered problems.

IV. OUR APPROACH

To generate effective candidate genotypes for ordered problems, the following constraints must be maintained:

C1 Each node in the problem must appear in the sequence and cannot be repeated, and

C2 A route can start at any node but must return to its original node or depot for the TSP and CVRP respectively.

C1 requires genotypes to be generated where there are N unique nodes in the sequence where the values range from $[1, \ldots, N]$, inclusively. The similarity guide matrix used by the BEGA framework can result in invalid genotypes where the duplicated nodes must be replaced with missing nodes retrospectively. A strategy for producing genotypes for these ordered problems must prevent duplicated or missing nodes in order to produce meaningful sequence-based genotypes. C2 describes the cyclical nature of ordered problems where the sequence of nodes is more important than their absolute positions. Thus, a method for designing and producing new genotypes must consider the ordered nature of sequence-based problems.

To model the above constraints of the ordered problems in the similarity guide matrix (SGM), we propose three new approaches to compute the SGM based on the Markov model, called the *Markov model based Similarity Guide Matrix* (MSGM). The first method uses the Markov model's transition matrix as the MSGM. The second method measures the average fitness of the subsequence of genes and can be considered as the contribution of a subsequence of nodes to a genotype's fitness. The final method is a hybrid approach that considers both the probability of a subsequence occurring and its contribution to a genotype's fitness.

Where the original BEGA framework uses a gene-wise approach (Hamming distance) to measure diversity in Eq. 4, in our approach we use the LCS distance based genotypic similarity measurement explored in our previous works [2]–[4] to measure diversity as shown in Eq. 6.

$$D_p = \frac{\mathrm{LCS}(X_p, X_r)}{N} \tag{6}$$

A. MARKOV MODEL BASED SGM

The BEGA framework utilises the SGM to express the distribution of a population or subpopulation in the solution space. This is then used as a reference point in the coding space order to measure the density of the solutions around that point and to produce candidate individuals. However, as highlighted in Fig. 3, this gene-wise approach can result in individuals being produced that do not consider the constraints and characteristics of ordered problems.

To address the aforementioned issues, we propose the use of a Markov model's transition matrix as the Markov model based SGM (MSGM). Where the original SGM had axes of coding position and value, a sequence-based approach considers the matrix axes as nodes from and to. If we consider the likelihood of a node preceding another node in a solution as a Markov chain with one order, the relationship can be modelled as in Eq. 7.

$$\begin{aligned} &P_r(q_1, q_2, \ldots, q_N) \\ &\quad = P_r(q_1) \times P_r(q_2|q_1) \times \ldots \times P_r(q_N|q_{q-1}) \qquad (7) \\ &\quad = P_r(q_1) \times \prod_{i=2}^{N} P_r(q_i|a_{q-1}) \qquad (8) \end{aligned}$$

P_r denotes the probability of the nodes appearing in a given sequence. However, this does not accurately portray the relationships between solution sequences and the fitness landscape in ordered problems. As a population begins to converge on clusters of optima, each cluster will being to show characteristics in its genotype sequence that are shared with nearby solutions.

When computing the MSGM, we can calculate the probability of a node proceeding another node to build a sequence-wise similarity guide matrix. We can express the MSGM (S^M) as a two dimensional matrix shown in Eq. 9. However, each member of S^M_{vj} represents the probability of node i following node j as shown in Eq. 10.

$$S^M = \{S^M_{v1}, \ldots, S^M_{vj}, \ldots, S^M_{vN}\} \quad 1 < j < N \tag{9}$$

$$S^M_{vj} = [P_r(1, j), \ldots, P_r(i, j), \ldots, P_r(N, j)] \quad 1 < \tag{10}$$

Where the original SGM expresses on the absolute positions of each gene node, the MSGM expresses on the probability of transition between nodes. This allows the MSGM to describe the sequence of nodes rather than their positions. The sum for each S^M_{vj} will always be 1 in ordered problems like the TSP and CVRP as each node must appear once in the sequence. Furthermore, for these problems, sequences can wrap around the genotype and remain similar to one another. For example, a sequence of [1, 2, 3, 4] should be considered similar to [3, 4, 1, 2] where both have the same $P_r(4, 1)$ and $P_r(2, 3)$. Thus $S^M = \{s^M_{ij}\}$ where s^M_{ij} is defined in Eq. 11.

$$s^M_{ij} = P_r(q_t = S^M_j | q_{t-1} = S^M_i) \quad 1 \le i, j \le N, i \ne j \tag{11}$$

a_{ij} represents the probability that node i follows node j given the assumptions that the i and j are the values from N nodes and that it is a steady state system. Further more, as $s^M_{ij} \ge 0$ and the genotype cannot repeat nodes in its sequence, it must eliminate the probability of j following i if j already appears in the genotype sequence. This can result

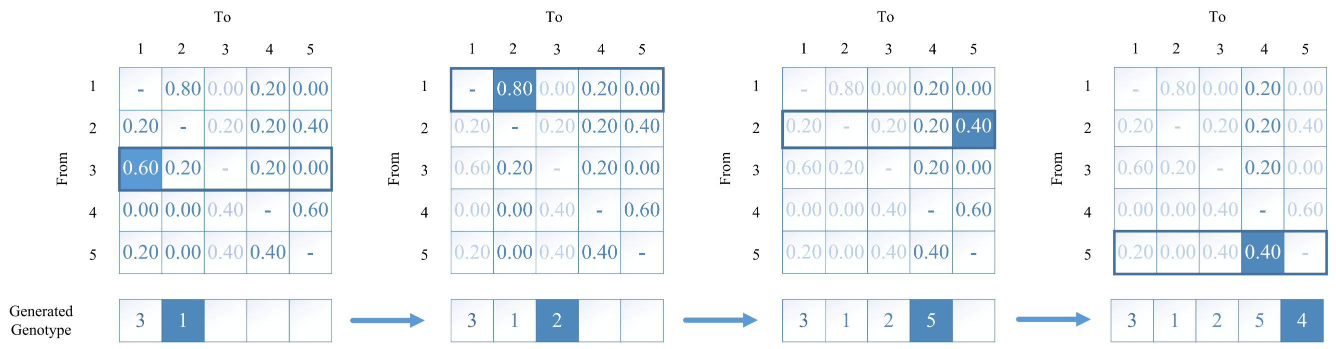
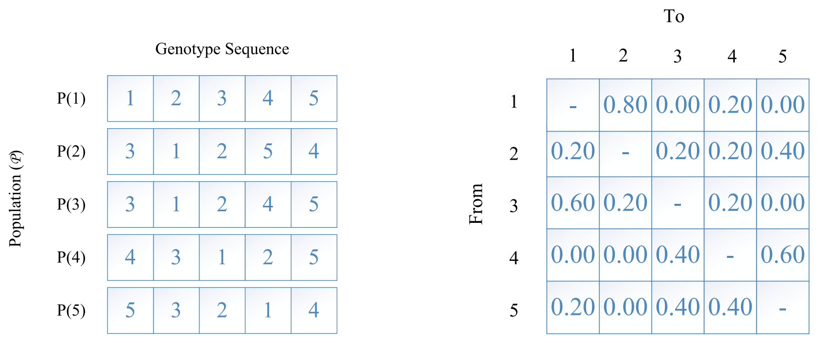


FIGURE 5. Process for creating a genotype from the MSGM involves eliminating the possibility of the sequence transitioning to a node that already exists in its sequence.


Genotype Sequence
Population ($\mathcal{P}$)
To
From

P(1)	1	2	3	4	5
P(2)	3	1	2	5	4
P(3)	3	1	2	4	5
P(4)	4	3	1	2	5
P(5)	5	3	2	1	4

(a) Population expressed as genotypes

	1	2	3	4	5
1	-	0.80	0.00	0.20	0.00
2	0.20	-	0.20	0.20	0.40
3	0.60	0.20	-	0.20	0.00
4	0.00	0.00	0.40	-	0.60
5	0.20	0.00	0.40	0.40	-

(b) Markov model based similarity guide matrix (MSGM) of the population

FIGURE 6. Expressing the transition probability of a sequence using a Markov model.

Algorithm 4 Computing MSGM

```
Input: $\mathcal{P}$: current population, $N$: genotype length
Output: $S^M$: Markov based SGM
Initialization: $S^M[N][N]$; $S_C[N][N]$;
foreach $x \in 1, \dots, |\mathcal{P}|$ do
    $X \leftarrow \mathcal{P}_x$ ;     // Genotype integer array
    foreach $k \in 2, \dots, N$ do
        $j \leftarrow X_{k-1}$ ;                  // From node
        $i \leftarrow X_k$ ;                       // To node
        $S_C[i][j] \leftarrow S_C[i][j] + 1$ ;    // Transition
        count matrix
foreach $i \in 1, \dots, N$ do
    foreach $j \in 1, \dots, N$ do
        $S^M[i][j] \leftarrow \frac{S_C[i][j]}{N}$;
return $S^M$
```

in cases where the only options available for j are cases where $s^M_{ij} = 0$. The sum of all the probabilities of nodes proceeding i must be $\sum_{j=1}^{N} s^M_{ij} = 1$.

S^M can be considered as a two dimensional matrix that expresses the distribution of the sequence of gene nodes. This can be seen in Fig. 6 where a first order Markov model can be used to calculate the subsequences of the population in Fig. 6a into the S^M (Fig. 6b). The vertical axis represents the current gene node while the horizontal axis represents the probabilities of the nodes that follow.

The process of creating a new genotype from the S^M is demonstrated in Fig. 5. An initial gene node can be selected according to its probability as a starting point with the proceeding node being selected according to its probability. Once a node has been added to the genotype, it is removed from the possible options in the vertical axis. This ensures that the constraints of the ordered problem are maintained. With the MSGM, the transition matrix can be used as the similarity guide and is shown in Algorithm 4.

For each individual in the population $\mathcal{P}$, the integer genotype sequence is expressed as vector X. S_C is the transition count matrix to store the number of genotypes in $\mathcal{P}$ where node i following node j. Thus, for each node in X, we increment the count at $S_C[i][j]$. Once the sum of transitions between all nodes for all individuals in P have been added to S_C, each value in the matrix is normalised by N to give a value ranging from 0 to 1. This is stored in the Markov Guide matrix (S^M) and returned as the transition matrix.

As the transition matrix describes the probabilities of one node following another, producing new genotypes is a stochastic process. As the population converges on a set of solutions, common subsequences in the population increases the probability of these subsequences appearing in genotypes produced by the MSGM. However, the stochastic nature of the Markov chains allows for the MSGM to introduce minor mutations to the overall sequence and can assist in the process for local search.

Algorithm 5 Computing Fitness Based MSGM

```
Input: $\mathcal{P}$: current population, $N$: genotype length
Output: $S^F$: Fitness based MSGM
Initialization: $S^F[N][N]$; $S_F[N][N]$; $f \leftarrow 0$
foreach $x \in 1, \ldots, |\mathcal{P}|$ do
    $X \leftarrow \mathcal{P}_x$ ;    // Genotype integer array
    $f \leftarrow f + \text{fitness}(X)$ ;    // Fitness sum
    foreach $k \in 2, \ldots, N$ do
        $j \leftarrow X_{k-1}$ ;    // From node
        $i \leftarrow X_k$ ;    // To node
        $S_F[i][j] \leftarrow S_F[i][j] + \text{fitness}(X)$; // Fitness sum matrix
foreach $i \in 1, \ldots, N$ do
    foreach $j \in 1, \ldots, N$ do
        $S^F[i][j] \leftarrow \frac{S_F[i][j]}{f}$;
return $S^F$
```

Algorithm 6 Computing Balanced MSGM

```
Input: $\mathcal{P}$: current population, $N$: genotype length
Output: $S^B$: Balanced MSGM
Initialization: $S^B[N][N]$; $S_F[N][N]$; $S_C[N][N]$;
foreach $x \in 1, \ldots, |\mathcal{P}|$ do
    $X \leftarrow \mathcal{P}_x$ ;    // Genotype integer array
    $f \leftarrow f + \text{fitness}(X)$ ;    // Fitness sum
    foreach $k \in 2, \ldots, N$ do
        $j \leftarrow X_{k-1}$ ;    // From node
        $i \leftarrow X_k$ ;    // To node
        $S_F[i][j] \leftarrow S_F[i][j] + \text{fitness}(X)$; // Fitness sum matrix
        $S_C[i][j] \leftarrow S_C[i][j] + 1$;    // Transition count matrix
foreach $i \in 1, \ldots, N$ do
    foreach $j \in 1, \ldots, N$ do
        $S^B[i][j] \leftarrow \frac{1}{2} \times (\frac{S_F[i][j]}{f} + \frac{S_C[i][j]}{N})$;
return $S^B$
```

B. FITNESS BASED MSGM

Where the MSGM approach considers areas of interests in the solution space as where the solutions are converging, it does not directly consider the fitness of the solutions. This is particularly important when taking into consideration the complexities in navigating a fitness landscape with many clusters of local optima.

The fitness-based MSGM computes the average fitness contribution of the gene sequence $[i, j]$ from the population which results in candidate genotypes more likely to inherit the fittest subsequences instead of the most common. This process is demonstrated in Algorithm 5 where S^F is a two dimensional matrix which contains the fitness distribution of node i following node j as opposed to the transition probabilities in S^M.

Similar to Algorithm 4, for each solution in $\mathcal{P}$, the genotype sequence is stored as X. As this is a fitness based transition matrix, the summation of the fitness is required later and is stored in f. For each node in the genotype X, the fitness of X is added to the matrix $S_F[i][j]$ where the sequence transitions from node j to i. Where the S^M in Algorithm 4 is the summation of transitions between nodes, S_F is the summation of the fitness of genotypes where a transition between nodes j and i exists. The fitness guide matrix (S^F) consists of the values of S_F normalised against the fitness sum of the individuals in $\mathcal{P}$ to give a value between 0 and 1 and illustrates the contribution of each pair of nodes to the average fitness.

C. BALANCED MSGM

A limitation of S^F is that it considers a lower fitness individual as being more diverse than an individual with a higher fitness and can be considered as being similar to a phenotypic approach to measuring diversity. Where as the limitation of S^M is its difficulty in conducting intensive search in multiple areas of interest. This highlights the necessity for a strategy that balances both converging towards good solutions and exploring new areas of the solution space. We propose a hybrid approach, called balanced MSGM that balances between genotypic diversity and fitness contribution. This allows BEGA to quickly converge on areas where known, highly fit solutions exists while also explore new areas of the solution space.

The balanced MSGM is given in Algorithm 6. Similar to Algorithms 4 and 5, the process iterates for all individuals x in $\mathcal{P}$ with the genotype sequence of x being expressed as X. For each node in the genotype, the transition from node j to i is added to the count matrix $S_C[i][j]$ while the fitness of the individual is added to the fitness sum matrix $S_F[i][j]$. Once S_F has accounted for all individuals in the population, the balanced guide matrix (S^B) is calculated as the average value of the normalised S^F and S^M values.

V. EXPERIMENTS

Here, we evaluate the effectiveness of our proposed Markov model based similarity guide matrix (MSGM) in optimising ordered problems by balanced-evolution genetic algorithms.

A. SETUP, ALGORITHMS AND PARAMETERS

This section describes the setup, algorithms, their parameters and evaluation metrics used in our experiments.

1) ENVIRONMENTAL SETUP

All algorithms are implemented using C#/.NET Core with experiments run on Windows 10 machines with AMD Ryzen 2600x CPU and 32GB of main memory. Each framework algorithm ran each instance until 20,000 generations with no improvements in the solution quality was reached. This was repeated for 20 runs. The C# implementations of the algorithms are made available online [37].

2) BENCHMARK ORDERED PROBLEMS AND OPERATORS

Instances of ordered problems were selected from the TSP benchmark library[1] according to their range in problem size. The new set of benchmark instances[2] proposed by Uchoa *et al.* [5] were selected for the CVRP tests. Due to the ordered nature of these problems, the modified ordered crossover operator and partially shuffled mutation operator were selected for their performance demonstrated in empirical tests [38].

3) ALGORITHMS AND THEIR PARAMETERS

The following algorithms have been implemented to conduct experiments in this article.

- **GA**: This is a standard genetic algorithm with no adaptive features.
- **BEGA**: The original BEGA framework (as described in Section III) is implemented in C# according to the authors' work [1]. This ensures for consistent comparisons between each implementation.
- **BEGA MSGM**: BEGA framework with LCS distance being used to calculate population diversity (D_p) as in Eq. 6. We also use MSGM (Alg. 4) instead of SGM.
- **BEGA MSGM Fitness**: BEGA framework with LCS distance being used to calculate population diversity (D_p) as in Eq. 6. We also use MSGM Fitness (Alg. 5) instead of SGM.
- **BEGA MSGM Balanced**: BEGA framework with LCS distance being used to calculate population diversity (D_p) as in Eq. 6. We also use the MSGM Balanced (Alg. 6) instead of SGM.

The BEGA framework parameter settings were kept consistent between each variant as recommended by Zhang *et al.* [1]. The original BEGA framework was implemented with its original operators. The GA, BEGA MSGM, BEGA MSGM Fitness and BEGA MSGM Balanced were implemented with the Modified Ordered Crossover (MOX) operator and the Partially Shuffled Mutation (PSM) operator due to their performance in ordered problems [38].

Population size ($|\mathcal{P}|$) for each algorithm is set to 90. The size of the elite sub-population $|\mathcal{P}_e| = 15$ with a minimum Hamming distance between individuals being $m_d = 3$ for the original BEGA. The BEGA MSGM variants require a minimum LCS distance $m_d = 3$, shift limit of diversity $D_{sl} = 0.075$. For BEGA and all of its variants, the multiplier factor for the mutation operator $m_s = 5$.

4) EVALUATION METRICS

To evaluate the algorithms, the best known solution (BKS) and problem size (N) has been included in the problem instance descriptions. To indicate the relative difference between each GA approach, the average error (Avg. Err) between the BKS and the solution found is reported with the coefficient of variation (C_v) demonstrating its consistency.

[1]http://www.math.uwaterloo.ca/tsp/data/index.html

[2]http://vrp.atd-lab.inf.puc-rio.br/index.php/en/

BKS Found is reported to indicate the number of runs in which the GA approach found the BKS (out of 20 runs). The average diversity (Avg. Diversity) is reported as the average of the diversity of the last generation for each run. The diversity of the final population is measured as the average LCS distance normalised against the length of the genotype.

B. PERFORMANCE EVALUATION

Tables 2-3 demonstrate the abilities of each of the GAs. The fitness is represented as the cost of the solutions where a lower value indicates a higher quality solution.

1) EFFECT OF PROBLEM SIZE

While the benchmark GA performs well on smaller problems, it can be seen to quickly converge on local optima as the problem size increases. The original BEGA implementation, while a significant improvement over the GA, also prematurely converges. However, applying a sequence-wise MSGM greatly improves the performance of BEGA in its ability to find better quality solutions for both smaller and larger problems. This improvement is also demonstrated when more constraints are introduced as the CVRP (Table 3). The fitness based MSGM (BEGA MSGM Fitness) appears perform worse than BEGA MSGM in both solution quality and consistency. While MSGM Fitness is effective at exploiting its found solutions, it can be seen to struggle with effectively searching the solution space for other peaks in the fitness landscape. However, by combining both of these approaches, the hybrid approach (MSGM Balanced) increases BEGA's ability to find better quality solutions and improve on its consistency in finding these solutions.

2) EXPLORATION VS EXPLOITATION

The three BEGA variants with sequence-wise approaches offer very significant improvements over the GA and many improvements over the original BEGA. However, the improvements of the MSGM Balanced approach over both the BEGA MSGM and BEGA MSGM Fitness offer more insight into the contributions of the two. The characteristics of each of the BEGA MSGM variants can be seen in Fig. 7.

While BEGA MSGM Fitness variant performs better in earlier generations, BEGA MSGM is capable of finding better solutions for more generations. This pattern is made clearer in the Figs. 7c-7d with the CVRP instance X-n10001-k43. With further constraints added by the CVRP, both BEGA MSGM Fitness and BEGA MSGM Balanced reach their completion criteria before BEGA MSGM. While BEGA MSGM can be seen to evolve for longer than BEGA MSGM Fitness, Tables 2-3 demonstrate that having the highest diversity does not directly result in the best solution. BEGA MSGM Balanced maintains a healthy level of diversity to evolve for longer and produces better solution quality.

TABLE 2. Performance for TSP instances highlighting the average error, coefficient of variation (C_V) in solution quality, the number of runs a GA framework found the best known solution, and the average diversity of the last population of each run.

Instance	Algorithm	Solution Quality			Avg. Diversity
		Avg. Err	C_v	BKS Found	
fnl4461	GA	0.12906	0.00930	0	0.061
BKS = 182,566	BEGA	0.08901	0.00820	0	0.026
N = 14,461	BEGA MSGM	0.00296	0.00140	18	**0.881**
	BEGA MSGM Fit.	0.00174	0.00310	15	0.635
	BEGA MSGM Bal.	**0.000000**	**0.00000**	**20**	0.793
fi10639	GA	0.12524	0.00120	0	0.045
BKS = 520,527	BEGA	0.11996	0.00107	0	0.109
N = 10,639	BEGA MSGM	0.00234	0.00025	9	**0.893**
	BEGA MSGM Fit.	0.00882	**0.00019**	6	0.632
	BEGA MSGM Bal.	**0.00070**	0.00030	**11**	0.874
usa13509	GA	0.03619	0.00091	0	0.042
BKS = 19,982,859	BEGA	0.07786	0.00072	0	0.081
N = 13,509	BEGA MSGM	0.00325	0.00033	9	**0.872**
	BEGA MSGM Fit.	0.00251	0.00040	2	0.534
	BEGA MSGM Bal.	**0.00062**	**0.00028**	**11**	0.846
xvb13584	GA	0.04948	0.00153	0	0.022
BKS = 37,083	BEGA	0.14440	0.00126	0	0.035
N = 13,584	BEGA MSGM	0.00253	0.00032	11	**0.863**
	BEGA MSGM Fit.	0.00313	0.00034	8	0.637
	BEGA MSGM Bal.	**0.00073**	**0.00030**	**14**	0.785
d15112	GA	0.18114	0.00135	0	0.018
BKS = 1,573,084	BEGA	0.14068	0.00123	0	0.062
N = 15,112	BEGA MSGM	0.00500	0.00010	0	0.819
	BEGA MSGM Fit.	0.01249	**0.00003**	0	0.593
	BEGA MSGM Bal.	**0.00140**	0.00017	**13**	**0.832**
it16862	GA	0.16727	0.00131	0	0.024
BKS = 557,315	BEGA	0.13543	0.00119	0	0.032
N = 16,862	BEGA MSGM	0.00660	0.00033	0	**0.900**
	BEGA MSGM Fit.	0.01115	**0.00031**	0	0.648
	BEGA MSGM Bal.	**0.00510**	0.00035	**2**	0.817
pjh17845	GA	0.18106	0.00125	0	0.032
BKS = 48,092	BEGA	0.13465	0.00119	0	0.034
N = 17,845	BEGA MSGM	0.00178	0.00025	3	0.797
	BEGA MSGM Fit.	0.01183	**0.00014**	0	0.635
	BEGA MSGM Bal.	**0.00110**	0.00037	**6**	**0.808**
d18512	GA	0.11345	0.00136	0	0.024
BKS = 645,238	BEGA	0.20134	0.00111	0	0.042
N = 18,512	BEGA MSGM	0.00910	0.00029	6	0.832
	BEGA MSGM Fit.	0.00167	0.00033	2	0.595
	BEGA MSGM Bal.	**0.00100**	**0.00025**	**11**	**0.839**
ido21215	GA	0.23140	0.00113	0	0.029
BKS = 63,517	BEGA	0.13257	0.00108	0	0.027
N = 21,215	BEGA MSGM	0.00117	0.00024	10	0.795
	BEGA MSGM Fit.	0.00112	0.00028	6	0.534
	BEGA MSGM Bal.	**0.00112**	**0.00020**	**11**	**0.816**
vm22775	GA	0.20397	0.00128	0	0.035
BKS = 569,288	BEGA	0.19367	0.00112	0	0.041
N = 22,775	BEGA MSGM	0.00125	0.00030	1	**0.857**
	BEGA MSGM Fit.	0.00130	0.00031	0	0.648
	BEGA MSGM Bal.	**0.00121**	**0.00028**	**4**	0.814
xrh24104	GA	0.19001	0.00138	0	0.021
BKS = 69,294	BEGA	0.16349	0.00114	0	0.033
N = 24,104	BEGA MSGM	**0.00159**	0.00022	12	**0.870**
	BEGA MSGM Fit.	0.01004	0.00028	0	0.612
	BEGA MSGM Bal.	0.00160	**0.00016**	**15**	0.802
sw24978	GA	0.21972	0.00134	0	0.026
BKS = 855,597	BEGA	0.11233	0.00120	0	0.031
N = 24,978	BEGA MSGM	0.00406	0.00032	4	**0.880**
	BEGA MSGM Fit.	0.00900	0.00028	0	0.601
	BEGA MSGM Bal.	**0.00055**	**0.00025**	**7**	0.822

C. DIVERSITY MAINTENANCE

In Tables 2-3, the average diversity highlights each GA framework's ability to maintain sequence-wise diversity. It is measured as the average LCS distance of the population at the completion of the run. As the value is normalised against the genotype length, a maximally diverse population approaches a value of 1 while 0 indicates maximal convergence. The capabilities for each GA framework in maintaining sequence-wise diversity is further highlighted in Fig. 8.

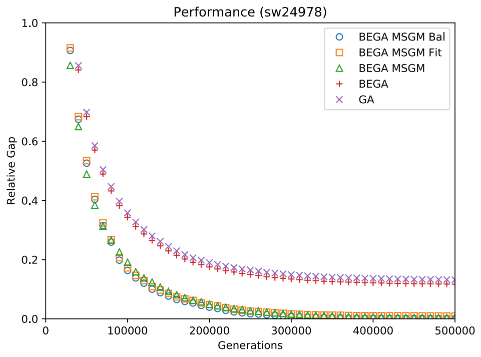
1) EFFECT OF PROBLEM SIZE AND CONSTRAINTS

In the TSP instances, the basic GA and original BEGA implementations can be seen to struggle to maintain a diverse population with a trend towards greater convergence with larger problem sizes. This suggests that while BEGA focuses on maintaining gene-wise diversity, almost half the sequences are common to all the solutions in the population.

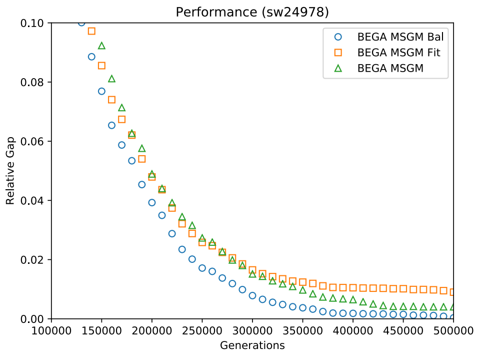
BEGA MSGM maintains a higher level of diversity regardless of the problem size. The BEGA framework

TABLE 3. Performance of Uchoa *et al.* [5] set from CVRP-Lib with instances grouped by *N* size highlighting the average error, average coefficient of variation (C_V), number of instances where the GA framework found the best known solution in its run and the average diversity of the final population for each run.

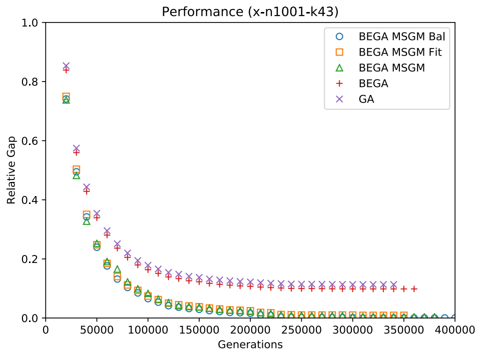
Instance	Algorithm	Solution Quality			Avg. Diversity
		Avg. Err	Avg. C_v	BKS Found	
$100 \leq N \leq 249$ Instances: 32	GA	0.0890	0.00030	1	0.061
	BEGA	0.0174	0.00031	3	0.080
	BEGA MSGM	0.0001	0.000020	30	**0.722**
	BEGA MSGM Fit	0.0101	0.00024	17	0.613
	BEGA MSGM Bal	**0.0000**	**0.00000**	**32**	0.686
$250 \leq N \leq 499$ Instances: 36	GA	0.1090	0.00030	1	0.012
	BEGA	0.0729	0.00041	1	0.030
	BEGA MSGM	0.0025	0.00002	17	**0.700**
	BEGA MSGM Fit	0.0031	0.00029	9	0.580
	BEGA MSGM Bal	**0.0011**	**0.00008**	**20**	0.665
$500 \leq N \leq 749$ Instances: 19	GA	0.1420	0.00037	0	0.056
	BEGA	0.0815	0.00048	0	0.035
	BEGA MSGM	0.0037	0.00025	9	0.682
	BEGA MSGM Fit	0.0053	0.00041	1	0.562
	BEGA MSGM Bal	**0.0016**	**0.00010**	**16**	**0.686**
$750 \leq N$ Instances: 13	GA	0.1346	0.00008	0	0.012
	BEGA	0.1183	0.00015	0	0.030
	BEGA MSGM	0.0049	0.00037	1	**0.678**
	BEGA MSGM Fit	0.0068	0.00038	0	0.513
	BEGA MSGM Bal	**0.0025**	**0.00014**	**9**	0.663



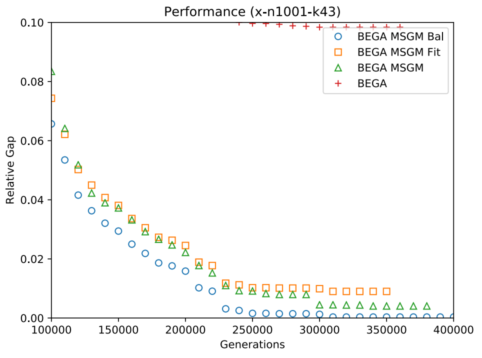
(a) TSP sw24978 relative gap between average solution and best known solution



(b) TSP sw24978 relative gap between average solution and best known solution where relative gap <= 0.10



(c) CVRP X-n0001-k43 relative gap between average solution and best known solution

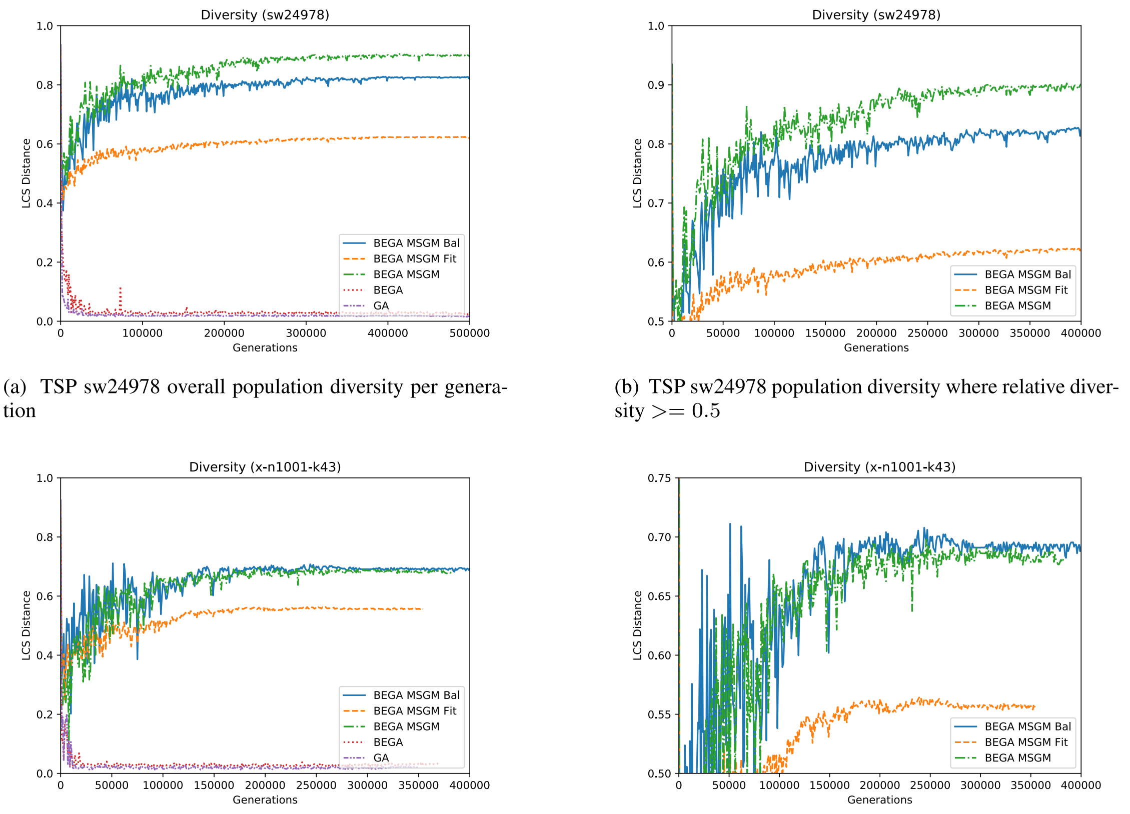


(d) CVRP X-n0001-k43 relative gap between average solution and best known solution where relative gap <= 0.10

FIGURE 7. The performance of each GA framework as demonstrated on the TSP sw24978 and CVRP X-n1001-k43 instances.

supports exploitation and exploration subpopulations where the exploitation subpopulation naturally converges on similar solutions. The higher levels of diversity in the BEGA MSGM Balanced variant suggests that while the exploration subpopulation is sufficiently diverse, the exploitation subpopulation also has a higher level of diversity allowing it to exploit multiple known peaks in the fitness landscape.

In Table 3, the effects of the additional constraints of the CVRP can be seen on the diminished population diversity. In particular, the ability to maintain population diversity is reduced in the BEGA MSGM Fitness variant as the problem size increases while both the BEGA MSGM and BEGA MSGM Balanced variant are able to maintain a more consistent level of population diversity.



(a) TSP sw24978 overall population diversity per generation

(b) TSP sw24978 population diversity where relative diversity $>= 0.5$

(c) CVRP X-n1001-k43 overall population diversity per generation

(d) CVRP X-n1001-k43 population diversity where relative diversity $>= 0.5$

FIGURE 8. The relative sequence-wise population diversity of each GA framework as demonstrated with a typical run of the TSP sw24978 and CVRP X-n1001-k43 instances.

2) EXPLORATION VS EXPLOITATION

In Fig. 8a both the basic GA and BEGA can be seen to converge very early in the search process. While the basic GA has no diversity maintenance processes, the original BEGA framework uses a gene-wise approach to measuring and maintaining diversity. This can result in a higher level of sequence-wise similarity in the population.

BEGA MSGM has the highest level of overall diversity due to its bias towards exploration as seen in Fig. 8b. As the framework aims to generate sequences that maximises the differences in sequences, this can naturally lead to an effective exploration process. The BEGA MSGM Fitness approach focuses on the exploitation of the fittest individuals. This is done through the fitness based guide matrix where the fitness of each sequence influences the likelihood of a subsequence appearing in a genotype. This results in the population converging on certain subsequences and lowering the overall sequence-wise diversity. Furthermore, the three BEGA MSGM variants can be seen to have a high level of fluctuation in population diversity in the early stages of their evolution. As the population begins to converge, the fluctuations in diversity significantly reduces. This can be seen as the second stage where the shift limit (D_{sl}) is used to maintain diversity as the LDI (D_l) is too small due to the level of convergence. What should be noted here is the degree of the fluctuations in diversity, particularly with the constraints of the CVRP in Figs. 8c-8d. BEGA MSGM and BEGA MSGM Balanced both display similar fluctuations between the peaks and valleys in diversity that suggests that diversity is introduced in a very acute and effective manner. However, the BEGA MSGM Fitness variant is not as effective.

While the BEGA MSGM Balanced variant has less diversity than the BEGA MSGM, it maintains diversity levels closer to that of the BEGA MSGM. This suggests that the mechanisms for managing the subpopulations are better equipped to maintain a high level of diversity for exploration but a smaller, more intense subpopulation for exploitation. When this is applied to solving ordered problems, BEGA MSGM Balanced is able to maintain a healthy level of diversity while also being able to effectively find good quality solutions. These characteristics are what likely enable the BEGA MSGM Balanced variant to outperform the other benchmark GAs.

D. DISCUSSION

The main objective of the BEGA framework is to balance between intensive local search (exploitation) and diverse global search (exploration). By a novel implementation of the similarity guide matrix, BEGA designs and produces genotypes to encourage convergence for the local search

TABLE 4. Statistical significance of solution quality of BEGA MSGM Balanced. "++" and "+" indicates a very significant improvement and significant improvement while "*" indicates no statistically significant difference.

Instance	BEGA MSGM-Balanced			
	GA	BEGA	BEGA MSGM	BEGA MSGM-F
fnl4461	++	++	++	++
fi10639	++	++	++	++
usa13509	++	++	++	++
xvb13584	++	++	++	++
d15112	++	++	*	++
it16862	++	++	++	++
pjh17845	++	++	++	++
d18512	++	++	++	++
ido21215	++	++	+	+
vm22775	++	++	+	+
xrh24104	++	++	*	*
sw24978	++	++	++	++
X-n101-k14 - X-n247-k50	++	++	++	++
X-n251-k28 - X-n491-k59	++	++	++	++
X-n502-k39 - X-n749-k98	++	++	++	++
X-n766-k71 - X-n1001-k43	++	++	++	++

subpopulation and introduces diversity to the global search to improve coverage of the fitness landscape.

In order to determine the statistical significance of any improvements demonstrated by the proposed methods, two-sample z-tests were conducted between a benchmark GA, the original BEGA implementation, as well as the MSGM, MSGM Fitness and MSGM Balanced implementations. The p values for the z-tests were 0.05 or less being considered a significant difference and 0.01 or less indicating a very significant difference. Table 4 highlights the statistical significance of the solutions found by BEGA MSGM Balanced.

In general, the MSGM approaches demonstrate a significant improvement in the BEGA framework's ability to find good quality solutions by adapting the gene-wise approaches to sequence-wise approaches. By implementing a Markov model to measure the probability of node transitions, BEGA is able to produce genotypes that better reflect the solution space and also reflect the ordered nature of the problems. By balancing between sequence-wise diversity and fitness contribution, our experiments demonstrate how our MSGM approaches are able to engineer genotypes for ordered problems and outperform the original BEGA framework.

VI. CONCLUSION AND FUTURE WORK

In this study we propose three methods for computing a Markov model based similarity guide matrix (MSGM) that takes into consideration the sequence of nodes for optimising ordered problems by the balanced-evolution genetic algorithms (BEGAs). We demonstrate how the MSGM can be used to genetically design and engineer genotypes to encourage convergence on good performing subsequences or encourage search in unexplored areas of the fitness landscape when implemented in an existing BEGA framework. Our experimental results highlights how a sequence based MSGM can introduce bias towards exploration while a fitness based MSGM biases towards exploitation. A balance between these two MSGM approaches demonstrated the ability to produce the best candidates for maintaining a healthy level of population diversity. Future research into improving the balance between exploration and exploitation for new genotypes should consider monitoring and maintaining this balance in an online manner.

REFERENCES

[1] H. Zhang, Y. Liu, and J. Zhou, "Balanced-evolution genetic algorithm for combinatorial optimization problems: The general outline and implementation of balanced-evolution strategy based on linear diversity index," *Natural Comput.*, vol. 17, no. 3, pp. 611–639, Sep. 2018.

[2] R. Ohira, M. S. Islam, J. Jo, and B. Stantic, "LCS based diversity maintenance in adaptive genetic algorithms," in *Proc. Australas. Data Mining Conf. (AusDM)*, 2018, pp. 56–68.

[3] R. Ohira and M. S. Islam, "A distributed genetic algorithm with adaptive diversity maintenance for ordered problems," in *Proc. 20th Int. Conf. Parallel Distrib. Comput., Appl. Technol. (PDCAT)*, Dec. 2019, pp. 308–313.

[4] R. Ohira and M. S. Islam, "GPU accelerated genetic algorithm with sequence-based clustering for ordered problems," in *Proc. IEEE Congr. Evol. Comput. (CEC)*, Jul. 2020, pp. 1–8.

[5] E. Uchoa, D. Pecin, A. Pessoa, M. Poggi, T. Vidal, and A. Subramanian, "New benchmark instances for the capacitated vehicle routing problem," *Eur. J. Oper. Res.*, vol. 257, no. 3, pp. 845–858, Mar. 2017.

[6] R.-E. Precup and R.-C. David, *Nature-Inspired Optimization Algorithms for Fuzzy Controlled Servo Systems*. Oxford, U.K.: Butterworth-Heinemann, 2019.

[7] C. Purcaru, R.-E. Precup, D. Iercan, L.-O. Fedorovici, R.-C. David, and F. Dragan, "Optimal robot path planning using gravitational search algorithm," *Int. J. Artif. Intell.*, vol. 10, no. 13, pp. 1–20, 2013.

[8] G. Ochoa and N. Veerapen, "Mapping the global structure of TSP fitness landscapes," *J. Heuristics*, vol. 24, no. 3, pp. 265–294, Jun. 2018.

[9] X. Chen, Y. Liu, X. Li, Z. Wang, S. Wang, and C. Gao, "A new evolutionary multiobjective model for traveling salesman problem," *IEEE Access*, vol. 7, pp. 66964–66979, 2019.

[10] C. Jiang, Z. Wan, and Z. Peng, "A new efficient hybrid algorithm for large scale multiple traveling salesman problems," *Expert Syst. Appl.*, vol. 139, Jan. 2020, Art. no. 112867.

[11] G. Ochoa and N. Veerapen, "Deconstructing the big valley search space hypothesis," in *Evolutionary Computation in Combinatorial Optimization*. Cham, Switzerland: Springer, 2016, pp. 58–73.

[12] G. Ochoa, N. Veerapen, D. Whitley, and E. K. Burke, "The multi-funnel structure of TSP fitness landscapes: A visual exploration," in *Proc. Evol. Artificielle*. Cham, Switzerland: Springer, 2015, pp. 1–13.

[13] S. Herrmann, G. Ochoa, and F. Rothlauf, "Communities of local optima as funnels in fitness landscapes," in *Proc. Genet. Evol. Comput. Conf. (GECCO)*, 2016, pp. 325–331.

[14] S. L. Thomson, F. Daolio, and G. Ochoa, "Comparing communities of optima with funnels in combinatorial fitness landscapes," in *Proc. Genet. Evol. Comput. Conf.*, Jul. 2017, pp. 377–384.

[15] Y. Nagata, "Population diversity measures based on variable-order Markov models for the traveling salesman problem," in *Parallel Problem Solving from Nature*. 2016, pp. 973–983.
[16] Y. Nagata, "High-order entropy-based population diversity measures in the traveling salesman problem," *Evol. Comput.*, pp. 1–25, Feb. 2020.
[17] A. Aleti and I. Moser, "A systematic literature review of adaptive parameter control methods for evolutionary algorithms," *ACM Comput. Surv.*, vol. 49, no. 3, pp. 1–35, Dec. 2016.
[18] M. Črepinšek, S.-H. Liu, and M. Mernik, "Exploration and exploitation in evolutionary algorithms: A survey," *ACM Comput. Surv.*, vol. 45, no. 3, p. 35, 2013.
[19] G. Karafotias, M. Hoogendoorn, and A. E. Eiben, "Parameter control in evolutionary algorithms: Trends and challenges," in *Proc. IEEE Congr. Evol. Comput. (CEC)*, Apr. 2015, vol. 19, no. 2, pp. 167–187.
[20] K. Q. Zhu, "A diversity-controlling adaptive genetic algorithm for the vehicle routing problem with time windows," in *Proc. IEEE Int. Conf. Tools With Artif. Intell.*, Nov. 2003, pp. 176–183.
[21] W. Lihong, T. Haikun, and Y. Guanghua, "A hybrid genetic algorithm for job-shop scheduling problem," in *Proc. CCECE*, 2015, pp. 271–274.
[22] V. Campos, M. Laguna, and R. Martí, "Context-independent scatter and tabu search for permutation problems," *INFORMS J. Comput.*, vol. 17, no. 1, pp. 111–122, Feb. 2005.
[23] C. Prins, "Two memetic algorithms for heterogeneous fleet vehicle routing problems," *Eng. Appl. Artif. Intell.*, vol. 22, no. 6, pp. 916–928, Sep. 2009.
[24] T. Vidal, T. G. Crainic, M. Gendreau, N. Lahrichi, and W. Rei, "A hybrid genetic algorithm for multidepot and periodic vehicle routing problems," *Oper. Res.*, vol. 60, no. 3, pp. 611–624, Jun. 2012.
[25] Y. Nagata and S. Kobayashi, "A powerful genetic algorithm using edge assembly crossover for the traveling salesman problem," *INFORMS J. Comput.*, vol. 25, no. 2, pp. 346–363, May 2013.
[26] D. Whitley and T. Starkweather, "GENITOR II: A distributed genetic algorithm," *J. Experim. Theor. Artif. Intell.*, vol. 2, no. 3, pp. 189–214, Jul. 1990.
[27] H. Shimodaira, "A diversity control oriented genetic algorithm (DCGA): Performance in function optimization," in *Proc. Genet. Evol. Comput. Conf. (GECCO)*, 2000, p. 366.
[28] M. Giger, D. Keller, and P. Ermanni, "AORCEA—An adaptive operator rate controlled evolutionary algorithm," *Comput. Struct.*, vol. 85, nos. 19–20, pp. 1547–1561, 2007.
[29] F. Lardeux, F. Saubion, and J.-K. Hao, "GASAT: A genetic local search algorithm for the satisfiability problem," *Evol. Comput.*, vol. 14, no. 2, pp. 223–253, Jun. 2006.
[30] B. McGinley, J. Maher, C. O'Riordan, and F. Morgan, "Maintaining healthy population diversity using adaptive crossover, mutation, and selection," *IEEE Trans. Evol. Comput.*, vol. 15, no. 5, pp. 692–714, Oct. 2011.
[31] L. J. Eshelman, "The CHC adaptive search algorithm: How to have safe search when engaging in nontraditional genetic recombination," in *Foundations of Genetic Algorithms*, vol. 1. 1991, pp. 265–283.
[32] T. Bäck, "Self-adaptation in genetic algorithms," in *Proc. Eur. Conf. Artif. Life (ECAL)*, 1992, pp. 263–271.
[33] Á. E. Eiben, R. Hinterding, and Z. Michalewicz, "Parameter control in evolutionary algorithms," *IEEE Trans. Evol. Comput.*, vol. 3, no. 2, pp. 124–141, Jul. 1999.
[34] T. Bäck, A. E. Eiben, and N. A. van der Vaart, "An emperical study on GAs 'without parameters,'" in *Proc. Int. Conf. Parallel Problem Solving from Nature*, 2000, pp. 315–324.
[35] F. G. Lobo and C. F. Lima, "Adaptive population sizing schemes in genetic algorithms," in *Parameter Setting in Evolutionary Algorithms*. 2007, pp. 185–204.
[36] G. J. LaPorte, J. Branke, and C.-H. Chen, "Adaptive parent population sizing in evolution strategies," *Evol. Comput.*, vol. 23, no. 3, pp. 397–420, Sep. 2015.
[37] R. Ohira and M. S. Islam. (Oct. 19, 2020). *BEGA-MSGM Source Code*. Accessed: Oct. 19, 2020. [Online]. Available: https://github.com/ryomaohira/bega-msgm/
[38] R. Ohira, M. S. Islam, J. Jo, and B. Stantic, "AMGA: An adaptive and modular genetic algorithm for the traveling salesman problem," in *Proc. Int. Conf. Intell. Syst. Design Appl. (ISDA)*, 2018, pp. 1096–1109.

RYOMA J. OHIRA (Student Member, IEEE) received the bachelor's degree (Hons.) in Information Technology from Griffith University, Australia, in 2017, where he is currently pursuing the Ph.D. degree with the School of Information and Communication Technology. His current research interests include online optimization, evolutionary computation, artificial intelligence, and deep learning.

MD. SAIFUL ISLAM (Member, IEEE) received the B.Sc. (Hons.) and M.S. degrees in computer science and engineering from the University of Dhaka, Bangladesh, in 2005 and 2007, respectively, and the Ph.D. degree in computer science and software engineering from the Swinburne University of Technology, Australia, in 2014. He is currently a Lecturer with the School of Information and Communication Technology, Griffith University, Australia. His current research interests are in the areas of database usability, spatial data management, AI, and big data analytics.

HUMAYUN KAYESH (Student Member, IEEE) received the M.Sc. degree in advanced computer science and IT management from The University of Manchester in 2017. He is currently pursuing the Ph.D. degree with the School of Information and Communication Technology, Griffith University, Australia. His current research interests include natural language processing, causality, social media analytics, conversational AI, and deep learning.

S. M. RIAZUL ISLAM (Member, IEEE) was an Assistant Professor with the University of Dhaka, Bangladesh, from 2005 to 2014, where he was also a Lecturer with the Department of Electrical and Electronic Engineering. In 2014, he was a Chief Engineer with the Department of Solution Laboratory, Samsung Research and Development Institute, Bangladesh. From 2014 to 2017, he was a Postdoctoral Fellow with the Wireless Communications Research Centre, Inha University, South Korea. From 2016 to 2017, he was a Postdoctoral Fellow with the Memorial University, Canada. He has been an Assistant Professor with the Department of Computer Science and Engineering, Sejong University, South Korea, since 2017. He is currently a Distinguished Professor with the Chongqing College of Electronic Engineering, China. His research interests include wireless communications, the Internet of Things, and applied artificial intelligence.