Interactive Markov Models of Evolutionary Algorithms

Haiping Ma, Dan Simon, Minrui Fei, and Hongwei Mo

***Abstract*—Evolutionary algorithms are global optimization methods that have been used in many real-world applications. In this paper we introduce a Markov model for evolutionary algorithms that is based on interactions among individuals in the population. This interactive Markov model has the potential to provide tractable models for optimization problems of realistic size. We propose two simple evolutionary algorithms with population-proportion-based selection and a modified mutation operator. The selection operator whose probability is linearly proportional to the number of individuals at each point of the search space. The mutation operator randomly modifies an entire individual rather than a single decision variable. We exactly model these evolutionary algorithms with the new interactive Markov model. We present simulation results to confirm the interactive Markov model theory. The main contribution is the introduction of interactive Markov theory to model simple evolutionary algorithms. We note that many other evolutionary algorithms, both new and old, might be able to be modeled by this method.**

***Index Terms*—Evolutionary algorithm; Markov model; population-proportion-based selection; transition probability; interactive Markov model**

I. Introduction

EVOLUTIONARY is algorithms (EAs) have received much attention over the past few decades due to their ability as global optimization methods for real-world applications [1, 2]. Some popular EAs include the genetic algorithm (GA) [3], evolutionary programming (EP) [4], differential evolution (DE) [5, 6], evolution strategy (ES) [7], particle swarm optimization (PSO) [8, 9], and biogeography-based optimization (BBO) [10, 11]. Inspired by natural processes, EAs are search methods that are fundamentally different than traditional, analytic optimization techniques; EAs are based on the collective learning process of a population of candidate solutions to an optimization problem. In this paper we often use the shorthand term *individual* to refer to a candidate solution.

The population in an EA is usually randomly initialized, and each iteration (also called a generation) evolves toward better and better solutions by selection processes (which can be either random or deterministic), mutation, and recombination (which is omitted in some EAs). The environment delivers quality information about individuals (fitness values for maximization problems, and cost values for minimization problems). Individuals with high fitness are selected to reproduce more often than those with lower fitness. All individuals have a small mutation probability to allow the introduction of new information into the population.

Each EA works on the principles of different natural phenomena. For example, the GA is based on survival of the fittest, DE is based on vector differences of candidate solutions, ES uses self-adaptive mutation rates, PSO is based on the flocking behavior of birds, BBO is based on the migration behavior of species, and ACO is based on the behavior of ants seeking a path between their colony and a food source. All of these EAs have certain features in common, and probabilistically share information between candidate solutions to improve the solution fitness. This behavior makes them applicable to all kinds of optimization problems. EAs have been applied to many optimization problems and have proven effective for solving various kinds of problems, including unimodal, multimodal, deceptive, constrained, dynamic, noisy, and multi-objective problems [12].

Evolutionary Algorithm Models

Although EAs have shown good performance on various problems, it is still a challenge to understand the kinds of problems for which each EA is most effective, and why. The performance of EAs depends on the problem representation and the tuning parameters. For many problems, when a good representation is chosen and the tuning parameters are set to appropriate values, EAs can be very effective. When poor choices are made for the problem representation or the tuning parameters, an EA might perform no better than random search. If there is a mathematical model that can predict the improvement in fitness from one generation to the next, it could be used to find optimal values of the problem representation or the tuning parameters.

For example, consider a problem with very expensive fitness function evaluations. For some problems we may even need to

Manuscript received December 3, 2013. This material was supported in part by the U.S. National Science Foundation under Grant No. 0826124, the National Natural Science Foundation of China under Grant No. 61305078, 61074032 and the Shaoxing City Public Technology Applied Research Project under Grant No. 2013B70004.

Haiping Ma was with the Department of Electrical Engineering, Shaoxing University, Shaoxing, Zhejiang, 312000, China. He is now with the Shanghai Key Laboratory of Power Station Automation Technology, School of Mechatronic Engineering and Automation, Shanghai University, Shanghai, 200072, China (e-mail: Mahp@usx.edu.cn).

Dan Simon is with the Department of Electrical and Computer Engineering, Cleveland State University, Cleveland, Ohio, 44115, USA (e-mail: d.j.simon@csuohio.edu).

Minrui Fei is with the Shanghai Key Laboratory of Power Station Automation Technology, School of Mechatronic Engineering and Automation, Shanghai University, Shanghai, 200072, China (e-mail: mrfei@staff.shu.edu.cn).

Hongwei Mo is with the Department of Automation, Harbin Engineering University, Harbin, Heilongjiang, China (e-mail: mhonwei@sina.com).

perform long, tedious, expensive physical experiments to evaluate fitness. If we can find a model that reduces fitness function evaluations in an EA, we can use the model during the early generations to adjust the EA tuning parameters, or to find out which EAs will perform the best. A mathematical model of the EA could be useful to develop effective algorithmic modifications. More generally, an EA model could be useful to produce insights to how the algorithm behaves, and under what conditions it is likely to be effective.

There has been significant research in obtaining mathematical models of EAs. One of the earliest approaches was schema theory, which analyzes the growth and decay over time of various bit combinations in discrete EAs [3]. It has several disadvantages, including the fact that it is only an approximate model. Perhaps the most developed EA model is based on Markov theory [13-15], which has been a valuable theoretical tool that has been applied to several EAs including genetic algorithms [16] and simulated annealing [17]. Infinite population size Markov models are discussed in detail in [18], and exact finite population size Markov models are discussed in [19].

In Markov models of EAs, a Markov state represents an EA population distribution. Each state describes how many individuals there are at each point of the search space. The Markov model reveals the probability of arriving at any population given any starting population, in the limit as the generation count approaches infinity. But the size of Markov model increases drastically with the population size and search space cardinality. These computational requirements restrict the application of the Markov model to very small problems, as we discuss in more detail in Section II.

Overview of the Paper

The goals of this paper are twofold. First, we present a Markov model of EAs that is based on interactions among individuals. The standard EA Markov model applies to the population as a whole; that is, it does not explicitly model interactions among individuals. This is an extremely limiting assumption when modeling EAs, and it leads directly to intractable Markov model sizes, as we show in Section II. In interactive Markov models, we define the states as the possible values of each individual. This gives a separate Markov model for each individual in the population, but the separate models interact with each other. The transition probabilities for each Markov model are functions of the states of other Markov models, which is a natural and powerful extension of the standard noninteractive Markov model. This method can lead to a better understanding of EA behavior on problems with realistic size.

The second goal of this paper is to propose two simple EAs that use population-based selection probabilities, which we call population-proportion-based selection. We use a modified mutation operator in the EAs. The modified mutation operator randomly modifies an entire individual rather than modifying a single decision variable, as in standard EA mutation. We exactly model these EAs with the interactive Markov model. Finally, we confirm the interactive Markov model with simulation on a set of test problems.

Section II introduces the preliminary foundations of interactive Markov models, presents two new simple EAs, models them using an interactive Markov model, and uses interactive Markov model theory to analyze their convergence. Section III discusses the similarities and differences between our two new simple EAs and standard, well-established EAs. Section IV explores the performance of the proposed EAs using both Markov theory and simulations. Section V presents some concluding remarks and recommends some directions for future work.

Notation: The symbols and notations used in this paper are summarized in Table 1.

Table 1 – Symbols and Notation

A, B	Interactive Markov model matrices
K	Search space cardinality = number of states in interactive Markov model
m_i	Fraction of population that is equal to x_i
m	Vector of population fractions: $m = [m_i]$
n	Vector of number of individuals that are allowed to change, or *movers*
N_1	Number of optimal individuals that are allowed to change (Strategy B)
N	Population size
p_{ij}	Probability of transition from s_j to s_i
p_m	Mutation probability
P	Markov transition matrix: $P = [p_{ij}]$
rand(a, b)	Uniformly distributed random number between a and b
s_i	Markov model state i
S	Number of elites (Strategy B)
t	Generation number
T	Number of states in standard Markov Model
x_i	Search space point
y_k	EA individual k
Y	EA population
α	Replacement pool probability
β	Selection parameter
λ	Modification probability
μ	Selection probability
φ	Selection pressure
σ	Elitism vector, or *stayers* (Strategy B): $\sigma = [\sigma_k]$

II. Interactive Markov models

This section presents the foundation for interactive Markov models (Section *A*), presents two new EA selection strategies (Section *B*), discusses their convergence properties (Section *C*), and analyzes the interactive Markov model transition matrix of the two new EAs (Section *D*).

A. Fundamentals of Interactive Markov Models

A standard, noninteractive Markov model is a random process with a discrete set of possible states s_i ($i = 1, \cdots, K$), where K is the number of possible states, also called the cardinality of the state space. The probability that the system transitions from state s_j to s_i is given by the probability p_{ij}, which is called a transition probability. The $K \times K$ matrix $P = [p_{ij}]$ is called the transition matrix. In standard, noninteractive Markov models of EAs, p_{ij} is the probability that the EA population

transitions from the *j*th possible population distribution to the *i*th possible population distribution in one generation. In the standard Markov model, one highly restrictive assumption is that the transition probabilities apply to the entire population rather than to individuals. That is, individual behavior is not explicitly modeled; it is only the population as a whole that is modeled.

To illustrate this point, consider an EA search space with a cardinality of $K \geq 2$; that is, there are K points in the search space. We denote these points as $\{x_1, x_2, \ldots, x_K\}$. Suppose there are a large number of EA individuals distributed over these K points. Suppose $m(t) = \left[m_i(t) \right]$ is the *K*-element column vector containing the fraction of the population that is equal to each point at generation *t*; that is, $m_i(t)$ is the fraction of the population that is equal to x_i at generation *t*. Then the equation $m(t+1) = Pm(t)$ describes how the fractions of the population change from one generation to the next. However, this equation assumes that the transition probabilities $P = \left[p_{ij} \right]$ do not depend on the value of $m(t)$; that is, they do not depend on the population distribution. Interactive Markov models, as discussed in this paper, deal with cases where the elements of P are functions of $m(t)$, so transition probabilities are functions of the Markov states. In this case P is written as a function of $m(t)$; that is, $P = P\left[m(t) \right]$, and the Markov model becomes interactive:

$$m(t+1) = P\left[m(t) \right] m(t) . \tag{1}$$

A standard Markov model with constant P can model EAs if the states are properly defined, and in fact this is exactly how EA Markov models have been defined up to this point. Given *K* points in search space and *N* individuals in the population, there are $T = \binom{K+N-1}{N}$ possible states for the population as a whole, and we can define a (T,T) transition matrix which contains the transition probabilities from every population distribution to every other population distribution [14]. However, this idea is not useful in practice because there is not a tractable way to handle the proliferation of states. For instance, for a small problem with *K*=100 and *N*=20, we obtain *T* on the order of 10^{22}. Markov models of that order are clearly not tractable. For a slightly larger problem with *K*=200 and *N*=30, we obtain *T* on the order of 10^{37}.

In contrast, interactive Markov models in the form of (1) prove to be tractable for much larger search spaces and populations. For instance, if *K*=100, then the interactive Markov model consists of only 100 states, regardless of the population size. An interactive Markov model of this size is simple to handle. The challenge that we address in this paper is how to define the interactive Markov model for a given EA, and how to obtain theoretical results based on the interactive model.

The interactive Markov model is a fundamentally different modeling approach than the standard (noninteractive) Markov model. The states of the standard Markov model consist of population distributions, while the states of the interactive Markov model consist of fractions of each individual in the search space. This difference means that the standard Markov model transition matrix is independent of the EA population, while the interactive Markov model transition matrix is a function of the population distribution. The standard Markov model transition matrix is thus larger (*T* states) but with a simple form, while the interactive Markov model transition matrix is much smaller (*K* states) but with a more complicated form. Both the standard and the interactive Markov models are exact. However, due to their fundamentally different approaches to the definition of state, neither one is a subset of the other, and neither one can be derived from the other.

For standard Markov models, many theorems are available for stability and convergence. But for interactive Markov models, such results do not come easily, given the immense variety of the possible forms of $P(\cdot)$. The literature [20] discusses a particular class of $P(\cdot)$ functions that are defined as follows for a *K*-state system:

$$\begin{aligned} & p_{ij}(m) = \frac{a_{ij} + b_{ij} m_i}{\sum_k \left(a_{kj} + b_{kj} m_k \right)} \quad \text{for all } (i,\ j) \in [1, K] \\ & \text{where } a_{ij} \geq 0,\ b_{ij} \geq -a_{ij} \text{ for all } (i,\ j) \in [1, K] \\ & \qquad \min_m \sum_k \left(a_{kj} + b_{kj} m_k \right) > 0 \text{ for all } j \in [1, K] \end{aligned} \tag{2}$$

where the summations go from 1 to *K*, and $m(t) = \left[m_i(t) \right]$ is abbreviated to $m = [m_i]$. Given values for the matrices $A = \left[a_{ij} \right]$ and $B = \left[b_{ij} \right]$, we have a complete interactive Markov model.

The specification of the transition matrix $P(m)$ is complete from (2), and the evolution of $m(t)$ from any initial value $m(0)$ is determined by (1). In a standard Markov model, the probability $p_{ij}(m)$ of a transition from state *j* to state *i* would be constant; that is, $p_{ij}(m) = a_{ij}$. But in an interactive Markov model, $p_{ij}(m)$ depends on m. The transition probability $p_{ij}(m)$ is, in a sense, a measure of the attractive power of the *i*th state. In (2), if $b_{ij} > 0$ then a greater population in the *i*th state makes it more attractive, and if $b_{ij} < 0$ then crowding in the *i*th state makes it less attractive.

Since the columns of $P(m) = \left[p_{ij}(m) \right]$ must each sum to one, a normalization of $p_{ij}(m)$ is needed. The division in (2) of each column of the matrix $\left[a_{ij} + b_{ij} m_i \right]$ by the corresponding column sum $\sum_k \left(a_{kj} + b_{kj} m_k \right)$ provides the desired normalization.

B. Selection Strategies

In this subsection, we modify two previously-published models of social processes to obtain two simple EAs that use population-based selection. We will see that these EAs have interactive Markov models of the form of (2).

1) Strategy A

The first new EA that we introduce, which we call Strategy A, is based on a social process model in Example 1 in [20]. Strategy A involves the selection of random individuals from the search space, and the replacement of individuals in the population with the randomly-selected individuals. The basic form of Strategy A does not include recombination or mutation, although it could be modified to include these features. Strategy A includes three tuning parameters: α, λ, and β. The population of Strategy A evolves according to the following two rules.

(*a*) Denote $\alpha \in [0,1]$ as the *replacement pool probability*. We randomly choose $\text{round}(\alpha N)$ individuals, where N is the population size. Denote $\lambda_j \in [0,1]$ as the *modification probability*, which is similar to crossover probability in GAs. λ_j is typically chosen as a decreasing function of fitness; that is, good individuals should have a smaller probability of modification than poor individuals.

(*b*) The jth individual chosen above, where $j \in [1, \text{round}(\alpha N)]$, has a probability of λ_j of being replaced with one of K individuals from the search space (recall that K is the cardinality of the search space, and $\{x_i : i = 1, \cdots, K\}$ is the search space). The probability of selecting the ith individual x_i is denoted as μ_i and is composed of two parts: $\beta \in [0,1]$ is assigned to each individual equally; and the remaining probability $1-\beta$ is assigned among the x_i individuals in the proportions $(m_1, \cdots, m_K)$, where m_i is the proportion of the x_i individuals in the population. That is, the selection probability of x_i is $\mu_i = \beta/K + (1-\beta)m_i$. Note that $\sum_{i=1}^{K} \mu_i = 1$.

Note that if the selection probability is independent of m_i, that is, $\beta = 1$, the selection probability of each individual x_i is $\mu_i = 1/K$. According to the above two rules, an EA that operates according to Strategy A can be written as shown in Algorithm 1. We see from the algorithm listing that Strategy A has four tuning parameters: N (population size), α (replacement pool probability), β (the constant component of the selection probability), and λ (modification probability, which is a function of fitness).

ALGORITHM 1 – AN EVOLUTIONARY ALGORITHM BASED ON STRATEGY A.

Generate an initial population of individuals $Y = \{y_k : k = 1, \cdots, N\}$
While not (termination criterion)
  Randomly choose $\text{round}(\alpha N)$ parent individuals
  For each chosen parent individual y_j ($j = 1, \cdots, \text{round}(\alpha N)$)
    Use modification probability λ_j to probabilistically decide whether to modify y_j
    If modifying y_j then
      Select x_i ($i = 1, \cdots, K$) with probability [$\beta/K + (1-\beta)m_i$]
      $y_j \leftarrow x_i$
    End modification
  Next individual: $j \leftarrow j+1$
Next generation

2) Strategy B

The second new EA that we introduce, which we call Strategy B, is based on a social process model in Example 6 in [20]. Similar to Strategy A, Strategy B also involves the selection of random individuals from the search space, and the replacement of individuals in the population with the randomly-selected individuals. However, Strategy B also includes elites. Strategy B includes four tuning parameters: α, λ, and β, which are similar to the same quantities in Strategy A; and σ_1, which is an elitism parameter. The population of Strategy B evolves according to the following two rules.

(*a*) For each individual y_k in the population, if y_k is the best individual in the search space, there is a probability σ_1 that it will be classified as elite, where $\sigma_1 \in [0,1]$ is a user-defined tuning parameter.

(*b*) If y_k is not classified as elite, use λ_k to probabilistically decide whether to modify y_k. If y_k is selected for modification, it is replaced with x_i with probability proportional to $\alpha + \beta m_i$, where m_i is the fraction of the population that is comprised of x_i individuals. Recall that $\{x_i : i = 1, \cdots, K\}$ is the search space.

Similar to Strategy A, Strategy B does not include recombination or mutation, although it could be modified to include these features. According to the above two rules, an EA that operates according to Strategy B can be written as shown in Algorithm 2. We see from the algorithm listing that Strategy B has the same four tuning parameters as Strategy A: N (population size), α (replacement pool probability), β (selection constant), and λ (modification probability, which is a function of fitness). However, note that α and β are used differently in Strategies A and B.

ALGORITHM 2 – AN EVOLUTIONARY ALGORITHM BASED ON STRATEGY B.

Generate an initial population of individuals $Y = \{y_k : k = 1, \cdots, N\}$
While not (termination criterion)
  For each individual y_k
    If y_k is not the best individual in the search space then
      Use modification probability λ_k to decide whether to modify y_k
      If modifying y_k then
        Select x_i ($i = 1, \cdots, K$) with probability $[\alpha + \beta m_i]$
        $y_k \leftarrow x_i$
      End modification
    End if
  Next individual: $k \leftarrow k+1$
Next generation

In Strategy B we set the selection probability of x_i to

$$\mu_i \equiv \Pr(y_k \leftarrow x_i) = \alpha + \beta m_i \tag{3}$$

for $i \in [1, K]$, where the population of individuals is $\{y_k : k = 1, \cdots, N\}$, N is the population size, and $(\alpha, \beta) \in [0,1]$ are user-defined tuning parameters. Equation (3) gives the probability that y_k is replaced by x_i, and this probability is a linear function of m_i, which is the proportion of x_i individuals in the population. Note that (3) holds for all $k \in [1, N]$ (assuming that the given y_k is selected for replacement).

3) Selection Pressure in Strategy A and Strategy B

The selection probabilities in Strategy A and Strategy B are both linear with respect to the fraction of x_i individuals in the population. Figure 1 depicts the selection probability.

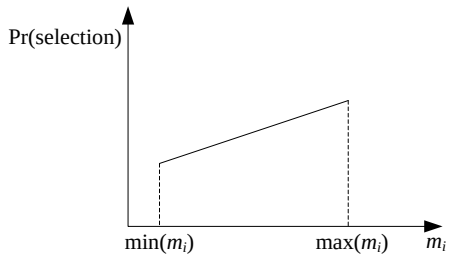

Figure 1 – Selection probability in Strategy A and Strategy B evolutionary algorithms. The *min* and *max* operators are taken over $i \in [1, K]$, where K is the cardinality of the search space.

In Figure 1, if the search space cardinality K is greater than the population size N, as in practical EA implementations, then $\min(m_i) = 0$. This is because there are not enough individuals in the population to completely cover the search space, so there are always some search space individuals x_i that are not represented in the population.

In Strategies A and B, if selection is overly-biased toward selecting populous individuals, the population may converge quickly to a uniformity while not widely exploring the search space. If selection is not biased strongly toward populous individuals, the population will be more widely scattered with a smaller representation of good individuals. Note that the most populous individuals will be the ones with highest fitness if we define modification probability λ_k as a decreasing function of fitness. We will see this effect later in our simulation results in Section IV.

A useful metric for quantifying the difference between various selection methods is selection pressure ϕ, which is defined as follows [3, p. 34]:

$$\phi = \frac{\max(\Pr(\text{selection}))}{\text{average}(\Pr(\text{selection}))} \tag{4}$$

where $\Pr(\text{selection})$ is the probability that an individual is selected as a replacement.

The most populous individual has the maximum fraction $\max(m_i)$, which we denote as $m_{\max}$. The average individual has the average fraction $(\max(m_i) + \min(m_i))/2 = m_{\max}/2$, assuming that $K > N$, as discussed earlier in this section. Then the selection pressure of (4), when applied to Strategy B (Algorithm 2), can be written as

$$\phi = \frac{\alpha + \beta m_{\max}}{\alpha + \beta m_{\max}/2} \tag{5}$$

If we normalize the selection probabilities so that they sum to 1, we get

$$\begin{aligned} \sum_{i=1}^{K} \Pr(y_k \leftarrow x_i) &= \sum_{i=1}^{K} (\alpha + \beta m_i) \\ &= K\alpha + \beta \sum_{i=1}^{K} m_i = K\alpha + \beta = 1 \end{aligned} \tag{6}$$

In the above equation we used the fact that $\sum_{i=1}^{K} m_i = 1$ because the sum of all the fractions of the x_i individuals in the population must equal 1. If we desire a given selection pressure ϕ, we can solve (5) and (6) for α and β to obtain the following:

$$\begin{aligned} \alpha &= \frac{m_{\max}(2-\phi)}{K m_{\max}(2-\phi) + 2(\phi-1)} \\ \beta &= \frac{2(\phi-1)}{K m_{\max}(2-\phi) + 2(\phi-1)} \end{aligned} \tag{7}$$

Note that (7) also holds for Strategy A (Algorithm 1) if α is replaced with β/K, and β is replaced with $(1-\beta)$.

C. Convergence

In this subsection, we present a theorem that summarizes the convergence conditions of interactive Markov models.

Theorem 1: Consider an interactive Markov model in the form of (1). If there exists a positive integer R such that $\prod_{t=1}^{R} P\left[m(t)\right]$ is positive definite for all $m(t) \geq 0$ such that $\sum_{i=1}^{K} m_i(t) = 1$, where $t = 1, 2, \cdots, R$, then $\prod_{t=1}^{R} P\left[m(t)\right]$ converges as $R \to \infty$ to a steady state which has all nonzero entries.

Proof: See [21] for a proof and discussion. ♦

Later in this section, we will use Theorem 1 to show that there is a unique limiting distribution for the states of the interactive Markov models in this paper. We will also show that the probability of each state of the interactive Markov model is nonzero at all generations after the first one. In particular, Theorem 1 will show that Algorithms 1 and 2 have a unique limiting distribution with nonzero probabilities for each point in the search space. This implies that Algorithms 1 and 2 will both eventually find the globally optimal solution to an optimization problem.

D. Interactive Markov Model Transition Matrices

The previous subsections presented two simple but new selection strategies, and showed that they both have a unique population distribution as the generation count approaches infinity. Now we analyze their interactive Markov models in more detail and find the solutions to the steady-state population distribution.

1) Interactive Markov Model for Strategy A

Selection:

We can use the development of Example 1 in [20] to obtain the following interactive Markov model of Strategy A:

$$p_{ij} = \begin{cases} (1-\alpha) + \alpha\left[\left(1-\lambda_j\right) + \lambda_j\left(\beta/K + (1-\beta)m_j\right)\right] & \text{if } i = j \\ \alpha\lambda_j\left(\beta/K + (1-\beta)m_i\right) & \text{if } i \neq j \end{cases} \quad (8)$$

for $(i, j) \in [1, K]$. The quantity p_{ij} gives the probability that a given individual in the population transitions from x_j to x_i in one generation. The first equality in (8), when $i = j$, denotes the probability that an individual does not change from one generation to the next. This probability is composed of three parts:

(a) the first term, $1-\alpha$, denotes the probability that the individual is not selected for the replacement pool;

(b) the first part of the second term is the product of the probability that the individual is selected for the replacement pool (α), and the probability that the individual is not selected for modification ($1-\lambda_j$);

(c) the second part of the second term is the product of the probability that the individual is selected for the replacement pool (α), the probability that the individual is selected for modification (λ_j), and the probability that the selected individual is replaced with itself ($\beta/K + (1-\beta)m_j$).

For the second equality in (8), when $i \neq j$, denotes the probability that an individual is changed from one generation to the next. This probability is very similar to the second part of the second term of the first equality as discussed in paragraph (c) above, the difference being that m_i is used instead of m_j; that is, the selected individual is changed from x_j to x_i.

Mutation:

Next we add mutation into the interactive Markov model of Strategy A. Typically, EA mutation is implemented by probabilistically complementing each bit in each individual. Then the probability that individual x_i mutates to become x_k can be written as

$$p_{ki} = \Pr\left(x_i \to x_k\right) = p_m^{H_{ik}}\left(1-p_m\right)^{q-H_{ik}} \quad (9)$$

where $p_m \in (0, 1)$ is the mutation rate, q is the number of bits in each individual, and H_{ij} is the Hamming distance between bit strings x_i and x_j.

But with this type of mutation, the transition matrix elements $p_{kj} = \sum_i p_{ki} p_{ij}$ would not satisfy the form of (2) and (8). So we use a modified mutation operator that randomly modifies an entire individual rather than a single decision variable of an individual. Mutation of the jth individual is implemented as follows.

```
For each individual y_j
    If rand(0, 1) < p_m
        y_j ← rand(L, U)
    End if
Next individual
```

In the above mutation logic, rand(a, b) is a uniformly distributed random number between a and b, and L and U are the lower and upper bounds of the search space. The above logic mutates each individual with a probability of p_m. If mutation occurs for a given individual, the individual is replaced with a random individual within the search domain. The descriptions of Strategy A and Strategy B with mutation are the same as Algorithm 1 and 2 except that we add the operation, "probabilistically decide whether to mutate each individual in the population" at the end of each generation. Note that mutation acts on all individuals $Y = \{y_k : k = 1, \cdots, N\}$. In particular, for Strategy B, we use elitism only to prevent selection, but not to prevent mutation.

Now, the transition probability of the modified mutation operator is described as

$$p_{ki} = \begin{cases} (1-p_m) + p_m/K & \text{if } k = i \\ p_m/K & \text{if } k \neq i \end{cases} \quad (10)$$

Then the transition probability of Strategy A with mutation, and the corresponding A and B matrices described in (2), can be written as follows:

$$p_{kj}=\sum_i p_{ki}p_{ij}$$
$$=\begin{cases}(1-\frac{(K-1)p_m}{K})(1-\alpha)+\alpha\left[(1-\frac{(K-1)p_m}{K})(1-\lambda_j)+\lambda_j\left((1-\frac{(K-1)p_m}{K})\frac{\beta}{K}+\frac{(K-1)p_m}{K^2}\beta+\frac{p_m}{K}(1-\beta)+(1-\beta)(1-p_m)m_j\right)\right] & \text{if } k=j\\ \frac{p_m}{K}(1-\alpha)+\alpha\left[\frac{p_m}{K}(1-\lambda_j)+\lambda_j\left((1-\frac{(K-1)p_m}{K})\frac{\beta}{K}+\frac{(K-1)p_m}{K^2}\beta+\frac{p_m}{K}(1-\beta)+(1-\beta)(1-p_m)m_k\right)\right] & \text{if } k\neq j\end{cases} \tag{11}$$

and

$$a_{kj}=\begin{cases}(1-\frac{(K-1)p_m}{K})(1-\alpha)+\alpha\left[(1-\frac{(K-1)p_m}{K})(1-\lambda_j)+\lambda_j\left((1-\frac{(K-1)p_m}{K})\frac{\beta}{K}+\frac{(K-1)p_m}{K^2}\beta+\frac{p_m}{K}(1-\beta)\right)\right] & \text{if } k=j\\ \frac{p_m}{K}(1-\alpha)+\alpha\left[\frac{p_m}{K}(1-\lambda_j)+\lambda_j\left((1-\frac{(K-1)p_m}{K})\frac{\beta}{K}+\frac{(K-1)p_m}{K^2}\beta+\frac{p_m}{K}(1-\beta)\right)\right] & \text{if } k\neq j\end{cases} \tag{12}$$
$$b_{kj}=\alpha\lambda_j(1-\beta)(1-p_m)$$

The derivation of (11) is in the appendix. Now we are in a position to state the main result of this subsection.

Theorem 2: The $K\times 1$ equilibrium population fraction vector m^* of Algorithm 1, which is exactly modeled by the interactive Markov model of (1) and (11), is equal to the dominant eigenvector (normalized so its elements sum to one) of the matrix $A_0=a_{ij}\Big/\left(b_j+\sum_{k=1}^K a_{kj}\right)$, where a_{ij} is given by (12), and b_j is shorthand notation for b_{ij} in (12) since all the rows of B are the same.

Proof: This theorem derives from Theorem 1 in [20]. We provide the proof in the appendix. ♦

Next we consider the special case $\mu_i=1/K$. In this case, the transition probability (11) is written as

$$p_{kj}=\begin{cases}(1-\frac{(K-1)p_m}{K})(1-\alpha)+\alpha\left[(1-\frac{(K-1)p_m}{K})(1-\lambda_j)+\frac{\lambda_j}{K}\right] & \text{if } k=j\\ \frac{p_m}{K}(1-\alpha)+\alpha\left[\frac{p_m}{K}(1-\lambda_j)+\frac{\lambda_j}{K}\right] & \text{if } k\neq j\end{cases} \tag{13}$$

which is independent of m_i; that is, the interactive Markov model reduces to a standard noninteractive Markov model. In this case we can use Theorem 1 in [20] or standard noninteractive Markov theory [13] to obtain the equilibrium population fraction vector m^*.

2) Interactive Markov Model for Strategy B

Before discussing the interactive Markov model of Strategy B, we define some notation. Suppose that $\sigma=[\sigma_k]$ denotes the fraction of individuals that always remain equal to x_k for each $k\in[1, K]$. Then $\sum_k\sigma_k$ is the fraction of all individuals that never change. We call these individuals *stayers*. $n=[n_1,\cdots,n_K]$ denotes the fractions of all individuals that are allowed to change, and we call these individuals *movers*. $m=[m_1,\cdots,m_K]$ denotes the fractions of all individuals in the population. The fraction of x_j individuals thus includes two parts: σ_j, which denotes the fraction of all x_j individuals that are not allowed to change; and $\left(1-\sum_{k=1}^K\sigma_k\right)n_j$, which denotes the fraction of all x_j individuals that may change in future generations. So the fraction vector m is given by

$$m=\sigma+\left(1-\sum_{k=1}^K\sigma_k\right)n \tag{14}$$

The σ vector is related to EA elitism since it defines the proportion of individuals in the population that are not allowed to change in subsequent generations. One common approach to elitism is to prevent only optimal individuals from changing in future generations [12]. That is, $\sigma_1>0$ (assuming, without loss of generality, that x_1 is the optimal point in the search space), and $\sigma_k=0$ for all $k>1$. In this case (14) becomes

$$m_i=\begin{cases}\sigma_1+(1-\sigma_1)n_1 & \text{for } i=1, \text{ which corresponds to the optimal state}\\ (1-\sigma_1)n_i & \text{for } i\neq 1, \text{ which correspond to nonoptimal states}\end{cases} \tag{15}$$

Note that Strategy B elitism is a little different than standard EA elitism. In Strategy B, we assume that we know if an individual is at the global optimum. This is not always the case in practice, but it may be the case for certain problems. If S individuals at the global optimum are retained as elites, then $\sigma_1=S/N$ (this quantity could change from one generation to the next). If N_1 individuals at the global optimum are allowed to change, then $n_1=N_1/(N-S)$. If an individual is not at the global optimum, then we must always allow it to change – that is, we can implement elitism only for individuals that are at the global optimum.

Example

To clarify the relationships between σ, n, and m, we present a simple example. Suppose we have a population of 14 individuals, and the search space size is 3; that is, $N = 14$ and $K = 3$. Then:

- 2 individuals in state 1 are stayers (they will never leave state 1);
- 3 individuals in state 1 are movers (they may transition out of state 1);
- 4 individuals are in state 2, and they are all movers; and
- 5 individuals are in state 3, and they are all movers.

Therefore:

- $n_1 = 3/12$ (fraction of movers that are in state 1);
- $\sigma_1 = 2/14$ (fraction of population that are stayers in state 1);
- $n_2 = 4/12$ (fraction of movers that are in state 2); and
- $n_3 = 5/12$ (fraction of movers that are in state 3).

We substitute these values in (15) and obtain $m_1 = 5/14$ (fraction of individuals that are in state 1), $m_2 = 4/14$ (fraction of individuals that are in state 2), and $m_3 = 5/14$ (fraction of individuals that are in state 3), which are the same as the distributions stated at the beginning of the example.

Summary of the Interactive Markov Model for Strategy B

Now we use Example 6 in [20], combined with (6) above, to obtain the following interactive Markov model of Strategy B selection:

$$p_{ij} = \begin{cases} (1-\lambda_1) + \lambda_1\left[\alpha + \beta\left(\sigma_1 + (1-\sigma_1)n_1\right)\right] & \text{if } i = j = 1, \text{ which is the best state} \\ \lambda_j\left[\alpha + \beta\left(\sigma_1 + (1-\sigma_1)n_1\right)\right] & \text{if } i = 1 \text{ and } j \neq 1 \\ (1-\lambda_j) + \lambda_j\left[\alpha + \beta(1-\sigma_1)n_i\right] & \text{if } i = j \neq 1 \\ \lambda_j\left[\alpha + \beta(1-\sigma_1)n_i\right] & \text{if } i \neq j, \text{ and } i \neq 1 \end{cases} \tag{16}$$

The above equation is explained as follows. For the first equality, when $i = j = 1$, which is the best state, the transition probability includes two parts: (a) the first term, which denotes the probability that the individuals is not changed $(1-\lambda_1)$; and (b) the second term, which denotes the probability that the individual is changed to itself, and which is the product of the probability that the individual is changed (λ_1), and the probability that the selected x_i term is equal to $\lambda_1\left(\mu_1 \Big/ \sum_{k=1}^{K}\mu_k\right)$. This second term can be written as

$$\begin{aligned} \lambda_1 \frac{\mu_1}{\sum_{k=1}^{K}\mu_k} &= \lambda_1 \frac{\alpha + \beta m_1}{\sum_{k=1}^{K}(\alpha + \beta m_k)} \\ &= \lambda_1 \frac{\alpha + \beta\left(\sigma_1 + (1-\sigma_1)n_1\right)}{K\alpha + \beta} \\ &= \lambda_1\left(\alpha + \beta\left(\sigma_1 + (1-\sigma_1)n_1\right)\right) \end{aligned} \tag{17}$$

The third equality in (16), for $i = j \neq 1$, is the same as the first equality, except that $\sigma_1 + (1-\sigma_1)n_j$ is replaced with $(1-\sigma_1)n_j$ as indicated by (15). In the second equality in (16), when $i = 1$ (corresponding to the best state) and $j \neq 1$ (corresponding to a suboptimal state), the transition probability only includes the probability that the individuals is changed, which is the same as the second term in the first equality. Finally, the fourth equality in (16) is the same as the third equality, except that $\sigma_1 + (1-\sigma_1)n_j$ is replaced with $(1-\sigma_1)n_j$ since $i \neq 1$ (that is, x_i is not the best state), as indicated by (15).

By incorporating the modified mutation probability described in (10), we obtain the transition probability of Strategy B with mutation. The *A* and *B* matrices described in (2) can be written as follows:

$$\begin{aligned} p_{kj} &= \sum_i p_{ki} p_{ij} \\ &= \begin{cases} p_m/K + (1-p_m)\left[(1-\lambda_1) + \lambda_1\left(\alpha + \beta\left(\sigma_1 + (1-\sigma_1)n_1\right)\right)\right] & \text{if } k = j = 1, \text{ which is the best state} \\ p_m/K + (1-p_m)\left[\lambda_j\left(\alpha + \beta\left(\sigma_1 + (1-\sigma_1)n_1\right)\right)\right] & \text{if } k = 1 \text{ and } j \neq 1 \\ p_m/K + (1-p_m)\left[(1-\lambda_j) + \lambda_j\left(\alpha + \beta(1-\sigma_1)n_k\right)\right] & \text{if } k = j \neq 1 \\ p_m/K + (1-p_m)\left[\lambda_j\left(\alpha + \beta(1-\sigma_1)n_k\right)\right] & \text{if } k \neq j, \text{ and } k \neq 1 \end{cases} \end{aligned} \tag{18}$$

and

$$a_{kj} = \begin{cases} p_m/K + (1-p_m)\left[(1-\lambda_1) + \lambda_1(\alpha + \beta\sigma_1)\right] & \text{if } k = j = 1, \text{ which is the best state} \\ p_m/K + (1-p_m)\left[\lambda_j(\alpha + \beta\sigma_1)\right] & \text{if } k = 1 \text{ and } j \neq 1 \\ p_m/K + (1-p_m)\left[(1-\lambda_j) + \lambda_j\alpha\right] & \text{if } k = j \neq 1 \\ p_m/K + (1-p_m)\left[\lambda_j\alpha\right] & \text{if } k \neq j, \text{ and } k \neq 1 \end{cases} \tag{19}$$

$$b_{kj} = (1-p_m)\lambda_j\beta(1-\sigma_1)$$

The derivation of (18) is in the appendix. Now we can state the main result of this subsection.

Theorem 3: Assume that the search space has a single global optimum. Then the $K\times1$ equilibrium fraction vector of movers n^* of Algorithm 2, which is exactly modeled by the interactive Markov model of (1) and (18), is equal to the dominant eigenvector (normalized so its elements sum to one) of the matrix $A_0 = a_{ij} \Big/ \left(b_j + \sum_{k=1}^{K} a_{kj} \right)$, where a_{ij} is given by (19), and b_j is shorthand notation for b_{ij} in (19) since all the rows of B are the same. Furthermore, the equilibrium fraction vector m^* is obtained by substituting n^* in (15).

Proof: The proof of Theorem 3 is analogous to that of Theorem 2, which is given in the appendix. ♦

E. Computational Complexity

The computational cost of Algorithms 1 and 2, like most other EAs, is dominated by the computational cost of the fitness function. Algorithms 1 and 2 compute the fitness of each individual in the population once per generation. Therefore, the computational cost of Algorithms 1 and 2 are the same order of magnitude as any other EA that uses a typical selection-evaluation-recombination strategy. Algorithms 1 and 2 also require a roulette-wheel process to select the replacement individual (x_i in Algorithms 1 and 2), which requires effort on the order of K^2, but that computational cost can be greatly reduced by using linear ranking [12, Section 8.7.5].

The computational cost of the interactive Markov model calculations requires the formation of the transition matrix components, which is (12) for Algorithm 1 and (19) for Algorithm 2. After the transition matrix is computed, the dominant eigenvector of a certain matrix needs to be computed in order to calculate the equilibrium population, as stated in Theorems 2 and 3. There are many methods for calculating eigenvectors, most of which have a computational cost on the order of K^3, where K is the order of the matrix, and which is equal to the cardinality of the search space in this paper.

In summary, using the interactive Markov chain model to calculate an equilibrium population requires three steps: (a) Calculation of the transition matrix components, as shown in (12) for Algorithm 1 and (19) for Algorithm 2; (b) Formation of a certain matrix, as shown in Theorem 2 for Algorithm 1 and Theorem 3 for Algorithm 2; (c) Calculation of a dominant eigenvector, as described in Theorem 2 for Algorithm 1 and Theorem 3 for Algorithm 2. The eigenvector calculation dominates the computational effort of these three steps, and is the on the order of K^3, where K is the cardinality of the search space.

III. Similarities of EAs

In this section we discuss some similarities between Strategy A, Strategy B, and other popular EAs.

First we point out the equivalences of Strategy A and Strategy B as presented in Algorithms 1 and 2 in Section II. Note that we only consider the case of selection here. If the replacement pool probability α = 1 in Strategy A, then it reduces to strategy B if elitism is not used (σ_1 = 0). Although the probability of selection of x_i in the two strategies appears different, they are essentially the same because they both use a selection probability that is a linear function of the fractions of the x_i individuals in the population.

Next we discuss the equivalence of Strategy B, and a genetic algorithm with global uniform recombination (GA/GUR) and elitism with no mutation, which is described in [22, Figure 3]. Strategy B and GA/GUR are equivalent under the following circumstances:

- In GA/GUR we replace an entire individual instead of only one decision variable at a time;
- In GA/GUR we use a selection probability that is proportional to the fractions of individuals in the population; and
- In strategy B we use modification probability $\lambda_k = 1$.

This implementation of GA/GUR has been called the Holland algorithm [23]. Algorithm 3 shows this implementation of GA/GUR.

ALGORITHM 3 – GENETIC ALGORITHM WITH GLOBAL UNIFORM RECOMBINATION (GA/GUR) WITH SELECTION PROBABILITY PROPORTIONAL TO THE FRACTIONS OF INDIVIDUALS IN THE POPULATION. THIS IS EQUIVALENT TO STRATEGY B IN ALGORITHM 2 IF λ_K = 1 FOR ALL K.

```
Generate an initial population of individuals Y= {y_k : k = 1,···,N }
While not (termination criterion)
    For each individual y_k
        If y_k is not the best individual in the search space then
            Use population proportions to probabilistically select x_i, i ∈ [1,K]
            y_k ← x_i
        End if
    Next individual: k ← k+1
Next generation
```

Next we discuss the equivalences of Strategy B, and biogeography-based optimization (BBO) [11] with elitism and no mutation. BBO is an evolutionary algorithm that is inspired by the migration of species between islands, and is described by Figure 2 in [22]. BBO involves the migration of decision variables between individuals in an EA population. If we allow

migration of entire individuals instead of decision variables, and if we set the BBO emigration probability proportional to the fractions of individuals in the population, then BBO becomes equivalent to EA Strategy B. Algorithm 4 shows this modified version of BBO.

ALGORITHM 4 – BIOGEOGRAPHY-BASED OPTIMIZATION WITH GLOBAL MIGRATION AND WITH SELECTION PROBABILITY PROPORTIONAL TO THE FRACTIONS OF INDIVIDUALS IN THE POPULATION. THIS IS EQUIVALENT TO STRATEGY B IN ALGORITHM 2.

Generate an initial population of individuals $Y = \{y_k : k = 1, \cdots, N\}$
While not (termination criterion)
  For each individua y_k
  If y_k is not the best individual in the search space then
    Use λ_k to probabilistically decide whether to immigrate to y_k
    If immigrating then
      Use population proportions to probabilistically select x_i, $i \in [1, K]$
      $y_k \leftarrow x_i$
    End immigration
  End if
  Next individual: $k \leftarrow k+1$
Next generation

There are many other well-established EAs, including evolutionary programming, differential evolution, evolution strategy, particle swarm optimization, ant colony optimization, and so on. Many of these algorithms are equivalent to each other under special circumstances [24], and so they may also be equivalent to our new Strategy A or Strategy B EAs under certain conditions. We leave this study to future research.

IV. SIMULATION RESULTS AND COMPARISONS

This subsection confirms the interactive Markov model theory of Section II with simulation results, and investigates the effects of tuning parameters on Strategy A and Strategy B.

A. Strategy A Results

In Strategy A, as outlined earlier in Algorithm 1, the main tuning parameters are the replacement pool probability α, and the parameter β of the selection probability $\beta/K + (1-\beta)m_i$. We use the fraction m_b^* of the best individual in the search space to compare performances with different tuning parameters. A larger fraction m_b^* indicates better performance. Test functions are limited to three problems with a search space cardinality of $K = 8$. Three fitness functions are investigated, which are given as

$$\begin{aligned} f_1 &= (1\ \ 2\ \ 2\ \ 3\ \ 2\ \ 3\ \ 3\ \ 4) \\ f_2 &= (4\ \ 2\ \ 2\ \ 3\ \ 2\ \ 3\ \ 3\ \ 4) \\ f_3 &= (4\ \ 1\ \ 1\ \ 2\ \ 1\ \ 2\ \ 2\ \ 3) \end{aligned} \tag{20}$$

These functions are chosen as representative functions because, when each individual is represented as three bits, f_1 is a unimodal one-max problem, f_2 is a multimodal problem, and f_3 is a deceptive problem.

We also use the two-dimensional multi-modal Ackley function to confirm the interactive Markov model of Strategy A. The Ackley function is described as follows:

$$\begin{aligned} f_4 = &-20\exp\left(-0.2\sqrt{\frac{\sum_{i=1}^{n} x_i^2}{n}}\right) \\ &-\exp\left(\frac{\sum_{i=1}^{n}\cos(2\pi x_i)}{n}\right) + 20 + e, \quad -32 \le x_i \le 32 \end{aligned} \tag{21}$$

We deal with discrete optimization functions, so the granularity or precision of each independent variable is set to 1 in this section.

In this section, we test $\alpha = 0.25, 0.50, 0.75$, and $\beta = 0.25, 0.50, 0.75$, with modification probabilities $\lambda_i = 1 - \mathit{fitness}(x_i)$, where $\mathit{fitness}(x_i)$ denotes the fitness value of individual x_i, which is normalized to the range [0, 1]. The other EA parameters are population size = 50, generation limit = 20,000, and 30 Monte Carlo runs for each test. Tables 2–4 show comparisons between theoretical interactive Markov results (Theorem 2 in Section II) and Strategy A simulation results with various values of parameters α, β, and the mutation rate p_m. Table 5 shows similar comparisons for Strategy A for the special case $\beta = 1$, which gives selection probability $\mu_i = 1/K$ for all i, which is independent of population fractions. The results in Tables 2–5 can be reproduced with MATLAB® code that is available at the authors' web site [25].

TABLE 2 – STRATEGY A RESULTS FOR TEST PROBLEMS WITH $\beta = 0.25$ AND DIFFERENT MUTATION RATES. THE NUMBERS IN THE TABLE SHOW THE PROPORTION OF OPTIMAL INDIVIDUALS IN THE POPULATION.

		No mutation		$p_m = 0.001$		$p_m = 0.01$		$p_m = 0.1$	
		Markov	Simulation	Markov	Simulation	Markov	Simulation	Markov	Simulation
f_1	$\alpha = 0.25$	0.8155	0.8154	0.8016	0.7956	0.6829	0.6844	0.2232	0.2265
	$\alpha = 0.50$	0.8155	0.8183	0.8086	0.7983	0.7471	0.7354	0.3342	0.3226
	$\alpha = 0.75$	0.8155	0.8126	0.8109	0.7996	0.7694	0.7642	0.4288	0.4209
f_2	$\alpha = 0.25$	0.8412	0.8463	0.8298	0.8249	0.7356	0.7234	0.3736	0.3782
	$\alpha = 0.50$	0.8412	0.8407	0.8356	0.8307	0.7858	0.7795	0.4748	0.4666
	$\alpha = 0.75$	0.8412	0.8445	0.8374	0.8270	0.8038	0.8032	0.5474	0.5424
f_3	$\alpha = 0.25$	0.8645	0.8616	0.8542	0.8450	0.7630	0.7671	0.2727	0.2719
	$\alpha = 0.50$	0.8645	0.8644	0.8593	0.8583	0.8128	0.8036	0.4315	0.4247
	$\alpha = 0.75$	0.8645	0.8643	0.8611	0.8567	0.8298	0.8250	0.5393	0.5267
f_4	$\alpha = 0.25$	0.8109	0.8137	0.8015	0.7961	0.7253	0.7278	0.2076	0.2079
	$\alpha = 0.50$	0.8109	0.8125	0.8031	0.8000	0.7327	0.7270	0.2135	0.2190
	$\alpha = 0.75$	0.8109	0.8131	0.8057	0.8056	0.7586	0.7613	0.3869	0.3856
Ave. CPU time (s)		–	81.7	–	83.4	–	86.2	–	89.3

TABLE 3 – STRATEGY A RESULTS FOR TEST PROBLEMS WITH $\beta = 0.5$ AND DIFFERENT MUTATION RATES. THE NUMBERS IN THE TABLE SHOW THE PROPORTION OF OPTIMAL INDIVIDUALS IN THE POPULATION.

		No mutation		$p_m = 0.001$		$p_m = 0.01$		$p_m = 0.1$	
		Markov	Simulation	Markov	Simulation	Markov	Simulation	Markov	Simulation
f_1	$\alpha = 0.25$	0.5667	0.5696	0.5544	0.5482	0.4598	0.4621	0.2012	0.2043
	$\alpha = 0.50$	0.5667	0.5655	0.5605	0.5489	0.5085	0.4928	0.2622	0.2561
	$\alpha = 0.75$	0.5667	0.5618	0.5626	0.5410	0.5268	0.5181	0.3095	0.3009
f_2	$\alpha = 0.25$	0.6658	0.6651	0.6570	0.6426	0.5872	0.5830	0.3528	0.3535
	$\alpha = 0.50$	0.6658	0.6623	0.6614	0.6562	0.6234	0.6140	0.4198	0.4180
	$\alpha = 0.75$	0.6658	0.6682	0.6628	0.6587	0.6368	0.6282	0.4650	0.4648
f_3	$\alpha = 0.25$	0.6631	0.6611	0.6522	0.6463	0.5623	0.5598	0.2338	0.2334
	$\alpha = 0.50$	0.6631	0.6660	0.6576	0.6430	0.6100	0.6097	0.3231	0.3230
	$\alpha = 0.75$	0.6631	0.6657	0.6595	0.6481	0.6271	0.6231	0.3882	0.3815
f_4	$\alpha = 0.25$	0.7404	0.7387	0.7184	0.7118	0.5341	0.5318	0.0413	0.0428
	$\alpha = 0.50$	0.7404	0.7337	0.7294	0.7204	0.6313	0.6300	0.0956	0.0880
	$\alpha = 0.75$	0.7404	0.7398	0.7331	0.7382	0.6671	0.6599	0.1698	0.1526
Ave. CPU time (s)		–	82.9	–	85.1	–	87.2	–	91.0

TABLE 4 – STRATEGY A RESULTS FOR TEST PROBLEMS WITH $\beta = 0.75$ AND DIFFERENT MUTATION RATES. THE NUMBERS IN THE TABLE SHOW THE PROPORTION OF OPTIMAL INDIVIDUALS IN THE POPULATION.

		No mutation		$p_m = 0.001$		$p_m = 0.01$		$p_m = 0.1$	
		Markov	Simulation	Markov	Simulation	Markov	Simulation	Markov	Simulation
f_1	$\alpha = 0.25$	0.3615	0.3695	0.3559	0.3452	0.3142	0.3129	0.1868	0.1883
	$\alpha = 0.50$	0.3615	0.3618	0.3587	0.3497	0.3354	0.3317	0.2232	0.2220
	$\alpha = 0.75$	0.3615	0.3610	0.3596	0.3593	0.3435	0.3425	0.2473	0.2518
f_2	$\alpha = 0.25$	0.5276	0.5260	0.5224	0.5158	0.4828	0.4850	0.3374	0.3337
	$\alpha = 0.50$	0.5276	0.5252	0.5224	0.5219	0.5034	0.5045	0.3834	0.3852
	$\alpha = 0.75$	0.5276	0.5289	0.5260	0.5194	0.5110	0.5094	0.4116	0.4122
f_3	$\alpha = 0.25$	0.4394	0.4396	0.4326	0.4321	0.3809	0.3814	0.2095	0.2099
	$\alpha = 0.50$	0.4394	0.4398	0.4360	0.4375	0.4075	0.4076	0.2599	0.2609
	$\alpha = 0.75$	0.4394	0.4389	0.4371	0.4370	0.4175	0.4100	0.2928	0.2959
f_4	$\alpha = 0.25$	0.3521	0.3566	0.3274	0.3204	0.1805	0.1834	0.0356	0.0362
	$\alpha = 0.50$	0.3521	0.3585	0.3395	0.3311	0.2463	0.2411	0.0547	0.0556
	$\alpha = 0.75$	0.3521	0.3539	0.3436	0.3406	0.2764	0.2805	0.0730	0.0711
Ave. CPU time (s)		–	88.6	–	90.4	–	93.5	–	97.6

TABLE 5 – STRATEGY A RESULTS FOR TEST PROBLEMS WITH $\beta = 1$ AND DIFFERENT MUTATION RATES. THE NUMBERS IN THE TABLE SHOW THE PROPORTION OF OPTIMAL INDIVIDUALS IN THE POPULATION.

		No mutation		$p_m = 0.001$		$p_m = 0.01$		$p_m = 0.1$	
		Markov	Simulation	Markov	Simulation	Markov	Simulation	Markov	Simulation
f_1	$\alpha = 0.25$	0.2667	0.2654	0.2644	0.2675	0.2469	0.2475	0.1767	0.1797
	$\alpha = 0.50$	0.2667	0.2665	0.2655	0.2667	0.2560	0.2572	0.2004	0.1988
	$\alpha = 0.75$	0.2667	0.2665	0.2659	0.2691	0.2594	0.2604	0.2142	0.2161
f_2	$\alpha = 0.25$	0.4444	0.4445	0.4416	0.4475	0.4198	0.4225	0.3258	0.3279
	$\alpha = 0.50$	0.4444	0.4453	0.4430	0.4462	0.4312	0.4305	0.3586	0.3574
	$\alpha = 0.75$	0.4444	0.4453	0.4436	0.4406	0.4354	0.4387	0.3772	0.3767
f_3	$\alpha = 0.25$	0.3077	0.3066	0.3049	0.3037	0.2833	0.2823	0.1935	0.1938
	$\alpha = 0.50$	0.3077	0.3067	0.3063	0.3062	0.2946	0.2896	0.2243	0.2232
	$\alpha = 0.75$	0.3077	0.3076	0.3068	0.3081	0.2987	0.3011	0.2420	0.2436
f_4	$\alpha = 0.25$	0.0842	0.0815	0.0818	0.0812	0.0662	0.0672	0.0307	0.0323
	$\alpha = 0.50$	0.0842	0.0840	0.0829	0.0818	0.0737	0.0711	0.0398	0.0400
	$\alpha = 0.75$	0.0842	0.0828	0.0833	0.0865	0.0768	0.0780	0.0461	0.0471
Ave. CPU time (s)		–	71.2	–	73.3	–	75.9	–	78.4

Several things are notable about Tables 2–5. First, we note that the parent selection probability α does not affect the proportion of individuals in the population in the case of no mutation. This is because α divides out of the $m^* = P(m^*)m^*$ equilibrium equation in this case. The finding is consistent with [20, p. 162]. However, we see that α can slightly affect the proportion of individuals in the case of nonzero mutation.

Second, for a given parent selection probability α and a given parameter β, the proportion of optimal individuals decreases with the mutation rate p_m. This indicates that low mutation rates have better performance for the test problems that we study. A high mutation rate of 0.1 results in too much exploration, and the population remains too widely distributed across the search space.

Third, for a given parent selection probability α and a given mutation rate p_m, the proportion of optimal individuals decreases with increasing β. This is because the modification probability λ_i tends to result in a population in which good individuals dominate, and increasing β causes individuals with high populations to be more likely to replace other individuals.

Fourth, Tables 2–5 show that the interactive Markov model results and the simulation results match well for all test problems, which confirms the interactive Markov model theory.

Fifth, the average CPU times in the last rows of Tables 2–5 show the simulation times of Strategy A for the four test problems. Strategy A runs faster with smaller mutation rates. The reason is that larger mutation rates require more mutation operations, which slightly increases computation time. However, in more realistic and interesting real-world problems, computational effort it dominated by fitness function evaluation, which is independent of the mutation rate.

B. Strategy B Results

In this section we investigate the effect of selection pressure ϕ, which influences the selection probability $\alpha + \beta m_i$ of x_i in Strategy B, as shown in (7). In this section we test only the unimodal one-max problem f_1 (recall that Theorem 3 assumes that the optimization problem is unimodal). Recall that EA selection pressure is constrained to the domain $\phi \in (1, 2)$ [3, p. 34]. In this section, we test $\phi = 1.25, 1.50, 1.75$ in (7) to compute parameters α, β of the population-proportion-based selection probability, and we test elitism probabilities $\sigma_1 = 0.25, 0.50, 0.75$. The other parameters of the EA are the same as those described above in the previous subsection. Table 6 shows comparisons between interactive Markov theory results and Strategy B simulation results. The results in Table 6 can be reproduced with MATLAB® code that is available at the authors' web site [25].

TABLE 6 – STRATEGY B RESULTS FOR TEST PROBLEM F_1.
THE NUMBERS IN THE TABLE SHOW THE PROPORTION OF OPTIMAL INDIVIDUALS IN THE POPULATION.

		No mutation		$p_m = 0.001$		$p_m = 0.01$		$p_m = 0.1$	
		Markov	Simulation	Markov	Simulation	Markov	Simulation	Markov	Simulation
$\phi = 1.25$	$\sigma_1 = 0.25$	0.3139	0.3164	0.3131	0.3134	0.3060	0.3081	0.2529	0.2526
	$\sigma_1 = 0.50$	0.3372	0.3354	0.3363	0.3329	0.3287	0.3295	0.2711	0.2736
	$\sigma_1 = 0.75$	0.3582	0.3559	0.3573	0.3561	0.3493	0.3467	0.2880	0.2925
$\phi = 1.50$	$\sigma_1 = 0.25$	0.4022	0.4056	0.4009	0.4019	0.3897	0.3926	0.3068	0.3063
	$\sigma_1 = 0.50$	0.4514	0.4517	0.4500	0.4479	0.4384	0.4431	0.3494	0.3451
	$\sigma_1 = 0.75$	0.4901	0.4938	0.4887	0.4929	0.4770	0.4780	0.3848	0.3842
$\phi = 1.75$	$\sigma_1 = 0.25$	0.5955	0.5914	0.5933	0.5950	0.5740	0.5749	0.4271	0.4262
	$\sigma_1 = 0.50$	0.6511	0.6522	0.6492	0.6452	0.6320	0.6323	0.4944	0.4922
	$\sigma_1 = 0.75$	0.6893	0.6869	0.6875	0.6864	0.6719	0.6687	0.5423	0.5393
Ave. CPU time (s)		–	72.4	–	75.3	–	79.1	–	85.2

We note several things from Table 6. First, for a given value of elitism probability σ_1 and mutation rate p_m, performance improves as selection pressure ϕ increases. This is expected because a larger value of ϕ exploits more information from the population. For a more complicated problem with a larger search space, we might arrive at different conclusions about the effect of ϕ on performance.

Second, for a given value of selection pressure ϕ and mutation rate p_m, performance improves as elitism probability σ_1 increases. Again, this is expected for simple problems such as the test problem studied in this section, but the conclusion may not hold for more complicated problems.

Third, for a given value of elitism probability σ_1 and selection pressure ϕ, performance improves as mutation rate p_m decreases. This again indicates that low mutation rates give better performance for the test problems that we study. A high mutation rate of 0.1 results in too much exploration, and the population remains too widely distributed across the search space.

Fourth, we see that the interactive Markov model theory and the simulation results match well, which confirms the interactive Markov model theory.

Fifth, we see that Strategy B runs faster with smaller mutation rates (the same observation we made for Strategy A in Tables 2–5). The reason is that larger mutation rates result in more mutation operations, which slightly increases computation time.

V. CONCLUSION

This paper first presented a formal interactive Markov model, which involves separate but interacting Markov models for each individual in an EA population. This is a new model for studying EAs. Then we proposed two simple EAs whose basic features are population-proportion-based selection and modified mutation, and analyzed them exactly with interactive Markov models. The theoretical results were confirmed with simulation results, and showed how the interactive Markov model can describe the convergence of the EAs. The theoretical and simulation results in Tables 2–6 can be reproduced with MATLAB® code that is available at the authors' web site [25].

The use of interactive Markov models to model evolutionary algorithms can lead to useful conclusions. Interactive Markov models prove to be tractable for much larger search spaces and populations than noninteractive Markov models. The noninteractive (standard) Markov model has a state space whose dimension grows factorially with search space cardinality and population size, while the interactive Markov model has a state space whose dimension grows linearly with the cardinality of the search space, and is independent of population size. Like the noninteractive Markov model, the interactive Markov model provides exact models for the behavior of the EA. Interactive Markov models can be studied as functions of EA tuning parameters to predict their impact on EA performance, and to provide real-time adaptation. Just as noninteractive Markov models have led to the development of dynamic system models, the same can happen with interactive

Markov models. Although the interactive Markov models in this paper explicitly provide only steady-state probabilities, they might also be used to understand transient EA behavior, and to obtain the probability of optimum-hitting each generation, and to obtain expected hitting times. We see some research in this direction for noninteractive Markov models [27], [28]; such results are impractical for real-world problems due to the large transition matrices of noninteractive Markov models, but such a limitation will not be as great a concern for interactive Markov models.

For future work beyond the suggestions listed above, we see several important directions. First, the interactive Markov model analysis of this paper was based on two simple EAs; future work should explore how to apply the model to other EAs. Second, we only used two examples from the earlier literature to derive new EAs, but we could use other previously-published examples to construct additional EA paradigms that use population-proportion-based selection. Third, population-proportion-based selection is a new selection strategy that does not require fitness calculations (possible computational cost savings), and future work could develop an entire family of modified EAs based on this selection strategy.

Fourth, we suggest for future work the combination of estimation of distribution algorithms (EDAs) with our newly-proposed population-proportion-based selection operator. Recall that EDAs use fitness values to approximate the distribution of an EA population's fitness values. In contrast, our population-proportion-based selection uses population sizes rather than fitness values for selection. However, EDA ideas could be incorporated in population-proportion-based selection by approximating the probability distribution of the population sizes, and then performing selection on the basis of approximate distribution. This idea would merge the advantages of EDAs with the advantages of population-proportion-based selection.

Finally, we note that methods will need to be developed to handle problems with realistic sizes. The interactive Markov model presented here enables tractability for problems of reasonable size, which is a significant advantage over the standard noninteractive Markov models published before now. However, the interactive Markov model is still the same size as the search space, which can be quite large. For realistic problem sizes, say with a search space on the order of trillions, the interactive Markov model will also be on the order of trillions. Methods will need to be developed to reduce the interactive Markov model to a tractable size.

APPENDIX

A. Here we derive the interactive Markov model of strategy A with mutation shown in (11). The selection transition matrix P_s of Strategy A in (8) can be written as

$$P_S=\left[p_{ij}\right]=\begin{bmatrix}(1-\alpha)+\alpha\left[(1-\lambda_1)+\lambda_1\left(\beta/K+(1-\beta)m_1\right)\right] & \alpha\lambda_2\left(\beta/K+(1-\beta)m_1\right) & \cdots & \alpha\lambda_K\left(\beta/K+(1-\beta)m_1\right)\\ \alpha\lambda_1\left(\beta/K+(1-\beta)m_2\right) & (1-\alpha)+\alpha\left[(1-\lambda_2)+\lambda_2\left(\beta/K+(1-\beta)m_2\right)\right] & \cdots & \alpha\lambda_K\left(\beta/K+(1-\beta)m_2\right)\\ \vdots & \cdots & \cdots & \vdots\\ \alpha\lambda_1\left(\beta/K+(1-\beta)m_K\right) & \alpha\lambda_2\left(\beta/K+(1-\beta)m_K\right) & \cdots & (1-\alpha)+\alpha\left[(1-\lambda_K)+\lambda_K\left(\beta/K+(1-\beta)m_K\right)\right]\end{bmatrix} \tag{A.1}$$

Transition matrix P_M of the modified mutation in (10) can be written as

$$P_M=\left[p_{ki}\right]=\begin{bmatrix}(1-p_m)+p_m/K & p_m/K & \cdots & p_m/K\\ p_m/K & (1-p_m)+p_m/K & \cdots & p_m/K\\ \vdots & \cdots & \cdots & \vdots\\ p_m/K & p_m/K & \cdots & (1-p_m)+p_m/K\end{bmatrix} \tag{A.2}$$

where p_m is the mutation rate. So the transition matrix of Strategy A with mutation can be computed by $P_A=\left[p_{kj}\right]=P_M\cdot P_S=\sum_i p_{ki}p_{ij}$:

$$P_A=\left[p_{kj}\right]=\begin{bmatrix}(1-p_m)+p_m/K & p_m/K & \cdots & p_m/K\\ p_m/K & (1-p_m)+p_m/K & \cdots & p_m/K\\ \vdots & \cdots & \cdots & \vdots\\ p_m/K & p_m/K & \cdots & (1-p_m)+p_m/K\end{bmatrix}$$
$$\cdot\begin{bmatrix}(1-\alpha)+\alpha\left[(1-\lambda_1)+\lambda_1\left(\beta/K+(1-\beta)m_1\right)\right] & \alpha\lambda_2\left(\beta/K+(1-\beta)m_1\right) & \cdots & \alpha\lambda_K\left(\beta/K+(1-\beta)m_1\right)\\ \alpha\lambda_1\left(\beta/K+(1-\beta)m_2\right) & (1-\alpha)+\alpha\left[(1-\lambda_2)+\lambda_2\left(\beta/K+(1-\beta)m_2\right)\right] & \cdots & \alpha\lambda_K\left(\beta/K+(1-\beta)m_2\right)\\ \vdots & \cdots & \cdots & \vdots\\ \alpha\lambda_1\left(\beta/K+(1-\beta)m_K\right) & \alpha\lambda_2\left(\beta/K+(1-\beta)m_K\right) & \cdots & (1-\alpha)+\alpha\left[(1-\lambda_K)+\lambda_K\left(\beta/K+(1-\beta)m_K\right)\right]\end{bmatrix} \tag{A.3}$$

Element $P_A(1,1)=p_{11}$ in (A.3) is obtained as follows.

$$
\begin{aligned}
P_A(1,1) &= \big((1-p_m)+p_m/K\big)\Big[(1-\alpha)+\alpha\big((1-\lambda_1)+\lambda_1\big(\beta/K+(1-\beta)m_1\big)\big)\Big]+(p_m/K)\Big[\alpha\lambda_1\big(\beta/K+(1-\beta)m_2\big)\Big]+\cdots+(p_m/K)\Big[\alpha\lambda_1\big(\beta/K+(1-\beta)m_K\big)\Big] \\
&= \big((1-p_m)+p_m/K\big)\Big[(1-\alpha)+\alpha\big((1-\lambda_1)+\lambda_1(\beta/K)\big)\Big]+(K-1)(p_m/K)(\alpha\lambda_1)(\beta/K)+\big((1-p_m)+p_m/K\big)(\alpha\lambda_1)(1-\beta)m_1 \\
&\quad +(p_m/K)(\alpha\lambda_1)(1-\beta)(m_2+m_3+\cdots m_K) \\
&= (1-\frac{(K-1)p_m}{K})\Big[(1-\alpha)+\alpha\Big((1-\lambda_1)+\lambda_1\Big(\frac{\beta}{K}\Big)\Big)\Big]+\frac{(K-1)p_m}{K}(\alpha\lambda_1)\Big(\frac{\beta}{K}\Big)+(1-\frac{(K-1)p_m}{K})(\alpha\lambda_1)(1-\beta)m_1+\frac{p_m}{K}(\alpha\lambda_1)(1-\beta)(1-m_1) \\
&= (1-\frac{(K-1)p_m}{K})(1-\alpha)+\alpha\Big[(1-\frac{(K-1)p_m}{K})(1-\lambda_1)+\lambda_1\Big((1-\frac{(K-1)p_m}{K})\frac{\beta}{K}+\frac{(K-1)p_m}{K^2}\beta+\frac{p_m}{K}(1-\beta)+(1-\beta)(1-p_m)m_1\Big)\Big]
\end{aligned}
\tag{A.4}
$$

Element $P_A(1,2)=p_{12}$ in (A.3) is obtained as follows.

$$
\begin{aligned}
P_A(1,2) &= \big((1-p_m)+p_m/K\big)\Big[\alpha\lambda_2\big(\beta/K+(1-\beta)m_1\big)\Big]+(p_m/K)\Big[(1-\alpha)+\alpha\big((1-\lambda_2)+\lambda_2\big(\beta/K+(1-\beta)m_2\big)\big)\Big]+\cdots+(p_m/K)\Big[\alpha\lambda_2\big(\beta/K+(1-\beta)m_K\big)\Big] \\
&= \big((1-p_m)+p_m/K\big)(\alpha\lambda_2)(\beta/K)+\big((1-p_m)+p_m/K\big)(\alpha\lambda_2)\big((1-\beta)m_1\big)+(p_m/K)\Big[(1-\alpha)+\alpha\big((1-\lambda_2)+\lambda_2(\beta/K)\big)\Big] \\
&\quad +(p_m/K)(\alpha\lambda_2)(\beta/K)+(p_m/K)(\alpha\lambda_2)(1-\beta)m_2+\cdots+(p_m/K)(\alpha\lambda_2)(\beta/K)+(p_m/K)(\alpha\lambda_2)\big((1-\beta)m_K\big) \\
&= \big((1-p_m)+p_m/K\big)(\alpha\lambda_2)(\beta/K)+(p_m/K)\Big[(1-\alpha)+\alpha\big((1-\lambda_2)+\lambda_2(\beta/K)\big)\Big]+(K-2)(p_m/K)(\alpha\lambda_2)(\beta/K) \\
&\quad +\big((1-p_m)+p_m/K\big)(\alpha\lambda_2)\big((1-\beta)m_1\big)+(p_m/K)(\alpha\lambda_2)(1-\beta)(m_2+\cdots+m_K) \\
&= \frac{(K-1)p_m}{K}(\alpha\lambda_2)\Big(\frac{\beta}{K}\Big)+(\frac{p_m}{K})\Big[(1-\alpha)+\alpha\Big((1-\lambda_2)+\lambda_2\Big(\frac{\beta}{K}\Big)\Big)\Big]+\frac{(K-2)p_m}{K}(\alpha\lambda_2)\Big(\frac{\beta}{K}\Big) \\
&\quad +(1-\frac{(K-1)p_m}{K})(\alpha\lambda_2)(1-\beta)m_1+\frac{p_m}{K}(\alpha\lambda_2)(1-\beta)(1-m_1) \\
&= \frac{p_m}{K}(1-\alpha)+\alpha\Big[\frac{p_m}{K}(1-\lambda_2)+\lambda_2\Big((1-\frac{(K-1)p_m}{K})\frac{\beta}{K}+\frac{(K-1)p_m}{K^2}\beta+\frac{p_m}{K}(1-\beta)+(1-\beta)(1-p_m)m_1\Big)\Big]
\end{aligned}
\tag{A.5}
$$

We follow the same process to obtain

$$
p_{kj}=\begin{cases}
(1-\frac{(K-1)p_m}{K})(1-\alpha)+\alpha\Big[(1-\frac{(K-1)p_m}{K})(1-\lambda_j)+\lambda_j\Big((1-\frac{(K-1)p_m}{K})\frac{\beta}{K}+\frac{(K-1)p_m}{K^2}\beta+\frac{p_m}{K}(1-\beta)+(1-\beta)(1-p_m)m_j\Big)\Big] & \text{if } k=j \\
\frac{p_m}{K}(1-\alpha)+\alpha\Big[\frac{p_m}{K}(1-\lambda_j)+\lambda_j\Big((1-\frac{(K-1)p_m}{K})\frac{\beta}{K}+\frac{(K-1)p_m}{K^2}\beta+\frac{p_m}{K}(1-\beta)+(1-\beta)(1-p_m)m_k\Big)\Big] & \text{if } k\neq j
\end{cases}
\tag{A.6}
$$

which is equivalent to (11), as desired.

B. Here we derive the interactive Markov model of strategy B with mutation as shown in (18). The selection transition matrix P_S of Strategy B in (16) can be written as

$$
P_S=[p_{ij}]=\begin{bmatrix}
(1-\lambda_1)+\lambda_1\big[\alpha+\beta(\sigma_1+(1-\sigma_1)n_1)\big] & \lambda_2\big[\alpha+\beta(\sigma_1+(1-\sigma_1)n_1)\big] & \cdots & \lambda_K(\alpha+\beta(1-\sigma_1)n_1) \\
\lambda_1(\alpha+\beta(1-\sigma_1)n_2) & (1-\lambda_2)+\lambda_2(\alpha+\beta(1-\sigma_1)n_2) & \cdots & \lambda_K(\alpha+\beta(1-\sigma_1)n_2) \\
\vdots & \cdots & \cdots & \vdots \\
\lambda_1(\alpha+\beta(1-\sigma_1)n_K) & \lambda_2(\alpha+\beta(1-\sigma_1)n_K) & \cdots & (1-\lambda_K)+\lambda_K(\alpha+\beta(1-\sigma_1)n_K)
\end{bmatrix}
\tag{B.1}
$$

Transition matrix P_M of the modified mutation operator (10) can be written as

$$
P_M=[p_{ki}]=\begin{bmatrix}
(1-p_m)+p_m/K & p_m/K & \cdots & p_m/K \\
p_m/K & (1-p_m)+p_m/K & \cdots & p_m/K \\
\vdots & \cdots & \cdots & \vdots \\
p_m/K & p_m/K & \cdots & (1-p_m)+p_m/K
\end{bmatrix}
\tag{B.2}
$$

So the transition matrix of Strategy B with mutation can be computed as $P_B=[p_{kj}]=P_M\cdot P_S=\sum_i p_{ki}p_{ij}$:

$$
\begin{aligned}
P_B=[p_{kj}] &= \begin{bmatrix}
(1-p_m)+P_m/K & p_m/K & \cdots & p_m/K \\
p_m/K & (1-p_m)+p_m/K & \cdots & p_m/K \\
\vdots & \cdots & \cdots & \vdots \\
p_m/K & p_m/K & \cdots & (1-p_m)+p_m/K
\end{bmatrix} \\
&\cdot \begin{bmatrix}
(1-\lambda_1)+\lambda_1\big[\alpha+\beta(\sigma_1+(1-\sigma_1)n_1)\big] & \lambda_2\big[\alpha+\beta(\sigma_1+(1-\sigma_1)n_1)\big] & \cdots & \lambda_K(\alpha+\beta(1-\sigma_1)n_1) \\
\lambda_1(\alpha+\beta(1-\sigma_1)n_2) & (1-\lambda_2)+\lambda_2(\alpha+\beta(1-\sigma_1)n_2) & \cdots & \lambda_K(\alpha+\beta(1-\sigma_1)n_2) \\
\vdots & \cdots & \cdots & \vdots \\
\lambda_1(\alpha+\beta(1-\sigma_1)n_K) & \lambda_2(\alpha+\beta(1-\sigma_1)n_K) & \cdots & (1-\lambda_K)+\lambda_K(\alpha+\beta(1-\sigma_1)n_K)
\end{bmatrix}
\end{aligned}
\tag{B.3}
$$

Element $P_B(1,1)=p_{11}$ in (B.3) is obtained as follows.

$$
\begin{aligned}
P_B(1,1) &= \big((1-p_m)+p_m/K\big)\Big[(1-\lambda_1)+\lambda_1\big[\alpha+\beta\big(\sigma_1+(1-\sigma_1)n_1\big)\big]\Big]+(p_m/K)\Big[\lambda_1\big(\alpha+\beta(1-\sigma_1)n_2\big)\Big]+\cdots+(p_m/K)\Big[\lambda_1\big(\alpha+\beta(1-\sigma_1)n_K\big)\Big] \\
&= \big((1-p_m)+p_m/K\big)\big[(1-\lambda_1)+\lambda_1\alpha\big]+(K-1)(p_m/K)(\alpha\lambda_1)+\big((1-p_m)+p_m/K\big)\big(\lambda_1\beta\big(\sigma_1+(1-\sigma_1)n_1\big)\big) \\
&\quad +(p_m/K)\big(\lambda_1\beta(1-\sigma_1)(n_2+n_3+\cdots n_K)\big) \\
&= \big((1-p_m)+p_m/K\big)\big[(1-\lambda_1)+\lambda_1\alpha\big]+(K-1)(p_m/K)(\alpha\lambda_1)+\big((1-p_m)+p_m/K\big)\big(\lambda_1\beta\big(\sigma_1+(1-\sigma_1)n_1\big)\big) \\
&\quad +(p_m/K)\big(\lambda_1\big(1-K\alpha-\beta\big(\sigma_1+(1-\sigma_1)n_1\big)\big)\big) \\
&= p_m/K+(1-p_m)\big[(1-\lambda_1)+\lambda_1\big(\alpha+\beta\big(\sigma_1+(1-\sigma_1)n_1\big)\big)\big]
\end{aligned}
\tag{B.4}
$$

Element $P_B(1,2)=p_{12}$ in (B.3) is obtained as follows.

$$
\begin{aligned}
P_B(1,2) &= \big((1-p_m)+p_m/K\big)\lambda_2\big[\alpha+\beta\big(\sigma_1+(1-\sigma_1)n_1\big)\big]+(p_m/K)\big[(1-\lambda_2)+\lambda_2\big(\alpha+\beta(1-\sigma_1)n_2\big)\big]+\cdots+(p_m/K)\big[\lambda_2\big(\alpha+\beta(1-\sigma_1)n_K\big)\big] \\
&= \big((1-p_m)+p_m/K\big)(\alpha\lambda_2)+\big((1-p_m)+p_m/K\big)\big(\lambda_2\beta\big(\sigma_1+(1-\sigma_1)n_1\big)\big)+(p_m/K)\big(\lambda_2\beta(1-\sigma_1)(n_2+n_3+\cdots+n_K)\big) \\
&\quad +(p_m/K)(1-\lambda_2)+(K-1)(p_m/K)\lambda_2\alpha \\
&= \big((1-p_m)+p_m/K\big)(\alpha\lambda_2)+\big((1-p_m)+p_m/K\big)\big(\lambda_2\beta\big(\sigma_1+(1-\sigma_1)n_1\big)\big)+(p_m/K)(1-\lambda_2)+(K-1)(p_m/K)\lambda_2\alpha \\
&\quad +(p_m/K)\big(\lambda_2\big(1-K\alpha-\beta\big(\sigma_1+(1-\sigma_1)n_1\big)\big)\big) \\
&= p_m/K+(1-p_m)\big[\lambda_2\big(\alpha+\beta\big(\sigma_1+(1-\sigma_1)n_1\big)\big)\big]
\end{aligned}
\tag{B.5}
$$

Element $P_B(2,2)=p_{22}$ in (B.3) is obtained as follows.

$$
\begin{aligned}
P_B(2,2) &= (p_m/K)\lambda_2\big[\alpha+\beta\big(\sigma_1+(1-\sigma_1)n_1\big)\big]+\big((1-p_m)+p_m/K\big)\big[(1-\lambda_2)+\lambda_2\big(\alpha+\beta(1-\sigma_1)n_2\big)\big]+\cdots+(p_m/K)\big[\lambda_2\big(\alpha+\beta(1-\sigma_1)n_K\big)\big] \\
&= (p_m/K)\lambda_2\beta\sigma_1+\big((1-p_m)+p_m/K\big)(1-\lambda_2)+\big((1-p_m)+p_m/K\big)\lambda_2\alpha+(K-1)(p_m/K)\lambda_2\alpha+\big((1-p_m)+p_m/K\big)\big(\lambda_2\beta(1-\sigma_1)n_2\big) \\
&\quad +(p_m/K)\big[\lambda_2\beta(1-\sigma_1)(n_1+n_3+\cdots+n_K)\big] \\
&= (p_m/K)\lambda_2\beta\sigma_1+\big((1-p_m)+p_m/K\big)(1-\lambda_2)+\big((1-p_m)+p_m/K\big)\lambda_2\alpha+(K-1)(p_m/K)\lambda_2\alpha+\big((1-p_m)+p_m/K\big)\big(\lambda_2\beta(1-\sigma_1)n_2\big) \\
&\quad +(p_m/K)\big(\lambda_2\big(1-K\alpha-\beta\sigma_1-\beta(1-\sigma_1)n_2\big)\big) \\
&= p_m/K+(1-p_m)\big[(1-\lambda_2)+\lambda_2\big(\alpha+\beta(1-\sigma_1)n_2\big)\big]
\end{aligned}
\tag{B.6}
$$

Element $P_B(2,1)=p_{21}$ in (B.3) is obtained as follows.

$$
\begin{aligned}
P_B(2,1) &= (p_m/K)\Big[(1-\lambda_1)+\lambda_1\big(\alpha+\beta\big(\sigma_1+(1-\sigma_1)n_1\big)\big)\Big]+\big((1-p_m)+p_m/K\big)\lambda_1\big(\alpha+\beta(1-\sigma_1)n_2\big)+\cdots+(p_m/K)\Big[\lambda_1\big(\alpha+\beta(1-\sigma_1)n_K\big)\Big] \\
&= (p_m/K)(1-\lambda_1)+(K-1)(p_m/K)\lambda_1\alpha+(p_m/K)(\lambda_1\beta\sigma_1)+\big((1-p_m)+p_m/K\big)(\alpha\lambda_1)+\big((1-p_m)+p_m/K\big)\big(\lambda_1\beta(1-\sigma_1)n_2\big) \\
&\quad +(p_m/K)\big(\lambda_1\beta(1-\sigma_1)(n_1+n_3+\cdots+n_K)\big) \\
&= (p_m/K)(1-\lambda_1)+(K-1)(p_m/K)\lambda_1\alpha+(p_m/K)(\lambda_1\beta\sigma_1)+\big((1-p_m)+p_m/K\big)(\alpha\lambda_1)+\big((1-p_m)+p_m/K\big)\big(\lambda_1\beta(1-\sigma_1)n_2\big) \\
&\quad +(p_m/K)\big(\lambda_1\big(1-K\alpha-\beta\sigma_1-\beta(1-\sigma_1)n_2\big)\big) \\
&= p_m/K+(1-p_m)\big[\lambda_1\big(\alpha+\beta(1-\sigma_1)n_2\big)\big]
\end{aligned}
\tag{B.7}
$$

We can follow the same process to obtain

$$
P_{kj}=\begin{cases}
p_m/K+(1-p_m)\Big[(1-\lambda_1)+\lambda_1\big(\alpha+\beta\big(\sigma_1+(1-\sigma_1)n_1\big)\big)\Big] & \text{if } k=j=1\text{, which is the best state} \\
p_m/K+(1-p_m)\big[\lambda_j\big(\alpha+\beta\big(\sigma_1+(1-\sigma_1)n_1\big)\big)\big] & \text{if } k=1 \text{ and } j\neq 1 \\
p_m/K+(1-p_m)\big[\big(1-\lambda_j\big)+\lambda_j\big(\alpha+\beta(1-\sigma_1)n_k\big)\big] & \text{if } k=j\neq 1 \\
p_m/K+(1-p_m)\big[\lambda_j\big(\alpha+\beta(1-\sigma_1)n_k\big)\big] & \text{if } k\neq j, \text{ and } k\neq 1
\end{cases}
\tag{B.8}
$$

Which is equivalent to (18), as desired.

C. Here we derive Theorem 2. Before proceeding with the proof, we establish the preliminary foundation. Time indices and function arguments will usually be suppressed to simplify notation: $m=m(t), m_i=m_i(t), p_{ij}=p_{ij}(m)$. The notation $\Delta m=[\Delta m_i]$ is defined by $\Delta m=m(t+1)-m(t)$. The symbol u indicates the $(K,1)$ vector of ones $[u'=(1,\cdots,1)]$. The ith equation of the interactive Markov model (1) will often be written in the form

$$
\begin{aligned}
\Delta m_i &= \sum_j p_{ij}m_j-m_i=\sum_{j\neq i}p_{ij}m_j-(1-p_{ii})m_i \\
&= \sum_{j\neq i}p_{ij}m_j-\sum_{j\neq i}p_{ji}m_i=\sum_{j\neq i}\big(p_{ij}m_j-p_{ji}m_i\big)
\end{aligned}
\tag{C.1}
$$

where $\sum_j$ means the sum over all j from 1 to K; and $\sum_{j\neq i}$ means the sum over all j from 1 to K except $j=i$.

Next, we formally prove Theorem 2. It follows from the definition of A_0 and (2) that

$$
P(m)=A_0+mb_0 \text{ and } b_0=u'(I-A_0) \tag{C.2}
$$

Namely, $b_0 = \left[b_j \Big/ \left(b_j + \sum_k a_{kj} \right) \right]$ because all the rows of B are the same. Thus the Markov chain can be written as

$$\begin{aligned} m(t+1) &= (A_0 + mb_0)m = \left[A_0 + (b_0 m) I \right] m \\ &= \left[A_0 + \left(u'(I - A_0)m \right) I \right] m = \left[A_0 + (1 - u'A_0 m) I \right] m \end{aligned} \tag{C.3}$$

To find the equilibrium m^*, set $m(t+1) = m = m^*$ in this equation and rearrange terms to get

$$A_0 m^* = \left(u' A_0 m^* \right) m^* \tag{C.4}$$

Thus, an equilibrium $m^* > 0$ for the Markov chain exists if and only if (C.3) has a solution m^* such that $m^* > 0$ with $\sum_k m_k^* = 1$.

Since A_0 is indecomposable by the equations of the interactive Markov model of strategy A, it has a real positive dominant eigenvalue and a corresponding positive eigenvector. We call the root λ and the eigenvector z (normalized so that its elements sum to one). Then we obtain:

$$A_0 z = \lambda z,\ u' A_0 z = \lambda,\ A_0 z = \left(u' A_0 z \right) z \tag{C.5}$$

The first equation is true by the definitions of λ and z; the second equation follows from the first on pre-multiplying by u'; and the third follows from the first two. However, comparison of (C.4) to the third equation of (C.5) shows that $m^* = z$ is a solution to (C.4). Thus, we have an equilibrium $m^* = z$ which is a positive dominant eigenvector of A_0 with eigenvalue

$$\lambda = u' A_0 z = u' A_0 m^* = 1 - u' m^* + u' A_0 m^* = 1 - u'(I - A_0) m^* = 1 - b_0 m^* \tag{C.6}$$

Furthermore, $m^* = z$ is a unique solution to (C.4) since a nonnegative indecomposable matrix cannot have two nonnegative and linearly independent eigenvectors. This completes the proof of Theorem 2.

REFERENCES

[1] C. Ahn, Advances in Evolutionary Algorithms: Theory, Design and Practice. New York: Springer Publishing, 2006.

[2] D. H. Wolpert, and W. G. Macready, "No free lunch theorems for optimization", IEEE Transactions on Evolutionary Computation, vol. 1, no. 1, pp. 67–82, 1997.

[3] C. Reeves, and J. Rowe, Genetic Algorithms: Principles and Perspectives, Norwell, Massachusetts: Kluwer Academic Publisher, 2003, pp. 173–200.

[4] X. Yao, Y. Liu, and G. Lin, "Evolutionary programming made faster", IEEE Transactions on Evolutionary Computation, vol. 3, no. 2, pp. 82–102, 1999.

[5] S. Das and P. N. Suganthan, "Differential evolution – A survey of the state-of-the-art", IEEE Transactions on Evolutionary Computation. vol. 15, no. 1, pp. 4–31, Feb. 2011.

[6] R. Storn and K. V. Price, "Differential Evolution – a simple and efficient heuristic for global optimization over continuous spaces", Journal of Global Optimization, vol. 11, no. 4, pp. 341–359, Dec. 1997.

[7] H. G. Beyer, "Toward a theory of evolution strategies: the (μ, λ)-theory", Evolutionary Computation, vol. 2, no. 4, pp. 381–407, Winter, 1994.

[8] D. Bratton and J. Kennedy, "Defining a standard for particle swarm optimization", in Proceedings of IEEE Swarm Intelligence Symposium, Honolulu, Hawaii, pp. 120–127, 2007.

[9] J. Kennedy, and R. Eberhart, "Particle swarm optimization", in Proceedings of IEEE International Conference on Neural Networks, Perth, WA, pp. 1942–1948. 1995.

[10] H. Ma, "An analysis of the equilibrium of migration models for biogeography-based optimization", Information Sciences, vol. 180, no. 18, pp. 3444–3464, 2010.

[11] D. Simon, "Biogeography-based Optimization", IEEE Transactions on Evolutionary Computation, vol. 12, no. 6, pp. 702–713, 2008.

[12] D. Simon, Evolutionary Optimization Algorithms, John Wiley & Sons, 2013.

[13] A. Nix, and M. Vose, "Modeling genetic algorithms with Markov chains", Annals of Mathematics and Artificial Intelligence, vol. 5, no. 1, pp. 79–88, 1992.

[14] D. Simon, M. Ergezer, D. Du, and R. Rarick, "Markov models for biogeography-based optimization", IEEE Transactions on Systems, Man, and Cybernetics - Part B: Cybernetics, vol. 41, no. 1, pp. 299–306, 2011.

[15] D. Simon, "A dynamic system model of biogeography-based optimization", Applied Soft Computing, vol. 11, pp. 5652–5661, 2011.

[16] J. Suzuki, "A Markov chain analysis on simple genetic algorithms", IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics, vol. 25, no. 4, pp. 655–659, 1995.

[17] P. J. Van Laarhoven, E. H. Aarts, Simulated Annealing: Theory and Applications, New York: Springer Publishing, 1987.

[18] M. D. Vose, The Simple Genetic Algorithm: Foundations and Theory. The MIT Press, 1999.

[19] W. M Spears, Evolutionary Algorithms: The Role of Mutation and Recombination, New York: Springer Publishing, 2000.

[20] J. Conlisk, "Interactive Markov chains", Journal of Mathematical Sociology, vol.4, pp. 157-185, 1976.

[21] Y. Gerchak, "On interactive chains with finite population", Journal of Mathematical Sociology, vol. 9, pp. 255-258, 1983.

[22] D. Simon, R. Rarick, M. Ergezer, and D. Du, "Analytical and numerical comparisons of biogeography-based optimization and genetic algorithms", Information Sciences, vol. 181, no. 7, pp. 1224–1248, 2011.

[23] J. Dieterich and B. Hartke, "Empirical review of standard benchmark functions using evolutionary global optimization", http://arxiv.org/abs/1207.4318.

[24] H. Ma, D. Simon, M. Fei, and Z. Chen, "On the Equivalences and Differences of Evolutionary Algorithms", Engineering Applications of Artificial Intelligence, vol. 26, no. 10, pp. 2397-2407, 2013.

[25] H. Ma, D. Simon, M. Fei, and H. Mo, "Interactive Markov Models of Evolutionary Algorithms (web site)", http://academic.csuohio.edu/simond/InteractiveMarkov, November 2013.

[26] J. He, and X. Yao, "Towards an analytic framework for analysing the computation time of evolutionary algorithms", Artificial Intelligence, vol. 145, pp. 59–97, 2003.

[27] K. De Jong, W. Spears, and D. Gordon, "Using Markov chains to analyze GAFOs," Foundations of Genetic Algorithms, vol. 3 (L. Whitley and M. Vose, editors), Morgan Kaufmann, pp. 115-138, 1995.

[28] J. He, F. He, and X. Yao, "A unified Markov chain approach to analysing randomized search heuristics," arXiv:1312.2368 , 2013.

Haiping Ma received the M.Sc. degree in Control Engineering from Taiyuan University of Technology. He is a Ph.D. candidate with the Shanghai Key Laboratory of Power Station Automation Technology, School of Mechatronic Engineering and Automation, Shanghai University. His main research interests include evolutionary computation, information fusion, intelligent control and signal processing. He has published 15 papers on evolutionary algorithms.

Dan Simon received his B.S. from Arizona State University, his M.S. from the University of Washington, and his Ph.D. from Syracuse University, all in electrical engineering. Before joining academia, he had 14 years of experience in various engineering industries, including aerospace, automotive, biomedical, process control, and software engineering. He continues his relationship with industry by teaching short curses and through regular consulting. He joined Cleveland State University in 1999 and has been a Full Professor in the Electrical and Computer Engineering Department since 2008. His teaching and research interests include control theory, computer intelligence, and embedded systems. He is an associate editor for the journals Aerospace Science and Technology, Mathematical Problems in Engineering, and International Journal of Swarm Intelligence. His research has been funded by the NASA Glenn Research Center, the Cleveland Clinic, the National Science Foundation, and several industrial organizations. He has written over 100 refereed publications, and is the author of the textbooks Optimal State Estimation (John Wiley & Sons, 2006) and Evolutionary Optimization Algorithms (John Wiley & Sons, 2013).

Hongwei Mo received his BS and PhD degrees from Automation College of Harbin Engineering University. He is presently a professor of Automation College of Harbin Engineering University. He was a visiting Scholar of UC Davis (California, USA) from 2003-2004. His main research interests include natural computing, artificial immune systems, data mining, intelligent systems, and artificial intelligence. He has published 30 papers and two books on AIS. He is a director at the Biomedicine Engineering Academy of Heilongjiang Province, commissioner of the China Neural Network Committee, and a senior member of the Computer Academy of China. He is secretary-general chairman and associate chairman of the organizing committee of the 16th China Neural Network Conference and 1st Conference of Special Topic on Artificial Immune Systems, a member of the program committee of the 2nd International Conference on Natural Computing, Fuzzy Systems, and Knowledge Discovery (ICNC-FSKD2006), 1st International Conference on Rough Sets and Knowledge Discovery, 6th International Conference on Simulation Learning and Evolution, 13th IEEE International Conference on Mechanics and Automation(ICMA2007), Biology Inspired Computing 2008, and numerous other conferences. He served as a member of the editorial board for the International Journal on Information Technology Research.