2015

Interactive Markov Models of Evolutionary Algorithms

Haiping Ma
Shanghai University

Daniel J. Simon
Cleveland State University, d.j.simon@csuohio.edu

Minrui Fei
Shanghai University

Hongwei Mo
Harbin Engineering University

Interactive Markov Models of Optimization Search Strategies

Haiping Ma, Dan Simon, *Senior Member, IEEE*, Minrui Fei, and Hongwei Mo

***Abstract*—This paper introduces a Markov model for evolutionary algorithms (EAs) 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 discrete optimization search strategies with population-proportion-based selection and a modified mutation operator. The probability of selection 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 optimization search strategies with interactive Markov models. We present simulation results to confirm the interactive Markov model theory. We show that genetic algorithms and biogeography-based optimization perform better with the addition of population-proportion-based selection on a set of real-world benchmarks. We note that many other EAs, both new and old, might be able to be improved with this addition, or modeled with this method.**

***Index Terms*—Evolutionary algorithm (EA), interactive Markov model, Markov model, optimization search strategy, population-proportion-based selection.**

I. Introduction

EVOLUTIONARY algorithms (EAs) and swarm intelligence (SI) algorithms have received much attention over the past few decades due to their ability as global optimization search methods for real-world applications [1]–[3]. Some popular EAs include the genetic algorithm (GA) [4], [5], evolutionary programming [6], differential evolution (DE) [7]–[10], evolution strategy (ES) [11], biogeography-based optimization (BBO) [12], [13]. SI algorithms include ant colony optimization (ACO) and particle swarm optimization (PSO) [14]–[16]. Inspired by natural processes, EAs and SI 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 and SI algorithm works on 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, ACO is based on the behavior of ants seeking food, and BBO is based on migration behavior. EAs all have certain features in common and probabilistically share information between candidate solutions to improve solution fitness. 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 multiobjective problems [17].

A. 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 a mathematical model could predict the improvement

Manuscript received June 23, 2015; accepted September 1, 2015. This work was supported in part by the National Science Foundation (NSF) under Grant 1344954, in part by the National Natural Science Foundation of China under Grant 61305078 and Grant 61533010, and in part by the Shaoxing City Public Technology Applied Research Project under Grant 2013B70004. The work of D. Simon was supported in part by the NASA Glenn Research Center, in part by the Cleveland Clinic, in part by the NSF, and in part by the several industrial organizations. The work of H. Ma and M. Fei was supported in part by the NSF, and in part by several industrial organizations. This paper was recommended by Associate Editor M. K. Tiwari.

H. Ma was with the Department of Electrical Engineering, Shaoxing University, Shaoxing 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).

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

M. 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).

H. Mo is with the Department of Automation, Harbin Engineering University, Harbin 150001, China (e-mail: mhonwei@sina.com).

Digital Object Identifier 10.1109/TSMC.2015.2507588

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 perform long, tedious, expensive physical experiments to evaluate fitness. If we can find a model that reduces the number of required 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 [4]. It has several disadvantages, including the fact that it is only approximate. Perhaps the most widely developed EA model is based on Markov theory [18]–[20], which has been a valuable theoretical tool that has been applied to several EAs, including GAs [21] and simulated annealing [22]. Infinite population size Markov models are discussed in detail in [23], and exact finite population size Markov models are discussed in [24].

In Markov models, 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.

B. *Overview of the Paper*

The first goal of this paper is to present a Markov model of EAs which 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 approach is extremely limiting and leads 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 Markov model. This method can lead to a better understanding of EA behavior on problems with more realistic sizes.

The standard Markov model has a state space whose dimension grows factorially with search space cardinality and population size, while the interactive Markov model presented here has a state space whose dimension is independent of population size and grows linearly with the cardinality of the search space. The interactive Markov model presented here is limited to EAs with a discrete search space and is still limited to relatively small problems, but is not nearly as limited as the standard Markov model, as we will see in Section II.

The second goal of this paper is to propose two simple optimization strategies that use population-based selection probabilities. We call these strategies "population-proportion-based EAs." We use a modified mutation operator in the EAs. The modified mutation operator randomly modifies (replaces) an entire individual rather than modifying a single decision variable. We exactly model these EAs with interactive Markov models. We confirm the interactive Markov model with simulation on a set of test problems. We then empirically compare standard GA and BBO with population-proportion-based GA and BBO on a set of real-world benchmarks to show the performance improvement that is possible.

Section II introduces the preliminary foundations of interactive Markov models, presents two simple EAs, models them using an interactive Markov model, uses interactive Markov model theory to analyze their convergence, and discusses their computational complexity. Section III discusses the similarities and differences between our two simple EAs and standard, well-established EAs, including GA and BBO. Section IV explores the performance of the proposed optimization search strategies using both Markov theory and simulations, and shows that the inclusion of population-proportion-based selection significantly improves both GA and BBO performance on real-world benchmarks. 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 I.

II. Interactive Markov Models

Section II-A presents the foundation for interactive Markov models. Section II-B presents two proportion-selection-based EAs, and Section II-C discusses their convergence properties. Section II-D analyzes the interactive Markov model transition matrix of the two EAs, and Section II-E discusses their computational complexities.

A. *Fundamentals of Interactive Markov Models*

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

TABLE I
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 individuals that are allowed to change, or *movers* (Strategy B)
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	ith point in the search space ($i = 1, \ldots, K$)
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): $\sigma = [\sigma_k]$

space, denotes as $\{x_1, x_2, \ldots, x_K\}$. Suppose $m(t) = [m_i(t)]$ 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 $P = [p_{ij}]$ do not depend on the population distribution $m(t)$.

Interactive Markov models deal with cases, where P is a function of $m(t)$, so transition probabilities are functions of the Markov states. P is written as a function of $m(t)$; that is, $P = P[m(t)]$, and the Markov model is interactive

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

A standard Markov model with constant P can model an EA. 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, and we can define a $T \times T$ transition matrix that contains transition probabilities between each population distribution [19]. 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 problem with $K = 100$ and $N = 20$, T is on the order of 10^{22}. For a larger problem with $K = 200$ and $N = 30$, T is on the order of 10^{37}.

In contrast, interactive Markov models in the form of (1) are tractable for 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. The challenge that we address in this paper is how to define the interactive Markov model for an 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. The standard Markov model transition matrix is larger (T states) but with a simple form, while the interactive Markov model transition matrix is smaller (K states) but with a more complicated form. Both the standard and the interactive Markov models are exact; no approximations are involved. However, due to their fundamentally different approaches to the definition of state, neither one is a subset of the other (in general), so neither one can be derived from the other.

For standard Markov models, many theorems exist for stability and convergence. But for interactive Markov models such results do not come easily because of the immense variety of possible forms of $P(\cdot)$. Conlisk [25] discussed a particular class of $P(\cdot)$ defined as follows for a K-state system:

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

where the summations go from 1 to K, and $m(t) = [m_i(t)]$ is abbreviated to $m = [m_i]$. Values for the matrices $A = [a_{ij}]$ and $B = [b_{ij}]$ specify the interactive Markov model. P is the transition matrix of the interactive model because it defines the transition probabilities of (1). Matrices A and B in (2) are intermediate matrices used to simply the expression for P.

Equation (2) specifies the transition matrix $P(m)$, 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 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 a measure of the attractive power of the ith state. In (2), if $b_{ij} > 0$ then crowding in the ith state makes it more attractive, and if $b_{ij} < 0$ then crowding in the ith state makes it less attractive.

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

B. Optimization Search Strategies

In this section, 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 EA adaptation, which we call strategy A, is based on a social process model [25, Example 1]. Strategy A involves the replacement of individuals with

TABLE II
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; That Is, the Fraction of the Population That is Optimal

		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	$\alpha = 0.25$	0.9208	0.9237	0.8725	0.8701	0.7844	0.7836	0.2047	0.2018
	$\alpha = 0.50$	0.9208	0.9205	0.8778	0.8749	0.7887	0.7859	0.2185	0.2192
	$\alpha = 0.75$	0.9208	0.9231	0.8790	0.8775	0.7908	0.7932	0.2869	0.2850
Ave. CPU time (s)		—	81.7	—	83.4	—	86.2	—	89.3

randomly-selected individuals. Strategy A does not include recombination or mutation, although it could be modified to include these features. The population of strategy A evolves according to the following two rules.

1) Denote $\alpha \in [0, 1]$ as the replacement pool probability. Randomly choose round(αN) individuals, where N is the population size. Denote $\lambda_j \in [0, 1]$ as the modification probability. λ_j is typically chosen as a decreasing function of fitness; that is, good individuals should have a smaller probability of replacement than poor individuals.
2) The jth individual chosen in the previous step, 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 $\{x_i : i = 1, \ldots, K\}$ is the search space). The probability of selecting the ith individual x_i as a replacement is denoted as μ_i and is composed of two parts: a) $\beta \in [0, 1]$ is assigned to each individual equally and b) the remaining probability $1-\beta$ is assigned among the x_i individuals in the proportions $(m_1, \ldots, m_K)$, where m_i is the proportion of the x_i individuals in the population; that is, m_i is the number of x_i individuals in the population divided by the population size. So the probability of selecting x_i as a replacement is $\mu_i = \beta/K + (1-\beta)m_i$. Note that $\sum_{i=1}^{K} \mu_i = 1$. If the selection probability is independent of m_i, that is, $\beta = 1$, then the probability of selecting x_i as a replacement is $\mu_i = 1/K$ for all i.

Algorithm 1 shows an EA that operates according to strategy A. We see from Algorithm 1 that strategy A has four tuning parameters: 1) N (population size); 2) α (replacement pool probability); 3) β (the constant component of the selection probability); and 4) λ (modification probability, which is a function of fitness).

Note that strategy A (Algorithm 1) is not an EA when $\beta = 1$; in this case the algorithm is equivalent to a Monte Carlo search strategy in which random candidate

Algorithm 1 EA Based on Strategy A. α and β are Tuning Parameters in the Range [0, 1], λ is a Decreasing Function of Fitness, and K is the Cardinality of the Search Space. The Interactive Markov Model for this Algorithm will be Shown in Section II-D1. The Equilibrium Population of this Algorithm will be Established in Theorem 1

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

solutions are evaluated. As $\beta \to 0$ the algorithm becomes more and more evolutionary in the sense that individual fitness implicitly drives the algorithm more strongly. Tables II–V show that strategy A performs better as $\beta \to 0$.

2) Strategy B: The second EA adaptation that we introduce, which we call strategy B, is based on a social process model [25, Example 6]. Strategy B is similar to strategy A in its selection of random individuals from the search space, and the replacement of individuals with those randomly-selected individuals. However, strategy B includes elites. The population of strategy B evolves according to the following two rules.

1) 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.
2) 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 a probability

TABLE III
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; THAT IS, THE FRACTION OF THE POPULATION THAT IS OPTIMAL

		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	$\alpha = 0.25$	0.7512	0.7583	0.7275	0.7218	0.5475	0.5439	0.0425	0.0431
	$\alpha = 0.50$	0.7512	0.7537	0.7386	0.7304	0.6492	0.6410	0.0936	0.0903
	$\alpha = 0.75$	0.7512	0.7584	0.7449	0.7482	0.6851	0.6823	0.1701	0.1722
Ave. CPU time (s)		–	82.9	–	85.1	–	87.2	–	91.0

TABLE IV
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; THAT IS, THE FRACTION OF THE POPULATION THAT IS OPTIMAL

		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	$\alpha = 0.25$	0.3752	0.3719	0.3451	0.3412	0.2145	0.2102	0.0396	0.0375
	$\alpha = 0.50$	0.3752	0.3740	0.3566	0.3530	0.2678	0.2631	0.0513	0.0524
	$\alpha = 0.75$	0.3752	0.3714	0.3670	0.3613	0.2912	0.2900	0.0744	0.0718
Ave. CPU time (s)		–	88.6	–	90.4	–	93.5	–	97.6

proportional to $\alpha + \beta m_i$, where m_i is the fraction of the population that is comprised of x_i individuals. Recall that $\{x_i : i = 1, \ldots, K\}$ 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. Algorithm 2 illustrates an EA that operates according to strategy B. We see from Algorithm 2 that strategy B has the same four tuning parameters as strategy A: 1) N (population size); 2) α (replacement pool probability); 3) β (selection constant); and 4) λ (modification probability, which is a function of fitness). However, note that α and β are used differently in strategies A and B (that is, in Algorithms 1 and 2).

Note that strategy B (Algorithm 2) is purely random when $\beta = 0$. In this case the only role of the fitness values is to decide which individuals to replace. The fitness values do not guide the formation of new individuals since the individuals that are kept are essentially useless. The same outcome would be provided with a population of a single individual being replaced every generation by a randomly chosen individual from the search space. As $\beta \rightarrow 1$ the algorithm becomes more and more evolutionary in that individual fitness implicitly drives the algorithm more strongly. Later in the paper, Table VI will show that strategy B performs better as $\beta \rightarrow 1$.

The selection probability of x_i in strategy B is

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

for $i \in [1, K]$. Equation (3) is 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 $k \in [1, N]$ for which y_k is selected for replacement.

3) Selection Pressure in Strategies A and B: The selection probabilities in strategies A and B are both linear with respect to the fraction of x_i individuals in the population. Fig. 1 depicts the selection probability.

TABLE V

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; That Is, the Fraction of the Population That Is Optimal

		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	$\alpha = 0.25$	0.2845	0.2819	0.2830	0.2812	0.2801	0.2744	0.0419	0.0420
	$\alpha = 0.50$	0.2845	0.2842	0.2826	0.2810	0.2787	0.2710	0.0486	0.0475
	$\alpha = 0.75$	0.2845	0.2837	0.2819	0.2805	0.2756	0.2705	0.0527	0.0571
Ave. CPU time (s)		–	71.2	–	73.3	–	75.9	–	78.4

Algorithm 2 EA Based on Strategy B. α and β Are Tuning Parameters in the Range [0, 1], λ is a Decreasing Function of Fitness, and K is the Cardinality of the Search Space. The Interactive Markov Model for This Algorithm Will be Shown in Section II-D2. The Equilibrium Population of this Algorithm will be Established in Theorem 2.

```
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 replacement probability $\lambda_k$ to decide whether
            to replace $y_k$
            If replacing $y_k$ then
                Select one of the $x_i (i = 1, \cdots, K)$, each
                with probability $[\alpha + \beta m_i]$
                $y_k \leftarrow x_i$
            End replacement
        End if
    Next individual: $k \leftarrow k + 1$
Next generation
```

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

If selection is overly-biased toward populous individuals, the EA may converge quickly to a uniform population while not widely exploring the search space. If selection is not biased strongly toward populous individuals, the population will be more scattered with fewer 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 in our results in Section IV.

A useful metric for quantifying the difference between various selection methods is selection pressure ϕ, which is

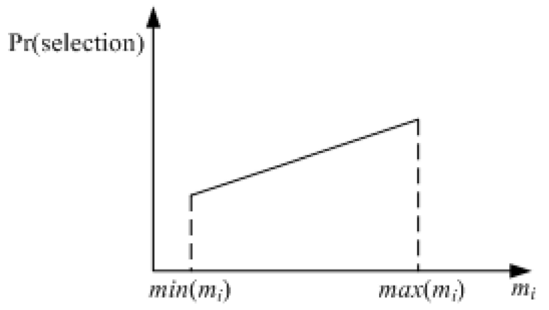

Fig. 1. Selection probability in strategies A and B. The min and max operators are taken over $i \in [1, K]$, where K is the cardinality of the search space.

defined as follows [4, p. 34]:

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

where Pr(selection) is the probability that an individual is selected to replace another individual.

The most populous individual has the fraction $\max(m_i)$ individuals, which we denote as $m_{\max}$. Since selection probability is affine (Fig. 1), average selection probability is an affine function of $(\max(m_i) + \min(m_i))/2 = m_{\max}/2$, assuming $K > N$, as discussed earlier. In this case the selection pressure of strategy B can be calculated from (4) as

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

If we normalize the selection probabilities to sum to 1, we get

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

TABLE VI
STRATEGY B RESULTS FOR TEST PROBLEM f_1. THE NUMBERS IN THE TABLE SHOW THE PROPORTION OF OPTIMAL INDIVIDUALS IN THE POPULATION; THAT IS, THE FRACTION OF THE POPULATION THAT IS OPTIMAL

		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

where we used the fact that $\sum_{i=1}^{K} m_i = 1$ since the sum of the fractions must equal 1. If we desire a given selection pressure we can solve (5) and (6) for α and β to obtain

$$\alpha = \frac{m_{\max}(2-\phi)}{Km_{\max}(2-\phi)+2(\phi-1)}$$
$$\beta = \frac{2(\phi-1)}{Km_{\max}(2-\phi)+2(\phi-1)}. \quad (7)$$

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

We begin this section with a preliminary theorem of 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 $\prod_{t=1}^{R} P[m(t)]$ is positive definite for all $m(t) \geq 0$ such that $\sum_{i=1}^{K} m_i(t) = 1$, where $t = 1, 2, \ldots, R$, then $\prod_{t=1}^{R} P[m(t)]$ converges as $R \to 8$ to a steady state which has all nonzero entries.

Proof: See [26] 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 strategies A and B. We will also show that the probability of each state of the model is nonzero at all generations after the first one. 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 a discrete optimization problem.

D. Interactive Markov Model Transition Matrices

The previous sections presented two simple EA adaptations 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 explicit solutions to the steady-state population distributions. Firstly, we discuss strategy A.

1) Interactive Markov Model for Strategy A:

a) Selection: We can use [25, Example 1] to obtain the following interactive Markov model of strategy A:

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

for $(i, j) \in [1, K]$. p_{ij} is 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.

1) The first term, $1-\alpha$, is the probability that the individual is not selected for the replacement pool.
2) 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 $(1-\lambda_j)$.
3) 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 $(\beta/K + (1-\beta)m_j)$.

The second equality in (8), when $i \neq j$, is the probability that an individual is changed from one generation to the next. This probability is similar to the second part of the second term of the first equality as discussed in 3) above, the difference being that m_i is used instead of m_j because the selected individual is changed from x_j to x_i.

b) Mutation: Next we add mutation to the interactive Markov model of strategy A. Typically EA mutation is implemented by probabilistically complementing each bit in each individual. Then the probability that x_i mutates to x_k can be written as

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

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

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

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

But with single-bit 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 replaces an entire individual rather than a single bit of an individual. Mutation of the jth individual is implemented as follows.

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

In the above mutation (replacement) logic, rand(L, U) is a uniformly distributed random number between L and U, and L and U are the lower and upper bounds of the search space. The above logic replaces each individual with probability p_m. The descriptions of strategies A and B with mutation are the same as Algorithms 1 and 2 except that we add the above replacement logic at the end of each generation. Note that replacement acts equally on all individuals $Y = \{y_k : k = 1, \ldots, N\}$. In strategy B, we use elitism to prevent selection but not to prevent replacement.

The transition probability of the modified mutation (replacement) operator is described as follows:

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

Then the transition probability of strategy A with mutation, and the corresponding A and B matrices in (2) can be written as (11) and (12), shown at the top of this page.

The derivation of (11) and (12) is in the Appendix. Now we are in a position to state the main result of this section.

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 $A_0 = a_{kj}/(b_j + \sum_{i=1}^{K} a_{ij})$, where a_{kj} is given by (12), and b_j is shorthand notation for b_{kj} in (12) since all the rows of B are the same.

Proof: This theorem derives from [25, Th. 1]. We provide the proof in the Appendix. ■

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

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

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 [25, Th. 1] or standard noninteractive Markov theory [18] 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. In an EA, the stayers are comprised of a user-specified fraction of the best individuals in the population. Vector $n = [n_1, \ldots, n_K]$ consists of the fractions of all individuals that are allowed to change, and we call these individuals movers. $m = [m_1, \ldots, m_K]$ is the fraction of all individuals in the population. The fraction of x_j individuals thus includes two parts: 1) σ_j, which is the fraction of all x_j individuals that are not allowed to change and 2) $(1 - \sum_{k=1}^{K} \sigma_k)n_j$, which is the fraction of all x_j individuals that may change in future generations. So the fraction vector m is given by

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

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-enough individuals from changing in future generations [17]. 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

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

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

Strategy B elitism is a little different than standard EA elitism. In strategy B we assume that we know whether or not an individual has acceptable performance in the judgment of the decision maker. This is not always the case in practice, but it may be the case for certain problems. For example, suppose we are trying to solve a state estimation problem. In this case, the Cramer–Rao bound gives a lower bound on the estimation error, along with conditions that tell us if such an estimator exists, but does not tell us what estimator achieves the lower bound [27]. An engineer may also know ahead of time the acceptable performance for an estimator, regardless of the Cramer–Rao bound. The same situations arise with the performance bounds and design of H_2 and H_∞ controllers [28]. As another example, if the fitness of a product design is measured by the qualitative judgments of test subjects, we might know ahead of time that a rating of 9 out of 10 is acceptably optimal, although we might not know how to achieve that rating.

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 $n_1 = N_1/(N-S)$. Multiple individuals at the global optimum could mean that there are duplicate individuals in the population, or it could mean that there are multiple solutions to the optimization problem. If an individual is not globally optimal, then we must always allow it to change—that is, we can implement elitism only for individuals that are globally optimal. Even if we have globally optimal individuals in the population, we may want to continue the search. This is because other solutions near (but not at) the global optimum may have desirable properties. To return to our previous example, an estimator with an error larger than the lower bound may be more desirable than one at the lower bound if the first estimator is more robust. Similarly, a product rating less than perfect might be more desirable than one with a perfect rating because of other factors.

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:

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

Therefore:

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

In this example, we have assumed that the engineer has specified that the best 2/14 of the population will be stayers and that the rest of the population will be movers. 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 this example.

a) Summary of the interactive Markov model for strategy B: Now we use [25, Example 6], combined with (6) above, to obtain the following interactive Markov model of strategy B selection.

Equation (16), as shown at the top of this page, can be explained as follows. For the first equality, when $i = j = 1$, which is the best state, the transition probability includes two parts: 1) the first term, which denotes the probability that the individuals is not changed $(1-\lambda_1)$ and 2) 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 $\lambda_1(\mu_1/\sum_{k=1}^K \mu_k)$. This second term can be written as

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

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 second 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 shown in (2) can be derived as shown in the Appendix and can be written as (18) and (19), shown at the top of next page.

Theorem 3: Assume that the search space has a single global optimum. Then the $K \times 1$ 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 $A_0 = a_{kj}/(b_j + \sum_{i=1}^K a_{ij})$, where a_{kj} is given by (19), and b_j is

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

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

shorthand notation for b_{kj} 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, which is computed for each individual in the population once per generation. Therefore, the computational costs of Algorithms 1 and 2 are of the same order 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 of the order K^2, but that computational cost can be greatly reduced by using linear ranking [17, Sec. 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 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 of the order K^3, where K is the order of the matrix and is equal to the cardinality of the search space.

In summary, using the interactive Markov chain model to calculate an equilibrium population requires three steps: 1) calculation of the transition matrix components, as shown in (12) for Algorithm 1 and (19) for Algorithm 2; 2) formation of a certain matrix, as shown in Theorem 2 for Algorithm 1 and Theorem 3 for Algorithm 2; and 3) 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 of the order K^3.

The above analysis is a per-generation analysis. In practice, an EA runs for a certain number of generations before termination. Per-generation run time is not meaningful unless a convergence rate can be established. Section II-C guarantees convergence for Algorithms 1 and 2, but does not establish the convergence rate. Future work could explore the convergence rate of the algorithms, perhaps using Markov process-based approaches [29] or run-time analysis [30].

III. Similarities of EAs

In this section, we first compare strategies A and B as shown in Algorithms 1 and 2. Note that we only consider selection here. If the replacement pool probability $\alpha = 1$ in strategy A, then it reduces to strategy B if elitism is not used ($\sigma_1 = 0$). 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 GA with global uniform recombination (GA/GUR) and elitism with no mutation as described in [31, Fig. 3]. Strategy B and GA/GUR are equivalent under the following circumstances.

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

This implementation of GA/GUR is also called the Holland algorithm [32] and is shown in Algorithm 3. Note that some researchers would not consider GA/GUR as a genuine GA since it does not include classical crossover. It might instead be more correctly referred to as "EA/GUR." However, we retain the GA/GUR terminology here for consistency with previous work [31].

Next we discuss the equivalence of strategy B and BBO with elitism and no mutation [13]. BBO is an EA that is inspired by the migration of species between islands and is described by [31, Fig. 2]. 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

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 $\lambda_k = 1$ for all k.

```
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 population proportions to probabilistically select $x_i$, $i \in [1, K]$
            $y_k \leftarrow x_i$
        End if
    Next individual: $k \leftarrow k+1$
Next generation
```

Algorithm 4 Biogeography-Based Optimization (BBO) 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 individual $y_k$
    If $y_k$ is not the best individual in the search space then
        Use $\lambda_k$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
```

becomes equivalent to EA strategy B. Algorithm 4 shows this modified version of BBO.

There are many other EAs that are equivalent to each other under special circumstances [33] and so they may also be equivalent to strategies A or B under certain conditions. We leave this study to future research.

IV. Simulation Results and Comparisons

In the following, Sections IV-A and IV-B confirm the interactive Markov model theory of Section II with simulation results, and investigate the effects of tuning parameters on strategies A and B. Section IV-C compares population-proportional selection with two standard EAs (GA and BBO) on a set of real-world benchmark problems.

A. Strategy A—Confirmation of Interactive Markov Model

In strategy A (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 how many individuals are equal to the optimum to compare performances with different tuning parameters. A larger fraction m_b^* indicates better performance.

The first three test functions have a search space cardinality of $K = 8$; that is, they are three-bit problems with fitness functions that correspond to bit strings (0 0 0), (0 0 1), (0 1 0), (0 1 1), (1 0 0), (1 0 1), (1 1 0), and (1 1 1). These three fitness functions are given as follows:

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

These functions are chosen as representative functions because f_1 is a unimodal one-max problem, f_2 is a multimodal problem, and f_3 is a deceptive problem. Note that all three of these problems are maximization problems.

We also use the multimodal Ackley function with 20 dimensions to confirm the interactive Markov model of strategy A. The Ackley function is described as follows:

$$\begin{aligned} f_4 = &-20\exp\left(-0.2\sqrt{\frac{\sum_{i=1}^n x_i^2}{n}}\right) \\ &- \exp\left(\frac{\sum_{i=1}^n \cos(2\pi x_i)}{n}\right) + 20 + e, \quad -30 \le x_i \le 30. \end{aligned} \tag{21}$$

The granularity or precision of each independent variable is set to 1 here to allow for experimental confirmation of the interactive Markov model theory while still providing for reasonable computational effort. Note that the Ackley function is a minimization problem.

We use $\alpha = 0.25, 0.50, 0.75$ and $\beta = 0.25, 0.50, 0.75$, with modification probabilities $\lambda_i = 1-\text{fitness}(x_i)$, where fitness$(x_i)$ denotes the fitness value of individual x_i, which is normalized to the range $[0, 1]$. We use population size = 50, generation limit = 20 000, and 30 Monte Carlo runs for each test. Tables II–IV show comparisons between theoretical interactive Markov results (Theorem 2 in Section II) and strategy A simulation results. Table V shows similar comparisons for strategy A for the special case $\beta = 1$, which gives selection probability $\mu_i = 1/K$ for all i, independent of population fractions. The results in Tables II–V can be reproduced with MATLAB code that is available at the authors' Web site [34].

The numbers in the tables show the proportion of optimal individuals in the population. For example, the optimal individual for function f_1 is [1, 1, 1]. The first entry in Table II (α = 0.25, no mutation) shows the number 0.8155, which means that 81.55% of the population is comprised of the bit string [1, 1, 1].

Note that the purpose of Tables II–V is not to show the superiority of one algorithm over another, or of one set of tuning parameters over another. The purpose is rather to show that simulation results confirm the theoretical Markov results.

Note first from Tables II−V 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 [25, p. 162]. However, we see that α can slightly affect the proportion of individuals in the case of nonzero mutation.

Second, for a given replacement pool probability α and parameter β, the proportion of optimal individuals decreases with increasing p_m. This indicates that low mutation rates have better performance. 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 replacement pool probability α and 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 II–V 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 II–V show the simulation times for the four test problems. Strategy A runs faster with smaller mutation rates. The reason is that larger mutation rates require more mutations, which slightly increases computation time. However, in real-world problems, computational effort is dominated by fitness function evaluation, which is independent of the mutation rate.

B. Strategy B—Confirmation of Interactive Markov Model

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 (Theorem 3 assumes that the optimization problem is unimodal). Recall that EA selection pressure is constrained to the domain $\phi \in (1, 2)$ [4, p. 34]. In this section, we test $\phi = 1.25, 1.50, 1.75$ in (7) to compute parameters α, β of the 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 in the previous section. Table VI shows comparisons between interactive Markov theory results and strategy B simulation results. The results in Table VI can be reproduced with MATLAB code that is available at the authors' Web site [34].

Note first from Table VI that 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.

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

TABLE VII
REAL-WORLD BENCHMARK RESULTS. THE "/P" VERSION OF EACH ALGORITHM IS THE POPULATION-PROPORTION-SELECTION MODIFICATION OF THE ALGORITHM. THE NUMBERS SHOW THE MEAN VALUE OF 25 INDEPENDENT SIMULATIONS. THE BEST RESULT IN EACH ROW IS SHOWN IN BOLD FONT

Problem	GA	GA/P	BBO	BBO/P
P01	9.19E−05	**0.00E+00**	7.35E−17	**0.00E+00**
P02	−1.64E+01	−2.11E+01	−2.83E+01	**−2.97E+01**
P03	**1.15E−05**	**1.15E−05**	**1.15E−05**	**1.15E−05**
P04	3.04E+01	**1.30E+01**	1.43E+01	1.37E+01
P05	−1.37E+01	−3.45E+01	−9.20E+00	**−3.55E+01**
P06	−1.62E+01	−2.10E+01	−2.92E+01	**−2.97E+01**
P07	9.47E−01	**8.08E−01**	9.71E−01	9.28E−01
P08	**2.20E+02**	**2.20E+02**	**2.20E+02**	**2.20E+02**
P09	1.59E+03	**−1.35E+01**	1.04E+03	−6.27E+00
P10	−1.18E+01	−1.64E+01	**−2.18E+01**	−1.74E+01
P11.1	5.78E+05	5.92E+04	9.25E+04	**5.61E+04**
P11.2	6.24E+06	3.27E+06	**1.05E+06**	5.78E+06
P11.3	1.56E+04	**1.53E+04**	1.54E+04	1.54E+04
P11.4	7.26E+04	1.82E+04	8.89E+04	**1.79E+04**
P11.5	7.05E+04	**3.25E+04**	6.29E+04	5.11E+04
P11.6	3.47E+05	1.86E+05	3.32E+05	**1.49E+05**
P11.7	4.85E+06	2.15E+06	**1.91E+06**	2.10E+06
P11.8	1.88E+06	**1.06E+05**	9.23E+05	1.93E+06
P11.9	3.50E+06	**1.14E+05**	9.30E+05	1.16E+06
P11.10	2.38E+06	1.12E+06	**9.24E+05**	1.09E+06
P12	1.58E+01	**1.49E+01**	1.64E+01	1.50E+01
P13	**8.77E+00**	1.31E+01	1.43E+01	1.33E+01

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

C. Comparisons on Real-World Benchmarks

This section uses real-world optimization problems from the 2011 IEEE Congress on Evolutionary Computation (CEC) [35] to compare population-proportional selection GA and BBO (Algorithms 3 and 4) with standard GA and BBO. For the modified EAs with population-proportion-based selection, we use selection probability $(\alpha+\beta m_i)$, where m_i denotes the fraction of x_i individuals in the population. For the GAs we use real coding, roulette wheel selection, single-point crossover with a crossover probability of 1, and a mutation probability of 0.001. For BBO we use maximum immigration and emigration rates of 1, linear migration curves [13], and a mutation probability of 0. Each algorithm uses a population size of 50, an elitism size of 2, and a fitness function evaluation limit of 100 000.

Algorithm 3 is equivalent to a GA with population-proportional selection (GA/P), and Algorithm 4 is equivalent to BBO with population-proportional selection (BBO/P). These equivalences motivate us to compare GA with GA/P, and BBO with BBO/P. The only difference between standard GA and GA/P in this section is that standard GA uses fitness-based selection while GA/P uses population-based selection.

TABLE VIII
WILCOXON TEST RESULTS FOR PAIR-WISE ALGORITHM COMPARISONS. IF THE DIFFERENCE BETWEEN THE ALGORITHMS IS STATISTICALLY SIGNIFICANT, THE PAIRS ARE MARKED AS FOLLOWS: "X-O" SHOWS THAT THE LEFT ALGORITHM IS BETTER THAN THE RIGHT ONE; "O-X" SHOWS THAT THE RIGHT ALGORITHM IS BETTER THAN THE LEFT ONE. THE B/S/W ROW AT THE BOTTOM SHOWS THE TOTAL SCORES, WHERE "B" DENOTES THE NUMBER OF TIMES THE LEFT ALGORITHM PERFORMS BETTER, "S" DENOTES THE NUMBER OF TIMES THE TWO ALGORITHMS PERFORM THE SAME, AND "W" DENOTES THE NUMBER OF TIMES THE LEFT ALGORITHM PERFORMS WORSE THAN THE RIGHT ONE

Problem	GA vs. GA/P	BBO vs. BBO/P
P01	o-x	o-x
P02	o-x	o-x
P03	–	–
P04	o-x	o-x
P05	o-x	o-x
P06	o-x	–
P07	o-x	–
P08	–	–
P09	o-x	o-x
P10	o-x	x-o
P11.1	o-x	o-x
P11.2	o-x	x-o
P11.3	–	–
P11.4	o-x	o-x
P11.5	o-x	o-x
P11.6	o-x	o-x
P11.7	o-x	–
P11.8	o-x	x-o
P11.9	o-x	x-o
P11.10	o-x	x-o
P12	–	o-x
P13	x-o	o-x
B/S/W	1/4/17	5/6/11

Similarly, the only difference between standard BBO and BBO/P in this section is that standard BBO uses fitness-based selection while BBO/P uses population-based selection.

Table VII summarizes the performances of the EAs on the real-world optimization problems. All results are averages of 25 independent simulations. The performance data in Table VII for standard GA and BBO is taken from [33]. Computational time is the same for standard GA and GA/P, and is the same for standard BBO and BBO/P. This is because the algorithms execute identically except for the difference between fitness-based selection and population-based selection.

We use the Wilcoxon method to test for statistical significance [36]. The Wilcoxon test results are shown in Table VIII, where a pair is marked if the difference between the pair of algorithms is statistically significant.

The results in Table VIII are divided into the GA versus GA/P group, and the BBO versus BBO/P group. For each pair of algorithms we calculate B/S/W scores, where "B" denotes the number of times that the left algorithm performs better than the right one, "S" denotes the number of times that the left algorithm performs statistically the same as the right, and "W" denotes the number of times that the left algorithm performs worse than the right one.

Table VIII shows that for GA versus GA/P, the B/S/W score is 1/4/17, which indicates that GA outperforms GA/P one time, GA is statistically the same as GA/P four times, and GA/P outperforms GA 17 times. For BBO versus BBO/P, the B/S/W score is 6/5/11, which indicates that BBO outperforms BBO/P six times, BBO is statistically the same as BBO/P five times, and BBO/P outperforms BBO 11 times. Population-proportion-based EAs significantly outperform standard EAs on the CEC 2011 real-world problems.

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. Then we proposed two simple optimization search strategies whose basic features are population-proportion-based selection and modified mutation, which we called population-proportion-based EAs, 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 II–VI can be reproduced with MATLAB code that is available at the authors' Web site [34]. Tables VII and VIII showed that GA and BBO perform better on real-world benchmark problems if population-proportion-based selection is used.

The use of interactive Markov models to model EAs 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 could 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 [37]–[39]; 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 used two examples

$$
\begin{aligned}
P_S &= \left[p_{ij}\right]\\
&= \begin{bmatrix}
(1-\alpha)+\alpha\left[(1-\lambda_1)+\lambda_1(\beta/K+(1-\beta)m_1)\right] & \alpha\lambda_2(\beta/K+(1-\beta)m_1) & \cdots & \alpha\lambda_K(\beta/K+(1-\beta)m_1)\\
\alpha\lambda_1(\beta/K+(1-\beta)m_2) & (1-\alpha)+\alpha\left[(1-\lambda_2)+\lambda_2(\beta/K+(1-\beta)m_2)\right] & \cdots & \alpha\lambda_K(\beta/K+(1-\beta)m_2)\\
\vdots & \cdots & \cdots & \vdots\\
\alpha\lambda_1(\beta/K+(1-\beta)m_K) & \alpha\lambda_2(\beta/K+(1-\beta)m_K) & \cdots & (1-\alpha)+\alpha\left[(1-\lambda_K)+\lambda_K(\beta/K+(1-\beta)m_K)\right]
\end{bmatrix}
\end{aligned}
\tag{A1}
$$

$$
\begin{aligned}
P_A &= \left[p_{kj}\right]\\
&= \begin{bmatrix}
(1-p_m)+p_m/K & p_m/K & \cdots & p_m/K\\
p_m/K & (1-p_m)+p_m/K & \cdots & p_m/K\\
\vdots & \cdots & \cdots & \vdots\\
p_m/K & p_m/K & \cdots & (1-p_m)+p_m/K
\end{bmatrix}\\
&\quad\times\begin{bmatrix}
(1-\alpha)+\alpha\left[(1-\lambda_1)+\lambda_1(\beta/K+(1-\beta)m_1)\right] & \alpha\lambda_2(\beta/K+(1-\beta)m_1) & \cdots & \alpha\lambda_K(\beta/K+(1-\beta)m_1)\\
\alpha\lambda_1(\beta/K+(1-\beta)m_2) & (1-\alpha)+\alpha\left[(1-\lambda_2)+\lambda_2(\beta/K+(1-\beta)m_2)\right] & \cdots & \alpha\lambda_K(\beta/K+(1-\beta)m_2)\\
\vdots & \cdots & \cdots & \vdots\\
\alpha\lambda_1(\beta/K+(1-\beta)m_K) & \alpha\lambda_2(\beta/K+(1-\beta)m_K) & \cdots & (1-\alpha)+\alpha\left[(1-\lambda_K)+\lambda_K(\beta/K+(1-\beta)m_K)\right]
\end{bmatrix}
\end{aligned}
\tag{A3}
$$

$$
\begin{aligned}
P_A(1,1) &= ((1-p_m)+p_m/K)\left[(1-\alpha)+\alpha((1-\lambda_1)+\lambda_1(\beta/K+(1-\beta)m_1))\right]+(p_m/K)\left[\alpha\lambda_1(\beta/K+(1-\beta)m_2)\right]+\cdots+(p_m/K)\left[\alpha\lambda_1(\beta/K+(1-\beta)m_K)\right]\\
&= ((1-p_m)+p_m/K)\left[(1-\alpha)+\alpha((1-\lambda_1)+\lambda_1(\beta/K))\right]+(K-1)(p_m/K)(\alpha\lambda_1)(\beta/K)+((1-p_m)+p_m/K)(\alpha\lambda_1)(1-\beta)m_1\\
&\quad+(p_m/K)(\alpha\lambda_1)(1-\beta)(m_2+m_3+\cdots m_K)\\
&= \left(1-\frac{(K-1)p_m}{K}\right)\left[(1-\alpha)+\alpha\left((1-\lambda_1)+\lambda_1\left(\frac{\beta}{K}\right)\right)\right]+\frac{(K-1)p_m}{K}(\alpha\lambda_1)\left(\frac{\beta}{K}\right)+\left(1-\frac{(K-1)p_m}{K}\right)(\alpha\lambda_1)(1-\beta)m_1+\frac{p_m}{K}(\alpha\lambda_1)(1-\beta)(1-m_1)\\
&= \left(1-\frac{(K-1)p_m}{K}\right)(1-\alpha)+\alpha\left[\left(1-\frac{(K-1)p_m}{K}\right)(1-\lambda_1)+\lambda_1\left(\left(1-\frac{(K-1)p_m}{K}\right)\frac{\beta}{K}+\frac{(K-1)p_m}{K^2}\beta+\frac{p_m}{K}(1-\beta)+(1-\beta)(1-p_m)m_1\right)\right]
\end{aligned}
\tag{A4}
$$

from the literature to derive two 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 the combination of estimation of distribution algorithms (EDAs) with the 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. 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 (A.1), shown at the top of the page.

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

$$
\begin{aligned}
P_M &= \left[p_{ki}\right]\\
&= \begin{bmatrix}
(1-p_m)+p_m/K & p_m/K & \cdots & p_m/K\\
p_m/K & (1-p_m)+p_m/K & \cdots & p_m/K\\
\vdots & \cdots & \cdots & \vdots\\
p_m/K & p_m/K & \cdots & (1-p_m)+p_m/K
\end{bmatrix}
\end{aligned}
\tag{A2}
$$

where p_m is the mutation rate. So the transition matrix of strategy A with mutation can be computed by $P_B = [p_{kj}] = P_M P_S = \sum_i p_{ki}p_{ij}$ as (A.3), shown at the top of the page.

Combining this result with the equation $\sum_{i=1}^{K} m_i = 1$, element $P_B(1,1) = p_{11}$ in (A.3) is obtained as (A.4), shown at the top of the page.

Element $P_B(1,2) = p_{12}$ in (A.3) is obtained as (A.5), shown at the top of the next page.

We follow the same process to obtain as (A.6), shown at the top of the next page, which is equivalent to (11), as desired.

$$\begin{aligned}
P_A(1,2) &= ((1-p_m)+p_m/K)\big[\alpha\lambda_2(\beta/K+(1-\beta)m_1)\big]+(p_m/K)\big[(1-\alpha)+\alpha((1-\lambda_2)+\lambda_2(\beta/K+(1-\beta)m_2))\big] \\
&\quad+\cdots+(p_m/K)\big[\alpha\lambda_2(\beta/K+(1-\beta)m_K)\big] \\
&= ((1-p_m)+p_m/K)(\alpha\lambda_2)(\beta/K)+((1-p_m)+p_m/K)(\alpha\lambda_2)((1-\beta)m_1)+(p_m/K)\big[(1-\alpha)+\alpha((1-\lambda_2)+\lambda_2(\beta/K))\big] \\
&\quad+(p_m/K)(\alpha\lambda_2)(1-\beta)m_2+\cdots+(p_m/K)(\alpha\lambda_2)(\beta/K)+(p_m/K)(\alpha\lambda_2)((1-\beta)m_K) \\
&= ((1-p_m)+p_m/K)(\alpha\lambda_2)(\beta/K)+(p_m/K)\big[(1-\alpha)+\alpha((1-\lambda_2)+\lambda_2(\beta/K))\big]+(K-2)(p_m/K)(\alpha\lambda_2)(\beta/K) \\
&\quad+((1-p_m)+p_m/K)(\alpha\lambda_2)((1-\beta)m_1)+(p_m/K)(\alpha\lambda_2)(1-\beta)(m_2+\cdots+m_K) \\
&= \frac{(K-1)p_m}{K}(\alpha\lambda_2)\left(\frac{\beta}{K}\right)+\left(\frac{p_m}{K}\right)\left[(1-\alpha)+\alpha\left((1-\lambda_2)+\lambda_2\left(\frac{\beta}{K}\right)\right)\right]+\frac{(K-2)p_m}{K}(\alpha\lambda_2)\left(\frac{\beta}{K}\right) \\
&\quad+\left(1-\frac{(K-1)p_m}{K}\right)(\alpha\lambda_2)(1-\beta)m_1+\frac{p_m}{K}(\alpha\lambda_2)(1-\beta)(1-m_1) \\
&= \frac{p_m}{K}(1-\alpha)+\alpha\left[\frac{p_m}{K}(1-\lambda_2)+\lambda_2\left(\left(1-\frac{(K-1)p_m}{K}\right)\frac{\beta}{K}+\frac{(K-1)p_m}{K^2}\beta+\frac{p_m}{K}(1-\beta)+(1-\beta)(1-p_m)m_1\right)\right]
\end{aligned} \tag{A5}$$

$$p_{kj}=\begin{cases}\left(1-\frac{(K-1)p_m}{K}\right)(1-\alpha)+\alpha\left[\left(1-\frac{(K-1)p_m}{K}\right)(1-\lambda_j)+\lambda_j\left(\left(1-\frac{(K-1)p_m}{K}\right)\frac{\beta}{K}+\frac{(K-1)p_m}{K^2}\beta+\frac{p_m}{K}(1-\beta)+(1-\beta)(1-p_m)m_j\right)\right] & \text{if } k=j \\ \frac{p_m}{K}(1-\alpha)+\alpha\left[\frac{p_m}{K}(1-\lambda_j)+\lambda_j\left(\left(1-\frac{(K-1)p_m}{K}\right)\frac{\beta}{K}+\frac{(K-1)p_m}{K^2}\beta+\frac{p_m}{K}(1-\beta)+(1-\beta)(1-p_m)m_k\right)\right] & \text{if } k\neq j\end{cases} \tag{A6}$$

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

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

Appendix 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 (B.1), shown at the top of the page.

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

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

So the transition matrix of strategy B with mutation can be computed by $P_B=\big[p_{kj}\big]=P_MP_S=\sum_i p_{ki}p_{ij}$ as (B.3), shown at the top of the page.

Combining this result with the sum $\sum_{i=1}^{K}p_{ij}=1$ of each column of P_S, element $P_B(1,1)=p_{11}$ in (B.3) is obtained as (B.4), shown at the top of the next page.

Element $P_B(1,2)=p_{12}$ in (B.3) is obtained as (B.5), shown at the top of the next page.

Element $P_B(2,2)=p_{22}$ in (B.3) is obtained as (B.6), shown at the top of the next page.

Element $P_B(2,1)=p_{21}$ in (B.3) is obtained as (B.7), shown at the top of the next page.

We can follow the same process to obtain as (B.8), shown at the top of the next page, which is equivalent to (18), as desired.

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

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

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

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

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

Appendix 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: that is, $m = m(t)$, $m_i = m_i(t)$, $p_{ij} = p_{ij}(m)$. 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 $\big[u' = (1, \ldots, 1)\big]$. The ith equation of the interactive Markov model (1) will often be written in the following 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:

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

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

$$\begin{aligned}m(t+1) &= (A_0 + mb_0)m = [A_0 + (b_0m)I]m\\ &= \big[A_0 + \big(u'(I - A_0)m\big)I\big]m = \big[A_0 + \big(1 - u'A_0m\big)I\big]m.\end{aligned}\tag{C.3}$$

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

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

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

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

The first equation is true by the definitions of λ and z; the second equation follows from the first on premultiplying 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 the following eigenvalue:

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

Furthermore, $m^* = z$ is a unique solution to (C.4) since a non-negative indecomposable matrix cannot have two non-negative 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, NY, USA: Springer, 2006.

[2] D. H. Wolpert and W. G. Macready, "No free lunch theorems for optimization," *IEEE Trans. Evol. Comput.*, vol. 1, no. 1, pp. 67–82, Apr. 1997.

[3] N. Todorovic and S. Petrovic, "Bee colony optimization algorithm for nurse rostering," *IEEE Trans. Syst., Man, Cybern., Syst.*, vol. 43, no. 2, pp. 467–473, Mar. 2013.

[4] C. R. Reeves and J. E. Rowe, *Genetic Algorithms: Principles and Perspectives*. Norwell, MA, USA: Kluwer Academic, 2003.

[5] Z. Ding, J. Liu, Y. Sun, C. Jiang, and M. Zhou, "A transaction and QoS-aware service selection approach based on genetic algorithm," *IEEE Trans. Syst., Man, Cybern., Syst.*, vol. 45, no. 7, pp. 1035–1046, Jul. 2015.

[6] X. Yao, Y. Liu, and G. Lin, "Evolutionary programming made faster," *IEEE Trans. Evol. Comput.*, vol. 3, no. 2, pp. 82–102, Jul. 1999.

[7] S. Das and P. N. Suganthan, "Differential evolution: A survey of the state-of-the-art," *IEEE Trans. Evol. Comput.*, vol. 15, no. 1, pp. 4–31, Feb. 2011.

[8] R. Storn and K. V. Price, "Differential evolution—A simple and efficient heuristic for global optimization over continuous spaces," *J. Glob. Optim.*, vol. 11, no. 4, pp. 341–359, Dec. 1997.

[9] P. Rakshit, A. Konar, S. Das, L. C. Jain, and A. K. Nagar, "Uncertainty management in differential evolution induced multiobjective optimization in presence of measurement noise," *IEEE Trans. Syst., Man, Cybern., Syst.*, vol. 44, no. 7, pp. 922–937, Jun. 2014.

[10] P. Rakshit *et al.*, "Realization of an adaptive memetic algorithm using differential evolution and Q-learning: A case study in multirobot path planning," *IEEE Trans. Syst., Man, Cybern, Syst.*, vol. 43, no. 4, pp. 814–931, Jul. 2013.

[11] H.-G. Beyer, "Toward a theory of evolution strategies: The (μ, λ)-theory," *Evol. Comput.*, vol. 2, no. 4, pp. 381–407, 1994.

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

[13] D. Simon, "Biogeography-based optimization," *IEEE Trans. Evol. Comput.*, vol. 12, no. 6, pp. 702–713, Dec. 2008.

[14] D. Bratton and J. Kennedy, "Defining a standard for particle swarm optimization," in *Proc. IEEE Swarm Intell. Symp.*, Honolulu, HI, USA, 2007, pp. 120–127.

[15] J. Kennedy and R. Eberhart, "Particle swarm optimization," in *Proc. IEEE Int. Conf. Neural Netw.*, vol. 4. Perth, WA, Australia, 1995, pp. 1942–1948.

[16] Q. Hui and H. Zhang, "Optimal balanced coordinated network resource allocation using swarm optimization," *IEEE Trans. Syst., Man, Cybern., Syst.*, vol. 45, no. 5, pp. 770–787, May 2015.

[17] D. Simon, *Evolutionary Optimization Algorithms*. Hoboken, NJ, USA: Wiley, 2013.

[18] A. E. Nix and M. D. Vose, "Modeling genetic algorithms with Markov chains," *Ann. Math. Artif. Intell.*, vol. 5, no. 1, pp. 79–88, 1992.

[19] D. Simon, M. Ergezer, D. Du, and R. Rarick, "Markov models for biogeography-based optimization," *IEEE Trans. Syst., Man, Cybern. B, Cybern.*, vol. 41, no. 1, pp. 299–306, Feb. 2011.

[20] D. Simon, "A dynamic system model of biogeography-based optimization," *Appl. Soft Comput.*, vol. 11, no. 8, pp. 5652–5661, 2011.

[21] J. Suzuki, "A Markov chain analysis on simple genetic algorithms," *IEEE Trans. Syst., Man, Cybern., Syst.*, vol. 25, no. 4, pp. 655–659, Apr. 1995.

[22] P. J. van Laarhoven and E. H. L. Aarts, *Simulated Annealing: Theory and Applications*. New York, NY, USA: Springer, 1987.

[23] M. D. Vose, *The Simple Genetic Algorithm: Foundations and Theory*. Cambridge, MA, USA: MIT Press, 1999.

[24] W. M. Spears, *Evolutionary Algorithms: The Role of Mutation and Recombination*. New York, NY, USA: Springer, 2000.

[25] J. Conlisk, "Interactive Markov chains," *J. Math. Sociol.*, vol. 4, no. 2, pp. 157–185, 1976.

[26] Y. Gerchak, "On interactive chains with finite populations," *J. Math. Sociol.*, vol. 9, no. 3, pp. 255–258, 1983.

[27] H. L. Van Trees and K. L. Bell, Eds., *Bayesian Bounds for Parameter Estimation and Nonlinear Filtering/Tracking*. Hoboken, NJ, USA: Wiley, 2007.

[28] Y. Ebihara and N. Sebe, "Lower bound analysis of H_∞ performance achievable via decentralized LTI controllers," *Int. J. Robust Nonlin. Control*, vol. 24, no. 16, pp. 2423–2437, Nov. 2014.

[29] H. Ma, D. Simon, and M. Fei, "On the convergence of biogeography-based optimization for binary problems," *Math. Probl. Eng.*, vol. 2014, May 2014, Art. ID 147457.

[30] D. Sudholt, "A new method for lower bounds on the running time of evolutionary algorithms," *IEEE Trans. Evol. Comput.*, vol. 17, no. 3, pp. 418–435, Jun. 2013.

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

[32] J. M. Dieterich and B. Hartke. "Empirical Review of Standard Benchmark Functions Using Evolutionary Global Optimization. *Appl. Math.*, vol. 3, no. 10A, pp. 1552–1564, Oct. 2012.

[33] H. Ma, D. Simon, M. Fei, and Z. Chen, "On the equivalences and differences of evolutionary algorithms," *Eng. Appl. Artif. Intell.*, vol. 26, no. 10, pp. 2397–2407, 2013.

[34] H. Ma, D. Simon, M. Fei, and H. Mo. (Jun. 2014). *Interactive Markov Models of Evolutionary Algorithms*. [Online]. Available: http://academic.csuohio.edu/simond/InteractiveMarkov

[35] P. N. Suganthan. (Feb. 2011). *CEC11 Competition on Testing Evolutionary Algorithms on Real-World Numerical Optimization Problems*. [Online]. Available: http://www3.ntu.edu.sg/home/EPNSugan/

[36] J. Derrac, S. Garcia, D. Molina, and F. Herrera, "A practical tutorial on the use of nonparametric statistical tests as a methodology for comparing evolutionary and swarm intelligence algorithms," *Swarm Evol. Comput.*, vol. 1, no. 1, pp. 3–18, 2011.

[37] J. He and X. Yao, "Towards an analytic framework for analysing the computation time of evolutionary algorithms," *Artif. Intell.*, vol. 145, nos. 1–2, pp. 59–97, 2003.

[38] K. De Jong, W. M. Spears, and D. F. Gordon, "Using Markov chains to analyze GAFOs," in *Foundations of Genetic Algorithms*, vol. 3, L. Whitley and M. Vose, Eds. San Mateo, CA, USA: Morgan Kaufmann, 1995, pp. 115–138.

[39] J. He, F. He, and X. Yao, "A unified Markov chain approach to analysing randomized search heuristics," 2013, unpublished paper. [Online]. Available: http://arxiv.org/abs/1312.2368