2011

A Dynamic System Model of Biogeography-Based Optimization

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

A dynamic system model of biogeography-based optimization☆

Dan Simon

Cleveland State University, Department of Electrical and Computer Engineering, Cleveland, OH, United States

1. Introduction

Biogeography is the study of the migration, speciation, and
extinction of species [1,2]. Biogeography has often been considered
as a process that enforces equilibrium in the number of species in
habitats. However, equilibrium in a system can also be considered
as a minimum-energy configuration, so we see that biogeography
can be viewed as an optimization process. This idea is further dis-
cussed in [3].

Biogeography-based optimization (BBO) is an evolutionary
algorithm (EA) motivated by the optimality perspective of nat-
ural biogeography, and was initially developed in [4]. Just as
species migrate back and forth between islands, BBO operates
by sharing information between individuals in a population of
candidate solutions. BBO has shown good performance both on
benchmark problems [3–5] and on real-world problems, includ-
ing aircraft engine sensor selection [4], power system optimization
[6,7], groundwater detection [8], mechanical gear train design [9],
satellite image classification [10], and neuro-fuzzy system training
for biomedical applications [11].

Section 1.1 summarizes some of the important notation used in
this paper. Section 1.2 gives an overview of BBO. Section 1.3 gives
an overview and outline of the remainder of this paper.

1.1. Notation

This section summarizes some of the notation used in BBO and
in this paper. Some of these terms may be more general or more
specific in other contexts. The definitions indicated here are not
universal, but are commonly used, and more importantly for our
purposes, are specifically used in this paper.

BBO: Biogeography-based optimization, an EA which is moti-
vated by the mathematical principles of natural biogeography.

Dynamic system: A process whose state at time step $(t+1)$
depends only on the state at time t. The transition of the state from
one time step to the next is deterministic.

Emigration: The sharing of a solution feature in BBO from one
individual to another. The emigrating solution feature remains in
the emigrating individual. This is similar to emigration of a species
in biogeography, in which representatives of a species leave an
island but the species does not become extinct from the emigrating
island.

GASP: A genetic algorithm with single-point crossover.

GAGUR: A genetic algorithm with global uniform recombination.
This means that a solution feature of an offspring can receive each
solution feature from a different parent. The likelihood that any
given solution feature in the offspring comes from any given parent
is proportional to that parent's fitness.

Individual: A candidate solution in an EA.

Immigration: The replacement of an old solution feature in an
individual with a new solution feature from another individual. The
solution feature comes from the contributing individual by way of

☆ This work was supported by NSF Grant 0826124 in the CMMI Division of the
Engineering Directorate.
  E-mail address: d.j.simon@csuohio.edu

emigration. The immigrating solution feature replaces a feature in the immigrating individual.

Markov process: A process whose state at time step $(t+1)$ depends only on the state at time t. The transition of the state from one time step to the next is probabilistic.

Population distribution: The distribution of individuals in the search space. For example, if the search space consists of four possible solutions, $\{x_1, x_2, x_3, x_4\}$, and the population size is three, then a population distribution might consist of one copy of x_1, zero copies of x_2, two copies of x_3, and one copy of x_4.

Solution feature: An independent variable of an optimization problem. For example, if the solution domain is a bit string, then a solution feature is a bit.

Generational EA: An EA in which recombination is performed to create an entire new population before any of the old population members are replaced.

Steady-state EA: An EA in which recombination is performed to create a single new individual which replaces one of the old individuals in the population before the next recombination is performed.

1.2. Biogeography-based optimization

This section gives an overview of BBO. BBO operates by migrating information between individuals, thus resulting in a modification of existing individuals. Individuals do not die at the end of a generation. In addition, a high-fitness BBO individual is unlikely to accept information from a low-fitness individual. This is motivated by biogeography and does not have an analog in GAs. In natural biogeography, a very habitable island is unlikely to accept immigrants from a less habitable island [12]. This is due to two reasons. First, the very habitable island is already saturated with species and does not have many additional resources to support immigrants. Second, the inhabitable island does not have very many species to begin with, and so it does not have many potential emigrants.

BBO is motivated by biogeography but is not intended to be a simulation of biogeography. The analogy between biogeography and BBO breaks down at several points. For example, in biogeography the number of species varies from island to island, while in BBO the number of solution features is constant for all individuals and is equal to the problem dimension. BBO can currently only deal with optimization problems of constant dimension; its extension to variable-sized problems is a topic for future research. Although the analogy is not perfect, the key point in BBO is that the migration of solution features between individuals is motivated by the mathematical theory of species migration in biogeography.

Like other EAs, BBO operates probabilistically. The probability that an individual shares a feature with the rest of the population is proportional to its fitness. The probability that an individual receives a feature from the rest of the population decreases with its fitness. When a copy of feature s from individual z_j replaces a feature in individual y_k, we say that s has emigrated from z_j and immigrated to y_k; that is, $y_k(s) \leftarrow z_j(s)$.

Although nonlinear migration curves may give better optimization results [5], in this paper we use linear curves as shown in Fig. 1. Fig. 1 depicts two BBO individuals. S_1 depicts a poor solution and S_2 depicts a good solution. The immigration probability for S_1 will thus be higher than that of S_2, and the emigration probability for S_1 will be lower than that of S_2. Note that if a good solution is obtained in the population, then there may be a high probability that the population will converge towards that solution, resulting in premature convergence. As with any other EA, an appropriate mutation rate needs to be used in BBO to balance exploration and exploitation.

There are several ways to implement BBO. The original BBO algorithm, which we use in this paper, is called partial immigration-based BBO [13]. In this approach, for each feature in each individual
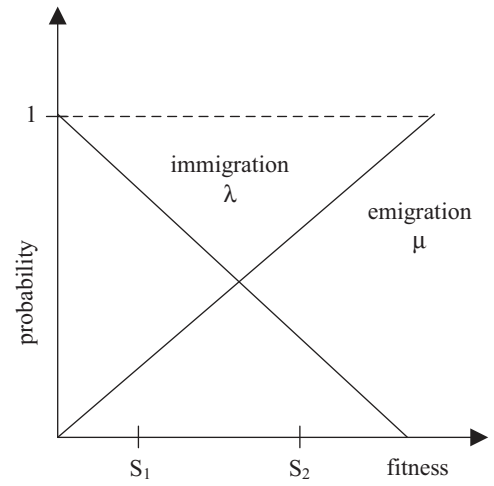

Fig. 1. Illustration of two BBO individuals using linear migration curves. S_1 represents a poor solution and S_2 represents a good solution.

we probabilistically decide whether or not to immigrate. If immigration is decided upon, then a fitness-based probabilistic method (e.g., roulette wheel selection) is used to select the emigrating individual. This gives the algorithm shown in Fig. 2 as a description of one BBO generation. We perform migration and mutation for each individual in the current generation before any individuals are replaced, resulting in a generational EA [14]. The migration decision requires that the individuals be sorted in order of fitness, which is a computational consideration. However, in almost all real-world EA applications, fitness function evaluation comprises the vast majority of computational effort.

1.3. Overview and outline

In previous work we derived a Markov chain model for BBO [15,16]. A Markov chain is a random process which has T possible state values [17, chap. 11]. The probability that the system transitions from state i to state j is given by the probability P_{ij}. The $T \times T$ matrix $P = [P_{ij}]$ is called the transition matrix. Each state in the BBO Markov model represents a population distribution, that is, a distribution of individuals in the search space. Probability P_{ij} is the probability that the population transitions from the ith population distribution to the jth population distribution in one generation.

Although we use BBO Markov theory in this paper to derive a dynamic system model, the states in the dynamic system model are not the same as the states in the Markov model. BBO Markov states represent population distributions. BBO dynamic system states represent the proportion of each individual in the population; that is, the ith state is the proportion of the ith individual in the population. The state dimension of the BBO dynamic system will thus be only a small fraction of the state dimension of the BBO Markov model. This makes the dynamic system model applicable to larger problems than the Markov model.

Section 2 reviews the BBO Markov model which forms the basis for this work and which was derived in [15,16]. Section 3 and following comprise the new contributions of this paper. Section 3 derives the BBO dynamic system model and some of its properties. We also extend the dynamic system model to a GA with global uniform recombination (GAGUR). Section 4 compares the dynamic system models of BBO, GAGUR, and GA with single-point crossover (GASP). We provide concluding remarks and suggestions for future work in Section 5.

For each individual z_k, define emigration probability $\mu_k \propto$ fitness of z_k, with $\mu_k \in [0, 1]$
For each individual z_k, define immigration probability $\lambda_k = 1 - \mu_k$
$y \leftarrow z$ (i.e., create a temporary population y)
For each individual y_k $(k = 1, \ldots, N)$
    For each solution feature s
        Use λ_k to probabilistically decide whether to immigrate to y_k
        If immigrating then
            Use the μ values to probabilistically select the emigrating individual z_j
            $y_k(s) \leftarrow z_j(s)$
        end if
    next solution feature
    Probabilistically mutate y_k
next individual
$z \leftarrow y$

Fig. 2. One generation of the BBO algorithm. z is the entire population of individuals, z_k is the kth individual, and $z_k(s)$ is the sth feature of z_k.

2. A Markov model for BBO

In [15,16] we derived a Markov model for BBO. In this section we review that model as a foundation for our later development of a dynamic system model in Section 3. The BBO Markov model in this section is based on several assumptions. First, we use a generational BBO algorithm rather than a steady-state BBO algorithm, as can be seen from the use of the temporary population y in Fig. 2. Second, an individual can emigrate a feature to itself. This means that in the statement "use the μ values to probabilistically select the emigrating individual z_j" in Fig. 2, j might be chosen to be equal to k. This is similar to the possibility of selecting the same parent twice in a GASP, which results in an offspring which is a clone of its parent. However, in BBO it is not the individual, but rather the solution feature, that is cloned in this situation. Third, the migration rates are independent of the population distribution; that is, absolute fitness values rather than rank-based fitness values [18] are used to obtain migration rates. Fourth, our optimization problem has a binary search space. Fifth, we ignore the possibility of mutation; mutation will be considered in Section 3 when we derive the dynamic system model.

Suppose that the search space consists of all bit strings x_i which have q bits each. The cardinality of the search space is therefore $n = 2^q$. The population size is denoted by N. The n-element population-count vector v denotes the number of each x_i individual in the population. Therefore,

$$\sum_{i=1}^{n} v_i = N. \qquad (1)$$

The kth individual in the population is denoted by y_k. The y_k values are ordered so that identical individuals are grouped. Furthermore, the order of the y_k values is same as that of the x_i values. The BBO population can thus be depicted as

Population $= \{y_1, \ldots, y_N\}$

$$= \{\underbrace{x_1, x_1, \ldots, x_1}_{v_1 \text{ copies}}, \underbrace{x_2, x_2, \ldots, x_2}_{v_2 \text{ copies}}, \ldots \underbrace{x_n, x_n, \ldots, x_n}_{v_n \text{ copies}}\}. \qquad (2)$$

The emigration probability of x_i is denoted as μ_i, and the immigration probability of x_i is denoted as λ_i. The sth

bit of x_i is denoted as $x_i(s)$. We use $\mathcal{J}_i(s)$ to denote the set of search space indices j such that $x_j(s) = x_i(s)$. That is,

$$\mathcal{J}_i(s) = \{j : x_j(s) = x_i(s)\}, \quad i \in [1, n]. \qquad (3)$$

Note that $|\mathcal{J}_i(s)| = n/2$ for all i and s. From (2) we see that

$$y_k = \begin{cases} x_1 & \text{for } k = 1, \ldots, v_1 \\ x_2 & \text{for } k = v_1 + 1, \ldots, v_1 + v_2 \\ x_3 & \text{for } k = v_1 + v_2 + 1, \ldots, v_1 + v_2 + v_3 \\ \vdots & \vdots \\ x_n & \text{for } k = \sum_{i=1}^{n-1} v_i + 1, \ldots, N \end{cases} \qquad (4)$$

which can also be written as

$$y_k = x_{m(k)} \quad \text{for } k = 1, \ldots, N$$

$$m(k) = \min r \quad \text{such that } \sum_{i=1}^{r} v_i \geq k. \qquad (5)$$

We use the additional subscript t to denote generation number. For example, $y_k(s)_t$ is the value of the sth bit of the kth individual at generation t. With these definitions, it is shown in [15,16] that the probability that $y_k(s)_{t+1} = x_i(s)$ can be written as

$$\Pr(y_k(s)_{t+1} = x_i(s)) = (1 - \lambda_{m(k)})\mathbf{1}[x_{m(k)}(s) = x_i(s)] + \lambda_{m(k)} \frac{\displaystyle\sum_{j \in \mathcal{J}_i(s)} v_j \mu_j}{\displaystyle\sum_{j=1}^{n} v_j \mu_j} \qquad (6)$$

where $\mathbf{1}[\cdot]$ is the predicate function; that is, $\mathbf{1}[\mathcal{A}] = 1$ if $\mathcal{A}$ is true, and 0 otherwise. There are q bits in each y_k. Therefore, if the population-count vector is equal to v at the tth generation, then the probability that immigration results in $y_{k,t+1} = x_i$ is denoted by $P_{ki}(v)$ and can be written as

$$P_{ki}(v) = \Pr(y_{k,t+1} = x_i)$$

$$= \prod_{s=1}^{q} \left[(1 - \lambda_{m(k)})\mathbf{1}[x_{m(k)}(s) = x_i(s)] + \lambda_{m(k)} \frac{\sum\limits_{j \in \mathcal{J}_i(s)} v_j \mu_j}{\sum\limits_{j=1}^{n} v_j \mu_j} \right]. \tag{7}$$

$P_{ki}(v)$ can be computed for each $k \in [1, N]$ and each $i \in [1, n]$ in order to form the $N \times n$ matrix $P(v)$. This gives the probability that each of N BBO migration trials results in each of n search space individuals. We use w_i to denote the total number of times that individual x_i is obtained after all N migration trials have been completed for a given generation, and we define $w = [\, w_1 \ \cdots \ w_n \,]^T$. The probability that we obtain the population-count vector w at the $(t+1)$ st generation, given that v is the population-count vector at the tth generation, can be obtained from the generalized multinomial theorem [15,16,19] as

$$\Pr(w|v) = \sum_{J \in Y(w)} \prod_{k=1}^{N} \prod_{i=1}^{n} P_{ki}^{J_{ki}}(v) \tag{8}$$

$$Y(w) = \left\{ J \in \mathbf{R}^{N \times n} : J_{ki} \in \{0, 1\}, \quad \sum_{i=1}^{n} J_{ki} = 1 \text{ for all } k, \quad \sum_{k=1}^{N} J_{ki} = w_i \text{ for all } i \right\}.$$

The Markov transition matrix is obtained by computing (8) for each possible v vector and each possible w vector. The transition matrix is thus a $T \times T$ matrix, where T is the total number of possible population distributions. That is, T is the number of all possible $n \times 1$ integer vectors such that $\sum T_i = N$ and $0 \le T_i \le N$. In [20] it is shown that T can be calculated using the formula for combinations:

$$T = \binom{n + N - 1}{N}. \tag{9}$$

Other methods for calculating T are discussed in [15].

3. A dynamic system model for BBO

In this section we use the Markov model of Section 2 to derive a dynamic system model for BBO and GAGUR. Eq. (7) gives $P_{ki}(v)$, which is the probability that the kth migration results in $y_k = x_i$ at the $(t+1)$ st generation, assuming that v is the population-count vector at the tth generation. In order to derive a dynamic system model for BBO, we make a slight change in the algorithm of Fig. 2. We still cycle through the immigration loop N times, where N is the population size. However, instead of deterministically cycling through each population member y_k for immigration, we randomly select a population member for immigration each of the N times through the loop. Therefore, each time through the immigration loop, each y_k has a $1/N$ chance of being selected for immigration. Note that as $N \to \infty$, this is equivalent to the algorithm of Fig. 2. With this change, the BBO algorithm is modified to become the algorithm shown in Fig. 3.

With this algorithm, the probability that the hth migration trial M_h results in x_i is

$$\Pr(M_h = x_i) = \frac{1}{N} \sum_{k=1}^{N} \Pr(y_k = x_i). \tag{10}$$

Now note that

$$\Pr(y_{k_1} = x_i) = \Pr(y_{k_2} = x_i) \quad \text{if } y_{k_1} = y_{k_2}. \tag{11}$$

We can combine (4), (5) and (11) to get

$$\Pr(y_{k_1} = x_i) = \Pr(y_{k_2} = x_i) \quad \text{if } (k_1, k_2) \in \left[\begin{array}{cc} \sum\limits_{i=1}^{m(k-1)} v_i + 1, & \sum\limits_{i=1}^{m(k)} v_i \end{array} \right]. \tag{12}$$

Therefore (10) can be written as

$$\Pr(M_h = x_i) = \frac{1}{N} \sum_{k=1}^{n} v_k \left\{ \prod_{s=1}^{q} \left[(1 - \lambda_k)\mathbf{1}[x_k(s) = x_i(s)] + \lambda_k \frac{\sum\limits_{j \in \mathcal{J}_i(s)} v_j \mu_j}{\sum\limits_{j=1}^{n} v_j \mu_j} \right] \right\}. \tag{13}$$

Now we define the proportionality vector p as

$$p = \frac{v}{N}. \tag{14}$$

That is, p_i is the proportion of x_i individuals in the population for $i \in [1, n]$, and the elements of p add up to 1. Eq. (13) can then be written as

$$\Pr(M_h = x_i) = \sum_{k=1}^{n} p_k \left\{ \prod_{s=1}^{q} \left[(1 - \lambda_k)\mathbf{1}[x_k(s) = x_i(s)] + \lambda_k \frac{\sum\limits_{j \in \mathcal{J}_i(s)} v_j \mu_j}{\sum\limits_{j=1}^{n} v_j \mu_j} \right] \right\}$$

$$= \sum_{k=1}^{n} \frac{p_k}{(p^T \mu)^q} \left\{ \prod_{s=1}^{q} \left[p^T \mu (1 - \lambda_k)\mathbf{1}[x_k(s) = x_i(s)] + \lambda_k \sum\limits_{j \in \mathcal{J}_i(s)} v_j \mu_j \right] \right\}. \tag{15}$$

The quantities on the right side of the above equation are defined at the tth generation. The left side of the above equation gives the probability of obtaining x_i at the $(t+1)$ st generation. We can use [21, Theorem 2] to see that the left side of the above equation is equal to the proportion of x_i individuals in the population at the $(t+1)$ st generation:

$$p_i(t+1) = \sum_{k=1}^{n} \frac{p_k(t)}{(p^T(t)\mu)^q} \left\{ \prod_{s=1}^{q} \left[p^T(t)\mu(1 - \lambda_k)\mathbf{1}[x_k(s) = x_i(s)] + \lambda_k \sum\limits_{j \in \mathcal{J}_i(s)} v_j(t)\mu_j \right] \right\} \tag{16}$$

where we have now explicitly shown that p is a function of t. Writing the above equation for $i \in [1, n]$ gives a nonlinear dynamic system for the evolution of the proportionality vector:

$$p(t+1) = f(p(t)). \tag{17}$$

To include the possibility of mutation, we denote the $n \times n$ mutation matrix as U, where U_{ij} is the probability that x_j mutates to x_i. If the elements of the search space $\{x\}$ are in natural binary order and each bit has a probability u of mutation, then

$$U_{ij} = u^{d_{ij}}(1 - u)^{q - d_{ij}} \tag{18}$$

where d_{ij} is the Hamming distance between x_i and x_j [22, p. 153]. This gives the dynamic system equation

$$p(t+1) = Uf(p(t)). \tag{19}$$

If mutation is not used in the BBO algorithm, then U is the identity matrix and (19) reduces to (17).

For each individual z_k, define emigration probability $\mu_k \propto$ fitness of z_k, with $\mu_k \in [0, 1]$
For each individual z_k, define immigration probability $\lambda_k = 1 - \mu_k$
$y \leftarrow z$ (i.e., create a temporary population y)
For $h = 1, \ldots, N$
    Randomly select one of the y_k individuals $(k = 1, \ldots, N)$
    For each solution feature s
        Use λ_k to probabilistically decide whether to immigrate to y_k
        If immigrating then
            Use the μ values to probabilistically select the emigrating individual z_j
            $y_k(s) \leftarrow z_j(s)$
        end if
    next solution feature
    Probabilistically mutate y_k
next h
$z \leftarrow y$

Fig. 3. One generation of the BBO algorithm with random selection of the immigrating individual.

3.1. Special case: $\lambda = 0$

It is instructive to consider the dynamic system when $\lambda_k = 0$ for all k. In this case, there is no possibility of immigration and (15) reduces to

$$p_i(t + 1) = \sum_{k=1}^{n} p_k(t) \prod_{s=1}^{q} \mathbf{1}[x_k(s) = x_i(s)] \quad . \tag{20}$$

Since each x_i is distinct, we see that

$$\prod_{s=1}^{q} \mathbf{1}[x_k(s) = x_i(s)] = \mathbf{1}[k = i] \tag{21}$$

which gives

$$p_i(t + 1) = \sum_{k=1}^{n} p_k(t)\mathbf{1}[k = i] = p_i(t). \tag{22}$$

That is, with no immigration and no mutation, the proportionality vector does not change from one generation to the next, which agrees with intuition.

3.2. Special case: $\lambda = 1$ and random feature selection

If $\lambda_k = 1$ for all k, then the BBO algorithm of Fig. 3 becomes a special type of a genetic algorithm with global uniform recombination (GAGUR) [23]. GAGUR can be implemented in many different ways, but if it is implemented with the entire population as potential contributors to the next generation [24], and with fitness-based selection for each solution feature in each offspring, then it is equivalent to BBO with $\lambda_k = 1$ for all k. In this case, immigration takes place for all individuals in the population, and the new individual that results from each immigration can be thought of as an offspring of the previous generation. Suppose also that in addition to $\lambda_k = 1$ for all k, each immigration trial migrates one randomly selected bit. Then the BBO algorithm of Fig. 3 becomes the GAGUR algorithm of Fig. 4.

The probability that y_k at the $(t + 1)$ st generation is equal to x_i, given that solution feature s was selected for migration, can be written as

$$\Pr(y_{k,t+1} = x_i | s) = \Pr[y_{k,t}(r : r \neq s) = x_i(r : r \neq s)]\Pr(y_{k,t+1}(s) = x_i(s)). \tag{23}$$

The first term on the right side of (23) is the proportion of the population which has all bits r such that $r \neq s$, equal to the corresponding bits in x_i. We denote the indices of these individuals as $\mathcal{L}_i(s)$:

$$\mathcal{L}_i(s) = \{j : x_j(r : r \neq s) = x_i(r : r \neq s)\}, \quad i \in [1, n]. \tag{24}$$

Note that $|\mathcal{L}_i(s)| = 2$ for all (i, s). Now we can write (23) as

$$\Pr(y_{k,t+1} = x_i | s) = \left(\frac{\sum\limits_{j \in \mathcal{L}_i(s)} v_j}{\sum\limits_{j=1}^{n} v_j} \right) \left(\frac{\sum\limits_{j \in \mathcal{J}_i(s)} v_j \mu_j}{\sum\limits_{j=1}^{n} v_j \mu_j} \right). \tag{25}$$

We can use (1) and (14) to write the above equation as

$$\Pr(y_{k,t+1} = x_i | s) = \sum_{j \in \mathcal{L}_i(s)} p_j \frac{\sum\limits_{j \in \mathcal{J}_i(s)} p_j \mu_j}{n}. \tag{26}$$

Fig. 4 shows that each bit $s \in [1, q]$ has a $1/q$ probability of being selected as the migrating feature. Therefore,

$$\Pr(y_{k,t+1} = x_i) = \frac{1}{qp^T \mu} \sum_{s=1}^{q} \left[\sum_{j \in \mathcal{L}_i(s)} p_j \right] \left[\sum_{j \in \mathcal{J}_i(s)} p_j \mu_j \right]. \tag{27}$$

This is a quadratic function of the p_i terms and can thus be written as

$$\Pr(y_{k,t+1} = x_i) = \sum_{a=1}^{n} \sum_{b=1}^{n} Y_{i,ab} p_a p_b. \tag{28}$$

For each individual z_k, define emigration probability $\mu_k \propto$ fitness of z_k
$y \leftarrow z$ (i.e., create a temporary population y)
For $h = 1, \ldots, N$
      Randomly select one of the y_k individuals $(k = 1, \ldots, N)$
      Randomly select a solution feature s
      Use $\{\mu\}$ to probabilistically select the emigrating individual z_j
      $y_k(s) \leftarrow z_j(s)$
      Probabilistically mutate y_k
next h
$z \leftarrow y$

Fig. 4. One generation of GAGUR with random selection of the immigrating individual and random selection of the migrating solution feature.

Eq. (27) shows that the p_m^2 coefficient on the right side of (28), where $m \in [1, n]$, can be written as

$$Y_{i,mm} = \sum_{s=1}^{q} \mu_m \mathbf{1}[m \in \mathcal{J}_i(s)]\mathbf{1}[m \in \mathcal{L}_i(s)] = \sum_{s=1}^{q} \mu_m \mathbf{1}[m \in (\mathcal{J}_i(s) \cap \mathcal{L}_i(s))].$$

(29)

From the definition of $\mathcal{L}_i(s)$ in (24) we know that $i \in \mathcal{L}_i(s)$. We also know that there is only one other element in $\mathcal{L}_i(s)$. The other element in $\mathcal{L}_i(s)$, say α, has a bit string such that $x_\alpha(r) = x_i(r)$ for all $r \neq s$. But since $\alpha \neq i$ we know that $x_\alpha(s) \neq x_i(s)$, which means that $\alpha \notin \mathcal{J}_i(s)$. Therefore

$$\mathcal{J}_i(s) \cap \mathcal{L}_i(s) = \{i\} \quad \text{for all } s.$$

(30)

Eq. (29) can therefore be written as

$$Y_{i,mm} = \sum_{s=1}^{q} \mu_m \mathbf{1}[m = i] = q\mu_m \mathbf{1}[m = i].$$

(31)

We can use (27) to show that the $p_m p_k$ coefficient $(k \neq m)$ on the right side of (28) can be written as

$$Y_{i,mk} + Y_{i,km} = \mu_m \sum_{s=1}^{q} \mathbf{1}[m \in \mathcal{J}_i(s)]\mathbf{1}[k \in \mathcal{L}_i(s)]$$

$$+ \mu_k \sum_{s=1}^{q} \mathbf{1}[k \in \mathcal{J}_i(s)]\mathbf{1}[m \in \mathcal{L}_i(s)] \quad \text{for } m \neq k.$$

(32)

The GAGUR dynamic system model can thus be written as the following set of n coupled quadratic equations:

$$p_i(t + 1) = p^T(t)Y_i p(t), \quad i \in [1, n]$$

(33)

where $Y_{i,mk}$ is the element in the mth row and kth column of Y_i. If mutation is included in the GAGUR algorithm, then

$$p(t + 1) = U \operatorname{diag}(p^T(t)Y_i p(t))$$

(34)

where $\operatorname{diag}(p^T(t)Y_i p(t))$ is the $n \times n$ diagonal matrix consisting of $p^T(t)Y_1 p(t), \ldots, p^T(t)Y_n p(t)$.

4. Dynamic system model results

Section 4.1 verifies the dynamic system model derived in the previous section. Section 4.2 compares the dynamic system models of GA with single-point crossover (GASP), GAGUR, and BBO.

4.1. Verification of dynamic system models

The dynamic system model for BBO is given in (16)–(19) with μ_k proportional to fitness, and $\lambda_k = 1 - \mu_k$. The dynamic system model for GAGUR is given in (16)–(19) with $\lambda_k = 1$, which is equivalent to (34). The dynamic system model for GASP with roulette-wheel selection was originally developed in [25]. It is summarized in [22, chap. 6] as

$$p_i(t + 1) = \frac{p^T(t)\operatorname{diag}(\mu)U^T C(i)U \operatorname{diag}(\mu)p(t)}{(p^T(t)\mu)^2}$$

(35)

where $\operatorname{diag}(\mu)$ is the $n \times n$ diagonal matrix consisting of the elements of μ (fitness), and U is the mutation matrix given in (18). $C(i)$ is an $n \times n$ matrix such that the element in its mth row and kth column is the probability that x_m and x_k cross over to produce x_i.

To verify the dynamic system models, we consider a simple three-bit problem ($n = 8$) with a per-bit mutation rate $u = 0.2$. The fitness values, which are equivalent to unnormalized BBO emigration rates, are given as follows:

$$
\begin{aligned}
&\mu(000) = 8, &&\mu(001) = 1, \\
&\mu(010) = 1, &&\mu(011) = 1, \\
&\mu(100) = 1, &&\mu(101) = 1, \\
&\mu(110) = 1, &&\mu(111) = 9.
\end{aligned}
$$

(36)

This is a relatively difficult optimization problem because $x_1 = 000$ has a high fitness, and every time we add a 1 bit to it the fitness decreases dramatically, but the individual with all 1's has the highest fitness. We begin with an initial population with proportionality vector

$$p(0) = [\, 0.8 \quad 0.1 \quad 0.1 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0 \,]^T.$$

(37)

Figs. 5–7 show some dynamic system model results and simulation results for EAs with a population size of 1000. The plots provide confirmation for the dynamic system models presented earlier. The simulation results oscillate around their mean value, which is expected because of the mutation operator. The simulation results will vary from one simulation to the next, and will never exactly match the theory, due to the stochastic nature of the simulations. That is why the dynamic system models can be more useful than simulation; the models are exact while simulation results are only approximate.
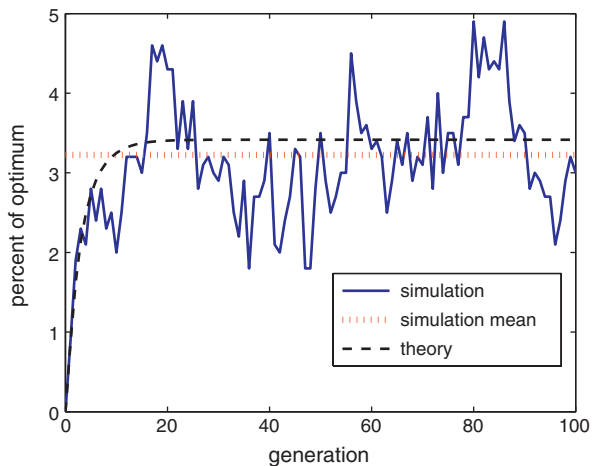
Fig. 5. BBO dynamic system results giving confirmation that simulation matches theory. The traces show the proportion of the optimal individuals for a typical simulation, the mean of that proportion over all generations, and the proportion according to the dynamic system model.

Fig. 6. GAGUR dynamic system results giving confirmation that simulation matches theory. The traces show the proportion of the optimal individuals for a typical simulation, the mean of that proportion over all generations, and the proportion according to the dynamic system model.
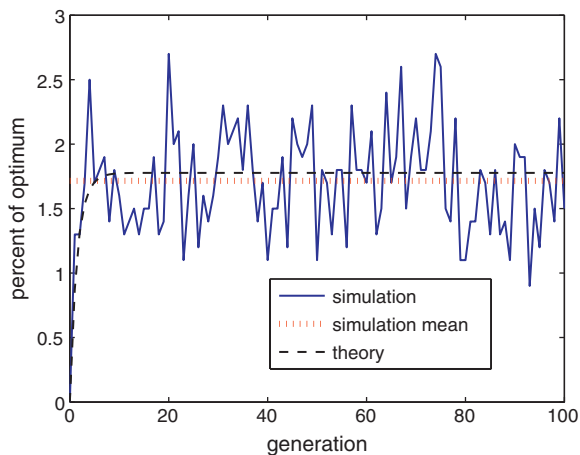

Fig. 7. GASP dynamic system results giving confirmation that simulation matches theory. The traces show the proportion of the optimal individuals for a typical simulation, the mean of that proportion over all generations, and the proportion according to the dynamic system model.
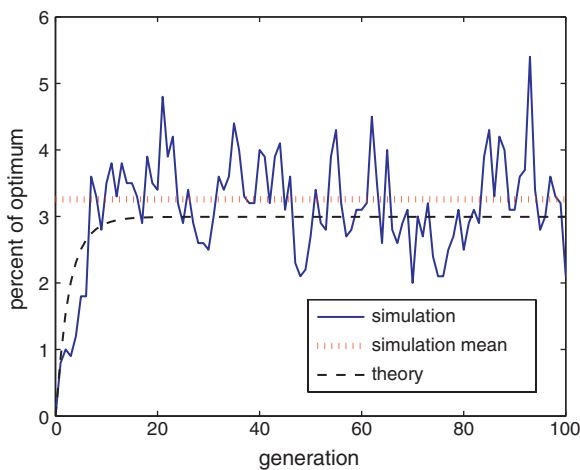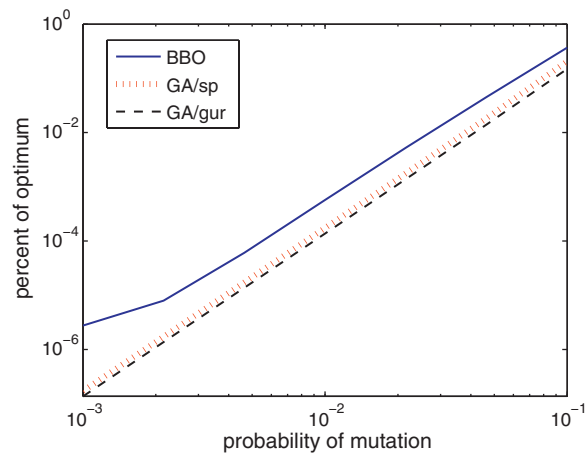

Fig. 8. Dynamic system model results for a 3-bit problem (search space cardinality $n = 8$) showing the steady-state proportion of optimal individuals.

4.2. Comparison of dynamic system models

Next we compare dynamic system model results between BBO, GAGUR, and GASP. We consider a problem whose fitness values are given as

$$\mu_i = \begin{matrix} 8 & \text{for } x_i = (0 \cdots 0) \\ 9 & \text{for } x_i = (1 \cdots 1) \\ 1 & \text{for all other } x_i. \end{matrix} \qquad (38)$$

This is the same as (36) except that it is generalized for an arbitrary number of bits. The proportionality vector of the initial population is given as

$$p(0) = \frac{1}{n-1} [\, 1 \quad \ldots \quad 1 \quad 0 \,]^T. \qquad (39)$$

That is, the initial population is evenly distributed among the sub-optimal individuals, and there are no optimal individuals. Figs. 8–10 show steady-state dynamic system model results for three different search space cardinalities, plotted as functions of mutation rate. Fig. 8 shows that BBO is much better at achieving a high percentage of optimal individuals than GASP and GAGUR for small problems. Figs. 9 and 10 show that as the problem dimension gets larger, BBO performance gets worse relative to the GAs for large mutation rates. However, BBO remains many orders of magnitude better than the GAs for small mutation rates, which are more typical for real-world problems.
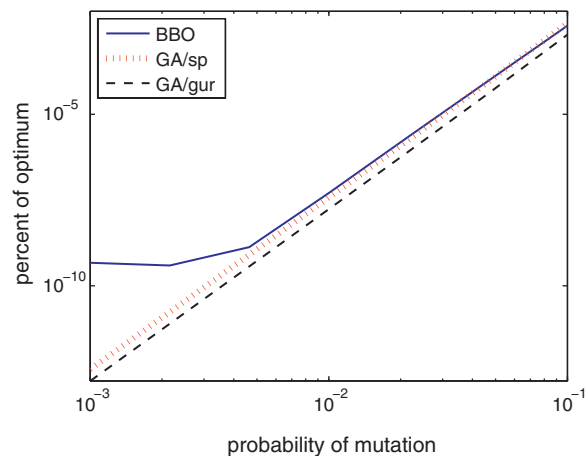

Fig. 9. Dynamic system model results for a 5-bit problem (search space cardinality $n = 32$) showing the steady-state proportion of optimal individuals.
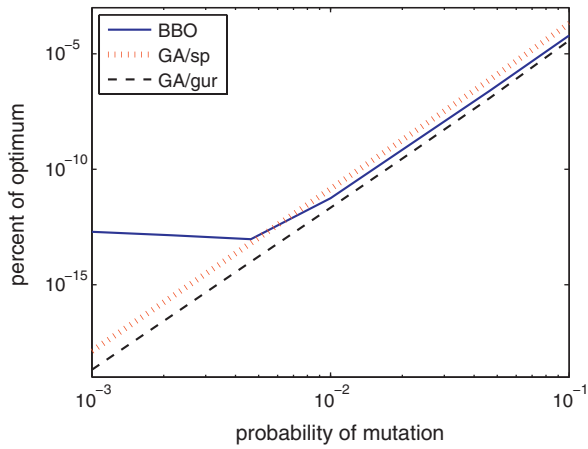
Fig. 10. Dynamic system model results for a 7-bit problem (search space cardinality *n* = 128) showing the steady-state proportion of optimal individuals.
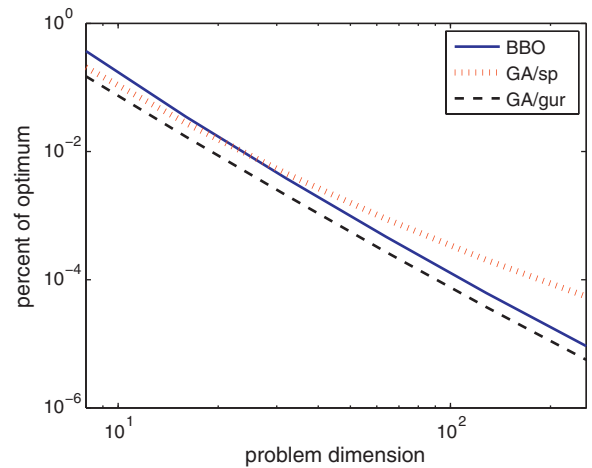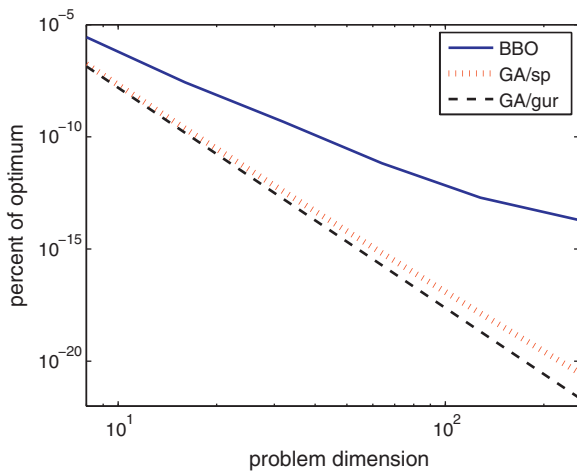

Fig. 11. Dynamic system model results for mutation rate = 0.1% per bit showing the steady-state proportion of optimal individuals.

Figs. 11–13 depict the same information as that shown in Figs. 8–10, but presented in a different way. Figs. 11–13 show dynamic system model results for three different mutation rates, plotted as functions of problem dimension. Fig. 11 shows that BBO is much better than the GAs for all problem dimensions if the
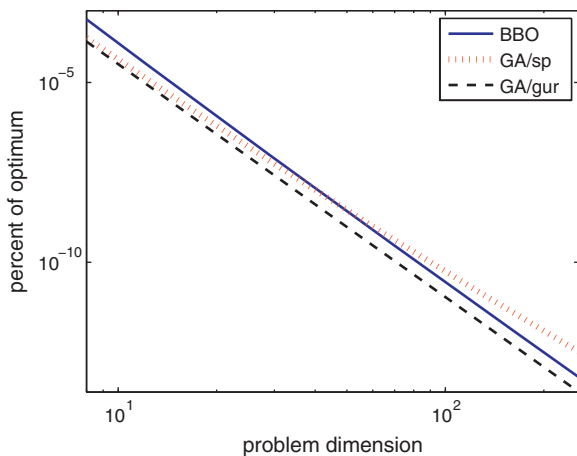

Fig. 12. Dynamic system model results for mutation rate = 1% per bit showing the steady-state proportion of optimal individuals.

Fig. 13. Dynamic system model results for mutation rate = 10% per bit showing the steady-state proportion of optimal individuals.

mutation rate is low, as is typical of real-world problems. Figs. 12 and 13 show that as the mutation rate increases, BBO remains better than the GAs for small problem dimensions, but becomes worse than GASP as the problem dimension increases.

As seen in Fig. 11, with realistic mutation rates BBO is much better than the GAs for all problem dimensions. Furthermore, the relative advantage of BBO increases as the problem dimension increases. This is consistent with the conclusions presented in [23] which were based on a different type of analysis and which were confirmed with a variety of standard benchmark simulations.

Next we compare the dynamic system model results of BBO, GASP, and GAGUR, on standard benchmark functions. The Needle Function is given in (38). The Onemax Function has a fitness that is proportional to the number of one-bits in each bit string. The Deceptive Function is the same as the Onemax Function, except that the bit string with all zeros has the highest fitness. The continuous functions that we use are listed in Table 1 and are documented in [26–28]. We implemented the continuous functions as two-dimensional functions whose independent variables are coded with three or four bits per independent variable. This gives an optimization problem with either six or eight bits total, which results in a search space cardinality of either 64 or 256. We initialized the population with a uniform distribution over all of the non-optimal solutions. The initial population did not have any optima. We recorded the percent of optimal solutions in the population after 10 generations, which gives an idea of how fast each algorithm converges. Table 1 shows the results. Note that these are not simulation results, but exact dynamic system model results.

For both the 64-bit and 256-bit problems, BBO performed the best in 15 out of 19 benchmarks. More importantly, for very difficult problems (the Needle and Deceptive Functions), BBO performed better than GASP and GAGUR by orders of magnitude.

These results are not intended to give a comprehensive comparison of BBO and GAs; extensive comparisons between BBO and other EAs using standard benchmark functions are shown in [3]. The theory and results here are instead intended to show how dynamic system models can be used to compare EAs in situations where probabilities are extremely small and where Monte Carlo simulations are therefore not useful. Dynamic system models can also be used to study the effect of various parameter settings and learning approaches. Dynamic system models can also aid in the development of adaptive algorithms or parameter update schemes that work well on many different types of problems. Our dynamic system models can also be used to help understand the behavior of BBO; for example, how and why it works well, or does not

Table 1
Dynamic system results on benchmark functions. The number in each cell indicates the percentage of optimal individuals in the population after 10 generations. The best result for each benchmark/cardinality combination is shown in boldface font.

Function	Cardinality = 64			Cardinality = 256		
	BBO	GAGUR	GASP	BBO	GAGUR	GASP
Needle	**6.97×10^{-3}**	2.32×10^{-10}	2.17×10^{-6}	**6.72×10^{-3}**	9.95×10^{-6}	6.48×10^{-6}
Onemax	46.04	**47.67**	42.64	22.44	**23.84**	19.43
Deceptive	**3.89×10^{-3}**	2.25×10^{-4}	2.33×10^{-4}	**1.03×10^{-3}**	5.36×10^{-5}	5.13×10^{-5}
Rosenbrock	**10.37**	3.56	9.13	**6.60**	4.18	6.49
Ackley	58.92	25.40	**71.38**	43.40	54.17	**70.00**
Fletcher	74.30	**88.41**	28.88	**32.19**	23.15	23.73
Griewank	**67.82**	44.18	49.00	**36.12**	15.07	25.61
Penalty #1	**29.69**	24.98	27.77	**26.22**	25.02	25.68
Penalty #2	**32.83**	24.96	28.98	13.03	12.51	**14.08**
Quartic	**39.09**	26.13	34.12	**18.64**	12.49	17.01
Rastrigin	43.93	**50.05**	41.79	**46.85**	14.64	42.43
Schwefel 1.2	**69.37**	25.29	49.21	**35.91**	14.02	25.07
Schwefel 2.22	**69.14**	25.80	51.27	**38.10**	13.36	26.84
Schwefel 2.21	**75.65**	66.84	62.89	**49.03**	39.82	38.60
Schwefel 2.26	**15.93**	0.47	1.30	**38.07**	15.69	8.70
Sphere	**69.63**	47.69	49.21	**35.95**	12.50	24.94
Ellipsoid	**65.82**	56.07	48.84	**34.15**	16.08	24.80
Michalewicz	**34.01**	33.30	30.38	14.47	12.32	**15.67**
Step	**50.04**	25.04	37.56	**25.51**	12.52	19.45

work well, on certain type of problems. This paper does not explore these many possible research directions, but we envision that the groundwork laid here will encourage these ideas and make their pursuit possible.

5. Conclusion

Mature EAs, such as GAs, have a well-structured theory. This paper attempts to extend BBO theory in that direction to enable its use and study as an alternative EA. We have summarized a previously derived BBO Markov model and used it to obtain a new dynamic system model for BBO. We have further shown how it can be used to derive a new dynamic system model for GAGUR. We compared the dynamic system models between BBO, GAGUR, and GASP on some simple optimization problems. We have seen that BBO far outperforms GAs when mutation rates are low (0.1% per bit), as are typically used for real-world problems. In addition, the relative advantage of BBO increases as the problem dimension increases. This indicates that BBO can be especially useful for large, real-world problems when small mutation rates are used.

For some real-world problems where small population sizes are required due to huge computational complexity for fitness evaluation, it may be desirable to use large mutation rates on the order of 10% per bit [29]. In this case BBO still outperforms the GAs for small problem dimensions, but GASP is the best performer for large problems. Note that these conclusions are similar to those of [23], which were obtained using a different approach. Reference [23] also provides a more extensive discussion of the conceptual similarities and differences between GAs and BBO.

There are many interesting possibilities for future work. One is to extend BBO using features of natural biogeography theory [1,2]. This could include modeling nonlinear migration curves, modeling species populations and their effects on migration, modeling predator/prey relationships, including species mobilities in the migration model, including directional migration momentum as a temporal effect from one generation to the next, modeling habitat area and isolation, and many others.

In this paper we used Markov theory for partial immigration-based BBO to derive a dynamic system model. In addition, we analyzed a generational BBO algorithm; that is, modification of each individual in the current generation occurs before any individuals are replaced in the population [14]. Future work could include the extension of our Markov and dynamic system models for other variations of BBO. These variations include partial emigration-based BBO, single immigration-based BBO, single emigration-based BBO [13], oppositional BBO [30], and a steady-state BBO in which individuals in the population are modified with replacement. It would also be of interest to extend the Markov theory and dynamic system model to BBO with nonlinear migration curves [5]. Also, our results have been restricted to problems with binary representations, and it would be interesting and useful to develop Markov and dynamic system models of BBO with other types of representations. Finally, our results are exact only in the limit as the population size approaches infinity. Many EA applications have large populations, but many others do not. Further work could focus on obtaining a dynamic system model for small population sizes, perhaps by combining states into "meta-states," or perhaps by using the BBO Markov model in either [13] or [15].

These extensions would allow analytical comparisons between different types of BBO algorithms rather than a reliance on simulation results. Although simulation results are important and necessary, if used apart from theory they can be misleading. For example, Fig. 11 shows that BBO is orders of magnitude better than GAs for large problems with a low mutation rate. However, since the probabilities in Fig. 11 are so small, it would be difficult to derive such a conclusion from simulation results because of the huge amount of computation that would be required.

Another suggestion for future work is to combine BBO with other EAs. Hybrid EAs are an important topic of research and can exploit the strengths of multiple methods in a single optimization algorithm [31]. BBO has already been combined with opposition-based learning [30]. Future work could focus on combining BBO with the many other EAs that are available. These hybrid algorithms should be studied not only with benchmarks and real-world optimization problems, but also with analytical approaches such as the Markov and dynamic system theory used in this paper. Such analytical tools include the stochastic character of EAs, but do not depend on the random results of simulation studies to draw general conclusions.

References

[1] M. Lomolino, B. Riddle, J. Brown, Biogeography, Sinauer Associates, 2009.
[2] R. Whittaker, Island Biogeography, Oxford University Press, 1998.
[3] H. Ma, An analysis of the equilibrium of migration models for biogeography-based optimization, Information Sciences 180 (2010) 3444–3464.
[4] D. Simon, Biogeography-based optimization, IEEE Transactions on Evolutionary Computation 12 (December) (2008) 702–713.

[5] H. Ma, S. Ni, M. Sun, Equilibrium species counts and migration model trade-offs for biogeography-based optimization, in: IEEE Conference on Decision and Control, Shanghai, December 2009, pp. 3306–3310.

[6] R. Rarick, D. Simon, F. Villaseca, B. Vyakaranam, Biogeography-based optimization and the solution of the power flow problem, in: IEEE Conference on Systems, Man, and Cybernetics, October 2009, pp. 1029–1034.

[7] P. Roy, S. Ghoshal, S. Thakur, Biogeography-based optimization for economic load dispatch problems, Electric Power Components and Systems 38 (January) (2010) 166–181.

[8] H. Kundra, A. Kaur, V. Panchal, An integrated approach to biogeography based optimization with case based reasoning for retrieving groundwater possibility, in: 8th Annual Asian Conference and Exhibition on Geospatial Information, Technology and Applications, August 2009.

[9] V. Savsani, R. Rao, D. Vakharia, Discrete optimisation of a gear train using biogeography based optimisation technique, International Journal of Design Engineering 2 (2009) 205–223.

[10] V. Panchal, P. Singh, N. Kaur, H. Kundra, Biogeography based satellite image classification, International Journal of Computer Science and Information Security 6 (January) (2009) 269–274.

[11] M. Ovreiu, D. Simon, Biogeography-based optimization of neuro-fuzzy system parameters for diagnosis of cardiac disease, in: Genetic and Evolutionary Computation Conference, July 2010, pp. 1235–1242.

[12] R. MacArthur, E. Wilson, The Theory of Biogeography, Princeton University Press, 1967.

[13] D. Simon, A probabilistic analysis of a simplified biogeography-based optimization algorithm, Evolutionary Computation, in print, available at http://embeddedlab.csuohio.edu/BBO.

[14] F. Vavak, T. Fogarty, Comparison of steady state and generational genetic algorithms for use in nonstationary environments, in: IEEE International Conference on Evolutionary Computation, May 1996, pp. 192–195.

[15] D. Simon, M. Ergezer, D. Du, R. Rarick, Markov models for biogeography-based optimization, IEEE Transactions on Systems, Man, and Cybernetics – Part B: Cybernetics 41 (January) (2011) 299–306.

[16] D. Simon, M. Ergezer, D. Du, Population distributions in biogeography-based optimization algorithms with elitism, in: IEEE Conference on Systems, Man, and Cybernetics, October 2009, pp. 1017–1022.

[17] C. Grinstead, J. Snell, Introduction to Probability, American Mathematical Society, 1997.

[18] S. Venkatraman, G. Yen, A simple elitist genetic algorithm for constrained optimization, in: IEEE Congress on Evolutionary Computation, June 2004, pp. 288–295.

[19] N. Beaulieu, On the generalized multinomial distribution, optimal multinomial detectors, and generalized weighted partical decision detectors, IEEE Transactions on Communications 39 (February) (1991) 193–194.

[20] A. Nix, M. Vose, Modeling genetic algorithms with Markov chains, Annals of Mathematics and Artificial Intelligence 5 (March) (1992) 79–88.

[21] M. Vose, Random heuristic search, Theoretical Computer Science 229 (November) (1999) 103–142.

[22] C. Reeves, J. Rowe, Genetic Algorithms: Principles and Perspectives, Kluwer, 2003.

[23] D. Simon, R. Rarick, M. Ergezer, D. Du, Analytical and numerical comparisons of biogeography-based optimization and genetic algorithms, Information Sciences 181 (April) (2011) 1224–1248.

[24] D. Ackley, A Connectionist Machine for Genetic Hillclimbing, Kluwer Academic Publishers, 1987.

[25] M. Vose, The Simple Genetic Algorithm, MIT Press, 1999.

[26] T. Back, Evolutionary Algorithms in Theory and Practice, Oxford University Press, 1996.

[27] X. Yao, Y. Liu, G. Lin, Evolutionary programming made faster, IEEE Transactions on Evolutionary Computation 3 (July) (1999) 82–102.

[28] Z. Cai, Y. Wang, A multiobjective optimization-based evoluationary algorithm for constrained optimization, IEEE Transactions on Evolutionary Computation 10 (December) (2006) 658–675.

[29] R. Haupt, Optimum population size and mutation rate for a simple real genetic algorithm that optimizes array factors, in: International Symposium on Antennas and Propagation, July 2000, pp. 1034–1037.

[30] M. Ergezer, D. Simon, D. Du, Oppositional biogeography-based optimization, in: IEEE Conference on Systems, Man, and Cybernetics, San Antonio, TX, October 2009, pp. 1035–1040.

[31] A. Engelbrecht, Computational Intelligence, John Wiley & Sons, 2007.