In Thomas Back, editor, Proceedings of the Seventh International
Conference on Genetic Algorithms, San Mateo, CA: Morgan Kauffman, 1997

# A Random Function Based Framework for Evolutionary Algorithms

**Laurence D. Merkle**
Center for Plasma Theory and Computation
3550 Aberdeen Ave. SE
Phillips Laboratory
Kirtland AFB, NM 87117

**Gary B. Lamont**
Dept. of Electrical and Computer Engineering
Graduate School of Engineering
Air Force Institute of Technology
Wright-Patterson AFB, OH 45433

## Abstract

Evolutionary algorithms (EAs) are stochastic, population-based algorithms inspired by the natural processes of recombination, mutation, and selection. EAs are often employed as optimum seeking techniques. A formal framework for EAs is proposed, in which evolutionary operators are viewed as mappings from parameter spaces to spaces of random functions. Formal definitions within this framework capture the distinguishing characteristics of the classes of recombination, mutation, and selection operators. A specific EA, the generalized fast messy genetic algorithm, is defined within the proposed framework.

## 1 Introduction

Stochastic population-based algorithms inspired by recombination, mutation, and selection processes are *evolutionary algorithms (EAs)*. Well-known members of this class include genetic algorithms (GAs) [8], evolution strategies (ESs) [11, 12] and evolutionary programming (EP) [4]. Bäck [1] provides an excellent review of these three major EA paradigms, including a historical perspective. Bäck and Schwefel [2] propose a general specification for EAs, which they refine for each paradigm. The definitions of various mappings appearing therein are quite broad. So much so that they overlook essential operator characteristics. This research develops formal definitions (Sections 2 and 3) which capture these characteristics. Using these definitions, Section 4 extends Bäck and Schwefel's specification. Finally, the proposed framework is used to define a novel EA, the generalized fast messy genetic algorithm (Section 5).

## 2 Representation

Associated with each EA is a non-empty set $I$, called the *individual space* of the algorithm. When the EA is used as an optimum seeking technique, each *individual* $\mathbf{a} \in I$ represents a candidate solution to the optimization problem at hand. The representation scheme is formally defined by the *decoding function.*

**Definition 1 (Decoding function):** *Let $I$ be a non-empty set, and $f : \mathbb{R}^n \longrightarrow \mathbb{R}$ (the* objective function*). If $D : I \longrightarrow \mathbb{R}^n$ is total, i.e. the domain of $D$ is all of $I$, then $D$ is called a* decoding function. □

The mapping $D$ is not necessarily surjective; the range of $D$ determines the subset of $\mathbb{R}^n$ actually available for exploration by the evolutionary algorithm.

The *fitness* of an individual $\mathbf{a} \in I$ is an indication of the quality of the candidate solution $D(\mathbf{a}) \in \mathbb{R}^n$. The mapping which yields this indication is the *fitness function.* It is the fitness function which the evolutionary algorithm actually attempts to optimize.

**Definition 2 (Fitness function):** *Let $I$ be a non-empty set, $D : I \longrightarrow \mathbb{R}^n$, $f : \mathbb{R}^n \longrightarrow \mathbb{R}$, and $T_s : \mathbb{R} \longrightarrow \mathbb{R}$ (the* fitness scaling function. *Then $\Phi \overset{\triangle}{=} T_s \circ f \circ D$ is called a* fitness function. □

In this definition it is understood that the objective function $f$ is determined by the application, while the specification of the decoding function $D$ and the fitness scaling function $T_s$ are design issues.

Execution of an EA typically begins by randomly sampling with replacement from $I$. The resulting collection is the *initial population*, denoted $P(0)$. More generally, a *population* is a collection[1] $P = \{\mathbf{a_1}, \ldots, \mathbf{a}_\mu\}$ of individuals $\mathbf{a_i} \in I$. The number of individuals $\mu$ in the population is the *population size.*

[1]Populations are treated interchangeably as $n$-tuples of individuals or multisets of individuals, as convenient.

Following initialization, execution proceeds iteratively. Each iteration consists of application of one or more *evolutionary operators*. The combined effect of the evolutionary operators applied in a particular *generation* $t \in \mathbb{N}$ is to transform the current population $P(t)$ into a new population $P(t+1)$.

## 3 Evolutionary Operators

Bäck and Schwefel describe evolutionary operators (EOs) as directly mapping populations into populations, with the mapping being "controlled" by the parameters of the operator. This research proposes a more formal view of EOs as mappings from parameter spaces to random functions [3] with values in the set of population transformations. This view precisely identifies the relationships among the operator parameters and the various mappings.

A population transformation (PT) is any mapping from populations to populations (see Figure 1).

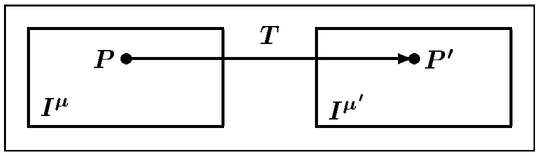


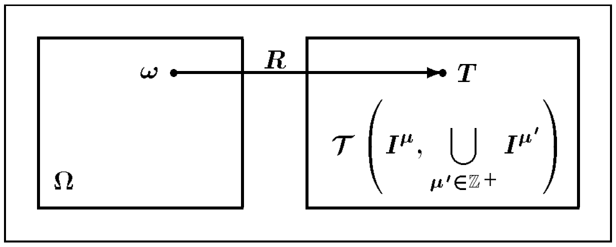
Figure 1: The PT $T$ deterministically maps the parent population $P$ (of size $\mu$) to the offspring population $P'$ (of size $\mu'$).

**Definition 3 (Population transformation):** *Let $I$ be a non-empty set, and $\mu, \mu' \in \mathbb{Z}^+$ (the* parent *and* offspring *population sizes, respectively). A mapping $T : I^\mu \longrightarrow I^{\mu'}$ is called a* population transformation*. If $T(P) = P'$ then $P$ is called a* parent population *and $P'$ is called an* offspring population*. If $\mu = \mu'$, then they are called simply the* population size*.* □

The PT resulting from an EO often depends on the outcome of a random experiment, thus motivating the concept of a random PT (RPT) (Figure 2). The set of mappings from $\mathcal{S}_1$ to $\mathcal{S}_2$ is denoted $\mathcal{T}(\mathcal{S}_1, \mathcal{S}_2)$.

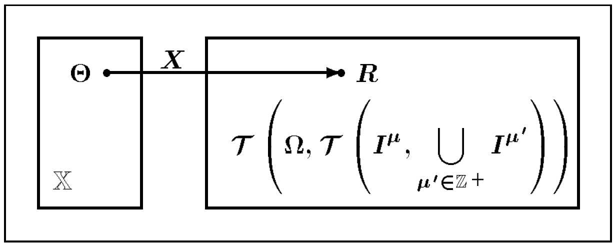
**Definition 4 (Random population transformation):** *Let $I$ be a non-empty set, $\mu \in \mathbb{Z}^+$, and $\Omega$ a set (the* sample space*). A random function $R : \Omega \longrightarrow \mathcal{T}\left(I^\mu, \bigcup_{\mu' \in \mathbb{Z}^+} I^{\mu'}\right)$ is called a* random population transformation*.* □

The distribution of PTs resulting from the application of an EO depends on the operator parameters, i.e. an EO maps its parameters to a RPT (Figure 3).



$\omega$
$R$
$T$
$\mathcal{T}\left(I^\mu, \bigcup_{\mu' \in \mathbb{Z}^+} I^{\mu'}\right)$
$\Omega$

Figure 2: The RPT $R$ maps the random event $\omega$ to the PT $T$, which maps parent populations of size $\mu$ (which is independent of $\omega$) to offspring populations of some fixed size $\mu' \in \mathbb{Z}^+$ (which may depend on $\omega$).



$\Theta$
$X$
$R$
$\mathcal{T}\left(\Omega, \mathcal{T}\left(I^\mu, \bigcup_{\mu' \in \mathbb{Z}^+} I^{\mu'}\right)\right)$
$\mathbb{X}$

Figure 3: The EO $X$ maps the exogenous parameter(s) $\Theta$ to the RPT $R$. The underlying sample space of $R$ is $\Omega$. Each of the possible PTs acts on populations of size $\mu$. The offspring population size $\mu' \in \mathbb{Z}^+$ may depend on $\Theta$ as well as the random event $\omega \in \Omega$.

**Definition 5 (Evolutionary operator):** *Let $I$ be a non-empty set, $\mu \in \mathbb{Z}^+$, $\mathbb{X}$ a set (the* parameter space*), and $\Omega$ a set. A mapping*

$$X : \mathbb{X} \longrightarrow \mathcal{T}\left(\Omega, \mathcal{T}\left(I^\mu, \bigcup_{\mu' \in \mathbb{Z}^+} I^{\mu'}\right)\right) \qquad (1)$$

*is called an* evolutionary operator*. The set of evolutionary operators in the form of Equation 1 is denoted $\mathcal{EVOP}(I, \mu, \mathbb{X}, \Omega)$.* □

The RPT $X(\Theta)$ is denoted $X_\Theta$. The PT $X_\Theta(\omega)$ is also denoted $X_\Theta$ to maintain consistency with the notation of Bäck and Schwefel, except where confusion may arise. In particular, the offspring population $[X_\Theta(\omega)](P)$ is denoted $X_\Theta(P)$. Finally, if $X$ has no parameters, i.e. $X \in \mathcal{EVOP}(I, \mu, \{\}, \Omega)$, then the offspring population is denoted $X(P)$.

The specific EOs used are typically biologically inspired. The guiding principle in their design is typically loose analogy to Darwin's principle of "survival of the fittest." The most commonly used EOs are *recombination*, *mutation*, and *selection*.

Recombination operators are the most general of the three. The distinguishing characteristic of recombination operators is that at least some of the individuals in the offspring population may depend on more than one individual in the parent population. The following definition reflects this characteristic. Because of this, it is more restrictive than the definition adopted by Bäck and Schwefel, which admits any PT $r : I^{\mu} \longrightarrow I^{\mu'}$ where $\mu, \mu' \in \mathbb{Z}^+$.

**Definition 6 (Recombination operator):** *Let $r \in \mathcal{EVOP}(I, \mu, \mathbb{X}, \Omega)$. If there exist $P \in I^{\mu}$, $\Theta \in \mathbb{X}$, and $\omega \in \Omega$ such that at least one individual in the offspring population $r_{\Theta}(P)$ depends on more than one individual of $P$ then $r$ is called a* recombination operator. □

In contrast to recombination operators, the distinguishing feature of mutation operators is that each of the individuals in the offspring population depends on at most one individual in the parent population.

**Definition 7 (Mutation operator):** *Let $m \in \mathcal{EVOP}(I, \mu, \mathbb{X}, \Omega)$. If for every $P \in I^{\mu}$, every $\Theta \in \mathbb{X}$, and every $\omega \in \Omega$, each individual in the offspring population $m_{\Theta}(P)$ depends on at most one individual of $P$ then $m$ is called a* mutation operator. □

This definition of mutation is more general than Bäck and Schwefel's, which assumes that parent and offspring population sizes are equal.

The distinguishing characteristics of selection operators are that every individual in the offspring population is also a member of the parent population, and that the PT depends on the fitnesses of the individuals in the parent population. The following definition reflects these characteristics, in contrast to Bäck and Schwefel's, which admits any PT $s : (I^{\mu'} \cup I^{\mu'+\mu}) \longrightarrow I^{\mu}$.

**Definition 8 (Selection operator):** *Let $s \in \mathcal{EVOP}(I, \mu, \mathbb{X} \times \mathcal{T}(I, \mathbb{R}), \Omega)$. If for every $P \in I^{\mu}$, every $\Theta \in \mathbb{X}$, and every fitness function $\Phi : I \longrightarrow \mathbb{R}$, $s$ satisfies $\mathbf{a} \in s_{(\Theta,\Phi)}(P) \Longrightarrow \mathbf{a} \in P$, then $s$ is called a* selection operator. □

## 4 Algorithmic Specification

The preceding definitions of the various types of evolutionary operators permit the following formal definition of an evolutionary algorithm, which extends that of Bäck and Schwefel [2]. The definition is general in the sense that essentially every EA in the literature may be shown to satisfy the definition through suitable choices of the individual space, population sizes, fitness function, termination criterion, evolutionary operators, and operator parameters. It is specific in the sense that every algorithm which satisfies the definition exhibits the essential characteristics typically associated with EAs.

**Definition 9 (Evolutionary algorithm):** *Let $I$ be a non-empty set (the individual space), $\{\mu^{(i)}\}_{i \in \mathbb{N}}$ a sequence in $\mathbb{Z}^+$ (the parent population sizes), $\{\mu'^{(i)}\}_{i \in \mathbb{N}}$ a sequence in $\mathbb{Z}^+$ (the offspring population sizes), $\Phi : I \longrightarrow \mathbb{R}$ a fitness function, $\iota : \bigcup_{i=1}^{\infty} (I^{\mu})^i \longrightarrow$ $\{\texttt{true},\texttt{false}\}$ (the termination criterion), $\chi \in \{\texttt{true},\texttt{false}\}$, $r$ a sequence $\{r^{(i)}\}$ of recombination operators $r^{(i)} : \mathbb{X}_r^{(i)} \longrightarrow \mathcal{T}\left(\Omega_r^{(i)}, \mathcal{T}\left(I^{\mu^{(i)}}, I^{\mu'^{(i)}}\right)\right)$, $m$ a sequence $\{m^{(i)}\}$ of mutation operators $m^{(i)} : \mathbb{X}_m^{(i)} \longrightarrow \mathcal{T}\left(\Omega_m^{(i)}, \mathcal{T}\left(I^{\mu'^{(i)}}, I^{\mu'^{(i)}}\right)\right)$, $s$ a sequence $\{s^{(i)}\}$ of selection operators $s^{(i)} : \mathbb{X}_s^{(i)} \times \mathcal{T}(I, \mathbb{R}) \longrightarrow \mathcal{T}\left(\Omega_s^{(i)}, \mathcal{T}\left(\left(I^{\mu'^{(i)} + \chi\mu^{(i)}}\right), I^{\mu^{(i+1)}}\right)\right)$, $\Theta_r^{(i)} \in \mathbb{X}_r^{(i)}$ (the* recombination parameters*), $\Theta_m^{(i)} \in \mathbb{X}_m^{(i)}$ (the* mutation parameters*), and $\theta_s^{(i)} \in \mathbb{X}_s^{(i)}$ (the* selection parameters*). Then the algorithm shown in Figure 4 is called an* evolutionary algorithm. □

```
$t := 0$;
initialize $P(0) := \{\mathbf{a_1}(0), \ldots, \mathbf{a}_{\mu}(0)\} \in I^{\mu^{(0)}}$;
while $(\iota(\{P(0), \ldots, P(t)\}) \neq \texttt{true})$ do
    recombine: $P'(t) := r^{(t)}_{\Theta_r^{(t)}}(P(t))$;
    mutate: $P''(t) := m^{(t)}_{\Theta_m^{(t)}}(P'(t))$;
    select:
        if $\chi$
            then $P(t+1) := s^{(t)}_{(\theta_s^{(t)}, \Phi)}(P''(t))$;
            else $P(t+1) := s^{(t)}_{(\theta_s^{(t)}, \Phi)}(P''(t) \cup P(t))$;
        fi
    $t := t + 1$;
od
```

Figure 4: Outline of an Evolutionary Algorithm

This definition differs from Bäck and Schwefel's in several ways. First, and most importantly, the population sizes, operators, and parameters are all represented as sequences, reflecting the fact that certain evolutionary algorithms use varying population sizes, use multiple phases of execution in which different operators are applied, and vary their parameters over the course of execution.

Another difference between the definitions is that in Figure 4, the termination condition $\iota$ depends on the set of populations $\{P(0), \ldots, P(t)\}$. Many evolutionary algorithms terminate after a fixed number of generations (corresponding to a termination criterion satisfying $\iota(\{P(0),\ldots,P(t)\}) = \texttt{true} \iff \mathrm{card}\,(\{P(0),\ldots,P(t)\}) > t_f$), or based on conditions involving populations previous to the current generation. Both definitions fail to include evolutionary algorithms which terminate based on conditions involving the number of function evaluations performed.

Two further differences are notational. The variable $\chi$ is introduced to preserve Bäck and Schwefel's explicit representation of selection operators which act on populations of size $\mu' + \mu$, as well as those which act on populations of size $\mu'$. In Bäck and Schwefel's definition, selection acts on the population $P''(t) \cup Q$, where $Q \in \{\{\}, P(t)\}$.

Finally, the fitness function is represented as a parameter of the selection operator. Consequently, explicit statement of the evaluation step is unnecessary.

## 5 Generalized Fast Messy GAs

The formal framework proposed in Section 4 permits precise specification of novel evolutionary algorithms. This section demonstrates this aspect of the framework. The proposed algorithm is an example of a linkage-friendly genetic algorithm [9], and shares the high-level structure and representation scheme (Section 5.1) of the fast messy genetic algorithm (fmGA) proposed by Goldberg, et al. [6]. Consequently, it is convenient to refer to the algorithm as the *generalized fast messy genetic algorithm (gfmGA)*.

In the gfmGA initialization phase, a *competitive template* is selected and an initial population is randomly generated. The gfmGA *primordial phase* uses the *probabilistic building block filtering (BBF)* (Section 5.3) and *binary tournament selection (BTS) with probabilistic thresholding* (Section 5.4) operators. Each is a novel generalization of the corresponding fmGA operator. The *juxtapositional phase* uses the *cut-and-splice* operator (Section 5.2), as well as the BTS with probabilistic thresholding operator.

### 5.1 Representation

Loci are represented explicitly and individuals are not necessarily of uniform length.

**Definition 10 (Linkage-friendly genetic algorithm (lfGA) individual space):** *Let $\mathcal{A}$ be a nonempty set (the* genic alphabet*), $\ell \in \mathbb{Z}^+$ (the* nominal string length*), $\mathcal{L} \triangleq \{1,\ldots,\ell\}$ (the* loci*), and $o \in \mathbb{R}$ such that $o \geq 1$ (the* overflow factor*). Then $I \triangleq \bigcup_{\lambda=0}^{\lfloor o\cdot\ell\rfloor} (\mathcal{A}\times\mathcal{L})^\lambda \simeq \bigcup_{\lambda=0}^{\lfloor o\cdot\ell\rfloor} (\mathcal{A}^\lambda \times \mathcal{L}^\lambda)$ is called an* lfGA individual space *over $\mathcal{A}$.* □

Each $a_i \in \mathcal{A}$ is an *allele*, each $l_i \in \mathcal{L}$ is a *locus* (plural *loci*), and each ordered pair $(a_i, l_i)$ is a *gene*. It is convenient to define the set of *length-$\lambda$ non-overspecified individuals*

$$I(\lambda) \triangleq \{(\mathbf{a},\mathbf{l}) = ((a_1,\ldots,a_\lambda),(l_1,\ldots,l_\lambda)) \in I : l_i = l_j \iff i = j\} \ . \quad (2)$$

An individual $(\mathbf{a},\mathbf{l})$ is *fully specified* if each locus occurs exactly once, i.e. if $(\forall i \in \mathcal{L})(\exists! j \in \mathcal{L})[l_j = i]$. The set of fully specified individuals is thus $I_F \triangleq I(\ell)$. Given a fully specified individual $\mathbf{c} = (\mathbf{b},\mathbf{m}) \in I_F$, referred to as a *competitive template*, the overlay mapping associates every individual $\mathbf{x} \in I$ (fully specified or otherwise) with an $\ell$-vector of alleles.

**Definition 11 (Overlay mapping):** *Let $I$ be an lfGA individual space over the genic alphabet $\mathcal{A}$ with nominal string length $\ell$, and $I_F \triangleq I(\ell)$ defined by Equation 2. The mapping $\Gamma : I \times I_F \longrightarrow \mathcal{A}^\ell$ such that for each $i \in \{1,\ldots,\ell\}$*

$$[\Gamma((\mathbf{a},\mathbf{l}),(\mathbf{b},\mathbf{m}))]_i \triangleq \begin{cases} a_j & \text{, if } j \triangleq \min\{k : l_k = i\} \text{ exists} \\ b_j & \text{, where } m_j = i \text{ , if } \forall k : l_k \neq i \end{cases}$$

*is called the* overlay mapping *for $I$.* □

The association of each individual $\mathbf{x} \in I$ with a vector of alleles via the overlay mapping may be thought of as the first step in assigning a fitness to $\mathbf{x}$. Subsequent steps include mapping the vector of alleles to the parameter space of the objective function, evaluation of the objective function, and possibly fitness scaling. The composition of these mappings is the lfGA fitness function.

**Definition 12 (Linkage-friendly genetic algorithm (lfGA) fitness function):** *Let $I$ be an lfGA individual space over the genic alphabet $\mathcal{A}$ with nominal string length $\ell$, $I_F \triangleq I(\ell)$ defined by Equation 2, $\Gamma : I \times I_F \longrightarrow \mathcal{A}^\ell$ the overlay mapping for $I$, $D : \mathcal{A}^\ell \longrightarrow \mathbb{R}^n$, $f : \mathbb{R}^n \longrightarrow \mathbb{R}$, $T_s : \mathbb{R} \longrightarrow \mathbb{R}$, and $\Phi \triangleq T_s \circ f \circ D \circ \Gamma : I \times I_F \longrightarrow \mathbb{R}$. Then $\Phi(\mathbf{x},\mathbf{c})$ denotes the* fitness *of $\mathbf{x} \in I$ with respect to $\mathbf{c} \in I_F$. Furthermore, given $\mathbf{c} \in I_F$ define $\Phi_\mathbf{c} : I \longrightarrow \mathbb{R}$ by $\Phi_\mathbf{c}(\cdot) \triangleq \Phi(\cdot,\mathbf{c})$. Then $\Phi_\mathbf{c}$ is called an* lfGA fitness function *for $I$.* □

Of course, an lfGA fitness function $\Phi_\mathbf{c}$ may be written as the composition $T_s \circ f \circ D_\mathbf{c} : I \longrightarrow \mathbb{R}$, where $D_\mathbf{c}(\cdot) \triangleq D(\Gamma(\cdot,\mathbf{c}))$. Thus, lfGA fitness functions are fitness functions in the sense of Definition 2.

## 5.2 Recombination

The *cut-and-splice operator* is a recombination operator acting on non-uniform length individuals, otherwise similar to single-point crossover. It is convenient to define the cut-and-splice operator in terms of the composition of distinct *cut* and *splice* operators.

A cut operator maps pairs of individuals (the *parents*) to 4-tuples of individuals (the *fragments*). For $\mathbf{a} = (a_1, \ldots, a_\lambda) \in (\mathcal{A} \times \mathcal{L})^\lambda$, the following definition denotes by $\mathbf{a}_{i:j}$ the fragment $(a_i, \ldots, a_j) \in (\mathcal{A} \times \mathcal{L})^{j-i+1}$, where $1 \leq i \leq j \leq \lambda$. Some fragments may be trivial, i.e. of length 0; these are denoted $\{\}$.

**Definition 13 (Cut operator):** *Let $I$ be an lfGA individual space, $\Omega \triangleq [0,1]^4$, $\omega \triangleq (X_a, X_b, \hat{Y}_a, \hat{Y}_b) \sim U(\Omega)$, and $\kappa : \mathbb{R} \longrightarrow \mathcal{T}(\Omega, \mathcal{T}(I^2, I^4))$ an evolutionary operator. If for every $p_c \in [0,1]$ (the* cut probability*), every $(\mathbf{a}, \mathbf{b}) \in (\mathcal{A}\times\mathcal{L})^{\lambda_a} \times (\mathcal{A}\times\mathcal{L})^{\lambda_b} \subseteq I^2$ (the* parents*), $Y_a \triangleq \lceil(\lambda_a - 1) \cdot \hat{Y}_a\rceil$ and $Y_b \triangleq \lceil(\lambda_b - 1) \cdot \hat{Y}_b\rceil$ (the* cut points*), $\kappa$ satisfies Equation 3 (Figure 5) then $\kappa$ is called a* cut operator. □

$$\kappa_{p_c}(\mathbf{a}, \mathbf{b}) \triangleq \begin{cases} (\mathbf{a}_{1:Y_a}, \mathbf{a}_{Y_a+1:\lambda_a}, \mathbf{b}_{1:Y_b}, \mathbf{b}_{Y_b+1:\lambda_b}) \ , \\ \quad \text{if } \lambda_a\lambda_b > 0,\ X_a \leq p_c, \text{ and } X_b \leq p_c \\ (\mathbf{a}_{1:Y_a}, \mathbf{a}_{Y_a+1:\lambda_a}, \mathbf{b}, \{\}) \ , \\ \quad \text{if } \ \lambda_a > 0,\ X_a \leq p_c, \text{ and} \\ \qquad \text{either } \lambda_b = 0 \text{ or } X_b > p_c \\ (\mathbf{a}, \mathbf{b}_{1:Y_b}, \mathbf{b}_{Y_b+1:\lambda_b}, \{\}) \ , \\ \quad \text{if } \ \lambda_b > 0,\ X_b \leq p_c, \text{ and} \\ \qquad \text{either } \lambda_a = 0 \text{ or } X_a > p_c \\ (\mathbf{a}, \mathbf{b}, \{\}, \{\}) \ , \\ \quad \text{if } \ \text{either } \lambda_a = 0 \text{ or } X_a > p_c, \text{ and} \\ \qquad \text{either } \lambda_b = 0 \text{ or } X_b > p_c \end{cases} \tag{3}$$

Figure 5: Cut operator

A splice operator maps 4-tuples of individuals (the *fragments*) to $n$-tuples of individuals (the *offspring*), where $n \in \{2, 3, 4\}$. In the following definition, if $\mathbf{a} = (a_1, \ldots, a_{\lambda_a}) \in (\mathcal{A} \times \mathcal{L})^{\lambda_a}$ and $\mathbf{b} = (b_1, \ldots, b_{\lambda_b}) \in (\mathcal{A} \times \mathcal{L})^{\lambda_b}$ are fragments, then the offspring $(a_1, \ldots, a_{\lambda_a}, b_1, \ldots, b_{\lambda_b})$ is denoted $\mathbf{ab}$.

**Definition 14 (Splice operator):** *Let $I$ be an lfGA individual space, $\Omega \triangleq [0,1]^3$, $\omega \triangleq (X_{ab}, X_{bc}, X_{cd}) \sim U(\Omega)$, and $\zeta : \mathbb{R} \longrightarrow \mathcal{T}(\Omega, \mathcal{T}(I^4, I^2 \cup I^3 \cup I^4))$ an evolutionary operator. If for every $p_s \in [0,1]$ (the* splice probability*), and every $(\mathbf{a}, \mathbf{b}, \mathbf{c}, \mathbf{d}) \in I^4$ (the* fragments*), $\zeta$ satisfies Equation 4 (Figure 6) then $\zeta$ is called a* splice operator. □

$$\zeta_{p_s}(\mathbf{a}, \mathbf{b}, \mathbf{c}, \mathbf{d}) \triangleq \begin{cases} (\mathbf{ab}, \mathbf{cd}) \ , \text{if } X_{ab} \leq p_s \text{ and } X_{cd} \leq p_s \\ (\mathbf{ab}, \mathbf{c}, \mathbf{d}) \ , \text{if } X_{ab} \leq p_s \text{ and } X_{cd} > p_s \\ (\mathbf{a}, \mathbf{bc}, \mathbf{d}) \ , \text{if } X_{ab} > p_s \text{ and } X_{bc} \leq p_s \\ (\mathbf{a}, \mathbf{b}, \mathbf{cd}) \ , \\ \quad \text{if } X_{ab} > p_s,\ X_{bc} > p_s, \text{ and } X_{cd} \leq p_s \\ (\mathbf{a}, \mathbf{b}, \mathbf{c}, \mathbf{d}) \ , \\ \quad \text{if } X_{ab} > p_s,\ X_{bc} > p_s, \text{ and } X_{cd} > p_s \end{cases} \tag{4}$$

Figure 6: Splice operator

A *local cut-and-splice operator* is an evolutionary operator which produces population transformations expressible as the composition of the population transformations resulting from a cut operator, a permutation of the resulting fragments (possibly depending on the parameters and random events of the cut operator), and a splice operator.

**Definition 15 (Local cut-and-splice operator):** *Let $I$ be an lfGA individual space, $\Omega \triangleq [0,1]^4 \times [0,1]^3$, $\omega \triangleq (\omega_c, \omega_s) \sim U(\Omega)$, $\kappa$ a cut operator, $\sigma : \mathbb{R} \times [0,1]^4 \longrightarrow \pi_4$, $\zeta$ a splice operator, and $r' : \mathbb{R}^2 \longrightarrow \mathcal{T}(\Omega, \mathcal{T}(I^2, I^2 \cup I^3 \cup I^4))$ an evolutionary operator. If $r'$ satisfies*

$$\begin{aligned} &[r'_{(p_c,p_s)}(\omega)](\mathbf{a}, \mathbf{b}) = \\ &\quad [\zeta_{p_s}(\omega_s)](([\kappa_{p_c}(\omega_c)](\mathbf{a}, \mathbf{b}))_{[\sigma(p_c,\omega_c)](1)}, \ldots, \\ &\quad ([\kappa_{p_c}(\omega_c)](\mathbf{a}, \mathbf{b}))_{[\sigma(p_c,\omega_c)](4)}) \ , \end{aligned}$$

*then $r'$ is called a* local cut-and-splice operator. □

The permutation mapping $\sigma$ in Definition 15 is arbitrary. Different mappings correspond to different local cut-and-splice operators and result in different sets of potential offspring. Goldberg, et al. [7] propose a local cut-and-splice operator, for which the potential sets of nontrivial offspring[2] are illustrated in Figure 7. The permutation mapping of Goldberg's local cut-and-splice operator is intended to closely resemble the behavior of single-point crossover for individuals of length close to the nominal string length.

**Definition 16 (Goldberg's local cut-and-splice operator):** *Let $I$, $\Omega$, $\omega$, $\kappa$, $\zeta$, and $r'$ be as in Defi-*

[2] In practice, only nontrivial individuals are included in the offspring population.

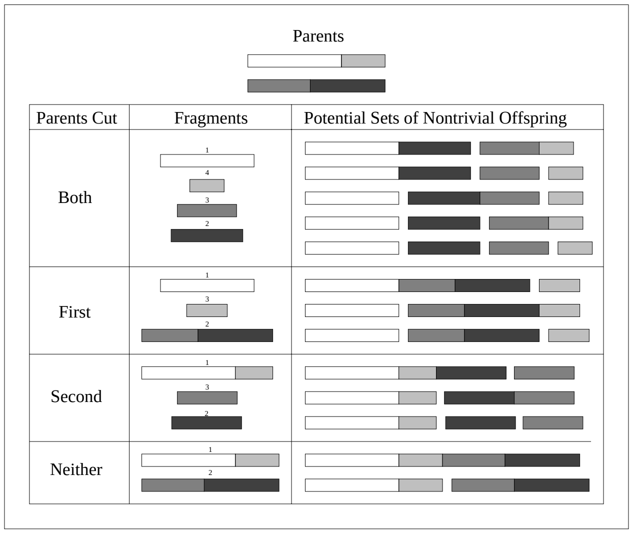


Figure 7: Potential Nontrivial Offspring Resulting From Goldberg's Local Cut-and-splice Operator

*nition 15, with* $\sigma : \mathbb{R} \times [0,1]^4 \longrightarrow \pi_4$ *defined by*

$$\sigma(p_c, \omega_c) \triangleq \begin{cases} (1,4,3,2) & \text{, if } X_a \le p_c \text{ and } X_b \le p_c \\ (1,2,3,4) & \text{, if } X_a > p_c \text{ and } X_b > p_c \\ (1,3,2,4) & \text{, otherwise} \end{cases} .$$

*Then* $r'$ *is called* Goldberg's local cut-and-splice operator. □

A *cut-and-splice operator* is an evolutionary operator which extends a local cut-and-splice operator to operate on populations of arbitrary size (i.e. a macro-operator corresponding to a local cut-and-splice operator). In contrast to the situation with single-point crossover, for which every pair of parents results in exactly two offspring, a local cut-and-splice operator probabilistically results in between 1 and 4 offspring for each pair of parents. Because of this uncertainty, it is convenient to recursively define the population produced by a cut-and-splice operator.

**Definition 17 (Cut-and-splice operator):** *Let* $I$ *be an lfGA individual space,* $\mu \in \mathbb{Z}^+$, $\mu' \in \mathbb{Z}^+$, $\xi \triangleq \lceil \frac{2\mu'}{\mu} \rceil$, $\Omega \triangleq \pi_\mu^\xi \times ([0,1]^4 \times [0,1]^3)^{\mu'}$, $\omega \triangleq ((\sigma_1, \ldots, \sigma_\xi), (\omega_1, \ldots, \omega_{\mu'})) \sim U(\Omega)$, $r'$ *a local cut-and-splice operator,* $r \in \mathcal{EVOP}(I, \mu, \mathbb{R}^2, \Omega)$, *and* $\hat{r}$ *satisfies Equation 5 (Figure 8). If for every* $p_c \in [0,1]$, *every* $p_s \in [0,1]$, *and every* $P \in I^\mu$, $r$ *satisfies* $r_{(p_c,p_s)}(P) = \hat{r}(\{\}; \mu, \mu, \xi, \mu'; P, p_c, p_s, \omega)$, *then* $r$ *is called a* cut-and-splice operator. *If* $r'$ *is Goldberg's local cut-and-splice operator, then* $r$ *is called* Goldberg's cut-and-splice operator. □

$$\hat{r}(P'; i_0, i, j, k; P, p_c, p_s, \omega) \triangleq \begin{cases} P' \quad \text{, if } k = 0 \\ \hat{r}(P'; i_0, i_0, j-1, k; P, p_c, p_s, \omega) \quad , \\ \quad \text{if } k > 0 \text{ and } i = 0 \\ \hat{r}\big( \quad P' \cup \{P_{\sigma_j(i)}\}; \\ \qquad i_0, i_0, j-1, k-1; \\ \qquad P, p_c, p_s, \omega \qquad \big) \quad , \\ \quad \text{if } k > 0 \text{ and } i = 1 \\ P' \cup \{Q_1, \ldots, Q_k\} \quad , \\ \quad \text{if } 0 < k < 4 \text{ and } i > 1 \\ \hat{r}\big( \quad P' \cup \{Q_1, \ldots, Q_{\dim \mathbf{Q}}\}; \\ \qquad i_0, i - \dim \mathbf{Q}, j, k - \dim \mathbf{Q}; \\ \qquad P, p_c, p_s, \omega \qquad \big) \quad , \\ \quad \text{if } k > 0 \text{ and } i > 1 \end{cases} \tag{5}$$

where $\mathbf{Q} = (Q_1, \ldots, Q_{\dim \mathbf{Q}})$ denotes the offspring $[r'_{(p_c,p_s)}(\omega_i)](P_{\sigma_j(i)}, P_{\sigma_j(i-1)})$ of an invocation of $r'$.

Figure 8: Cut-and-splice operator

### 5.3 Mutation

The gfmGA uses the probabilistic BBF operator. While the deterministic BBF operator of the fmGA deletes the same fixed number $\lambda^{(t)} - \lambda^{(t+1)}$ of genes from each individual in generation $t$, the probabilistic BBF operator adds or deletes a random number of genes, determined independently for each individual.

If $\lambda^{(t+1)} < \lambda^{(t)}$, the deleted genes (equivalently, the retained genes) are drawn uniformly without replacement from the individual's genes. If $\lambda^{(t+1)} > \lambda^{(t)}$, new loci are drawn uniformly without replacement from the set of loci for which the individual does not already contain a gene. New alleles are drawn uniformly and independently from the genic alphabet. Thus, the operator preserves the non-overspecified property of primordial phase individuals. It is convenient to define the probabilistic BBF operator in terms of the generalized local BBF operator.

**Definition 18 (Generalized local building block filtering operator):** *Let* $I$ *be an lfGA individual space over genic alphabet* $\mathcal{A}$ *with nominal string length* $\ell$,

$$\mathcal{S}_\pi \triangleq \left\{ \hat{\sigma} \in \mathcal{T}\left( \{0, \ldots, \ell\}, \bigcup_{i=1}^{\ell} \pi_i \right) : \hat{\sigma}(i) \in \pi_i \right\} \quad , \tag{6}$$

$\Omega \triangleq \mathcal{S}_\pi \times \mathcal{S}_\pi \times \mathcal{A}^\ell$, $\omega \triangleq (\hat{\sigma}_1, \hat{\sigma}_2, (\alpha_1, \ldots, \alpha_\ell)) \sim U(\Omega)$, *sort*$(\{\beta_1, \ldots, \beta_\lambda\}) \triangleq (\beta_{n_1}, \ldots, \beta_{n_\lambda})$ *such that*

$\beta_{n_1} < \ldots < \beta_{n_\lambda}$, $m' \in \mathcal{EVOP}(I, 1, \mathbb{N}, \Omega)$, *and* $\hat{m}$ *defined by Equation 7 (Figure 9). If for every* $\lambda_f \in \{0, \ldots, \ell\}$ *(the* offspring individual length*), and every* $\mathbf{a} = ((a_1, l_1), \ldots, (a_{\lambda_0}, l_{\lambda_0})) \in I$, $m'_{\lambda_f}(\mathbf{a}) = \hat{m}(\mathbf{a}, (\alpha_1, \ldots, \alpha_\ell), sort(\{1, \ldots, \ell\} - \{l_1, \ldots, l_{\lambda_0}\}), \hat{\sigma}_1(\lambda_0), \hat{\sigma}_2(\ell - \lambda_0), \lambda_f)$*, then* $m'$ *is a* generalized local building block filtering operator. $\Box$

$$\begin{aligned} \hat{m}\ [\ & ((a_1, l_1), \ldots, (a_{\lambda_0}, l_{\lambda_0})), \\ & (\alpha_1, \ldots, \alpha_\ell), (\beta_{\lambda_0+1}, \ldots, \beta_\ell), \sigma_1, \sigma_2, \lambda_f \ ] \\ \triangleq & \begin{cases} ((a_{\sigma_1(1)}, l_{\sigma_1(1)}), \ldots, (a_{\sigma_1(\lambda_f)}, l_{\sigma_1(\lambda_f)})) \\ \quad \text{if } \lambda_f \leq \lambda_0 \ , \\ ((a_{\sigma_1(1)}, l_{\sigma_1(1)}), \ldots, (a_{\sigma_1(\lambda_f)}, l_{\sigma_1(\lambda_f)}), \\ (\alpha_{\sigma_2(1)+\lambda_0}, \beta_{\sigma_2(1)+\lambda_0}), \ldots, \\ (\alpha_{\sigma_2(\lambda_f-\lambda_0)}, \beta_{\sigma_2(\lambda_f-\lambda_0)})) \\ \quad \text{if } \lambda_f > \lambda_0 \ , \end{cases} \end{aligned} \tag{7}$$

Figure 9: Probabilistic BBF operator

The number of genes added or deleted from a parent individual in generation $t \in \{0, \ldots, t_p - 1\}$ is determined by the offspring individual length $\lambda^{(t+1)}$, which is a random variable chosen according to $\psi^{(t)}(\lambda) \triangleq \Pr[\lambda^{(t+1)} = \lambda]$ where each $\psi^{(t)}(\lambda)$ is an exogenous *filtering parameter*. Because each $\psi^{(t)}$ is a probability density function of the discrete type, $\psi^{(t)}(0) \triangleq 1 - \sum_{\lambda=1}^{\ell} \psi^{(t)}(\lambda)$, and the $\psi^{(t)}(\lambda)$'s are subject to the constraints

$$\sum_{\lambda=1}^{\ell} \psi^{(t)}(\lambda) \leq 1 \text{ and } (\forall \lambda \in \{1, \ldots, \ell\})[\psi^{(t)}(\lambda) \geq 0] \ . \tag{8}$$

**Definition 19 (Probabilistic building block filtering operator):** *Let* $I$ *be an lfGA individual space over genic alphabet* $\mathcal{A}$ *with nominal string length* $\ell$, $\mu = \mu' \in \mathbb{Z}^+$, $\mathcal{S}_\pi$ *defined by Equation 6,* $\Omega \triangleq (\mathcal{S}_\pi \times \mathcal{S}_\pi \times \mathcal{A}^\ell)^{\mu'}$, $\omega \triangleq (\omega_1, \ldots, \omega_{\mu'}) \sim U(\Omega)$, $m'$ *a generalized local BBF operator, and* $m : [0,1]^\ell \longrightarrow \mathcal{T}(\Omega, \mathcal{T}(I^{\mu'}, (I(\lambda_f))^{\mu'}))$ *an evolutionary operator. If for every* $\psi \triangleq (\psi(1), \ldots, \psi(\ell))$ *satisfying Equations 8, every* $P \in I^\mu$*, and every* $i \in \{1, \ldots, \mu'\}$*,* $m$ *satisfies*

$$\Pr\Big\{[m_\psi(P)]_i = [m'_{\lambda_f}(\omega_i)](P_i)\Big\} = \begin{cases} \psi(\lambda_f) & , \text{ if } 1 \leq \lambda_f \leq \ell \\ 1 - \sum_{\lambda=1}^{\ell} \psi(\lambda) & , \text{ if } \lambda_f = 0 \end{cases} ,$$

*then* $m$ *is called a* probabilistic building block filtering operator. $\Box$

## 5.4 Selection

The gfmGA primordial phase uses the BTS with probabilistic thresholding operator. Competition is restricted to *compatible* individuals. Individuals $\mathbf{a} \in I(\lambda_a)$ and $\mathbf{b} \in I(\lambda_b)$ and sharing $\lambda_c = \Lambda_c(\mathbf{a}, \mathbf{b})$ common defining loci are compatible with probability $\theta^{(t)}(\lambda_c; \lambda_a, \lambda_b)$, where each $\theta^{(t)}(\lambda_c; \lambda_a, \lambda_b)$ is an exogenous *thresholding parameter*.

**Definition 20 (Binary tournament selection with probabilistic thresholding operator and finite shuffle size $n_{sh}$):** *Let* $I$ *be a non-empty set,* $\ell \in \mathbb{Z}^+$, $\mu \in \mathbb{Z}^+$, $\mu' \in \mathbb{Z}^+$, $n_{sh} \in \mathbb{Z}^+$ *(the* shuffle size*),* $\Omega \triangleq (\{1, \ldots, \mu\}^{n_{sh}})^{\mu'} \times [0,1]^{n_{sh} \times \mu'}$,

$$\begin{aligned} \omega \ \triangleq \ & ((\omega_0(1), \ldots, \omega_{n_{sh}}(1)), \ldots, \\ & (\omega_0(\mu'), \ldots, \omega_{n_{sh}}(\mu'), X) \sim U(\Omega) \ , \end{aligned}$$

$s \in \mathcal{EVOP}(I, \mu, \mathcal{T}(I^2, [0,1]) \times \mathcal{T}(I, \mathbb{R}), \Omega)$, *and* $j : \{1, \ldots, \mu'\} \times \mathcal{T}(I^2, [0,1]) \longrightarrow \{0, \ldots, n_{sh}\}$ *defined by*

$$j(i, \theta) \triangleq \begin{cases} 0 \ , \text{ if } (\forall k)[X_{ik} > \theta(P_{\omega_0(i)}, P_{\omega_k(i)})] \\ \min\{k : X_{ik} \leq \theta(P_{\omega_0(i)}, P_{\omega_k(i)})\} \ , \\ \quad \text{otherwise} \ . \end{cases}$$

*If for every* $\theta : I^2 \longrightarrow [0,1]$ *(the* threshold mapping*), every fitness function* $\Phi : I \longrightarrow \mathbb{R}$*, and every population* $P \in I^\mu$*,* $s$ *satisfies*

$$[s_{(\theta,\Phi)}(P)]_i = \begin{cases} P_{\omega_0(i)} & , \text{ if } \Phi(P_{\omega_0(i)}) \geq \Phi(P_{\omega_{j(i,\theta)}(i)}) \\ P_{\omega_{j(i,\theta)}(i)} & , \text{ otherwise} \ , \end{cases}$$

*then* $s$ *is called a* binary tournament selection with thresholding operator. $\Box$

## 5.5 Algorithmic Specification

The preceding sections describe the novel operators used by the gfmGA. This section specifies the gfmGA in the formal framework of Sections 2 and 3. The specification, facilitated by the framework defined in Section 4, is concise and unambiguous.

**Definition 21 (Generalized fast messy genetic algorithm):** *Let* $I$ *be an lfGA individual space over the genic alphabet* $\{0,1\}$ *with nominal string length* $\ell$ *and overflow factor* $o$*,* $I(\lambda)$ *defined by Equation 2,* $k \in \{1, \ldots, \ell\}$ *(the* building block size*),* $t_f \in \mathbb{Z}^+$ *(the* final generation*),* $t_p \in \{0, \ldots, t_f\}$ *(the* final primordial phase generation*),* $\lambda^{(0)} \triangleq \ell - k$ *(the* initial individual length*),* $\psi$ *a sequence* $\{\psi^{(t)}\}_{t=0}^{t_p} \subset [0,1]^\ell$ *satisfying Equations 8 (the* filtering parameters*),* $\alpha \in [0,1]$

*(the* probability of selection error*), $z_\alpha \in \mathbb{R}$ such that* $Z \sim N(0,1) \Longrightarrow \Pr[Z \geq z_\alpha] = 1 - \alpha$, $\beta^2 \in \mathbb{R}^+$ *(the maximum* inverse signal-to-noise ratio per subfunction *to be detected), $\mu = \mu' \triangleq \frac{\ell!(\ell-2k)!}{(\ell-k)!^2} 2 z_\alpha^2 \beta^2 (\lceil \frac{\ell}{k} \rceil - 1) 2^k$ (the* population size*), $\mathbf{c} \in I_F \triangleq I(\ell)$ (the* competitive template), $\Phi_{\mathbf{c}} : I \longrightarrow \mathbb{R}$ *is an lfGA fitness function, $\iota : \bigcup_{i=1}^{\infty} (I^\mu)^i \longrightarrow \{\texttt{true},\texttt{false}\}$ (the termination criterion) such that $\iota(\{P(0), \ldots, P(t)\}) = \texttt{true} \Longleftrightarrow card(\{P(0), \ldots, P(t)\}) > t_f$, $r$ a sequence $\{r^{(t)}\}$ of Goldberg's cut-and-splice operators $r^{(t)} : \mathbb{R}^2 \longrightarrow \mathcal{T}\left(\Omega_r^{(t)}, \mathcal{T}\left(I^{\mu^{(t)}}, I^{\mu^{(t)}}\right)\right)$, $m$ a sequence $\{m^{(t)}\}$ of evolutionary operators, for $0 \leq t < t_p$, $m^{(t)} : [0,1]^\ell \longrightarrow \mathcal{T}\left(\Omega_m^{(t)}, \mathcal{T}\left(I^{\mu^{(t)}}, I^{\mu^{(t)}}\right)\right)$ a probabilistic BBF operator, for $t_p \leq t < t_f$, $m^{(t)}$ an identity evolutionary operator, $s$ a sequence $\{s^{(t)}\}$ of BTS with probabilistic thresholding operators $s^{(t)} : \mathcal{T}(I^2, [0,1]) \times \mathcal{T}(I, \mathbb{R}) \longrightarrow \mathcal{T}\left(\Omega_s^{(t)}, \mathcal{T}\left(I^{\mu^{(t)}}, I^{\mu'^{(t)}}\right)\right)$, $\Theta_m^{(t)} \triangleq \lambda^{(t)}$ for $0 \leq t < t_p$ (the* filtering parameters*), $p_c^{(t)} = p_s^{(t)} \triangleq 0$ for $0 \leq t < t_p$, $\Theta_r^{(t)} \triangleq (p_c^{(t)}, p_s^{(t)}) \in \mathbb{R}^2$ for $0 \leq t < t_f$ (the* cut-and-splice parameters*), and $\theta$ a sequence $\{\theta^{(t)}\}$ of threshold mappings $\theta^{(t)} : I^2 \longrightarrow [0,1]$. Then the algorithm shown in Figure 4 is called a* generalized fast messy genetic algorithm. □

Building on the recommendation of Goldberg, et al. regarding the mGA [7], the algorithm may be applied iteratively for $2 \leq k \leq k_{max}$. The objective of iteration $k$ is to identify an order-$k$ optimal individual, given an order-$(k-1)$ optimal competitive template $\mathbf{c}$. The optimality condition on $\mathbf{c}$ for the second iteration ($k = 2$) is order-1, which may be satisfied efficiently by hillclimbing in $I(\ell)$.

## 6 Conclusions and Recommendations

The concepts developed in this research include population transformations, random population transformations, and general evolutionary operators, as well as recombination, mutation, and selection operators. The development results in a general yet precise formal framework for the class of evolutionary algorithms (EAs). The generalized fast messy genetic algorithm is defined in the context of this framework, demonstrating the usefulness of the framework in the precise specification of novel EAs.

Although it has not been done here, the framework may be used to prove relationships between various EAs. For example, it may be used to prove that certain EAs are equivalent, or that one EA is a special case of another.

## References

[1] Thomas Bäck. *Evolutionary Algorithms in Theory and Practice.* Oxford University Press, New York, 1996.

[2] Thomas Bäck and Hans-Paul Schwefel. An overview of evolutionary algorithms for parameter optimization. *Evolutionary Computation*, 1(1):1–23, 1993.

[3] A. Blanc-Lapierre and R. Fortet. *Theory of Random Functions.* Gordon and Breach, New York, 1965.

[4] L. J. Fogel, A. J. Owens, and M. J. Walsh. *Artificial Intelligence Through Simulated Evolution.* Wiley Publishing, New York, 1966.

[5] Stephanie Forrest, editor. *Proceedings of the Fifth International Conference on Genetic Algorithms*, San Mateo CA, July 1993. Morgan Kaufmann Publishers, Inc.

[6] David E. Goldberg, Kalyanmoy Deb, Hillol Kargupta, and Georges Harik. Rapid, accurate optimization of difficult problems using fast messy genetic algorithms. In Forrest [5], pages 56–64.

[7] David E. Goldberg, B. Korb, and K. Deb. Messy genetic algorithms: Motivation, analysis, and first results. *Complex Systems*, 3:493–530, 1989.

[8] John H. Holland. *Adaptation in Natural and Artificial Systems.* MIT Press, Cambridge, MA, First MIT Press edition, 1992.

[9] Laurence D. Merkle. *Analysis of Linkage-Friendly Genetic Algorithms.* PhD thesis, Graduate School of Engineering, Air Force Institute of Technology (AU), Wright-Patterson AFB OH 45433, December 1996.

[10] Gregory J. E. Rawlins, editor. *Foundations of Genetic Algorithms.* Morgan Kaufmann, San Mateo, CA, 1991.

[11] Ingo Rechenberg. *Evolutionsstratrategie: Optimierung Technischer Systeme nach Prinzipien der Biologischen Evolution.* Frommann-Holzboog, Stuttgart, 1973.

[12] Hans-Paul Schwefel. *Numerische Optimierung von Computer-Modellan mittels der Evolutionsstrategie.* Wiley, New York, 1981.