Classification with Scaled Genetic Algorithms in a Coevolutionary Setting

Lothar M. Schmitt

The University of Aizu, Aizu-Wakamatsu City, Fukushima Pref. 965-8580, Japan
lothar@u-aizu.ac.jp

Abstract. This work discusses asymptotic convergence of scaled genetic algorithms in a coevolutionary setting where the underlying population contains fixed numbers of creatures of various types. These types of creatures can act on each other in cooperative or competitive manner. The genetic algorithm uses common mutation and crossover operators as well as proportional fitness selection. By a scaled genetic algorithm, we mean that the mutation and crossover rates have to be annealed to zero in proper fashion over the course of the algorithm, and power-law scaling is used for the fitness function with (unbounded) logarithmic growth in the exponent. In the case that a scaled (non-elitist, non-memory) genetic algorithm is used as function optimizer, it has been shown [Theoret. Comput. Sci. 310, p. 181] that such an algorithm is able to find global optima asymptoticly with probability one. However, in the case of a coevolutionary setting, global optima need not exist. Based upon a properly defined, population-dependent fitness function, the set of creatures of a specific type can be canonically grouped into equivalence classes such that all members of one equivalence class are either inferior or superior to all members of another class. We show that in the situation of a coevolutionary setting, a scaled genetic algorithm at least retains the property of converging to a probability distribution over such populations that contain only copies of one creature from the *top-class* for every or a selected group of types while on the other hand maintaining a noisy field of test-cases.

1 Introduction

This work discusses a scaled, non-elitist, non-memory genetic algorithm in a coevolutionary setting that, in principle, is able to find global optima asymptotically. The algorithm, which is a significant extension of algorithms described in [21,22,24] is new in that: (1) it allows for parts of the population to be optimized while other parts remain "noisy" and provide for a "challenging" environment in which the optimization takes place; and (2) some "unnatural" condition on the fitness function used in [21,22] and, in case of a coevolutionary interpretation, also in [24, Thm. 3.3.2, Cors. 3.3.3–4] is removed such that a classification-algorithm is obtained in a very general situation.

1.1. Optimization Tasks in a Coevolutionary Setting. In order to give a precise account what is to be optimized in a coevolutionary setting in contrast to,

K. Deb et al. (Eds.): GECCO 2004, LNCS 3103, pp. 138–149, 2004.

e.g., investigations of population dynamics as in [8,12], let us start the discussion by listing a few examples:

Example 1.1.1 (Single-species competitive setting): A typical example for a single-species setting where a coevolutionary optimization is performed is that of deterministic game-playing strategies $\mathcal{C}$. Such strategies (agents, programs) are supposed to be encoded here as strings of symbols over a finite alphabet[1] $\mathcal{A}$ which are bounded in length with fixed upper bound $\ell \in \mathbb{N}$. If $c, d \in \mathcal{C}$ are game-playing strategies, then they determine a number $<c, d> \in \{0, \pm 1\}$ depending upon the outcome of the game. We set $<c, d> = -1$, if c wins; $<c, d> = 0$ in case of a draw; $<c, d> = 1$, if d wins. The simplest definition for a positive, population-dependent fitness function $f(c, p)$, —where p is a population in which c resides—, is then given by:

$$f(c, p) = \sum_{c \neq d \in p} (1 - <c, d>). \tag{1}$$

While there may be no strategy $c \in \mathcal{C}$ that is *not inferior to all other* strategies in the above setting (*i.e.*, a global maximum), there likely exists a minimal subset $\mathcal{C}^{\max} \neq \mathcal{C}$ of "superior" strategies such that for every $c \in \mathcal{C}^{\max}$, $d \notin \mathcal{C}^{\max}$ and every population p, one has $f(c, p) > f(d, p)$, *i.e.*, elements of $\mathcal{C}^{\max}$ are in any population p superior to elements not in $\mathcal{C}^{\max}$. Our approach to optimization in this situation is to search for elements of $\mathcal{C}^{\max}$. What we shall outline below is a scaled genetic algorithm that asymptotically finds elements of $\mathcal{C}^{\max}$ essentially without further uncontrollable assumptions on f or the coevolutionary setting.

Example 1.1.2 (Two-species competitive setting): A typical example for a two-species competitive setting is obtained, if we distinguish in Example 1.1.1 between first-move strategies $\mathcal{C}_1$ and second-move strategies $\mathcal{C}_2$. For $c_1 \in \mathcal{C}_1$ and $c_2 \in \mathcal{C}_2$ the fitness function can then be defined as:

$$f(c_1, p) = \exp(\gamma_1 \sum_{c_2 \in p \cap \mathcal{C}_2} <c_1, c_2>), \qquad \gamma_1 = -1, \text{ and} \tag{2}$$

$$f(c_2, p) = \exp(\gamma_2 \sum_{c_1 \in p \cap \mathcal{C}_1} <c_1, c_2>), \qquad \gamma_2 = 1. \tag{3}$$

We assume here that the number of elements $s_j > 0$ of creatures $c_j \in p \cap \mathcal{C}_j$ is fixed for $j = 1, 2$ over the course of the algorithm, *i.e.*, we do not allow for population-dynamics.

Another example for a two-species, adversary setting would be to measure the performance (*i.e.*, execution time) $<c_1, c_2>$ of sorting programs[2] $c_1 \in \mathcal{C}_1$ acting on unsorted tuples (test-instances) $c_2 \in \mathcal{C}_2$ of finite length $\ell_2 \in \mathbb{N}$. Here, we can define the fitness function as in lines (*2*) and (*3*). Sorting programs $c_1 \in \mathcal{C}_1$ aim for short execution times while unsorted tuples $c_2 \in \mathcal{C}_2$ aim for long execution times.

---

[1] A larger alphabet is favorable in several regards: (1) The length of creatures stays smaller; consequently, the population-size can stay smaller (*cf.*, lines (*22,24,26*)) and the fitness-evaluation is computationally less extensive (*cf.*, lines (*2*) and (*3*) and the table in Sec. 2.2). (2) The mutation rate can be annealed to zero by a faster schedule (*cf.*, Def. 3.2.2). (3) The fitness-function can be exponentiated at a *slower* rate (*cf.*, lines (*23,25,27*)) allowing the mixing operators more time in exploring the search space.

[2] Programs of length bounded by a fixed $\ell_1 \in \mathbb{N}$.

Example 1.1.3 (Two-species cooperative setting): A typical example for a two-species cooperative setting is the simultaneous performance-optimization of interacting components of a system. Specifically, one may be interested in optimizing the performance of a mechanical robot by computer-simulation in regard to minimizing energy $E = <\cdot,\cdot>$ of motion. For example on the hardware-side $\mathcal{C}_1$, one could optimize the layout or placement of parts. See [25] for application of a genetic algorithm to such a setting. For example on the software-side $\mathcal{C}_2$, one could be interested in finding a neural network [29] which is optimized[3] to control/correct certain aspects of the motion but is required to be "robust", *i.e.*, handle various types of situations uniformly well. Here, we can define the fitness function as in lines (*2*) and (*3*) with $\gamma_1 = \gamma_2 = -1$.

In the settings of both Examples 1.1.2 and 1.1.3, we intend to find "superior" elements in the *top-classes* $\mathcal{C}_1^{\max} \subset \mathcal{C}_1$ and $\mathcal{C}_2^{\max} \subset \mathcal{C}_2$ similar to the case of Example 1.1.1 (see also footnote 4). Altogether, we have:

Consider a coevolutionary setting as in the above examples 1.1.2 and 1.1.3 where creatures of different types of species $\mathcal{C}_j$ interact via an $\mathbb{R}$-valued duality or evaluation procedure $<\cdot,\cdots>$. Such a setting gives rise to a species- and population-dependent fitness function f in a canonical way as in lines (2) and (3). No global optima for f may exist within some or all of the species $\mathcal{C}_j$. In this presentation, we shall aim to describe a dynamically scaled genetic algorithm that finds creatures (candidate solutions) that are "optimal" in a simple and natural sense without uncontrollable restrictions on the coevolutionary setting.

1.2. Optimization with Genetic Algorithms. Genetic algorithms, a particular case of evolutionary algorithms, were invented by Holland [11] and are by now a well-established tool for search and optimization. Techniques using the operational framework of genetic algorithms have significant applications in mechanical and electrical engineering such as, *e.g.*, the construction of the turbine for the engine of the Boeing 777 airplane [26, discussion of p. 2357–8] or design of electric circuits [15] and antennas [16,26]. The common usage of a genetic algorithm as function optimizer for a (fitness-)function $f : \mathcal{C} \to \mathbb{R}^+$ as described, *e.g.*, in [10,27] or [20,24] is as follows: first a population p is initiated as an s-tuple of creatures $c_1, \ldots c_s \in \mathcal{C}$ ($s \in 2\mathbb{N}$), then three operations —crossover, mutation, and fitness-selection— are applied cyclically and iteratively to the creatures in the population until a termination condition is satisfied. The combined crossover-mutation phase of a genetic algorithm is also called the mixing-phase of the algorithm, *cf.* [27, p. 32]. Crossover is inspired by exchange of genetic information in living organisms, *e.g.*, during the process of sexual reproduction. Mutation is inspired by random change of genetic information in living organisms, *e.g.*, through the effects of radiation or chemical mismatch. Fitness selection models increased reproductive success of organisms c that are better adapted to their environment (*i.e.*, have higher value $f(c)$). Usually, fitness selection includes a random arrangement of selected creatures/individuals in the population.

[3] We note that the often-used back-propagation algorithm for neural networks [29, p. 187] is a gradient method which is not guaranteed to find a global minimum.

The introduction of [24, pp. 183-4, 186-7] lists a collection of references in regard to theoretical approaches to asymptotic convergence of genetic algorithms. Most of these approaches require satisfaction of some auxiliary condition on the algorithm or problem-setting in order to achieve convergence such as using the elitist strategy [18] or an infinite population limit [27, p. 147]. In [24, Thm. 3.3.2, Cors. 3.3.3-4] asymptotic convergence to global optima is shown for a genetic algorithm with arbitrary fitness function that satisfies *all* goals (1)–(4) formulated in [4, p. 270] and is much in the spirit of the simulated annealing algorithm. In particular, the scaled genetic algorithm described in [24] operates with a small population-size s that can be set by the user as low as $\ell+1$ (ℓ length of creatures). In addition, the techniques and results in [24] essentially solve the single-species coevolutionary optimization problem as described in Example (1.1.1) which, however, shall be considerably extended below.

1.3. De Jong's Challenge. In [7], the need for a theoretical framework for coevolutionary genetic algorithms and possible convergence theorems in regard to coevolutionary optimization ("arms races") was emphasized. Such a theoretical framework requires, in particular, treatment of a population-dependent fitness function. While there is a substantial amount of work in: (1) practical applications, and (2) experiments with the setting of coevolutionary algorithms, theoretical advance in regard to convergence seems to be limited. Some static aspects of coevolutionary genetic algorithms in regard to the order on creatures (*i.e.*, in our notation elements of $\mathcal{C}_1$) and test-instances (*i.e.*, elements of $\mathcal{C}_2$) have been investigated, *e.g.*, in [1,2]. Work in [6] discusses convergence of the evaluation procedure towards an ideal evaluation function. A thorough investigation of complexity/convergence of a $(1+1)$ coevolutionary algorithm without crossover for maximization of pseudo-Boolean functions can be found in [14]. However, there seems to be no black-box-scenario, coevolutionary global optimization theorem in the literature except results in [20, Thm. 8.6, Rem. 8.7] for a population-dependent fitness function with single dominant creature in a single-species setting, [24, Thm. 3.3.2, Cors. 3.3.3-4] for a population-dependent fitness function with a *'set of strictly dominant creatures of equal fitness'* in a single-species setting, and —already inspired by [7]— results with distinct crossover operators in [21,22] for a population-dependent fitness function with a *'set of strictly dominant creatures of equal fitness for every species that is optimized'* in a multi-species setting.

The new algorithm described below addressees, in particular, the following two problems:

1. The condition of a *'set of strictly dominant creatures of equal fitness for every species that is optimized'* shall be removed. Thus, the algorithm shall asymptotically deliver creatures from the *top equivalence class(es)*[4] pertaining to the fitness function with probability one. This addresses and solves (or circumvents) the problem of *intransitivity superiority* [28, p. 702] in a reasonable manner.

[4] The fitness function f defines a canonical relation $\leq_f$ on every type of species: $c \leq_f d \Leftrightarrow \exists p\colon f(c,p) \leq f(d,p)$. If $\lessdot_f$ denotes the transitive closure of $\leq_f$, then the equivalence class of c is given by $[c]=\{d : c \lessdot_f d, \text{ and } d \lessdot_f c\}$.

2. Part of the population is kept "noisy" driven by constant, non-zero mutation. This part need not be governed by a selection mechanism that converges against the elitist strategy. This allows for maintenance of a larger diverse collection of good "test-instance" within the population. In principle, techniques proposed in [5], [6], and [9] could be incorporated in a properly designed enhancement/selection mechanism acting on the noisy part of the population.

2 Alphabets, Creatures, and Populations

In what follows, we shall restrict ourselves to the case of two species and shall leave consideration of a coevolutionary setting for more species to the reader along the discussion in [21,22]. Note also, that we do not treat here the version of mixing put forward in [27], which produces only one child per mixing operation. However, [21,22] and [24, Sec. 4.3] discuss how to incorporate this mixing in the present framework by doubling the population-size s of the model for the algorithm and considering "selector masks" for selection.

2.1. Alphabets and Creatures. Suppose that two disjoint alphabets

$$\mathcal{A}_j = \{a_j(0), a_j(1) \dots a_j(\alpha_j - 1)\}, \qquad j{=}1, 2, \tag{4}$$

are given which are used to encode creatures in $\mathcal{C}_j$ as strings over $\mathcal{A}_j$ of length ℓ_j, *i.e.*, $\mathcal{C}_j{=}(\mathcal{A}_j)^{\ell_j}$, $j{=}1, 2$.

As outlined in [20, Sec. 3.1], it may be advantageous to consider larger alphabets representing finite, equidistant sets of real numbers in applications where real parameters are optimized in a compact domain of $\mathbb{R}^{\ell_j}$. Such a point of view is also supported in [17]. In addition, see footnote 1. Let $\mathcal{V}_j^{(1)} \equiv \mathbb{C}^{\alpha_j}$ be the free vector space over $\mathcal{A}_j$. Let $\mathcal{A}{=}\mathcal{A}_1 \cup \mathcal{A}_2$ be the combined set of letters considered here.

Let $n_j{\in}\mathbb{N}$ such that $n_j < \alpha_j/2$. We shall say that $a_j(\iota), a_j(\iota'){\in}\mathcal{A}_j$ are *close neighbors*, if $\iota{\neq}\iota'$ and $\min\{|\iota - \iota'|, \alpha_j - |\iota - \iota'|\} \leq n_j$.

Let $\mathcal{C}{=}\mathcal{C}_1 \cup \mathcal{C}_2$ be the set of all possible creatures considered here. In contrast to, *e.g.*, [5], creatures (phenotypes) are identified with their genetic information (genotypes). We suppose that an evaluation procedure or duality

$$<\cdot, \cdot> : \mathcal{C}_1 \times \mathcal{C}_2 \to \mathbb{R} \tag{5}$$

exists which lets creatures of different types interact and gives rise to a fitness function f defined in lines (*2*) and (*3*) above.

2.2. Populations. Let $s_1, s_3{\in}2\mathbb{N}{+}2$, $s_2, s_4{\in}2\mathbb{N}_0$. Let $\wp_{2j-n} = (\mathcal{C}_j)^{s_{2j-n}}$ for $j{=}1, 2$, $n{=}0, 1$. The set of populations is now given by

$$\wp = \wp_1 \times \wp_2 \times \wp_3 \times \wp_4. \tag{6}$$

Thus, $s{=}s_1{+}s_2{+}s_3{+}s_4$ is the number of creatures in a population. The length of a population as word over $\mathcal{A}$ is given by $L{=}(s_1{+}s_2)\ell_1{+}(s_3{+}s_4)\ell_2$.

We shall call the population $p = (p_1, p_2, p_3, p_4)$ '*multi-uniform at dedicated positions*' or '**partly uniform**' for short, if the sub-populations p_1 and p_3 contain only copies of a single creature $c_1{\in}\mathcal{C}_1$ and $c_2{\in}\mathcal{C}_2$ respectively.

Let $\mathcal{V}_\wp = \mathcal{V}(\wp)$ be the free complex vector space over $\wp$. The defintion of $\wp$ in line (*6*) yields a canonical tensor-product decomposition of $\mathcal{V}_\wp$:

$$\mathcal{V}_\wp = \mathcal{V}(\wp_1) \otimes \mathcal{V}(\wp_2) \otimes \mathcal{V}(\wp_3) \otimes \mathcal{V}(\wp_4). \tag{7}$$

Let $\mathcal{S}_\wp$ be the set of probability distributions over $\wp$ which is the positive part of the unit sphere of $\mathcal{V}_\wp$ with respect to the ℓ^1-norm.

Let $\mathcal{U} \subset \mathcal{V}_\wp$ be the free vector space over all populations which are partly uniform. Thus, $\wp \cap \mathcal{U}$ is the set of partly uniform populations. In addition, $P_\mathcal{U}$ shall denote the orthogonal projection onto $\mathcal{U}$.

A population p and the action of the genetic algorithm described below are then structured as shown in the following table:

Popul. p=	p_1	p_2	p_3	p_4
Creatures in	$\mathcal{C}_1$	$\mathcal{C}_1$	$\mathcal{C}_2$	$\mathcal{C}_2$
Position σ of creatures	$1 \dots s_1$	$s_1{+}1 \dots$ $s_1{+}s_2$	$s_1{+}s_2{+}1 \dots$ $s_1{+}s_2{+}s_3$	$s_1{+}s_2{+}s_3{+}1 \dots$ s
Position $\hat{\lambda}$ of letters	$1 \dots s_1\ell_1$	$s_1\ell_1{+}1 \dots$ $(s_1{+}s_2)\ell_1$	$(s_1{+}s_2)\ell_1{+}1 \dots$ $(s_1{+}s_2)\ell_1{+}s_3\ell_2$	$(s_1{+}s_2)\ell_1{+}s_3\ell_2{+}1$ $\dots L$
Purpose	**optimize**	noisy test-set	**optimize**	noisy test-set
Mutation rate	scaled $\mu{\to}0$	constant $\mu'{>}0$	scaled $\mu{\to}0$	constant $\mu'{>}0$
Crossover rate	scaled $\chi{\to}0$	constant $\chi'{>}0$	scaled $\chi{\to}0$	constant $\chi'{>}0$
Fitness evaluation	unbounded scaled fitness-prop.	any standard method	unbounded scaled fitness-prop.	any standard method

3 The Genetic Operators

The genetic operators used in this exibition allow for (1) multi-spot mutation $M_{\hat{\mu},\mu}$ with a local action (*i.e.*, the spot mutation matrix) on the alphabet level that implements a localized search; (2) practically any know crossover operator $C(\chi)$; and (3) selection S_t with scaled fitness-proportional selection on the parts of the population that the user wants to optimize.

3.1. Spot Mutation Matrices. Let $\hat{\mu}{\in}[0,1]$. For $1 \le \hat{\lambda} \le L$, the spot mutation matrices $\mathbf{m}_j(\hat{\mu})$ model *change* within the alphabets $\mathcal{A}_j$ for the letter $a_j(\iota)$ at single spot $\hat{\lambda}$ in the combined genome of a population ($j{=}1,2$). Let $a_j(\iota), a_j(\iota'){\in}\mathcal{A}_j$ such that $\iota{\neq}\iota'$, $0 \le \iota, \iota' \le \alpha_j{-}1$. In what follows, we shall use the stochastic matrix $\mathbf{m}_j(\hat{\mu})$ with zero-entries on the diagonal given by:

$$<a_j(\iota'), \mathbf{m}_j(\hat{\mu}) a_j(\iota)> \; = (1-\hat{\mu})/(2n_j) + \hat{\mu}/(\alpha_j{-}1), \tag{8}$$

if $a_j(\iota')$ and $a_j(\iota)$ are close neighbors in the sense of Sec. 2.1, and otherwise

$$<a_j(\iota'), \mathbf{m}_j(\hat{\mu}) a_j(\iota)> \; = \hat{\mu}/(\alpha_j{-}1). \tag{9}$$

Discussion of other choices for $\mathbf{m}_j(\hat{\mu})$ is left to the reader. Local change determined by $\mathbf{m}_j(\hat{\mu})$ is a continuous function of $\hat{\mu}$. The case $\hat{\mu}{=}0$ corresponds to

uniform change within the preferred "small set of close neighbors" of the current letter $a_j(\iota)$ at spot $\hat{\lambda}$ in the combined genome of a population in the spirit of the simulated annealing algorithm. The case $\hat{\mu}=1$ corresponds to uniform random change in $\mathcal{A}_j$.

Note that by [19, p. 5: eq. (7')] or [23, eq. (7)] any stochastic matrix has operator norm 1 with respect to the ℓ^1-norm on the underlying vector space. Hence, its spectrum is contained[5] in the closed unit disk in $\mathbb{C}$. Elementary spectral calculus yields that a matrix of the form $(1-\mu)\mathbf{1}+\mu\mathbf{m}_j(\hat{\mu})$, $\mu\in(0,1/2)$, is invertible since its spectrum cannot contain 0.

3.2. Mutation. Let $\hat{\mu},\hat{\mu}'\in(0,1]$ and $\mu,\mu'\in(0,1/2)$. We shall keep $\hat{\mu}'$ and μ' fixed — a discussion of possible annealing schedules for $\hat{\mu}'$ and μ' is left to the reader. The mutation operator $M_{\hat{\mu},\mu}$ is then given by the following procedure:

Definition 3.2.1 (Mutation operator): For $\hat{\lambda}=1\dots L$, execute the next two steps: (STEP 1) Decide probabilistically whether or not to change the letter at spot $\hat{\lambda}$ in the current population. The decision for change is made positively with probability μ for $\hat{\lambda}\in[1,s_1\ell_1]\cup[(s_1+s_2)\ell_1+1,(s_1+s_2)\ell_1+s_3\ell_2]$ in which case we set $\tau=\hat{\mu}$. Otherwise, the decision for change is made positively with constant probability μ', and we set $\tau=\hat{\mu}'$. (STEP 2) If the decision has been made positively in step 1, then the letter at spot $\hat{\lambda}$ is altered in accordance with the transition probabilities for letters set by the spot mutation matrix $\mathbf{m}_j(\tau)$ where $j\in\{1,2\}$ is chosen appropriately. ⌋

Let $M_{\hat{\mu},\mu}$ also denote the fully positive, stochastic matrix associated with multiple-spot mutation that acts on $\mathcal{V}_{\wp}$ and describes transition probabilities for entire populations. It is easy to see that the coefficients of $M_{\hat{\mu},\mu}$ are greater than $K\cdot(\hat{\mu}\mu)^{L_o}$ where $K>0$ is a fixed constant and $L_o=s_1\ell_1+s_3\ell_2$.

Since $\mathcal{V}_{\wp}$ is an L-fold tensor-product of spaces $\mathcal{V}_j^{(1)}$, $j\in\{1,2\}$, the matrix $M_{\hat{\mu},\mu}$ is the L-fold tensor-product of invertible matrices of type $(1-\mu)\mathbf{1}+\mu\mathbf{m}_j(\hat{\mu})$ and is therefore invertible (see, *e.g.*, [24, line (7), Prop. 2.2.2.2 and Sec. 5] for details). Using [24, Lemma 1.4.2.2], we can conclude that the steady state probability distribution w_t of a single step G_t of the scaled genetic algorithm as defined in line (*21*) is uniquely determined. This allows to compute w_t via Cramer's rule as in [24, p. 213: proof of Thm. 3.3.2], and makes it relatively easy to establish strong ergodicity of the the inhomogeneous Markov chain which describes the probabilistic behavior of the scaled genetic algorithm considered in this work.

Definition 3.2.2 (Mutation rate annealing schedule): Let $\varphi>0$ and $t_o\in\mathbb{N}$ be such that $\varphi t_o^{-1/(\kappa_o L_o)}<1/2$, where $\kappa_o\in[1,\infty)$ is set by the user as described below. Now, the annealing schedule for the mutation rate is given by:

$\mu=\mu(t)=\varphi\cdot t^{-1/(\kappa_o L_o)}, \quad t\in\mathbb{N}\cap[t_o,\infty)$.

In addition, let $\hat{\mu}=\hat{\mu}(t)$ be defined by one of the following annealing schedules:

1. *Constant local noise.* Set $\kappa_o=1$. $\hat{\mu}\in(0,1]$ is kept constant.
2. *Decreasing local noise.* Choose $\kappa_o\in(1,\infty)$ and $\hat{\varphi}\in(0,\mu(t_o)^{1-\kappa_o}]$. Define: $\hat{\mu}(t)=\hat{\varphi}\cdot\mu(t)^{\kappa_o-1}\in(0,1]$ ⌋

[5] For a proof, consider stretching eigenvectors with respect to the ℓ^1-norm.

As a consequence of Def. 3.2.2, we obtain by [24, Lemma 1.3.1] that the inhomogeneous Markov chain as defined in line (*21*) which describes the probabilistic behavior of the scaled genetic algorithm considered in this work is *weakly ergodic*. To show *strong ergodicity*, one employs [13, p. 160: Thm. V.4.3] or [23, Thm. 3.3.2]. The prerequisites of these Theorems are verified using the facts that: (1) the matrix-entries of the stochastic matrices $C(\chi_t)$, $M_{\hat{\mu}(t),\mu(t)}$, and S_t representing the genetic operators crossover, mutation and selection have a "nice" functional form[6]; (2) the steady state probability distribution of a single step G_t of the scaled genetic algorithm as defined in line (*21*) can be computed via Cramer's rule as in [24, p. 213: proof of Thm. 3.3.2]; and (3) techniques established in [24, Lemma 3.3.1, p. 213 bottom: proof of Thm. 3.3.2].

Finally, let us discuss the *mutation-flow inequality* associated with the setting described thus far. First, define:

$$\beta = \beta(\hat{\mu}, \mu) = \min\{||P_{\mathcal{U}} M_{\hat{\mu},\mu} p||_1 : p \in \wp \cap \mathcal{U}\} \in (0,1). \tag{10}$$

Then, one obtains the mutation flow inequality with the argument in [24, proof of Prop. 2.2.3, second part] (Recall that $\mathcal{U}$ refers to 'partly uniform' here.):

$$\forall v \in \mathcal{S}_\wp\colon ||(\mathbf{1} - P_{\mathcal{U}}) M_{\hat{\mu},\mu} v||_1 \leq 1 - \beta + \beta ||(\mathbf{1} - P_{\mathcal{U}}) v||_1. \tag{11}$$

The mutation flow inequality is one of two key ingredients to show convergence to partly uniform populations as $\mu \to 0$ in the course of the scaled genetic algorithm.

3.3. Crossover. We consider a similar framework for crossover as in [24]. For crossover rate $\chi \in [0,1]$, the crossover operator is represented by a stochastic matrix $C = C(\chi)$ with entries that are rational functions in χ. The matrix $C(\chi)$ acts on $\mathcal{V}_\wp$ and describes transition probabilities for entire populations. It satisfies:

$$C(0) = \mathbf{1}_{\mathcal{V}(\wp_1)} \otimes C_2^o \otimes \mathbf{1}_{\mathcal{V}(\wp_3)} \otimes C_4^o, \quad \text{and} \quad \forall p \in \wp \cap \mathcal{U}\colon C(\chi) p \in \mathcal{U}. \tag{12}$$

Here, $\mathcal{V}(\wp_1)$ and $\mathcal{V}(\wp_3)$ and the tensor product refer to the decomposition of $\mathcal{V}_\wp$ given in line (*7*). If no further information about the crossover operator is known, then let the annealing schedule for the crossover rate be given by:

$$\chi_t = \phi_c \mu(t)^{\kappa_o(\ell_1+\ell_2)+1}, \qquad \phi_c \in (0, \mu(t_o)^{-\kappa_o(\ell_1+\ell_2)-1}]. \tag{13}$$

Compare the settings in line (*13*) with [24, Thm. 3.3.2, eqs. (41), (45)].

Now suppose that either the crossover operation C_n^{loc} on sub-populations $\wp_n$, $n=1,2,3,4$, is given by regular one-, two-, or multiple-cutpoint crossover, *i.e.*, the creatures are paired sequentially (c_1, c_2), (c_3, c_4), ... (c_{s-1}, c_s), and with probability χ for every pair $(c_{2\sigma-1}, c_{2\sigma})$ letters (genes) are exchanged within the pairs of creatures in accordance with randomly chosen cutpoints; or suppose that C_n^{loc} is given by regular uniform or gene-lottery crossover. See, *e.g.*, [10, pp. 16–17], [27, p. 43] or [24, Sec. 2.4-5]. Suppose that $\chi, \chi' \in (0,1]$ are crossover rates, χ' is kept fixed over the course of the algorithm, and $C(\chi)$ is given by:

$$C(\chi) = C_1^{\mathrm{loc}}(\chi) \otimes C_2^{\mathrm{loc}}(\chi') \otimes C_3^{\mathrm{loc}}(\chi) \otimes C_4^{\mathrm{loc}}(\chi') \tag{14}$$

[6] See the second sentence of Sec. 3.3 for crossover; see the discussion after Def. 3.2.1 in regard to tensor products for mutation, or consult [24, Prop. 2.2.2.1]; and see [24, line (33)] for selection.

in regard to the tensor product decomposition of $\mathcal{V}_\wp$ given in line (*7*). For reason of simplicity, we shall keep $C_1^{\mathrm{loc}} = C_3^{\mathrm{loc}}$. Discussion of mixed cases shall be left to the reader.

Let $m \in [1, \infty)$. We shall set the annealing schedule for the crossover rate as:

$$\chi_t = \phi_c \mu(t)^{1/m}, \qquad \phi_c \in (0, \mu(t_o)^{-1/m}]. \tag{15}$$

3.4. Selection and Convergence. We shall suppose that the (raw) fitness function f is given as in lines (*2*) and (*3*) with $\gamma_{1,2} \in \{\pm 1\}$. The fitness function f shall be power-law scaled, *i.e.*, exponentiated over the course of the algorithm. In fact, we set:

$$f_t(c,p) = (f(c,p))^{g(t)}, \qquad g(t) = B \cdot \log(t - t_o + 2), \tag{16}$$

for $p \in \wp$, $c \in p \cap \mathcal{C}$, $t \in \mathbb{N} \cap [t_o, \infty)$, and fixed $B > 0$. In addition, set $f(c,p) = f_t(c,p) = 0$, if $p \in \wp$ and $c \in \mathcal{C} \setminus p$. In order to define the selection procedure, we set in accordance with the table in Sec. 2.2, and the positions of creatures in the sub-populations p_1, p_2, p_3, p_4 of a population $p \in \wp$ defined there:

$$\begin{array}{llllll} \rho_1 = 1 & \rho_1' = s_1 & \rho_3 = \rho_2' + 1 & \rho_3' = \rho_2' + s_3 & F_{1,t} = F_{3,t} = f_t & \text{scaled} \\ \rho_2 = \rho_1' + 1 & \rho_2' = \rho_1' + s_2 & \rho_4 = \rho_3' + 1 & \rho_4' = s & F_{2,t} = F_{4,t} = f & \text{unscaled} \end{array}$$

Now, suppose that $p = (c_1, c_2, \ldots, c_s)$, $c_\sigma \in \mathcal{C}$, $1 \le \sigma \le s$ is the current population. At time or step $t \in \mathbb{N} \cap [t_o, \infty)$, we attempt to select creatures randomly with probability proportional to their individual fitness-strength $F_{n,t}$, $n = 1, 2, 3, 4$, within the particular sub-population $p_n = (c_\sigma : \sigma = \rho_n \ldots \rho_n')$ of p in which the creature resides. If $c \in \mathcal{C}$, then let let $\#(c, p_n)$ denote the number of copies of c in p_n. Now, we can define the selection procedure:

Definition 3.4.1 (Selection operator S_t): With the current population p as above assemble the new population $q = (d_1, d_2, \ldots, d_s) \in \wp$ with $d_\sigma \in \mathcal{C}$, $1 \le \sigma \le s$, in the following manner: For $n = 1, 2, 3, 4$ do: for $\sigma = \rho_n, \ldots, \rho_n'$ do:
Select creature $d_\sigma \in q$ probabilisticly among the creatures in p_n such that a particular $c \in p_n$ has relative probability for being selected as d_σ given by:

$$\left(\textstyle\sum_{\rho_n \le \sigma' \le \rho_n'} F_{n,t}(c_{\sigma'}, p)\right)^{-1} \cdot \#(c, p_n)\, F_{n,t}(c, p). \qquad \rfloor$$

Note that our definition of selection includes unscaled fitness proportional selection for segments p_2 and p_4 of the population p. This was chosen here for reason of simplicity of presentation. In fact, most standard methods for selection such as tournament selection can be used for segments p_2 and p_4; and this shall not alter the main results of the discussion below.

Let S_t also denote the stochastic matrix associated with scaled proportional fitness selection. S_t acts on $\mathcal{V}_\wp$ and describes transition probabilities for entire populations. It is easy to obtain an explicit formula for the coefficients of S_t as in [24, line (33)] which is needed for the proofs of some of the statements made in this exposition. However, we shall only list below some key properties of S_t which differ from corresponding statements in [24, p. 206]:

$$\forall p \in \wp \cap \mathcal{U}\colon\ S_t p \in \mathcal{U}. \tag{17}$$

$$S_t P_{\mathcal{U}} = P_{\mathcal{U}} S_t P_{\mathcal{U}} \Rightarrow (\mathbf{1} - P_{\mathcal{U}}) S_t = (\mathbf{1} - P_{\mathcal{U}}) S_t (\mathbf{1} - P_{\mathcal{U}}). \tag{18}$$

$$\forall p \in \wp \cap \mathcal{U}\colon\ ||P_{\mathcal{U}} S_t p||_1 \ge (s_1)^{-s_1+1} (s_3)^{-s_3+1} =_{\mathrm{def}} 1 - \theta. \tag{19}$$

$$\forall v{\in}\mathcal{S}_\wp\colon\ ||(\mathbf{1}-P_{\mathcal{U}})S_t v||_1 \le \theta\cdot||(\mathbf{1}-P_{\mathcal{U}})v||_1. \tag{20}$$

The properties established in lines (*11*), (*12*), (*18*) and (*20*) together with an adaptation of the techniques in the proof of [24, Thm. 3.1.1] show the following:
• *The genetic algorithm considered here converges asymptotically to a probability distribution w_∞ over partly uniform populations only.*

Finally, we can consider the inhomogeneous Markov chain that describes the probabilistic behavior of the scaled genetic algorithm considered in this exposition. In fact, a single step at time $t{\in}\mathbb{N}\cap[t_o,\infty)$ is described by the following stochastic matrix (t_o is the intial value for t):

$$G_t = S_t\cdot C(\chi_t)^{1-k}\cdot M_{\hat{\mu}(t),\mu(t)}\cdot C(\chi_t)^k,\ \text{where } k{=}0,\ \text{or } k{=}1. \tag{21}$$

Note that we do NOT suppose that crossover $C(\chi_t)$ and mutation $M_{\hat{\mu}(t),\mu(t)}$ commute. Let $w_t{=}G_t w_t{\in}\mathcal{S}_\wp$ denote the uniquely determined steady-state distribution of an individual step G_t of the scaled genetic algorithm (*cf.*, the discussion following Def. 3.2.1). Let $H_t{=}\prod_{\tau=t}^{t_o} G_\tau$. We have already established that $(H_t)_{t\in\mathbb{N}\cap[t_o,\infty)}$ is strongly ergodic in the discussion following Def. 3.2.2. Hence, for every $v_o{\in}\mathcal{S}_\wp$: $\lim_{t\to\infty} H_t v_o = \lim_{t\to\infty} w_t =_{\text{def}} w_\infty$.

We have outlined in the discussion after line (*20*) how to obtain that w_∞ is positive only over partly uniform populations. What remains to show is that:
• *w_∞ is positive only over such populations that contain elements from the 'top equivalence classes' $\mathcal{C}_1^{\max}\subset\mathcal{C}_1$ and $\mathcal{C}_2^{\max}\subset\mathcal{C}_2$ as defined in footnote 4.*
In order to achieve this, one establishes a '*steady-state flow inequality*' for the w_t considered here similar to [24, lines (45–7)]. This yields the following conditions:

$$\text{General type crossover:}\quad \kappa_o(\ell_1+\ell_2) < \min(s_1,s_3) \tag{22}$$

$$\text{General type crossover:}\quad \kappa_o(\ell_1+\ell_2) < \kappa_o L_o B\log(\rho_2(f))+1 \tag{23}$$

$$\text{Regular crossover:}\quad 2m\kappa_o(\ell_1+\ell_2) < \min(s_1,s_3) \tag{24}$$

$$\text{Regular crossover:}\quad \kappa_o(\ell_1+\ell_2) < \kappa_o L_o B\log(\rho_2(f))+1/m \tag{25}$$

$$\text{Gene-lottery crossover:}\quad m\kappa_o(\ell_1+\ell_2) < \min(s_1,s_3) \tag{26}$$

$$\text{Gene-lottery crossover:}\quad \kappa_o(\ell_1+\ell_2) < \kappa_o L_o B\log(\rho_2(f))+1/m \tag{27}$$

Here, $\rho_2(f)$ is a constant measuring the strength of 'second-to-best elements' in populations containing elements of $\mathcal{C}_1^{\max}$ and $\mathcal{C}_2^{\max}$. In fact, $\rho_2(f)$ is given by:

$$T=\{(p,\hat{c},c): p{\in}\wp,\ p_{2j-1}{\cap}\mathcal{C}_j^{\max}{\neq}\emptyset,\ p_{2j-1}{\setminus}\mathcal{C}_j^{\max}{\neq}\emptyset\ \text{ for both } j{=}1,2,\text{ and } \hat{c}{\in}p_{2j-1}{\cap}\mathcal{C}_j^{\max},\ c{\in}p_{2j-1}{\setminus}\mathcal{C}_j^{\max}\ \text{ for one } j{=}1,2\}$$

$$\rho_2(f)=\min\{f(\hat{c},p)/f(c,p):(p,\hat{c},c){\in}T\}\quad(\text{assume } T\neq\emptyset). \tag{28}$$

$\rho_2(f)$ is easy to determine, if one employs rank ($=f$) for the fitness selection mechanism based upon an originally-given raw fitness function f_r.

Note that lines (*24*) and (*26*) show (with mathematical theory and not experimentally) the remarkable effect that with increasing population size, one is allowed to use a more relaxed cooling schedule for crossover. Thus, for larger population size, the part of the algorithm-design, *i.e.*, definition of creatures (data-structures), which is exploited by crossover plays a more important role.

Conclusion

We have obtained a new, all-purpose coevolutionary genetic algorithm suitable for optimization in a multi-species setting for which a *global optimization theorem* holds. The proposed algorithm is very much in the spirit of the simulated annealing algorithm. It is realistic in that the population-size and consequently evaluation-time for the fitness function stay relatively small. Explicite annealing schedules for crossover/mutation and exponentiation schedules for the fitness-function scaling are given such that the proposed algorithm, in fact, can be readily implemented. The selection operator which uses fitness-proportional selection applies the scaled fitness function only on part of the population in order to keep the complementary part as a "noisy", non-converging field of "test-instances". Thus, the proposed algorithm improves and generalizes a previously published approach for a coevolutionary genetic algorithm in a multi-species setting by removing some technical conditions on the latter and introducing new features such as maintaining a noisy field of test-instances without loosing convergence to global optima. Such convergence is here understood as convergence toward populations contain only elements of canonical equivalence classes of "superior" creatures ($\Leftrightarrow$ global optima) formed in relation to the fitness function. One important aspect of future research on this algorithm should be the incorporation of various enhancement procedures for maintenance of "balanced and rich" sets of test-instances within the population that have been proposed in the literature.

References

1. A. Bucci, J.B. Pollack: A Mathematical Framework for the Study of Coevolution. *Proc. FOGA 7*, K. De Jong, *et al.* (eds.), Morgan Kaufmann, San Francisco, CA, USA (2003)
2. A. Bucci, J.B. Pollack: Focusing versus Intransitivity: Geometrical Aspects of Coevolution. In: [3], pp. 250–261
3. E. Cantu-Paz *et al.* (eds.): *Proc. GECCO 2003*, LNCS 2723, Springer Verlag, Berlin, Germany (2003)
4. T.E. Davis, J.C. Principe: A Markov Chain Framework for the Simple Genetic Algorithm. *Evolut. Comput.* **1** (1993), pp. 269–288
5. E.D. De Jong: Representation Development from Pareto-Coevolution. In: [3], pp. 262–273
6. E.D. De Jong, J.B. Pollack: Learning the Ideal Evaluation Function. In: [3], pp. 277–288
7. K. De Jong: Lecture on Coevolution. In: Seminar "*Theory of Evolutionary Computation*", H.-G. Beyer *et al.* (chairs), Max Planck Inst. for Comput. Sci. Conf. Cntr., Schloß Dagstuhl, Saarland, Germany (2002)
8. S.G. Ficci, J.B. Pollack: Effects of Finite Populations on Evolutionary Stable Strategies. *Proc. GECCO 2000*, D. Whitley *et al.* (eds.), Morgan Kaufmann, San Francisco, CA, USA (2000), pp. 927–934
9. S.G. Ficici, J.B. Pollack: A Game-Theoretic Memory Mechanism for Coevolution. In: [3], pp. 286–297
10. D.E. Goldberg: *Genetic Algorithms, in Search, Optimization & Machine Learning.* Addison-Wesley Publishers, Boston, MA, USA (1989)

11. J.H. Holland: *Adaptation in Natural and Artificial Systems.* University of Michigan Press, Ann Arbor, MI, USA(1975), MIT Press, Cambridge, MA, USA (1992)
12. J. Horn, J. Cattron: The Paradox of the Plankton: Oscillations and Chaos in Multispecies Evolution. In: [3], pp. 298–309
13. D.L. Isaacson, R.W. Madsen: *Markov Chains: Theory and Applications.* Prentice-Hall Publishers, Upper Saddle River, NJ, USA (1961)
14. T. Jansen, R.P. Wiegand: Exploring the Explorative Advantage of the Cooperative Coevolutionary $(1 + 1)$ EA. In: [3], pp. 310–321
15. J.R. Koza, M.A. Keane, M.J. Streeter: *Genetic Programming IV: Routine Human-Competitive Machine Intelligence.* Kluwer Academic Publishers, Dordrecht, The Netherlands (2003)
16. D.S. Linden: Antenna Design Using Genetic Algorithms. *Proc. GECCO 2002*, W.B. Langdon *et al.* (eds.), Morgan Kaufmann Publishers, San Francisco, CA, USA (2002), pp. 1133–1140
17. A. Márkus, G. Renner, J. Vanza. Spline Interpolation with Genetic Algorithms. *Proc. Int. Conf. Shape Modeling*, Aizu Univ. (1997), T.L. Kunii *et al.* (chairs), IEEE Computer Soc. Press, Los Alamitos, CA, USA (1997), pp. 47–54
18. G. Rudolph: Convergence Analysis of Canonical Genetic Algorithms. *IEEE Trans. Neural Networks* **5** (1994), pp. 96–101
19. H.H. Schaefer. *Banach Lattices and Positive Operators.* Springer Verlag, Berlin, Germany (1974)
20. L.M. Schmitt: Theory of Genetic Algorithms. *Theoret. Comput. Sci.* **259** (2001), pp. 1–61
21. L.M. Schmitt: Optimization with Genetic Algorithms in Multi-Species Environments. *Proc. ICCIMA 2003*, Xidian Univ., L. Jiao, *et al.* (eds.), IEEE Computer Soc. Press, Los Alamitos, CA, USA, pp. 194–199
22. L.M. Schmitt: Theory of Coevolutionary Genetic Algorithms. *Proc. ISPA 2003*, Aizu Univ., M. Guo, L.T. Yang (eds.), LNCS 2745, Springer Verlag, Berlin, Germany (2003), pp. 285–293
23. L.M. Schmitt: Asymptotic Convergence of Scaled Genetic Algorithms to Global Optima —A gentle introduction to the theory—. In: *Frontiers of Evolutionary Computation*, A. Menon (ed.), Genetic Alg. and Evol. Comput. Ser. **11**, Kluwer Publishers, Dordrecht, The Netherlands (2004), pp. 157–192
24. L.M. Schmitt: Theory of Genetic Algorithms II — Models for genetic operators over the string-tensor representation of populations and convergence to global optima for arbitrary fitness function under scaling. *Theoret. Comput. Sci.* **310** (2004), pp. 181–231
25. L.M. Schmitt, T. Kondoh: Optimization of Mass Distribution in Articulated Figures with Genetic Algorithms. *Proc. IASTED Int. Conf. ASM 2000*, Banff, Alberta, Canada, M.H. Hamza (ed.), IASTED-ACTA Press, Anaheim-Calgary-Zürich (2000), pp. 191–197
26. S. Tong, D.J. Powell: Genetic Algorithms: A Fundamental Component of an Optimization Toolkit for Improved Engineering Designs. In: [3], pp. 2346–2359
27. M.D. Vose: *The Simple Genetic Algorithm: Foundations and Theory.* MIT Press, Cambridge, MA, USA (1999)
28. R.A. Watson, J.B. Pollack: Coevolutionary Dynamics in a Minimal Substrate. *Proc. GECCO 2001*, L. Spector *et al.* (eds.), Morgan Kaufmann Publishers, San Francisco, CA, USA (2001), pp. 702–709
29. J.M. Zurada: *Introduction to Artificial Neural Systems.* West Publ. Co., St. Paul, MN, USA (1992)