Int. J. Appl. Math. Comput. Sci., 2012, Vol. 22, No. 4, 971–984
DOI: 10.2478/v10006-012-0072-z

THE ISLAND MODEL AS A MARKOV DYNAMIC SYSTEM

ROBERT SCHAEFER [*], ALEKSANDER BYRSKI [*], MACIEJ SMOŁKA [**]

[*] Department of Computer Science
AGH University of Science and Technology, Al. Mickiewicza 30, 30-059 Kraków, Poland
e-mail: {schaefer,olekb}@agh.edu.pl

[**] Institute of Computer Science
Jagiellonian University, ul. Łojasiewicza 6, 30-348 Kraków, Poland
e-mail: smolka@ii.uj.edu.pl

Parallel multi-deme genetic algorithms are especially advantageous because they allow reducing the time of computations and can perform a much broader search than single-population ones. However, their formal analysis does not seem to have been studied exhaustively enough. In this paper we propose a mathematical framework describing a wide class of island-like strategies as a stationary Markov chain. Our approach uses extensively the modeling principles introduced by Vose, Rudolph and their collaborators. An original and crucial feature of the framework we propose is the mechanism of inter-deme agent operation synchronization. It is important from both a practical and a theoretical point of view. We show that under a mild assumption the resulting Markov chain is ergodic and the sequence of the related sampling measures converges to some invariant measure. The asymptotic guarantee of success is also obtained as a simple issue of ergodicity. Moreover, if the cardinality of each island population grows to infinity, then the sequence of the limit invariant measures contains a weakly convergent subsequence. The formal description of the island model obtained for the case of solving a single-objective problem can also be extended to the multi-objective case.

Keywords: genetic algorithms, asymptotic analysis, global optimization, parallel evolutionary algorithms, Markov chain modeling.

1. Introduction

In the introductory part we present an extensive motivation, state of the art and preliminaries necessary to sketch out the contents and results of this contribution.

1.1. Motivation. Evolutionary algorithms turned out to be a universal optimization technique which had already been successful in many practical problems (see, e.g., Bäck *et al*., 2000; Brabazon and O'Neill, 2006; Mesghouni *et al*., 2004; Kowalczuk and Białaszewski, 2006) that must be approached with the use of heuristic algorithms. However, when dealing with heuristic algorithms, an effort should be made to prove that they are really able to find or even get close to the desired set of solutions. Such a conclusion may be drawn on the basis of qualitative analysis of certain features of algorithms, such as, e.g., a guarantee of success (see, e.g., Rinnoy Kan and Timmer, 1987), the type and speed of convergence.

Many results were obtained for Monte Carlo type strategies in which the sampling measure is not modified during computation (see, e.g., Wood and Zabinsky, 2002). Adaptation of probability measures of heuristic algorithms (being an effect of a "learning" process conducted by the algorithm) makes mathematical modelling more difficult.

A significant number of mathematical models of evolutionary algorithms (based on the analysis of Markov chains) which have already been proposed successfully to model single-population algorithms, e.g., the formal model presented by Vose (1998) and Rudolph (1997) constituted a basis for further analysis of stochastic features of fundamental genetic mechanisms. In particular, they proved the asymptotic guarantee of success (Horst and Pardalos, 1995; Rinnoy Kan and Timmer, 1987) of their behavior, which formally confirmed their application in global optimization.

Further examples of research on the modeling of Genetic Algorithms (GAs) proposed by other researchers provide a deeper insight into the long term, steady

state behavior of large population GAs (e.g., Davis and Principe, 1991; Suzuki, 1993; Rudolph, 1994) or modeling specific features of GAs such as selection, genetic drift, niching, etc. (e.g., Goldberg and Segrest, 1987; Mahfoud, 1991; Horn, 1993).

Many authors use a Markov-based model of GAs for evaluating the convergence rate and computational complexity (see, e.g., Rudolph, 1997; Mühlenbein, 1992). Liekens (2005) successfully extended the work of Vose to study long term evolutionary behavior of small populations of haploid and diploid individuals in dynamic fitness environments. Schmitt (2001) provides a complete study of various aspects of genetic algorithms, with the stress on asymptotic features such as convergence, additionally exploring the ergodicity of their Markovian model.

Parallel versions of genetic algorithms (e.g., the island model) proved to be important, advantageous models of computations, because they follow the idea of allopatric speciation (Li and Yang, 2008; Skolicki and de Jong, 2004; Alba and Tomassini, 2002; Paredis, 1998; Potter and De Jong, 2000), which helps to increase the population diversity and allows achieving a reasonable balance between exploration and exploitation by means of population decomposition and migration (Back *et al.*, 1997; Cantú, 1995; 2000). Another advantage is that they create technical possibilities of decreasing the computation time, thus allowing implementation of these systems in distributed environments according to, e.g., the master-slave model (see, e.g., Cantú-Paz, 1995; Whitley *et al*., 1997).

The modeling of various population and multi-population algorithms has been carried out for many years. A construction of such models was conducted in the cases of simulation of biological populations (finite and infinite cases) as early as in the late 1970s (Nagylaki, 1979). There have also been more recent works that apply Markov chains to model population dynamics for ecological simulations (for a broader reference, see, e.g., the work of Buckley *et al.* (2010).

A complete model for examining stochastic dynamics of Island Models (IMs) for finite populations has not been proposed yet. One of the possible reasons is the complexity of this task (Skolicki, 2007). On the other hand, there have been several approaches to model parallel versions of evolutionary algorithms, taking into account different features of these models, e.g., takeover time, convergence, computation speed (cf. Lässig and Sudholt, 2010; Rudolph, 2006; Whitley *et al*., 1997; Whitley, 1992; Cantú-Paz, 2000). These approaches usually focus on particular cases of algorithms or problems (e.g., the $(1 + 1)$ evolutionary strategy or linearly separable problems). The stochastic model of the multi-deme strategy HGS solving a single problem by using various encodings and various accuracies was introduced by Schaefer *et al*. (2012).

Designing appropriate models allowing analyses of complex computing techniques, such as parallel evolutionary algorithms or other metaheuristics, will surely make it possible to take advantage of the rigorous analytical tools in more complex applications, such as, e.g., described by Kołodziej and Xhafa (2011) or Terzo *et al.* (2011).

1.2. Formal analysis of island models: State of the art. The popularity of biologically inspired algorithms has generated the need for an enhancement of single-population approaches, including separation of individuals, which results in a qualitative change in the behavior of evolutionary algorithms (Cantú-Paz, 1995; Skolicki, 2007; Rudolph, 2006). It is usually carried out in two ways:

- grouping individuals into subpopulations (coarse-grained models, island models, hierarchical genetic strategy, etc.) (Gordon *et al*., 1992; Schaefer and Telega, 2007),
- locating individuals on a grid and restricting their interaction to some neighborhood (fine-grained models, cellular models) (Manderick and Spiessens, 1989; Mühlenbein, 1989; Tomassini, 2005).

A further taxonomy may use the following features:

- *encoding*: in order to easily perform operations such as migration, islands should use the same encoding scheme;
- *genetic engine*: the system may be hetero- or homogeneous, therefore each island may run a different evolutionary algorithm which is easily characterized by setting the selection scheme, crossover and mutation operators;
- *constant or variable number of individuals*: all islands may be of the same or different cardinality; the cardinality of the population on an individual island may be constant or vary during evolution;
- *migration topology*: islands may be connected by a full graph, a ring, a torus, etc.;
- *strategies for emigration, migration and immigration*: way of selecting migrants (Rudolph, 1994), choosing a target island and incorporating migrants into the target population.

In this subsection, a short survey of the models constructed for such evolutionary algorithms is presented.

Whitley (1992) uses a special case of the model introduced by Vose and Liepins (1991) to propose the

way for computing the number of certain schemata in a single population, as well as in the island model (with ring topology) for optimization of deceptive problem functions. Whitley *et al*. (1997) report the island model applied to optimization of linearly separable problems to be more effective in finding extrema than a single-population genetic algorithm.

Cantú-Paz (2000) presents a model considering the so-called "gambler's ruin problem" (Harik *et al*., 1999), which is used to compute the number of correct building blocks (representing desired solutions) present in the population. The author assumes that the islands first operate autonomously (without exchanging individuals). After converging to a single building block, migration is undertaken. The author gives a formula to compute the probability of converging to the correct building block of a certain number of demes, as well as the distribution of the number of demes that converge correctly after a certain number of epochs. His discussion is supported by a mathematical analysis based mainly on using the Bernoulli binomial probability distribution, and a number of experiments carried out in the case of different migration rates and topologies. His observations have proved that parallel genetic algorithms reach solutions of the same quality as single-deme genetic algorithms, though much faster. This makes them an interesting alternative and paves the way for taking advantage of various possibilities of supporting such computations, using various flavors of distributed computing.

The studies by Droste and co-workers (Droste *et al.*, 1998a; 1998b) found continuation, e.g., in the work of Lässig and Sudholt (2010), who extend the case considered by Droste to a parallel evolutionary algorithm, and point out that such a decomposition of population leads to a significant speedup of computation without increasing the total number of evaluations.

Rudolph (2006) provides a stochastic analysis leading to the evaluation of upper bounds of takeover time in the parallel evolutionary algorithm with migration among the demes connected with different topologies.

Whitley *et al.* (1997) analyze experimentally the behavior of parallel genetic algorithms constructed according to Vose's model for linearly separable functions; however, they do not provide the reader with any analysis of the model—just state that they extend the heuristic function and introduce migration inside a ring topology with a specific migration policy. Their results are discouraging for parallel evolutionary algorithms. Fortunately, the application of the system is very specific and of course these results should not be generalized to all problems (according to the so-called "no free lunch theorem" (Wolpert and Macready, 1997)).

Byrski and Schaefer (2009) as well as Schaefer *et al*. (2009) try to formalize a certain type of systems from the class of memetic computing: evolutionary multi-agent systems. The description of the space of system states and Markovian transition functions are given. The proposed framework may become useful for analysis of a broader class of parallel population-based algorithms.

1.3. Outline of the proposed model. Let us assume that the domain of the problem to be solved is represented by a finite genetic universum U, $\#U = r < +\infty$. Assuming an arbitrary linear order in U, we may use it for indexing vectors $x \in \mathbb{R}^r$ so that $x = (x_\xi)_{\xi \in U}$.

The problem is also characterized by the fitness function, $f : U \to [0, \Delta]$, $\Delta < +\infty$, which may sometimes be identified with the vector of its values $f = (f_\xi)_{\xi \in U}$.

We assume that this strategy leads to solving the global optimization problem that may be formulated as finding $\arg\max_{\xi \in U}\{f_\xi\}$. In other words, all $\hat{\xi}$ such that $f_{\hat{\xi}} \geq f_\xi, \forall \xi \in U$ should be found. The problem itself does not affect in any way our formalism, therefore our deliberations are not problem-dependent as in some other works (e.g., Whitley *et al*., 1997).

Following the taxonomy presented in Section 1.2, we would like to state that our multi-deme genetic strategy covers the following cases:

- coarse-grained, island-like models of evolution, population decomposed into a number of genetic islands; the number of individuals residing on each island may vary among them (see Section 3, Eqn. (8)),
- finite encoding, the same genetic universum U is used on all islands (see Section 2.1);
- at the beginning of the system's work, individuals of each population are randomly created using a predefined, particular probability distribution that may vary among the islands (see Section 6.1);
- two succession schemes for obtaining the next generation on each island are considered (see Section 2.2);
  - SSS: the standard one, in which an intermediate population (mating pool) is constructed,
  - VSS: the one used in the model of Vose (1998), in which the subsequent generation is constructed one-by-one through picking individuals from the previous generation and after mutating and crossing over adding the offspring to the subsequent generation;
- possible different mixing operations (mutation, crossover) used on different islands (see Section 3);
- uniform selection scheme on all islands (see Section 3);

- stochastic/deterministic emigration policies; migrants may be chosen using a deterministic rule, or sampled using some probability distribution (see Sections 4.1, 4.2),
- stochastic/deterministic migration topologies, an actual migration target may be chosen randomly or deterministically (see Sections 4.1, 4.2);
- common selection of target and migrant populations used as the immigration policy (see Section 4.3);
- agent-based synchronization mechanism (following the method presented by Byrski and Schaefer (2009) as well as Schaefer *et al*. (2009)), necessary to construct one Markov chain for the whole system (see Section 5).

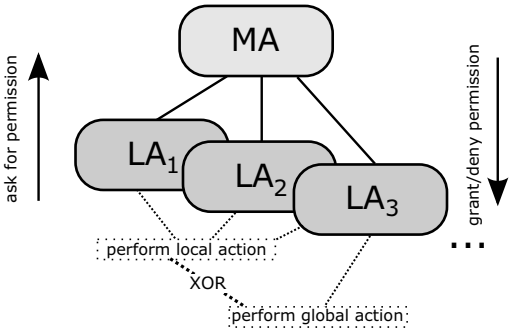

Fig. 1. Agent-based synchronization mechanism.

The island-based model is a concurrent processing model with local memory (not all of its elements may access and modify its memory contained on the islands; however, an exchange of information exists between the islands: individuals migration). A precise model for such a system does not exist.

In order to construct such a model, one of the following, well-known approaches can be followed: trace theory (Diekert and Rozenberg, 1995), process algebra (Hennessy, 1988), Petri nets (Peterson, 1981), message passing Pi-calculus (Milner, 1990), or message passing actor model (Hewitt *et al*., 1973). The last of the above-mentioned models (message passing actor model) seems best suited for the problem of synchronisation in the parallel island model, as it is very close to the practical implementation of such a system and may be easily enhanced by introducing the notion of an agent (that will perform additional actions, instead of synchronisation, such as load balancing (Grochowski *et al*., 2004)).

Such a definition of concurrency becomes an inevitable step which has to precede the construction of the stochastic Markovian model. Subsequent synchronization points must be identified, so that they can be mapped into appropriate transformations between Markov chain steps. A more relaxed synchronization scheme is possible to be constructed (based on, e.g., message queues), though it would be very hard to model it as a single Markov chain.

Therefore, an evolution process performed on the islands is synchronized by the predefined agents (local: LA_i and master: MA) (see Fig. 1) that communicate among themselves to arrange proper scheduling for making changes to the system state. The changes of the system state are divided into two types: local (processing the population) and global (performing migration). The master agent makes a decision on which type of state changes should be performed, with a certain probability at the current moment (population processing or migration), and orders the local agents to behave accordingly. The local actions are performed in parallel on each island. Every local action changes only the state of the island where it is performed. The global action depends on and can change the state of at least two islands.

The details of this synchronization mechanism together with pseudo-codes of both types of agents will be given later (in Section 5). This description has to be preceded by a precise, formal definition of all stochastic sampling and decision operations, which will be performed in Sections 2–4.

In Section 6, the Markov probability transition function for the whole strategy is constructed. Later we show that under a mild assumption the Markov chain is ergodic and the sequence of sampling measures converges (see Theorem 1, Corollary 1), which in particular leads to the asymptotic guarantee of success. Moreover, if the cardinality of each island population grows to infinity, then the sequence of the limit invariant measures contains a weakly convergent subsequence (see Theorem 2).

2. Sampling measures for a single population

Let us introduce some notions and constructions treating a single population GA, necessary for later modeling of the parallel strategy.

2.1. Preliminaries. The finite population being a multiset of elements from the genetic universum U may be denoted as a tuple $P = (U, \eta_P)$, where the function $\eta_P : U \to \mathbb{N} \cup \{0\}$ returns the number of clones of each genotype. Moreover, the cardinality of such a population may be computed as $\mu = \sum_{\xi \in U} \eta_P(\xi)$ (see, e.g., Aparicio *et al*., 1999; Schaefer and Telega, 2007).

Let us introduce "Vose's simplex":

$$\mathbb{R}^r \supset \Lambda^r = \{x \in \mathbb{R}^r : 0 \le x_\xi \le 1, \quad \forall\, \xi \in U, \ \sum_{\xi \in U} x_\xi = 1\}, \quad (1)$$

associated with the finite genetic universum U which contains all frequency vectors of populations of individuals from U. The frequency vector $x^P \in \Lambda^r$ of the population P has entries $x^P_\xi = (1/\mu)\ \eta_P(\xi),\ \xi \in U$. Of course, if $\mu < +\infty$, then x^P may belong to a finite subset X_μ ($\#X_\mu = n < +\infty$) of the simplex Λ^r only. In other words, each element of X_μ corresponds to exactly one population of cardinality $\mu < +\infty$ composed of clones of genes from U.

Note that Λ^r can also be identified with the space of all probabilistic measures $\mathcal{M}(U)$ on the set of genotypes (see, e.g., Schmitt, 2001; Schaefer and Telega, 2007). In the sequel, $\mathcal{M}(A)$ denotes the space of probabilistic measures defined on a Σ-algebra over the set A. To avoid confusion, we formally introduce a one-to-one mapping:

$$\Theta : \Lambda^r \ni x \to \Theta(x) \in \mathcal{M}(U) \tag{2}$$

that allows us to identify a frequency population vector $x \in \Lambda^r$ with the related probabilistic measure $\Theta(x) \in \mathcal{M}(U)$.

We will intensively use the operators that map Λ^r into Λ^r (introduced by Vose and Nix), which allows us to describe comprehensively the dynamics of finite and infinite population genetic algorithms with finite genetic universa (see Nix and Vose, 1992; Vose, 1998; Schmitt, 2001). We will consider the *selection operator* $F : \Lambda^r \to \Lambda^r$ that represents the stochastic effect of selection and the *mixing operator* $M : \Lambda^r \to \Lambda^r$ associated with the genetic operations, namely, the crossover coupled with the mutation.

The ξ-th coordinate of the selection operator returns the probability of selecting an individual with the genotype $\xi \in U$ from the population represented by the vector $x \in \Lambda^r$. Formally, we denote this probability by $\Theta_\xi(F(x))$. Similarly, $\Theta_\xi(M(x))$ is the probability of obtaining an individual with the genotype $\xi \in U$ by mixing from the population represented by the vector $x \in \Lambda^r$.

In the case of a genetic universum composed of binary strings (e.g., the SGA case, $U = \Omega$ (see Vose, 1998)) the selection operators imposed by proportional, tournament and ranking selection schemes are described by Vose (1998, Section 4.2), along with mixing consist in binary crossover and positional, bitwise mutation (Vose, 1998, Sections 4.3–4.5).

2.2. Succession schemes.

Observation 1. Given the probability distribution $\rho \in \mathcal{M}(U)$ such that ρ_ξ is the probability of obtaining an individual with a genotype $\xi \in U$ in the next epoch, the probability of obtaining the next population of cardinality $\mu < +\infty$, represented by a vector $y \in X_\mu$, is given by the polynomial probability distribution:

$$\Pr^\mu_\rho(y) = \frac{\mu!}{\prod_{\xi \in U} (\mu \cdot y_\xi)!} \prod_{\xi \in U} (\rho_\xi)^{\mu \cdot y_\xi}. \tag{3}$$

The observation immediately stems from the polynomial distribution features (see, e.g., Billingsley, 1995). The evaluation of a population sampling probability by using the polynomial distribution was introduced by Nix and Vose (1992).

We will apply two succession schemes (the schemes of obtaining the next epoch population from the current one). The first one, called the *Standard Succession Scheme* (SSS), consists in selecting the intermediate population of parental individuals of the same cardinality μ by independent μ-fold sampling according to the probability distribution $\Theta(F(x))$. The probability of obtaining the intermediate population might be computed using the formula (3) by setting $\rho = \Theta(F(x))$, where $x \in X_\mu$ stands for the current population vector. The next-epoch population is then derived from the intermediate one by means of admissible genetic operations (mixing is a composition of these operations). The probability of the next-epoch population may be evaluated again using (3) by setting $\rho = \Theta(M(z))$, where $z \in X_\mu$ denotes now the intermediate population vector. Summing up, according to the Bayes rule (see, e.g., Billingsley, 1995) the probability of the next-epoch population obtained by the standard succession rule is described by the following function:

$$\tau_{St} : X_\mu \times X_\mu \ni (x, y) \to \tau_{St}(x, y) = \sum_{z \in X_\mu} \Pr^\mu_{\Theta(F(x))}(z) \cdot \Gamma_{zy}, \tag{4}$$

where the $n \times n$ matrix Γ matrix is given by the formula

$$\Gamma_{xy} = \Pr^\mu_{\Theta(M(x))}(y), \quad x, y \in X_\mu. \tag{5}$$

The second scheme uses the composition $G = M \circ F : \Lambda^r \to \Lambda^r$ called the *the heuristic operator* (or simply *heuristic*). It was introduced by Vose (1998) for the simple genetic algorithm. The probability of the next epoch population is given now by the following function:

$$\tau_G : X_\mu \times X_\mu \ni (x, y) \to \tau_G(x, y) = \Pr^\mu_{\Theta(G(x))}(y). \tag{6}$$

We will call it the *Vose Succession Scheme* (VSS). The VSS might be implemented by μ-fold execution of the following three steps (Vose, 1998, Chapter 5):

1. select two parental individuals from the current population,
2. mutate each of them,

3. cross the mutated parents and add one randomly selected child to the next epoch population.

Because X_μ is finite ($\#X_\mu = n < +\infty$), the transition probability functions (4), (6) can be represented as the transition matrix Q so that for $x, y \in X_\mu$ we have

$$Q_{xy} = \begin{cases} \tau_{St}(x,y) & \text{if SSS is applied,} \\ \tau_G(x,y) & \text{if VSS is used.} \end{cases} \tag{7}$$

The exact meaning of the matrix Q will depend on the context in which the succession scheme is determined.

3. Formal description of sampling measures for the island model

We assume that there are $s \in \mathbb{N}$ locations called "islands" containing populations $P^1, \ldots, P^s$ being the multisets

$$P^i = (U, \eta_{P^i}), \eta_{P^i} : U \to \mathbb{N} \cup \{0\}$$

composed of clones of genotypes from a single finite genetic universum $U, \#U = r < +\infty$. The populations are finite: $\#P^i = \mu_i < \infty$ for $i = 1, \ldots, s$. Each population P^i is associated with its frequency vector $x^{P^i} \in X_{\mu_i} \subset \Lambda^r, \#X_{\mu_i} = n_i < +\infty, i = 1, \ldots, s$. In order to simplify expressions in the sequel, we will use the notation $x^{P^i} = x^i$ with $i = 1, \ldots, s$.

The state of the island system will then be an element of the Cartesian product

$$X \stackrel{\text{df}}{=} X_{\mu_1} \times, \ldots, \times X_{\mu_s} \subset (\Lambda^r)^s, \tag{8}$$

where X_{μ_i} is the space of states for the i-th island. Instances $x = (x^1, \ldots, x^s) \in X$ will be called *states of the island model*, whereas their coordinates $x^i, i = 1, \ldots, s$ will be called local state instances or the *island states*.

If a particular i-th island evolves independently (it does not run the immigration policy), then each step of its evolution is governed by the selection operator $F : \Lambda^r \to \Lambda^r$, which is common to all of the islands and the specific mixing operator $M^i : \Lambda^r \to \Lambda^r$. The selection operator parameterized by the fitness function represents the influence of the global optimization problem to be collectively solved by all of the islands. The mixing operator expresses the random effect of applying genetic operations (e.g., mutations, crossovers, inversions) to the genotypes of the i-th island. Different kinds of operations should be established for a particular island for the whole computation process, but they may vary among islands, so in general M^i might differ from M^j for $i \neq j$. Furthermore, the composition $G^i = M^i \circ F : \Lambda^r \to \Lambda^r$ constitutes the "heuristic" of the i-th island.

We also introduce $\mu_i \times \mu_i$ matrices Q^i, Γ^i defined by means of formulas analogous to (7), (5) with M, G, X_μ replaced by M^i, G^i, X_{μ_i}, namely, for $x^i, y^i \in X_{\mu_i}, i = 1, \ldots, s$ we have

$$\Gamma^i_{x^i\, y^i} = \Pr^{\mu_i}_{\Theta(M^i(x^i))}(y^i),$$

$$Q^i_{x^i\, y^i} = \begin{cases} \sum_{z^i \in X_{\mu_i}} \Pr^{\mu_i}_{\Theta(F(x^i))}(z^i) \cdot \Gamma^i_{z^i\, y^i} & \text{for SSS,} \\ \Pr^{\mu_i}_{\Theta(G^i(x^i))}(y^i) & \text{for VSS.} \end{cases} \tag{9}$$

4. Migration with common selection

Next, for each island let us define a sub-population called the "migrant",

$$E^j = (U, \eta_{E^j}), \eta_{E^j} : U \to \mathbb{N} \cup \{0\},$$

composed of individuals selected from the j-th island population P^j. We assume that all migrants are of the same finite size m, so that

$$\#E^j = \sum_{\xi \in U} \eta_{E^j}(\xi) = m$$

for all $j = 1, \ldots, s$. The most common situation is when $m < \mu_i$ for $i = 1, \ldots, s$. We will denote by $e^j \in X_m \subset \Lambda^r$ migrants' frequency vectors and by $n_m = \#X_m < +\infty$ the cardinality of their family.

Islands are interconnected with a set of directed paths $Path \subset \{1, \ldots, s\}^2$ along which migrants may pass. Moreover, we denote by

$$I_i = \{j \in \{1, \ldots, s\}; (j, i) \in Path\}$$

the set of numbers of islands from which migrants may come to the i-th island. Moreover, we denote by $K_i = \#I_i$ the cardinality of these sets for all $1, \ldots, s$.

Creating the migrant E^j consists in cloning selected individuals from P^j (deterministically or randomly), so the source population is not changed after this operation.

4.1. Emigration policies. Let us now assume that the migrant E^i with the frequency vector $e^i \in X_m \subset \Lambda^r$ is selected deterministically by means of the following function:

$$\Phi^i : X_{\mu_i} \ni x^i \mapsto e^i \in X_m, \tag{10}$$

which defines a specific emigration policy for each island. A typical example of such a selector might take the values

$$\Phi^i_\xi(x^i) = \begin{cases} 1 & \text{if } \xi = \min\{j \in U; \\ & \quad f_j = \max_{k \in U}\{f_k \lceil x^i_k \rceil\}\}, \\ 0 & \text{otherwise,} \end{cases} \tag{11}$$

where $\xi \in U$, $x^i \in \Lambda^r$, the ceiling function $\lceil x^i_k \rceil$ returns 1 when $x^i_k \neq 0$ and 0 otherwise, and $f_j = f(j)$ is the fitness of $j \in U$. The above formula expresses the case in which

the migrant is composed of clones of one of the best fitted individuals in the population represented by $x \in \Lambda^r$.

Let us assume in turn that migrants are chosen randomly, according to probability distributions determined by operators $D^i : \Lambda^r \to \Lambda^r$. The probability of selecting a particular migrant $e^i \in X_m$ from the i-th island equals $\Pr^m_{\Theta(D^i(x^i))}(e^i)$, according to (3) (see Observation 1).

4.2. Migration policies. We now assume that migrants come to an arbitrary i-th island deterministically along all possible paths, i.e., all paths starting on islands with numbers from I_i. Denote by

$$W^i = P^i \cup \bigcup_{j \in I_i} E^j = (U, \eta_{P^i} + \sum_{j \in I_i} \eta_{E^j}) \tag{12}$$

the population appearing on the i-th island after such a migration. The frequency vector $x^{W^i} \in X_{\mu_i + K_i \cdot m}$ of this population is given by the formula

$$x^{W^i} = \frac{1}{\mu_i + K_i \cdot m} \left(\mu_i x^i + m \sum_{j \in I_i} e^j \right). \tag{13}$$

Next, denote by

$$k_i = \frac{m}{\mu_i},\ i = 1, \ldots, s \tag{14}$$

the proportion between the size of migrants and the size of the i-th island's population. Then (13) gets the form

$$x^{W^i} = \frac{1}{1 + K_i \cdot k_i} \left(x^i + k_i \sum_{j \in I_i} e^j \right). \tag{15}$$

We now pass to the case in which migrants are transferred along connections selected randomly from the set $Path$. To this end, let us introduce a family of simple random decision variables

$$\omega_{ij} : X \to \mathcal{M}(\{0,1\}),\ (i,j) \in Path, \tag{16}$$

such that $\omega_{ij}(x)(1)$ is the probability of migration from the i-th to the j-th island assuming the state $x = (x^1, \ldots, x^s)$ of all islands. Moreover, we assume that $\omega_{ij}(x)$ is independent of $\omega_{kl}(x)$ if $(i,j) \neq (k,l)$ for all $x \in X$. These variables are evaluated by the master agent during the IM evolution (see Pseudocode 5.2).

Let us define the set of auxiliary random variables $\alpha^i : X \to \mathcal{M}(2^{I_i}), i = 1, \ldots, s$ such that for all $A \subset I_i$ we have

$$\alpha^i(x)(A) = \prod_{j \in A} \omega_{ij}(x)(1) \cdot \prod_{j \in I_i \setminus A} \omega_{ij}(x)(0). \tag{17}$$

Now the population appearing on the i-th island after the migration becomes also a random variable which takes the value

$$W^i_A = P^i \cup \bigcup_{j \in A} E^j = (U, \eta_{P^i} + \sum_{j \in A} \eta_{E_j}) \tag{18}$$

with the probability $\alpha^i(x)(A)$. The frequency vector related to W^i_A is now given by

$$x^{W^i_A} = \frac{1}{1 + K_A \cdot k_i} \left(x^i + k_i \sum_{j \in A} e^j \right), \tag{19}$$

where $K_A = \#A$. Of course, $x^{W^i_A} \in X_{\mu_i + K_A \cdot m}$.

4.3. Immigration policy. The general immigration policy on each island $i = 1, \ldots, s$ consists in performing a common selection from the native population together with all incoming migrants W^i. The result of such common selection serves in turn as a basis for the proper succession schemes SSS or VSS (see Section 2.2).

The common selection may be characterized by the family of operators

$$S^i : X \to \Lambda^r,\ i = 1, \ldots, s, \tag{20}$$

such that $S^i_\xi(x)$ stands for the probability of selecting an individual with the genotype $\xi \in U$ as a parent on the i-th island, assuming that $x \in X$ is the current state of the whole IM. The generic mapping family $\{S^i\}$, $i = 1, \ldots, s$, depends on both the emigration policy, i.e. the way migrants are distinguished from their source islands, and the migration policy, i.e., the way migrants are passed to the destination islands. Exact formulas defining these operators for emigration and migration policies described in Sections 4.1 and 4.2 will be drawn in the next section.

4.4. Probability distributions of common selection. Let us start with the simplest case in which distinguishing and sending migrants are performed in a deterministic way. Gathering the formulas (10) and (15) leads to the following observation.

Observation 2. If the island model is in a state $x = (x^1, \ldots, x^s)$ and the deterministic emigration and migration policies are applied, then the common selection is performed using the probability distribution $\Theta(S^i(x)) \in \mathcal{M}(U)$ so that

$$S^i(x) = F\left(\frac{1}{1 + K_i \cdot k_i} \left(x^i + k_i \sum_{j \in I_i} \Phi^j(x^j) \right) \right) \tag{21}$$

for all islands labeled by $i = 1, \ldots, s$.

Now we pass to the case in which migrants are sent along the edges randomly selected. If the global state of the IM equals $x \in X$, then migrants are coming to the i-th island from islands labeled by the elements of the set $A \subset I_i$ (see (17)) producing the extended population W_A^i whose frequency vector is $x^{W_A^i}$ (see (18), (19)) with the probability $\alpha^i(x)(A)$. Then applying the Bayes rule (see, e.g., Billingsley, 1995), we can extend the result of Observation 2 to the following one.

Observation 3. If the island model is in a state $x = (x^1, \ldots, x^s)$ and the deterministic emigration policy and the stochastic migration policy are applied, then for an arbitrary island labeled $i \in \{1, \ldots, s\}$ the common selection is performed using the probability distribution $\Theta(S^i(x)) \in \mathcal{M}(U)$ so that

$$S^i(x) = \sum_{A \subset I_i} \alpha^i(x)(A) \cdot F(x^{W_A^i}), \tag{22}$$

where $x^{W_A^i}$ is given by (19).

It is easy to notice that the random migration case generalizes the deterministic one as described before. It is enough to assume $\omega_{ij}(x)(1) = 1, \forall x \in X, \forall (i,j) \in Path$; then $\alpha^i(x)(I_i) = 1$ and $\alpha^i(x)(A) = 0$, $\forall A \neq I_i, \forall x \in X$, so (22) is reduced to (21).

Next, we will consider the case in which migrants are obtained by random sampling and then sent deterministically to the destination island. Let us assume the global IM state $X \ni x = (x^1, \ldots, x^s)$; then the probability of sampling the migrants $E^{j_1}, \ldots, E^{j_{K_i}}$ with the frequency vectors $e^{j_1}, \ldots, e^{j_{K_i}}$ from the island populations $P^{j_1}, \ldots, P^{j_{K_i}}$ equals $\prod_{\gamma=1,\ldots,K_i} \Pr^m_{\Theta(D^j(x^j))}(e^{j_\gamma})$, where $I_i = \{j_1, \ldots, j_{K_i}\}$, because sampling is performed independently on each island (see Section 4.1). Now, applying the Bayes rule (see, e.g., Billingsley (1995)), we can extend again the result of Observation 2.

Observation 4. If the island model is in a state $x = (x^1, \ldots, x^s)$ and the stochastic emigration policy and the deterministic migration policy are applied, then for an arbitrary island labeled $i \in \{1, \ldots, s\}$ the common selection is performed using the probability distribution $\Theta(S^i(x)) \in \mathcal{M}(U)$ so that (23) is satisfied, where $I_i = \{j_1, \ldots, j_{K_i}\}$.

Using a similar justification as in the case of Observation 3, we may extend the result of Observation 4 to the case of non-deterministic migrant transfer.

Observation 5. If the island model is in a state $x = (x^1, \ldots, x^s)$ and the stochastic emigration and migration policies are applied, then for an arbitrary island labeled $i \in \{1, \ldots, s\}$ the common selection is performed using the probability distribution $\Theta(S^i(x)) \in \mathcal{M}(U)$ so that

$$S^i(x) = \sum_{A \subset I_i} \alpha_A^i(x) \cdot S_A^i(x), \tag{24}$$

where $A = \{j_1, \ldots, j_{K_A}\} \subset I_i$ and $S_A^i(x)$ is defined by (25).

As mentioned before, the random migration case generalizes the deterministic one. Obviously, by setting $\omega_{ij}(x)(1) = 1, \forall x \in X, \forall (i,j) \in Path$, the formulas (24), (25) are reduced to (23).

5. Agent-based concurrent island model

The island model considered in this paper is governed by two kinds of autonomous agents:

- LA_i, $i = 1, \ldots, s$: *island agent*, manages the i-th island, contains the proper data structures representing the population P^i and actions used for proper operations on it. At the end of every generation the agent computes some predefined statistics (by calling the function $islandStatistics()$) and passes them to the master agent.

- MA: *master agent*, synchronizes actions of island agents, orders them to perform some groups of actions. The MA collects island statistics in order to verify the stopping condition (by calling the function $evaluateStopCondition()$).

The necessary synchronization among agents will be based on the following communication primitives:

- $send(address, m_1, m_2, \ldots)$ is used to send optional data (denoted by the list $m_1, m_2, \ldots$) to the agent with the provided $address$;

- $b\_receive(address, m_1, m_2, \ldots)$ is used to receive optional data (denoted by the list $m_1, m_2, \ldots$) from the agent with the provided $address$. The activity of the agent invoking $b\_receive$ is suspended until properly structured data from the given $address$ are received (blocking receive).

In order to perform the action of migration throughout the system, an additional mechanism for asynchronous communication is needed, because in the most general case migration between islands may become very complex (one location may send migrants to many of its neighbours and accept the migrants from many sources). Therefore, we leverage the concept of message queues and associate each local island with one of them. In this case, each location may send migrants to the message queue of its neighbors and receive migrants from others:

$$S^i(x) = \sum_{(e^{j_1},\ldots,e^{j_{K_i}})\in(X_m)^{K_i}} F\left(\frac{1}{1+K_i\cdot k_i}\left(x^i + k_i \sum_{\gamma=1,\ldots,K_i} e^{j_\gamma}\right)\right)\cdot \prod_{\gamma=1,\ldots,K_i} \Pr^m_{\Theta(D^j(x^j))}(e^{j_\gamma}), \quad (23)$$

$$S^i_A(x) = \sum_{(e^{j_1},\ldots,e^{j_{K_A}})\in(X_m)^{K_A}} F\left(\frac{1}{1+K_A\cdot k_i}\left(x^i + k_i \sum_{\gamma=1,\ldots,K_A} e^{j_\gamma}\right)\right)\cdot \prod_{\gamma=1,\ldots,K_A} \Pr^m_{\Theta(D^j(x^j))}(e^{j_\gamma}). \quad (25)$$

- $qsend(to, emigrant)$ simply sends the migrant to the neighboring islands whose addresses are contained in the set to;
- $qreceive(from)$ is more complex, it waits for all migrants coming from the islands whose addresses are contained in the set $from$ to appear in the message queue of the current island and then returns the set being a union of these migrants.

Each island agent LA_i may perform actions represented by the following primitives :

- $INIT_i()$ creates a new island population by μ_i-time sampling with replacement from U according to the probability distribution $\sigma_i \in \mathcal{M}(U)$ (see Section 6.1).
- $STEP_i(pop)$ performs one step of evolution starting from the population vector pop. This step is implemented by selection followed by genetic operations performed according to one of the possible succession models (VSS or SSS) (see Section 2.2).
- $NSTEP_i(pop)$ performs $nstep_i$ epochs of evolution on the i-th island starting from the population vector pop, where $nstep_i \geq 1$ stands for the parameter of each island (see Section 6.2).
- $EXPEL_i(pop)$ expels migrants from the i-th island population with the population vector pop (see Section 4). Two cases are considered:
  - deterministic: migrants are chosen according to the deterministic rule, e.g., by hard selection (see (10), (11) Section 4.1);
  - stochastic: the migrant is obtained by the m-times sampling with return from U, according to the probability distribution $D^i(x) \in \mathcal{M}(U)$, where $x \in X$ is the current state of the island model (see Section 4.1).
- $ACCEPTi(pop, immigrant)$ performs the common selection of the current island population denoted by the population vector pop together with migrants contained in the $immigrant$ structure and mixing, according to the succession rule specific for the i-th island.

Below, we will introduce pseudocodes of both the island and the master agent. The primitives introduced above, as well as in the previous and the next sections, were intensively utilized in their description. Moreover, for the arbitrary state of the island model $x \in X$ we will denote by $\underline{\delta(x)}, \underline{\omega_{ij}(x)} \in \{0,1\}, i,j = 1,\ldots,s$, the evaluation of the decision random variables, e.g., the results of the one-time sampling from $\{0,1\}$, according to the probability distributions $\delta(x), \omega_{ij}(x), i,j = 1,\ldots,s$, respectively (see Section 6.3 and the formula (16) in Section 4.2).

In Pseudocode 5.1, an algorithm for LA_i is presented. After initializing the island population, the island agent communicates its willingness to the master agent and awaits the order $reply$. If the order is $LOCAL$, it performs locally $nstep_i$ epochs of evolution. When it receives the $MIGR$ order from the master agent with appropriate parameters (neighbors that will expel and accept migrants), it uses its procedures to send migrants to the queues of its neighbors and accept migrants from them. The implementation of the $qreceive$ procedure may be based on pooling the queues until all expected migrants arrive. Then, the common selection and mixing are performed. If the set of incoming migrants is empty, then it performs a single step of the local evolution.

The master agent (see Pseudocode 5.2) waits for the willingness messages from all island agents and then chooses randomly whether a local or global (migration) action should be performed. If a local action is chosen, each LA_i is ordered to perform $nstep_i$ epochs of evolution. Otherwise, edges along which migration will occur are chosen randomly and island agents are notified of the addresses of those neighbours who are willing to accept and expel migrants. Moreover, the master agent receives statistical data from the island agents and evaluates the stopping condition. The information about reaching the stopping condition is reported to the island agents. The version of the MA algorithm for the deterministic migration can be obtained by a proper setting of the decision variables $\omega_{ij}(x), i,j = 1,\ldots,s$

(see Sections 4.2 and 4.4).

Pseudocode 5.1: ALGORITHM OF LA_i.

```
pop ← INIT_i()
stopCondition ← false
while not stopCondition
do { send(MA)
     b_receive(MA, reply, to, from)
     switch reply
       case LOCAL pop ← NSTEP_i(pop)
       case MIGR { if to ≠ ∅
                   then { emigrant ← EXPEL_i(pop)
                          qsend(to, emigrant)
                   if from ≠ ∅
                   then { immigrant ← qreceive(from)
                          pop ← ACCEPT_i(pop, immigrant)
                   else pop ← STEP_i(pop)
     send(MA, islandStatistics())
     b_receive(MA, stopCondition)
```

Pseudocode 5.2: ALGORITHM OF MASTER AGENT.

```
islandStatistics ← ∅
stopCondition ← false
local ← {i : i = 1, ..., s}
while not stopCondition
do { for each j ∈ local do b_receive(j)
     if δ(x) = 0
       then for each j ∈ local do send(j, LOCAL)
       else { edges ← ∅
              for each (i, j) ∈ Path
                do { if ω_ij(x) = 1
                     then edges ← edges ∪ (i, j)
              for each j ∈ local
                do send(LA_j, MIGR,
                   {k : (k, j) ∈ edges},
                   {k : (j, k) ∈ edges})
     for each j ∈ local
       do { b_receive(j, iStat)
            islandStatistics ←
            islandStatistics ∪ {iStat}
     stopCondition ←
     evaluateStopCondition(islandStatistics)
     for each j ∈ local do send(j, stopCondition)
```

6. Stochastic dynamics of the island model for finite populations

6.1. Initial step. The island model starts with an "initial step" in which the island populations are randomly initialized according to the scheme described by Observation 1. The probability of sampling initial population P^i with the frequency vector $x^i \in X_{\mu_i}$ is described by the polynomial distribution $\Pr_{\sigma^i}^{\mu_i}(x^i)$ (see the formula (3)), where $\sigma^i \in \mathcal{M}(U)$ are probability distributions given arbitrarily for all $i \in 1, \ldots, s$.

6.2. Local state transitions. The "local step" consists in executing $nstep_i$ genetic epochs concurrently by each i-th island.

Observation 6. The probability transition function of $nstep_i$ epochs of local evolution on the i-th island $\tau_{nstep}^i : X_{\mu_i} \to \mathcal{M}(X_{\mu_i})$ is given by the following formula:

$$\tau_{nstep}^i(x^i, y^i) = ((Q^i)^{nstep_i})_{x^i y^i}, \quad \forall x^i, y^i \in X_{\mu_i}. \tag{26}$$

Observation 7. The probability transition function $\tau_{cs+m}^i : X \to \mathcal{M}(X_{\mu_i})$ of the common selection on the i-th island composed of mixing is given by the following formula:

$$\tau_{cs+m}^i(x, y^i) = \begin{cases} \sum\limits_{z^i \in X_{\mu_i}} \Pr_{\Theta(S^i(x))}^{\mu_i}(z^i) \cdot \Gamma_{z^i\, y^i}^i & \text{for SSS,} \\ \Pr_{\Theta(M^i(S^i(x)))}^{\mu_i}(y^i) & \text{for VSS,} \end{cases} \tag{27}$$

where $x \in X$ is the current state of the IM, and $y^i \in X_{\mu_i}$ is the next step population on the i-th island. $S^i(x)$ is computed using one of the formulas (21)–(25), according to the type of migration policy.

6.3. Global state transitions. Immediately from Observation 6 and from the isolation of islands during the "local step", the following observation can be drawn.

Observation 8. Let $X \ni x = (x^1, \ldots, x^s)$ be the current state of the island model; then the transition probability function for the "local step" $\tau_{loc} : X \to \mathcal{M}(X)$ is given by the following formula:

$$\tau_{loc}(x, y) = \prod_{i=1,\ldots,s} \tau_{nstep}^i(x^i, y^i), \ \forall x, y \in X. \tag{28}$$

Similarly, from Observation 7 and owing to immigration, common selection and mixing are performed independently on each island, we may obtain the following observation.

Observation 9. Let $X \ni x = (x^1, \ldots, x^s)$ be the current state of the island algorithm; then the probability transition function for the "migration step" $\tau_{migr} : X \to \mathcal{M}(X)$ is given by the following formula:

$$\tau_{migr}(x, y) = \prod_{i=1,\ldots,s} \tau^i_{cs+m}(x, y^i), \quad \forall x, y \in X. \quad (29)$$

We assume that the execution of the "migration step" is performed with the probability $p_{migr}(x)$ that also depends on $x \in X$, therefore the family of random variables $\{\delta(x)\}_{x\in X} \subset \mathcal{M}(\{1, 0\})$ with the probability distributions $\{p_{migr}(x), 1 - p_{migr}(x)\}_{x\in X}$ is given. Of course the "local step" may be performed with the probability $1 - p_{migr}(x)$.

Observation 10. The probability transition function $\tau : X \to \mathcal{M}(X)$ for the island algorithm is given by the following formula:

$$\tau(x, y) = p_{migr}(x)\, \tau_{migr}(x, y) + (1 - p_{migr}(x))\, \tau_{loc}(x, y), \quad \forall x, y \in X.$$

6.4. Features of the Markov model. In this subsection we shall examine some asymptotic features of the described Markov model. All of them are connected with the ergodicity of the system. Let us then start with some remarks on Markov chain ergodicity. There exist several definitions of this notion and we shall use the one by Iosifescu (1980, Section 2.6.1). According to the author, a finite Markov chain is called ergodic if it is *irreducible*, i.e., if all states of the chain communicate, which means that it is possible to get from every state to every other state with a positive probability in a finite number of steps. In fact, it turns out (see below) that our chain possesses a stronger property. Namely, it is *regular*, which means that its states are aperiodic. Note that many authors (see, e.g., Kushner, 1971, Section 2.7.2) include, aperiodicity of states to the definition of the ergodic Markov chain.

Theorem 1. *If all mixing operators are strictly positive (i.e.* $(M^i(x^i))_\xi > 0 \; \forall\, \xi \in U, \forall x^i \in X_{\mu_i}, i = 1, \ldots, s$*), then the Markov chain associated with the island model is regular (i.e., ergodic in the stronger sense).*

Proof. If $(M_i(x))_\xi$ are positive for all $\xi \in U, x \in X_{\mu_i}, i = 1, \ldots, s$, then all matrices $Q^i, i = 1, \ldots, s$ have only strictly positive entries (see the formulas (3), (9)). Moreover, for the same reason, $\tau^i_{cs+m}(x, y)$ takes only positive values for all $x, y \in X$. Finally, $\tau_{migr}(x, y)$, $\tau_{loc}(x, y)$ and $\tau(x, y)$ are also strictly positive for all $x, y \in X$. The probability of reaching an arbitrary state $y \in X$ from the arbitrary previous one $x \in X$ in a single step is also strictly positive. Thus all states communicate and have period 1 (i.e., they are aperiodic). ■

Remark 1. Given the assumptions of Theorem 1, the island model possesses an asymptotic guarantee of success (Horst and Pardalos, 1995; Rinnoy Kan and Timmer, 1987). The ergodicity guarantees that the island model can reach any state from the space X in a finite number of steps. In particular, it can reach all states representing populations containing local extrema.

Corollary 1. *A simple issue of the ergodic theorem (see, e.g., Billingsley, 1995) is that for all* $x \in X$ *the sequence* $\{\tau^p(x, \cdot)\}$ *converges weakly to some measure* $\varrho_{\mu_1,\ldots,\mu_s} \in \mathcal{M}(X)$ *while* $p \to \infty$*. The measure* $\varrho_{\mu_1,\ldots,\mu_s}$ *does not depend on the starting state* $x \in X$ *and is strictly positive (i.e.,* $\varrho_{\mu_1,\ldots,\mu_s}(\{y\}) > 0, \forall y \in X$*).*

Observation 11. Observations 6–10 can be applied to the case of the most conventional island model in which SGA constitutes the basic mechanism governing island populations. Then $U = \Omega$ will be the universum of binary genotypes, $\#U = r = 2^l$, where l stands for the length of binary strings, F is the proportional selection operator common to all islands and M_i is the mixing operator associated with the binary mutation and crossover on the i-th island (see Vose, 1998). Moreover, $nstep_i = 1$, $i = 1, \ldots, s$. Theorem 1 and Corollary 1 are also true in this case when the mutation rate (the probability of changing any arbitrary bit in each genotype) is strictly positive (see Vose, 1998).

Remark 2. Note that any probabilistic measure $\varrho \in \mathcal{M}(X)$ can be naturally extended by zero to a probabilistic measure on the whole $(\Lambda^r)^s$. We shall identify ϱ with such an extension.

Theorem 2. *Let* $\{(\mu^n_1, \ldots, \mu^n_s)\}_{n=1,2,\ldots}; \mu^n_i \in \mathbb{N}, i = 1, \ldots, s$ *be any sequence such that* $\mu^n_i \to \infty$ $(n \to \infty)$ *for all* $i = 1, \ldots, s$*. Then the associated sequence* $\{\varrho_{\mu^n_1,\ldots,\mu^n_s}\}$ *of invariant measures (cf. Corollary 1) has a subsequence which converges weakly to a probabilistic measure* $\varrho^* \in \mathcal{M}((\Lambda^r)^s)$*.*

Proof. According to Remark 2, measures $\varrho_{\mu^n_1,\ldots,\mu^n_s}$ can be considered measures over the whole $(\Lambda^r)^s$, so $\varrho_{\mu^n_1,\ldots,\mu^n_s} \in \mathcal{M}((\Lambda^r)^s)$. Let us note that, since Λ^r is compact, so is $(\Lambda^r)^s$. Thus the following obvious equality:

$$\varrho_{\mu^n_1,\ldots,\mu^n_s}((\Lambda^r)^s) = \varrho_{\mu^n_1,\ldots,\mu^n_s}(X) = 1, \quad \forall\, (\mu^n_1, \ldots, \mu^n_s) \in \mathbb{N}^s$$

implies that the sequence $\{\varrho_{\mu^n_1,\ldots,\mu^n_s}\}$ is *tight* (see Billingsley, 1995). Then the thesis is a straightforward consequence of Prokhorov's theorem (Billingsley, 1995, Theorem 29.3). ■

7. Conclusions

Parallel, multi-deme evolutionary algorithms usually outperform the single-population ones (e.g., because of the reduction of the computation time and the ability to perform a much broader search). However, the problem of the asymptotic guarantee of success in their existing mathematical models does not seem to have been addressed extensively.

We have presented an island-like model for parallel evolutionary algorithms governed by software agents, which deals in a rigorous way with population activity scheduling (see Section 5). This approach has allowed us to use the single stationary Markov chain as the basis for our model of the whole system.

The appropriate space of system states and stochastic operators, including special common selection operators (see (20)) used for migration purposes, were introduced. The Markov transition probability function was effectively established (see Observations 6–10).

We have shown that under a mild assumption (the mixing operators are strictly positive) the resulting Markov chain is ergodic (see Theorem 1). Moreover, the sequence of the related sampling measures converges to some invariant measure (see Corollary 1). In particular, this conclusion is true if the simple genetic algorithm with a strictly positive mutation governs the evolution on each island (see Observation 11).

The asymptotic guarantee of success for the island model was also obtained as a simple issue of ergodicity. The island model can reach any state from the space X (in a finite number of steps), and therefore all states representing populations containing local extrema are reachable (see Remark 1).

If the cardinality of each island population grows to infinity, then the sequence of the limit invariant measures contains a weakly convergent subsequence (see Theorem 2).

The global optimization problem itself does not affect in any way our formalism, and therefore our deliberations are not problem-dependent as in some other works (e.g., Whitley *et al*., 1997).

The model proposed can be extended to the case of various encodings used for creating genetic spaces for each island, as well as to the case of various selection and common selection schemes on each island. Allowing such relaxation would lead to a considerable growth of complexity and make the paper unreadable.

The formal description of the island model obtained for the case of solving a single-objective problem may be extended to the multi-objective case. The selection operator on each island must be determined by the proper selection operator as in the work of Gajda *et al*. (2010), and the emigration policy has to be properly designed.

Moreover, it seems that the agent-based scheduling scheme suggested by the authors can be extended to other types of multi-deme genetic strategies, so that the asymptotic of these strategies might be analysed in the same way as in the case of the island model.

Acknowledgment

The work presented in this paper was partially supported by the Polish National Science Centre Project No. N 519 405737, and by the grant *Biologically inspired mechanisms in planning and management of dynamic environments* funded by the Polish National Science Centre (No. N N516 500039).

References

Alba, E. and Tomassini, M. (2002). Parallelism and evolutionary algorithms, *IEEE Transactions on Evolutionary Computation* **6**(5): 443–462.

Aparicio, J., Correia, L. and Moura-Pires, F. (1999). Populations are multisets-plato, *in* W. Banzhaf, J. Daida, A.E. Eiben, M.H. Garzon, V. Honavar, M. Jakiela and R.E. Smith (Eds.), *Proceedings of the Genetic and Evolutionary Computation Conference, Orlando, Florida, USA, 13–17 July 1999*, Vol. 2, Morgan Kaufmann, San Francisco, CA, pp. 1845–1850.

Bäck, T., Fogel, D. and Michalewicz, Z. (2000). *Evolutionary Computation: Basic Algorithms and Operators*, Vols. 1 and 2, Institute of Physics Publishing, Bristol/Philadelphia, PA.

Back, T., Hammel, U. and Schwefel, H.-P. (1997). Evolutionary computation: Comments on the history and current state, *IEEE Transactions on Evolutionary Computation* **1**(1): 3–17.

Billingsley, P. (1995). *Probability and Measure*, Wiley-Interscience, Hoboken, NJ.

Brabazon, A. and O'Neill, M. (2006). *Biologically Inspired Algorithms for Financial Modeling*, Springer Verlag, Berlin/Heidelberg.

Buckley, F., Nicol, S. and Pollett, P. (2010). Preface to the selected papers on modeling and control of metapopulation networks, *Ecological Modeling* **221**(21): 2512–2514.

Byrski, A. and Schaefer, R. (2009). Stochastic model of evolutionary and immunological multi-agent systems: Mutually exclusive actions, *Fundamenta Informaticae* **95**(2–3): 263–285.

Cantú-Paz, E. (1995). A summary of research on parallel genetic algorithms, *IlliGAL Report No. 95007,* University of Illinois, Chicago, IL.

Cantú-Paz, E. (2000). *Efficient and Accurate Parallel Genetic Algorithms*, Kluwer Academic Publishers, Norwell, MA.

Davis, T.E. and Principe, J.C. (1991). A simulated annealing like convergence theory for the simple genetic algorithm, *Proceedings of the 4th International Conference on Genetic Algorithms, San Diego, CA, USA*, pp. 174–181.

Diekert, V. and Rozenberg, G. (1995). *The Book of Traces*, World Scientific, Singapore.

Droste, S., Jansen, T. and Wegener, I. (1998a). On the optimization of unimodal functions with the (1+1) evolutionary algorithm, *Proceedings of the 5th International Conference on Parallel Problem Solving from Nature, Amsterdam, The Netherlands*, pp. 13–22.

Droste, S., Jansen, T. and Wegener, I. (1998b). A rigorous complexity analysis of the (1+1) evolutionary algorithm for separable functions with Boolean inputs, *Evolutionary Computation* **6**(2): 185–196.

Gajda, E., Schaefer, R. and Smołka, M. (2010). Evolutionary multiobjective optimization algorithm as a Markov system, *Proceedings of the 11th International Conference on Parallel Problem Solving from Nature, PPSN XI, Kraków, Poland*, pp. 617–626.

Goldberg, D.E. and Segrest, P. (1987). Finite Markov chain analysis of genetic algorithms, *Proceedings of the 2nd International Conference on Genetic Algorithms on Genetic Algorithms and Their Application, Cambridge, MA, USA*, pp. 1–8.

Gordon, V., Whitley, D. and Bohn, A. (1992). Data flow parallelism in genetic algorithms, *in* R. Manner and B. Manderick (Eds.), *Parallel Problem Solving from Nature 2*, Elsevier Science, Amsterdam, pp. 553–542.

Grochowski, M., Schaefer, R. and Uhruski, P. (2004). Diffusion based scheduling in the agent-oriented computing systems, *in* R. Wyrzykowski, J. Dongarra, M. Paprzycki and J. Waśniewski (Eds.), *Parallel Processing and Applied Mathematics*, Lecture Notes in Computer Science, Vol. 3019, Springer, Berlin/Heidelberg, pp. 97–104.

Harik, G., Cantú-Paz, E., Goldberg, D.E. and Miller, B.L. (1999). The gambler's ruin problem, genetic algorithms, and the sizing of populations, *Evolutionary Computation* **7**(3): 251–253.

Hennessy, M. (1988). *Algebraic Theory of Processes*, The MIT Press, Cambridge, MA.

Hewitt, C., Bishop, P. and Steiger, R. (1973). A universal modular ACTOR formalism for artificial intelligence, *Proceedings of the 3rd International Joint Conference on Artificial Intelligence, Stanford, CA, USA*, pp. 235–245.

Horn, J. (1993). Finite Markov chain analysis of genetic algorithms with niching, *Proceedings of the 5th International Conference on Genetic Algorithms, Urbana-Champaign, IL, USA*, pp. 110–117.

Horst, R. and Pardalos, P. (1995). *Handbook of Global Optimization*, Kluwer, Norwell, MA.

Iosifescu, M. (1980). *Finite Markov Processes and Their Applications*, John Wiley & Sons, Alphen aan den Rijn.

Kołodziej, J. and Xhafa, F. (2011). Modern approaches to modeling user requirements on resource and task allocation in hierarchical computational grids, *International Journal of Applied Mathematics and Computer Science* **21**(2) 243–257, DOI: 10.2478/v10006-011-0018-x.

Kowalczuk, Z. and Białaszewski, T. (2006). Niching mechanisms in evolutionary computations, *International Journal of Applied Mathematics and Computer Science* **16**(1): 59–84.

Kushner, H. (1971). *Introduction to Stochastic Control*, Rinehart and Winston, Holt.

Lässig, J. and Sudholt, D. (2010). General scheme for analyzing running times of parallel evolutionary algorithms, *in* R. Schaefer, C. Cotta, J. Kołodziej and G. Rudolph (Eds.), *Proceedings of the 11th International Conference on Parallel Problem Solving from Nature: Part I*, Springer-Verlag, pp. 234–243.

Li, C. and Yang, S. (2008). An island based hybrid evolutionary algorithm for optimization, *in* X. Li, M. Kirley, M. Zhang, D.G. Green, V. Ciesielski, H.A. Abbass, Z. Michalewicz, T. Hendtlass, K. Deb, K.C. Tan, J. Branke and Y. Shi (Eds.), *SEAL*, Lecture Notes in Computer Science, Vol. 5361, Springer, Berlin/Heidelberg, pp. 180–189.

Liekens, A. (2005). *Evolution of Finite Populations in Dynamic Environments*, Ph.D. thesis, Technische Universiteit Eindhoven, Eindhoven.

Mahfoud, S. (1991). Finite Markov chain models of an alternative selection strategy for the genetic algorithm, *Complex Systems* **7**(2): 155–170.

Manderick, B. and Spiessens, P. (1989). Fine-grained parallel genetic algorithms, *in* J. Schaffer (Ed.), *Proceedings of the Third International Conference on Genetic Algorithms*, Morgan Kauffman, San Francisco, CA, p. 428.

Mesghouni, K., Hammadi, S. and Borne, P. (2004). Evolutionary algorithms for job-shop scheduling, *International Journal of Applied Mathematics and Computer Science* **14**(1): 91–103.

Milner, R. (1990). Functions as processes, *in* M. Paterson (Ed.), *Automata, Languages and Programming*, Lecture Notes in Computer Science, Vol. 443, Springer, Berlin/Heidelberg, pp. 167–180.

Mühlenbein, H. (1989). Parallel genetic algorithms, population genetic and combinatorial optimization, *in* J. Schaffer, (Ed.), *Proceedings of the Third International Conference on Genetic Algorithms*, Morgan Kauffman, San Francisco, CA, pp. 416–421.

Mühlenbein, H. (1992). How genetic algorithms really work: Mutation and hillclimbing, *in* R. Männer and B. Manderick (Eds.), *Proceedings of PPSN '92*, Elsevier, Amsterdam, pp. 15–26.

Nagylaki, T. (1979). The island model with stochastic migration, *Genetics* **91**(1): 163–76.

Nix, A.E. and Vose, M.D. (1992). Modeling genetic algorithms with Markov chains, *Annals of Mathematics and Artificial Intelligence* **5**(1): 79–88.

Paredis, J. (1998). Coevolutionary algorithms, *in* T. Bäck, D. Fogel and Z. Michalewicz (Eds.), *Handbook of Evolutionary Computation*, 1st Suppl., IOP Publishing/Oxford University Press, Bristol/Oxford.

Peterson, J.L. (1981). *Petri Net Theory and the Modeling of Systems*, Prentice Hall, Upper Saddle River, NJ.

Potter, M.A. and De Jong, K.A. (2000). Cooperative coevolution: An architecture for evolving coadapted subcomponents, *Evolutionary Computation* **8**(1): 1–29.

Rinnoy Kan, A. and Timmer, G. (1987). Stochastic global optimization methods, *Mathematical Programming* **39**: 27–56.

Rudolph, G. (1994). Massively parallel simulated annealing and its relation to evolutionary algorithms, *Evolutionary Computation* **1**(4): 361–383.

Rudolph, G. (1997). Stochastic processes (Chapter B.2.2), Models of stochastic convergence (Chapter B.2.3), *in* T. Bäck, D.B. Fogel and Z. Michalewicz (Eds.), *Handbook of Evolutionary Computations*, Oxford University Press, Oxford.

Rudolph, G. (2006). Takeover time in parallel populations with migration, *Proceedings of the 2nd International Conference on Bioinspired Optimization Methods and Their Applications (BIOMA 2006), Ljubljana, Slovenia*, pp. 63–72.

Schaefer, R., Byrski, A., Kołodziej, J. and Smołka, M. (2012). An agent-based model of hierarchic genetic search, *Computers and Mathematics with Applications,* DOI: 10.1016/j.camwa.2012.02.052, (accepted).

Schaefer, R., Byrski, A. and Smołka, M. (2009). Stochastic model of evolutionary and immunological multi-agent systems: Parallel execution of local actions, *Fundamenta Informaticae* **95**(2–3): 325–348.

Schaefer, R. and Telega, H. (2007). *Foundation of Global Genetic Optimization*, Studies in Computational Intelligence, Vol. 74, Springer Verlag, Berlin/Heidelberg/New York, NY.

Schmitt, L.M. (2001). Theory of genetic algorithm, *Theoretical Computer Science* **259**(1): 1–61.

Skolicki, Z. (2007). *An Analysis of Island Models In Evolutionary Computation*, Ph.D. thesis, George Mason University, Fairfax, VA.

Skolicki, Z. and de Jong, K. (2004). Improving evolutionary algorithms with multi-representation island models, *8th International Conference on Parallel Problem Solving from Nature, PPSN, Birmingham, UK*, pp. 420–429.

Suzuki, J. (1993). A Markov Chain Analysis on a Genetic Algorithm, *in* S. Forrest (Ed.), *Proceedings of the 5th International Conference on Genetic Algorithms, Urbana-Champaign, IL, USA, June 1993*, Morgan Kaufmann, San Francisco, CA, pp. 146–154.

Terzo, O. Mossucca, L., Cucca, M. and Notarpietro R. (2011). Data intensive scientific analysis with grid computing, *International Journal of Applied Mathematics and Computer Science* **21**(2): 219–228, DOI: 10.2478/v10006-011-0016-z.

Tomassini, M. (2005). *Spatially Structured Evolutionary Algorithms: Artificial Evolution in Space and Time*, Natural Computing Series, Springer, Berlin/Heidelberg.

Vose, M. (1998). *The Simple Genetic Algorithm: Foundations and Theory*, MIT Press, Cambridge, MA.

Vose, M. and Liepins, G. (1991). Punctuated equilibria in genetic search, *Complex Systems* **5**: 31–44.

Whitley, D. (1992). An executable model of a simple genetic algorithm, *in* L.D. Whitley (Ed.), *Foundations of Genetic Algorithms 2*, Morgan Kaufmann, San Francisco, CA, pp. 45–62.

Whitley, W.D., Rana, S.B. and Heckendorn, R.B. (1997). Island model genetic algorithms and linearly separable problems, *in* D. Corne and J.L. Shapiro (Eds.), *Selected Papers from the AISB Workshop on Evolutionary Computing*, Springer-Verlag, London, pp. 109–125.

Wolpert, D.H. and Macready, W.G. (1997). No free lunch theorems for optimization, *IEEE Transactions on Evolutionary Computation* **1**(1): 67–82.

Wood, G.R. and Zabinsky, Z.B. (2002). Stochastic adaptive search, *in* P.M. Pardalos and H.E. Romeijn (Eds.), *Handbook of Global Optimization,* Vol. 2, Kluwer, Norwell, MA.

Robert Schaefer is a full professor at the Department of Computer Science and Electronics, AGH University of Science and Technology in Kraków, Poland. His main, recent areas of research include genetic algorithms in solving continuous global optimization problems and computing multi-agent systems. His former research areas were modeling the blood flow in arteries modeling nonlinear flows in porous media.

Aleksander Byrski obtained a Ph.D. in 2007 at the AGH University of Science and Technology in Kraków. His research interests include agent-based computation, biological-inspired computation and artificial intelligence.

Maciej Smołka obtained a Ph.D. in 2000 at the Jagiellonian University in Kraków. His research interests include stochastic modeling of computational systems, artificial intelligence, stochastic and deterministic optimal control.

Received: 23 November 2011
Revised: 19 May 2012