Migration policies in dynamic island models

Frédéric Lardeux, Jorge Maturana, Eduardo Rodriguez-Tello, Frédéric Saubion

Natural Computing manuscript No.
(will be inserted by the editor)

On Dynamic Migration Policies in Island Models

Frédéric Lardeux · Jorge Maturana · Frédéric Saubion

Received: date / Accepted: date

Abstract Dynamic island models are population-based algorithms for solving optimization problems, where the individuals of the population are distributed on islands. These subpopulations of individuals are processed by search algorithms on each island. In order to share information within this distributed search process, the individuals migrate from their initial island to another island at regular steps. In dynamic island models, the migration process evolves during the search according to the observed performance on the different islands. The purpose of this dynamic/adaptive management of the migrations is to send the individuals to the most promising islands, with regards to their current states. Therefore, this approach is related to the adaptive management of search operators in evolutionary algorithms. In this work, our main purpose is thus precisely analyses dynamic migration policies. We propose a testing framework that assigns gains to the algorithms applied on the islands in order to assess the adaptive ability of the migration policies, with regards to various situations. Instead of having one dynamic migration policy that is applied to the whole search process, as it has already been studied, we propose to associate a migration policy to each individual, which allows us to use simultaneously different migration policies.

Keywords Adaptive Evolutionary Algorithms · Island Models

F. Lardeux
LERIA, University of Angers (France)
E-mail: frederic.lardeux@univ-angers.fr

J. Marturana
Universidad Austral de Chile (Chile)
E-mail: jorge.maturana@inf.uach.cl

F. Saubion
LERIA, University of Angers (France)
E-mail: frederic.saubion@univ-angers.fr

1 Introduction

Island models (IM) [Whitley et al(1998),Skolicki(2007)] have been introduced in evolutionary computation in order to avoid premature convergence in population-based algorithms when solving optimization problems. The main idea of IM is to use a set of sub-populations instead of a panmictic one, in order to improve the performance of the evolutionary search process. IM are thus closely related to parallel evolutionary computation [Luque and Alba(2011),Melab et al(2005)]. Each sub-population evolves independently on each island and interacts periodically with other islands by means of migrations [Rucinski et al(2010)]. The impact of migration has been carefully studied [Lässig and Sudholt(2013),Luque and Alba(2010)]. Migrations may be actually used in order to reinforce the most efficient islands [Skolicki and Jong(2005),Gustafson and Burke(2006),Araujo et al(2009)]. Note that the impact of the frequency of migrations has been studied in [Mambrini and Sudholt(2014)]. Considering the same algorithm on all islands, one may be interested in assessing the convergence ability by evaluating two complementary aspects: (1) the ability to converge on each island (e.g., the ability to obtain the best individuals on all islands using for instance the notion of takeover time [Rudolph(2000), Luque and Alba(2010)]) and (2) the ability to ensure a good global diversity to avoid sub-populations to get stuck in local optima and finally reach a global optima (e.g., using specific problems, difficult to handle with a single panmictic population [Lässig and Sudholt(2010)]).

Context
Classically, in above-mentioned works, islands models use the same algorithm on each island and the islands differ only by their populations. In [Lardeux and Goëffon(2010)] it has been proposed to consider different algorithms on the islands - restricted in fact to a basic evolutionary algorithm with only one variation operator - and to define dynamic migration policies. In this approach, called Dynamic Island Models (DIM), migration probabilities are modified during the evolutionary process according to the impact of previous analogue migrations, by means of a learning process. Compared to classic island models DIM has be indeed related to adaptive operator selection techniques for evolutionary algorithms [Da Costa et al(2008)Da Costa] since only one operator is used on each island. DIM should be able to identify a subset of islands that are currently the most suitable for improving individuals, but also to quickly react to changes when new operators become more efficient. Therefore, DIM can be compared to other classic operator selection mechanisms [Candan et al(2012)].

Contributions
The purpose of this paper is to carefully study different configuration of the DIM with dynamic migration policies, as well as their ability to adapt to changes during the solving process. Such changes occur when the solving process explores different areas of the search space. Therefore, the basic search operators (or heuristics) may become more or less efficient according to the current state of the search. Considering the case where each island may use its own specific search algorithm, we propose here a testing scenarios in order to simulate the evolution of the search efficiency on the islands. In such scenarios, gains are associated to operators of the island in order to reflect their performances. Compared to previous testing scenarios (see for instance [Thierens(2005),Da Costa et al(2008)Da Costa,

Candan et al(2013)], we consider here a gain matrix that takes into account possible interactions between operators, i.e., the efficiency of an operator applied on a given individual may depend on the previous operators applied on it. This is motivated by the fact that, in search processes, such dependencies may occur between operators alternating intensification and diversification stages or using complementary neighborhoods, for instance by means of local search based operators. We consider several types of scenarios from fixed scenarios to different dynamic scenarios that may reflect different possible search processes.

While in previous works [Lardeux and Goëffon(2010), Candan et al(2012), Candan et al(2013)] the same dynamic migration policy has been investigated, we propose here to study several possible configuration of the DIM, by considering more possible components, including learning and migration processes. Moreover, instead of having a predefined migration policies, we propose to take advantage of this multi-individuals algorithm by associating policies to individuals. This cooperative model allows us to use simultaneously several migration policies in order to benefit from their respective properties. Our study highlights that DIM is efficient for tracking interactions between islands and to quickly react to efficiency changes during the search.

Organization of the paper

This paper is organized as follows. Section 2 presents the dynamic island model. Section 3 discusses how to measure the efficiency of migration policies. The experiments are presented in Section 4 to finally draw conclusions and outline future works.

2 Dynamic Island Models

In this section we propose to generalize the definition of dynamic island models proposed in [Lardeux and Goëffon(2010)] by considering different possible options for the main components of the algorithm that manage the migration process. The purpose is to evaluate the relative advantages of the resulting possible configurations. A Dynamic Island Model (DIM) is defined by:

- its size n.
- a set of islands $\mathcal{I} = \{i_1, \cdots, i_n\}$ and a set of algorithms $\mathcal{A} = \{a_1, \cdots, a_n\}$. Each algorithm a_k is assigned to island i_k.
- a set of populations $\mathcal{P} = \{p_1, \cdots, p_n\}$. Each population is a subset of individuals. Each population p_k is assigned to island i_k. The size of the entire population is fixed but the size of each p_k changes continuously according to the migrations. $a_k(p_k)$ is the population obtained after applying algorithm a_k on population p_k.
- a topology given by an undirected graph $(\mathcal{I}, V)$ where $V \subseteq \mathcal{I} \times \mathcal{I}$ is a set of edges between islands (here we will consider a complete graph).
- an initial migration matrix M of size $n \times n$ with $M(i,j) \in [0..1]$. M is supposed to be coherent with the topology, i.e., if $(i,j) \notin V$ then $M(i,j) = 0$, $\mathcal{M}$ is the set of migration matrices.
- a migration policy $\Pi : \mathcal{I} \times \mathcal{M} \rightarrow \mathcal{I}$ that selects a migration island given an initial island and a migration matrix.

```
input  : a DIM, a gain function
output: a solution s*
local  : a reward matrix R of size n × n
s* ← best(P);
R ← 0;
Initialiaze(M);
while not stop condition do
    for k ← 1 to n do
        Reward(R, p_k);
        p_k ← a_k(p_k);
        for s ∈ p_k do
            i_l ← Migrate(i_k, M);
            p_l ← p_l ∪ {s};
            p_k ← p_k \ {s};
        Learn(M, R);
    b ← best(P);
    if b > s* then
        s* ← b;
```

Algorithm 1: Dynamic Island Model

Given a DIM, its computational behaviour is described by Algorithm 1.

Description of the components of the algorithm:

– Note that, in this paper, we do not address a particular optimization problem, but we rather aim at testing scenarios in order to evaluate different possible settings of the DIM. Therefore, we define a notion of gain associated to each algorithm located on the islands that simulates the effect of its application on the individuals of the population. For instance, this gain can be the fitness improvement with regards to a classic optimization problem. Of course, this approach does not take into account the fact that the performance of an algorithm a depends, most of the time, on the semantics - phenotype and/or genotype - of the individuals in real problem. Nevertheless, such testing scenarios for EAs have been widely used for studying control of operators [Thierens(2005), Da Costa et al(2008)Da Costa] and are indeed useful for evaluating general properties of these adaptive control mechanisms.

Definition 1 (Gain and fitness of individuals) We consider a function $gain : \mathcal{A} \times \mathbb{N} \to \mathbb{R}$, such that $gain(a, t)$ is the gain of algorithm a when processed at iteration t of the DIM. Individuals may be abstracted by the sum of their successive gains. For an individual $s \in p_i$ at iteration t, we define its value (fitness) at iteration t as $v(s, t) = \Sigma_{\tau=1}^{t} gain(a_{s(\tau)}, \tau)$, where $s(\tau)$ is the island where s was located at iteration τ.

– The value $R(i, j)$ of reward matrix R evaluates the benefit (by means of fitness improvement) of sending individual from island i to island j. R is used to update the migration matrix M by means of a reinforcement learning based process. Note that R is initialized with 0 values. M can be initialized with equal values for all $M(i, j)$ corresponding to equal probabilities of migration for any pair of islands.
– The function *best* computes the best current individual of the whole population, $best(\mathcal{P}) = best(\cup_{i \in \mathcal{I}}(p_i))$, according to their values.

– The stop condition is, as usual, a limited number of iterations or the fact that an optimal solution has been found in the global population $\mathcal{P}$.

Let us now focus on the most important components. Since M and R will be changed at each iteration of the algorithm, let us denote $M^{(t)}$ and $R^{(t)}$ the value of these matrices at iteration t of the algorithm.

2.1 Learn Function

Basic Reinforcement Learning Function

The basic learning principle consists in sending more individuals to the islands that have previously improved individuals coming from the current island and less to the islands that are currently less efficient. The learning process is achieved by an adaptive update of the migration matrix at iteration t, $M^{(t)}$, performed as:

$$M^{(t+1)}(i,k) = (1-\beta)(\alpha.M^{(t)}(i,k) + (1-\alpha)R_i^{(t)}(k)) + \beta.N^{(t)}(k)$$

where $N^{(t)}$ is a stochastic noise vector. The parameter α represents the importance of the knowledge accumulated (inertia or exploitation) and β is the amount of noise, which is necessary to explore alternative actions. The influence of these parameters has been studied in [Candan et al(2012)].

QLearning Based Function

We use here a classic QLearning (see [Sutton and Barto(1998)] for more details) algorithm in order to update the transition matrix. Compared to previous approach learning function, QLearning takes into account an estimation of the future optimal value that can be obtained after a migration has been performed.

$$M^{(t+1)}(i,k) = M^{(t)}(i,k) + \delta(R_i^{(t)}(k) + \gamma max_j M^{(t)}(k,j) - M^{(t)}(i,k))$$

where δ is the learning rate and γ is a discount factor that allows to control the importance of the estimation of expected future gains.

2.2 Reward Function

$R_i^{(t)}(k)$ corresponds to the reward assigned to individuals that were on island i at iteration $t-1$ and that have been processed on island k at iteration t. We consider two possible reward functions for computing $R_i^{(t)}(k)$.

$$\textbf{Elitist Reward: } R_i^{(t)}(k) = \begin{cases} \frac{1}{|B|} & \text{if } k \in B, \\ 0 & \text{otherwise,} \end{cases} \text{ with}$$
$$B = \underset{k \in \{1,\ldots,n\}}{\operatorname{argmax}} \left(\{v(s,t) - v(s,t-1) | s(t) = k, s(t-1) = i\}\right)$$

Note that B is the set of the indices of the islands k where individuals coming from i at iteration $t-1$ have obtained the best gain improvements at iteration t.

$$\textbf{Proportional Reward: } R_i^{(t)}(k) = \frac{\Sigma_{s \in K} v(s,t)}{|K|},$$
$$\text{with } K = \{s \in p_k^{(t)} | s(t-1) = i\}$$

Note that K is the set of the individuals of the island k at iteration t that were on island i at iteration $t-1$.

2.3 Migrate Function

We consider four possible migration functions:

- *Elitist migration*: individuals from island i migrate to the island j that has the highest value in line vector, i.e. $\text{argmax}_j\, M^{(t)}(i,j)$. Such migration policy promotes intensification of the search process toward the most efficient islands.
- *Proportional migration*: for each individual s on island i the classic migration process consists in sending this individual according to a probability on line vector $M_i^{(t)}$. Note that M is normalized in order to insure good probability properties.
- *Uniform migration*: individuals from island i migrate to the island j at random uniformly.
- *UCB based migration*: the selection of the next possible migration can be considered as a reinforcement learning problem itself. Therefore, we consider this selection as a multi-armed bandit problem (see [Cesa-Bianchi and Lugosi(2006)] for instance) and we select the next migration using the following formula:
  $UCB(i,j) = M_i^{(t)} + \sqrt{\frac{2\log(\sum_{1\leq j\leq n} nb_i^{(t)}(k))}{nb_i^{(t)}(j)}}$
  where $nb_i^{(t)}(k)$ is the number of individuals that have migrated from i to k at iteration t. Note that we use here indeed the migration matrix as an estimation of the gain that has been obtained by migration from i to other islands. The island with maximal UCB value is selected for next migration.

2.4 Configurations of the DIM: Setting the Policy

While in previous works [Lardeux and Goëffon(2010), Candan et al(2012), Candan et al(2013)], the proposed DIM was using a single configuration for its migration policies (i.e., choice of reward, learn and migrate functions). In this work, we propose to consider more possible components and to generalize the study of different configurations in order to highlight their respective behaviours. In particular we consider an alternative learning mechanism based on QLearning, as well as alternative migration functions.

Instead of considering configurations at the algorithm level, we want to fully benefit from the collaborative model induced by the DIM and we propose to link configurations to individuals, allowing thus the DIM to use simultaneously different policies within the same search process. This is motivated by the fact that the DIM is particularly well suited to the management of collaborative policies. Moreover, we previously observed that different policies leads to different search behaviour and we propose here to check that cooperation may lead to better results that the use of a single policy.

We may first remarks that different migrate functions may be used in the same DIM while its is not possible to use different Learn and reward functions at the same time since they manage the matrices R and M differently, involving incompatible update processes. Based on the previously described components, we propose the following taxonomy in order to define different configurations of the DIM. A policy for a DIM will be described by a tuple

$$(type_l, type_r, type_m)$$

where

- $type_l$ corresponds to the type of learning functions (see Section 2.1), $type_l \in \{C, Q\}$, (C)lassic or (Q)learning
- $type_r$ corresponds to the type of reward functions (see Section 2.2), $type_r \in \{E, P\}$, (E)litist or (P)roportional
- $type_r$ corresponds to composition of the population concerning the migration functions (see Section 2.3) and is a tuple $(Elit, Prop, Unif, UCB)$ with $Elit, Prop, Unif, UCB \in \{0, 1\}$. For instance, $(1, 1, 0, 1)$ means that one third of the individuals use elitist migration, one third use proportional migration, and one third UCB based migration.

Note that we may consider here pure policies, i.e., policies that use only one type of individuals as well as mixed policies were the migration policy is not the same for all individuals. For instance, the policy $(C, E, (0, 1, 0, 0))$ corresponds to the basic dynamic island model that has been previously studied in [Candan et al(2012), Candan et al(2013)].

Note that DIM migration policies have been previously compared to adaptive selection operator policies based on reinforcement learning technique, considering AOS as a multi-armed bandit problem. These previous works [Goëffon et al(2016), Candan et al(2012), Candan et al(2013)] have shown that DIM is an efficient alternative to these approaches. In this paper, since we consider gains that depend from previously visited island, classic AOS approaches, as fully described in [Maturana et al(2012)], are not efficient (as we have checked experimentally). As baseline for comparisons, we consider the following policies:

- A myopic oracle (Oracle) which knows the hidden matrix and selects, at iteration $t + 1$, $\text{argmax}_j H^{(t+1)}(i, j)$ if action a_i has been selected at iteration t.
- A uniform selection (U) that selects uniformly an action at each iteration.

3 Assessing the Efficiency of the DIM Configurations

In this section, we focus on assessing the ability of DIM to dynamically select the most promising islands according to the current state of the search. This corresponds to assess that the migration policies are able to adapt to changes that may occur during the search process, which will be simulated by search scenarios. In order to define these scenarios, we are particularly interested in two main aspects:

- introduce changes in the efficiency of the islands, which corresponds to the fact that different exploration and exploitation stages are usually required along a search process, involving different search operators
- take into account dependencies between islands in order to discover possible cooperative sequences, which corresponds to the fact that, when using different operators into a search algorithm (e.g., metaheuristics or hyperheuristics), they can be combined to achieved an efficient search (for instance due to complementary effects). For instance, after having reached a local optimum with a given search operator, one may use another operator to escape from this local optimum (e.g., using different neighborhoods that correspond to different local move operators).

The purpose of this paper is to assess the efficiency of different dynamic migration policies using evaluation models that consider both above-mentioned aspects. Given an DIM, the efficiency of the policy can be estimated according to the fitness of the individuals that are generated during the solving process. As mentioned before, we will use here generic testing scenarios (i.e., not related to a specific optimization problem) for assessing this efficiency .

Given a DIM and a time horizon T, the efficiency of its migration policy is defined by the value $\Sigma_{t=1}^{T} gain(a_{i(t)}, t)$ obtained by its best individual s^* after T iterations, where $a_{i(t)}$ is the algorithm that has been applied on this individual on island $i(t)$ at iteration t. In this context, an optimal policy corresponds to the best sequence $a_{i(1)}, \cdots, a_{i(T)}$ (that also corresponds to the best visiting sequence of islands $i(1), \cdots, i(T)$). In order to assess the ability to adapt the migration policy to changes, we introduce a hidden gain matrix.

Definition 2 (Hidden Gain Matrix) Given a DIM we define a sequence of matrices $H^{(t)}$, for each iteration t of the algorithm, of size $|\mathcal{A}|^2$ such that $gain(a_k, t) = H^{(t)}(j, k)$ if $a_{i(t-1)} = a_j$ (i.e., gain from j to k).

The gain obtained by action a_k depends on the action a_j that has been previously applied on the considered individual. This general model allows us to take into account dependencies between search operators that should be used sequentially. Of course, H is not known by the DIM. Note that while $H^{(t)}$ encodes gains, $M^{(t)}$ encodes migration probabilities. Nevertheless, the accuracy of the learning process will be easy to assess by comparing the structures of H and M. Note that, for an individual s, $v(s, t) = \Sigma_{k=2}^{t} H^{(k)}(s(k-1), s(k))$. When solving real problems, the gains associated to the application of the search operators are likely to change over time. In order to simulate this behavior in our model, H will be a dynamic in our experiments, with changing values $H^{(t)}$.

Moreover, in order to simulate the fact that search operators can be often stochastic, we will consider two different types of gains, extending Definition 2:

- fixed gain functions, where $gain(a_k, t)$ is always equal to the value $H^{(t)}(j, k)$ as mentioned in Definition 2.
- probabilistic gain functions, where $gain(a_k, t) = 1$ according to the probability defined by $H^{(t)}(j, k)$. For this reason, the values of H will always belong to interval $[0, 1]$.

Therefore a scenario is fully defined by a sequence of hidden matrices $H^{(0)}, H^{(1)}, \cdots, H^{(T)}$.

4 Experiments

The purpose of this section is to evaluate and to analyze the respective performances of the different possible configurations of the DIM, that implement different migration policies, on various possible testing scenarios. According to our evaluation methodology described in Section 3, we consider both static and dynamic hidden gain matrices. The goal is to provide representative scenarios, where interaction between islands may be stable or may evolve during the search process. Some dynamic scenarios are defined by alternating different matrices selected from a set of predefined fixed matrices, at different change frequencies. We use also

Markov inspired scenarios in order to simulate a search process that changes progressively and reach a stable state after a certain time. Remind that our main motivation is to focus on the management of the migration of individuals within the DIM, whose efficiency is assessed by scenarios that take into account the benefit of visiting the islands (and thus being processed by their corresponding algorithms).

4.1 Hidden Gain Matrix

In this section, we present the three different types of scenarios that will be used in our experiments.

4.1.1 Fixed Hidden Matrix

Three basic 10×10 gain matrices A, B and C are defined. These matrix represent typical situations with different types of dependencies between islands. Based on A, B and C, we will define either constant H such that $H^{(t)}$ is always equal to one of these matrix or changing H. The gains are illustrated on Figure 1. The thickness of the arrows from i to j is proportional to the associated gain $H^{(t)}(i, j)$ (as in Definition 1). Of course, even if there is no edge between some islands, migrations are possible but with a null gain $H^{(t)}(i, j)$, which means that no benefit is obtained by using the operator j after the operator i (e.g., if operators have opposite or incompatible effects).

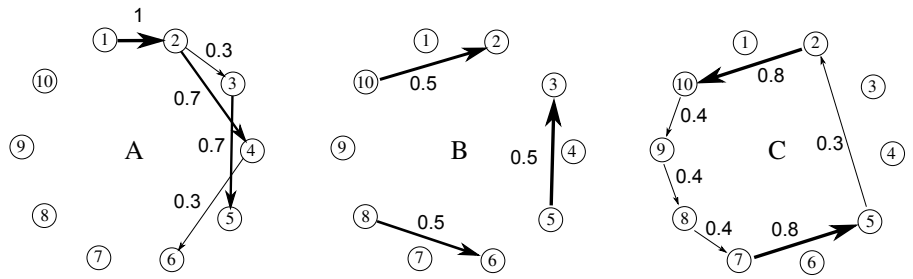

Fig. 1 Representation of possible gains of H.

Matrix A has two possible gain paths of length 3 that provide a total gain of 2. Nevertheless, once reaching the end of these paths, one needs to start again from island 1. Therefore, using one of these paths is suboptimal compared to a cycle $(1-2-4-1)$. This matrix is introduced to check if the policy is able to avoid being trapped into suboptimal paths. Matrix B is introduced to checked that with only 3 efficient paths, the policy is able to identify short gain paths (e.g., performing a cycle $(10-2-10)$). C has a clear gain cycle $(2-10-9-8-7-5-2)$ with no null gain transition. However this cycle is suboptimal compared to the optimal cycle $(2-10-7-5-2)$.

4.1.2 Dynamic Hidden Matrix

In order to define a dynamic hidden matrix it is possible to use a sequence of fixed matrices or to use a matrix based on Markov chain transitions.

– Dynamic hidden matrix based on fixed matrices (Hidden Dyn)
In this model, we use the three previous matrices A,B and C, which are alternated at defined time steps. Given a change frequency FC and a time horizon T the Hidden Dyn matrix can be defined as:

$$H^{(t)} = A \text{ if } (FC > 1 \wedge (t \text{ div } FC) \bmod 3 = 0) \text{ or } (FC = 1 \wedge t \bmod 3 = 1)$$
$$H^{(t)} = B \text{ if } (FC > 1 \wedge (t \text{ div } FC) \bmod 3 = 1) \text{ or } (FC = 1 \wedge t \bmod 3 = 2)$$
$$H^{(t)} = C \text{ if } (FC > 1 \wedge (t \text{ div } FC) \bmod 3 = 2) \text{ or } (FC = 1 \wedge t \bmod 3 = 0)$$

– Dynamic hidden matrix based on Markov chain transitions (Hidden Markov)
We use a Markov chain transitions matrix ([Kemeny and Snell(1960)]) as dynamic hidden matrix.

$$HMarkov^{(0)} = \begin{bmatrix} 0.6 & 0.4 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0.3 & 0.3 & 0 & 0.4 & 0 & 0 & 0 & 0 & 0 \\ 0.2 & 0 & 0.8 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0.6 & 0.4 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0.3 & 0.7 & 0 & 0 & 0 \\ 0 & 0 & 0.2 & 0.5 & 0 & 0 & 0.3 & 0 & 0 & 0 \\ 0 & 0 & 0.2 & 0.8 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0.5 & 0.2 & 0.3 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0.1 & 0.5 & 0.4 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0.1 & 0.4 & 0.5 \end{bmatrix}$$

As for dynamic hidden matrix based on fixed matrices, we also use a frequency of changes FC. Each change corresponds to the update of the hidden matrix, such that $H^{(t)} = H^{(t-1)} \times H^{(0)}$. This Hidden Markov matrix converges to an equilibrium for $t = 50$, and stabilizes after 50 steps.

We consider two possible types of gains: either constant gains, in this case as mentioned in Section 3, $gain(a_k, t) = H^{(t)}(j, k)$ if $a_{i(t-1)} = a_j$ or probabilistic gains, in this case $gain(a_k, t) = 1$ with a probability equal to $H^{(t)}(j, k)$. This allows us to simulate scenarios where the effect of an operator can be subjectied to stochastic phenomena, which is expecially the case in metaheuristics that use generic operators (e.g., uniform crosser or basic mutation operators in evolutionnary algorithms).

4.2 Description of the Test Instances and Parameters of the Configurations

Using the previous matrices, we define the following sets of test scenarios over a time horizon $T = 600$.

– Changing dynamic hidden gain matrix with fixed gains and change frequencies $1, 5, 10, 25, 50, 100$ and 200

- Markov hidden gain matrix with fixed gains and change frequencies of $1, 5, 10, 25, 50, 100$ and 200
- Changing dynamic hidden gain matrix with probabilistic gains and change frequencies of $1, 5, 10, 25, 50, 100$ and 200
- Markov hidden gain matrix with probabilistic gains and changes frequencies of $1, 5, 10, 25, 50, 100$ and 200
- Fixed matrices A,B and C with fixed gains
- Fixed matrices A,B and C with probabilistic gains

We consider thus 34 scenarios for each configuration of the DIM. Each configuration is evaluated on 20 independent runs for each scenario.

Note that some components require parameters (in particular the learning functions, see Section 2.1). They have been obtained using a methodological tuning process with statistical tests. For the basic reinforcement learning function, we have used $\alpha = 0.8, \beta = 0.01$. For QLearning, we have used $\delta = 0.1$ and $\gamma = 0.8$.

4.3 Experimental Results

The experimental results are presented according to the previously described scenarios on the different configurations/policies of the DIM, with regards to our taxonomy (see Section 2.4).

4.3.1 Configurations Based on QLearning

In this paper, we introduce QLearning as a possible learning technique in order to update the migration matrix M as defined in Section 2.1. Therefore, we first study this new configuration of the DIM before going on into more complete comparisons.

According to Section 2.4, this learning approach can be combined with two possible reward functions providing a set of possible configurations of the form $QE*$ or $QP*$, where $*$ is any tuple of $\{0, 1\}^4$. We have run the previously defined testing scenarios and we observed that both configurations provide similar results for each scenarios (according to a T-test). Therefore, in the following, we only consider one type of configuration, which will be called $Q*$, with the different values of $*$ as above.

Concerning the QLearning Q^* configurations, we aim first at evaluating different configurations with only one individual in order to observe the performance on this new learning mechanism in this context of migration policy. To this aim, we run single individual versions of the four following configurations $Q1000, Q0100, Q0010, Q0001$, where the population size in thus 1. Each configuration has been run 20×20 times for each scenario and the best results of each sequence of 20 consecutive runs has been used to compute the mean value. In this case, we compare the mean value with the results obtained with other configurations with 20 individuals. Table 1 shows that the configuration $Q1000$ provides the best results, followed by $Q0100$, which means that for QLearning, as expected, a greedy based strategy is useful for selecting next action.

Once having compared the respective performances of the different "pure" configurations based on QLearning, we aim at studying the influence of using several individuals with different migration policies, instead of a single one, i.e. assessing

Strategies	Fixed Matrices A	B	C
Q1000	165,80	64,33	328,10
Q0100	125,20	50,53	223,42
Q0010	19,31	9,18	20,26
Q0001	57,31	21,65	74,64

Strategies	Hidden Dyn with Different Change Frequencies 1	5	10	25	50	100	200
Q1000	102,74	47,82	41,80	46,56	67,30	125,05	163,10
Q0100	51,60	44,42	43,08	49,99	57,91	85,44	128,74
Q0010	16,80	16,49	15,92	17,25	15,90	16,90	16,63
Q0001	18,70	17,04	19,18	26,96	27,57	39,30	66,65

Strategies	Hidden Markov with Different Change Frequencies 1	5	10	25	50	100	200
Q1000	146,35	158,86	167,15	185,18	198,10	222,17	256,54
Q0100	130,43	131,18	132,93	135,56	143,72	159,30	184,50
Q0010	60,79	60,70	60,47	60,64	60,41	60,36	60,20
Q0001	83,41	83,44	83,48	83,89	84,63	90,63	106,20

Table 1 QLearning Configurations - Scenarios with Fixed gains - One individual

the benefit of using a cooperative management of the migration process and its influence on the knowledge that is collected by a population of individuals. We focus thus on the configuration $Q1000$ and run several experiments with different population sizes. Note that for all experiments, the same computation power is assigned to each configuration, i.e. when the number of individuals is lower then the number of executions used to compute the mean values is increased, as mentioned above. The number of individuals ranges from 1 to 400 for the different scenarios with fixed gains.

The results observed on Table 2 show that the performance increases as the number of individuals increases. This results is interesting, since from the QLearning point of view, it means that using a DIM with several individuals, which share their learned information by means of the migration matrix, may improve the global learning process and, consequently, improves the management of the migrations. It also appears that, when using more than 20 individuals, the improvement slows down. Therefore, we consider that using 20 individuals for next experiments constitutes a good compromise.

In this section, we have restricted the presentation of our experiments to fixed gain scenarios, since results obtained with probabilistic scenarios are rather similar. We have only assess the benefit of using several individuals instead of a single one for improving the performance of configurations that use only one type of individuals. In next section, we will consider general configurations using several type of individuals - and thus several migration policies - simultaneously.

4.3.2 Comparison of the Different Configurations

According to previous remarks of Section 4.3.1, we consider 15 configurations for QLearning (all possible combinations types, ignoring the reward function as already mentioned), 30 configurations for the classic learning mechanism C associ-

nb ind	Fixed Matrices A	B	C
1	165,80	64,33	328,10
4	266,48	71,73	345,24
10	285,13	74,39	353,65
20	292,99	76,37	353,81
50	295,43	77,30	360,81
100	297,00	78,96	365,94
400	297,76	79,95	390,00

nb ind	Hidden Dyn with Different Change Frequencies 1	5	10	25	50	100	200
1	146,35	158,86	167,15	185,18	198,10	222,17	256,54
4	193,23	211,27	215,45	224,48	235,66	258,63	292,62
10	223,17	233,99	236,93	240,27	254,33	284,98	327,23
20	241,84	247,34	250,43	251,38	263,74	296,93	339,57
50	250,05	251,92	252,54	253,96	269,89	310,78	364,17
100	253,14	253,50	254,13	256,44	270,75	310,77	362,44
400	255,65	256,03	256,50	257,92	274,71	327,93	395,20

nb ind	Hidden Markov with Different Change Frequencies 1	5	10	25	50	100	200
1	102,74	47,82	41,80	46,56	67,30	125,05	163,10
4	136,01	90,15	74,12	97,93	153,16	180,51	209,37
10	144,88	105,75	100,74	125,75	176,00	200,90	232,96
20	146,01	134,55	113,53	131,48	190,27	218,73	243,21
50	134,32	135,21	121,66	132,30	195,63	223,57	245,90
100	130,66	147,71	127,72	134,09	196,28	226,87	248,31
400	137,92	149,15	127,07	132,08	175,98	225,61	250,79

Table 2 $Q1000$ - Scenarios with Fixed gains - Different Population Sizes

ated to all possible rewards and combinations of migration functions, the uniform blind migration policy (U) and the myopic oracle (Oracle). Hence, 47 configurations are evaluated on 34 scenarios. In order to provide a more readable analysis of the results, we decide to focus on the 12 best performing policies (first quartile).

In the following tables, for a given column, in case of different performance, the darkest the cells, the better the results. Cells that have the same background colour correspond to statistical similar results according to an ANOVA test that has been performed over 20 executions of each method for each instance. For instance, for the first column all configuration have provided statistically equivalent results. FC is the frequency at which the hidden matrix are changed.

We first analyze the results for Fixed, Hidden Dyn and Hidden Markov matrices with fixed gains.

Table 3 shows that depending on the structure on the hidden matrix that may contain improvement cycles or not, and suboptimal gain cycles, the performance of the policies may differ. In particular, it is interesting to remark that the Oracle may be confused with matrix A, where there exist two possible gain paths (no cycle), one being suboptimal. Q* policies are more efficient to detect such gain cycles and identify the best ones. Since on matrix B no cycle occurs, classic C* policies can be efficient to detect short gain paths or cycle paths, as on matrix C. Of course it is noticeable that Q* policies provides good performance as soon

Fixed Matrices					
A		B		C	
Q1001 Q1101	(44)	Q1000	(40)	Oracle	(46)
Q1000	(43)	Oracle	(39)	CP1001 Q1001	(43)
Q1011 Q1100	(41)	Q1110 Q1100 CP1111	(33)	Q1010 Q1100	(37)
Q1111	(40)	Q1010	(32)	Q1101 Q1011 Q1110 Q1111	(36)
Q1110 Q0101	(38)	Q1011	(31)		
		Q0110	(30)		
Q0100 Q1011	(36)	Q0100 C P 0011	(29)		
				C P1011	(30)
Q1101 Q1111	(35)	Q0111	(27)	CE1101	(28)
		CP 0111	(26)	CE1111	(27)

Table 3 Basic Fixed Matrix (A - B - C)- Fixed Gain - Top 12 Configurations

Frequency of changes (FC)													
1		5		10		25		50		100		200	
Q1100	(45)	Oracle	(46)	Oracle	(46)	Oracle	(46)	Oracle	(46)	Q1000		Q1000	(45)
Q0101	(38)	Q1100	(41)	Q0101	39	Q1010	(45)	Q1110	(45)	Q1100	(42)	Q1100	
Oracle	(37)	Q1101		Q1001	(38)	Q1110		Q1010		Q1101		Q0100	(42)
Q1000 Q1101	(33)	Q0100	(39)	CP1010 Q1101	(33)	Q0110		Q1011		Q0100	(41)	Q1110	
		CP1001 Q0101	(38)			Q0111	(40)	Q1111	(40)	Q1011	(40)	Q1101	(41)
Q0100 Q1111	(31)			Q1000	(32)	Q1011		Q1000		Q1111	(39)	Q1001	(40)
		Q1000	(37)	Q0110 Q1010	(30)	Q1111		Q0100		Oracle Q1110	(38)	Q1111	
Q1001		CP0001	(35)			Q1000	(36)	Q0110	(36)			Q1011	(39)
CP1010	(30)	Q1001	(34)	Q1100		Q0101	(33)	Q1100	(33)	Q1010	(37)	Q0110	
CP1011		Q1110	(33)	Q1011	(29)	Q1100		Q1101		CP0111	(33)	Q0101	(35)
CP1111		Q1111		Q1110		Q0011	(31)	CP0111	(31)	Q0111	(32)	Q0111	

Table 4 Hidden Dyn - Fixed Reward - Top 12 Configurations

as they use either elitist or proportional migration to promote greedy choices. For instance, let us consider Q1001, which mixes elitist choice and UCB, using greedy choice and a slight exploration process. This policy is indeed known as a good strategy when using QLearning.

We consider now the dynamic hidden matrices based on alternating between A, B and C, again with fixed gains. These matrices will be called Hidden Dyn in the remaining of this paper. We may observe on Table 5 that, if the changes are too frequent then, since the learning process is unable to detect any stable knowledge, most of the policies are statistically equivalent. Nevertheless, note that, if we considered more than the 12 best policies it will not be still the case. Since the Oracle was rather efficient for the fixed matrices case, it still have good properties. In most of these experiments, classic learning process (either CP or CE) exhibit good results. It is also interesting to mention that, in these experiments, configurations Q^* that use several types of individuals obtain better results than configuration using only one type of individual.

On Table **??**, results are more contrasted between the different policies. When the frequency of changes if high (i.e., lowest values), then the matrix stabilizes sooner (remind that the matrix is stable after 50 changes). Therefore, methods that were efficient for fixed matrices are also efficient here (e.g., CP* or Q*). It is interesting to see that when the changes are less frequent, due to the struc-

Frequency of changes (FC)														
1		5		10		25		50		100		200		
Oracle	(46)	Oracle	(46)	Oracle	(46)	Oracle	(46)	Oracle	(46)	Oracle	(46)	Oracle	(46)	
Q1101 CE1001 Q1001 CP1001	(38)	CP1001 Q1001 CE1001	(37)	Q1001	(40)	Q1001	(39)	Q1101	(34)	Q1101	(36)	Q1110 CP1011	(29)	
				CE1001	(39)	CP1001 CE1001	(38)	Q1100 Q1001	(33)	Q1100	(35)			
				CP1001 Q1101	(36)					Q1110	(34)	Q1101	(28)	
		Q1101	(36)			Q1101	(35)	Q1010		Q1011	(33)	Q1100		
CE1011 Q1011	(35)	Q1011	(35)	CE1011 Q1011	(33)	Q1011	(34)	Q1011		Q1010		Q1001		
		CE1011				CE1011		CP1001	(31)	Q1001	(32)	Q1011	(27)	
CE1101	(34)	Q1100	(32)	Q1100	(32)	CE1101	(30)	Q1110		Q1111		Q1111		
Q1100 Q1110 Q1111	(32)	CE1101 Q1111		CE1101 Q1111 Q1110	(30)	Q1100 Q1111		Q1111		CE1111	(26)	CP1111		
		CE1111	(30)					CE1101	(30)	CP1011		Q1010	(26)	
						Q1110	(29)	Q1000	(29)	CE1011	(25)	Q0011		
Q1010	(31)	Q1110		Q1010	(29)	Q1010	(28)	CE1001	(27)	CE0011		CE0111	(25)	

Table 5 Hidden Markov - Fixed Gains - Top 12 Configurations

Fixed Matrices					
A		B		C	
Q1000 Q1001	(41)	Oracle Q1000	(39)	Oracle	(46)
				CP1001	(37)
Q1011 Q1100 Q1101 Q1111	(39)	Q1110	(35)	Q1001	(36)
		Q1100	(34)	Q1010 Q1100	(35)
		Q0100 Q0110 Q1010	(30)	Q1101	(33)
Q1110	(37)			Q1011 Q1110 Q1111	(32)
Q0100 Q0101 Q1010	(36)	CP0011 CP0100 CP1111	(28)		
				CP1111	(28)
Q0110 Q0111	(35)	Q1011		CE1111	(26)
		CP0111	(24)	CE0011	(25)

Table 6 Basic Fixed Matrix (A - B - C)- Probabilistic Gain - Top 12 Configurations

ture of the initial matrix, which does not have specific gain paths or cycles, Q^* are really efficient and may become even better than the Oracle. This phenomena is certainly related to the successive states reached by the matrix during its stabilization process.

The following Tables 6,7 and 8 show the results obtained with probabilistic gains. These results are somehow quite similar for many policies. Nevertheless, when using probabilistic gains, it seems that Q^* policies have better properties than $C*$ policies.

In order to give a clearer overview of our experiments, we present in Figure 2 and Figure 3, a global comparison of the methods that have been ranked in the top 12 for all previous experiments, distinguishing between fixed and probabilistic gains. We may draw the following conclusions from these experiments:

– The newly introduced Q^* policies provide good results allowing to reach even better adaptive control than a myopic Oracle. In presence of fixed gains, classic configurations C^* are still interesting. Note that all the best configurations use individuals that migrate according to an elitist choice (i.e. greedy choice of the best possible next decision). When this migration component is not present, it can be compensated by a proportional migration that allows the individuals to benefit from a less elitist migration process. The UCB migration policy mixing

Frequency of changes (FC)														
1		5		10		25		50		100		200		
Q0101	(38)	Oracle	(46)	Oracle	(46)	Oracle	(46)	Oracle	(46)	Q1000	(42)	Q1000	(45)	
Oracle		CP1001	(42)	Q0101	(40)	Q1010	(44)	Q1010		Q1100	(41)	Q0100	(43)	
Q0100	(37)	Q1100	(41)	Q1001		Q1110		Q1011	(42)	Q0100		Q1100		
Q1000		Q0100	(40)	CP1010		Q0110		Q1110		Q1101	(40)	Q1110	(42)	
Q1001	(36)	Q1101	(38)	Q1000	(32)	Q0111	(40)	Q1111		Q1110		Q1101	(41)	
Q1101		CP1010		Q1101		Q1011		Q1000	(38)	Q1111		Q1001	(40)	
Q1100	(34)	Q0101	(36)	Q0100	(31)	Q1111		Q0100		Oracle	(38)	Q1111		
CP1001		Q1001		Q1100		Q1000	(38)	Q0110	(37)	Q1011		Q1010	(38)	
Q1110	(30)	CP0001	(35)	Q0110		Q1101	(36)	Q0111		Q1010	(36)	Q1011		
Q1111		CP1011	(34)	Q1010	(30)	Q0011		Q1100		CP0011		Q0110	(37)	
CP1101	(29)	Q1000		Q1011		Q0100	(33)	Q1101	(35)	Q0101	(32)	Q0101	(35)	
CP1010	(27)	CP0011	(32)	Q1110		Q0101		CP0111	(32)	Q0111		Q0111		

Table 7 Hidden Dyn - Probabilistic Gains - Top 12 Configurations

Frequency of changes (FC)														
1		5		10		25		50		100		200		
Q1101	(39)	Oracle	(37)	Oracle	(39)	CE1011		Oracle	(43)	Oracle	(45)	Oracle	(46)	
Oracle	(38)	CE1011	(36)	CE1011		Oracle	(35)	Q1100	(40)	Q1100	(42)	Q1100	(38)	
CE1101	(36)	Q1101	(35)	CE1101	(34)	Q1101		Q1000		Q1010	(38)	Q1001	(37)	
CE1001		CP1001	(34)	CP1001		CP1001	(34)	Q1010	(33)	Q1101		Q1010		
CE1011	(34)	Q1100		Q1101		CE1001		Q1101		Q1001		Q1101	(34)	
Q1001		Q1011		CE1001		CE1101		Q1011	(32)	Q1011	(35)	Q1011	(32)	
Q1011	(32)	Q1010	(33)	Q1001	(33)	Q1001	(33)	CE1011	(29)	Q1110		Q1110	(31)	
CP1001		Q1001		Q1010		Q1011		CP1001		Q1111	(28)	Q1111	(30)	
CP1101	(31)	CE1101		Q1011		Q1100		Q1001	(27)	CE1011	(26)	Q0011	(24)	
Q1010		CE1001		Q1100	(32)	Q1010	(32)	Q1110		Q1000	(25)	CP0011	(23)	
Q1100		Q1111	(30)	CP1011	(31)	CP1011	(30)	CE1101	(26)	CE0011	(21)	CE0111	(18)	
CP1011	(30)	CP1101		CP1101		Q1000		CE0011	(23)	CE0111	(20)	Q1000		

Table 8 Hidden Markov - Probabilistic Gains - Top 12 Configurations

exploitation (greedy choice) and exploration (random choice) seems to be also an interesting diversification mechanism, if combined with more exploitative strategies, while not being efficient if used alone (as seen in Section 4.3.1).

– Using simultaneously several types of individuals is really beneficial. In fact, some individuals play the role of explorers: even if they obtain poor gains, they contribute to the global learning process, by updating the reward matrix and, consequently, the migration matrix. Individuals that focus on elitism can be considered as champions, whose role is to collect the best possible scores, according to the current knowledge with regards to estimated expected gains. Therefore, instead of achieving the classic trade-off between exploration and exploitation at the algorithm level, it is performed by means of the cooperation between individuals. In particular, let us note that C^* configurations using several types of individuals achieve better result than the classic initial DIM (i.e., CE 0 1 0 0). In general, good results can be obtained when individuals using an elitist migration policy are associated to individuals using a policy that permits more or less diversification (from uniformly random choice to proportional choice).
– Concerning QLearning based configurations, remark that, of course, this learning process is really interesting for controlling migrations in DIM, which had not been experimented before. Moreover, it is interesting to remark that using

simultaneously different type of individuals provide an increase of efficiency of the learning process compare to a pure strategy (e.g., Q 1 0 0 0). In particular, it should be noticed that the configuration Q 1 1 0 0, that mixes elitist and proportional migration, always belong to the 12 best configurations for the 34 scenarios.

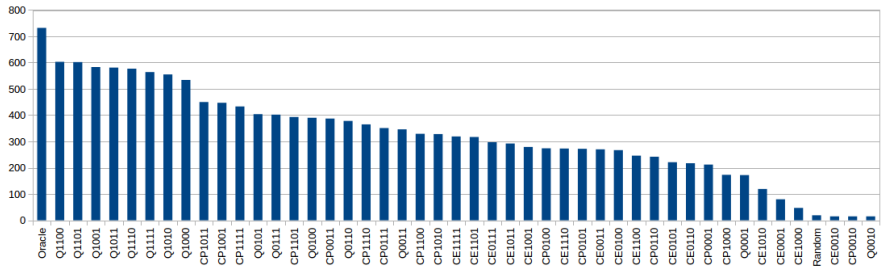

Fig. 2 Ranking of configurations for fixed gains

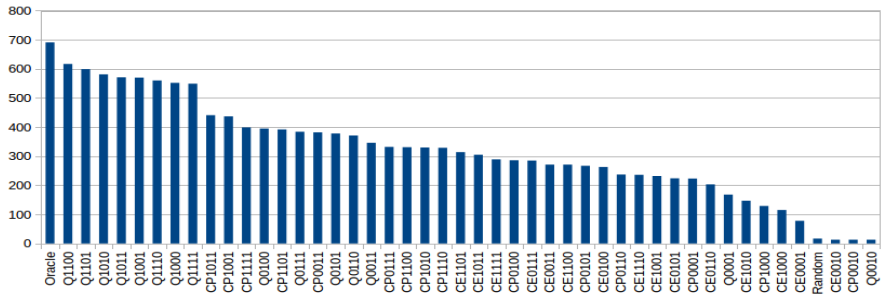

Fig. 3 Ranking of configurations for probabilistic gains

5 Conclusions

In this work, we propose to study the management of migrations in islands models. In Dynamic Island Models, the migration mechanism is updated during the search in order to better adapt to the current state of the search process and therefore to improve its overall performance. DIM are using different sub-populations that are processed by different search operators or algorithms. These operators may have complementary or opposite effects during the search. Hence, a key feature for insuring a good trade-off between exploration and exploitation of the search space is to manage the distribution of the individuals on these sub-populations, which are considered to be located on islands. In order to provide a wide range of possible search situations, we propose to use search scenarios that take into account possible interactions between the operators as well as the dynamic evolution of

their efficiency during the search. Based on a generic DIM, we propose a rather complete set of possible components that allows us to define different configuration of the DIM, involving different management of the individuals and of the migration processes. Instead of having one dynamic migration policy that is applied to the whole search process, as it has already been studied, we propose here to associate this policy to each individual, which allows us to use simultaneously different migration policies within the same DIM. Therefore, these individuals cooperate and share their information into a common migration matrix. We also consider in this work, a QLearning approach in order to learn what are the best migration choices for individuals at a given state of the search. We evaluate these possible configurations of the DIM on a set of scenarios. Our results highlights that:

- using QLearning is interesting for managing migrations in dynamic island models,
- using a population of individuals that cooperate by exchanging informations improves the QLearning performance for scheduling the operators that must be used along the search compared to a single learning process,
- considering simultaneously different type of individuals that use different migration policies is also beneficial and leads to good results compared to previously proposed dynamic migration approaches in the context of our search scenarios.

References

[Araujo et al(2009)] Araujo L, Guervós JJM, Mora A, Cotta C (2009) Genotypic differences and migration policies in an island model. In: GECCO, pp 1331–1338

[Candan et al(2012)] Candan C, Goëffon A, Lardeux F, Saubion F (2012) A dynamic island model for adaptive operator selection. In: Genetic and Evolutionary Computation Conference (GECCO'12), pp 1253–1260

[Candan et al(2013)] Candan C, Goëffon A, Lardeux F, Saubion F (2013) Non stationary operator selection with island models. In: Genetic and Evolutionary Computation Conference, GECCO '13, Amsterdam, The Netherlands, July 6-10, 2013, pp 1509–1516

[Cesa-Bianchi and Lugosi(2006)] Cesa-Bianchi N, Lugosi G (2006) Prediction, Learning, and Games. Cambridge University Press, New York, NY, USA

[Da Costa et al(2008)Da Costa] Da Costa L, Fialho A, Schoenauer M, Sebag M (2008) Adaptive operator selection with dynamic multi-armed bandits. In: M Keijzer et al (ed) Proc. GECCO'08, ACM Press, pp 913–920

[Goëffon et al(2016)] Goëffon A, Lardeux F, Saubion F (2016) Simulating non-stationary operators in search algorithms. Appl Soft Comput 38:257–268, DOI 10.1016/j.asoc.2015.09.024, URL `http://dx.doi.org/10.1016/j.asoc.2015.09.024`

[Gustafson and Burke(2006)] Gustafson S, Burke EK (2006) The speciating island model: An alternative parallel evolutionary algorithm. J Parallel Distrib Comput 66(8):1025–1036

[Kemeny and Snell(1960)] Kemeny JG, Snell JL (1960) Finite markov chains, vol 356. van Nostrand Princeton, NJ

[Lardeux and Goëffon(2010)] Lardeux F, Goëffon A (2010) A dynamic island-based genetic algorithms framework. In: SEAL, pp 156–165

[Lässig and Sudholt(2010)] Lässig J, Sudholt D (2010) The benefit of migration in parallel evolutionary algorithms. In: Genetic and Evolutionary Computation Conference, GECCO 2010, ACM, pp 1105–1112

[Lässig and Sudholt(2013)] Lässig J, Sudholt D (2013) Design and analysis of migration in parallel evolutionary algorithms. Soft Comput 17(7):1121–1144

[Luque and Alba(2010)] Luque G, Alba E (2010) Selection pressure and takeover time of distributed evolutionary algorithms. In: Pelikan M, Branke J (eds) Genetic and Evolutionary Computation Conference, GECCO 2010, ACM, pp 1083–1088

[Luque and Alba(2011)] Luque G, Alba E (2011) Parallel Genetic Algorithms. Theory and Real World Applications,. Springer-Verlag

[Mambrini and Sudholt(2014)] Mambrini A, Sudholt D (2014) Design and analysis of adaptive migration intervals in parallel evolutionary algorithms. In: Genetic and Evolutionary Computation Conference, GECCO '14, Vancouver, BC, Canada, July 12-16, 2014, pp 1047–1054

[Maturana et al(2012)] Maturana J, Fialho Á, Saubion F, Schoenauer M, Lardeux F, Sebag M (2012) Autonomous Search, Springer, chap Adaptive Operator Selection and Management in Evolutionary Algorithms, pp 161–190

[Melab et al(2005)] Melab N, Mezmaz MS, Talbi EG (2005) Parallel hybrid multi-objective island model in peer-to-peer environment. In: 19th International Parallel and Distributed Processing Symposium (IPDPS 2005), IEEE Computer Society

[Rucinski et al(2010)] Rucinski M, Izzo D, Biscani F (2010) On the impact of the migration topology on the island model. CoRR abs/1004.4541

[Rudolph(2000)] Rudolph G (2000) Takeover times and probabilities of non-generational selection rules. In: roceedings of the Genetic and Evolutionary Computation Conference (GECCO '00, Morgan Kaufmann, pp 903–

[Skolicki(2007)] Skolicki Z (2007) An analysis of island models in evolutionary computation. PhD thesis, George Mason University

[Skolicki and Jong(2005)] Skolicki Z, Jong KD (2005) The influence of migration sizes and intervals on island models. In: GECCO, pp 1295–1302

[Sutton and Barto(1998)] Sutton R, Barto A (1998) Reinforcement Learning: An Introduction. MIT Press

[Thierens(2005)] Thierens D (2005) An adaptive pursuit strategy for allocating operator probabilities. In: Genetic and Evolutionary Computation Conference, GECCO, ACM, pp 1539–1546

[Whitley et al(1998)] Whitley D, Rana S, Heckendorn R (1998) The island model genetic algorithm: On separability, population size and convergence. Journal of Computing and Information Technology 7:33–47