

A Life in the Galapagos: migration effects on neuro-controller design

Christos Ampatzis, Dario Izzo, Marek Ruciński, and Francesco Biscani

Advanced Concepts Team,
Keplerlaan 1 - 2201 AZ Noordwijk - The Netherlands
{Christos.Ampatzis,Dario.Izzo,Marek.Rucinski,Francesco.Biscani}@esa.int
<http://www.esa.int/act>

Abstract. The parallelization of evolutionary computation tasks using a coarse-grained approach can be efficiently achieved using the island migration model. Strongly influenced by the theory of punctuated equilibria, such a scheme guarantees an efficient exchange of genetic material between niches, not only accelerating but also improving the evolutionary process. We study the island model computational paradigm in relation to the evolutionary robotics methodology. We let populations of robots evolve in different islands of an archipelago and exchange individuals along allowed migration paths. We show, for the test-case selected, how the exchange of genetic material coming from different islands improves the overall design efficiency and speed, effectively taking advantage of a parallel computing environment to improve the methodology of evolutionary robotics, often criticized for its computational cost.

Key words: Stochastic Optimisation, Evolutionary Robotics, Island Model, Parallel Computing

1 Introduction

The theory of punctuated equilibria is a theory in evolutionary biology which essentially states that evolution proceeds with rapid changes after long periods of stasis [9]. The theory inspired Cohoon et al. [4] in formulating the island migration model, a coarse-grained parallelisation approach to global optimisation (GO). More specifically, the authors originally focused on the parallelization of genetic algorithms. Thus, the underlying mechanisms of this stochastic global optimization technique inspired by Darwinian evolution, namely recombination, mutation and selection, are complemented by migration. Initially isolated populations, while evolving in parallel, exchange individuals at a certain rate, thus interacting with each other. As a result, not only do GA populations evolve faster, but their performance is also improved. The island model paradigm has also been applied to the parallelisation of other evolutionary algorithms, as differential evolution (DE [11]) and applied on difficult and high dimensional global optimisation problems [5].

Evolutionary Robotics In the past years the application of artificial evolution to the optimisation of neuro-controllers for autonomous robots has been gaining a lot of momentum. The approach called Evolutionary Robotics (ER) is, essentially, a methodological tool to automate the design of robots' controllers [10]. It is typically based on the use of artificial evolution to find sets of parameters for artificial neural networks that guide the robots to the accomplishment of their task. With respect to other design methods, ER has the theoretical advantage that it does not require the designer to make strong assumptions concerning what behavioural and communication mechanisms are needed by the robots.

However, to-date, the complexity of the tasks solved by agents controlled by evolved neuro-controllers is lower than the complexity achieved by other methods using hand-coded controllers driven by expert knowledge. Also, even if automatic techniques could in principle reduce the human effort required to design controllers, this is usually not the case [8]. In other words, the complexity achieved by automatic approaches seems incommensurate with the effort expended in setting up and configuring the evolutionary system. Therefore, despite the theoretical advantages of automating the design problem for autonomous agents, the robotics control community cannot yet claim to having reaped its benefits. More effort should be put by researchers in reducing the computation time required until a solution to a problem at hand is obtained with these techniques, and at the same time in creating a framework that can generate more complex solutions without a significant effort overhead on the side of the experimenter. Various approaches in the literature have explored the possibility of enhancing the efficiency of automatic design tools, such as incremental evolution and symbiotic evolution, but the focus of such methods is not on creating a generic and simple design framework and certainly not on the algorithmic side.

Evolution in robotic islands Typically, ER researchers launch a number of independent evolutionary runs, each differentiated by a different initial random seed. Subsequently, for each run, the best individual of the last generation, or alternatively the best individual encountered throughout evolution is identified and analysed (post-evaluated). Thus, some of the runs end up successful, and others not. By successful run we mean a run where the fitness obtained is the maximum, or one where a neuro-controller is produced that is able to drive the robots to the accomplishment of their task. Every evolutionary run is characterised by the prevalence of a certain genetic material that gets spread through recombination and survives through selection. Variation of genetic material within the population comes through random mutation only.

This paper introduces the idea of adding the island model paradigm to the ER arsenal, thus using the island model to evolve artificial neural networks controlling robots. In this case, the experimenter will still be launching multi-start evolutionary runs, only this time, the dynamics of the stochastic search in a single run will not be fully determined by the initial random seed, but also co-determined by the migration among runs and information exchange. Populations will be invaded with genetic material shaped elsewhere. Migrants might represent solutions radically different than the solutions developed in a given population.

The analogies between the biological observations and the effect of migration in GO still hold when the application is the ER methodology. In particular, exploration arises from migration and exploitation from isolated evolution in islands. Finally, while in GO the island model introduces new local minima and optimisers to a pool of solutions, in ER migration represents introducing new ways of solving the task in a pool of existing solutions. This is because a solution is, apart from a fitness number, the behavioural capabilities of autonomous, simulated or real agents that are evaluated in a given environment and with respect to a given task. This paves the way to addressing more complex ER scenarios, including evolving populations for different tasks in different islands, and then letting migration perform the mixing of genetic material and behavioural capabilities.

2 Simulating Evolution in an Archipelago

The simulation environment we used for this study has been developed entirely by the European Space Agency’s Advanced Concepts Team and is freely available as an open source project named Parallel Global Multiobjective Optimiser (PaGMO). The code, entirely written in C++ and tested on Windows XP, Ubuntu, and Leopard OS, implements the asynchronous island model paradigm for generic optimisation purposes. It defines Individuals, Populations, Islands and Archipelagi as C++ objects, providing an easy-to-use computational environment where to simulate the concurrent evolution of populations. PaGMO objects can also be instantiated directly from Python and each island can be assigned the same or a different algorithm to evolve its population. The evolution in each island is assigned to a different thread of the underlying operating system so that parallelisation is obtained when multiple processors are available. Communication (migration) between threads (islands) occur in an asynchronous fashion. An *archipelago* is defined by the islands it contains, but also by their geographical location which determines the possible migration routes an individual can take. We describe such migration routes as a graph where the nodes represent different islands and the edges represent the possible connections between islands. We refer to this graph as the migration topology. An *island* is defined by a population of individuals, by an evolutionary strategy (or a global optimisation algorithm) and by a task (or a global optimisation problem). A *population* is defined by the individuals it contains and, finally, an *individual* is defined by its chromosome. Such a generic framework facilitates studies on the island model impact on optimisation algorithms in general and population based meta-heuristics in particular.

Evolutionary robotics and stochastic global optimisation Here we briefly describe ER problems and their mathematical structure, as encountered in the literature, trying to highlight those elements that make of ER problems a very special case of optimisation problems. We indicate with $\mathcal{N}(\mathbf{x})$ a generic robot controller tasked to translate robot sensory inputs into actuation commands. The controller is defined by a number of parameters we indicate with $\mathbf{x} \in R^N$. The

ultimate aim is choosing the best \mathbf{x} . To this aim, a function $f(\mathbf{x}, \mathbf{s})$ is constructed that maps the chromosome to a fitness value identifying how well the robot is able to solve a given task using the controller $\mathcal{N}(\mathbf{x})$ during a particular experiment defined by \mathbf{s} . Typically, \mathbf{s} includes the robots' initial conditions/configuration, but also variables defining uncertain environment characteristics and sensory noise; \mathbf{s} is a stochastic variable with known distribution. The objective function assigns a value to the decisions taken by the controller \mathcal{N} both in terms of achieving or not the task required and in terms of their optimality. The generic ER problem can thus be written as:

$$\begin{aligned} & \text{find: } \mathbf{x} \in R^n \\ & \text{to maximize: } J(\mathbf{x}) = E[f(\mathbf{x}, \mathbf{s})] \end{aligned} \quad (1)$$

where $E[\cdot]$ indicates the expected value of its argument. One wants to find a controller \mathcal{N} allowing the robot to solve an assigned task for all (or almost all) possible realisations of the experimental set-ups \mathbf{s} and that is optimal with respect to the expected value of the defined objective function. The ER problem described above shares a lot of similarities with problems studied in stochastic programming and indeed some of the techniques used there are also used by ER researchers. The sample average approximation (SAA) method [6], for example, may be used to approximate the objective function so that $J(\mathbf{x}) \approx \frac{1}{n} \sum_{j=1}^n f(\mathbf{x}, \mathbf{s}_j)$ where a sample $\mathcal{S} = \mathbf{s}_1 \dots \mathbf{s}_n$ is considered, where each \mathbf{s}_j is an independent identically distributed random variable (expected value is the average) and has the same probability distribution as \mathbf{s} . The SAA is not an algorithm, but just an approximation method and one has still to solve the resulting problem (no longer stochastic) with an appropriate numerical procedure. Typically in ER an evolutionary algorithm is used to solve the SAA approximated problem (genetic algorithm or evolutionary strategies). This is not a necessary choice and global optimisation heuristics based on paradigms other than the darwinian evolution may, in principle, be able to solve the problem. Once the approximated problem is solved, its solution $\hat{\mathbf{x}}$ is a candidate solution to the original stochastic programming problem and its quality as such needs to be assessed. This is done by performing one evaluation of $J(\mathbf{x})$ using a different and typically larger sample \mathcal{S}' (called in ER works post-evaluation of a solution).

The evolutionary strategy adopted In this paper we use differential evolution [11] as a global optimisation heuristic to solve the ER task considered. In PaGMO terms this means that each island will be assigned a population which will evolve, before each migration happens, via an instance of DE for N generations. The objective function for the evolution is the SAA approximation of the chosen task. The sample \mathbf{s} (and thus the objective function) is changed every $m = N$ generations to avoid the optimiser to take advantage of particularities of one particular sample. One iteration of DE is made of three phases: mutation, crossover and selection. In the mutation phase, and for every individual in the population of size NP, a so-called mutant individual is formed. To create such a mutant we use the DE2 variant of the algorithm (see [11]):

$$v_{i,G+1} = x_{r_1,G} + F \cdot (x_{r_2,G} - x_{r_3,G});$$

where G is the generation number, r_{1-3} are 3 different, randomly selected individuals, and $F > 0$ is the constant amplification factor. Exponential crossover is then performed between the mutant and the original individual with a step probability CR. In the experiments, we used $F = 0.8$, $CR = 0.8$, $NP = 30$ and sample size $n = 40$, with the sample changing every $m = 50$ generations .

Migration and topology Setting up an experiment which involves optimisation with the island model requires making cross-related choices about the following parameters [3]: (1) the *number of islands*, (2) the *archipelago topology* which defines feasible migration paths between islands, (3) the *migration rate* which tells how many individuals migrate at a time, (4) the *migration frequency* which determines how often migration occurs, and (5) the *migration algorithm*. The latter includes defining if the migration is synchronous (all islands are forced to migrate at the same time during which no optimisation is performed) or asynchronous (every island performs migration as soon as it is ready to do so), which individuals from the population migrate, how they are distributed among neighbour islands, and which (if any) individuals in the destination population are replaced by migrants and under what conditions. All these choices have an impact on the overall robustness of the archipelago, however fine-tuning these parameters is out of scope of this paper. That is why we decided to make choices commonly found in the existing literature (see for instance [7, 12]), as this should be sufficient to demonstrate that the island model can enhance the potential of ER as an optimisation technique.

In our experiments, the archipelago consists of 10 islands connected with a one-way ring topology (see figure 1a). Of course, there is an infinite number of other possible choices which could include lattices, hypercubes, and even more complex networks, like small-world Barabasi-Albert model networks (see figure 1b) for large numbers of islands. We decided to use an asynchronous migration algorithm as it is more suitable to model modern large-scale distributed computational environments in which synchronisation of all nodes is either impossible or extremely expensive. Every island migrates its best individual every 50 generations of DE with probability 0.4. On arrival, the migrating individual replaces a random individual in the destination population if it has a better fitness value. The following pseudo-algorithm defines the computations performed by each island (thread):

- 1: create a starting population \mathcal{P}_i of dimension NP
- 2: while !*stopping-criteria*
- 3: generate a sample s and the SAA approximated problem
- 3: evolve \mathcal{P}_i for N generations using DE
- 5: migrate the best individual to the neighbour island
- 6: insert the migrating individual from the neighbour island into \mathcal{P}_i (if available)

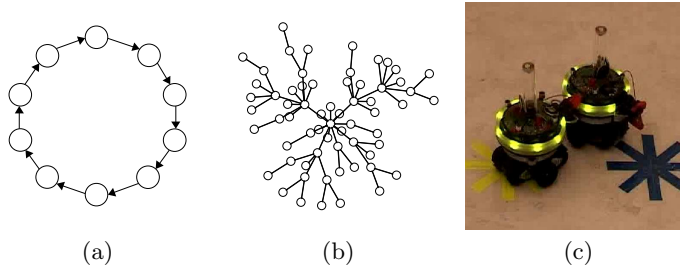


Fig. 1. (a) Example of a ring topology with 10 nodes; (b) a Barabasi-Albert model topology with 64 nodes; (c) a snapshot of the robot and the task.

3 Test bed: evolving self-assembling robots

The task we consider as a test-bed is described in detail in [1, 13] and it can be roughly summarised as follows. Two robots are initialised at a given distance between a lower and an upper bound with given initial orientations; the robots do not have means for explicit communication; they are only able to sense their distance to their group mate. The agents should coordinate their movements in order to allocate roles and connect to each other via the gripping mechanism (see figure 1c). The controllers are evolved in a simulation environment which models some of the hardware characteristics of the *s-bot*, a small mobile autonomous robot with self-assembling capabilities. The network $\mathcal{N}(\mathbf{x})$ used to control the robots is a Continuous Time Recurrent Neural Network (CTRNN [2]), with a fixed feed-forward architecture comprising an arrangement of 11 input neurons, 10 hidden neurons and 3 output neurons. Decay constants (τ), connection weights between two neurons (ω) and bias terms (β) are genetically specified parameters. Each CTRNN is thus encoded by a vector \mathbf{x} comprising 263 real values. A random population of CTRNNs is generated by initialising each component of \mathbf{x} to values randomly chosen from a uniform distribution in $[-10, 10]$.

The control is homogeneous since the robots share the same dynamical controller (i.e., the CTRNN). We consider the problem in the form of Eq.(1) where the function $\mathbf{f}(\mathbf{x}, \mathbf{s})$ assesses the ability of the two robots to approach each other and physically assemble through the gripper, without interfering or dictating the role allocation strategy the robots should use. The experimental conditions \mathbf{s} include the robots' initial conditions (i.e. orientations and mutual distance) and the sensory noise. The value of $\mathbf{f}(\mathbf{x}, \mathbf{s})$ can vary between 0 and 100. The maximum value corresponds to collision-free successful robot self-assembly. The samples \mathcal{S} used to obtain the SAA approximation to the problem contain 40 experimental conditions \mathbf{s}_j generated consistently with [1, 13].

4 Experiment and results

The problem defined above has been solved in [1, 13], where an evolutionary strategy ((μ, λ) -ES) was also employed, and the resulting neuro-controller was

successfully transferred to real robots. In order to study the effects of the island model and migration in particular, we create an archipelago of ten islands containing different populations of 30 CTRNN. We allow each population to evolve with DE in each island for 10,000 generations, i) with no migration (this is equivalent to a set of 10 independent sequential runs of DE), and ii) allowing migration using a ring topology as described in section 2.

The comparison is done with respect to the maximum fitness achieved during evolution and in terms of the number of generations elapsed up to that point. The maximum fitness is defined as the best fitness across all individuals and across all the islands. In figure 2c we can see the results for the two experiments. When no migration is allowed (sequential algorithm), DE manages to reach around 80% of the maximum fitness. When migration takes place, the island model implementation of DE does reach the value of 100.¹

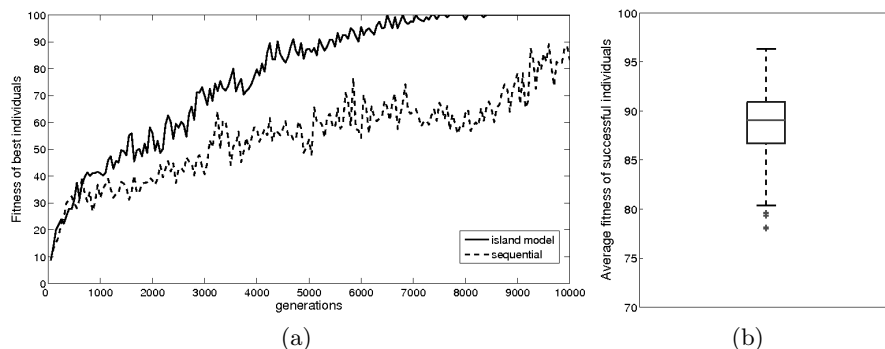


Fig. 2. (a) Best fitness across generations for sequential and island model setups; (b) Boxplot of the average fitness of all successful individuals after post-evaluation.

The quality of the candidate solutions found $\hat{\mathbf{x}}$ needs to be assessed once the evolution has completed as explained previously. This is needed to exclude the possibility that the solutions discovered have taken advantage of favourable conditions during evolution. We post-evaluate successful individuals (with fitness 100) on a set \mathcal{S}' of 1,000 experimental conditions, all differing with respect to the random noise applied on sensors/actuators, initial robot positions and orientations. In figure 2b we show a boxplot of the average fitness obtained by all successful individuals in post-evaluations. The best scoring individual in these tests achieved an average fitness score over 96, which confirms that the quality of the solution obtained with the presented framework is very high across almost all possible realisations of the experimental set-up \mathbf{s} . Still, as we can see from the boxplot, this is not the case with all individuals that scored maximum during evolution, as some achieve considerably lower fitness values in post-evaluations.

¹ Notice that for the sequential case we have prolonged the evolution for 5,000 generations more but without achieving the maximum—results not shown on the graph.

5 Conclusion

In this article, we have introduced the island model paradigm to the evolutionary robotics methodology. We have presented one test case where migration of genetic material across evolving populations not only accelerates evolution but also creates better individuals, managing to obtain solutions to the problem at hand. Future work will include a more thorough understanding of how the parallel version improves the algorithm performance, the effect of migration on the population diversity, the generation and analysis of a more statistically sound sample of experiments, as well as the exploitation of the island model for solving more complex tasks with ER.

References

1. C. Ampatzis, E. Tuci, V. Trianni, A. L. Christensen, and M. Dorigo. Evolving self-assembly in autonomous homogeneous robots: experiments with two physical robots. *Artificial Life*, 15(4):1–20, 2009.
2. R. D. Beer and J. C. Gallagher. Evolving dynamical neural networks for adaptive behavior. *Adaptive Behavior*, 1:91–122, 1992.
3. Erick Cantu-Paz. *Efficient and Accurate Parallel Genetic Algorithms*. Kluwer Academic Publishers, Norwell, MA, USA, 2000.
4. J. Cohoon, S. Hegde, W. Martin, and D. Richards. Punctuated equilibria: a parallel genetic algorithm. In *Proceedings of the Second International Conference on Genetic Algorithms and their application*, pages 148–154, Hillsdale, NJ, USA, 1987. L. Erlbaum Associates Inc.
5. D. Izzo, M. Ruciński, and C. Ampatzis. Parallel global optimisation meta-heuristics using an asynchronous island-model. In *Proceedings of CEC2009*, 2009.
6. A. J. Kleywegt, A. Shapiro, and T. Homem de Mello. The sample average approximation method for stochastic discrete optimization. *SIAM Journal on Optimization*, 12(2):479–502, 2002.
7. W. N. Martin, J. Lienig, and J. P. Cohoon. *Island (migration) models: evolutionary algorithms based on punctuated equilibria*, chapter C6.3:1-C6.3:16. Institute of Physics Publishing, Bristol, UK, 1997.
8. M. Mataric and D. Cliff. Challenges in evolving controllers for physical robots. *Robotics and Autonomous Systems*, 19(1):67–83, 1996.
9. N. Eldredge and S. J. Gould. *Punctuated Equilibria: An Alternative to Phyletic Gradualism*. Freeman, Cooper and Co. New York, NY, 1972.
10. S. Nolfi and D. Floreano. *Evolutionary Robotics: The Biology, Intelligence, and Technology of Self-Organizing Machines*. MIT Press, Cambridge, MA, 2000.
11. R. Storn and K. Price. Differential evolution – a simple and efficient heuristic for global optimization over continuous spaces. *Journal of Global Optimization*, 11(4):341–359, 1997.
12. D. Tasoulis, N. Pavlidis, V. Plagianakos, and M. Vrahatis. Parallel differential evolution. In *IEEE Congress on Evolutionary Computation*, 2004.
13. E. Tuci, C. Ampatzis, A. L. Christensen, V. Trianni, and M. Dorigo. Self-assembly in physical autonomous robots: the evolutionary robotics approach. In *Proceedings of ALIFE XI*, pages 616–623. MIT press, 2008.