Distributed Embodied Evolution for Collective Tasks: Parametric Analysis of a Canonical Algorithm

Pedro Trueba
GII, Universidade da Coruña, Spain
pedro.trueba@udc.es

Abraham Prieto
GII, Universidade da Coruña, Spain
abprieto@udc.es

Francisco Bellas
GII, Universidade da Coruña, Spain
fran@udc.es

ABSTRACT
This paper deals with the formal parametric analysis of an evolutionary paradigm inspired in natural evolution, the distributed Embodied Evolution (EE). The main drawback of this approach is in the high parametric sensitivity it presents, being in addition highly task-dependent. With the aim of understanding the parameter relevance in different fitness landscapes, in this work we present a canonical version of a distributed EE algorithm, propose a set of intrinsic parameters that define its response and analyze them in eight collective problem landscapes.

Categories and Subject Descriptors
I.2.11 [**Artificial Intelligence**]: Distributed Artificial Intelligence – *Multiagent systems*

General Terms
Algorithms

Keywords
Embodied evolution, Open-ended evolution, Multiagent Systems, Collective Intelligence, Algorithm parameterization

1. INTRODUCTION
Embodied Evolution (EE) is an evolutionary paradigm inspired by Darwinian evolution in terms of the use of decentralized and asynchronous open-ended evolution that has been developed to solve collective problems [4]. In EE, the individuals that make up the population are embodied and situated, that is, all of them "live" in an environment (real or simulated) as in the case of ALife and ecology simulations [4].

The most active field in the development of EE algorithms has been that of robotics [1][2][3], mainly due to the self-organization and adaptation properties this type of algorithm intrinsically presents, which makes it highly suitable for real time autonomous systems. Two main approaches have arisen in EE algorithms [2]. In one hand, we have *encapsulated* EE, where each individual carries a whole population of controllers and an independent evolutionary algorithm runs over it. In the other, the *distributed* EE algorithms follow the original idea from Watson [4], and each individual in the population carries only its own genotype. Distributed EE algorithms have received less attention due to the fact that they require larger populations of robots or agents to converge. Nevertheless, they present a high potential coming from the dynamics of the complex interactions that occur within the population, which lead to the emergence of self-adaptive cooperative behaviors. Distributed EE algorithms are highly sensitive to the configuration parameters [1][2], which must be carefully regulated for obtaining valid solutions, and such a regulation is highly task-dependent. To face this problem, here we present a canonical version of a distributed EE algorithm and a set of intrinsic parameters that define its operation.

GECCO'13 Companion, July 6–10, 2013, Amsterdam, The Netherlands.
ACM 978-1-4503-1964-5/13/07.

2. CANONICAL ALGORITHM
The canonical distributed EE used here is based in the AsiCo (Asynchronous Situated Coevolution) algorithm developed by Prieto et al [3]. This algorithm has been reduced to its "barebones", isolating the processes that guide individual evolution from the specific scenario. The pseudo-code of the resulting canonical distributed EE is the following:

```
Initialization
    for individual_i ∈ population
        φ_i ← generateRandomGenome
end
while simulation active:
    for individual_i ∈ population
        fitness_i ← Ψ(φ_i,φ_{i+1},...,φ_{i+n})     // Assign individual fitness
        if random() < P_mating                      // Mating mode
            individual[i].pending_matings++
            find individuals with pending_matings > 0
            select the best individual for mating
            embryo_i ← recombination(φ_i,φ_j)       // Recombination
        if random() < P_replacement                 // Replacement operator
            genotype_i ← embryo_i
            reset individual
    end
end
```

3. INTRINSIC PARAMETERS
From the analysis of general EE algorithms we have extracted that their operation can be decomposed in two fundamental processes, mating and replacement. Mating involves the sub-processes of evaluating candidates, selecting them and combining their genetic codes, and replacement implies the creation of a new individual and the elimination of its preceding one. As so the following intrinsic parameters has been defined:

Mating frequency: it is modeled with a uniform probability (P_{mating}) that represents the probability for an individual to find a mating candidate in every time step. It is calculated as the inverse of the average time between mating, based on the maximum lifetime (T_{max}) and the maximum number of matings (S_{max}):

$$P_{mating} = \frac{1}{T_{mating}} = \frac{S_{max}}{T_{max}}$$

Mating selection criteria: in this work the selection criterion is based on the relative fitness of the candidates (the individual with the best fitness is selected).

Genotypic recombination: it is performed as a combination of two recombination strategies, a local search operator and a bipolar crossover (see [3] for details). It has been defined here as the probability of using the local search (P_{ls}). Therefore, the crossover probability can be defined as $P_{cr} = 1 - P_{ls}$. This parameter can also be considered as a balance between the exploration (crossover) and exploitation (local search) of the recombination.

Replacement: after a certain number of mating replacement occurs and that event is triggered based on a probability P_{rep} based on the expected lifetime (T_{exp}): $P_{rep} = \frac{1}{T_{exp}}$

With T_{exp} being defined as a piecewise function based on the current fitness of the individual (Q_i) using a linear model. It introduces two new parameters, the minimum lifetime (maturity time T_{mat}) and a *2/3 lifetime coefficient* ($C_{2/3}$) which represents the T_{exp} for individuals with a fitness value equal to 2/3 of the maximum current fitness (Q_{max}).

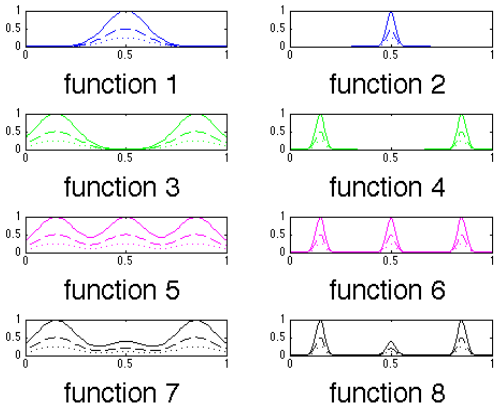

Figure 1: Individual fitness functions. The x-axis represents the genotype and the y-axis the fitness value

4. RESULTS

The effect of four intrinsic parameters, T_{mat}, $C_{2/3}$, S_{max} and P_{ls}, has been studied in detail (T_{max} is a time scale parameter and it has been fixed to 10000 time steps). A set of random values of these four parameters within a predefined range has been considered. T_{mat} was set to be up to 10% of T_{max}. S_{max} ranges between 1 and 20 (being 20 the population size in this case). $C_{2/3}$ ranges for its whole domain, from 0 to 1. Finally, P_{ls} ranges from 0 to 1 too, but in the result's discussion will be presented as a percentage. The canonical algorithm has been executed during 100.000 iterations and repeated five times, up to a total number of 2000 different parameter configurations. The global fitness is calculated as the average of the individual fitness of individuals along the whole simulation.

Figure 1 displays the eight different individual fitness functions that were considered, corresponding to tasks requiring one species (functions 1 and 2), two (functions 3 and 4), three (functions 5 and 6) and three with a sub-optimal one (functions 7 and 8). In each case, the second function is defined with a narrow peak in the optima, making it more complex. The lines displayed in Figure 1 represent the individual fitness for different configurations of the rest of genotypes: the solid line is obtained when the rest of the population is optimally distributed, the dotted line corresponds to a randomly created population and the dashed line is for a situation in between.

In Figure 2 we present the overall results obtained for each intrinsic parameter in the eight functions. For the sake of clarity, only the parameter configurations that produced a fitness value greater than the 90th percentile of the whole set of the 2000 runs were depicted. As it can be observed, $C_{2/3}$ must be fixed to a higher value as the complexity of the problem is increased (functions 2, 4, 6 and 8), leading to a more explorative strategy that maintains the worse individuals during a longer period of time. T_{mat} presents an optimal value in its lowest one (T_{mat}=1), being this low sensitivity a very interesting conclusion for us. P_{ls} results confirm that, in general, a pure local search strategy (P_{ls}>90%) provides better overall results, being crossover highly destructive when dealing with species. Finally, S_{max} is remarkably stable in its highest value (S_{max}=20) meaning that a exploitative strategy is better in general, although as the complexity increases exploration is beneficial (in the same line as $C_{2/3}$ behavior).

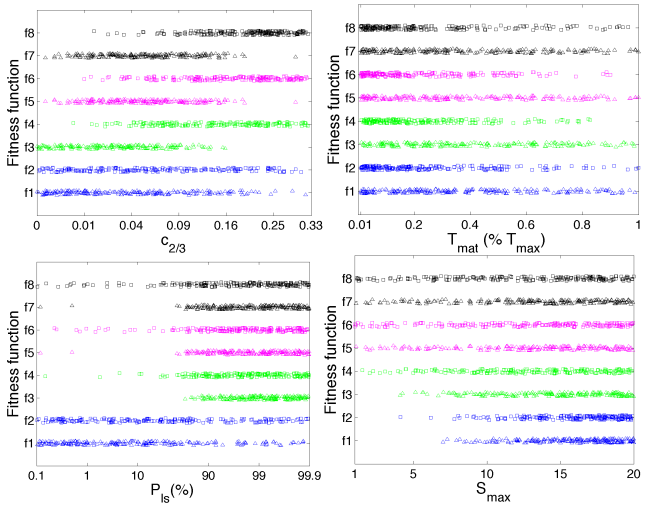

Figure 2. $C_{2/3}$, T_{mat}, P_{ls} and S_{max} variation for the eight functions of Figure 1 in the configurations with best fitness

5. CONCLUSIONS

A canonical version of a distributed Embodied Evolution algorithm has been presented together with a set of four intrinsic parameters that characterize its response. It has provided formal and general conclusions about the parameter influence in this type of algorithm over different types of fitness landscapes.

6. REFERENCES

[1] Bredeche, N., Montanier, J.M., Liu, W., Winfield, A. 2012. Environment-driven Distributed Evolutionary Adaptation in a Population of Autonomous Robotic Agents, *Mathematical and Computer Modelling of Dynamical Systems, 18*: 101-129

[2] Karafotias, G., Haasdijk, E., Eiben, A.E. 2011. An algorithm for distributed on-line, on-board evolutionary robotics, *In Proceedings GECCO '11,* pp. 171-178

[3] Prieto, A., Becerra, J.A., Bellas, F., Duro, R.J. 2010. Open-ended Evolution as a means to Self-Organize Heterogeneous Multi-Robot Systems in Real Time, *Robotics and Autonomous Systems*, vol. 58, pp. 1282-1291.

[4] Watson, R., Ficici, S., Pollack, J. 2002. Embodied evolution: Distributing an evolutionary algorithm in a population of robots, *Robotics and Autonomous Systems,* 39(1): 1-18