HIT-EE: a Novel Embodied Evolutionary Algorithm for Low Cost Swarm Robotics
Nicolas Bredeche

To cite this version:
Nicolas Bredeche. HIT-EE: a Novel Embodied Evolutionary Algorithm for Low Cost Swarm Robotics. ACM GECCO, 2019, Prague, Czech Republic. hal-03175256

HAL Id: hal-03175256
https://hal.sorbonne-universite.fr/hal-03175256
Submitted on 19 Mar 2021

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L’archive ouverte pluridisciplinaire HAL, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d’enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.
HIT-EE: a Novel Embodied Evolutionary Algorithm for Low Cost Swarm Robotics
Nicolas Bredeche

To cite this version:
Nicolas Bredeche. HIT-EE: a Novel Embodied Evolutionary Algorithm for Low Cost Swarm Robotics. ACM GECCO, 2019, Prague, Czech Republic. hal-03175256

HAL Id: hal-03175256
https://hal.sorbonne-universite.fr/hal-03175256
Submitted on 19 Mar 2021

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L’archive ouverte pluridisciplinaire HAL, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d’enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.
HIT-EE: a Novel Embodied Evolutionary Algorithm for Low Cost Swarm Robotics

Nicolas Bredeche
Sorbonne Université, CNRS, Institut des Systèmes Intelligents et de Robotique, ISIR, F-75005 Paris, France
nicolas.bredeche@sorbonne-universite.fr

ABSTRACT
This paper presents a novel distributed on-line evolutionary learning algorithm for swarm robotics that can cope with very limited hardware, as expected from using a swarm of low cost robots. The algorithm is able to deal with hardware constraints over the communication bandwidth by sharing only a limited amount of information, using a recombination operator inspired from bacterial conjugation. Using a classic foraging task, we show that the algorithm converges towards stable and efficient solutions even though, as expected, it converges slower when the bandwidth is limited. However, we also show that the proposed algorithm performs a trade-off between convergence speed and absolute performance that depends on the amount of bandwidth available. The recombination operator yields better performance if communication is limited, as recombination makes the most from the genetic material already present in the population. In other words, quality outweighs convergence speed if the bandwidth is limited.

KEYWORDS
Embodied evolution, low cost robots, swarm robotics

ACM Reference Format:

1 ALGORITHM
The HIT-EE algorithm (short for “Horizontal Information Transfer for Embodied Evolution”) differs from other embodied evolution algorithms [1] by introducing two new ideas. Firstly, a maturation time is set that allows any new individual to remain protected while its quality with respect to a reward function is estimated. Combined with a sliding window, it allows to maintain an estimation of the average reward of a particular behaviour that can be compared to other.

Secondly, we introduce a recombination operator, termed the transfer operator, that can send only a subset of one’s genome. Whenever the sender is deemed better than the receiver, the bits of genome sent will overwrite the corresponding bits in the receiver. The amount of information sent will then depends on the available bandwidth. Depending on the transferRate (tsf for short), the receiver’s genome may be slightly altered (tsf = ε, very low bandwidth), recombined (moderate bandwidth) or completely overwritten (tsf = 1.0, large bandwidth).

Our algorithm is loosely inspired from bacterial conjugation, which is a mechanism for horizontal gene transfer used by bacteria. Similar to bacteria where genes are sent from a living donor to a receiver though physical contact, two robots within communication range may send a subset of their control parameters (cf. also [2] for a classic GA implementation of the same idea).

Algorithm 1 The HIT-EE algorithm
1: selectionRate = m // m is a value between 0.0 and 1.0
2: transferRate = t // t is a value between 0.0 and 1.0
3: maturationDelay = d // d is an integer value (strictly positive
4: genome.initialize() // e.g. random values
5: reward = 0 // similar to “fitness value”
6: age = 0
7: newGenome = False
8: while forever do
9:  move() // execute the agent’s controller for one step.
10:  reward = updateReward() // e.g.: use a sliding window of size t
11:  if age > maturationDelay then
12:      age = age + 1
13:      broadcast(genome, transferRate, reward) // (subset of) genome
14:      incomingPackets = listen() // returns received packets since last iteration
15:      for p in incomingPackets do
16:         if p.reward >= reward then
17:            copy p.genomeBits to genome // i.e. conjugation
18:            mutate genome using mutationRate
19:            newGenome = True
20:      end if
21:      end for
22:      if newGenome == True then
23:         reward = 0, age = 0, newGenome = False
24:      end if
25:      end if
26:   end while

2 EXPERIMENTS
The obvious question is how the HIT-EE algorithm can cope when communication is constrained. We use a foraging task where the reward (i.e. the fitness function) is given by the number of items captured by a robot. Each robot is controlled by a simple Perceptron which weights are evolved. We perform a first set of experiments with two different values for the transfer operator. We experiment with transfer rates where either half (tsf = 0.5) or most (tsf = 0.9) of the genome (i.e. the Perceptron’s weights) can be transferred (cf. Table 1 for other settings). Mutation rate is arbitrarily set to zero: while this may be sub-optimal in terms of exploration, this allows a clear understanding of the effect of the transfer operator.

Results for the two different settings are presented in Figures 2 and 3. The tsf = 0.50 and tsf = 0.90 variants both converge to a
Figure 1: Arena with 150 robots (small dots) and 100 items (big dots). The fitness function for each robot counts the number of items captured (i.e., foraging task).

Table 1: Control parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>General parameters</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Population size</td>
<td>150</td>
<td></td>
</tr>
<tr>
<td>Number of items</td>
<td>100</td>
<td></td>
</tr>
<tr>
<td>Arena size</td>
<td>1400x800</td>
<td></td>
</tr>
<tr>
<td>Robot size</td>
<td>5x5</td>
<td></td>
</tr>
<tr>
<td>Sensor&amp;communication range</td>
<td>16</td>
<td>max.2000 gens</td>
</tr>
<tr>
<td>iterations</td>
<td>800400</td>
<td></td>
</tr>
<tr>
<td>replications</td>
<td>64</td>
<td></td>
</tr>
<tr>
<td>Controller and encoding</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Sensory inputs</td>
<td>163</td>
<td>16 sensors</td>
</tr>
<tr>
<td>Motor outputs</td>
<td>2</td>
<td>left and right</td>
</tr>
<tr>
<td>Genome size</td>
<td>326</td>
<td></td>
</tr>
<tr>
<td>HIT-EE parameters</td>
<td></td>
<td></td>
</tr>
<tr>
<td>maturation time</td>
<td>400</td>
<td></td>
</tr>
<tr>
<td>sliding window</td>
<td>400</td>
<td></td>
</tr>
<tr>
<td>transfer</td>
<td>0.5 or 0.9</td>
<td></td>
</tr>
<tr>
<td>mutation</td>
<td>0.0</td>
<td>no mutation here</td>
</tr>
</tbody>
</table>

Figure 2: using a 50% genome transfer per encounter. Mean = 1.6845614385965, standard deviation = 0.1128009972487

Figure 3: using a 90% genome transfer per encounter. Mean = 1.5773333272727, standard deviation = 0.21354009301578

The relatively large genome size is due to redundant and/or useless sensory information fed to the controller (which is itself a simple Perceptron, with no hidden layer). Such a large input space is useful to avoid the initialisation of a "lucky" candidate at generation 0.

ACKNOWLEDGMENTS

This work was supported by the Agence Nationale pour la Recherche under Grant ANR-18-CE33-0006.

REFERENCES