Fate Agent Evolutionary Algorithms with Self-Adaptive Mutation

Arthur Ervin Avramiea  Giorgos Karafotias  A.E. Eiben
VU University Amsterdam  VU University Amsterdam  VU University Amsterdam
a.e.avramiea@student.vu.nl  g.karafotias@vu.nl  gusz@cs.vu.nl

ABSTRACT
Fate Agent EAs form a novel flavour or subclass in EC. The idea is to decompose the main loop of traditional evolutionary algorithms into three independently acting forces, implemented by the so-called Fate Agents, and create an evolutionary process by injecting these agents into a population of candidate solutions. This paper introduces an extension to the original concept, adding a mechanism to self-adapt the mutation of the Breeder Agents. The method improves the behaviour of the original Fate Agent EA on dynamically changing fitness landscapes.

Categories and Subject Descriptors
I.2.8 [Problem Solving, Control Methods, and Search]: Heuristic methods

Keywords
Distributed EAs; parameter control; dynamic problems

1. INTRODUCTION
Evolutionary computation implemented through Fate Agents is a new approach introduced in [2]. The motivational vision is grounded in Adaptive Collective Systems, for example in swarm robotics or, more generally, in distributed systems of autonomous agents, e.g. wireless sensor networks, smart devices, smart vehicles, etc. The main idea is to make such a collective system adaptive by injecting a force that can adjust the behavioural policies (a.k.a. controllers) of the individual units.

Such systems need to be able to adjust their initial pre-deployment settings to unexpected circumstances and they should be able to cope with changes. Using evolutionary techniques the swarm can be naturally considered as a population where each individual is (the controller of) a robotic unit. The "only" challenge is to add selection and variation operators that work on these individuals. To this end, it is important that the evolutionary process (i) is decentralised (hence scalable), (ii) can self-calibrate its own parameters on the fly (needs no user tuning), and (iii) can cope with changes (can re-calibrate its parameters).

The Fate Agents approach attempts to achieve this by perceiving the controllers of the robots as individuals of a population and it "evolutionarizes" this system by adding three types of Fate Agents to the population. These agents implement the three principal evolutionary operators parent selection, reproduction, and survival selection, acting on the original individuals (robot controllers) and on the Fate Agents themselves.

Our work addresses the classic theme of adaptivity and parameter control in EC [6]. The Fate Agents EA is a spatially structured EA [8], [1], however, unlike most spatial EAs, it does not rely on a centralised "oracle". The combination of spatial structure and parameter control has been studied in [5] and [4]. Our system can also be related to meta-evolution [3], in particular the so-called local meta-evolutionary approaches [7] (not the meta-GA lookalikes).

2. FATE AGENTS
Our Fate Agents EA is situated in a (virtual) space where agents move and interact. The evolving population consists of passive Candidate Solutions and active Fate Agents that embody EA operators. Candidate Solution agents only carry a genome representing a solution to the given problem. Fate Agents embody evolutionary operators and evolve themselves because they act not only upon candidate solutions but also upon each other. They have a limited range of perception and action, thus, the evolutionary process is fully distributed as there is no central authority but different parts of the space are regulated by different agents. A Fate Agent is evaluated by the fitness of the fittest candidate solution in its range.

There are three types of Fate Agents. Cupids and reapers realise parent and survival selection respectively using tournaments. Their genome consists of tournament sizes and selection probabilities for each agent type. Breeders perform variation (recombination and mutation). In our initial experiments with numeric optimisation, breeders used Gaussian mutation\(^1\) and their genome consisted of three values: the mutation step sizes for candidate solutions, cupids and reapers. A step size for mutating breeders was not included; breeders act differently upon themselves and upon other agent types. If the breeders’ mutation step size was evolved within their genome it would be used to mutate itself leading to a positive feedback loop and exploding values. Instead, breeders are mutated using the Evolution Strategies’ self-adaptation rule \(x' = x + e^{\tau N(0,1)}\), where \(\tau\) is the learning rate constant.

3. ADAPTIVE BREEDERS LEARNING RATE
The constant learning rate for breeders introduces inflexibility to the otherwise self-regulated Fate Agents EA. Breeders are the source of adaptivity but they themselves have a constant rate of

\(^1\)And averaging recombination that has no parameters.
mutation. This can be most restrictive when solving dynamic problems, especially in the case of cataclysmic changes in the environment when the population has to quickly respond to the new situation. For this reason we introduced the Adaptive Breeders Learning Rate (ABLR) mechanism. It affects the learning rate $\tau$ by using a mutation. This can be most restrictive when solving dynamic problems.

<table>
<thead>
<tr>
<th>Scn</th>
<th>Alg</th>
<th>Epo</th>
<th>Epo</th>
<th>Epo</th>
<th>Epo</th>
<th>Epo</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>Original ABLR</td>
<td>0.2285</td>
<td>0.1963</td>
<td>0.1029</td>
<td>0.0255</td>
<td>0.0008</td>
</tr>
<tr>
<td></td>
<td>ABLR</td>
<td>0.9994</td>
<td>0.2371</td>
<td>0.0521</td>
<td>0.0083</td>
<td></td>
</tr>
<tr>
<td>B</td>
<td>Original ABLR</td>
<td>0.4631</td>
<td>0.1262</td>
<td>0.0998</td>
<td>0.0684</td>
<td>0.5963</td>
</tr>
<tr>
<td></td>
<td>ABLR</td>
<td>0.5383</td>
<td>0.183</td>
<td>0.6301</td>
<td>0.5963</td>
<td>0.9172</td>
</tr>
<tr>
<td>C</td>
<td>Original ABLR</td>
<td>0.6158</td>
<td>0.0596</td>
<td>0.3664</td>
<td>0.298</td>
<td>0.0124</td>
</tr>
<tr>
<td></td>
<td>ABLR</td>
<td>0.5383</td>
<td>0.5667</td>
<td>0.4394</td>
<td>0.5685</td>
<td>0.1589</td>
</tr>
</tbody>
</table>

5. CONCLUSIONS

We aimed at improving the Fate Agents EA’s ability to cope with changes. To this end, we redefined the working of the reproduction agents by adding the ABLR mechanism that makes the mutation operator they use self-adaptive. For an experimental assessment we used synthetic fitness landscapes and defined 3 scenarios composed of 5 epochs with cataclysmic changes between them.

The experimental results showed that the Fate Agents EA with the ABLR mechanism performs better than the original exhibiting much higher best fitnesses after each epoch. Furthermore, the ABLR enhanced algorithm very often makes a “full” recovery as opposed to the original that almost never does.

6. REFERENCES


2http://coco.gforge.inria.fr/doku.php?id=bbob-2013

3Source code at www.few.vu.nl/~gks290/resources/FateABLR.tar.gz

4Converged to a very different previous landscape.