Towards Efficient Online Reinforcement Learning Using Neuroevolution

Jan Hendrik Metzen, Frank Kirchner
Robotics Lab
DFKI GmbH, Bremen, Germany
{jhm,frank.kirchner}@informatik.uni-bremen.de

Mark Edgington, Yohannes Kassahun
Robotics Group
University of Bremen, Bremen, Germany
{edgimar,kassahun}@informatik.uni-bremen.de

ABSTRACT

For many complex Reinforcement Learning (RL) problems with large and continuous state spaces, neuroevolution has achieved promising results. This is especially true when there is noise in sensor and/or actuator signals. These results have mainly been obtained in offline learning settings, where the training and the evaluation phases of the systems are separated. In contrast, for online RL tasks, the actual performance of a system matters during its learning phase. In these tasks, neuroevolutionary systems are often impaired by their purely exploratory nature, meaning that they usually do not use (i. e. exploit) their knowledge of a single individual's performance to improve performance during learning. In this paper we describe modifications that significantly improve the online performance of the neuroevolutionary method Evolutionary Acquisition of Neural Topologies (EANT) and discuss the results obtained in the Mountain Car benchmark.

Categories and Subject Descriptors

I.2.6 [**Artificial Intelligence**]: Learning—*Connectionism and neural nets*

General Terms

Algorithms

Keywords

Reinforcement Learning, Neuroevolution, Online-Learning

1. ONLINE NEUROEVOLUTION

Neuroevolution (NE) is a combination of Evolutionary Algorithms (EAs) with Artificial Neural Networks (ANNs), in which an ANN is encoded as a genome. The EA searches for a genotype whose corresponding phenotype ANN maximizes a fitness measure for a given task. When applied to a Reinforcement Learning (RL) task, the ANN usually directly encodes the policy, and the fitness is set to the cumulative reward accrued during a fixed period of time. In *offline* RL, the goal is to find a policy which maximizes this fitness measure after a given amount of preliminary training. Thus, the performance of the agent during training does not matter. In contrast, in *online* RL the agent must try to maximize its cumulative reward from the very beginning. This goal is difficult to achieve in stochastic environments due to the fact that an agent's performance can only be assessed based on a long-term average of its performance; its short-term performance might be affected by the stochastic nature of the environment.

When applying a neuroevolutionary method to an Online RL task, there are three conflicting objectives: (1) The NE method should prefer to apply policies that are known to perform well (exploitation), (2) the NE method should try new policies in order to find policies which perform even better than the best-known policies (exploration), and (3) the NE method should estimate the fitness of a policy as accurately as possible (accuracy). The tradeoff between these three objectives is illustrated in Figure 1.

In NE, normally only the exploitation-exploration tradeoff is considered, which is well known in the field of RL. The other two tradeoffs are usually handled implicitly by assigning the same number of evaluations to each individual. In the context of online RL this leads to a significant decrease in performance since even very bad policies are evaluated several times in order to obtain an accurate estimate of their fitness. Whiteson et al. [2] propose methods that can handle the exploitation-accuracy tradeoff. One method they discuss is softmax policy selection which distributes a fixed number of overall evaluations available for a generation among the members of the generation's population such that better performing individuals are evaluated more often. However, the exploration-accuracy tradeoff is not handled automatically.

We propose a way of handling all three tradeoffs discussed above that uses a steady-state (non-generational) evolutionary algorithm and evaluates each individual until either the hypotheses h_1 that its true fitness is below a (dynamic)

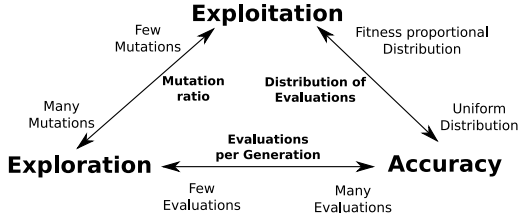

Figure 1: The three conflicting objectives of neuroevolution (exploitation, exploration, and accuracy), and how they can be controlled by a neuroevolutionary method.

GECCO'08, July 12–16, 2008, Atlanta, Georgia, USA.
ACM 978-1-60558-130-9/08/07.

```
Online-EANT(num_indiv):
    adolescents ← Create-Individuals(num_indiv)
    matures ← ∅
    while True do
        individual ← Choose-Randomly(adolescents)
        Evaluate-Individual(individual)
        if Likely-Worse(individual,matures) then
            adolescents.replace(individual,Create-New-
            Offspring(matures))
        else if
        Fitness-Accurately-Estimated(individual)
        then
            if Fit-Enough(individual,matures) then
                matures.add(individual)
            adolescents.replace(individual,Create-New-
            Offspring(matures))
```

Algorithm 1: The online (steady-state) NE algorithm.

threshold t, or the hypothesis h_2 that its current fitness estimate differs from its true fitness by less than β percent, is valid with significance levels α_1 and α_2, respectively. The proposed algorithm, *Online-EANT*, is outlined in Algorithm 1. Instead of utilizing one population, the algorithm holds a set of "mature" individuals and "adolescent" individuals. The mature individuals are those whose fitness has already been accurately estimated and who have been found to perform sufficiently well. They are the only individuals that are allowed to produce offspring, and the threshold t is defined as the average of their (estimated) fitness values. The adolescent individuals are evaluated until either h_1 or h_2 is valid with significance level α_1 (α_2). If h_2 is valid and the individual's fitness estimate is above t, it becomes a member of the mature individuals and the oldest mature individual "dies". If h_2 is valid but the individual's fitness estimate is below t or if h_1 is valid (i. e. the individual is performing worse than the average member of the mature individuals), the individual "dies" prematurely and is replaced by a new offspring of the mature individuals. The exploitation-accuracy tradeoff and the exploration-accuracy tradeoff are thus handled by the choice of the necessary significance level α_1 and α_2, and the allowed estimation error β.

2. MOUNTAIN CAR BENCHMARK

We have tested three different RL methods in the Mountain Car benchmark [1]: The TD method Sarsa(λ), Offline-EANT, and Online-EANT. For Sarsa, we used the Cerebellar Model Articulation Controller (CMAC) [1] function approximator with a superposition of 10 independent tilings, an eligibility trace decay rate of $\lambda = 0.95$, a learning rate of $\alpha = 0.5$, and a discount factor of $\gamma = 1.0$. Two different values for ϵ (the ratio of choosing a random, non-greedy action) have been tested, namely $\epsilon = 0.0$ and $\epsilon = 0.01$. The initial Q-values were all set optimistically to 0 to enforce initial exploration. For Offline-EANT, a population size of 20 was used, each individual was evaluated for 10 episodes, and fitness sharing was disabled. For Online-EANT, 20 individuals were initially created, the required significance levels α_1 and α_2 were set to 0.05, and the allowed error rate β was set to 20%. The maximum number of steps an individual was allowed to take to reach the goal was restricted to 500. If the goal was not reached after this number of steps, the next individual took over control of the car at its current position. Thus, artificially terminating an episode was not necessary, and neuroevolution was never left with an unfeasible policy that could not reach the goal.

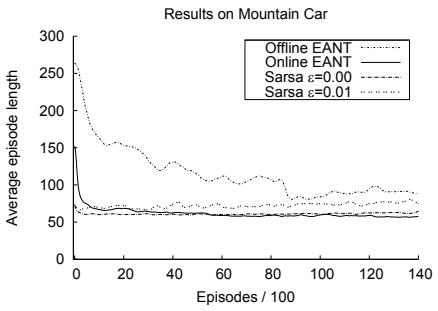

Figure 2: Performance Comparison on the Mountain Car Benchmark. Each curve is an average over at least 10 independent runs and was smoothed using a moving window.

Figure 2 shows the performance of the three methods in the benchmark (the plotted values are the averages over 10 independent runs for each method and are smoothed using moving window averages of length 500). Online-EANT's average episode length is significantly better (i. e. smaller) than Offline-EANT's ($p < 0.003$) over the whole $1.4 * 10^4$ episodes of evaluation. This shows that Online-EANT's choice of evaluating more promising individuals more often not only improves initial performance but also does not interfere with the long-term progress of the evolution. Compared to Sarsa ($\epsilon = 0.0$), Online-EANT performes worse in the early phase of a trial (the difference is significant during the first 250 episodes ($p < 0.05$)) but reaches a similar level after about 4000 and achieves better online performance in the long run (the difference is significant after 12200 episodes ($p < 0.05$)). In contrast, Offline-EANT's performance remains significantly worse than Sarsa's ($p < 0.025$) over the whole evaluation time. Sarsa's superior performance in the initial phase of the trials can be explained by the fact that it makes use of the instantaneous reward supplied by the environment (instead of assessing the policies as a whole) while its worse performance in the long run might be explained by the fact that it is more easily trapped in locally optimal policies (in particular if the learning rate ϵ is set to 0). However, setting ϵ to 0.01 did not yield in an improved online performance (see Figure 2). Compared to the results presented in [2], Online EANT achieves better online performance than the combination of NEAT with any policy selection strategy.

3. REFERENCES

[1] R. Sutton and A. Barto. *Reinforcement Learning. An Introduction*. MIT Press, Massachusetts, London, 1998.

[2] S. Whiteson, M. E. Taylor, and P. Stone. Empirical studies in action selection with reinforcement learning. *Adaptive Behavior - Animals, Animats, Software Agents, Robots, Adaptive Systems*, 15(1):33–50, 2007.