Effects of Speciation on Evolution of Neural Networks In Highly Dynamic Environments

Peter Krčah

Computer Center, Charles University
Ovocný Trh 5, 116 36 Prague 1, Czech Republic
peter.krcah@ruk.cuni.cz

Abstract. Using genetic algorithms for solving dynamic optimization problems is an important area of current research. In this work, we investigate effects of speciation in NeuroEvolution of Augmenting Topologies (NEAT), a well-known method for evolving neural network topologies, on problems with dynamic fitness function. NEAT uses speciation as a method of maintaining diversity in the population and protecting new solutions against competition. We show that NEAT outperforms non-speciated genetic algorithm (GA) not only on problems with static fitness function, but also on problems with gradually moving optimum. We also demonstrate that NEAT fails to achieve better performance on problems where the optimum moves rapidly. We propose a novel method called DynNEAT, which extends NEAT by changing the size of each species based on its historical performance. We demonstrate that DynNEAT outperforms both NEAT and non-speciated GA on problems with rapidly moving optimum, while it achieves performance similar to NEAT on problems with static or slowly moving optimum.

Keywords: NEAT, speciation, neural networks, dynamic optimization

1 Introduction

Real-world optimization problems often contain various sources of uncertainty. When evolutionary algorithms are used to solve such problems, this uncertainty translates into dynamic fitness function, which often poses a challenge for standard EAs. One common approach to improve efficiency of the search is to use multiple concurrently evolving populations, each population tracking a different optimum (a comprehensive survey of methods is given in [2]). In this work, we investigate behavior of NEAT [4], a well-known method for optimizing both structure and weights of a neural network, on problems with dynamic fitness function. We first confirm that speciation is a significant component of NEAT on selected static and dynamic fitness functions. We then show that NEAT performs poorly on problems where the optimum of the fitness function changes rapidly between generations. We propose a new method, called DynNEAT, which extends NEAT by taking into account multiple previous generations when choosing the size of the species in the next generation, thus stabilizing the speciation process and improving performance of the search.

2 Methods

NeuroEvolution of Augmenting Topologies (NEAT) NEAT is a method for evolving both structure and connection weights of artificial neural networks [4].

NEAT uses speciation to maintain diversity in the population by protecting new solutions from direct competition with currently best individuals. Speciation in NEAT works in the following way. In the first generation, each individual is assigned to a different species and its fitness function is evaluated. Each subsequent generation is constructed by first dividing all slots in the population among all species present in the previous generation. Species sizes in generation $i+1$ are allocated proportionally to $s_{NEAT}(i+1)$, an average fitness of all individuals belonging to the given species in the previous generation:

$$s_{NEAT}(i+1) = \frac{\sum_{j=1}^{N_i} f_{ij}}{N_i},$$

where f_{ij} is the fitness value of j^{th} individual of the given species in generation i and N_i is the number of individuals in generation i in the given species. When the new size of each species is known, each slot is populated by performing crossover and mutation of individuals selected from the given species in the previous generation. Newly created individuals are assigned to species not based on their ancestral species, but by comparing them one at a time to representatives of each species from the previous generation and assigning them to the first species whose representative is sufficiently similar (based on a defined threshold). If an individual is not sufficiently similar to a representative of any existing species, a new species is created for it. When all individuals are assigned to species, NEAT continues by evaluating their fitness and repeating the same process for the new generation. Two other major components of NEAT are *historical markings* of neurons and growing neural networks incrementally. Comprehensive description of NEAT is available in [4].

NeuroEvolution of Augmenting Topologies for Dynamic Fitness Functions (DynNEAT) On highly dynamic problems, speciation scheme used by NEAT can be disadvantageous. In NEAT, the size of the species is chosen based solely on the average fitness value of individuals from the previous generation. In highly dynamic problems, this value will change dramatically from generation to generation, leading to dramatic changes in the size of the species. Such radical changes in species size can be detrimental to the progress of the search by removing novel solutions from a species before they can be optimized. To improve the behavior of speciation on such problems, we propose DynNEAT method. In DynNEAT, decisions about the size of the species are based not just on the previous generation, but on t previous generations. Species sizes in generation $i+1$ are allocated proportionally to $s_{DynNEAT}(i+1)$, the maximum average fitness of last t generations:

$$s_{DynNEAT}(i+1) = \max_{j=i-t+1}^{i} \frac{\sum_{k=1}^{N_j} f_{jk}}{N_j}.$$

Such method of sizing the species ensures stability of the species across generations even when the optimum of a fitness function moves rapidly. Parameter t controls for how long can DynNEAT maintain the size of a species when the average fitness of individuals in the species decreases over generations. In this work, the parameter value was set to 5 in all experiments.

3 Experiments

To evaluate the performance of NEAT and DynNEAT on dynamic optimization problems, we perform three different experiments, using increasingly dynamic versions of the *function approximation* problem. Each fitness evaluation consists of 600 steps, during which a single input value increases linearly from -1 to $+1$. The resulting fitness value is computed as $1 - min(\mu, 1)$, where μ is the mean squared error of the differences between expected and real output value measured in each step. In all three experiments, non-speciated GA is compared to NEAT and DynNEAT. As a non-speciated GA, we use a standard GA with the addition of historical markings used for crossover of neural networks. Each configuration was tested in 50 runs. Each run was stopped after 200 generations (after which most runs achieved a plateau). Significance levels were computed using Student's t-test. Population size was set to 300, to allow more species to form concurrently and cover different optima of the changing fitness function.

In the first experiment, with a static fitness function, the target function $f_A(x)$ consists of a linear part, a constant part and a sine-wave part and is defined in the following way (see solid line in fig. 1):

$$f_A(x) = \begin{cases} 8x - 1 & \text{if } 0 \leq x < \frac{1}{4} \\ 1 & \text{if } \frac{1}{4} \leq x < \frac{1}{2} \\ cos(4\pi n(x - \frac{1}{2})) & \text{if } \frac{1}{2} \leq x \leq 1 \end{cases},$$

where $x = (k-1)/599$ is the current evaluation step scaled to interval $[0, 1]$.

In the second experiment, with a slowly moving optimum (SMO), the target function is defined in the following way (see fig. 1):

$$f_B(x, y) = \frac{1}{2}(1 + \sin\frac{2\pi y}{50}) f_A(x),$$

where x is defined as in $f_A(x)$ and y is the generation counter. The target function oscillates between $f_A(x)$ and 0 with a period of 50 generations.

In the third experiment, with rapidly moving optimum (RMO), the target function is defined in the following way:

$$f_C(x, y) = \begin{cases} 1 & \text{if } (x < 0.5) \text{ xor } (y \text{ is even}) \\ -1 & \text{otherwise} \end{cases},$$

where x and y are defined as in $f_B(x, y)$. For odd generations, $f_C(x, y)$ is a step function with value 1 in first half of the domain and -1 in the second half of the domain. For even generations the function values are reversed.

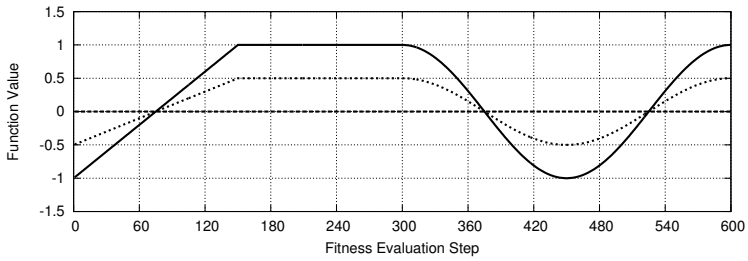

Fig. 1. Expected Output in Experiments with Static (solid line) and Slowly Moving Optimum (solid, dashed and dotted lines).

4 Results

In the experiment with the static fitness function, both DynNEAT and NEAT were able to consistently find good solutions (see fig. 2). The average maximum fitness achieved by NEAT in the 200th generation was 0.980 (σ=0.009), while DynNEAT achieved 0.988 (σ=0.0067). Non-speciated GA achieved average maximum fitness of only 0.741 (σ=0.169), with 30 of 50 runs failing to find a solution with fitness value above 0.603. Differences between any two methods are statistically significant ($p < 0.01$) in each generation since generation 60.

Results of experiments with slowly moving optimum (see fig. 2) reflect the periodicity of the problem. Since zero target output is trivial to solve compared to more complex outputs, the difficulty of the problem changes from generation to generation. Fig. 2 shows that all methods were able to find successful solutions when fitness function was close to zero, but their performance differed in generations where fitness function is furthest away from zero. In these generations, both DynNEAT and NEAT significantly outperformed non-speciated GA ($p < 0.01$ since generation 20, except 36-40 and 85-90). Differences between NEAT and DynNEAT were not statistically significant ($p > 0.2$).

In the experiment with rapidly moving optimum, DynNEAT was the only method capable of consistently finding good solutions to both target functions (see fig. 2). The average maximum fitness achieved by DynNEAT was 0.9547 (σ=0.0295), while neither NEAT nor non-speciated GA achieved fitness over 0.8. NEAT performed only marginally better than GA, with the average maximum fitness of 0.7794 (σ=0.1044) compared to 0.7376 (σ=0.1280) in GA ($p < 0.01$).

5 Discussion and Future Work

The experiment with the static fitness confirmed that the lack of speciation results in a significant drop in the performance of non-speciated GA. Moreover, the same effect occurs with a slowly moving optimum (SMO), which shows that advantages of speciation can also be utilized in dynamic problems. However, the experiment with rapidly moving optimum (RMO) demonstrates that when

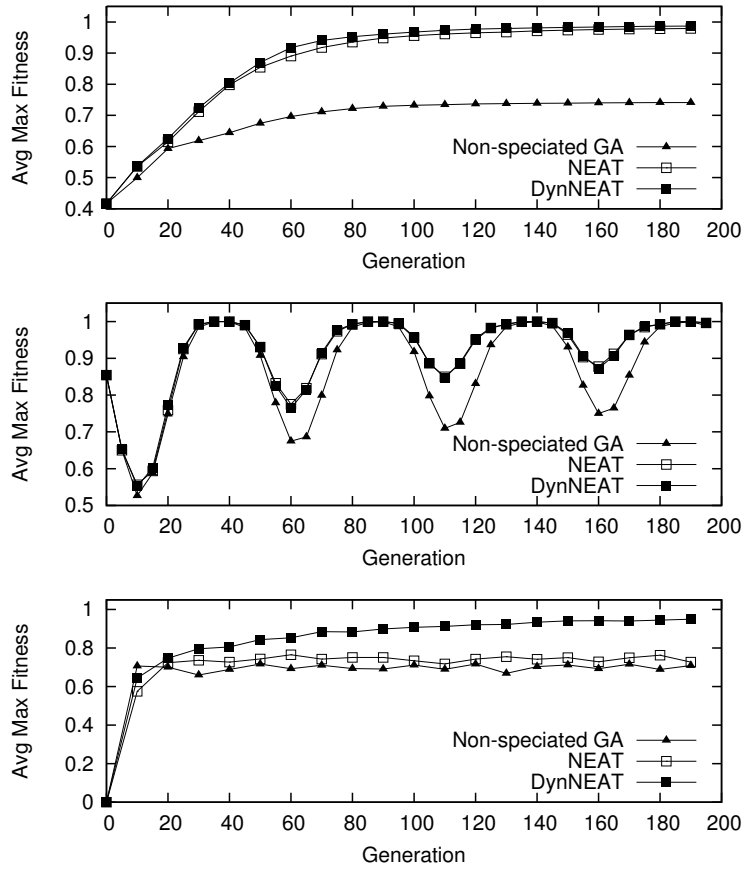

Fig. 2. Comparison of Average Maximum Fitness on a Fitness Function with Static, Slowly Moving and Rapidly Moving Optimum (from top to bottom).

the fitness is highly dynamic, NEAT fails to provide significant benefit over non-speciated GA (see fig. 2). The drop in performance can be explained by examining the dynamics of the speciation process. In order for the speciation to be effective, species must be long-lived to give their members enough time for adaptation. In RMO experiment, NEAT required 7.16 times more species than in SMO experiment (580 vs. 80.9), with an average lifespan shorter by a factor of 6.85 (3.64 vs. 24.94). DynNEAT, on the other hand, achieves high average lifespan of species even in RMO experiment (10.13 generations vs. 3.64 in NEAT) and smaller average number of species (202 vs. 580 in NEAT). This is further demonstrated by the distribution of species lifespans shown in fig. 3.

We applied DynNEAT to a problem where fitness alternates between two simple states. Such problem was chosen to clearly demonstrate the differences in speciation dynamic between NEAT and DynNEAT. In future works we would

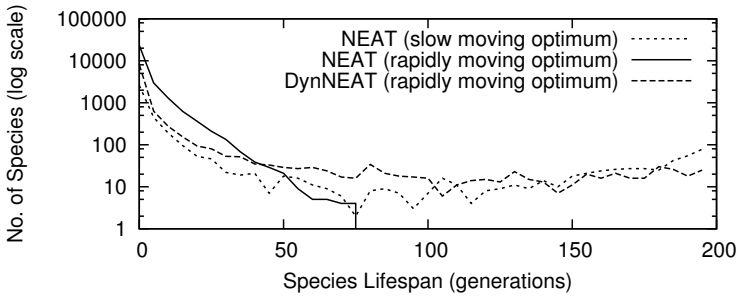

Fig. 3. Distribution of Species Based on Their Lifespan. All species from all 50 runs are included in the distribution for each method. Lifespan of a species is the number of generations in which at least one individual belonged to that species.

like to extend these results to study the influence of parameter t (which was fixed in this work) and compare DynNEAT to other dynamic optimizers using benchmark functions (e.g. MPB [1]). Another direction for future research is to extend these results to methods derived from NEAT, such as HyperNEAT [3].

6 Conclusions

In this work, we investigate the effects of speciation on three increasingly more dynamic problems using NEAT method. We have shown that the dynamics of speciation in NEAT is disrupted when the problem becomes highly dynamic, significantly impacting NEAT performance. To address this problem, we proposed DynNEAT, an extension of NEAT capable of maintaining species even in a highly dynamic environment. We have shown that DynNEAT significantly outperforms NEAT on a highly dynamic problem and achieves similar performance on problems with static or slow-changing optimum. Analysis of the speciation has confirmed that DynNEAT achieves its performance by stabilizing speciation, which allows long-lived species to form even in cases when fitness of individuals dramatically changes between generations.

References

1. J. Branke. Memory enhanced evolutionary algorithms for changing optimization problems. In *Congress on Evolutionary Computation, CEC'99*, pages 1875–1882. IEEE, 1999.
2. Y. Jin and J. Branke. Evolutionary optimization in uncertain environments-a survey. *IEEE Trans. Evolutionary Computation*, pages 303–317, 2005.
3. K. O. Stanley, D. B. D'Ambrosio, and J. Gauci. A hypercube-based encoding for evolving large-scale neural networks. *Artificial Life*, 15(2):185–212, 2009.
4. K. O. Stanley and R. Miikkulainen. Efficient reinforcement learning through evolving neural network topologies. In *Proceedings of the Genetic and Evolutionary Computation Conference*, San Francisco, CA, 2002. Morgan Kaufmann.