Using Behavioral Exploration Objectives to Solve Deceptive Problems in Neuro-evolution

Jean-Baptiste Mouret
ISIR, CNRS UMR 7222
Université Pierre et Marie Curie (UPMC)
4 place Jussieu, F-75005, Paris, France
mouret@isir.fr

Stéphane Doncieux
ISIR, CNRS UMR 7222
Université Pierre et Marie Curie (UPMC)
4 place Jussieu, F-75005, Paris, France
doncieux@isir.fr

ABSTRACT

Encouraging exploration, typically by preserving the diversity within the population, is one of the most common method to improve the behavior of evolutionary algorithms with deceptive fitness functions. Most of the published approaches to stimulate exploration rely on a distance between genotypes or phenotypes; however, such distances are difficult to compute when evolving neural networks due to (1) the algorithmic complexity of graph similarity measures, (2) the competing conventions problem and (3) the complexity of most neural-network encodings.

In this paper, we introduce and compare two conceptually simple, yet efficient methods to improve exploration and avoid premature convergence when evolving both the topology and the parameters of neural networks. The two proposed methods, respectively called *behavioral novelty* and *behavioral diversity*, are built on multiobjective evolutionary algorithms and on a user-defined distance between behaviors. They can be employed with any genotype. We benchmarked them on the evolution of a neural network to compute a Boolean function with a deceptive fitness. The results obtained with the two proposed methods are statistically similar to those of NEAT and substantially better than those of the control experiment and of a phenotype-based diversity mechanism.

Categories and Subject Descriptors

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

General Terms

Algorithms

Keywords

Neural networks; multiobjective evolutionary algorithm; diversity; deceptive problems

GECCO'09, July 8–12, 2009, Montréal Québec, Canada.

1. INTRODUCTION

Finding good solutions for deceptive problems is one of the main challenge in evolutionary computation since the pioneers' work in this field [12, 13]. Starting with the assumption that a well spread population has a higher probability to find new fitness peaks than a population focused around only one peak, one of the most classic modification of the basic evolutionary algorithm to tackle deceptive problems is to explicitly maintain a high diversity of candidate solutions [17, 14]. Improving the diversity within the population can also be viewed as a way to tune the exploration/exploitation trade-off underlying the evolutionary process in favor of more exploration: some inefficient candidate solutions can be kept in the population only because they are different from their counterparts.

Artificial neural networks have drawn considerable attention for the last twenty years mainly thanks to their versatility, as their range of application covers from the control of robots to data clustering. Evolutionary algorithms have often been successfully employed to design their topology and set their weights, leading to *neuro-evolution* processes (e.g. [15, 29, 27, 10, 22]). However, despite the early interest in diversity for parameter optimization, current diversity preservation approaches are still difficult to exploit in neuro-evolution. One of the main reason is the algorithmic complexity to compute the similarity between candidate solutions, a NP-hard problem for unconstrained oriented graphs [3] and therefore for most neural networks. Another reason is the complexity of encodings used in neuro-evolution, making each diversity maintenance method specific to a particular and often complex setup. The competing convention problem (see e.g. [29]) is a third difficulty to overcome when trying to compare neural networks. Furthermore, setting the parameters that controls the exploration/exploitation trade-off is often challenging and empiric, but critical for the efficiency of the evolutionary process.

In this paper, we introduce and compare two simple, generic, and efficient ways to improve exploration to bypass the problems posed by deceptive fitness functions when evolving both the topology and the parameters of neural networks. The two proposed methods, respectively called *behavioral novelty* and *behavioral diversity*, rely on Pareto-based multiobjective evolutionary algorithms (see [7]) and on a user-defined distance between *behaviors*. To assess the efficiency of this behavioral approach, we employed the approximation of a Boolean function by a neural network with a deceptive fitness. The results obtained with both meth-

ods are compared with those of NEAT [27], which includes a sophisticated diversity maintenance mechanism and with those obtained using a genotype diversity based on a graph distance (graph probing, [21]).

2. RELATED WORK

Many methods have been published to maintain diversity in evolutionary algorithms in order to improve their exploration capabilities and avoid premature convergence (e.g. [14, 4, 6]). Among those, *fitness sharing* [14] is probably the most common one: the search landscape is modified by reducing the fitness of individuals in densely-populated regions, thus promoting search in less explored regions. To that aim, the fitness of each individual i is divided by a "niche coefficient", defined as the number of individuals whose distance to i is below a user-defined threshold. Typical distances are computed between genotypes, one of the simplest example being a Hamming distance between binary strings in a binary-based genetic algorithm.

Dividing the fitness by the niche coefficient, as in fitness sharing, is an attempt to combine a performance objective, maximizing the fitness, and an exploration objective, minimizing the niche coefficient. Recent research in evolutionary computation proposed numerous algorithms to simultaneously optimize several objectives instead of aggregating them to a single one. Most of them rely on the concept of dominance:

DEFINITION 1. *A solution $\mathbf{x}^{(1)}$ is said to dominate another solution $\mathbf{x}^{(2)}$, if both conditions 1 and 2 are true:*

1. *the solution $\mathbf{x}^{(1)}$ is not worse than $\mathbf{x}^{(2)}$ with respect to all objectives;*
2. *the solution $\mathbf{x}^{(1)}$ is strictly better than $\mathbf{x}^{(2)}$ with respect to at least one objective.*

The non-dominated set of the entire feasible search space is the Pareto-optimal set and its image in objective space is called the Pareto front. A typical multiobjective algorithm sorts individuals with respect to dominance and aims at finding a well spread sample of the Pareto front.

The success of Pareto-based multi-objective evolutionary algorithms (MOEA, see e.g. [7, 5]) suggested to add the diversity of the population as an objective on its own in a multiobjective optimization. A MOEA can then be employed to obtain the set of all Pareto-optimal trade-offs between original candidate solutions and efficient ones. Several recent papers deal with this idea in diverse contexts. In genetic programming, [6] added the mean distance to the rest of the population as a second objective for the problems of 3, 4 and 5-parity. [28] adopted a similar method by using the sum of Euclidean distance between each individual and the others; they also proposed to use only the distance to the closest individual. [1] and [2] analyzed numerous ways to add an objective to improve diversity and compared them to single objective mechanisms: the generation of appearance of an individual (to be minimized), a random value, the opposite of the main fitness, the distance of the nearest neighbor in the genotype space, the mean distance to individuals of the current population and the distance to the best individuals of the current population. They concluded that the best results were obtained using a multiobjective evolutionary algorithm combined with the distance to the closest neighbors or the mean distance to the whole population. A direct consequence of their results is that modern diversity preservation mechanisms seem to have much to gain by using multiobjective algorithms.

Such addition of new objectives to help the optimization process is sometimes termed a *multiobjectivization* [19, 16].

In each of the considered diversity maintenance approaches, single or multiobjective, N^2 distances must be computed at each generation for a population of N individuals. While this computational cost may appear negligible when solutions are represented by real-valued vectors or binary strings, it may easily rise when more complex genotypes are used. In particular, the edit distance between two graphs is NP-hard [3] and consequently not suitable to be used in a diversity maintenance mechanism. Nonetheless, the numerous applications in pattern matching lead to many heuristics to approximate a measure of the similarity between two graphs. Among recently published results (see [3] for an overview of classical methods), [21] proposed a fast algorithm based on the *graph probing* paradigm. According to this paradigm, "probes" are functions that examine the vertex and edge structure of a graph by counting in- and out- degrees. The results of such probes for each graph are then compared. A theoretical analysis [21] demonstrates that graph probing provides a lower bound on the true edit distance between two graphs. Empirical results showed a good correlation with the latter measure. The speed of this approach makes it suitable to be used in a evolutionary context, as did [26] for a pattern recognition task. In this paper, we will investigate its use to maintain the diversity in a population of neural networks.

The competing conventions are another problem when trying to compute the distance between two neural networks. Two given neural networks may only differ by the order of appearance of their hidden units and therefore be topologically equal while having a different genotype. This makes the design of a useful genotype-based distance very difficult and often computationally expensive.

NEAT [27], one of the most successful method to evolve neural networks, provides an interesting approach to bypass both the computational cost of graph distance and the competing convention problems. NEAT starts with only one topology for all the individuals and, during the evolutionary process, attributes a unique and global *innovation number* to new connections and new nodes. To compute the similarity between two genotypes, genes with the same innovation numbers are lined up and those that do not match are counted. A linear combination of the number of unmatched genes and weights differences results in a scalar measuring the similarity between two genotypes. Last, NEAT uses this measure to modify the raw fitness by fitness sharing. Despite the good results obtained by NEAT, one should note that this similarity measure requires to start the evolutionary process using only one topology and that it strongly depends on the used genotype.

3. APPROACH

3.1 Behavioral Diversity

An infinity of neural networks can return the same output for a given input data set. For instance, all the neural networks in which there exists no path between the inputs and the outputs have the same null output. Whatever their

topology and their parameters are, these candidate solutions will obtain the same fitness. More generally, in most situations, rising the diversity of neural networks can be achieved by adding useless neurons and connections. This simple strategy slows down the simulation of neural networks but has no reason to enhance the process.

An alternative and more original view is to maintain the diversity of *behaviors*. All the neural networks with unconnected outputs have the same behavior – a null output; similarly, adding a neuron in a fully saturated network often has no effect on the overall behavior. While encouraging new topologies may not improve the process, we hypothesize here that allowing the selection of new behaviors, even if they are inefficient, can bring better results.

Let's suppose we have a distance $d_b(x, y)$ between the behavior of x and y. Following the conclusions published by [2], we propose to define the originality $B(x)$ of the behavior of individual x as the mean distance between x and the rest of the population. Maximizing $B(x)$ lead naturally to the maximization of the diversity of behaviors. As suggested in [6], [28] and [2], this objective can then be added to the set of performance objectives to optimize all of them simultaneously with a Pareto-based MOEA. The expected results is, at each generation, an approximation of the Pareto-optimal trade-offs between performance and originality. Hence, instead of maximizing a set of performance objectives $F_1(x), F_2(x), ..., F_k(x)$, we maximize the following objectives:

$$Maximize : \begin{cases} F_1(x) \\ \vdots \\ F_k(x) \\ B(x) = \frac{1}{|P_n|} \sum_{j \in P_n} d_b(x, j) \end{cases}$$

where P_n denotes the population at the current generation n and j an individual of P_n.

We now have to design a distance between behaviors. Since interesting behaviors are trivially application-dependent, this distance should be user-defined. While this may appear as a considerable challenge, all the experiments we conducted, such as the results presented in this paper, showed that surprisingly simple and even naive distances substantially improved the evolutionary process.

For instance, when evolving neural networks to control a mobile robot interacting with some objects, a vector $\mathbf{v}^{(x)}$ can contain the position of objects at the end of the experiment; if a robot has to explore a maze, its final position can also be stored in such a real-valued vector; if it has to reach some zones in an arena, such as feeding zones, $\mathbf{v}_k^{(x)}$ can take the value 1 if zone k has been reached and 0 otherwise. If neural networks have to perform a pattern classification task, a similar vector with a size equal to the number of training samples can track the output of the neural network for each case. The distance $d_b(x, y)$ can then be defined as the Euclidean distance between $\mathbf{v}^{(x)}$ and $\mathbf{v}^{(y)}$:

$$d_b(x, y) = \left|\left|\mathbf{v}^{(y)} - \mathbf{v}^{(x)}\right|\right|$$

Many other distances between behaviors can be designed, even if the evolved genotype does not represent a neural network. The only constraint is to be able to define a concept of behavior in the handled problem. Nonetheless, not every MOEA can be employed due to the dynamic nature of $B(x)$. For each individual i, $B(x)$ depends on the behavior of the rest of the population. As a consequence, $B(x)$ will change at each generation and a non-dominated individual thanks to a high $B(x)$ value could be dominated at the next generation because of a lower $B(x)$ value. This dynamic objective is compatible with MOEAs that compute the non-dominated set of solutions at each generation such as NSGA-II [8] or MOGA [11]. However, archive-based MOEAs, for instance ε-MOEA [9], need to be slightly modified to recreate the archive at each generation (this does not require any re-evalution of the fitness function).

3.2 Behavioral Novelty

In a recent work about open-endness in evolutionary computation, [20] proposed to maximize the "novelty" of behaviors *instead of* the fitness. They then argued that this process may be more efficient to find good solutions for a given objective – although unknown to the evolutionary process – by overcoming the deceptiveness of the fitness function. The authors computed the novelty of an individual using a distance between behaviors, similar to the one discussed in the present paper. However, contrary to the behavioral diversity defined in the previous section, novelty takes into account the set of *all behaviors previously encountered* (including the current population) and not only the current population. More precisely, [20] measured the novelty $\rho(x)$ of an individual i by computing the mean behavioral distance between i and its k nearest neighbors:

$$\rho(x) = \frac{1}{k} \sum_{j=0}^{k} d_b(x, \mu_j)$$

where k is a user-defined parameter and μ_j is the j-th nearest neighbor of x with respect to the distance d_b in the archive of behaviors. The neighbors are computed using the current population and an *archive* of all the previous novel individuals. An individual is added to the archive if its novelty is above some minimal threshold.

Extending the work of [20] to a multiobjective context, this archive-based novelty measure can be used as a second objective in a multiobjective optimization. Hence, an individual will be non-dominated if and only if it represents a Pareto-optimal trade-off between novelty and efficiency. However, by relying on all the previously explored individuals *and* the current population, this approach mixes behavioral diversity – related to the current population – and behavioral novelty – related to the archive. To estimate the importance of novelty with regards to diversity, we propose to build an archive of behaviors and to use as an additional objective the mean distance to the nearest archived behaviors, the current population being ignored. More formally, using behavioral novelty consists in transforming the multiobjective maximization problem $F_1(x), F_2(x), ..., F_k(x)$ to:

$$Maximize : \begin{cases} F_1(x) \\ \vdots \\ F_k(x) \\ \rho(x) \end{cases}$$

Compared to mechanisms based only on the current population, this kind of archive-based objective has a substantial computational cost because the archive may easily be larger than current population. Finding the nearest neighbors in such a large set can therefore become costly.

0000	0001	0010	0011	0100	0101	0110	0111
0	0	0	0	0	1	1	0
1000	1001	1010	1011	1100	1101	1110	1111
0	1	1	0	0	0	0	0

Table 1: Truth table of the function $\big[(a \oplus b) \wedge (c \oplus d)\big]$, where a, b, c et d are Boolean values and $\oplus$ the exclusive or (xor). 75% of outputs are "false".

3.3 Probe-based Graph Diversity

To further understand the advantages of the described behavioral objectives over genotypic or phenotypic diversity, we implemented a diversity objective based on the graph-probing distance measure [21]. Let G_1 and G_2 denotes two neural networks and V_G the set of vertices of network G. For each network, we can count vertices with in-degree i and collect the results in a vector I_G such that

$$\mathbf{I_G} = (n_0, n_1, ..., n_k) \text{ where } n_i = \Big|\{v \in V_G | indeg(v) = i\}\Big|$$

where k denotes the maximum number of input connections of a neuron in our typical neural networks (e.g. 15). A similar vector can collect the out-degrees:

$$\mathbf{O_G} = (m_0, m_1, ..., m_k) \text{ where } n_i = \Big|\{v \in V_G | outdeg(v) = i\}\Big|$$

A network G can be described by the vector $\mathbf{P_G}$:

$$\mathbf{P_G} = (n_0, n_1, ..., n_k, m_0, m_1, ..., m_k)$$

Last, $d_p(G_1, G_2)$ is defined as the Euclidean distance between $\mathbf{P_{G_1}}$ and $\mathbf{P_{G_2}}$ and the graph diversity objective consists in maximizing the mean distance between the considered individual and the rest of the population:

$$\text{Maximize} \begin{cases} F_1(x) \\ \vdots \\ F_k(x) \\ D(x) = \frac{1}{|P_n|} \sum_{j \in P_n} d_p(x, j) \end{cases}$$

where:

$$d_p(x) = \frac{1}{N} \sum_{j=1}^{N} \Big|\Big|\mathbf{P_{G_i}} - \mathbf{P_{G_j}}\Big|\Big|$$

4. EXPERIMENTS

4.1 Genotype and Fitness

To benchmark these different approaches of exploration enhancement, we evolved neural-networks to compute the Boolean function $\big[(a \oplus b) \wedge (c \oplus d)\big]$, where a, b, c and d are Boolean values and $\oplus$ denotes the exclusive "or" operator (xor). The truth table of $\big[(a \oplus b) \wedge (c \oplus d)\big]$ (table 1) shows that a simple neural network that returns "false" for any input would have a 75% success rate, a good score for this task, especially at the beginning of the evolutionary process. As a consequence, these degenerated neural networks can easily invade the population whereas they do not constitute a good starting point. This makes the typical single-objective fitness for this function – the sum of errors – very deceptive. This has been empirically illustrated in [18]: using an elitist evolutionary algorithm, a population of 1000 individuals and a graph-based direct encoding for NAND gates, [18] reports that only 72% of experiments found a solution in less than 10^5 generations; in other words, more than a quarter of their experiments were trapped in an inescapable local optima despite the large number of individuals and generations.

To keep our experiments simple and repeatable, we employed a typical graph-based direct encoding for neural-networks in which two kinds of mutations are possible:

- structural mutation: addition/removal of a random neuron or a random connection;
- parametric mutation: change of a randomly chosen synaptic weight or a neuronal bias; we used here a change in a set of 9 possible values (see appendix).

Cross-over was not employed.

We used the classical sum of errors for each possible set of inputs as the main fitness for an individual x (expressed in a fitness maximization scheme):

$$F_{xax}(x) = 1 - \frac{1}{16} \sum_{i=1}^{16} |o_{x,i} - d_i|$$

where $o_{x,i}$ is the output of the neural network x for the input set i and d_i the desired output. Each neural network is simulated during 100 time-steps. Since we do not constrain the structure of the neural networks, nothing prevents them from oscillating. To avoid such behaviors, we attribute an arbitrary low fitness if the output is not constant during the last 10 time-steps.

4.2 Behavior Vectors

To describe behaviors, each individual x is associated to a Boolean *behavior vector* $\mathbf{v^{(x)}}$ that contains the output of the neural network for each of the 16 different input sets:

$$\mathbf{v}_i^{(x)} = nn_x(b_i), i = 1, 2, ..., 16; b_i \in \{0000, 0001, ..., 1111\}$$

The distance between behaviors is then straightforward:

$$d_b(x, y) = \big|\big|\mathbf{v}^{(y)} - \mathbf{v}^{(x)}\big|\big|$$

To implement behavioral novelty, an archive A_n stores all the behavior vectors $\mathbf{v}^{(x)}$ present in the populations $P_0, ..., P_n$, where n denotes the generation number:

$$A_n(\mathbf{v}) = \begin{cases} \mathbf{v} \text{ if } \mathbf{v} \in P_0, P_1, ..., P_n \\ \emptyset \text{ otherwise} \end{cases}$$

The novelty score is equal to the mean distance to the $k = 10$ nearest behaviors in A_n, where k was experimentally chosen using our experiments and those of [20].

4.3 Experiments

We launched 5 sets of 16 experiments; using the previously defined notations, for an individual x:

1. *Control experiment.* Maximize $F_{xax}(x)$
2. *Behavioral diversity.*
$$\text{Maximize} \begin{cases} F_{xax}(x) \\ B(x) = \frac{1}{|P_n|} \sum_{j \in P_n} d_b(x, j) \end{cases}$$
3. *Behavioral novelty.*
$$\text{Maximize} \begin{cases} F_{xax}(x) \\ B(x) = \frac{1}{k} \sum_{j=0}^{k} d_b(x, \mu_j) \end{cases}$$

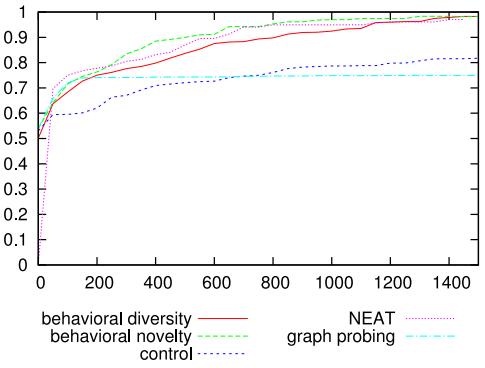

Figure 1: Mean maximum fitness (over 16 runs of each experiment), with respect to generation number. Behavioral diversity, behavioral novelty and NEAT reached a value close to the maximum (1) in less than 1500 generations whereas the control experiment and the graph probing approach did not exceed 0.75 (the attractive local maximum).

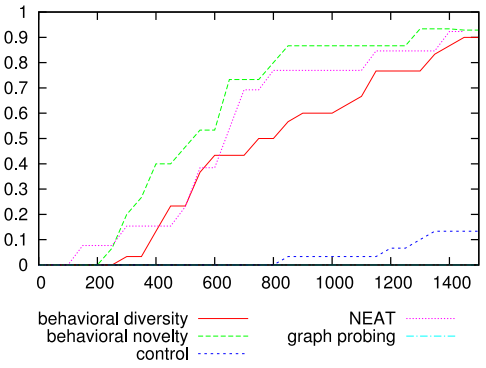

Figure 2: Convergence rate (over 16 runs of each experiment), with respect to generation number. 90% of the runs corresponding to behavioral diversity, behavioral novelty and NEAT converged in less than 1500 generations whereas only a few control runs converged. No run converged in the graph probing experiments.

where μ_j is the j-th nearest neighbors in the archive A_n,

4. *Probe-based diversity.*

$$\text{Maximize} \begin{cases} F_{xax}(x) \\ D(x) = \frac{1}{|P_n|} \sum_{j \in P_n} d_p(x, j) \end{cases}$$

5. *NEAT.* Maximize $F_{xax}(x)$ using NEAT [27].

With the exception of the NEAT experiment, in which we employed the NEAT algorithm, each set of objectives has been maximized using NSGA-II [8], one of the most efficient MOEAs [7]. It should be emphasized that all these experiments (except NEAT) use a standard algorithm and a classic genotype; repeating them should be easy.

The detailed parameters and the url to download the source are available in appendix.

5. RESULTS

5.1 Maximum Fitness and Convergence Rates

Figure 2 shows the mean maximum fitness with respect to generation for each set of experiments. The best fitnesses are obtained by behavioral diversity, behavioral novelty and NEAT, with a mean maximum fitness of more than 0.95 after 1500 generations. The control experiment and probe-based diversity led to significantly worse results, with a mean fitness around 0.75, i.e. the best networks are not better than a network that returns "false" for any input.

These results are confirmed by analyzing the convergence rates (figure 2): while 90% of the experiments made with behavioral diversity, behavioral diversity and NEAT converged (the fitness of the best individual was more than 0.95) in less than 1500 generations, only a few (12%) of the control experiments did so. Probe-based diversity led to worse results than the control experiment, no run having converged after 1500 generations.

We then analyzed the mean convergence generation for behavioral diversity, behavioral novelty and NEAT (table 2).

	B. div	B. nov.	NEAT
Mean gen.	921.7	734.6	757.3
Std. dev.	463.1	494.7	499.4
T-test div.	p=1.0	p=0.2401	p=0.3025
T-test nov.	p=0.2401	p=1.0	p=0.9083
T-test NEAT	p=0.3025	p=0.9083	p=1.0

Table 2: Mean convergence generation (maximum $F_{xax} > 0.95$) and Student T-test. Behavioral diversity seems to require more generations to converge but the difference is not statistically significant.

Behavioral novelty and NEAT required slightly fewer generations to converge (about 750 versus about 900) than behavioral diversity but a Student T-test reveals that this difference is not statistically significant.

5.2 Behavioral Analysis

To draw a picture of the behavioral diversity process in an experiment, we introduce here an original type of diagram, called *behavioral diversity diagram* (figures 3, 4, 5 and 6). For each generation, we counted the proportion of individuals that returned the correct output for each of the 16 input sets. On the diagrams, this proportion is reported as the gray level of a small vertical stripe for each input set, aligned with the generation number. In a population with diverse behaviors, we expect a diagram almost uniformly gray because all behaviors should be present in the population, at any generation. Contrarily, in a population with uniform behaviors, which probably lacks diversity, we expect to see large horizontal white stripes (behaviors shared by 100% of the population during a significant amount of generations) and black horizontal stripes (behaviors absent of the population during many generations). Last, the fitness of the best individual is plotted with respect to the generation number in order to see how best fitnesses are related to behaviors.

Unsurprisingly, the diagrams corresponding to the control experiments (figure 3) shows such black and white stripes. In the run analyzed on figure 3, more than 90% of the pop-

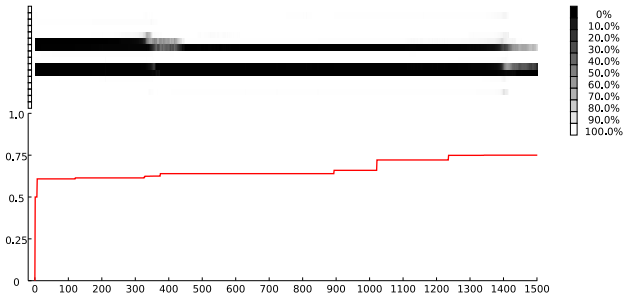

Figure 3: Behavioral diversity diagram for a typical run of the control experiment. (top) Ratio of the population that returns the desired output for each of the 16 data sets, with respect to generation number. During the first 300 generations, almost all the population had the same behavior: returning "false" for each data set and consequently having the right answer 12 times over 16. This is illustrated by the black stripes. (bottom) Fitness of the best individual with respect to generation number. It does not exceed 0.75.

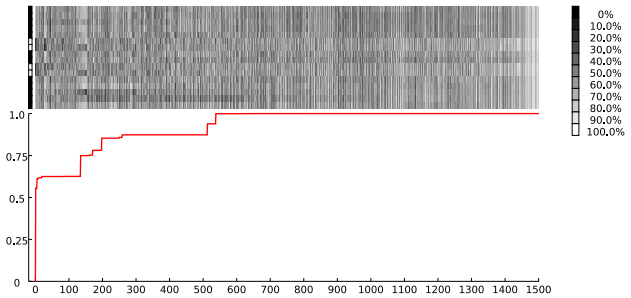

Figure 4: Behavioral diversity diagram for a typical run of behavioral diversity. No behavior seems to be shared by more than half of the population as no white zone can be observed. Similarly, no behavior seems absent from the population as there are no black zones. The alternation of light and dark grays denotes small (one to a few generations) cycles in which a behavior is first original, therefore kept in the next generation, then less original because of the offspring of the kept individual. (bottom) Fitness of the best individual with respect to generation number. The optimum (1.0) is reached after about 500 generations.

ulation returned "false" for all the possible inputs for about 300 generations; this is translated as long white stripes for the 12 input sets that should lead to "false" and four black stripes for the 4 ones that should lead to "true". In this run, a novel behavior appeared at generation 300 and quickly invaded the population. All the runs of the control experiments we analyzed led to similar diagrams, confirming the diagnosis of a lack of diversity.

These black and white stripes contrast with the uniformity and the grayness of figure 4, which shows the diagram corresponding to a typical experiment using behavioral diversity. All the behaviors are present in the population, and only short cycles – displayed as changes between dark and clear gray levels – can be distinguished.

The behavioral novelty approach led to different and more complex patterns (figure 5) in which diversity episodes succeed to uniform behaviors. These diversity episodes were correlated with rapid increases of the archive size. The diversity-uniformity cycles reveal that once in a while a new "invention" gives birth to many new behaviors; they are then selected and only the ones with the best fitnesses are kept. After the best individual reached the maximum fitness, the only selective pressure is towards novelty. This results in a high diversity episode, which fills the archive. Once the archive is almost full, all the population converged to the same behavior, the optimal one; this is illustrated by a large white zone on the diagram.

While behavioral novelty adds a selective advantage to new behaviors when they appear, this advantage is not maintained for many generations because the behaviors are not new anymore. This results in a situation in which a novel individual has an influence on the evolutionary dynamics only when it has almost no neighbors. Once one individual with each of the close behavior has been generated, nothing prevents the evolutionary algorithm from premature convergence to a small subset of behaviors. Taking into account the current population when computing the nearest neighbors, as did Lehman and Stanley [20], could have added a selective pressure towards diversity. However, since the size of the archive is large compared to the population size, this pressure will decrease during the evolutionary process.

Last[1], the diagram corresponding to probe-based diversity (figure 6) shows that this mechanism maintains more diversity than the control experiment. It is therefore surprising that it did not improve the convergence rate and even decreased it. This issue should be investigated in future work.

6. DISCUSSION AND CONCLUSION

The experiments reported in this paper show at least that:

- without any diversity preservation mechanism, the diversity of behaviors is very low (figure 3);
- behavioral diversity, behavioral novelty and NEAT are three efficient solutions to override the deceptiveness of a fitness when evolving neural-networks (figure 2); the difference between them was not statistically significant in these experiments (table 2) but the observed tendency suggests that NEAT and behavioral novelty might converge faster than behavioral diversity;
- the phenotype-based approach to diversity investigated did not improve the convergence rate and even deteriorated it.

These results add weights to the previous experiments that report performance enhancements with multiobjective diversity mechanisms [6, 1, 28, 2]. They also demonstrate that maintaining the diversity of the *behaviors* of candidate

[1]The diagram corresponding to NEAT was not generated for technical reasons.

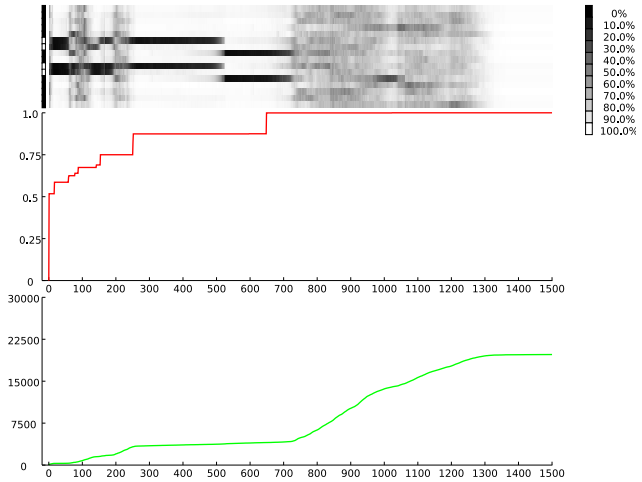

Figure 5: Behavioral diversity diagram for a typical run of behavioral novelty. (top) Episodes of high diversity succeed to low diversity ones. (middle) Fitness of the best individual with respect to generation number. A fitness increase often precedes diversity increase. (bottom) Size of the archive. Each high diversity episode results in an increase of the archive size.

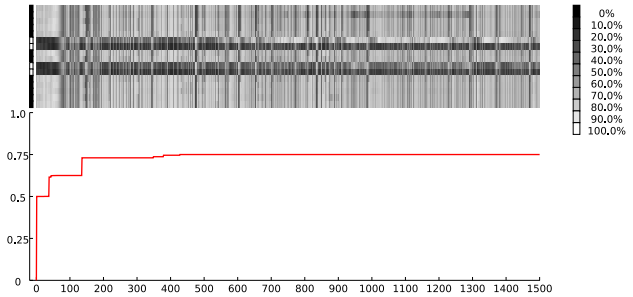

Figure 6: Behavioral diversity diagram for a typical run of the graph probing experiment. While the gray levels are not as uniform as in figure 4, this figure shows that many different behaviors are present in the population. (bottom) Fitness of the best individual with respect to generation number. Despite the diversity of behaviors, the best fitness never exceeds 0.75**.**

solutions instead of preserving the diversity of their *genotype or phenotype* is a simple and efficient approach. Algorithmically and conceptually simple distances can be designed while being independent of the encoding. Moreover, no special evolutionary process is required; classic and efficient MOEA can be employed.

The computational cost of behavioral novelty over behavioral diversity does not seem justified in these experiments but some other benchmarks problems might lead to more differences between these two methods. Moreover, their combination may be investigated in future work: while keeping diverse behaviors can be done using a diversity objective, a selective pressure to encourage new behaviors could be added as a third objective. However, adding many objectives in addition to the main ones can decrease the efficiency of the evolutionary process, as reported in many papers [24, 25].

The efficiency of NEAT could suggest the conclusion that the presented method is not an improvement over the literature. However, at least two points should be taken into account. First, by starting with a fully-connected feed-forward neural network and disabling recurrent connections, NEAT constrains the search process to feed-forward networks. This is an intuitively useful bias in the considered experiment that was not included in the other methods. Second, implementing NEAT is more complex than using a direct encoding, notably because the encoding scheme is tightly linked to the evolutionary algorithm. Fitness sharing in NEAT, i.e. the modified evolutionary process, requires innovation numbers, which are deeply integrated in the encoding. At the opposite, behavioral methods can be used with any encoding and do not require any specific evolutionary algorithm. They can even be used with other phenotypes than neural networks such as Bayesian networks or fuzzy rules based systems. Last, NEAT is based on the assumption that the experimenter is able to provide a starting topology, an assumption that may not be valid in every context.

As a general conclusion, behavioral diversity and novelty appear to be simple and efficient methods to improve neuro-evolution when facing a deceptive problem, at least when simple behavioral distance can be designed. These results highlight that although considerable efforts have been put in designing neural networks encodings, working on the selective pressures can also be fruitful while being simpler. Future work should extend the behavioral diversity concept to more general contexts than neuro-evolution. For instance, the diversity of robots' behaviors can be maintained using a similar approach to the one presented here [23].

7. REFERENCES

[1] H. A. Abbass and K. Deb. Searching under multi-evolutionary pressures. *Proceedings of the Fourth Conference on Evolutionary Multi-Criterion Optimization*, 2003.

[2] L. Bui, H. A. Abbass, and J. Branke. Multiobjective optimization for dynamic environments. *The 2005 IEEE Congress on Evolutionary Computation (CEC)*, 3:2349–2356 Vol. 3, 2005.

[3] H. Bunke. Recent developments in graph matching. *Proc. 15th International Conference on Pattern Recognition*, pages 117–124, 2000.

[4] E. Burke, S. Gustafson, and G. Kendall. A survey and analysis of diversity measures in genetic programming. *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO'02)*, pages 716–723, 2002.

[5] C. A. Coello Coello and G. B. Lamont. *Applications Of Multi-Objective Evolutionary Algorithms*. World Scientific, 2004.

[6] E. D. de Jong, R. A. Watson, and J. B. Pollack. Reducing bloat and promoting diversity using multi-objective methods. *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO'01)*, pages 11–18, 2001.

[7] K. Deb. *Multi-objectives optimization using evolutionnary algorithms*. Wiley, 2001.

[8] K. Deb, S. Agrawal, A. Pratab, and T. Meyarivan. A Fast Elitist Non-Dominated Sorting Genetic Algorithm for Multi-Objective Optimization: NSGA-II. In *Proceedings of the Parallel Problem Solving from Nature VI Conference*, pages 849–858, 2000.

[9] K. Deb, M. Mohan, and S. Mishra. Evaluating the ε-Domination Based Multi-Objective Evolutionary Algorithm for a Quick Computation of Pareto-Optimal Solutions. *Evolutionary Computation*, 13(4):501–525, 2005.

[10] S. Doncieux and J.-A. Meyer. Evolving PID-like neurocontrollers for non-linear control problems. *International Journal of Control and Intelligent Systems (IJCIS). Special Issue on nonlinear adaptive PID control*, 33(1):55–62, 2005.

[11] C. M. Fonseca and P. J. Fleming. Genetic algorithms for multiobjective optimization : formulation, discussion and generalization. In *Proceedings of the Fourth International Conference on Evolutionary Programming*, pages 416–423, 1993.

[12] D. E. Goldberg. Simple genetic algorithms and the minimal, deceptive problem. *Genetic Algorithms and Simulated Annealing*, 74, 1987.

[13] D. E. Goldberg, K. Deb, and J. Horn. Massive multimodality, deception, and genetic algorithms. In *Proceedings of the Second Conference on Parallel Problem Solving from Nature*, pages 37–48, 1992.

[14] D. E. Goldberg and J. Richardson. Genetic algorithms with sharing for multimodal function optimization. In *Proceedings of the Second International Conference on Genetic Algorithms*, pages 148–154, 1987.

[15] F. Gruau. Automatic definition of modular neural networks. *Adaptive Behaviour*, 3(2):151–183, 1995.

[16] J. Handl, S. C. Lovell, and J. Knowles. Multiobjectivization by decomposition of scalar cost functions. In *Proceedings of the 10th international conference on Parallel Problem Solving from Nature: PPSN X*, pages 31–40. Springer, 2008.

[17] J. H. Holland. *Adaptation in Natural and Artificial Systems*. MI: University of Michigan Press, 1975.

[18] N. Kashtan and U. Alon. Spontaneous evolution of modularity and network motifs. *Proceedings of the National Academy of Sciences*, 102(39):13773–13778, 2005.

[19] J. D. Knowles, R. A. Watson, and D. W. Corne. Reducing Local Optima in Single-Objective Problems by Multi-objectivization. *First International Conference on Evolutionary Multi-Criterion Optimization*, 1993:268–282, 2001.

[20] J. Lehman and K. Stanley. Exploiting open-endedness to solve problems through the search for novelty. In *Artificial Life XI: Proceedings of the Eleventh International Conference on the Simulation and Synthesis of Living Systems*, pages 329–336, 2008.

[21] D. Lopresti and G. Wilfong. A fast technique for comparing graph representations with applications to performance evaluation. *International Journal on Document Analysis and Recognition*, 6(4):219–229, 2003.

[22] J.-B. Mouret and S. Doncieux. Mennag: a modular, regular and hierarchical encoding for neural-networks based on attribute grammars. *Evolutionary Intelligence*, 1:187–207, 2008.

[23] J.-B. Mouret and S. Doncieux. Overcoming the bootstrap problem in evolutionary robotics using behavioral diversity. In *IEEE Congress Evolutionary Computation (CEC).*, page to appear, 2009.

[24] K. Praditwong and X. Yao. How well do multi-objective evolutionary algorithms scale to large problems. In *IEEE Congress on Evolutionary Computation (CEC).*, pages 3959–3966, 2007.

[25] R. C. Purshouse and P. J. Fleming. On the Evolutionary Optimization of Many Conflicting Objectives. *IEEE Transactions on Evolutionary Computation*, 11(6):770–784, 2007.

[26] R. Reveaux, B. Eugen, H. Locteau, S. Adam, P. Heroux, and E. Trupin. A Graph Classification Approach Using a Multi-objective Genetic Algorithm Application to Symbol Recognition. In *IAPR GBR Workshop on Graph-based Representations in Pattern Recognition*, volume 4538, page 361, 2007.

[27] K. O. Stanley and R. Miikkulainen. Evolving neural networks through augmenting topologies. *Evolutionary Computation*, 10(2)(99-127), 2002.

[28] A. Toffolo and E. Benini. Genetic diversity as an objective in multi-objective evolutionary algorithms. *Evolutionary Computation*, 11(2):151–167, 2003.

[29] X. Yao. Evolving artificial neural networks. *Proceedings of the IEEE*, 87(9):1423–1447, 1999.

APPENDIX

A. PARAMETERS: DIRECT ENCODING

- MOEA: NSGA-II (pop. size : 400)
- Neural network (direct encoding):
  - available weights / bias: $\{-2.0, -1.5, -1.0, -0.5, 0.0, 0.5, 1.0, 1.5, 2.0\}$
  - proba. of weight/bias change (for each value): 0.1
  - number of inputs / outputs: 4/1
  - min./max. neurons (random gen.): 10/20
  - min./max. connections (random gen.): 20
  - proba. of adding/deleting a conn.: 0.15/0.25
  - proba. of changing a conn.: 0.1
  - proba. of adding/deleting a neuron: 0.025/0.025
  - activation function for neurons: $y_i = \varphi\left(\sum_j w_{ij} x_j\right)$ where $\varphi(x) = \frac{1}{1+\exp(b-kx)}$
- Source code : `http://webia.lip6.fr/~mouret/papers/sferes2-gecco09-cec09.tgz`

B. PARAMETERS: NEAT

- Source code: "original" NEAT C++ available at `http://nn.cs.utexas.edu/?neat`
- Parameters:
  - population size: 400
  - starting topology: feed-forward neural network in which each input is connected to the output (4 inputs, 1 bias input, 1 output)
  - configuration files: `http://webia.lip6.fr/~mouret/papers/neat-gecco09.tgz`