Encouraging Behavioral Diversity in Evolutionary Robotics: an Empirical Study

Jean-Baptiste Mouret, Stéphane Doncieux

Encouraging Behavioral Diversity in Evolutionary Robotics: an Empirical Study

J.-B. Mouret jean-baptiste.mouret@upmc.fr
ISIR, Université Pierre et Marie Curie-Paris 6, CNRS UMR 7222, Paris, F-75252, Paris Cedex 05, France

S. Doncieux stephane.doncieux@upmc.fr
ISIR, Université Pierre et Marie Curie-Paris 6, CNRS UMR 7222, Paris, F-75252, Paris Cedex 05, France

Preprint – Accepted for publication Evolutionary Computation (2011)

Abstract

Evolutionary Robotics (ER) aims at automatically designing robots or controllers of robots without having to describe their inner workings. To reach this goal, ER researchers primarily employ phenotypes that can lead to an infinite number of robot behaviors and fitness functions that only reward the achievement of the task—and not how to achieve it. These choices make ER particularly prone to premature convergence.

To tackle this problem, several papers recently proposed to explicitly encourage the diversity of the robot *behaviors*, rather than the diversity of the genotypes as in classic evolutionary optimization. Such an approach avoids the need to compute distances between structures and the pitfalls of the non-injectivity of the phenotype/behavior relation; however, it also introduces new questions: how to compare behavior? should this comparison be task-specific? and what is the best way to encourage diversity in this context?

In this article, we review the main published approaches to behavioral diversity and benchmark them in a common framework. We compare each approach on three different tasks and two different genotypes. Results show that fostering behavioral diversity substantially improves the evolutionary process in the investigated experiments, regardless of genotype or task. Among the benchmarked approaches, multi-objective methods were the most efficient and the generic, Hamming-based, behavioral distance was at least as efficient as task-specific behavioral metrics.

Keywords

evolutionary robotics; diversity; multi-objective evolutionary algorithm; neural networks.

1 Introduction

Before being recognized as efficient optimization methods, the first evolutionary algorithms stemmed from a desire to transfer the richness and the efficiency of living organisms to artifacts and, in particular, to artificial agents (Fogel et al., 1966; Schwefel, 1984; Holland, 1992). This envisioned future inspired a whole field of research in which evolutionary algorithms (EAs) are employed to automatically design robots or controllers of robots; this field is now called *Evolutionary Robotics* (ER) (Meyer et al., 1998; Nolfi and Floreano, 2004; Lipson, 2005; Doncieux et al., 2009). According to ER

researchers, EAs are a promising candidate to tackle the automatic design challenge because of their ability to act as "blind watchmakers": they only care about the result and do not constrain the working principles of the solutions; consequently, an EA could design a robot based on unusual properties that don't fit the classic engineering methods. This reasoning is perfectly illustrated by the Golem project (Lipson and Pollack, 2000), in which Lipson and Pollack let the structure of a robot and its neuro-controller evolve, while selecting robots for their speed of translation. By an evolutionary process, these authors obtained original morphologies and simple but efficient gaits. In another typical study, Doncieux and Meyer evolved the neural network that controlled a lenticular blimp (Doncieux and Meyer, 2003) and the EA found an original behavior that exploited the shape of the blimp to create additional lift. In these papers, as in many others (e.g. Floreano and Mondada (1994); Kodjabachian and Meyer (1997); Hornby and Pollack (2002); Walker et al. (2003); Mouret et al. (2006)), EAs were interesting not only because they led to efficient solutions but also because these solutions were not biased by human considerations. They were therefore at least unexpected, if not innovative.

To find these original solutions, EAs employed in ER typically rely on genotypes that can encode an infinity of different solutions. For instance, researchers in ER often evolve the structure and the parameters of neural controllers, because this theoretically allows an infinite number of robot behaviors. The genotype is then developed into a phenotype, for instance a neural network, which is tested in different situations. The behavior[1] of the robot is observed during these tests and the fitness function reflects how well the targeted task has been achieved. To limit human biases, the typical fitness functions reward only the accomplishment of the task and provide as few hints as possible about *how* to achieve this goal. These two choices, an open infinite space and high-level fitness functions, makes ER experiments particularly prone to the bootstrap problem (Mouret and Doncieux, 2009a) and to premature convergence (Goldberg, 1987; Bongard and Hornby, 2010; Mouret and Doncieux, 2009b). In both cases, any EA practitioner will diagnose a lack of exploration and an over-exploitation of current best candidate solutions.

The most popular approach to favor exploration over exploitation in EAs is undoubtedly to foster the diversity of the population: For many problems, diversity mechanisms improved the efficiency of EAs both empirically (Goldberg, 1987; Sareni and Krahenbuhl, 1998; Mahfoud, 1997) and theoretically (Friedrich et al., 2008). In the typical setup, the user first defines a distance between solutions, for example a Hamming distance if the genotype is a bit-string. The algorithm then modifies the fitness of each individual with regard to how close it is from the other members of the population. Several fitness modifications have been proposed, such as fitness sharing (Goldberg, 1987; Sareni and Krahenbuhl, 1998), crowding (Mahfoud, 1997) or, more recently, multi-objective optimization (De Jong et al., 2001; Toffolo and Benini, 2003; Bui et al., 2005; Mouret and Doncieux, 2009a,b; Doncieux and Mouret, 2010).

In light of the potential improvements in exploration, it seems natural to include such diversity mechanisms in ER; but defining a distance between candidate solutions is more challenging than it appears. Let's first assume that, as in many ER studies, we evolve a neural network to control a robot. If we don't evolve the structure, that is we are in the simplest case, computing a distance between two genotypes faces the "competing convention problem": two neural networks may only differ by the order of

[1]Some authors consider that the behavior is part of the phenotype. We will here separate the behavior from the phenotype to make our description of behavioral diversity as clear as possible.

appearance of their hidden units and therefore be topologically and functionally identical, while having a different genotype. If the structure of the neural network is also evolved, we have to compute the distance between two graphs. Since this computation is NP-hard (Bunke and Shearer, 1998), it is unrealistic to use such a distance in a diversity preservation mechanism. From these arguments, maintaining diversity when evolving neural networks is challenging because comparing neural networks is difficult. The same conclusion can be drawn when the morphology of robots is evolved, since it also involves the comparison of structures.

An alternative line of thought emerged during the last few years: whatever is evolved, the goal in evolutionary robotics is ultimately to find a *behavior* (Lehman and Stanley, 2008; Trujillo et al., 2008a,b; Mouret and Doncieux, 2009a,b; Mouret, 2011; Gomez, 2009; Moriguchi and Honiden, 2010; Lehman and Stanley, 2010; Doncieux and Mouret, 2010). This behavior results from the interaction of the robot with its environment, and is thus influenced by the robot controller—may it be a neural network or anything else—and its morphology—may it be evolved or not. By comparing behaviors instead of genotypes or phenotypes, the previously described problems of comparing structures disappear and the diversity of behavior can be encouraged. However, this process requires defining a distance between behaviors. Several recent papers proposed to define task-specific distances (Lehman and Stanley, 2008, 2010; Trujillo et al., 2008a,b; Mouret and Doncieux, 2009a,b; Mouret, 2011; Moriguchi and Honiden, 2010); some other introduced generic distances based on the flow of perceptions and actions (Gomez, 2009; Doncieux and Mouret, 2010). Additionally, this behavioral diversity can be fostered with fitness sharing (Moriguchi and Honiden, 2010) or with a multi-objective approach (Mouret and Doncieux, 2009a,b; Mouret, 2011; Doncieux and Mouret, 2010). Whatever the specific implementation was, all these techniques showed substantial improvements of convergence rates in the particular setup chosen by the authors. Some papers even suggest that such techniques may allow tackling many ER tasks that are currently out of reach (Doncieux and Mouret, 2010).

In spite of these encouraging results, there is currently a lack of studies that compare these approaches. Overall, all these papers showed promising preliminary results but each of them has been proposed independently and, mostly, concurrently. As a consequence, these authors did not compare their proposition to others. Additionally, they only examined one task and often one diversity mechanism, each time a different one, hence preventing them from drawing any general conclusion. This raises numerous questions: *which features of behavioral diversity mechanisms are critical to improve ER? Can a generic distance be employed? What are the specific advantages and drawbacks of different techniques? And how do these advantages depend on the considered task?*

To provide first answers, the present paper reviews the main published approaches to encourage behavioral diversity and reports benchmarks to compare them. More precisely, we compare multi-objective behavioral diversity (Mouret and Doncieux, 2009a,b; Mouret, 2011; Doncieux and Mouret, 2010) and behavioral diversity encouraged via fitness sharing (Moriguchi and Honiden, 2010). In combination with these diversity mechanisms, three distances are investigated: task-specific behavioral distances (Lehman and Stanley, 2008, 2010; Trujillo et al., 2008a,b; Mouret and Doncieux, 2009a,b; Mouret, 2011), Hamming distance on the sensory-motor flow (Doncieux and Mouret, 2010; Gomez, 2009) and genotype-based distances (Goldberg, 1987; Mahfoud, 1997; Stanley and Miikkulainen, 2002). To guarantee the generality of the benchmarks, we compare each approach on two different genotypes—a direct encoding for neural networks and an Elman network whose parameters are evolved—and on three differ-

ent problems of mobile robotics—a deceptive maze (Lehman and Stanley, 2010), a light-seeking task (Mouret and Doncieux, 2009a) and a ball-collecting task (Doncieux and Mouret, 2010). Lastly, the results on the three problems are compared with NEAT (Stanley and Miikkulainen, 2002), one of the most successful neuro-evolution method. Overall 18 combinations of genotype/diversity mechanism are investigated for each of the 3 tasks.

2 Background

2.1 Fitness Sharing

Preserving or encouraging the diversity of the population has a long history in evolutionary computation, because diversity is intuitively a key to avoid premature convergence and to explore multi-modal fitness landscapes (see e.g. Mauldin (1984)).

Among the numerous approaches to diversity preservation (Mahfoud, 1997), fitness sharing is probably the most classic method. Researchers first observed that living organisms share natural resources only with those who live in the same ecological niche. As a result, organisms are selected within their own niche, without taking into account their counterparts from the other niches. To transpose this concept to evolutionary computation, Holland (1992) and Goldberg (1987) proposed that individuals share their fitness score with those who are genetically close to them. This process, named *fitness sharing*, should improve the diversity of the population: An individual vastly different from its counterparts but with a low fitness would obtain a fitness similar to those who have a better fitness but are very similar to their neighbors, and so "live" in a crowded niche.

To implement this idea, a *sharing function* $Sh(d_{ij})$ is first defined:

$$Sh(d_{ij}) = \begin{cases} 1 - (\frac{d_{ij}}{\sigma_{\text{share}}})^{\alpha}, & \text{if } d \leq \sigma_{\text{share}} \\ 0 & \text{otherwise} \end{cases} \quad (1)$$

where d_{ij} is the distance between the genotype or the phenotype of i and j and σ_{share} is a user-defined parameter. In most studies, α is set to 1 because empirical studies showed that it was a good setting (Holland, 1992; Goldberg, 1987).

The fitness of individual i is then divided by the niche count c_i, to decrease the payoff in densely populated regions of the search space:

$$F_i = \frac{F(x_i)}{c_i} \text{ where } c_i = \sum_{j=1}^{N} Sh(d_{ij}) \quad (2)$$

It has been shown empirically (Goldberg, 1987; Goldberg et al., 1992; Darwen and Yao, 1996; Mahfoud, 1997; Sareni and Krahenbuhl, 1998) that fitness sharing substantially improves the efficiency of evolutionary algorithms, in particular when the fitness function is deceptive. This performance has been recently confirmed by theoretical proofs for simple cases (Friedrich et al., 2008). The success of fitness sharing was followed by the introduction of several other niching strategies, such as crowding or clearing (Mahfoud, 1997; Sareni and Krahenbuhl, 1998). Nonetheless, none of them proved to be efficient enough to replace fitness sharing as the *de facto* standard diversity mechanism.

All these mechanisms rely on one or several user-defined parameters whose tuning is critical for their efficiency. In the case of fitness sharing, several studies highlighted how it critically depends on the choice of σ (Goldberg, 1987; Goldberg et al.,

1992; Darwen and Yao, 1996; Mahfoud, 1997; Sareni and Krahenbuhl, 1998). Setting this parameter requires *a priori* knowledge of how far apart the optima are, a knowledge that is most of the time unavailable. Moreover, the same σ is employed for every niche, which assumes that all fitness peaks are nearly equidistant in the domain. These two problems emphasize the need for a parameter-free method to maintain diversity.

2.2 Multi-objective Diversity

Dividing the fitness by the niche count in fitness sharing is an attempt to put together a performance objective, maximizing the fitness, with an exploration objective, minimizing the niche coefficient. To combine them, an alternative approach is to use a Pareto-based multi-objective evolutionary algorithm (MOEA, see e.g. Deb (2001)) with two objectives: the fitness and a measure of originality. This leads to a *multi-objectivization* (Knowles et al., 2001; Mouret, 2011) in which two objectives are maximized:

$$\text{Maximize} \begin{cases} F(x) \\ Orig(x) \end{cases} \tag{3}$$

where $F(x)$ is the fitness of individual x and $Orig(x)$ is a measure of its originality.

Numerous evolutionary algorithms have been developped to simultaneously optimize several objectives (Deb, 2001); most of them rely on the concept of Pareto dominance, defined as follows:

Definition 1 (Pareto dominance) *A solution x^* is said to dominate another solution x, if both conditions 1 and 2 are true:*

1. *the solution x^* is not worse than x with respect to all objectives;*
2. *the solution x^* is strictly better than x with respect to at least one objective.*

The non-dominated set of the entire feasible search space is the globally Pareto-optimal set (Pareto front). It represents the set of optimal trade-offs*, that is solutions that cannot be improved with respect to one objective without decreasing their score with respect to another one.*

In the particular case of multi-objective diversity, an individual will be considered Pareto-optimal, and therefore selected, if no other individual is both more original and more fit. This multi-objectivization is in effect a parameter-free method to maintain the diversity of the population.

Several papers deal with this idea in diverse contexts. In genetic programming, De Jong et al. (2001) added the mean distance to the rest of the population as a second objective for the problems of 3, 4 and 5-parity. In evolutionary computation, Toffolo and Benini (2003) 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. Abbass and Deb (2003) as well as Bui et al. (2005) 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.

3 Improving Exploration in ER

3.1 Computing a Distance Between Structures

The previously described diversity mechanisms all rely on a similarity measure between solutions. In a typical optimization context, candidate solutions are represented as list of parameters, it is then easy to compute a distance between them, for instance a Euclidean distance. Sometimes, the genetic operators do not act directly on candidate solutions but on an underlying representation. In this case, the encoding defines a genotype/phenotype map (figure 1, left) that may change the distance relations: two solutions close in the genotype space may be far away in the phenotype space, and vice versa. In such situations, it is often more informative to compute the distance in the phenotype space.

Unfortunately, distances between structures used in ER (morphology, neural controllers, ...) are hard to compute in genotype space, but also in phenotype space. Typical genotypes are directed graphs (Lipson and Pollack, 2000; Doncieux and Meyer, 2003; Mouret and Doncieux, 2009b) or trees (Gruau, 1995; Hornby and Pollack, 2002), and phenotypes are often graphs. In both cases, computing the generic similarity measure—the edit distance—is NP-hard (Zhang et al., 1992; Bunke and Shearer, 1998) whereas diversity techniques require computing many distances at each generation ($O(n^2)$ distances at each generation, where n is the size of the population). Mouret and Doncieux (2009b) investigated the use of a fast approximation of the graph edit distance, called graph probing (Lopresti and Wilfong, 2003), but this didn't lead to any fitness or convergence improvement. Furthermore, the genotype/phenotype maps in ER are usually non-injective because identical structures can be encoded in many different ways, a problem called the "competing convention problem" in the neural network literature.

The NEAT encoding (Stanley and Miikkulainen, 2002) 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 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. Last, the number of unmatched genes and the weight differences are summed to reflect the similarity between two genotypes. This measure is used to modify the raw fitness by fitness sharing. This genotype-based diversity clearly improves the performance over a basic direct-encoding (Stanley and Miikkulainen, 2002), but it requires starting the evolutionary process using only one topology and it strongly depends on the genotype design.

3.2 Exploring the Space of Behaviors

So far, we have described two main challenges with computing a distance between genotypes or phenotypes in ER: the computational complexity and non-injection of the genotype/phenotype map. These challenges are common to all the methods that optimize structures such as graphs or trees, but ER must deal with another challenge that is specific to the *dynamic* nature of the evaluation of fitness. In ER, the fitness is evaluated by observing the behavior of the robot that corresponds to the genotype; this process is dynamic because the behavior at each time step depends on the whole history of the behavior and of the environment. For instance, two robots equipped with similar neuro-controllers should exhibit the same behaviors at the beginning of an experiment; but the accumulation of differences and the environment can lead to two qualitatively

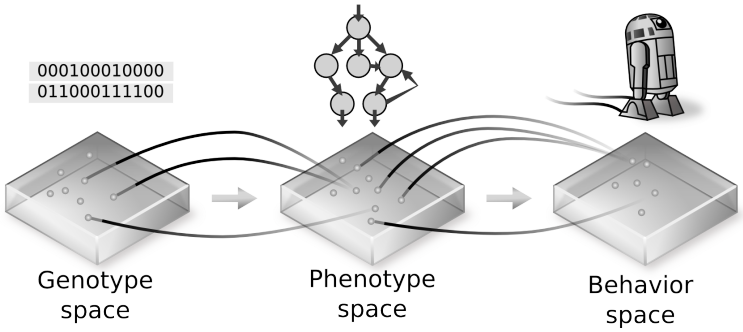


Figure 1: In a typical evolutionary algorithm, genetic operators manipulate genotypic representations of candidate solutions (left) that are developed into phenotypes (middle), whose fitness is evaluated. Several genotypes can correspond to the same phenotype, making the genotype/phenotype map non-injective. In evolutionary robotics, the phenotypes (e.g. a robotics morphology or a neural network) are simulated during several time-steps and the behavior of the robot is observed to evaluate the fitness (right). Since several phenotypes may drive the robot in the exact same manner (see figure 2(b)), the phenotype/behavior space is also non-injective.

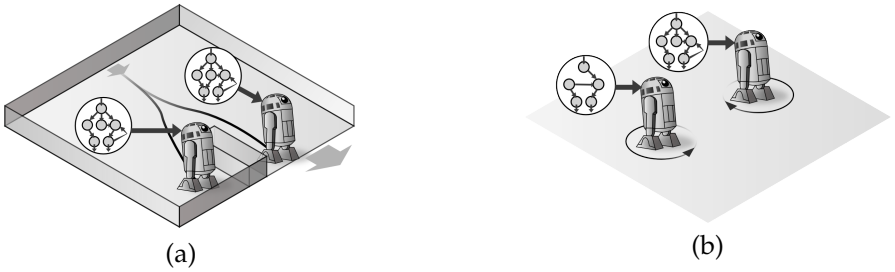


Figure 2: (a) Illustration of the non-linearity of the phenotype/behavior map: two very close neuro-controllers can lead to qualitatively different behaviors due to the interactions with the environment and the accumulation of differences during the evaluation of the fitness. (b) Illustration of the non-injectivity of the phenotype/behavior map: two structurally different neural networks can make the robot behave qualitatively similarly, for example the robot can turn on itself during the whole experiment.

different behaviors (figure 2(a)). At the opposite, an infinite number of controllers lead to the same qualitative behavior (figure 2(b)). An extreme example is the neural networks whose output are not connected to anything: whatever the topology and the parameters, the robot will not move. Hence, ER introduces a phenotype/behavior map (figure 1) that is highly non-linear but also non-injective.

Ultimately, evolving a neural network in ER (or any other structure in ER) is only a way to evolve behaviors while limiting the constraints on the space of available behaviors. Consequently, it is the exploration of novel behaviors that has to be fostered, and not only the exploration of genotypes or phenotypes. Although intuitive, this idea only surfaced during the last few years with the introduction of the novelty search algorithm (Lehman and Stanley, 2008, 2010). This algorithm relies on a radical and counter-intuitive idea: maximize the novelty of behaviors *instead of* the objective function (the fitness). This approach is designed to bring two benefits over traditional fitness-based evolutionary methods: (1) override the deceptiveness of most fitness functions and (2) make the evolutionary process more open-ended.

The first step when using novelty search is to define a task-specific distance between behaviors. Once such a distance has been setup, Lehman and Stanley propose to measure the novelty $\rho(i)$ of an individual i by computing the mean behavioral distance between i and its k nearest neighbors:

$$\rho(x) = \frac{1}{k} \sum_{j=0}^{k} d_b(x, \mu_j) \tag{4}$$

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. 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 a minimal threshold ρ_{min}, a parameter adjusted dynamically during an experiment.

In evolutionary robotics, novelty search substantially improved neuro-evolution in at least three noteworthy tasks, each of them associated with a task-specific distance:

- A wheeled robot must reach a goal in a maze that includes an attractive dead-end in the direction of the goal (Lehman and Stanley, 2008, 2010; Mouret, 2011); distance: Euclidean distance between positions at the end of experiment.
- A biped robot must walk as fast as possible (Lehman and Stanley, 2010); behaviors are characterized by the list of offsets of the bipeds center of mass sampled at one second intervals during the evaluation; distance: Euclidean distance between the lists of offsets.
- A wheeled robot must use its plastic neurons to learn how to find a reward in a T-maze (Risi et al., 2009); each behavior is characterized by a vector that records crashes and rewards in each trial; distance: Euclidean distance between the vectors.

While novelty search is not a diversity mechanism *stricto sensu*, it can be seen as a process in which only diversity is optimized. The current algorithm is more costly than diversity preservation because of the archive: as the archive grows, it becomes more and more costly to find the nearest neighbors[2]. At any rate, novelty search highlights

[2]If a suitable structure is employed to store behaviors (e.g. a kd-tree), the best complexity of the search for

that it is useful to explicitly explore the behavior space and has introduced the concept of behavioral distance. A less radical approach is to employ the same kind of user-specific distance in a diversity maintenance framework. Following this idea, Mouret (2011) compared novelty search with multi-objective diversity on the same deceptive maze as Lehman and Stanley, using the same task-specific behavioral distance. In this highly-deceptive task, novelty search found a solution faster than an EA improved with multi-objective behavioral diversity maintenance. This result is easily understood because the employed fitness does not provide any useful gradient until a good solution is found. However, the same results also show that both methods succeeded in finding a working solution in a few hundred of generations, whereas a standard fitness-based algorithm never found any solution. Additionally, the multi-objectivization led to slightly better fitness values because it adds a pressure towards efficiency that is absent from novelty search.

In another domain, Mouret and Doncieux (2009a) employed multi-objective diversity (section 2.2) to solve a sequential light-seeking task. In this setup, a wheeled robot is placed in an arena with seven illuminated light switches. At the beginning of an evaluation, all switches except one are off, but activating a lightened switch activates one or several other switches in the environment. The goal is to activate a particular switch, the exact circuit having to be discovered by the evolutionary process. Mouret and Doncieux empirically compared the behavioral diversity approach with an incremental evolution technique (Mouret and Doncieux, 2008). Each behavior was described by the vector of light switches that has been activated and the Euclidean distance was then used to compare behavior descriptors. Both methods managed to solve the sequential task whereas a standard algorithm never found any working solution, but behavioral diversity was slightly faster at finding the first working solution.

Outside of the ER field but still in neuro-evolution, Mouret and Doncieux (2009b) also benchmarked NEAT, a graph-probing multi-objective diversity mechanism and multi-objective behavioral diversity to evolve the topology and the parameters of neural networks that compute the "(a XOR b) AND (c XOR d)" Boolean function. Behaviors were described by the Boolean vector of outputs for each input pattern. Results showed that behavioral diversity was as efficient as NEAT while being substantially simpler to implement.

Last, Trujillo et al. (2008b,a) and Moriguchi and Honiden (2010) both proposed to replace the genotypic distance used in NEAT with a behavioral distance. Trujillo et al. investigated an obstacle avoidance task for mobile robots. The behavior exhibited by each neuro-controller is described by character strings which represent the traversed path of the robot within the training environment. Results show that the proposed method does not improve the performance over the standard NEAT, probably because the task is very simple, but the authors clearly observed an increase in the diversity of behaviors. In a more recent paper, Moriguchi and Honiden (2010) also modified the distance employed in the NEAT niching scheme to replace it with a task-specific behavioral distance. They tested their approach on a generic suite of problems of reinforcement learning that are based on "tunable" Markovian Decision Processes (tunable MDP). These authors concluded that the modified NEAT outperformed the original NEAT, in particular for problems with a high noise level and large state space.

---

the k-nearest neighbors is $O(\log(n))$, with n the size of the archive. As a consequence computing the novelty of a given individual should at best be $O(n \log(n))$. Nevertheless, when the size of the population is small with regard to the size of the archive, diversity mechanisms are more efficient.

3.3 Generic Behavioral Distances for ER

ER researchers often spend a long time designing fitness functions that lead to interesting solutions. Thanks to behavioral diversity, they should now be able to use simpler and more intuitive fitness functions. However, they also need to define a behavioral distance, a task that may also be challenging for real-world problems. Furthermore, defining this distance adds human biases to the process, whereas ER researchers aim at reducing the human knowledge required to solve a given task. In an ideal ER setup, ER researchers would only define a high-level fitness function and let the generic evolutionary process do the rest. This goal could be achieved with a generic behavioral distance function that could be used with most ER tasks while still improving the evolutionary process.

Any ER experiment involves robots with actuators and, most of the time, sensors. These data closely reflect the behavior of the robot from its point of view and they are easy to access. They therefore appear as a good starting point to design a generic behavioral distance for ER.

Following this line of thought, Gomez (2009) investigated several generic behavioral distances to maintain the behavioral diversity in the Tartarus problem: in a 6×6 discrete environment, a bulldozer controlled by a recurrent neural network must push blocks against the walls. The neural network had a fixed topology (only the parameters were evolved) and the diversity was sustained using the crowding mechanism. In the Tartarus problem, four actions are possible at each time step. The behavior of the robot at each time is described with a binary vector of four values and each vector is concatenated for each time step to form a large binary vector β that describes the behavior in a binary form. Using this idea, Gomez (2009) compared five distances:

- Fitness difference: since the fitness describes behaviors, the absolute value of difference between the fitness of two individuals gives a distance between their behaviors;
- Euclidean distance in genotype space;
- Hamming distance between β vectors;
- Relative entropy;
- Normalized Compression Distance (NCD) between β vectors: an approximation of the normalized information distance that uses a data compressor to estimate the Kolmogorov complexity of a data stream.

According to Gomez (2009), NCD is the most promising distance because it is less sensitive to mis-alignment between sequences. It also allows comparing two sequences of different length. However, NCD requires performing at least one compression for each distance computation. Its computational cost is therefore much higher than the other methods.

Gomez concluded that the three behavior-based distances outperformed the other analyzed distances. In these experiments, the NCD-based diversity led to the best fitness values. However, the Hamming distance surprisingly led to only slightly worse results, with a much lower cost of computation.

The Tartarus problem is very abstract and one can argue that its discrete formulation is far away from a real-world robot that must survive in a continuous world. Doncieux and Mouret (2010) extended Gomez's work by comparing several distances

in a continuous ER task. In such tasks, typical experiments last several thousands of time-steps, each one involving dozens of sensors and actuators; computing the NCD therefore appeared too costly for such setups. Doncieux and Mouret (2010) evolved the topology and the parameters of neuro-controllers that make a simulated robot go towards a ball, take it, find a basket, put the ball into the basket, perform a half-turn, search and take another ball, put it into the basket, etc. They used a multi-objective diversity based on the maximization of the mean distance to the population (section 2.2). They studied four different distances:

- task-specific: a Euclidean distance between vectors that describe the number of times the robot is in a given user-defined state;
- trajectory: a Euclidean distance between vectors that describes the number of time-steps spent in each part of the arena.
- Fourier coefficients: a Euclidean distance between the first coefficients of a discrete Fourier transform of the sensory-motor vector;
- Hamming distance: a Hamming distance between binary versions of the sensor-motor vector.

Doncieux and Mouret (2010) concluded that the Hamming-based experiments generated the most efficient controllers. The experiments based on the task-specific distance are the next most successful experiment. The worse results were obtained by the control experiment in which behavioral diversity is not encouraged.

3.4 Conclusion

Table 1 summarizes the main features of the papers published about sustaining behavioral diversity. The following conclusions can be drawn from this short review:

- sustaining behavioral diversity substantially improves the results of the evolutionary process;
- generic distance, and especially the Hamming distance between sensory-motor vectors, can be as efficient as *ad hoc* behavioral distances;
- most authors used different diversity mechanisms;
- all authors worked on a single but different task.

To go further, it is now necessary to compare several diversity mechanisms, several distances and several ER tasks in the same study. This is the objective of the present paper.

4 Method

Our general goal is to find which combinations of distance/diversity mechanism consistently outperform the other combinations, regardless of genotype or task. To reach this goal, we need to explore the whole set of combinations of diversity mechanisms, distances, genotypes and tasks. Unfortunately, we don't have infinite computational power, therefore we had to choose a subset of the published building blocks, then explore all the combinations.

We chose three mobile robotic tasks, in which neural networks are evolved to control a robot. The phenotype is therefore always a neural network. Neurons are classic sigmoid neurons with an output in $[0; 1]$ and a bias in $[0; 1]$ (in the NEAT experiments, an input is added for the bias).

Paper	Tasks	Distance	Div. mechanism	Gen.	Phen.
Lehman and Stanley (2008)	deceptive maze	task-specific gen.-based (NEAT)	nov. search fit. sharing	NEAT	NN
Trujillo et al. (2008b)	obstacle avoidance	N-GLD gen.-based (NEAT)	fit. sharing	NEAT	NN
Risi et al. (2009)	T-maze	task-specific gen.-based (NEAT)	nov. search fit. sharing	NEAT	NN
Mouret and Doncieux (2009b)	xor and xor	task-specific graph-probing (gen.) gen.-based (NEAT)	MOEA fit. sharing	NN	NN
Mouret (2011)	deceptive maze	task-specific	MOEA nov. search	NN	NN
Mouret and Doncieux (2009b)	light-seeking	task-specific	MOEA	Reals	MLP
Gomez (2009)	Tartarus (discrete)	Hamming distance Euclidean dist. (gen.) Fitness distance Hamming distance Relative entropy NCD	crowding	Reals	RNN
Moriguchi and Honiden (2010)	tunable MDP	task-specific gen.-based (NEAT)	fit. sharing	NEAT	NN
Lehman and Stanley (2010)	deceptive maze humanoid gait	task-specific gen.-based (NEAT) task-specific gen.-based (NEAT)	nov. search fit. sharing nov. search fit. sharing	NEAT	NN
Doncieux and Mouret (2010)	ball collecting	task-specific Hamming distance FFT distance traj. discretization	MOEA	NN	NN

Table 1: Overview of the published papers about the explicit exploration of the behavior space in evolutionary robotics, via diversity techniques or novelty search. NN: Neural-network with an evolved topology; RNN: recurrent neural network with a fixed topology; MLP: Multi-Layer Perceptron (feed-forward neural network) with a fixed topology; MOEA: Multi-Objective Evolutionary Algorithm; NCD: Normalized Compression Distance; N-GLD: Normalized Levenshtein distance; Reals: vector of real numbers; nov. search: novelty search; fit. sharing: fitness sharing.

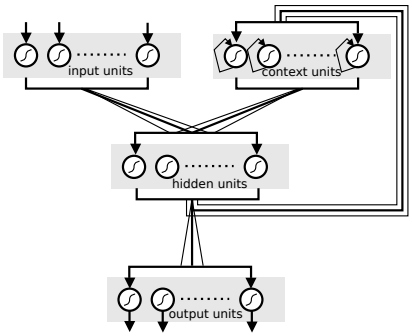


Figure 3: Topology of the Elman network used in this study. Each input unit is connected to all hidden units. Each hidden unit is connected to one context unit. Each context unit is connected to itself and to each hidden unit. Each hidden unit is connected to each output unit.

4.1 Genotype/Phenotype Maps

4.1.1 Elman(-Jordan) Neural Network

Elman-Jordan networks[3] (Elman, 1993) are simple recurrent neural networks that extend the multi-layer perceptron with a hidden layer of context units (figure 3). These context units allow the network to store data about the previous states in a network smaller than a fully connected neural network. We chose this structure as a fixed-topology neural network because using such a basic memory might be useful to solve the tasks studied in the present paper.

We fixed the number of context units and those of hidden units to the number of input units; it follows that for n_i inputs and n_o outputs, $2n_i^2+2n_i+n_in_o$ weights must be evolved. We employed a polynomial mutation (Deb, 2001) and did not use cross-over.

4.1.2 Direct Encoding of Neural Networks (DNN)

To evolve both the weights and the topology of neural networks, we designed a direct encoding as simple as possible, loosely inspired by NEAT. In this encoding, a neural network is described as a directed graph and five mutation operators are implemented:

- add a connection between two randomly chosen neurons;
- remove a randomly chosen connection;
- add a neuron by splitting an existing connection in two (the connection weight is kept on the two connections);
- delete a randomly chosen neuron and all related connections;
- change random weights using the polynomial mutation (Deb, 2001).

Cross-over is not employed. To initiate the evolutionary process, neural networks of the first generation are perceptrons without hidden layer, each one with randomly generated weights. This encoding has been previously employed in several previous papers (Doncieux et al., 2009; Mouret and Doncieux, 2009b; Doncieux and Mouret, 2009, 2010).

[3] In the remainder of this paper, we will call Elman-Jordan networks "Elman networks".

4.1.3 NEAT

NEAT (Stanley and Miikkulainen, 2002) is an advanced direct-encoding that exploits innovation numbers to sustain structural diversity and to implement cross-over. We included it in this study for two reasons: (1) it is the currently most successful neuro-evolution method and therefore a sensible reference point (2) it includes an efficient, genotype-based, mechanism to sustain the structural diversity of neural networks (see section 3.1).

4.2 Behavioral Diversity

4.2.1 Diversity methods

Fitness sharing. Fitness sharing is arguably the most popular single-objective method to sustain diversity. In the neuro-evolution literature, it is noticeably employed in NEAT (Stanley and Miikkulainen, 2002; Moriguchi and Honiden, 2010). This popularity makes fitness sharing the first mechanism we chose to sustain diversity. The σ parameter must be tuned and we chose to fix it at 0.05 for all the remaining experiments, because it showed reasonably good performances in preliminary experiments.

Multi-objective diversity (MO-diversity). Multi-objective diversity has been used several times to investigate behavioral diversity (Mouret and Doncieux, 2009a,b; Mouret, 2011; Doncieux and Mouret, 2010). The main strength of this method is the absence of parameters but several papers also showed that it can be more efficient than other diversity maintenance mechanisms (De Jong et al., 2001; Toffolo and Benini, 2003; Bui et al., 2005). Following the conclusions of Bui et al. (2005) (see section 2.2) and the previous work about MO-diversity (Mouret and Doncieux, 2009a,b; Mouret, 2011; Doncieux and Mouret, 2010), we chose to add an objective that corresponds to the mean distance to the rest of the population:

$$\text{Maximize} \left\{ \begin{array}{l} F(x^{(i)}) \\ \frac{1}{N} \sum_{j=0}^{j=N} d(x^{(i)}, x^{(j)}) \end{array} \right. \tag{5}$$

where $x^{(i)}$ denotes the i-th individual in the population, $F(x^{(i)})$ its fitness, N the size of the population and $d(x, y)$ the behavioral distance between x and y.

Multi-objective fitness sharing (MO-Sharing). Fitness sharing and the previously described MO-diversity differ by how they combine the diversity measure with the fitness function. However, they also differ in the way to estimate the "originality" of an individual: fitness sharing counts the number of close individuals whereas MO-diversity evaluates the mean distance to the population. To assess whether the potential differences between these two methods originate from the multi-objective approach or from the originality evaluation, we designed an intermediate variant that combines the originality measure used in fitness sharing with a multi-objective optimization.

Fitness sharing divides the fitness $F(x)$ by the niche count c (equation 2) to maximize $F(x)$ and minimize c. This leads to the following straightforward multi-objective formulation:

$$\text{Maximize} \left\{ \begin{array}{l} F(x^{(i)}) \\ C(x^{(i)}) = -\sum_{j=1}^{N} Sh(d(x^{(i)}, x^{(j)})) \end{array} \right. \tag{6}$$

where $Sh(x)$ denotes the sharing function (equation 1), $x^{(i)}$ denotes the i-th individual in the population, $F(x^{(i)})$ its fitness, N the size of the population and $d(x, y)$ the behavioral distance between x and y.

4.2.2 Distances

Task-specific (*ad hoc*). Most papers about behavioral diversity employed user-defined, task-specific, distances. For each task, we therefore designed a behavioral distance in the spirit of the published papers. These distances are described with the tasks in section 4.3.

Hamming distance. Generic distances are promising because they decrease the human knowledge inserted into the evolutionary process. Among the published methods, the Hamming distance appears to be both powerful, cheap to compute and easy to implement (Gomez, 2009; Doncieux and Mouret, 2010).

To define this distance, we first describe the set of sensor and effector data ϑ as follows:

$$\vartheta = \Big[\{\mathbf{s}(t), \mathbf{e}(t)\}, t \in [0, T] \Big] \tag{7}$$

where $\mathbf{s}(t)$ is the vector of size n_s of the perceptions at time t, i.e. the values coming from the n_s sensors; $\mathbf{e}(t)$ is the vector of size n_e of the effector values at time t; T is the observation length. For simplicity but without loss of generality, $\mathbf{s}(t)$ is assumed to be in $[0, 1]^{n_s}$ and $\mathbf{e}(t)$ in $[0, 1]^{n_e}$.

The Hamming distance counts the number of bits that differ between two binary sequences. It can be used to evaluate behavior similarity with, as inputs, ϑ_{bin}, the binarized version of ϑ, computed as follows:

$$\vartheta_{\mathbf{bin}} = [\vartheta_{\mathbf{bin}}(t), t \in [0, T]] = [\{\mathbf{s_{bin}}(t), \mathbf{e_{bin}}(t)\}, t \in [0, T]] \tag{8}$$

with $\mathbf{s_{bin}}(t) = \{s_{bin,0}(t), \ldots, s_{bin,n_s}(t)\}$, $\mathbf{e_{bin}}(t) = \{e_{bin,0}(t), \ldots, e_{bin,n_s}(t)\}$ and

$$s_{bin,i}(t) \quad = \begin{cases} 1 & \text{if } s_i(t) > 0.5 \\ 0 & \text{otherwise} \end{cases} \tag{9}$$

$$e_{bin,i}(t) \quad = \begin{cases} 1 & \text{if } e_i(t) > 0.5 \\ 0 & \text{otherwise} \end{cases} \tag{10}$$

with this definition, the Hamming distance is computed as follows:

$$\sigma_{ham}(\vartheta_1, \vartheta_2) = \sum_{t=0}^{T} h(\vartheta_{\mathbf{1},\mathbf{bin}}(\mathbf{t}), \vartheta_{\mathbf{2},\mathbf{bin}}(\mathbf{t}))$$

$$h(\vartheta_1, \vartheta_2) = \sum_{i=1}^{len(\vartheta_1)} 1 - \delta(\vartheta_1[i], \vartheta_2[j]) \tag{11}$$

where $len(\vartheta_1) = len(\vartheta_2)$ denotes the length of the binary sequences ϑ_1 and ϑ_2 and where $\delta(i, j)$ is the Kronecker delta:

$$\delta(i, j) = \begin{cases} 1 & \text{if } i = j \\ 0 & \text{otherwise} \end{cases} \tag{12}$$

Genotypic distance. Last, if the genotype is a vector of real numbers (like for the Elman network), a genotype distance can be computed by using the Euclidean distance between the vectors:

$$d(x, y) = ||x - y|| \tag{13}$$

where x and y are two real vectors that describe two individuals.

4.3 Tasks

Three tasks are investigated in this work: a deceptive maze (Lehman and Stanley, 2008; Mouret, 2011; Lehman and Stanley, 2010), a sequential light-seeking task (Mouret and Doncieux, 2008, 2009a) and a ball collecting task (Doncieux and Mouret, 2009, 2010). These tasks have been chosen for four main reasons:

- they are easy to reproduce and similar to many classical experiments of evolutionary robotics (Meyer et al., 1998; Nolfi and Floreano, 2004);
- despite their simplicity, they are hard to solve with classic methods from the literature;
- they have been previously employed to show the benefits of behavioral exploration in evolutionary robotics;
- they use the same simulator but pose different challenges.

These three tasks obviously cover a small part of the potential tasks that could be tackled in evolutionary robotics, but their strong roots in the evolutionary robotics literature make them good candidates to increment the knowledge of the field.

4.3.1 Deceptive Maze

This task is inspired by the deceptive maze introduced by Lehman and Stanley (2008, 2010) to demonstrate the efficiency of novelty search. A simulated mobile robot has to find a goal zone in a maze by relying only on its three laser range sensors and a goal sensor (see Fig. 4(a)).

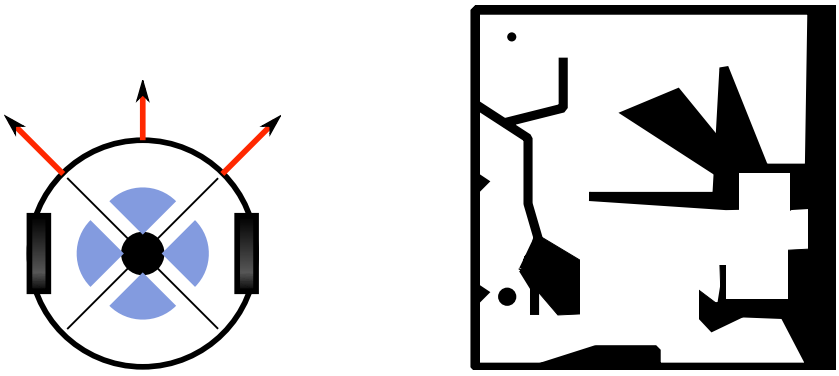
Fitness. An intuitive fitness is the opposite of the smallest distance to the goal, to be maximized:

$$F(x) = 1 - \frac{1}{K} \min_{t \in [0,T]} ||p_x(t) - p_g|| \tag{14}$$

where $p_x(t)$ is the position of the robot at time-step t, T the number of simulated time-steps, p_g the position of the goal and K the size of the map. On this particular maze, this fitness is highly deceptive because of the attractive dead-end to the north of the robot. To force the robot to use its sensors, the robot is stopped in its position if it touches a wall.

Task-specific distance. According to Lehman and Stanley (2008), the behavior of a robot can be described by its position *at the end of the experiment*. Two behaviors can be compared using a Euclidean distance between these descriptors:

$$d(x, y) = ||p_x(T) - p_y(T)|| \tag{15}$$



(a) (b)

Figure 4: (a) Overview of the simulated mobile robot. Each arrow outside of the robot's body is a range sensor (e.g. a laser range finder) that returns the normalized distance to the closest obstacle in that direction. The robot is also equipped with four pie-slice sensors that act as a compass towards the goal. This compass can "see" through walls. The robot is moved by two motors, the neural network setting the speed of each of them. (b) Map employed in this set of experiments (black area are walls). On this particular maze, this intuitive fitness (see text) is highly deceptive because of the attractive dead-end to the north of the robot. The size of the circle labeled "starting position" corresponds to the size of the robot.

Success Criterion An experiment is successful if the goal has been reached with a tolerance of the size of the robot, during the time of the simulation. This stops the evolutionary algorithm, therefore an individual that manages to reach the goal has the optimum fitness.

Number of inputs. 7: 3 range sensors and 4 inputs for the goal sensor.

Number of outputs. 2: the speed of each wheel.

4.3.2 Sequential Light-seeking

The sequential light-seeking task has been introduced by Mouret and Doncieux (2008) as a sequential task that can be solved with a simple multi-layer perceptron. In this task, a simulated wheeled robot is placed in an arena with seven different colored lights (figure 5). Each light is mounted on a switch which, when pressed, activates one or more lights in the arena. Once a light is on, it remains in the same state during the whole experiment. At the beginning of the experiment, all lights are off except one and the main goal is to turn on a particular light. The connection between lights and switches, a simple explicit dependency between tasks, is to be discovered by the evolutionary algorithm. The challenge for the evolutionary algorithm is to find a neuro-controller that "knows" the circuit.

To make this task a selection pressure challenge, and not an encoding challenge, the robot is provided with two sensors for each color of light, one on each side of the robot. Hence, this task can be solved with a simple feed-forward neural network, as shown by Braitenberg's vehicles that achieve phototaxis; therefore, a failure to solve the task will not indicate that the neural network is complex to find, but that the selective pressure was unable to drive the evolutionary process to the solution.

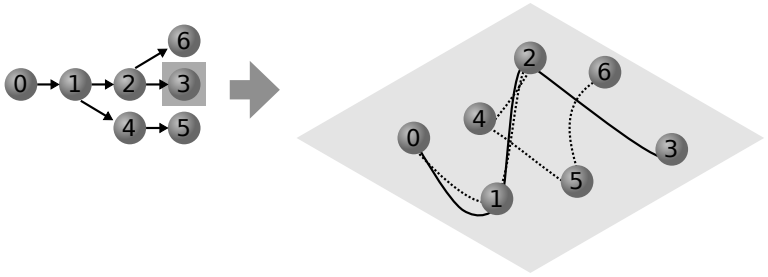


Figure 5: Overview of the sequential light-seeking task. Seven colored lights are placed in an arena, each of them being mounted on a button switch which turns on some other lights. (left) The light circuit (unknown to the algorithm) is represented by arrows, one switch being able to switch on several other lights to make the optimal sequence harder to find. The goal is to activate the switch of light 3, that is touching it while it is on. (right) Two examples of trajectory that reach the goal. Only the trajectory displayed with a solid line (0-1-2-3) corresponds to the success criterion, the robot that follows the dotted trajectory also solves the task but it uses a sub-optimal sequence.

Fitness function. An intuitive fitness function is the opposite of the number of time-steps to reach the goal light. To encourage a sensor-based behavior, we perform three experiments with the same circuit but with different locations of the lights. The values for the three experiments are averaged to get the final fitness $F(x)$:

$$f_e(x) = \begin{cases} 1 - t_g/T \text{ if the goal switch has been pressed} \\ 0 \text{ otherwise} \end{cases} \tag{16}$$

$$F(x) = \frac{1}{3} \sum_{e=1}^{3} f_e(x) \tag{17}$$

where t_g is the time-step at which the goal switch has been pressed and T the length of an experiment. This fitness is non-informative until the goal switch has been pressed. Given the size of the sequence (figure 5), it is unlikely that a random neuro-controller will obtain a non-zero score for the three experiments. Diversity will therefore be the main guide during the first part of the search.

Task-specific distance. Mouret and Doncieux (2009a) proposed to describe the behavior in one experiment with a vector b of binary values such as:

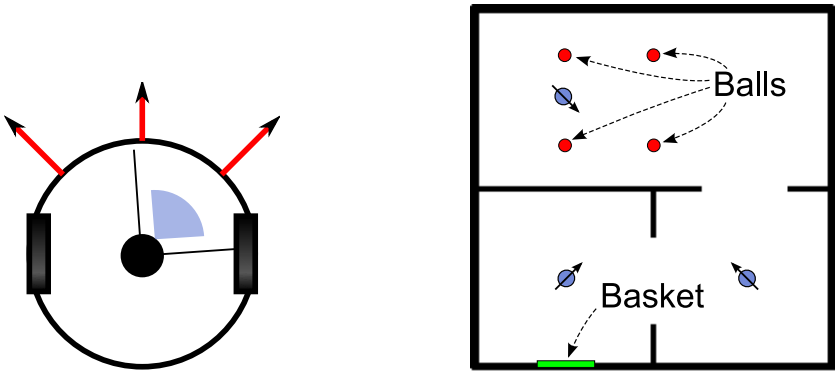
$$b_i^{(e)}(x) = \begin{cases} 1 \text{ if switch } i \text{ has been activated} \\ 0 \text{ otherwise} \end{cases} \tag{18}$$

where e is the number of the considered experiment. The vectors for the three experiments can be concatenated to form a vector β:

$$\beta(x) = \{b^{(1)}, b^{(2)}, b^{(3)}\} \tag{19}$$

These vectors are compared using the Hamming distance:

$$d(x, y) = \sum \delta(\beta(x), \beta(y)) \tag{20}$$



(a) (b)

Figure 6: (a) Overview of the simulated mobile robot for the ball-collecting task. Each arrow outside of the robot's body is a range sensor (e.g. a laser range finder) that returns the normalized distance to the closest obstacle in that direction. The robot is also equipped with two pie-slice sensors that detect the balls and two more that detect the basket. The view range of the right sensors is depicted on the figure, the left ones are symmetric. Contrary to the maze experiment, these sensors can't "see" through walls. The robot is moved by two motors, the neural network setting the speed of each of them. (b) Map employed in this set of experiments (black area are walls). The three different initial positions and orientations of the robot are depicted with a circle and an arrow.

Success criterion. An experiment is successful if the *optimal* sequence (i.e. no useless switches were pressed) has been performed by the robot *for the three experiments*. An evolutionary process that finds this sequence may still improve the fitness by making robots move faster, that is why reaching this success criterion does not stop the evolutionary process.

Number of inputs. 14: two for each light.

Number of outputs. 2: the speed of each wheel.

4.3.3 Ball-collecting Robot

The ball-collecting task has been introduced by Doncieux and Mouret (2009, 2010) as a task requiring basic navigation skills—obstacle avoidance, navigation towards different goals—and the ability to change behavior depending on the context. The goal of this task is to explore an arena to find balls and take them to a basket. A fitness point is granted each time a ball is put into the basket. Before getting a point, the robot must therefore learn to search for a ball, take it and keep it, search for the basket, reach it and then, and only then leave the ball. As with the sequential light-seeking task, the reward comes only after a particular sequence of actions. Relative to the light-seeking task, the behaviors of the ball collecting task involve other aspects than phototaxis behavior. Furthermore, the task is not completely fulfilled when the first ball is collected, as three other balls are to be collected: compared to the sequential light-seeking task, the ball collecting task requires *repeating* a sequence.

Like in (Doncieux and Mouret, 2009, 2010), balls disappear from the arena as soon as they are released, hence making the task more difficult. Contrary to Doncieux and

Mouret (2009, 2010), the robot has a circular shape to make collision detection simpler and faster and to use the same simulator as for the other tasks. The features of the arena, namely the placement of the balls and the walls, have also been modified to make the navigation more difficult (figure 6). An evaluation consists of three different experiments in which only the initial position of the robot changes (figure 6). Four balls can be collected during each experiment.

Fitness function. The fitness function is the number of balls put into the basket during three experiments with different initial positions of the robot, normalized by the maximum number of balls that can be collected:

$$F(x) = \frac{1}{3} \sum_{e=1}^{3} \frac{n_c^{(e)}}{n_{c,max}^{(e)}} \tag{21}$$

Task-specific distance. Considering that the most important features of the task are the balls, which the robot must move to a particular place, the task-specific distance is defined as the vector of final positions of the balls. If a ball is not moved, or if the robot doesn't release it, the position is the initial position of the ball, otherwise, it is the position of the robot when it released the ball. The behavior is described by a vector of positions:

$$b^{(e)}(x) = \left\{ (x_i, y_i), i \in [1, n_{c,max}^{(e)}] \right\} \tag{22}$$

$$\beta(x) = \left\{ b^{(1)}(x), b^{(2)}(x), b^{(3)}(x) \right\} \tag{23}$$

and the behavior similarity measure is the Euclidian distance:

$$d(x, y) = ||\beta(x) - \beta(y)|| \tag{24}$$

Success criterion. An experiment is considered successful if individuals that collect every ball and put them all into the basket have been generated for the three experiments.

Number of inputs. 10: three laser wall distance sensors, two bumpers, two ball presence sensors, two basket presence sensors and one carrying ball sensor. The presence sensors are binary sensors, their output is one if the detected object (ball or basket) is in the field of view of the sensor, 0 otherwise. The carrying ball sensor output is 1 if a ball is carried and 0 otherwise.

Number of outputs. 3: the speed of each wheel and a collecting ball effector. If the value of this effector is above 0.5, a carried ball is kept or, if no ball is carried, a ball is collected if its center is inside the robot, otherwise, nothing happens. If the effector value is below 0.5, a carried ball is ejected, otherwise nothing happens. If the robot is in front of the basket and touches it at this particular moment, one reward point is given, otherwise, the ball just disappears from the arena.

4.4 Treatments, Setups, and Experimental Choices

The selected evolutionary algorithm is NSGA-2 (Deb et al., 2000) because this MOEA is recognized as one of the most efficient and most well-spread algorithms. When only one objective is optimized, NSGA-2 is a classic elitist evolutionary algorithm with a tournament-based selection.

Identifier	No div.	Fit. shar.	MO-div	MO-shar.	Ad-hoc	Hamming	Gen-based
No div.	•						
Fit.Sharing/Ad-hoc		•			•		
Fit.Sharing/Hamming		•				•	
Fit.Sharing/Gen.		•					•
MO-div/Ad-hoc			•		•		
MO-div/Hamming			•			•	
MO-div/Gen.			•				•
MO-Sharing/Ad-hoc				•	•		
MO-Sharing/Hamming				•		•	
MO-Sharing/Gen.				•			•

Table 2: Elman treatments. Each treatment is tested on the three tasks.

Identifier	NEAT	No div.	Fit. shar.	MO-div	MO-shar.	Task-spec.	Hamming	Gen-based
No div.		•						
Fit.Sharing/Ad-hoc			•			•		
Fit.Sharing/Hamming			•				•	
MO-div/Ad-hoc				•		•		
MO-div/Hamming				•			•	
MO-Sharing/Ad-hoc					•	•		
MO-Sharing/Hamming					•		•	
NEAT	•		•					•

Table 3: DNN (direct encoding) treatments. Each treatment is tested on the three tasks.

In this work, we consider that a *treatment* is a combination of a diversity mechanism and a distance. To identify the best treatments, we test each of them on different *setups* that correspond to a combination task/genotype. By not trying several evolutionary algorithms, we assume in this work that the final fitness values and success rates depend more on the treatments than on the evolutionary algorithm employed. This assumption is justified by the fact that the published studies about behavioral diversity relied on different algorithms but all of them reported substantial improvements (see section 3.2).

Tables 2 and 3 summarize all the possible treatments with regard to each genotype; all of them are tested on the three tasks. Some combinations are impossible because of the specific genotype, for instance genotype-based distances are only studied with the Elman neural networks because we cannot compute the edit distance between two graphs. Additionally, we used the original c++ NEAT implementation[4] that depends on a specific evolutionary algorithm. Consequently, we have to define a setup especially for the NEAT experiments, whereas it is actually mostly a genotype-based diversity mechanism applied to a direct encoding.

For each of the 3 tasks, we investigate 10 treatments for the Elman setups, 7 treatments for the direct encoding setups and 1 experiment with NEAT. We therefore implemented $3 \times (18) = 54$ variants and launched 30 experimental units for each variant. The population is made of 200 individuals and each run lasts 5000 generations; put differently, our budget is $100,000$ evaluations of the fitness function for each run. The statistical differences between average fitness values are assessed with the Mann-Whitney U test, because we have no reason to assume a Gaussian distribution of the results; we even anticipate a non-Gaussian distribution with some runs stuck in a local optimum and others around the optimal value. Success rates are compared with the Fisher's exact test, a non-parametric test suitable to compare success rates (Yuan and Gallagher, 2009).

These experiments have been carried in the Sferes$_{v2}$ framework (Mouret and Don-

[4] Available on `http://eplex.cs.ucf.edu/software.html` and in the source code associated with the present paper.

cieux, 2010) and the source code to reproduce them is available on the following website: `http://www.isir.fr/evorob_db`

5 Results

5.1 Data

Figure 7 depicts the median fitness obtained for each run and figure 8 reports their success rate (defined using the task-specific success criterion). This large amount of data makes it difficult to extract clear tendencies and an overall ranking, that is why we investigated an alternate representation.

The goal of the present paper is to identify which treatments outperform the others, regardless of the task or the genotype used. An ideal method would allow assigning a global score to rank each treatment. Nonetheless, fitness values cannot be simply aggregated, because the results with regard to each task are numerically incomparable. An alternative point of view is to perform a Pareto ranking between treatments, the fitness in each setup defining a particular objective.

The definition of the Pareto dominance (section 2.2) must be slightly modified to account for the statistical significance of the differences. We first need to define how to compare treatments in a given setup, with regard to a performance measure $O(x)$ (e.g median fitness or success rates):

Definition 2 *A treatment $x^{(1)}$ is said* statistically worse *(reciprocally better) than $x^{(2)}$ with regard to the measure $O(x)$, if both conditions 1 and 2 are true:*

1. $O(x^{(1)}) < O(x^{(2)})$ *(reciprocally $O(x^{(1)}) > O(x^{(2)})$)*
2. $p < 0.05$, *where p is the p-value obtained using the statistical test associated with $O(x)$ (Fisher exact-test for success rates and Mann-Whithney U-test for medians).*

We use this definition to extends the classic Pareto dominance relation to include the statistical significance:

Definition 3 *A treatment $x^{(1)}$ is said to statistically dominate another treatment $x^{(2)}$, if both conditions 1 and 2 are true:*

1. *the solution $x^{(1)}$ is not* statistically *worse than $x^{(2)}$ with respect to all setups;*
2. *the solution $x^{(1)}$ is* statistically *better than $x^{(2)}$ with respect to at least one setup.*

The ideal treatment will be the best with regard to every setup; nevertheless, results will more likely exhibit *trade-offs* and we will therefore be mostly interested in the non-dominated set of treatments, that is treatments for which there exists no treatment statistically better on all setups.

The dominance relation is hard to apprehend by analyzing typical bar graphs and fitness plots. Fortunately, researchers in multi-objective optimization have investigated many ways to represent the efficiency of individuals when many objectives are involved (Deb, 2001). Relying on this literature, we chose to use the *parallel plot* to display the results of our experiments: each objective is represented by a vertical axis and a particular individual is depicted as the set of segments that links its performance with regard to each axis (figure 9 is a commented example of parallel plot). Figure 10 is the parallel plot of the median fitness values after $100,000$ evaluations (5000 generations). Figure 11 is the complementary parallel plot that displays the success rate of each treatment.

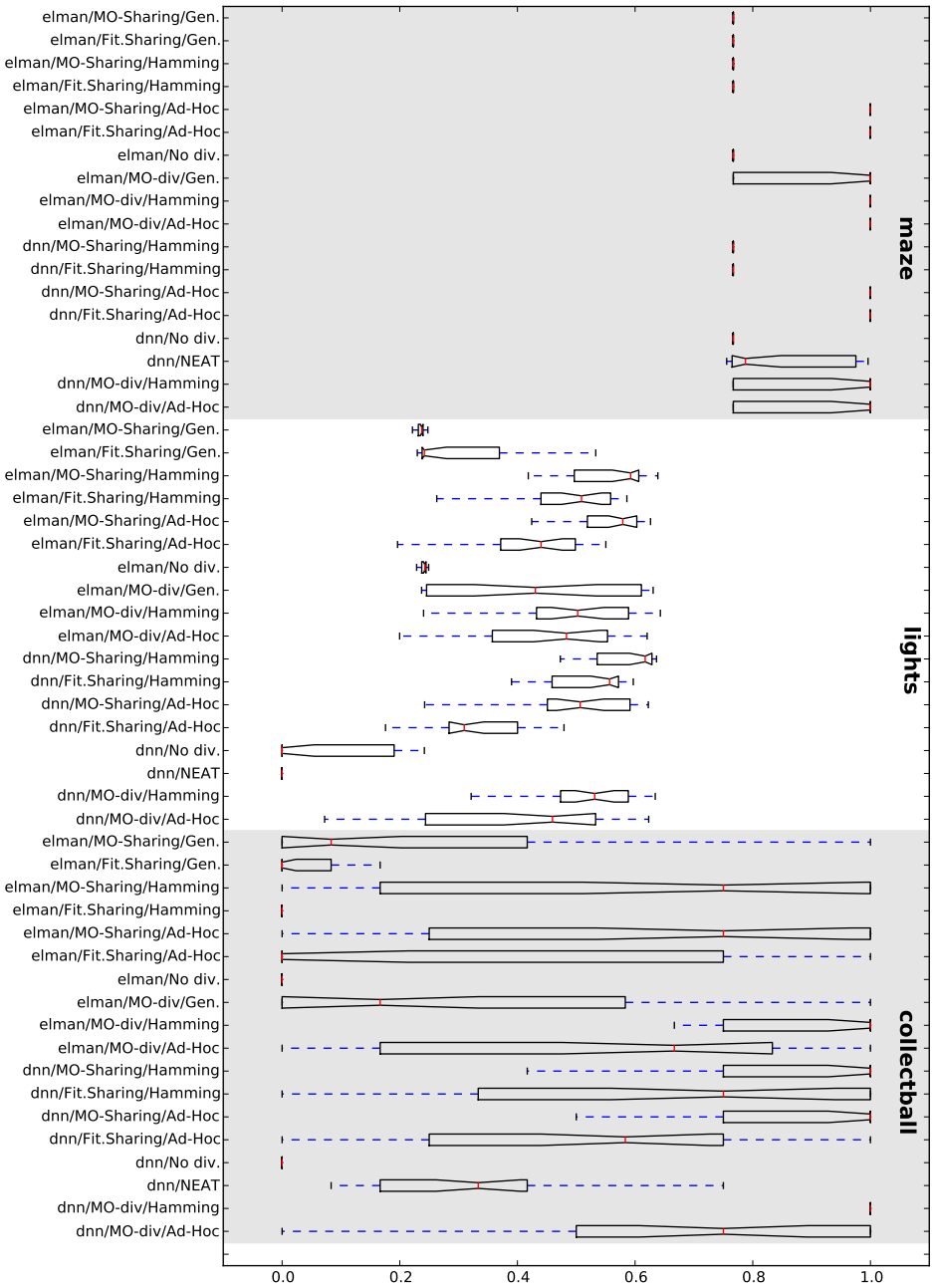


Figure 7: Box and whisker plot of the fitness values after 5000 generations (100,000 evaluations). The boxes extend from the lower to upper quartile values of the data, with a line at the median. The whiskers show the range of the data.

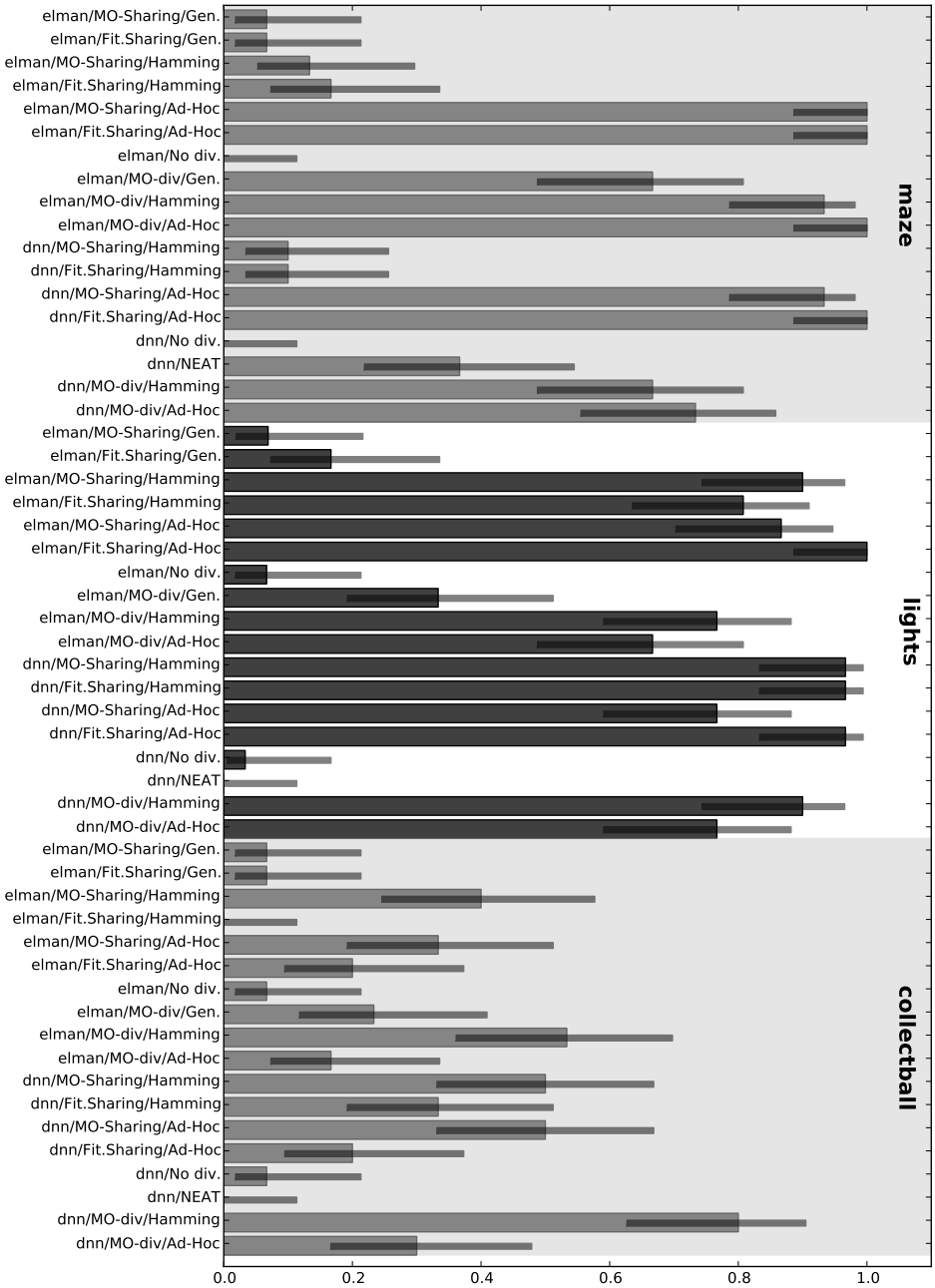


Figure 8: Success rates for each treatment. The small dark bars on the right of the main bars show the 95% confidence interval according to the Wilson procedure (Barrero et al., 2010).

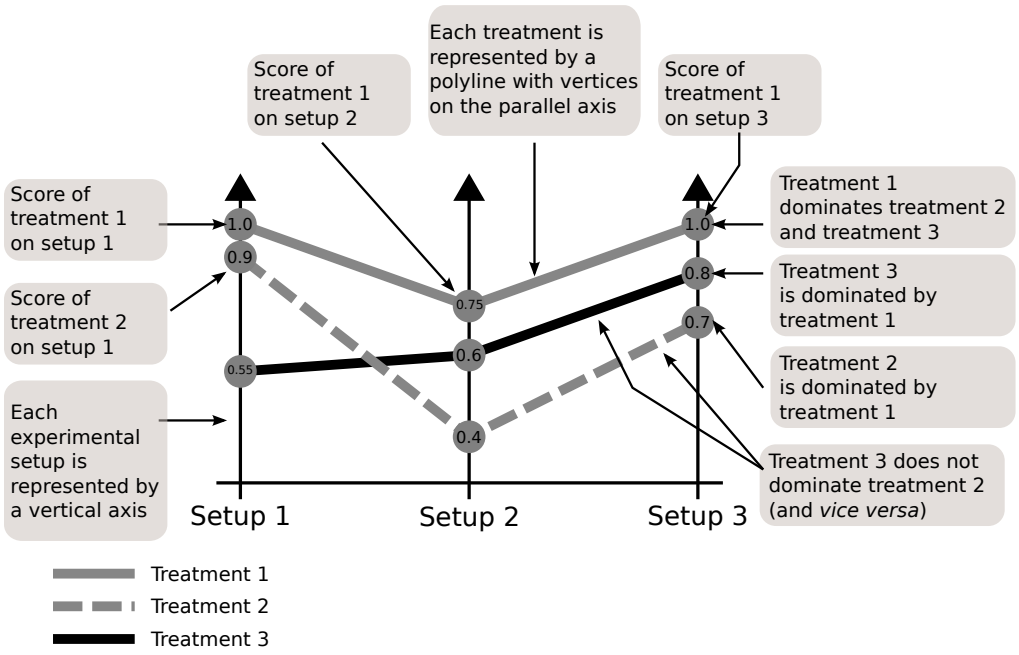


Figure 9: Commented example of parallel plot. The first Treatment obtained a score of 1.0 on the first setup, 0.75 on the second one and 1.0 on the third. It dominates the second treatment (0.9, 0.4 and 0.7) as well as the third one (0.9, 0.4 and 0.7) because it is better than the two other treatments on all setups; the first treatment is therefore a Pareto-optimal trade-off. However, the second treatment does not dominate the third one because it is better on one setup and worse in the two others; similarly, the third treatment does not dominate the second treatment because it is worse than the the second treatment on the first setup.

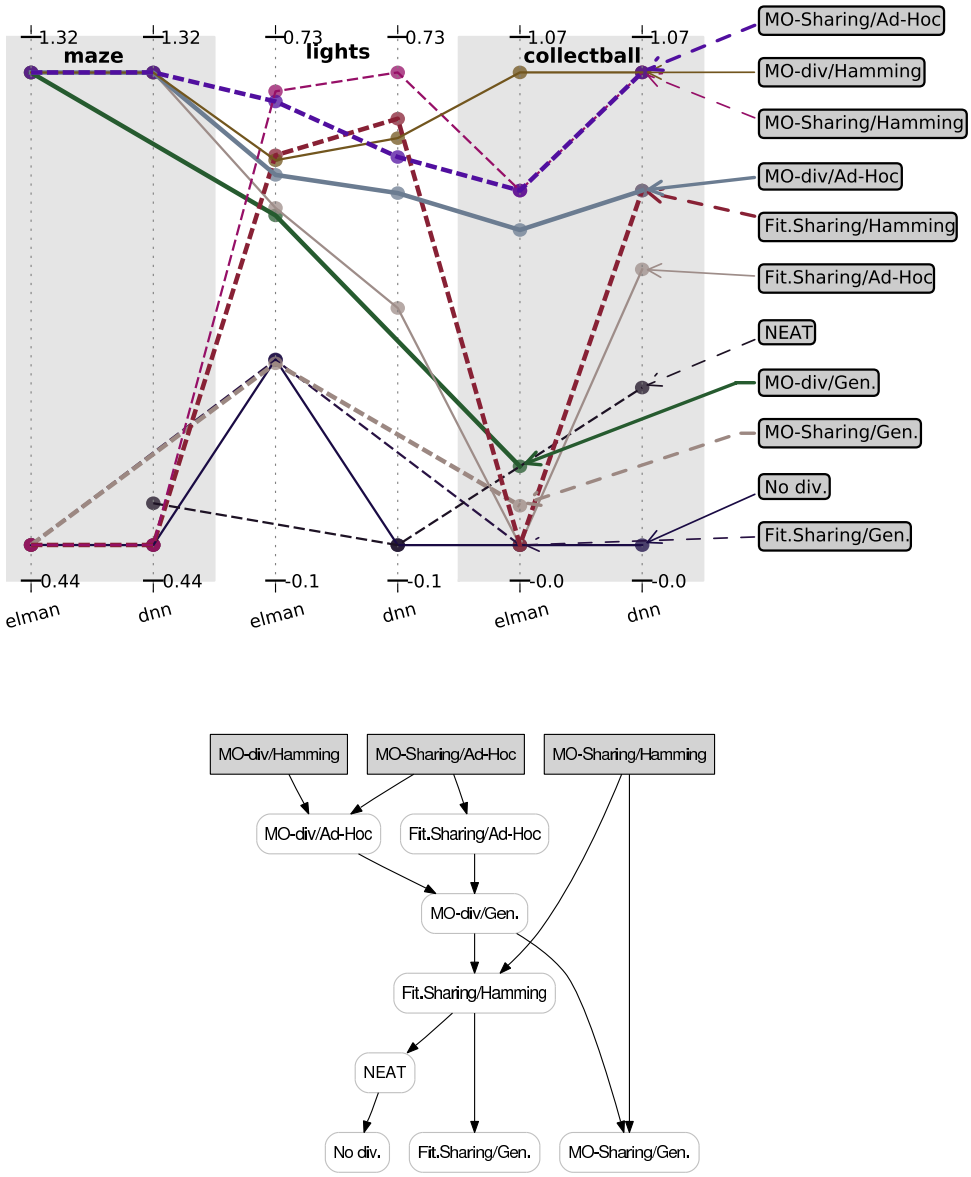


Figure 10: Median fitness for each treatment, after 5000 generations (100,000 evaluations). (top) Parallel plot: each vertical axis is a setup and each treatment is displayed as the line that links its score for each setup (see figure 9 for a commented example of a parallel plot). Line style of the arrows (bold and dashes) match the line style of the treatment. To enhance readability, values have been normalized using the range displayed on top and bottom of each vertical axis. (bottom) Pareto dominance relations, taking into account statistical significance (Mann-Whitney U-test, see appendix). Grey boxes denote non-dominated treatments.

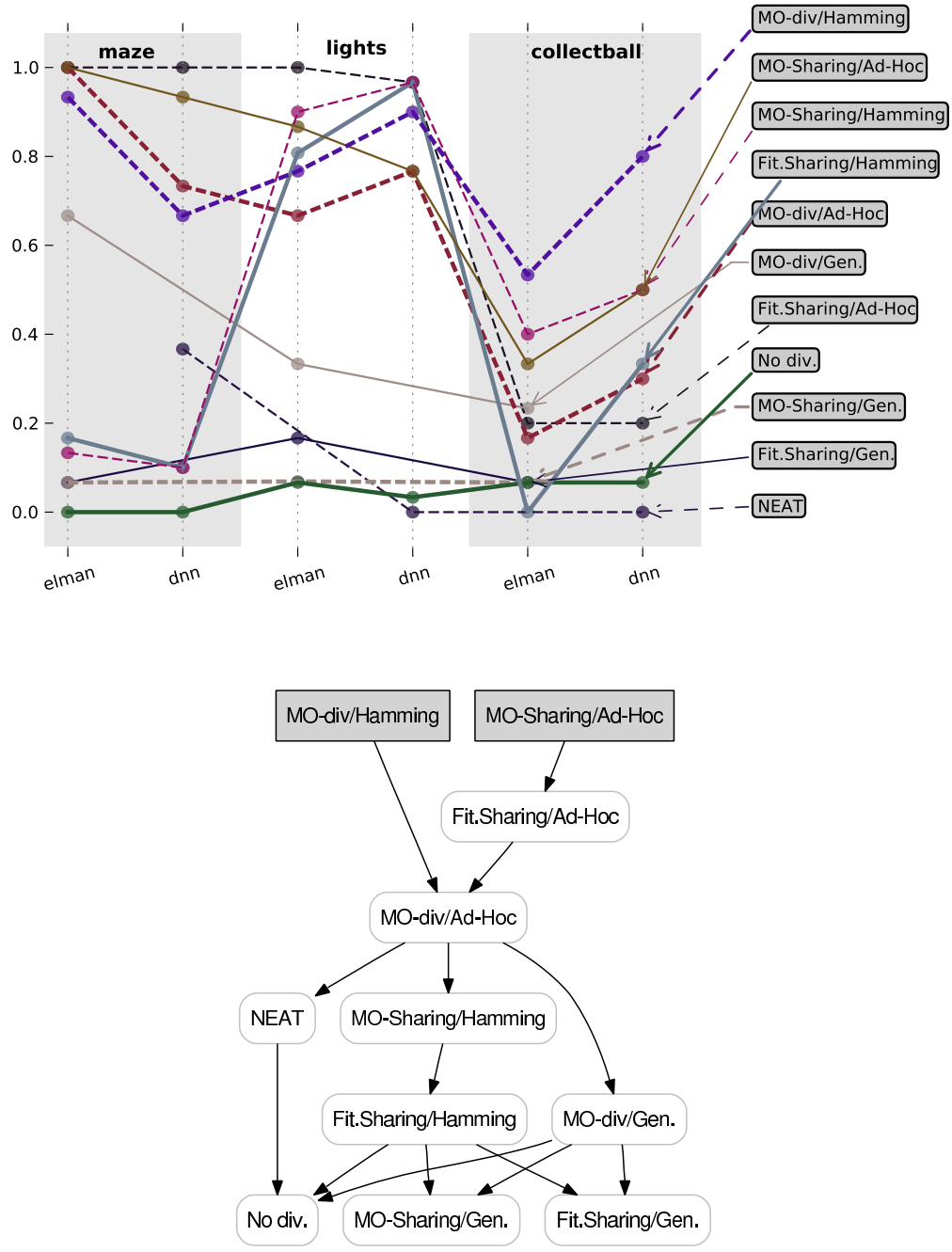


Figure 11: Success rate for each treatment, after 5000 generations (100,000 evaluations). (top) Parallel plot: each vertical axis is a setup and each treatment is displayed as the line that links its score for each setup (see figure 9 for a commented example of a parallel plot). Line style of the arrows (bold and dashes) matches the line style of the treatment. (bottom) Pareto dominance relations, taking into account statistical significance (Fisher's exact test, see appendix). Grey boxes denote non-dominated treatments

While the parallel plots visually display the Pareto dominance, they can be misleading because they don't show whether visual differences are statistically significant, for obvious readability reasons. We initially thought about computing the Pareto layers, that is layers of treatments such that treatments on the same layer do not dominate each other and every treatment on layer n dominate every treatments on layers $n+1$. Unfortunately, creating such layers is not possible with statistically significant dominance because two treatments A and B may be non-dominated (i.e. no treatment dominates them) while A and B may not statistically dominate the same treatments. In the extreme case, a treatment with a very high variance may be non-dominated while not dominating any other treatment.

Since we are unable to display Pareto layers, we supplement the parallel plots with dominance graphs that depict Pareto dominance relations, *taking the statistical significance into account*. To create these figures, we first computed a graph in which each treatment is represented by a node so that an edge links a treatment A and a treatment B if and only if A statistically dominates B. We then transformed this graph into its transitive reduction[5] (Aho et al., 1972) to make it easier to read.

When a treatment is impossible for a given setup, the comparison regarding this setup is ignored. Hence, if a genotype-based diversity mechanism is better than another treatment on all the setups taken into account, we consider that the former treatment dominates the latter. To ease comparisons, we considered that NEAT was a treatment to the direct encoding (DNN), whereas it uses a different implementation of a direct encoding.

5.2 Comments

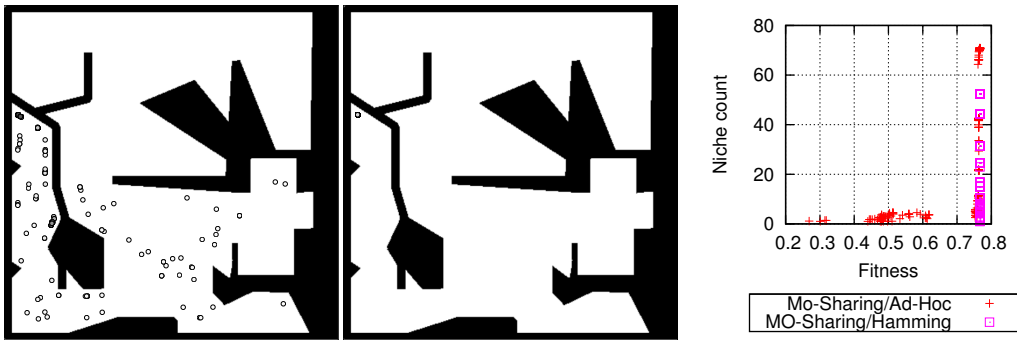
Control experiment The control experiment (No div.), in which diversity is not encouraged, does not dominate any treatment whatever the comparison criterion is (median fitness or success rate) and whatever the genotype/phenotype map is (DNN or Elman network). Figures 10 and 11 show that the control experiment gets near-zero success rates and a zero median fitness in all experiments except the lights-seeking task with an Elman network. This result justifies the need to improve the evolutionary process to solve the selected tasks.

Genotypic diversity. The genotypic diversity improved the results in all the treatments except when the sharing function is employed (figures 10 and 11, MO-Sharing/Gen and Fit.Sharing/Gen). This result confirms the published results stating that encouraging genotypic diversity improves the evolutionary process (Sareni and Krahenbuhl, 1998; Mahfoud, 1997; Friedrich et al., 2008).

The most efficient treatment with genotypic diversity is MO-div/Gen, which clearly dominates all other genotypic diversity treatments, including NEAT. The difference with MO-Sharing/Gen probably originates from a non-optimal choice of the σ value used in the sharing function. These results corroborate those that show that multi-objective diversity is an efficient alternative to fitness sharing to maintain genotypic diversity (De Jong et al., 2001; Abbass and Deb, 2003; Toffolo and Benini, 2003; Bui et al., 2005).

Nonetheless, MO-div/Gen obtained worse results than most behavioral diversity methods; as shown by the Pareto dominance graphs, only MO-Sharing/Hamming and Fit.Sharing/Hamming don't dominate MO-div/Gen. Additionally, it must be remem-

[5]The transitive reduction of a graph G is the smallest graph $R(G)$ such that $C(G) = C(R(G))$, where $C(G)$ is the transitive closure of G; the transitive closure $C(G)$ of a graph is a graph which contains an edge $\{u, v\}$ whenever there is a directed path from u to v.



(a) (b) (c)

Figure 12: (a) End position of robots (for the whole population) in a successful MO-Sharing/Ad-hoc run (generation 100). (b) End position of robots (for the whole population) in an unsuccessful MO-Sharing/Hamming run (generation 100). (c) Fitness and niche count of the individuals depicted on (a) and (b) (see equation 2 for the definition of the niche count). The niche count for MO-Sharing/Hamming is computed using the sharing distance and the value of σ used in our experiments (see appendix); similarly, the niche count for Mo-Sharing/Ad-hoc is computed using the Ad-hoc distance.

bered that these treatments are only possible when the Elman network is employed. Hence, only three setups are taken into account in the Pareto ranking (compared to six setups for the behavioral diversity treatments).

We can conclude that genotypic diversity is less efficient than most variants of behavioral diversity in the three investigated tasks. This confirms the published results about behavioral diversity being a promising alternative to genotypic diversity (Mouret and Doncieux, 2009b; Moriguchi and Honiden, 2010).

***ad hoc* distances.** Compared to genotypic diversity and the control experiment, the treatments that involves *ad hoc* distances improved both the success rate and the median value, for all tasks and diversity mechanisms. Combined with fitness sharing, the *ad hoc* distances obtained the best success rates in the maze experiments and in the light-seeking one. However, the same treatment obtained one of the worst success rates and medians in the ball-collecting experiment. Differences between the Fitness sharing/Ad-hoc and MO-Sharing/Ad-hoc treatments are not statistically significant ($p > 0.05$, see appendix), but MO-div/Ad-hoc leads to significantly worse results, especially in the case of the maze task with the Elman network. In these experiments, the sharing function is therefore more efficient with task-specific distances than the parameter-free multi-objective diversity (MO-div).

ad hoc treatments are dominated by Hamming treatments in the ball-collecting experiment, whereas they dominate them in most of the other setups. This result highlights one of the main pitfalls of task-specific distances: when correctly chosen, they may be very efficient but their design is not always obvious.

Hamming distance. Compared to *ad hoc* distance, Hamming distances mostly lead to a mirrored picture: sharing-based diversity mechanisms are dominated by several treatments while the MO-div treatment is non-dominated with regard to both the success rate and the median values.

When we consider the success rate, the two sharing-based treatments are domi-

nated because they spectacularly fail in the maze experiment, whereas they obtained good scores in the other setups. To understand this failure, we compared the position of robots at the end of their evaluation (that is, their *ad hoc* behavioral descriptor) in a successful MO-Sharing/Ad-hoc experiment and in an unsuccessful MO-Sharing/Hamming experiment (figure 12). The comparison of figure 12(a) and 12(b) immediately shows a lack of exploration in the MO-div/Hamming experiment: all individuals end in the exact same position, in the attractive local optimum of the maze. We don't know the exact trajectory of these "trapped" individuals, but they very likely stayed in the dead-end of the left part of the maze.

Two hypothesis can be drawn to explain this lack of diversity: (1) the Hamming distance does not differentiate individuals inside of the dead-end from those outside of the dead-end and (2) the sharing function is tuned such that individuals in the dead-end have a good diversity score (a high niche count). Hypothesis (1) can be rejected because MO-div/Hamming led to one of the best result in the same setup (figures 10 and 11). To confirm the second hypothesis, we plotted the niche count in the MO-Sharing/Ad-hoc (successful) and MO-Sharing/Hamming (unsuccessful) experiments (figure 12(c)). From this diagram, it appears that while all the individuals are in the dead-end and have almost the same fitness, they have very different niche count. Some of them even obtain a niche count of 1, meaning that no individual is closer to them than σ (equation 1, section 2.1). As a consequence, we can conclude that the failures probably originate from a non-optimal setting of the σ parameter.

The MO-div/Hamming is a Pareto-optimal trade-off that fails in no setup. While it obtains the best success rate only in the ball collecting experiment, it is the most robust trade-off.

Diversity mechanisms. We observe an advantage of MO-sharing over its single-objective counterpart: MO-Sharing/Ad-hoc dominates Fitness Sharing/Ad-hoc with regard to success rate and median fitness; MO-Sharing/Hamming dominates Fitness. This result suggests that the multi-objective formulation of fitness sharing is advantageous over the classic single-objective fitness sharing.

MO-sharing and MO-div both lead to Pareto-optimal trade-offs (non-dominated solutions), depending on the distance used. Overall, MO-sharing leads to more contrasted results than MO-div, as we obtain either the best scores or one among the worst. MO-div leads to more consistent results except in the Elman/Maze experiment. The contrasted results of MO-sharing probably stems from the choice of the parameter σ, that may not be optimally suited to all experiments.

NEAT. Figures 10 and 11 show that NEAT is more efficient than the control experiment but substantially less efficient than the other encodings combined with behavioral diversity. Additionally, NEAT led to surprisingly worse results than the Elman network with MO-div/Gen., while being better than the MO-Sharing/Gen. and Fitness Sharing/Gen. treatments. Overall, these results show that NEAT improves over a basic direct encoding but this improvement is not sufficient to reach the performance obtained with behavioral diversity.

Best trade-offs. The two best trade-offs are MO-sharing/Ad-hoc and MO-div/Hamming: they are non-dominated with regard to both the median fitness and the success rates. It is unexpected that these two treatments only share the concept of behavioral diversity: they use different distances *and* different diversity mechanisms. Our interpretation is that the sharing function is a better estimation of the density than the mean distance used in MO-div and that MO-sharing is more powerful than basic

fitness sharing. However, setting the σ parameter of the sharing function is critical and may be especially hard when using distances that compare very long and very similar vectors, such as the Hamming distance. Additionally, thanks to its generality, the Hamming distance seems more robust than the task-specific distances. This can be interpreted as follows:

- Hamming distance combined with MO-sharing and a fine-tuned σ could be the best method, provided that there is a way to find this parameter;
- Hamming distance combined with MO-div is not the most efficient approach in every scenario but this parameter-free method is the most robust to improve fitness and success rates.

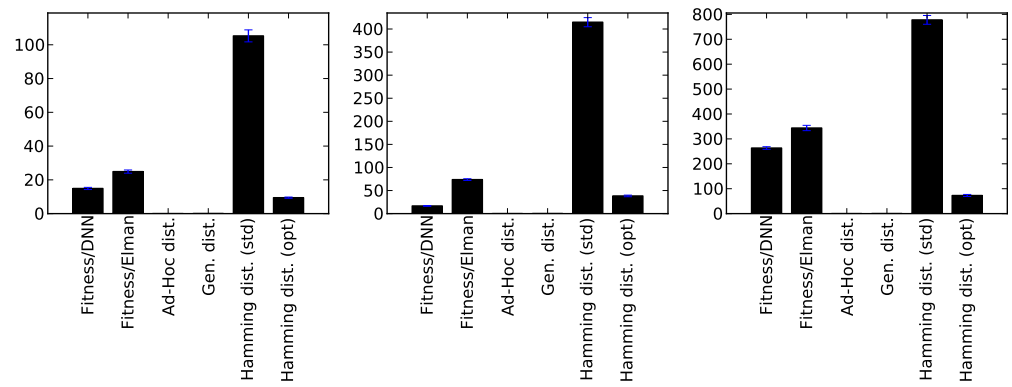
Experimental Conclusions.

- The two best treatments in our experiments are MO-div/Hamming and MO-Sharing/Ad-hoc. Both of them dominate the other treatments *in all the setups*, that is regardless of the genotype or the task.
- The two multi-objective diversity mechanisms dominate the single-objective fitness sharing. This confirms the results published in (De Jong et al., 2001; Bui et al., 2005).
- All MO-div treatments based on behavioral distances dominate the genotypic diversity treatments, the control experiment and NEAT. This confirms that behavioral diversity is a substantial improvement over the state-of-the-art of neuroevolution (Mouret and Doncieux, 2009a,b; Mouret, 2011; Gomez, 2009; Moriguchi and Honiden, 2010; Doncieux and Mouret, 2010).
- The generic Hamming distance can be as efficient as *ad hoc* distances, and even more efficient because it uses less knowledge from the experimenter. This confirms the results suggested in (Doncieux and Mouret, 2010, 2009).
- The sharing function is potentially more efficient than the average distance to the population but a non-optimal value for σ can make it ineffective. The average distance (MO-div) is more robust because it does not depend on any parameter.

6 Discussion

Run-time Overhead. The previous section demonstrated how encouraging behavioral diversity can substantially improve both the median fitness values and the success rates in typical evolutionary robotics experiments. Nevertheless, this improvement requires computing the diversity score of every individual at each generation, a process that could slow down the evolutionary algorithm.

Figure 13 compares the time to evaluate the fitness of each individual to that spent computing their diversity score. The evaluation time of the fitness depends on the size of the neural network, that is why we included the evaluation time obtained with a small neural network (the network used at the first generation in the direct encoding runs, labeled DNN) and that obtained with a large neural network (the Elman network employed in the present study). Whatever the size of the network is, the time spent to evaluate the *ad hoc* diversity and the genotype-based diversity are negligible with regard to the time spent to evaluate the fitness function (figure 13). Using the Hamming distance is more costly because the sequence that describe each individual is very



(a) Deceptive maze (b) Light-seeking (c) Ball collecting

Figure 13: Comparison of the time spent to evaluate the fitness and the diversity of 200 individuals, for 10 generations (fitness is never stopped before T; average over 30 runs, time is in seconds; benchmark launched on a single core of an Intel E5520 at 2.2Ghz). The fitness can be evaluated with a simple perceptron without hidden layers (Fit./DNN, because this is the initial neural network in the DNN experiments) or with the Elman network employed in the particular experiment (Fit./Elman). The diversity can be computed using the genotypic distance (Gen. Dist.) on the Elman network, the *ad hoc* distance (Ad-hoc dist.), the Hamming distance (Hamming dist. (std) for the naive implementation and Hamming dist. (opt) for the optimized implementation).)

long (about 10,000 bits in these experiments) and a good implementation can make a substantial difference. Hence, if the Hamming distance is implemented by comparing arrays of integers, computing the diversity of each individual can be about 5 times slower than evaluating the fitness. However, it is possible to implement it using bit sets (available in the C++ standard library) and thus reduce the time required to evaluate the diversity below that of evaluating the fitness (the difference varies for each experiment, see figure 13).

Parameter Setting. We chose to conduct this study using the same parameters for all the treatments, hence we can set a fair comparison between each of them. Parameters have been set with reasonable values from the literature: a population of 200 individuals is commonly employed to benchmark NSGA-2 (Deb, 2001); to evolve vectors of real numbers, mutation rates of $1/K$, where K is the size of the vector, is also a common value; the direct encoding for neural network uses parameters similar to those previously employed (Doncieux and Mouret, 2010; Mouret and Doncieux, 2009b,a; Mouret, 2011). By using only one reasonable set of parameters for all setups, we assumed that the choice of a behavioral diversity mechanism/distance has more effect on the final outcome than any other parameter. This assumption is based on the previously published results that consistently show large improvements when behavioral diversity is explicitely encouraged, in many different setups and with different encodings and evolutionary algorithms (see section 3.2).

Nonetheless, it cannot be excluded that some treatments could work significantly better with different settings. For example, our experiments show that the value of σ in fitness sharing was not adapted to the Hamming distance in the deceptive maze

setup, but there may exist a value of σ that allows this treatment to succeed in this setup. The best way to overcome this problem is probably to use an automatic tuning procedure (Smit and Eiben, 2009; Birattari et al., 2002) that would guarantee that each algorithm is tuned in a fair manner and according to the same performance criteria. Nevertheless, these procedures require running each algorithm several times (because evolutionary algorithms are stochastic) for a large number of parameter sets. Using such methods would have increased the computational power required for the present study so much that it would have been impossible with our 80 cores cluster[6].

The lack of automatic parameter tuning fortunately does not mean that the present study is meaningless: the assumption that diversity is a more important factor than any parameter has been verified in our experiments. The best illustration is the coherence of the results regardless of the genotype/phenotype map: In all tasks, the ranking of treatments applied to the evolution of Elman networks is the same as the one obtained with the direct encoding (when the treatment can be applied to both genotype/phenotype maps). These two genotypes are vastly different and they consequently use a completely different set of parameters; the fact that they lead to the same rankings of diversity methods shows how the present results do not depend on the precise parameter tuning of the encoding. Furthermore, even if the results on the three tasks are not identical, the general trends are similar. The Pareto analysis allowed us to identify treatments that are consistently better than the state of the art regardless of the genotype and the task. This robustness makes them valid candidates to tackle future tasks even if automatic parameter tuning is too costly.

Lastly, our results show that sharing methods require a fine-tuned σ, hence suggesting that a better tuning could lead these methods to another rank. However, knowing that a bad setting of this parameter can easily lead to substantially worse results than those obtained with a parameter-free method may be more important than knowing that, with an ideal but difficult to know σ, sharing methods are better than some other treatments.

Novelty Search. The present study clearly shows the importance of facilitating the exploration of new behaviors. An interesting follow-up would be to push this approach to the extreme by searching only for novel behaviors, as proposed in the novelty search algorithm (section 3.2). Novelty search may lead to good results in the studied benchmark and this question has to be examined in future work. Nevertheless, whatever will be the result of these investigations, studying behavioral diversity is interesting because it is close to classic objective-based search. Consequently it relies on the numerous successes of evolutionary computation and on the considerable amount of knowledge accumulated about objective-based search; on the contrary, novelty search is still a new approach and its domain of applicability has to be assessed, despite the promising preliminary results.

More importantly, novelty search critically depends on a good behavior characterization to create a gradient. Researchers in evolutionary robotics used to craft fitness function to create a perfect fitness gradient; novelty search users have to craft the behavior distance to create a similar gradient. This last option may be easier for some problems but eventually some distances will be hard to define.

The best of both worlds can be combined in a Pareto-based multi-objective fashion (Mouret and Doncieux, 2009b; Mouret, 2011): one objective for the fitness and an-

[6]The current benchmark required about one week of computational time on a 80-cores cluster of Intel Xeon E5520 (Nehalem generation) at 2.2 Ghz.

other objective for the novelty. Some deficiencies of the fitness gradient can be compensated by the novelty gradient and, conversely, deficiencies in the novelty gradient can be compensated by the fitness gradient. This approach may waste a part of the computational power to maintain a full Pareto front between fitness and novelty, but it has been previously shown that even in tasks designed for novelty search, this multi-objective approach can be qualitatively as efficient as novelty search (Mouret and Doncieux, 2009b; Mouret, 2011). The same studies also investigated multi-objective behavioral diversity (MO-div/Ad-hoc in the present study) and it led to the same success rates as multi-objective novelty search, while being both simpler to implement (no archive is needed) and computationally cheaper. Using Occam razor, behavioral diversity as investigated in the present paper appears to be a simple but powerful improvement over pure fitness-based search. The addition of an archive probably improves efficiency in some tasks but this gain has to be balanced with the computational cost of using an archive.

7 Conclusion

The experimental results presented in this article show that explicitly encouraging behavioral diversity leads to substantial improvements in three typical evolutionary robotics experiments with typical parameters. If we except the few methods whose parameters were not optimally set, all the investigated behavioral distance/diversity mechanism improved the success rates from almost zero, meaning that the problem cannot be tackled without these techniques, to values between 0.5 and 1.0, that is the task was solved most of the time when behavioral diversity was fostered. This result strengthens the published papers about behavioral diversity: when the behavioral diversity is fostered, many tasks that were previously hard to solve become easy to tackle. This makes these techniques a valuable improvement to the state of the art in evolutionary robotics.

Among the investigated diversity mechanisms, multi-objective methods are more efficient that single-objective fitness sharing. Among multi-objective methods, computing the originality of an individual as the average behavioral distance to the current population is more robust than sharing-based methods, because no parameter has to be set. Using *ad hoc* distances was efficient when the behavioral description was well designed, but the Hamming distance between binarized sensory-motor streams proved to be as efficient as *ad hoc* distances and it depends less on the user's choices. Using a genotype-based distance slightly improved the results in some experiments but behavioral distance outperformed genotype-based distances in all experiments.

While we were not interested in comparing encodings, our data incidentally describes how easily each analyzed encoding solves each task. When no diversity was encouraged, some encodings performed better than others; for instance, NEAT was often more efficient than the basic direct encoding. However, encouraging behavioral diversity led to much larger improvements than changing the genotype, to such an extent that most variations between encodings were negligible. This performance difference highlights how a well-designed selective pressure can be more critical to solve a complex problem than a well-designed encoding. In the last fifteen years, considerable efforts have been devoted to finding the best encoding for neural networks (e.g. Gruau (1995); Stanley and Miikkulainen (2002); Hornby and Pollack (2002); Floreano et al. (2008)), while most work on selective pressures focused on task-dependent incremental evolution schemes (e.g. Harvey et al. (1994); Mouret and Doncieux (2008)). The present work offers a new research avenue to improve the selective pressure with

task-independent principles. Future research in this direction could prove to be more critical to solve complex problems in evolutionary robotics than new encodings.

8 Acknowledgments

The authors thank Josh Auerbach and Charles Ollion for their useful feedback on this paper.

References

Abbass, H. A. and Deb, K. (2003). Searching under multi-evolutionary pressures. In *EMO'03: Proceedings of the 2nd international conference on Evolutionary Multi-Criterion Optimization*, volume 2632 of *LNCS*, pages 391–404. Springer.

Aho, A. V., Garey, M. R., and Ullman, J. D. (1972). The transitive reduction of a directed graph. *SIAM Journal on Computing*, 1:131–137.

Barrero, D. F., Camacho, D., and R-Moreno, M. D. (2010). Confidence intervals of success rates in evolutionary computation. In *GECCO '10: Proceedings of the 12th annual conference on Genetic and Evolutionary Computation*, pages 975–976. ACM.

Birattari, M., Stützle, T., Paquete, L., and Varrentrapp, K. (2002). A Racing Algorithm for Configuring Metaheuristics. In *GECCO'02: Proceedings of the 4th annual Genetic and Evolutionary Computation Conference*, pages 11–18. ACM.

Bongard, J. C. and Hornby, G. S. (2010). Guarding Against Premature Convergence while Accelerating Evolutionary Search. In *GECCO'10: Proceedings of the 12th annual conference on Genetic and Evolutionary Computation*, pages 111–118. ACM.

Bui, L., Branke, J., and Abbass, H. (2005). Diversity as a selection pressure in dynamic environments. In *GECCO'05: Proceedings of the 7th annual conference on Genetic and Evolutionary Computation*, pages 1557–1558. ACM.

Bunke, H. and Shearer, K. (1998). A graph distance metric based on the maximal common subgraph. *Pattern Recognition Letters*, 19(3):255–259.

Darwen, P. and Yao, X. (1996). Every niching method has its niche: Fitness sharing and implicit sharing compared. In *Parallel Problem Solving from Nature – PPSN IV*, volume 1141 of *LNCS*, pages 398–407. Springer.

De Jong, E. D., Watson, R. A., and Pollack, J. B. (2001). Reducing bloat and promoting diversity using multi-objective methods. In *GECCO'01: Proceedings of the 3rd annual conference on Genetic and Evolutionary Computation*, pages 11–18. ACM.

Deb, K. (2001). *Multi-objectives optimization using evolutionnary algorithms*. Wiley.

Deb, K., Agrawal, S., Pratab, A., and Meyarivan, T. (2000). A Fast Elitist Non-Dominated Sorting Genetic Algorithm for Multi-Objective Optimization: NSGA-II. In *Paralell Problem Solving form Nature – PPSN VI*, volume 1917 of *LNCS*, pages 849–858. Springer.

Doncieux, S. and Meyer, J.-A. (2003). Evolving neural networks for the control of a lenticular blimp. In *Applications of Evolutionary Computing, EvoWorkshops2003: EvoBIO, EvoCOP, EvoIASP, EvoMUSART, EvoROB, EvoSTIM*, volume 2611 of *LNCS*, pages 626–637. Springer.

Doncieux, S. and Mouret, J.-B. (2009). Single step evolution of robot controllers for sequential tasks. In *GECCO '09: Proceedings of the 11th annual conference on Genetic and evolutionary computation*, pages 1771–1772. ACM.

Doncieux, S. and Mouret, J.-B. (2010). Behavioral diversity measures for Evolutionary Robotics. In *IEEE Congress on Evolutionary Computation (CEC)*, pages 1303–1310.

Doncieux, S., Mouret, J.-B., and Bredeche, N. (2009). Evolutionary robotics: Exploring New Horizons. In *New Horizons in Evolutionary Robotics: Extended Contributions from the 2009 EvoDeRob Workshop*, volume 341 of *Studies in Computational Intelligence*, pages 3–25. Springer.

Elman, J. L. (1993). Learning and development in neural networks: the importance of starting small. *Cognition*, 48(1):71–99.

Floreano, D., Dürr, P., and Mattiussi, C. (2008). Neuroevolution: from architectures to learning. *Evolutionary Intelligence*, 1(1):47–62.

Floreano, D. and Mondada, F. (1994). Automatic creation of an autonomous agent: Genetic evolution of a neural-network driven robot. In *From Animals to Animats 3: Proceedings of the 3rd International Conference on Simulation of Adaptive Behavior (SAB'94)*, pages 421–430. MIT Press.

Fogel, L. G., Owens, A. J., and Walsh, M. J. (1966). *Artificial Intelligence through Simulated Evolution*. Wiley.

Friedrich, T., Oliveto, P. S., Sudholt, D., and Witt, C. (2008). Theoretical analysis of diversity mechanisms for global exploration. In *GECCO'08: Proceedings of the 10th annual conference on Genetic and Evolutionary Computation*, pages 945–952. ACM.

Goldberg, D. E. (1987). Simple genetic algorithms and the minimal, deceptive problem. In *Genetic algorithms and simulated annealing*, pages 74–88. Morgan Kaufman.

Goldberg, D. E., Deb, K., and Horn, J. (1992). Massive multimodality, deception, and genetic algorithms. In *Parallel Problem Solving from Nature – PPSN II*, pages 37–46. Elsevier.

Gomez, F. J. (2009). Sustaining diversity using behavioral information distance. In *GECCO'09: Proceedings of the 11th annual conference on Genetic and Evolutionary Computation*, pages 113–120. ACM.

Gruau, F. (1995). Automatic Definition of Modular Neural Networks. *Adaptive Behaviour*, 3(2):151–183.

Harvey, I., Husbands, P., and Cliff, D. (1994). Seeing the light: artificial evolution; real vision. In *From Animals to Animats 3: Proceedings of the third international conference on Simulation of Adaptive Behavior*, pages 392–401. MIT Press.

Holland, J. (1992). *Adaptation in natural and artificial systems*. The MIT Press.

Hornby, G. S. and Pollack, J. B. (2002). Creating high-level components with a generative representation for body-brain evolution. *Artificial Life*, 8(3):223–246.

Knowles, J., Watson, R., and Corne, D. (2001). Reducing local optima in single-objective problems by multi-objectivization. In *EMO'01: Proceedings of the First International Conference on Evolutionary Multi-Criterion Optimization*, volume 1993 of *LNCS*, pages 269–283. Springer.

Kodjabachian, J. and Meyer, J.-A. (1997). Evolution and development of neural networks controlling locomotion, gradient-following, and obstacle-avoidance in artificial insects. *IEEE Transactions on Neural Networks*, 9(5):796–812.

Lehman, J. and Stanley, K. O. (2008). Exploiting open-endedness to solve problems through the search for novelty. In *Proceedings of Artificial Life XI*, pages 329–336.

Lehman, J. and Stanley, K. O. (2010). Abandoning Objectives: Evolution through the Search for Novelty Alone. *Evolutionary Computation*, 19(2):189–223.

Lipson, H. (2005). Evolutionary Robotics and Open-Ended Design Automation. *Biomimetics*, 17(9):129–155.

Lipson, H. and Pollack, J. (2000). Automatic design and manufacture of robotic lifeforms. *Nature*, 406:974–978.

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

Mahfoud, S. (1997). *Handbook of Evolutionary Computation*, chapter Niching Methods. Taylor & Francis.

Mauldin, M. (1984). Maintaining diversity in genetic search. In *Proceedings of the national conference on artificial intelligence*, pages 247–250.

Meyer, J.-A., Husbands, P., and Harvey, I. (1998). Evolutionary robotics: A survey of applications and problems. In *Proceedings of The First European Workshop on Evolutionary Robotics – EvoRobot98.*, pages 1–21. Springer.

Moriguchi, H. and Honiden, S. (2010). Sustaining Behavioral Diversity in NEAT. In *GECCO'10: Proceedings of the 12th annual conference on Genetic and Evolutionary Computation*, pages 611–618. ACM.

Mouret, J.-B. (2011). Novelty-based Multiobjectivization. In *New Horizons in Evolutionary Robotics: Extended Contributions from the 2009 EvoDeRob Workshop*, volume 341 of *Studies in Computational Intelligence*, pages 139–154. Springer.

Mouret, J.-B. and Doncieux, S. (2008). Incremental Evolution of Animats' Behaviors as a Multi-objective Optimization. In *From Animals to Animats 10: Proceedings of the Tenth International Conference on Simlation of Adaptive Behavior (SAB2008)*, volume 5040 of *LNAI*, pages 210–219. Springer.

Mouret, J.-B. and Doncieux, S. (2009a). Overcoming the bootstrap problem in evolutionary robotics using behavioral diversity. In *IEEE Congress on Evolutionary Computation (CEC)*, pages 1161–1168.

Mouret, J.-B. and Doncieux, S. (2009b). Using Behavioral Exploration Objectives to Solve Deceptive Problems in Neuro-evolution. In *GECCO'09: Proceedings of the 11th annual conference on Genetic and Evolutionary Computation*, pages 627–634. ACM.

Mouret, J.-B. and Doncieux, S. (2010). Sferes_v2 : Evolvin in the Multi-Core World. In *IEEE Congress on Evolutionary Computation (CEC)*, pages 4079–4086.

Mouret, J.-B., Doncieux, S., and Meyer, J.-A. (2006). Incremental evolution of target-following neuro-controllers for flapping-wing animats. In *From Animals to Animats 9: Proceedings of the 9th International Conference on the Simulaiton of Adaptive Behavior (SAB2006)*, volume 4095 of *LNAI*, pages 210–219. Springer.

Nolfi, S. and Floreano, D. (2004). *Evolutionary robotics: The biology, intelligence, and technology of self-organizing machines*. The MIT Press.

Risi, S., Vanderbleek, S. D., Hughes, C. E., and Stanley, K. O. (2009). How novelty search escapes the deceptive trap of learning to learn. In *GECCO'09: Proceedings of the 11th annual conference on Genetic and Evolutionary Computation*, pages 153–160. ACM.

Sareni, B. and Krahenbuhl, L. (1998). Fitness sharing and niching methods revisited. *IEEE Transactions on Evolutionary Computation*, 2(3):97–106.

Schwefel, H. P. (1984). Evolution strategies: A family of non-linear optimization techniques based on imitating some principles of organic evolution. *Annals of Operations Research*, 1(2):165–167.

Smit, S. K. and Eiben, A. E. (2009). Comparing parameter tuning methods for evolutionary algorithms. In *IEEE Congress on Evolutionary Computation (CEC)*, pages 399–406.

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

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

Trujillo, L., Olague, G., Lutton, E., and De Vega, F. F. (2008a). Behavior-based speciation for evolutionary robotics. In *GECCO'08: Proceedings of the 10th annual conference on Genetic and Evolutionary Computation*, pages 297–298. ACM.

Trujillo, L., Olague, G., Lutton, E., and De Vega, F. F. (2008b). Discovering several robot behaviors through speciation. In *Application of Evolutionary Computing: 4th European Workshop on Bio-Inspired Heuristics for Design Automation*, volume 2279 of *LNCS*, pages 165–174. Springer.

Walker, J., Garrett, S., and Wilson, M. (2003). Evolving Controllers for Real Robots: A Survey of the Literature. *Adaptive Behavior*, 11(3):179–203.

Yuan, B. and Gallagher, M. (2009). An improved small-sample statistical test for comparing the success rates of evolutionary algorithms. In *GECCO'09: proceedings of the 11th annual conference on Genetic and Evolutionary Computation*, pages 1879–1880. ACM.

Zhang, K., Statman, R., and Shasha, D. (1992). On the editing distance between unordered labeled trees. *Information Processing Letters*, 42(3):133–139.

A Parameters

Source code is available on the following website:
`http://www.isir.fr/evorob_db`.

A.1 NSGA-2

population size:	200
number of generations:	5000

A.2 Direct Encoding (DNN)

initialization	fully connected feed-forward network without hidden layer
mutation rate, add connection	0.15
mutation rate, remove connection	0.05
mutation rate, change connection	0.15
mutation rate, add neuron	0.05
mutation rate, remove neuron	0.05
weight mutation type	polynomial (Deb, 2001)
minimum weight	-5
maximum weight	5
no crossover	

A.3 Elman Network

mutation type	polynomial (Deb, 2001)
mutation rate (for each parameter)	$10/N$, where N is the number of parameters
η_m (parameter of the polynomial mutation)	15
minimum weight and bias	-5
maximum weight and bias	5
no crossover	

A.4 Fitness Sharing

σ	0.05

A.5 NEAT

We used the parameters from the NEAT archive, except for the size of the population (200).

PopulationSize	200.0
MaxGenerations	5001.0
DisjointCoefficient	1.0
ExcessCoefficient	1.0
WeightDifferenceCoefficient	0.8
FitnessCoefficient	0.0
CompatibilityThreshold	4.0
CompatibilityModifier	0.3
SpeciesSizeTarget	8.0
DropoffAge	2000.0
AgeSignificance	1.0
SurvivalThreshold	0.4
MutateAddNodeProbability	0.005
MutateAddLinkProbability	0.10
MutateDemolishLinkProbability	0.00
MutateLinkWeightsProbability	0.6
MutateOnlyProbability	0.25
MutateLinkProbability	0.1
AllowAddNodeToRecurrentConnection	0.0
SmallestSpeciesSizeWithElitism	5.0
MutateSpeciesChampionProbability	0.0
MutationPower	2.5
AdultLinkAge	18.0
AllowRecurrentConnections	1.0
AllowSelfRecurrentConnections	0.0
ForceCopyGenerationChampion	1.0
LinkGeneMinimumWeightForPhentoype	0.0
GenerationDumpModulo	10.0
RandomSeed	-1.0
ExtraActivationFunctions	1.0
AddBiasToHiddenNodes	0.0
SignedActivation	0.0
ExtraActivationUpdates	9.0
OnlyGaussianHiddenNodes	0.0
ExperimentType	0.0

B Statistical Tests

	(0)	(1)	(2)	(3)	(4)	(5)	(6)	(7)
(0) MO-Sharing/Ad-Hoc		0.0	0.0	0.165	0.0	0.0	0.192	0.069
(1) Fit.Sharing/Ad-Hoc	0.0		0.0	0.0	0.0	0.0	0.0	0.011
(2) No div.	0.0	0.0		0.0	0.0	0.006	0.0	0.0
(3) MO-div/Hamming	0.165	0.0	0.0		0.001	0.0	0.364	0.005
(4) MO-Sharing/Hamming	0.0	0.0	0.0	0.001		0.0	0.0	0.0
(5) NEAT	0.0	0.0	0.006	0.0	0.0		0.0	0.0
(6) Fit.Sharing/Hamming	0.192	0.0	0.0	0.364	0.0	0.0		0.008
(7) MO-div/Ad-Hoc	0.069	0.011	0.0	0.005	0.0	0.0	0.008	

Table 4: P-value of Mann-Whitney U-test for the light-seeking experiments and the DNN genotype (direct encoding).

	(0)	(1)	(2)	(3)	(4)	(5)	(6)	(7)	(8)
(0) MO-Sharing/Ad-Hoc		0.0	0.0	0.0	0.007	0.375	0.0	0.007	0.001
(1) Fit.Sharing/Ad-Hoc	0.0		0.03	0.0	0.006	0.0	0.0	0.424	0.08
(2) Fit.Sharing/Hamming	0.0	0.03		0.0	0.2	0.0	0.0	0.212	0.389
(3) No div.	0.0	0.0	0.0		0.0	0.0	0.321	0.0	0.0
(4) MO-div/Hamming	0.007	0.006	0.2	0.0		0.011	0.0	0.044	0.188
(5) MO-Sharing/Hamming	0.375	0.0	0.0	0.0	0.011		0.0	0.008	0.001
(6) Fit.Sharing/Gen.	0.0	0.0	0.0	0.321	0.0	0.0		0.0	0.0
(7) MO-div/Gen.	0.007	0.424	0.212	0.0	0.044	0.008	0.0		0.225
(8) MO-div/Ad-Hoc	0.001	0.08	0.389	0.0	0.188	0.001	0.0	0.225	

Table 5: P-value of Mann-Whitney U-test for the light-seeking experiments and the Elman network.

	(0)	(1)	(2)	(3)	(4)	(5)	(6)	(7)
(0) MO-Sharing/Ad-Hoc		0.23	0.0	0.002	0.0	0.0	0.0	0.022
(1) Fit.Sharing/Ad-Hoc	0.23		0.0	0.015	0.0	0.0	0.0	0.045
(2) No div.	0.0	0.0		0.0	0.0	0.155	0.222	0.0
(3) MO-div/Hamming	0.002	0.015	0.0		0.0	0.0	0.0	0.351
(4) MO-Sharing/Hamming	0.0	0.0	0.0	0.0		0.447	0.002	0.0
(5) NEAT	0.0	0.0	0.155	0.0	0.447		0.292	0.0
(6) Fit.Sharing/Hamming	0.0	0.0	0.222	0.0	0.002	0.292		0.0
(7) MO-div/Ad-Hoc	0.022	0.045	0.0	0.351	0.0	0.0	0.0	

Table 6: P-value of Mann-Whitney U-test for the deceptive maze experiments and the DNN genotype (direct encoding).

	(0)	(1)	(2)	(3)	(4)	(5)	(6)	(7)	(8)	(9)
(0) MO-Sharing/Ad-Hoc		0.313	0.0	0.0	0.204	0.0	0.0	0.0	0.007	0.482
(1) Fit.Sharing/Ad-Hoc	0.313		0.0	0.0	0.248	0.0	0.0	0.0	0.003	0.315
(2) Fit.Sharing/Hamming	0.0	0.0		0.348	0.0	0.001	0.302	0.488	0.0	0.0
(3) No div.	0.0	0.0	0.348		0.0	0.002	0.211	0.441	0.0	0.0
(4) MO-div/Hamming	0.204	0.248	0.0	0.0		0.0	0.0	0.0	0.031	0.31
(5) MO-Sharing/Hamming	0.0	0.0	0.001	0.002	0.0		0.029	0.009	0.0	0.0
(6) MO-Sharing/Gen.	0.0	0.0	0.302	0.211	0.0	0.029		0.283	0.0	0.0
(7) Fit.Sharing/Gen.	0.0	0.0	0.488	0.441	0.0	0.009	0.283		0.0	0.0
(8) MO-div/Gen.	0.007	0.003	0.0	0.0	0.031	0.0	0.0	0.0		0.009
(9) MO-div/Ad-Hoc	0.482	0.315	0.0	0.0	0.31	0.0	0.0	0.0	0.009	

Table 7: P-value of Mann-Whitney U-test for the deceptive maze experiments and the Elman network.

	(0)	(1)	(2)	(3)	(4)	(5)	(6)	(7)
(0) Fit.Sharing/Hamming		0.019	0.154	0.001	0.006	0.005	0.0	0.218
(1) MO-Sharing/Hamming	0.019		0.001	0.045	0.336	0.0	0.0	0.057
(2) Fit.Sharing/Ad-Hoc	0.154	0.001		0.0	0.0	0.017	0.0	0.03
(3) MO-div/Hamming	0.001	0.045	0.0		0.063	0.0	0.0	0.002
(4) MO-Sharing/Ad-Hoc	0.006	0.336	0.0	0.063		0.0	0.0	0.013
(5) NEAT	0.005	0.0	0.017	0.0	0.0		0.0	0.0
(6) No div.	0.0	0.0	0.0	0.0	0.0	0.0		0.0
(7) MO-div/Ad-Hoc	0.218	0.057	0.03	0.002	0.013	0.0	0.0	

Table 8: P-value of Mann-Whitney U-test for ball collecting experiments and the DNN genotype (direct encoding).

	(0)	(1)	(2)	(3)	(4)	(5)	(6)	(7)	(8)	(9)
(0) MO-div/Gen.		0.0	0.01	0.026	0.006	0.0	0.005	0.434	0.361	0.027
(1) Fit.Sharing/Hamming	0.0		0.0	0.048	0.181	0.0	0.0	0.0	0.002	0.0
(2) MO-Sharing/Hamming	0.01	0.0		0.0	0.0	0.048	0.476	0.002	0.005	0.141
(3) Fit.Sharing/Gen.	0.026	0.048	0.0		0.227	0.0	0.0	0.02	0.069	0.0
(4) No div.	0.006	0.181	0.0	0.227		0.0	0.0	0.003	0.018	0.0
(5) MO-div/Hamming	0.0	0.0	0.048	0.0	0.0		0.033	0.0	0.0	0.001
(6) MO-Sharing/Ad-Hoc	0.005	0.0	0.476	0.0	0.0	0.033		0.001	0.002	0.129
(7) MO-Sharing/Gen.	0.434	0.0	0.002	0.02	0.003	0.0	0.001		0.386	0.007
(8) Fit.Sharing/Ad-Hoc	0.361	0.002	0.005	0.069	0.018	0.0	0.002	0.386		0.013
(9) MO-div/Ad-Hoc	0.027	0.0	0.141	0.0	0.0	0.001	0.129	0.007	0.013	

Table 9: P-value of Mann-Whitney U-test for ball collecting experiments and the Elman network.

	(0)	(1)	(2)	(3)	(4)	(5)	(6)	(7)
(0) MO-Sharing/Ad-Hoc		0.052	0.0	0.299	0.052	0.0	0.052	1.0
(1) Fit.Sharing/Ad-Hoc	0.052		0.0	0.612	1.0	0.0	1.0	0.052
(2) No div.	0.0	0.0		0.0	0.0	1.0	0.0	0.0
(3) MO-div/Hamming	0.299	0.612	0.0		0.612	0.0	0.612	0.299
(4) MO-Sharing/Hamming	0.052	1.0	0.0	0.612		0.0	1.0	0.052
(5) NEAT	0.0	0.0	1.0	0.0	0.0		0.0	0.0
(6) Fit.Sharing/Hamming	0.052	1.0	0.0	0.612	1.0	0.0		0.052
(7) MO-div/Ad-Hoc	1.0	0.052	0.0	0.299	0.052	0.0	0.052	

Table 10: P-value of the Fisher's exact test in the light-seeking experiments with the DNN genotype (direct encoding).

	(0)	(1)	(2)	(3)	(4)	(5)	(6)	(7)	(8)	(9)
(0) MO-Sharing/Ad-Hoc		0.112	0.731	0.0	0.506	1.0	0.0	0.0	0.0	0.125
(1) Fit.Sharing/Ad-Hoc	0.112		0.024	0.0	0.011	0.237	0.0	0.0	0.0	0.001
(2) Fit.Sharing/Hamming	0.731	0.024		0.0	1.0	0.472	0.0	0.0	0.001	0.382
(3) No div.	0.0	0.0	0.0		0.0	0.0	1.0	0.424	0.021	0.0
(4) MO-div/Hamming	0.506	0.011	1.0	0.0		0.299	0.0	0.0	0.002	0.567
(5) MO-Sharing/Hamming	1.0	0.237	0.472	0.0	0.299		0.0	0.0	0.0	0.057
(6) MO-Sharing/Gen.	0.0	0.0	0.0	1.0	0.0	0.0		0.424	0.021	0.0
(7) Fit.Sharing/Gen.	0.0	0.0	0.0	0.424	0.0	0.0	0.424		0.233	0.0
(8) MO-div/Gen.	0.0	0.0	0.001	0.021	0.002	0.0	0.021	0.233		0.019
(9) MO-div/Ad-Hoc	0.125	0.001	0.382	0.0	0.567	0.057	0.0	0.0	0.019	

Table 11: P-value of the Fisher's exact test in the light-seeking experiments with the Elman network.

	(0)	(1)	(2)	(3)	(4)	(5)	(6)	(7)
(0) MO-Sharing/Ad-Hoc		0.492	0.0	0.021	0.0	0.0	0.0	0.08
(1) Fit.Sharing/Ad-Hoc	0.492		0.0	0.001	0.0	0.0	0.0	0.005
(2) No div.	0.0	0.0		0.0	0.237	0.0	0.237	0.0
(3) MO-div/Hamming	0.021	0.001	0.0		0.0	0.038	0.0	0.779
(4) MO-Sharing/Hamming	0.0	0.0	0.237	0.0		0.03	1.0	0.0
(5) NEAT	0.0	0.0	0.0	0.038	0.03		0.03	0.009
(6) Fit.Sharing/Hamming	0.0	0.0	0.237	0.0	1.0	0.03		0.0
(7) MO-div/Ad-Hoc	0.08	0.005	0.0	0.779	0.0	0.009	0.0	

Table 12: P-value of the Fisher's exact test in the deceptive maze experiments with the DNN genotype (direct encoding).

	(0)	(1)	(2)	(3)	(4)	(5)	(6)	(7)	(8)	(9)
(0) MO-Sharing/Ad-Hoc		1.0	0.0	0.0	0.492	0.0	0.0	0.0	0.001	1.0
(1) Fit.Sharing/Ad-Hoc	1.0		0.0	0.0	0.492	0.0	0.0	0.0	0.001	1.0
(2) Fit.Sharing/Hamming	0.0	0.0		0.052	0.0	1.0	0.424	0.424	0.0	0.0
(3) No div.	0.0	0.0	0.052		0.0	0.112	0.492	0.492	0.0	0.0
(4) MO-div/Hamming	0.492	0.492	0.0	0.0		0.0	0.0	0.0	0.021	0.492
(5) MO-Sharing/Hamming	0.0	0.0	1.0	0.112	0.0		0.671	0.671	0.0	0.0
(6) MO-Sharing/Gen.	0.0	0.0	0.424	0.492	0.0	0.671		1.0	0.0	0.0
(7) Fit.Sharing/Gen.	0.0	0.0	0.424	0.492	0.0	0.671	1.0		0.0	0.0
(8) MO-div/Gen.	0.001	0.001	0.0	0.0	0.021	0.0	0.0	0.0		0.001
(9) MO-div/Ad-Hoc	1.0	1.0	0.0	0.0	0.492	0.0	0.0	0.0	0.001	

Table 13: P-value of the Fisher's exact test in the deceptive maze experiments with the Elman network.

	(0)	(1)	(2)	(3)	(4)	(5)	(6)	(7)
(0) Fit.Sharing/Hamming		0.295	0.382	0.001	0.295	0.001	0.021	1.0
(1) MO-Sharing/Hamming	0.295		0.029	0.029	1.0	0.0	0.0	0.187
(2) Fit.Sharing/Ad-Hoc	0.382	0.029		0.0	0.029	0.024	0.254	0.552
(3) MO-div/Hamming	0.001	0.029	0.0		0.029	0.0	0.0	0.0
(4) MO-Sharing/Ad-Hoc	0.295	1.0	0.029	0.029		0.0	0.0	0.187
(5) NEAT	0.001	0.0	0.024	0.0	0.0		0.492	0.002
(6) No div.	0.021	0.0	0.254	0.0	0.0	0.492		0.042
(7) MO-div/Ad-Hoc	1.0	0.187	0.552	0.0	0.187	0.002	0.042	

Table 14: P-value of the Fisher's exact test in the ball collecting experiments with the DNN genotype (direct encoding).

	(0)	(1)	(2)	(3)	(4)	(5)	(6)	(7)	(8)	(9)
(0) MO-div/Gen.		0.011	0.267	0.033	1.0	0.145	0.567	0.145	0.145	0.748
(1) Fit.Sharing/Hamming	0.011		0.0	0.0	0.024	0.492	0.001	0.492	0.492	0.052
(2) MO-Sharing/Hamming	0.267	0.0		0.438	0.158	0.005	0.789	0.005	0.005	0.084
(3) MO-div/Hamming	0.033	0.0	0.438		0.015	0.0	0.192	0.0	0.0	0.006
(4) Fit.Sharing/Ad-Hoc	1.0	0.024	0.158	0.015		0.254	0.382	0.254	0.254	1.0
(5) Fit.Sharing/Gen.	0.145	0.492	0.005	0.0	0.254		0.021	1.0	1.0	0.424
(6) MO-Sharing/Ad-Hoc	0.567	0.001	0.789	0.192	0.382	0.021		0.021	0.021	0.233
(7) MO-Sharing/Gen.	0.145	0.492	0.005	0.0	0.254	1.0	0.021		1.0	0.424
(8) No div.	0.145	0.492	0.005	0.0	0.254	1.0	0.021	1.0		0.424
(9) MO-div/Ad-Hoc	0.748	0.052	0.084	0.006	1.0	0.424	0.233	0.424	0.424	

Table 15: P-value of the Fisher's exact test in the ball collecting experiments with the Elman network.