Evolving Plastic Neural Networks with Novelty Search

Sebastian Risi*
Charles E. Hughes
Kenneth O. Stanley
Evolutionary Complexity Research Group
School of Electrical Engineering and Computer Science
University of Central Florida
4000 Central Florida Blvd.
Orlando, FL 32816-2362 USA

In: *Adaptive Behavior* journal 18(6), pages 470-491, London: SAGE, 2010

*Correspondence to: Sebastian Risi, School of EECS, University of Central Florida, Orlando, USA.
E-Mail: sebastian.risi@gmail.com
Tel.: +1 407 929 5113

Abstract

Biological brains can adapt and learn from past experience. Yet neuroevolution, i.e. automatically creating artificial neural networks (ANNs) through evolutionary algorithms, has sometimes focused on static ANNs that cannot change their weights during their lifetime. A profound problem with evolving adaptive systems is that learning to learn is highly deceptive. Because it is easier at first to improve fitness without evolving the ability to learn, evolution is likely to exploit domain-dependent static (i.e. non-adaptive) heuristics. This paper analyzes this inherent deceptiveness in a variety of different dynamic, reward-based learning tasks, and proposes a way to escape the deceptive trap of static policies based on the *novelty search* algorithm. The main idea in novelty search is to abandon objective-based fitness and instead simply search *only* for novel behavior, which avoids deception entirely. A series of experiments and an in-depth analysis show how behaviors that could potentially serve as a stepping stone to finding adaptive solutions are discovered by novelty search yet are missed by fitness-based search. The conclusion is that novelty search has the potential to foster the emergence of adaptive behavior in reward-based learning tasks, thereby opening a new direction for research in evolving plastic ANNs.

Keywords Novelty Search, Neural Networks, Adaptation, Learning, Neuromodulation, Neuroevolution

1 Introduction

Neuroevolution (NE), i.e. evolving artificial neural networks (ANNs) through evolutionary algorithms, has shown promise in a variety of control tasks (Floreano, Dürr, & Mattiussi, 2008; Reil & Husbands, 2002; Stanley, Bryant, & Miikkulainen, 2005; Stanley & Miikkulainen, 2002; Yao, 1999). However, the synaptic connections in ANNs produced by NE are normally static, which may limit the adaptive dynamics the network can display during its lifetime (Blynel & Floreano, 2002). While some tasks do not require the network to change its behavior, many domains would benefit from *online adaptation*. In other words, whereas evolution produces phylogenetic adaptation, learning gives the individual the possibility to react much faster to environmental changes by modifying its behavior during its lifetime. For example, a robot that is physically damaged should be able to adapt to its new circumstances without the need to re-evolve its neurocontroller. In this way, when the environment changes from what was encountered during evolution, adapting online is often necessary to maintain performance.

There is much evidence that evolution and learning are both integral to the success of biological evolution (Mayley, 1997; Nolfi & Floreano, 1999) and that lifetime learning itself can help to guide evolution to higher fitness (Hinton & Nowlan, 1987). Thus NE can benefit from combining these complementary forms of adaptation by evolving ANNs with synaptic plasticity driven by local learning rules (Baxter, 1992; Floreano & Urzelai, 2000; Stanley, Bryant, & Miikkulainen, 2003). Synaptic plasticity allows the network to change its internal connection weights based on experience during its lifetime. It also resembles the way organisms in nature, which possess plastic nervous systems, cope with changing and unpredictable environments (Floreano & Urzelai, 2000; Niv, Joel, Meilijson, & Ruppin, 2002; Soltoggio, Bullinaria, Mattiussi, Dürr, & Floreano, 2008). In this paper, the term *plastic ANNs* refers in particular to ANNs that can accordingly change their connection weights during their lifetime, while the term *adaptive ANNs* refers to the larger class of ANNs that can adapt through any means (e.g. through recurrent connections). In a recent demonstration of the power of the plastic approach, Soltoggio et al. (2008) evolved plastic Hebbian networks with *neuromodulation*, i.e. in which some neurons can enhance or dampen the neural plasticity of their target nodes, that acquired the ability to memorize the position of a reward from previous trials in a T-Maze learning problem. However, evolving adaptive controllers for more complicated tasks has proven difficult in part because learning to learn is deceptive, which is the focus of this paper.

Objective functions often exhibit the pathology of local optima (Goldberg, 2007; Mitchell, Forrest, & Holland, 1991), and the more ambitious the goal, the more likely it is that search can be deceived by suboptimal solutions (Lehman & Stanley, 2008, 2010a). In particular, if fitness does not reward the stepping stones that lead to the final solution in the search space, fitness-based search may be led astray. Deception in domains that require adaptation is particularly pathological for two primary reasons: (1) Reaching a mediocre fitness through *non-adaptive* behavior is often relatively easy, but any further improvement requires an improbable leap to sophisticated adaptive behavior, and (2) only sparse feedback on the acquisition of adaptive behavior is available from an objective-based performance measure. Because it is easier at first to improve fitness without evolving the ability to learn, objective functions may sometimes exploit domain-dependent static (i.e. non-adaptive) heuristics that can lead them further away from the adaptive solution in the genotypic search space, as analysis in this paper will confirm. Because of the problem of deception in adaptive domains, prior experiments in evolving plastic ANNs have needed to be carefully designed to ensure that no non-adaptive heuristics exist that could potentially lead evolution prematurely astray. This awkward requirement has significantly limited the scope of domains amenable to adaptive evolution and stifled newcomers from entering the research area.

To remedy this situation and open up the range of problems amenable to evolving adaptation, this paper proposes that the *novelty search* algorithm (Lehman & Stanley, 2008), which abandons the traditional notion of objective-based fitness, circumvents the deception inherent in adaptive domains. Instead of searching for a final objective behavior, novelty search rewards finding any instance whose behavior is significantly different from what has been discovered before. Surprisingly, this radical form of search has been shown to outperform traditional fitness-based search in several deceptive domains (Lehman & Stanley, 2008, 2010b, 2010c; Mouret, 2009), suggesting that it may be applicable to addressing the problem of deception in evolving plastic ANNs, which is the focus of this paper.

To demonstrate the potential of this approach, this paper first compares novelty search to fitness-based evolution in a dynamic, reward-based single T-Maze scenario first studied in the context of NE by Blynel and Floreano (2003) and further investigated by Soltoggio et al. (2008) to demonstrate the advantage of neuromodulated plasticity. In this scenario, the reward location is a variable factor in the environment that the agent must learn to exploit. Because the aim of this paper is to show that novelty search solves particular difficult problems in the

evolution of *plastic* networks and it has been shown that neuromodulation is critical to those domains (Soltoggio et al., 2008), all evolved ANNs employ this most effective form of plasticity.

Counterintuitively, novelty search significantly outperforms regular fitness-based search in the T-Maze learning problem because it returns more information about how behavior changes throughout the search space. To explain this result and understand the nature of deception in this domain, the locus of deception in the T-Maze is uncovered through a *Sammon's Mapping* visualization that shows how fitness-based search and novelty search navigate the high-dimensional genotypic search space. The main result is that genotypes that are leveraged by novelty search as stepping stones can in fact lead fitness-based search astray.

Furthermore, deceptiveness in reward-based scenarios can increase when learning is only needed in a low percentage of trials. In that case, evolution is trapped in local optima that do not require learning at all because high fitness is achieved in the majority of trials. By varying the number of times the reward location changes in the T-Maze domain, the effect of adaptation on the fitness function can be controlled to make the domain more or less deceptive for objective-based fitness. While fitness-based search performs worse with increased domain deception (as one would expect), novelty search is *not significantly affected*, suggesting an intriguing new approach to evolving adaptive behavior. The interesting aspect of this observation is that novelty search both *solves* the problem and solves it in a *general* way despite lacking any incentive to do so.

Additional experiments in the more complicated double T-Maze domain and a bee foraging task add further evidence to the hypothesis that novelty search can effectively overcome the deception inherent in many dynamic, reward-based scenarios. In these domains, novelty search still significantly outperforms fitness-based search under an increased behavioral search space and raised domain complexity.

The paper begins with a review of novelty search and evolving adaptive ANNs in the next section. The T-Maze domain is then described in Section 3, followed by the experimental design in Section 4. Results are presented in Section 5 and a detailed analysis of the inherent deception in the T-Maze domain is conducted in Section 6. The double T-Maze and bee domain experiments are described in Section 7. The paper concludes with a discussion and ideas for future work in Section 8.

2 Background

This section first reviews novelty search, which is the proposed solution to deception in the evolution of learning. Then an overview of evolving plastic ANNs is given, focusing on the neuromodulation-based model followed in this paper. The section concludes with a description of NEAT, which is augmented in this paper to encode neuromodulated plasticity.

2.1 The Search for Novelty

The problem with the objective fitness function in evolutionary computation is that it does not necessarily reward the intermediate stepping stones that lead to the objective. The more ambitious the objective, the harder it is to identify *a priori* these stepping stones.

This paper hypothesizes that evolving plastic ANNs is especially susceptible to missing the essential intermediate stepping stones for fitness-based search and therefore highly deceptive. Reaching a mediocre fitness through non-adaptive behavior is relatively easy, but any further improvement requires sophisticated adaptive behavior with only sparse feedback from an objective-based performance measure. Such deception is inherent in most dynamic, reward-based scenarios.

A potential solution to this problem is novelty search, which is a recent method for avoiding deception based on the radical idea of ignoring the objective (Lehman & Stanley, 2008, 2010a). The idea is to identify novelty as a proxy for stepping stones. That is, instead of searching for a final objective, the learning method is rewarded for finding any behavior whose functionality is significantly different from what has been discovered before. Thus, instead of an objective function, search employs a *novelty metric*. That way, no attempt is made to measure overall progress. In effect, such a process gradually accumulates novel behaviors. This idea is also related to the concept of *curiosity* and seeking novelty in reinforcement learning research (Schmidhuber, 2003, 2006).

Although it is counterintuitive, novelty search was actually *more effective* at finding the objective than a traditional objective-based fitness function in a deceptive navigation domain that requires an agent to navigate through a maze to reach a specific goal location (Lehman & Stanley, 2008; Mouret, 2009), in evolving biped locomotion (Lehman & Stanley, 2010a), and in evolving a program for an artificial ant benchmark task (Lehman & Stanley, 2010b). Thus novelty search might be a solution to the longstanding problem with training for adaptation.

The next section describes the novelty search algorithm (Lehman & Stanley, 2008) in more detail.

2.1.1 The Novelty Search Algorithm

Evolutionary algorithms are well-suited to novelty search because the population that is central to such algorithms naturally covers a wide range of expanding behaviors. In fact, tracking novelty requires little change to any evolutionary algorithm aside from replacing the fitness function with a novelty metric.

The novelty metric measures how different an individual is from other individuals, creating a constant pressure to do something new. The key idea is that instead of rewarding performance on an objective, the novelty search rewards diverging from prior behaviors. Therefore, novelty needs to be measured.

There are many potential ways to measure novelty by analyzing and quantifying behaviors to characterize their differences. Importantly, like the fitness function, this measure must be fitted to the domain.

The novelty of a newly generated individual is computed with respect to the observed *behaviors* (i.e. *not* the genotypes) of an *archive* of past individuals whose behaviors were highly novel when they originated. In addition, if the evolutionary algorithm is steady state (i.e. one individual is replaced at a time) then the current population can also supplement the archive by representing the most recently visited points. The aim is to characterize how far away the new individual is from the rest of the population and its predecessors in *novelty space*, i.e. the space of unique behaviors. A good metric should thus compute the *sparseness* at any point in the novelty space. Areas with denser clusters of visited points are less novel and therefore rewarded less.

A simple measure of sparseness at a point is the average distance to the k-nearest neighbors of that point, where k is a fixed parameter that is determined experimentally. Intuitively, if the average distance to a given point's nearest neighbors is large then it is in a sparse area; it is in a dense region if the average distance is small. The sparseness ρ at point x is given by

$$\rho(x) = \frac{1}{k} \sum_{i=1}^{k} \text{dist}(x, \mu_i), \tag{1}$$

where μ_i is the ith-nearest neighbor of x with respect to the distance metric *dist*, which is

a domain-dependent measure of behavioral difference between two individuals in the search space. The nearest neighbors calculation must take into consideration individuals from the current population and from the permanent archive of novel individuals. Candidates from more sparse regions of this behavioral search space then receive higher novelty scores. It is important to note that this novelty space cannot be explored purposefully, that is, it is not known *a priori* how to enter areas of low density just as it is not known a priori how to construct a solution close to the objective. Thus moving through the space of novel behaviors requires exploration. In effect, because novelty is measured relative to other individuals in evolution, it is driven by a coevolutionary dynamic.

If novelty is sufficiently high at the location of a new individual, i.e. above some minimal threshold ρ_{min}, then the individual is entered into the permanent archive that characterizes the distribution of prior solutions in novelty space, similarly to archive-based approaches in coevolution (De Jong, 2004). The current generation plus the archive give a comprehensive sample of where the search has been and where it currently is; that way, by attempting to maximize the novelty metric, the gradient of search is simply towards what is new, with no other explicit objective. To ensure that the archive continues to push the search to new areas and does not expand too fast, the threshold ρ_{min} is adjusted dynamically (e.g. by lowering ρ_{min} if no new individuals are added during a certain number of evaluations) to maintain a healthy rate of expansion.

It is important to note that novelty search resembles prior diversity maintenance techniques (i.e. speciation) popular in evolutionary computation (Darwen & Yao, 1996; Goldberg & Richardson, 1987; Hornby, 2006; Hu, Goodman, Seo, Fan, & Rosenberg, 2005; Mahfoud, 1995). The most well known are variants of fitness sharing (Darwen & Yao, 1996; Goldberg & Richardson, 1987). These also in effect open up the search by reducing selection pressure. However, in these methods, as in Hutter's fitness uniform selection (Hutter & Legg, 2006), the search is still ultimately guided by the fitness function. Diversity maintenance simply keeps the population more diverse than it otherwise would be. Also, most diversity maintenance techniques measure genotypic diversity as opposed to behavioral diversity (Darwen & Yao, 1996; Mahfoud, 1995). In contrast, novelty search takes the radical step of *only* rewarding behavioral diversity with no concept of fitness or a final objective, inoculating it to traditional deception.

Other related methods seek to accelerate search through neutral networks by recognizing neutral areas in the search space (Stewart, 2001; Barnett, 2001). Stewart (2001) explicitly

rewards drifting further away in genotype space from the center of the population once a neutral network is encountered. Similarly, Barnett (2001) seeks to accelerate movement across a neutral network of equal objective fitness by reducing the population to one individual. However, identifying when the search is actually stalled may be difficult in practice and while such approaches potentially decrease the search complexity, finding the objective might still take a long time depending on the deceptiveness of the task.

It is also important to note that novelty search is not a random walk; rather, it explicitly maximizes novelty. Because novelty search includes an archive that accumulates a record of where search has been, backtracking, which can happen in a random walk, is effectively avoided in behavioral spaces of any dimensionality. In this way, novelty search resembles tabu search (Glover & Laguna, 1997), which keeps a list of potential solutions to avoid repeatedly visiting the same points. However, tabu search still tries to measure overall progress and therefore can be potentially led astray by deception.

The novelty search approach in general allows any behavior characterization and any novelty metric. Although generally applicable, novelty search is best suited to domains with deceptive fitness landscapes, intuitive behavioral characterization, and domain constraints on possible expressible behaviors.

Changing the way the behavior space is characterized and the way characterizations are compared will lead to different search dynamics, similarly to how researchers now change the fitness function to improve the search. The intent is not to imply that setting up novelty search is easier than objective-based search. Rather, once novelty search is set up, the hope is that it can find solutions beyond what even a sophisticated objective-based search can currently discover. Thus the effort is justified in its returns.

In summary, novelty search depends on the following four main concepts:

- Individuals' behaviors are *characterized* so that they can be compared.
- The novelty of an individual is computed with respect to observed *behaviors* of other individuals and not others' genotypes.
- Novelty search replaces the fitness function with a novelty metric that computes the sparseness at any point in the novelty space.
- An archive of past individuals is maintained whose behaviors were highly novel.

The evolutionary algorithm that evolves neuromodulated plastic networks (explained later in Section 2.3) through novelty search in this paper is NeuroEvolution of Augmenting Topologies (NEAT; Stanley & Miikkulainen, 2002), which offers the ability to discover minimal effective plastic topologies.

The next section reviews the evolution of adaptive ANNs and details the model for neuromodulated plasticity in this paper, which is followed by an explanation of NEAT.

2.2 Evolving Adaptive Neural Networks

Researchers have been evolving adaptive ANNs for more than fifteen years. Early work often focused on combining the built-in adaptive capabilities of backpropagation with NE. For example, Nolfi and Parisi (1993, 1996) evolved self-teaching networks that trained a motor control network through backpropagation from the outputs of a teaching subnetwork. In separate work, they evolved a network that learns through backpropagation to predict what it would see after moving around in its environment (Nolfi, Parisi, & Elman, 1994). Learning to predict the next state during the network's lifetime was shown to enhance performance in a foraging task. Interestingly, Chalmers (1990) evolved a global learning rule (i.e. a rule that applies to every connection) and discovered that the evolved rule was similar to the well-known delta rule used in backpropagation. Furthermore, McQuesten and Miikkulainen (1997) showed that NE can benefit from parent networks teaching their offspring through backpropagation.

Baxter (1992) performed early work on evolving networks with synaptic plasticity driven by local learning rules, setting the stage for NE of plastic ANNs. He evolved a very simple network that could learn boolean functions of one value. Each connection had a rule for changing its weight to one of two possible values. Baxter's contribution was mainly to show that local learning rules are sufficient to evolve a plastic network. Floreano and Urzelai (2000) later showed that the evolution of local (node-based) synaptic plasticity parameters produces networks that can solve complex problems better than recurrent networks with fixed-weights.

In Floreano and Urzelai's experiment, a plastic network and a fixed-weight fully-recurrent network were evolved to turn on a light by moving to a switch. After the light turned on, the networks had to move onto a gray square. The plastic networks were compared to the fixed-weight networks. Each connection in the plastic network included a learning rule and a learning rate. The fixed-weight network only encoded static connection weights. The sequence of two actions proved difficult to learn for the fixed-weight network because the network could

not adapt to the sudden change in goals after the light was switched on. Fixed-weight networks tended to circle around the environment, slightly attracted by both the light switch and the gray square. Plastic networks, on the other hand, completely changed their trajectories after turning on the light, reconfiguring their internal weights to tackle the problem of finding the gray square. This landmark result established the promise of evolving plastic ANNs and that in fact plastic networks can sometimes evolve faster than static networks. The local learning rules in the evolved networks facilitated the policy transition from one task to the other.

Plastic ANNs have also been successfully evolved to simulate robots in a dangerous foraging domain (Stanley et al., 2003). Although this work also showed that recurrent fixed-weight networks can be more effective and reliable than plastic Hebbian controllers in some domains, more recent studies (Niv et al., 2002; Soltoggio et al., 2008; Soltoggio, Dürr, Mattiussi, & Floreano, 2007) suggest that both network types reach their limits when more elaborate forms of learning are needed. For example, classical conditioning seems to require mechanisms that are not present in most current network models. To expand to such domains, following Soltoggio et al. (2008), the study presented in this paper controls plasticity through *neuromodulation*.

2.2.1 Neuromodulated Plasticity

In the plastic ANNs presented in the previous section (e.g. Floreano & Urzelai, 2000; Stanley et al., 2003), the internal synaptic connection strengths change following a Hebbian learning rule that modifies synaptic weights based on pre- and postsynaptic neuron activity. The generalized Hebbian plasticity rule (Niv et al., 2002) takes the following form:

$$\Delta w = \eta \cdot [Axy + Bx + Cy + D], \tag{2}$$

where η is the learning rate, x and y are the activation levels of the presynaptic and postsynaptic neurons, and A–D are the correlation term, presynaptic term, postsynaptic term, and constant, respectively.

In a *neuromodulated* network, a special neuromodulatory neuron can change the degree of potential plasticity between two standard neurons based on their activation levels (Figure 1). In addition to its standard activation value a_i, each neuron i also computes its modulatory activation m_i:

$$a_i = \sum_{j \in Std} w_{ji} \cdot o_j, \tag{3}$$

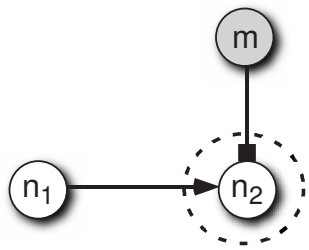


Figure 1: **Neuromodulated plasticity.** The weight of the connection between standard neurons n_1 and n_2 is modified by a Hebbian rule. Modulatory neuron m determines the magnitude of the weight change.

$$m_i = \sum_{j \in Mod} w_{ji} \cdot o_j, \tag{4}$$

where w_{ji} is the connection strength between presynaptic neuron j and postsynaptic neuron i and o_j is calculated as $o_j(a_j) = tanh(a_j/2)$. The weight between neurons j and i then changes following the m_i-modulated plasticity rule

$$\Delta w_{ji} = tanh(m_i/2) \cdot \eta \cdot [Ao_j o_i + Bo_j + Co_i + D]. \tag{5}$$

The benefit of adding modulation is that it allows the ANN to change the level of plasticity on specific neurons at specific times. That is, it becomes possible to decide when learning should *stop* and when it should *start*. This property seems to play a critical role in regulating learning behavior in animals (Carew, Walters, & Kandel, 1981) and neuromodulated networks have a clear advantage in more complex dynamic, reward-based scenarios: Soltoggio et al. (2008) showed that networks with neuromodulated plasticity significantly outperform both fixed-weight and traditional plastic ANNs without neuromodulation in the double T-Maze domain, and display nearly optimal learning performance.

The next section describes NEAT, the method that evolves plastic neuromodulated ANNs in this paper.

2.3 NeuroEvolution of Augmenting Topologies (NEAT)

The NEAT method was originally developed to evolve ANNs to solve difficult control and sequential decision tasks and has proven successful in a wide diversity of domains (Aaltonen et al., 2009; Stanley et al., 2003, 2005; Stanley & Miikkulainen, 2002; Taylor, Whiteson, & Stone, 2006; Whiteson & Stone, 2006). Evolved ANNs control agents that select actions based on their sensory inputs. NEAT is unlike many previous methods that evolved neural networks, i.e. neu-

roevolution methods, which traditionally evolve either fixed-topology networks (Gomez & Miikkulainen, 1999; Saravanan & Fogel, 1995), or arbitrary random-topology networks (Angeline, Saunders, & Pollack, 1994; Gruau, Whitley, & Pyeatt, 1996; Yao, 1999). Instead, NEAT begins evolution with a population of small, simple networks and complexifies the network topology into diverse species over generations, leading to increasingly sophisticated behavior. A similar process of gradually adding new genes has been confirmed in natural evolution (Martin, 1999; Watson, Hopkins, Roberts, Steitz, & Weiner, 1987) and shown to improve adaptation in a few prior evolutionary (Watson et al., 1987) and neuroevolutionary (Harvey, 1993) approaches. However, a key feature that distinguishes NEAT from prior work in complexification is its unique approach to maintaining a healthy diversity of complexifying structures simultaneously, as this section reviews. Complete descriptions of the NEAT method, including experiments confirming the contributions of its components, are available in Stanley and Miikkulainen (2002, 2004) and Stanley et al. (2005).

Before describing the neuromodulatory extension, let us review the three key ideas on which the basic NEAT method is based. First, to allow network structures to increase in complexity over generations, a method is needed to keep track of which gene is which. Otherwise, it is not clear in later generations which individual is compatible with which in a population of diverse structures, or how their genes should be combined to produce offspring. NEAT solves this problem by assigning a unique historical marking to every new piece of network structure that appears through a structural mutation. The historical marking is a number assigned to each gene corresponding to its order of appearance over the course of evolution. The numbers are inherited during crossover unchanged, and allow NEAT to perform crossover among diverse topologies without the need for expensive topological analysis.

Second, historical markings make it possible for the system to divide the population into species based on how similar they are topologically. That way, individuals compete primarily within their own niches instead of with the population at large. Because adding new structure is often initially disadvantageous, this separation means that unique topological innovations are protected and therefore have time to optimize their structure before competing with other niches in the population. The distance δ between two network encodings can be measured as a linear combination of the number of excess (E) and disjoint (D) genes, as well as the average weight differences of matching genes ($\overline{W}$), where *excess* genes are those that arise in the lineage of one parent at a time later than all the genes in the other parent and *disjoint* genes are any

other genes in the lineage of one parent but not the other one (Stanley & Miikkulainen, 2002, 2004):

$$\delta = \frac{c_1 E}{N} + \frac{c_2 D}{N} + c_3 \cdot \overline{W}. \tag{6}$$

The coefficients c_1, c_2, and c_3 adjust the importance of the three factors, and the factor N, the number of genes in the larger genome, normalizes for genome size (N is normally set to one unless both genomes are excessively large; accordingly, it is set to one in this paper). Genomes are tested one at a time; if a genome's distance to a representative member of the species is less than δ_t, a compatibility threshold, the genome is placed into this species.

Third, many systems that evolve network topologies and weights begin evolution with a population of random topologies (Gruau et al., 1996; Yao, 1999). In contrast, NEAT begins with a uniform population of simple networks with no hidden nodes, differing only in their initial random weights. Because of speciation, novel topologies gradually accumulate over evolution, thereby allowing diverse and complex phenotype patterns to be represented. No limit is placed on the size to which topologies can grow. New structures are introduced incrementally as structural mutations occur, and only those structures survive that are found to be useful through fitness evaluations. In effect, then, NEAT searches for a compact, appropriate topology by incrementally increasing the complexity of existing structure.

Few modifications to the standard NEAT algorithm are required to also encode neuromodulated plasticity. NEAT's genetic encoding is augmented with a new modulatory neuron type and each time a node is added through structural mutation, it is randomly assigned a standard or modulatory role. The neuromodulatory dynamics follow equations 3–5.

Also, importantly for this paper, novelty search is designed to work in combination with NEAT (Lehman & Stanley, 2008, 2010c). In particular, once objective-based fitness is replaced with novelty, the NEAT algorithm operates as normal, selecting the highest scoring individuals to reproduce. Over generations, the population spreads out across the space of possible behaviors, continually ascending to new levels of complexity (i.e. by expanding the neural networks in NEAT) to create novel behaviors as the simpler variants are exhausted. Thus, through NEAT, novelty search in effect searches not just for new behaviors, but for *increasingly complex* behaviors.

Therefore, the main idea is to evolve neuromodulatory ANNs with NEAT through novelty search. The hypothesis is that this combination should help to escape the deception inherent

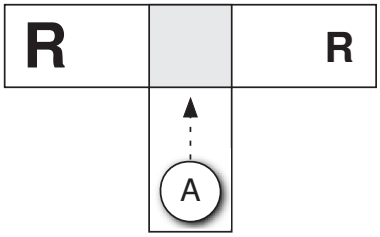


Figure 2: **The T-Maze.** In this depiction, high reward is located on the left and low reward is on the right side, but these positions can change over a set of trials. The goal of the agent is to navigate to the position of the high reward and back home to its starting position. The challenge is that the agent must remember the location of the high reward from one trial to the next.

in many adaptive domains. The next section describes such a domain, which is the initial basis for testing this hypothesis in this paper.

3 The T-Maze domain

The first domain in this paper is based on experiments performed by Soltoggio et al. (2008) on the evolution of neuromodulated networks for the T-Maze learning problem. This domain is ideal to test the hypothesis that novelty search escapes deception in adaptive domains because it is well-established from prior work (Blynel & Floreano, 2003; Dürr, Mattiussi, Soltoggio, & Floreano, 2008; Soltoggio et al., 2008, 2007) and can be adjusted to be more or less deceptive, as is done in this paper. Furthermore, it represents a typical reward-based dynamic scenario (i.e. the agent's actions that maximize reward intake can change during its lifetime), where optimal performance can only be obtained by an adaptive agent. Thus the results presented here should also provide more insight into the potential deceptiveness in similar learning problems.

The single T-Maze (Figure 2) consists of two arms that either contain a high or low reward. The agent begins at the bottom of the maze and its goal is to navigate to the reward position and return home. This procedure is repeated many times during the agent's lifetime. One such attempted trip to a reward location and back is called a *trial*. A *deployment* consists of a set of trials (e.g. 20 trials in the single T-Maze experiments in this paper are attempted over the course of a deployment). The goal of the agent is to maximize the amount of reward collected over deployments, which requires it to memorize the position of the high reward in each deployment. When the position of the reward sometimes changes, the agent should alter its strategy accordingly to explore the other arm of the maze in the next trial. In Soltoggio's original experiments (Soltoggio et al., 2008), the reward location changes at least once during

each deployment of the agent, which fosters the emergence of learning behavior.

However, the *deceptiveness* of this domain with respect to the evolution of learning can be increased if the reward location is not changed in all deployments in which the agent is evaluated. For example, an individual that performs well in the 99 out of 100 deployments wherein learning is not required and only fails in the one deployment that requires learning will most likely score a high fitness value. Thus such a search space is highly deceptive to evolving learning and the stepping stones that ultimately lead to an adaptive agent will not be rewarded. The problem is that learning domains often have the property that significant improvement in fitness is possible by discovering hidden heuristics that avoid lifetime adaptation entirely, creating a pathological deception against learning to learn.

If adaptation is thus only required in a small subset of deployments, the advantage of an adaptive individual over a non-adaptive individual (i.e. always navigating to the same side) in fitness is only marginal. The hypothesis is that novelty search should outperform fitness-based search with increased domain deception.

4 Single T-Maze Experiment

To compare the performance of NEAT with fitness-based search and NEAT with novelty search, each agent is evaluated on ten deployments, each consisting of 20 trials. The number of deployments in which the high reward is moved after ten trials varies among one (called the *1/10 scenario*), five (called the *5/10 scenario*), and ten (called the *10/10 scenario*), effectively controlling the level of deception. The high reward always begins on the left side at the start of each deployment.

Note that all deployments are deterministic, that is, a deployment in which the reward does not switch sides will always lead to the same outcome with the same ANN. Thus the number of deployments in which the reward switches is effectively a means to control the proportional influence of adaptive versus non-adaptive deployments on fitness and novelty. The question is whether the consequent deception impacts novelty as it does fitness.

Of course, it is important to note that a population rewarded for performance in the 1/10 scenario would not necessarily be expected to be attracted to a general solution. At the same time, a process like novelty search that continues to find new behaviors should ultimately encounter the most general such behavior. Thus the hypothesized advantage of novelty search

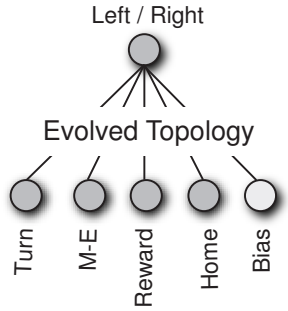


Figure 3: **ANN topology.** The network has four inputs, one bias, and one output neuron (Soltoggio et al., 2008). *Turn* is set to 1.0 at a turning point location. *M-E* is set to 1.0 at the end of the maze, and *Home* is set to 1.0 when the agent returns to the home location. The *Reward* input returns the level of reward collected at the end of the maze. The bias neuron emits a constant 1.0 activation that can connect to other neurons in the ANN. Network topology is evolved by NEAT.

in such scenarios follows naturally from the dynamics of these different types of search.

Figure 3 shows the inputs and outputs of the ANN (following Soltoggio et al., 2008). The *Turn* input is set to 1.0 when a turning point is encountered. *M-E* is set to 1.0 at the end of the maze and *Home* becomes 1.0 when the agent successfully navigates back to its starting position. The *Reward* input is set to the amount of reward collected at the maze end. An agent crashes if it does not (1) maintain a forward direction (i.e. activation of output neuron between -0.3 and 0.3) in corridors, (2) turn either right ($o > 0.3$) or left ($o < -0.3$) when it encounters the junction, or (3) make it back home after collecting the reward. If the agent crashes then the current trial is terminated.

The fitness function for fitness-based NEAT (which is identical to Soltoggio et al., 2008) is calculated as follows: Collecting the high reward has a value of 1.0 and the low reward is worth 0.2. If the agent fails to return home by taking a wrong turn after collecting a reward then a penalty of 0.3 is subtracted from fitness. On the other hand, 0.4 is subtracted if the agent does not maintain forward motion in corridors or does not turn left or right at a junction. The total fitness of an individual is determined by summing the fitness values for each of the 20 trials over all ten deployments.

Novelty search on the other hand requires a *novelty metric* to distinguish between different behaviors. The novelty metric for this domain distinguishes between learning and non-learning individuals and is explained in more detail in the next section.

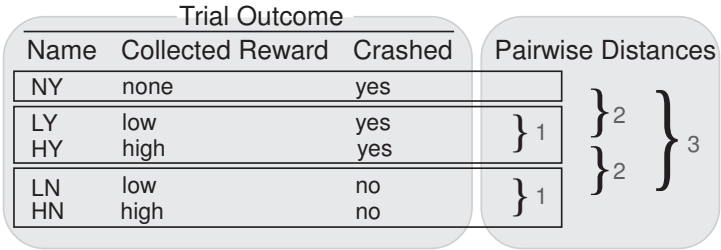


Figure 4: **The T-Maze novelty metric.** Each trial is characterized by (1) the amount of collected reward (2) whether the agent crashed. The pairwise distances (shown at right) among the five possible trial outcomes, NY, LY, HY, LN, and HN, depend on their behavioral similarities.

4.1 Measuring Novelty in the Single T-Maze

The aim of the novelty metric is to measure differences in behavior. In effect, it determines the behavior-space through which the search explores. Because the goal of this paper is to evolve adaptive individuals, the novelty metric must distinguish a learning agent from a non-learning agent. Thus it is necessary to characterize behavior so that different such behaviors can be compared. The behavior of an agent in the T-Maze domain is characterized by a series of trial outcomes (i.e. 200 trial outcomes for ten deployments with 20 trials each). To observe learning behavior, and to distinguish it from non-learning behavior, it is necessary to run multiple trials in a single lifetime, such that the agent's behavior before and after a reward switch can be observed. Importantly, the behavior space in the T-Maze domain is therefore significantly larger than in prior experiments (Lehman & Stanley, 2008), effectively testing novelty search's ability to succeed in a high-dimensional behavior space of 200 dimensions (versus only two dimensions in Lehman & Stanley, 2008). It is important to note that the dimensionality of the behavior space is not the only possible characterization of the dimensionality of the problem. For example, the dimensionality of the solution ANN is also significantly related to the difficulty of the problem.

Each trial outcome is characterized by two values: (1) the amount of reward collected (*high*, *low*, *none*) and (2) whether or not the agent crashed. These outcomes are assigned different *distances* to each other depending on how similar they are (Figure 4). In particular, an agent that collects the high reward and returns home successfully without crashing (HN) should be more similar to an agent that collects the low reward and also returns home (LN) than to one that crashes without reaching any reward location (NY). The novelty distance metric $dist_{novelty}$ is ultimately computed by summing the distances between each trial outcome of two individuals

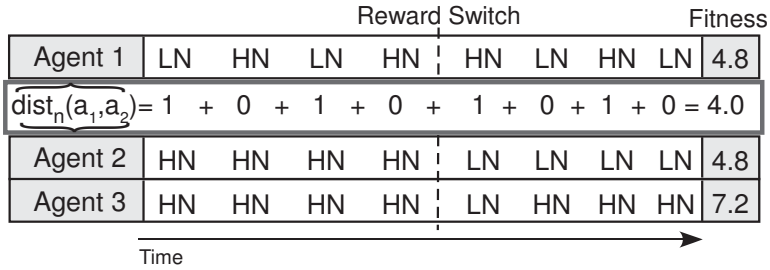


Figure 5: **Three sample behaviors.** These learning and non-learning individuals all exhibit distinguishable behaviors when compared over multiple trials. Agent three achieves the desired adaptive behavior. The vertical line indicates the point in time that the position of the high reward changed. While agents 1 and 2 look the same to fitness, novelty search notices their difference, as the distance calculation (inset line between agents 1 and 2) shows.

over all deployments.

Figure 5 depicts outcomes over several trials of three example agents. The first agent always alternates between the left and the right T-Maze arm, which leads to oscillating low and high rewards. The second agent always navigates to the left T-Maze arm. This strategy results in collecting the high reward in the first four trials and then collecting the low reward after the reward switch. The third agent exhibits the desired learning behavior and is able to collect the high reward in seven out of eight trials. (One trial of exploration is needed after the reward switch.)

Interestingly, because both agents one and two collect the same amount of high and low reward, they achieve the same fitness, making them indistinguishable to fitness-based search. However, novelty search discriminates between them because $dist_{novelty}(agent_1, agent_2) = 4.0$ (Figure 5). Recall that this behavioral distance is part of the novelty metric (Equation 1), which replaces the fitness function and estimates the sparseness at a specific point in behavior space.

Importantly, fitness and novelty both use the same information (i.e. the amount of reward collected and whether or not the agent crashed) to explore the search space, though in a completely different way. Thus the comparison is fair.

4.2 Generalization Performance

An important goal of the comparison between fitness and novelty is to determine which learns to adapt most efficiently in different deployment scenarios, e.g. 1/10, 5/10, and 10/10. Thus it is important to note that, because performance on different scenarios will vary based on the number of trials in which the reward location switches, for the purpose of analyzing the results there is a need for an independent measure that reveals the overall adaptive capabilities of each

individual.

Therefore, to test the ability of the individuals to generalize independently of the number of deployments in which the position of the high reward changes, they are tested for 20 trials on each of two different initial settings: (1) high reward starting left and (2) high reward starting right. In both cases, the position of the high reward changes after 10 trials. An individual passes the *generalization test* if it can collect the high reward and return back home in at least 18 out of 20 trials from both initial positions. Two low reward trials in each setting are necessary to explore the T-Maze at the beginning of each deployment and when the position of the high reward switches.

The generalization measure does not necessarily correlate to fitness. An individual that receives a high fitness in the 1/10 scenario can potentially perform poorly on the generalization test because it does not exhibit adaptive behavior. Nevertheless, generalization performance does follow a general upward trend over evaluations and reveals the ultimate quality of solutions (i.e. individuals passing the generalization test would receive high fitness scores in all scenarios).

4.3 Experimental Parameters

NEAT with fitness-based search and novelty search run with the same parameters in the experiments in this paper. The steady-state real-time NEAT (rtNEAT) package (Stanley, 2006-2008) is extended to encode neuromodulatory neurons. The population size is 500, with a 0.001 probability of adding a node (uniformly randomly chosen to be standard or modulatory) and 0.01 probability of adding a link. The weight mutation power is 1.8. The coefficients c_1, c_2 and c_3 for NEAT's genome distance (see Equation 6) are all set to 1.0. Runs last up to 125,000 evaluations. They are stopped when the generalization test is solved. The number of nearest neighbors for the novelty search algorithm is 15 (following Lehman & Stanley, 2008). The novelty threshold is 2.0. This threshold for adding behaviors to the archive dynamically changes every 1,500 evaluations. If no new individuals are added during that time the threshold is lowered by 5%. It is raised by 20% if the number of individuals added is equal to or higher than four. The novelty scores of the current population are reevaluated every 100 evaluations to keep them up to date (the archive does not need to be reevaluated). Connection weights range within [-10, 10]. These parameter values are shared by all experiments in this paper.

The coefficients of the generalized Hebbian learning rule used by all evolved neuromodulated networks in the T-Maze domain are $A = 0.0$, $B = 0.0$, $C = -0.38$, $D = 0.0$ and $\eta = -94.6$,

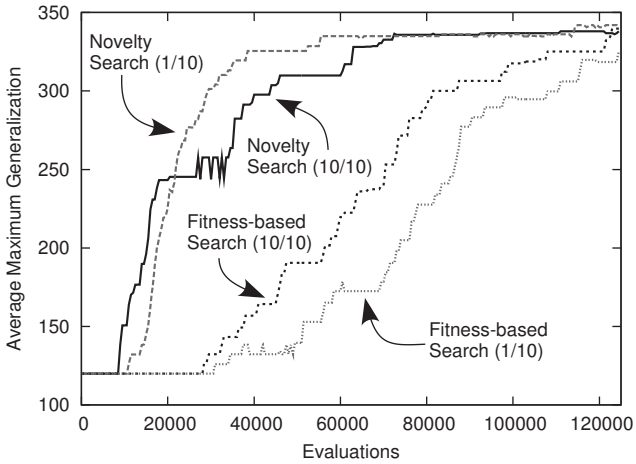


Figure 6: **Comparing generalization of novelty search and fitness-based search.** The change in performance (calculated like fitness) over evaluations on the generalization test is shown for NEAT with novelty search and fitness-based search in the 1/10 and 10/10 scenarios. All results are averaged over 20 runs. The main result is that novelty search learns a general solution significantly faster.

resulting in the following m_i-modulated plasticity rule:

$$\Delta w_{ji} = tanh(m_i/2) \cdot 35.95y. \tag{7}$$

These values worked well for a neuromodulated ANN in the T-Maze learning problem described by Soltoggio et al. (2008). Therefore, to isolate the effect of evolving based on novelty versus fitness, they are fixed at these values in the T-Maze experiments in this paper. However, modulatory neurons still affect the learning rate at Hebbian synapses as usual. For a more detailed description of the implications of different coefficient values for the generalized Hebbian plasticity rule, see Niv et al. (2002).

5 Single T-Maze Results

Because the aim of the experiment is to determine how quickly a general solution is found by fitness-based search and novelty search, an agent that can solve the generalization test described in Section 4.2 counts as a solution.

Figure 6 shows the average performance (over 20 runs) of the current best-performing individuals on the generalization test across evaluations for novelty search and fitness-based search, depending on the number of deployments in which the reward location changes. Novelty search performs consistently better in all scenarios. Even in the 10/10 domain that resembles the orig-

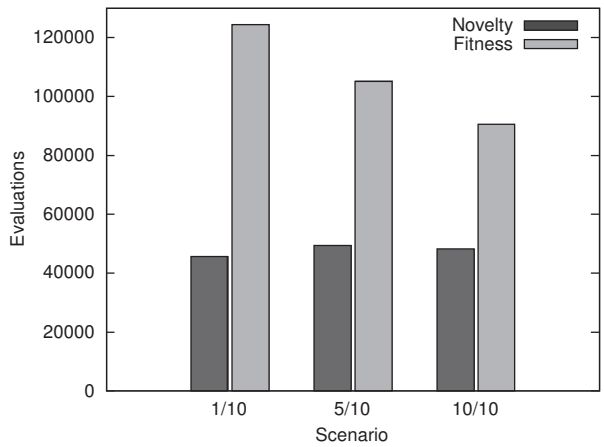


Figure 7: **Average evaluations to solution for novelty search and fitness-based search.** The average number of evaluations over 20 runs that it took novelty search and fitness-based search to solve the generalization test is shown. Novelty search performs significantly better in all scenarios and fitness-based search performs even worse when deception is high. Interestingly, novelty search performance does not degrade at all with increasing deception.

inal experiment (Soltoggio et al., 2008), it takes fitness significantly longer to reach a solution than novelty search. The fitness-based approach initially stalls, followed by gradual improvement, whereas on average novelty search rises sharply from early in the run.

Figure 7 shows the average number of evaluations (over 20 runs) that it took fitness-based and novelty-based NEAT to solve the generalization test in the 1/10, 5/10, and 10/10 scenarios. If no solution was found within the initial 125,000 evaluations, the current simulation was restarted (i.e. a new run was initiated). This procedure was repeated until a solution was found, counting all evaluations over all restarts.

Both novelty and fitness-based NEAT were restarted three times out of 20 runs in the 10/10 scenario. Fitness-based search took on average 90,575 evaluations ($\sigma = 52,760$) while novelty search was almost twice as fast at 48,235 evaluations on average ($\sigma = 55,638$). This difference is significant ($p < 0.05$). In the more deceptive 1/10 scenario, fitness-based search had to be restarted six times and it took 124,495 evaluations on average ($\sigma = 81,789$) to find a solution. Novelty search only had to be restarted three times and was 2.7 times faster ($p < 0.001$) at 45,631 evaluations on average ($\sigma = 46,687$).

Fitness-based NEAT performs worse with increased domain deception and is 1.4 times slower in the 1/10 scenario than in the 10/10 scenario. It took fitness on average 105,218 evaluations ($\sigma = 65,711$) in the intermediate 5/10 scenario, which is in-between its performance on the 1/10 and 10/10 scenarios, confirming that deception increases as the number of trials requiring adaptation decreases. In contrast, novelty search is not significantly affected by increased

domain deception: The performance differences among the 1/10, 5/10, and 10/10 scenarios is insignificant for novelty search, confirming its immunity to deception.

Recall that individuals cannot avoid spending at least two trials (when the position of the reward switches for both initial settings; Section 4.2) collecting the low reward to pass the generalization test. However, the action *after* collecting the low reward (e.g. crashing into a wall or taking a wrong turn on the way home) is not part of the criteria for passing the test. To ensure that stricter criteria do not change the results, a second experiment consisting of 50 independent runs was performed in which the agent also had to return back home after collecting the *low reward*. That way, the agent always must return home. It took novelty search on average 191,771 evaluations ($\sigma = 162,601$) to solve this harder variation while fitness-based search took 267,830 evaluations ($\sigma = 198,455$). In this harder scenario, both methods needed more evaluations to find a solution but the performance difference is still significant ($p < 0.05$).

Interestingly, novelty search not only outperforms fitness-based search in the highly deceptive 1/10 scenario but also in the intermediate 5/10 scenario and even in the 10/10 scenario, in which the location of the reward changes every deployment. There is no obvious deception in the 10/10 scenario (which resembles Soltoggio's original experiment; Soltoggio et al., 2008). However, the recurring plateaus in fitness common to all scenarios (Figure 6) suggest a general problem for evolving learning behavior inherent to dynamic, reward-based scenarios. The next section addresses this issue in more depth.

6 Analysis of Deception in the Single T-Maze

Why does novelty search outperform fitness-based search even when the position of the high reward changes in every deployment? To answer this question the analysis in this section is based on runs in a seemingly non-deceptive setting: Every individual is evaluated in one deployment consisting of 20 trials in which the position of the high reward switches after ten trials. The high reward is always located on the left side of the maze at the beginning of each deployment and an individual counts as a solution if it can collect at least 18 out of 20 high rewards and navigate back home. This reduction in deployments combined with the relaxed solution criteria simplifies an in-depth analysis (e.g. by yielding smaller novelty archives that are easy to visualize) and is analogous to the 10/10 scenario. The aim is to uncover the hidden source of deception and how exactly novelty search avoids it.

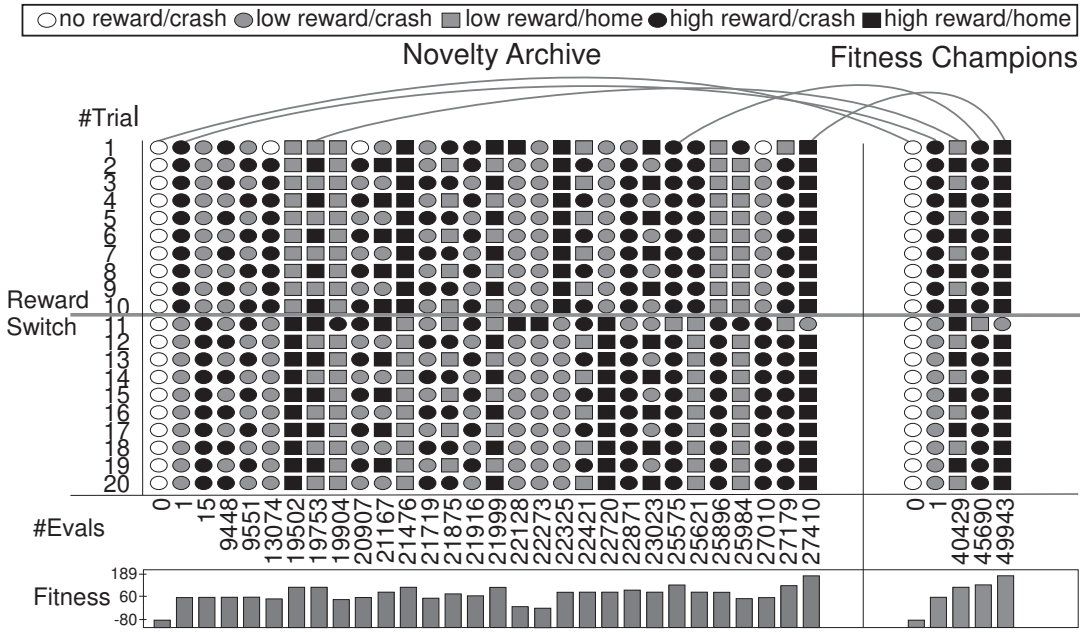


Figure 8: **Novelty search archive and fitness champions.** Behaviors archived by novelty search and the highest-fitness-so-far found by fitness-based search during evolution are shown together with their corresponding fitness and evaluation at which they were discovered. Agents are evaluated on 20 trials and the reward location switches after 10 trials. Arcs (at top) connect behaviors that were highly rewarded by both methods. Novelty search consistently archives new behaviors while fitness-based search improves maximum fitness only four times during the whole evolutionary run. Many of the behaviors found by novelty search would receive the same fitness, which means they are indistinguishable to fitness-based search. The main result is that a higher number of promising directions are explored by novelty search.

One interesting way to analyze different evolutionary search techniques is to examine what they consider *best*, i.e. which individuals have the highest chances to reproduce. For novelty search, these are the individuals that display the most novel behavior and therefore enter the archive. For fitness-based search, the most rewarded are the *champions*, i.e. the behaviors with the highest fitness found so far. Although the probabilistic nature of the evolutionary search means that such individuals are not guaranteed to produce offspring, they represent the most likely to reproduce.

Highlighting the dramatic difference between these contrasting reward systems, Figure 8 shows the behaviors archived by novelty search and the most fit individuals (when they first appear) found by fitness-based search during a typical evolutionary run. It took novelty search 27,410 evaluations to find a solution in this scenario while fitness-based search took almost twice as long with 49,943 evaluations. While novelty search finds 30 behaviors that are novel enough to enter the archive, fitness only discovers five new champions during the whole evolutionary run. A look at the fitness values of the archived novel behaviors reveals that many of them collapse

to the same score, making them indistinguishable to fitness-based search (also see Section 4.1 for discussion of such conflation). For example, the second through fifth archived behaviors in Figure 8, which represent different combinations of ten *HY* (high reward/crash) and ten *LY* (low reward/crash) events, all receive the same fitness. However, they are all highly rewarded by novelty search at the time they are discovered, which places them into the archive.

In the first 40,429 evaluations, fitness-based search does not discover *any* new champions, giving it little information about the direction in which the search should proceed. On the other hand, novelty search constantly produces novel behaviors *and* takes these behaviors and the current population into account to guide the search.

A visualization technique can help to gain a deeper understanding of how the two approaches navigate the high-dimensional genotypic search space. The most common technique to visualize evolution is to plot fitness over evaluations; although this technique reveals information about the quality of the solution found so far, it provides no information on how the search proceeds through the high-dimensional search space. Various methods have been proposed to illuminate the trajectory of the search (Barlow, Galloway, & Abbass, 2002; Kim & Moon, 2003; Vassilev, Fogarty, & Miller, 2000), most of which focus on visualizing the fitness landscape to gain a deeper understanding of its ruggedness.

However, the aim of this analysis is to visualize how the genotypes produced by both search methods traverse the search space *in relation to each other*. Two potential such visualization techniques are *Principal Component Analysis* (PCA) (Kittler & Young, 1973) and *Sammon's Mapping* (Sammon, 1969). Both methods provide a mapping of high-dimensional points in genotypic space ($\Re^p$) to points in $\Re^2$. However, while PCA tries to account for the most variance in the data at expense to their original Euclidian distances, Sammon's Mapping aims to *preserve the distances* of the genotypes in the mapping to a lower dimension (Dybowski, Collins, Hall, & Weller, 1996). Therefore, Sammon's mapping is chosen for this analysis because the distances between genotypes produced by fitness-based search and novelty search in the two dimensional visualization should be as close to their original distances as possible to understand how they relate. This approach facilitates the comparison between different regions of the search space that both methods explore.

Sammon's mapping maps a high-dimensional dataset onto a lower number of dimensions (typically two or three-dimensions), allowing a better understanding of the underlying structure of data. The mapping minimizes the stress measure E, which is the discrepancy between the

high dimensional distances δ_{ij} between all objects i and j and the resulting distances d_{ij} between the data points in the lower dimension:

$$E = \frac{1}{\sum_{i=1}^{n-1} \sum_{j=i+1}^{n} \delta_{ij}} \sum_{i=1}^{n-1} \sum_{j=i+1}^{n} \frac{(\delta_{ij} - d_{ij})^2}{\delta_{ij}}. \tag{8}$$

The stress measure can be minimized by a steepest descent procedure in which the resulting value of E is a good indicator of the quality of the projection.

For this study, Sammon's mapping projects high-dimensional *genotypes* produced over the course of evolution onto a two-dimensional space. The output of the mapping are x and y coordinates for every genotype that minimize stress measure E. The original high-dimensional distance δ_{ij} between two genotypes is based on NEAT's genome distance (Equation 6), which is a good indicator of the similarity of two network encodings. The distance d_{ij} between two objects i and j in the visualization space is calculated by their Euclidean distance $\sqrt{(i_x - j_x)^2 + (i_y - j_y)^2}$. To make the two-dimensional visualization clearer, not all genotypes created during evolution are part of the Sammon's mapping; instead, only those are shown that have either (1) a genome distance δ greater than 9.0 compared to already recorded genotypes or (2) have a distance smaller than 9.0 but display a different behavior (based on the novelty metric described in Section 4.1). These criteria ensure that a representative selection of genotypes is shown that is still sparse enough to be visible in the projection onto two dimensions.

Figure 9 shows a Sammon's mapping of 882 genotypes; 417 were found by novelty search and 465 were found by fitness-based search during a typical evolutionary run of each. In this example, novelty search found a solution after 19,524 evaluations while it took fitness-based search 36,124 evaluations. The low stress measure $E = 0.058$ indicates that the original genotypic distances have been conserved by the mapping. Genotypes that are close to each other in the two-dimensional output space are also close to each other in genotype space.

The mapping reveals that both methods discover different regions of high fitness and that the majority of behaviors simply crash without collecting any rewards (denoted by the smallest points). The main result is that while novelty search (light gray) discovers a genotypic region of high fitness and then quickly reaches the solution (i.e. a behavior that can collect the high reward in at least 18 out of 20 trials, denoted by D in Figure 9), fitness-based search (black) needs to cover more of the *genotypic* search space because it searches through many identical behaviors (though different genotypes) when it is stuck at a local optimum.

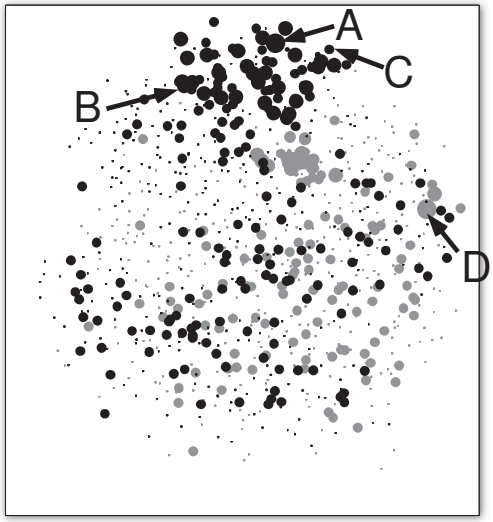


Figure 9: **Combined Sammon's Mapping.** The Sammon's mapping of 417 genotypes found by novelty search (gray) and 465 found by fitness-based search (black) is shown. The size of each mark corresponds to the fitness of the decoded network. Larger size means higher fitness. Fitness-based search covered more of the *genotypic* search space than novelty search because it searches through many identical behaviors (though different genotypes) when it is stuck at a local optimum. Four important individuals are identified: the final solution found by fitness-based search (A), a network that collects the high reward in the first ten trials and then the low reward (B), a network that collects the low reward in 18/20 trials (C) and the final solution found by novelty search (D). Although A and C are close they have significantly different fitnesses. Thus while the discovery of C could potentially serve as a stepping stone for novelty search, fitness-based search is led astray from the final solution. Points B and D are discussed in the text.

Interestingly, an intermediate solution found by fitness-based search discovers a behavior that collects 18 out of 20 *low* rewards and returns back home (denoted by C in Figure 9). The network that produces this behavior and the final solution (A) are close in genotypic space though they have very different fitness values (178 vs. 50). Thus while the discovery of this behavior could potentially serve as a stepping stone to finding the final solution for novelty search, rather than helping fitness it actually *deceives* it. Agents that collect the high reward in 10 out of 20 trials and return back home (B in Figure 9) receive a higher fitness than C-type agents even though they are actually *farther* away from the final solution in genotype space and therefore might lead fitness search astray.

Figure 10 examines the temporal progression of the two search methods in more detail by showing the Sammon's mapping from Figure 9 at different stages of evolution in the corresponding run. For each evaluation (i.e. snapshot in time) the mapping shows the genotypes found so far together with the behaviors archived by novelty search and the champions found by fitness-based search.

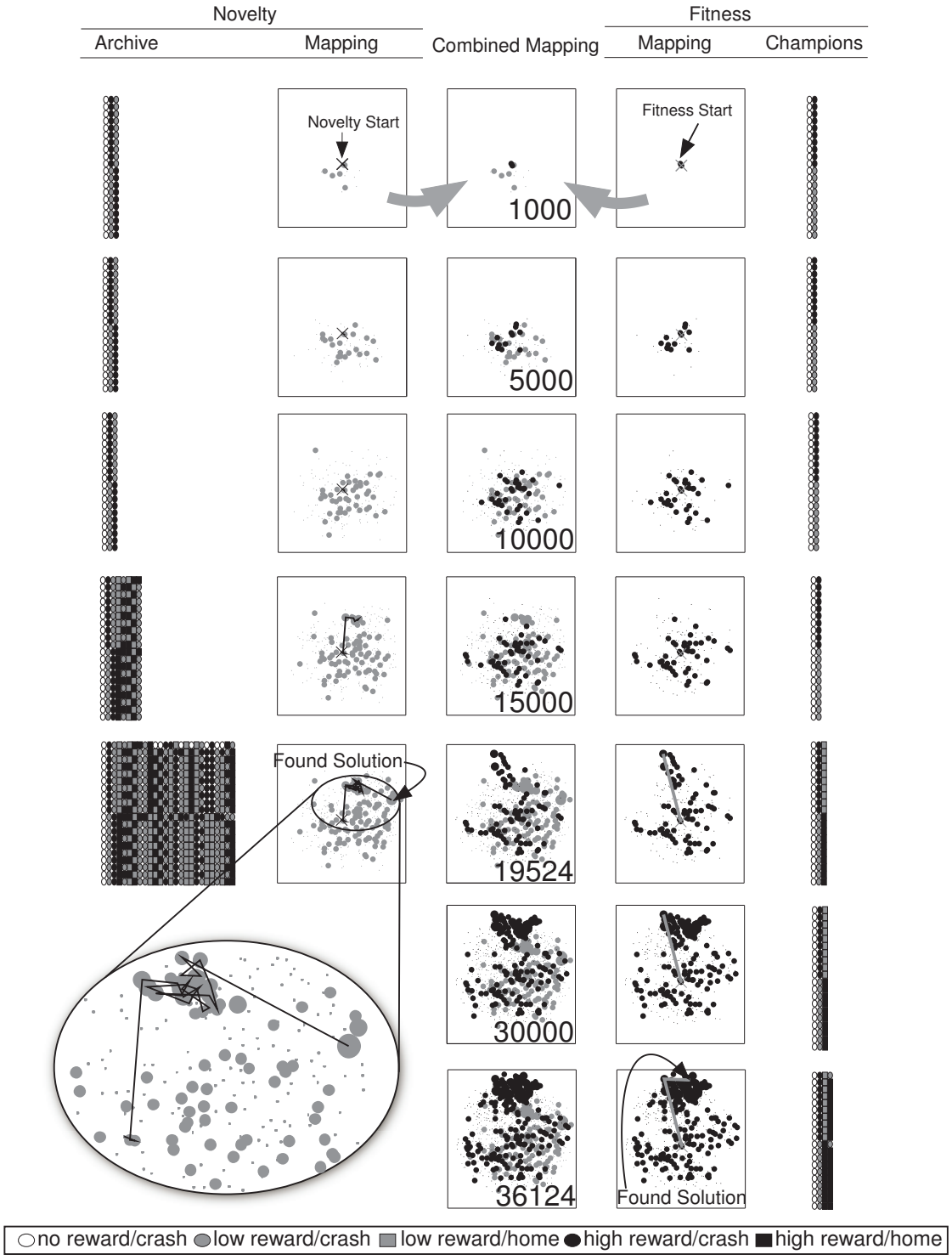


Figure 10: **Sammon's Mapping of novelty and fitness-based search at different stages of evolution.** A mapping of 882 recorded genotypes – 470 produced by novelty search (second column) and 465 by fitness-based search (fourth column) – is shown at seven different time steps together with the corresponding behavior characterizations added to the archive by novelty search and those of the champions found by fitness-based search. Larger markers in the Sammon's mapping denote higher fitness received by the decoded network. The archived behaviors found by novelty search and the champions found by fitness-based search are connected to show the progression of each search. The magnification (bottom left) of the novelty mapping shows a region of the genotypic space with many novel behaviors that have small genotypic distances to each other. Novelty search finds a solution significantly faster than fitness-based search by exploiting intermediate stepping stones to guide the search.

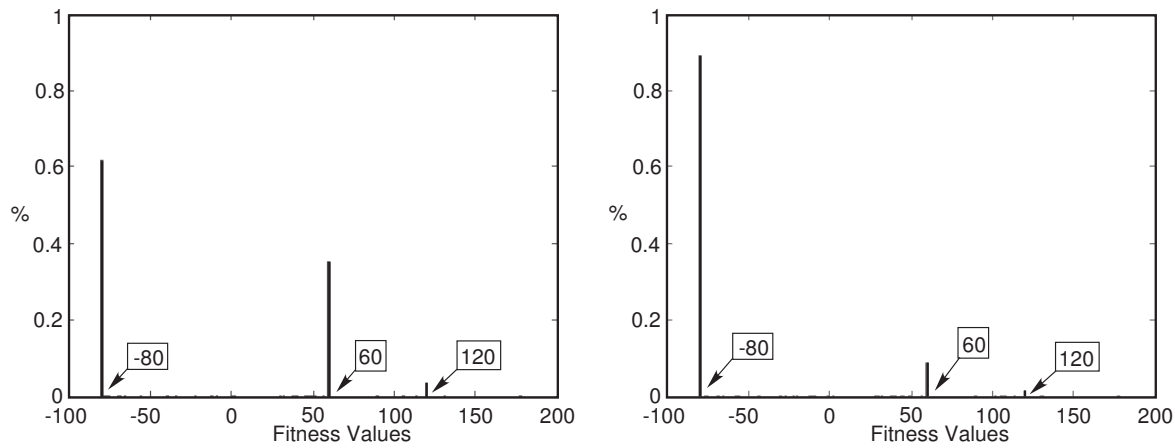
Novelty search explores a wider sampling of the search space than fitness-based search during the first 1,000 evaluations. After that, both methods explore similar behaviors until novelty search finds a novel behavior at evaluation 13,912 that collects the low reward and returns back home in the first ten trials and then collects the high reward and returns back home in the successive trials. The ability to successfully return back home after collecting a reward turns out to be a stepping stone to regions of higher fitness. It opens up a wide range of possible new behaviors that lead novelty search to discover 18 new archive members between evaluations 15,000 and 19,520. Interestingly, all the underlying network encodings for these behaviors are close to each other in genotypic space even though they produce significantly different behaviors. Finally, novelty search discovers a solution after 19,524 evaluations.

In contrast, fitness-based search is not able to exploit the same set of behaviors as potential stepping stones because many collapse to the same fitness. While fitness-based search discovers two new champions in the first 1,000 evaluations, it does not discover the next until evaluation 19,520. This more fit behavior is located within a cluster of high fitness genotypes close to the final solution. However, it takes fitness-based search another 17,439 evaluations to discover that solution. The problem again is that fitness-based search is deceived by genotypes that have a higher fitness than those that are actually closer to the solution (Figure 9).

In a sense, novelty search proceeds more systematically, discovering a region of novel behaviors and then discovering the final solution in fewer evaluations than fitness-based search by exploiting intermediate stepping stones to guide the search. In fact, the number of archived behaviors is always higher than the number of new champions found by fitness across all runs.

To gain a better understanding of the fitness landscape in this domain, Figure 11 shows histograms of fitness values for individuals discovered by novelty and fitness-based search in a typical run. The histograms are normalized so that the area sum is one. Interestingly, the vast majority of behaviors (for novelty and fitness-based search) receive one of three different fitness values resulting in three peaks in each distribution. In effect, many behaviors receive the same fitness, which is another indicator of the lack of intermediate stepping stones and the absence of a fitness gradient in the T-Maze domain. Moreover, the majority of behaviors (61% for fitness and 88% for novelty) simply crash without collecting any reward, suggesting that the encoded networks are brittle to small mutations.

Overall, the analysis in this section shows that novelty search is able to return more information about how behavior changes throughout the search space. It finds a solution significantly



(a) Fitness-based search

(b) Novelty search

Figure 11: **Distribution of fitness for novelty and fitness-based search for a typical run.** The fitness values for fitness-based search (a) and novelty search (b) both have three peaks in both histograms. There are other values, but they are relatively very rare. Thus many of the genotypes collapse to the same few fitness values, suggesting a lack of intermediate stepping stones for a fitness-based search method.

faster than fitness-based search by exploiting intermediate stepping stones to guide its search. Interestingly, genotypes that are potential stepping stones for novelty search can lead fitness-based search astray if fitness does not correlate with distance to the final solution (Figure 9).

7 Additional Experiments

To further demonstrate novelty search's ability to efficiently evolve plastic ANNs, two substantially more complex scenarios are investigated, which are explained in the next sections.

7.1 Double T-Maze

The double T-Maze (Figure 12) includes two turning points and four maze endings, which makes the learning task substantially more difficult than the single T-Maze studied in the previous sections (Soltoggio et al., 2008). In effect the agent must now memorize a location on a map that is twice as large.

The experiment follows the setup described in Section 3 with a slightly modified novelty metric to capture behaviors in the larger environment. The behavior of an agent is still characterized by a series of trial outcomes, but each such outcome is now determined by the corresponding trial fitness value (e.g. 0.2 for collecting the low reward). The behavioral difference between two behaviors is then calculated as the sum over all trial differences. Each evaluation consists

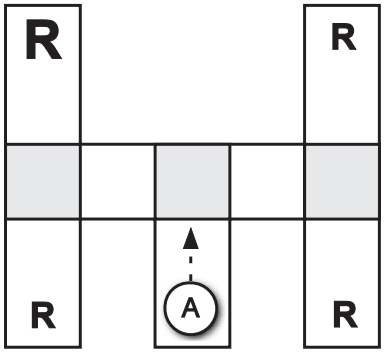


Figure 12: **The Double T-Maze.** The maze includes two turning points and four maze endings. In this depiction, high reward is located at the top-left of the maze, but its position can change over a set of trials. The goal of the agent is to navigate to the position of the high reward and remember that location from one trial to the next.

of two deployments with 200 trials each in which the high reward changes location after every 50 trials. Thus the behavior characterization includes 400 dimensions.

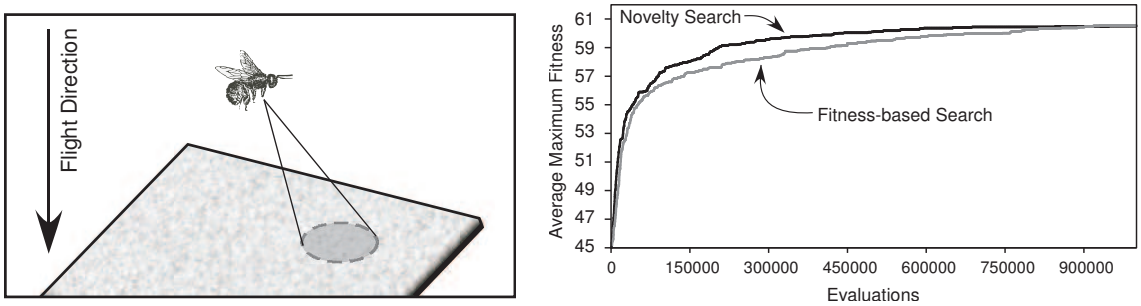
Fitness-based search had to be restarted five times and found a solution in 801,798 evaluations on average (σ=695,534). Novelty search found a solution in 364,821 evaluations on average (σ=411,032) and had to be restarted two times. Therefore, even with an increased behavioral characterization (200-dimensional for the single T-Maze vs. 400-dimensional for the double T-Maze) and increased domain complexity, novelty search still finds the appropriate adaptive behavior significantly faster than fitness-based search ($p < 0.05$).

7.2 Foraging Bee

Another domain studied for its reinforcement learning-like characteristics is the bee foraging task (Soltoggio et al., 2007; Niv et al., 2002). A simulated bee needs to remember which type of flower yields the most nectar in a stochastic environment wherein the amount of nectar associated with each type of flower can change.

The bee flies in a simulated three-dimensional environment (Figure 13a) that contains a 60×60 meter flower field on the ground with two different kind of flowers (e.g. yellow and blue). The bee constantly flies downward at a speed of 0.5m/s and can perceive the flower patch through a single eye with a 10-degree view cone. The outside of the flower patch is perceived as gray-colored.

The ANN controller has five inputs (following Soltoggio et al., 2007); the first three are *Gray*, *Blue*, and *Yellow*, which receive the percentage of each perceived color in the bee's view cone. The *Reward* input is set to the amount of nectar consumed after the bee successfully



(a) Simulated Bee (b) Average Maximum Fitness

Figure 13: **Comparing Novelty Search to Fitness-based Search in the Bee Domain.** The simulated bee flying in a three-dimensional space is shown in (a). The bee is constantly flying downwards but can randomly change its direction. The bee can perceive the flower patch with a simulated view cone (Soltoggio et al., 2007). (b) The change in fitness over time (i.e. number of evaluations) is shown for NEAT with novelty search and fitness-based NEAT, which are both averaged over 25 runs for each approach. The main result is that both methods reach about the same average fitness but novelty search finds a solution significantly faster.

lands on either a yellow or blue flower. It remains zero during the flight. The *Landing* input is set to 1.0 upon landing. The bee has only one output that determines if it stays on its current course or changes its direction to a new random heading.

Each bee is evaluated on four deployments with 20 trials each. The flower colors are inverted after 10 trials on average and between deployments, thereby changing the association between color and reward value. The amount of collected nectar (0.8 for high-rewarding and 0.3 for low-rewarding flowers) over all trials is the performance measure for fitness-based search.

To mitigate noisy evaluation in novelty search (i.e. a single random change in direction at the beginning of the flight can lead to a very different sequence of collected high and low rewards), each deployment is described by two values. The first is the number of high rewards collected before the color switch and the second is the number of high rewards collected afterwards. This novelty metric rewards the bee for collecting a different number of high rewards before and after the reward switch than other individuals. This measure reflects that what is important is displaying adaptive behavior that is dependent on the time of the reward switch. The coefficients of the generalized Hebbian learning rule for this experiment are $A = -0.79$, $B = 0.0$, $C = 0.0$, and $D = -0.038$. These values worked well for a neuromodulated ANN in the foraging bee domain described by Soltoggio et al. (2007). The other experimental parameters are kept unchanged.

A bee counts as a solution if it displays the desired learning behavior of associating the right color with the currently high-rewarding flower (which corresponds to a fitness of 61).

Both fitness-based search and novelty search discovered solutions in 13 out of 25 runs. Novelty search took on average 261,098 evaluations (σ=130,926) when successful and fitness-based search on average 491,221 evaluations (σ=277,497). Although novelty search still finds a solution significantly faster ($p < 0.05$), both methods quickly reach a high local optimum before that (Figure 13b).

8 Discussion and Future Work

Novelty search outperforms fitness-based search in all domains investigated and is not affected by increased domain deception. This result is interesting because it is surprising that without any other a priori knowledge an algorithm that is not even aware of the desired behavior would find such behavior at all, let alone in a general sense.

Fitness-based search also takes significantly more evaluations to produce individuals that exhibit the desired adaptive behavior when the impact of learning on the fitness of the agent is only marginal. Because it is easier at first to improve fitness without evolving the ability to learn, objective-based search methods are likely to exploit domain-dependent static (i.e. non-adaptive) heuristics.

In the T-Maze domain in this paper, agents initially learn to always navigate to one arm of the maze and back, resulting in collecting 20 high rewards (i.e. ten high rewards for each of the two starting positions) on the generalization test. Yet, because the reward location changes after ten trials for both initial settings, to be more successful requires the agents to exhibit learning behavior.

The natural question then is why novelty search outperforms fitness-based search in the seemingly non-deceptive 10/10 scenario? While the deception in this setting is not as obvious, the analysis presented in Section 6 addressed this issue in more depth. The problem is that evolving the right neuromodulated dynamics to be able to achieve learning behavior is not an easy task. There is little information available to incentivize fitness-based search to pass beyond static behavior, making it act more like random search. In other words, the stepping stones that lead to learning behavior are hidden from the objective approach behind long plateaus in the search space.

This analysis reveals that fitness-based search is easily led astray if fitness does not reward the stepping stones to the final solution, which is the case in the T-Maze learning problem

(Figure 9). Novelty search, on the other hand, escapes the deceptive trap and instead builds on the intermediate stepping stones to proceed through the search space more efficiently. Novelty search's ability to keep track of already-explored regions in the search space is probably another factor that accounts for its superior performance.

While in some domains the fitness gradient can be improved, i.e. by giving the objective-based search clues in which direction to search, such an approach might not be possible in dynamic, reward-based scenarios. The problem in such domains is that reaching a certain fitness level is relatively easy, but any further improvement requires sophisticated adaptive behavior to evolve from only sparse feedback from an objective-based performance measure. That is, novelty search returns more *information* about how behavior changes throughout the search space.

In this way, novelty search removes the need to carefully design a domain that fosters the emergence of learning because novelty search on its own is capable of doing exactly that. The only prerequisite is that the novelty metric is constructed such that learning and non-learning agents are separable, which is not necessarily easy, but is worth the effort if objective-based search would otherwise fail.

In fact, because NEAT itself employs the *fitness sharing* diversity maintenance technique (Goldberg & Richardson, 1987; Stanley & Miikkulainen, 2002) within its species (Section 2.3), the significant difference in performance between NEAT with novelty search and NEAT with fitness-based search also suggests that traditional diversity maintenance techniques do not evade deception as effectively as novelty search. Interestingly, novelty search has also been shown to succeed independently of NEAT (Mouret, 2009) in evolving ANNs and it also outperforms fitness-based search in genetic programming (Lehman & Stanley, 2010b). Thus evidence is building for its generality.

Novelty search's ability to build gradients that lead to stepping stones is evident in performance curves (Figure 6). The increase in generalization performance is steeper than for fitness-based NEAT, indicating a more efficient climb to higher complexity behaviors. In effect, by abandoning the objective, the stepping stones come into greater focus (Lehman & Stanley, 2008, 2010a). Although it means that the search is wider, the alternative is to be trapped by deception.

Of course, there are likely domains for which the representation is not suited to discovering the needed adaptive behavior or in which the space of behaviors is too vast for novelty search to

reliably discover the right one. However, even in the double T-Maze domain in which the length of the behavioral characterization is substantially larger (i.e. 400 dimensions), novelty search still significantly outperforms fitness-based search. There are only so many ways to behave and therefore the search for behavioral novelty becomes computationally feasible and is different than random search. On the other hand, even though novelty search is still significantly faster in the foraging bee task, fitness-based search reaches a local optimum that is very close to the final solution in about the same number of evaluations. A possible explanation for the more even performance in this domain is that the noisy environment offers a vast space of exploitable behavioral strategies. Future research will address the problem of noise in novelty search in more detail.

Overall, the results in this paper are important because research on evolving adaptive agents has been hampered largely as a result of the deceptiveness of adaptive tasks. Yet the promise of evolving plastic ANNs is among the most intriguing in artificial intelligence. After all, our own brains are the result of such an evolutionary process. Therefore, a method to make such domains more amenable to evolution has the potential to further unleash a promising research direction that is only just beginning. To explore this opportunity, a promising future direction is to apply novelty search to other adaptive problems without the need to worry about mitigating their potential for deception.

For example, an ambitious domain that may benefit from this approach is to train a simulated biped to walk *adaptively*. Lehman and Stanley (2010a) already showed that novelty search significantly outperforms objective-based search in a biped walking task. However, as in previous work (Bongard & Paul, 2001; Hase & Yamazaki, 1999; Reil & Husbands, 2002), static ANNs were evolved. Although plastic biped-controlling ANNs have been evolved in the past (Ishiguro, Fujii, & Hotz, 2003; McHale & Husbands, 2004), new advances in evolving neuromodulated ANNs (Dürr et al., 2008; Soltoggio et al., 2008) can potentially allow such controllers to be more robust to environmental changes and to morphological damage. Moreover, unlike past evolved biped controllers, such networks could be deployed into a wide range of body variants and seamlessly adapt to their bodies of origin, just as people can walk as they grow up through a wide array of body sizes and proportions. As is common when novelty search succeeds, this adaptive domain likely suffers from deception. Initially, the robot simply falls down on every attempt, obfuscating to the objective function any improvements in leg oscillation (Panne & Lamouret, 1995). However, after exploring all the different ways to fall down, novelty search

should lead evolution to more complex behaviors and eventually to walking. Novelty search in combination with neuromodulated ANNs might be the right combination to evolve such a new class of plastic controllers.

Characterizing when and for what reason novelty search fails is also an important future research direction. Yet its performance in this paper and in past research (Lehman & Stanley, 2008, 2010b, 2010c; Mouret, 2009) has proven surprisingly robust. While it is not always going to work well, this paper suggests that it is a viable new tool in the toolbox of evolutionary computation to counteract the deception inherent in evolving adaptive behavior.

9 Conclusions

This paper showed that (1) reinforcement learning problems like the T-Maze and bee foraging task are inherently deceptive for fitness-based search and (2) novelty search can avoid such deception by exploiting intermediate stepping stones to guide its search. A detailed analysis revealed how the two different approaches explore the genotypic space and demonstrated that novelty search, which abandons the objective to search only for novel behaviors, in fact facilitates the evolution of adaptive behavior. Results in a variety of domains demonstrated that novelty search can significantly outperform objective-based search and that it performs consistently under varying levels of domain deception. Fitness-based NEAT on the other hand performs increasingly poorly as domain deception increases, adding to the growing body of evidence (Lehman & Stanley, 2008, 2010b, 2010c; Mouret, 2009) that novelty search can overcome the deception inherent in a diversity of tasks. The main conclusion is that it may now be more realistic to learn interesting adaptive behaviors that have been heretofore seemingly too difficult.

Acknowledgments

This research was partially supported by the National Science Foundation under grants DRL0638977 and IIP0750551 and in part by DARPA under grant HR0011-09-1-0045 (Computer Science Study Group Phase 2). Special thanks to Andrea Soltoggio for sharing his wisdom in the T-Maze and bee domains.

References

Aaltonen, T., et al. (2009). Measurement of the top quark mass with dilepton events selected using neuroevolution at CDF. *Physical Review Letters*.

Angeline, P. J., Saunders, G. M., & Pollack, J. B. (1994). An evolutionary algorithm that constructs recurrent neural networks. *Neural Networks, IEEE Transactions on*, *5*(1), 54–65.

Barlow, M., Galloway, J., & Abbass, H. A. (2002). Mining evolution through visualization. In E. Bilotta et al. (Eds.), *ALife VIII Workshops* (pp. 103–112).

Barnett, L. (2001). Netcrawling-optimal evolutionary search with neutral networks. In *Evolutionary computation, 2001. proceedings of the 2001 congress on* (Vol. 1).

Baxter, J. (1992). The evolution of learning algorithms for artificial neural networks. In D. Green & T. Bossomaier (Eds.), *Complex Systems* (pp. 313–326). IOS Press.

Blynel, J., & Floreano, D. (2002). Levels of Dynamics and Adaptive Behavior in Evolutionary Neural Controllers. In *From Animals to Animats 7.* (In B. Hallam, D. Floreano, J. Hallam, G. Hayes, and J.-A. Meyer (eds))

Blynel, J., & Floreano, D. (2003). Exploring the T-Maze: Evolving learning-like robot behaviors using CTRNNs. In *2nd European Workshop on Evolutionary Robotics (EvoRob 2003).*

Bongard, J. C., & Paul, C. (2001). Making evolution an offer it can't refuse: Morphology and the extradimensional bypass. In *Proceedings of the 6th European Conference on Advances in Artificial Life (ECAL 2001)* (pp. 401–412). London, UK: Springer-Verlag.

Carew, T., Walters, E., & Kandel, E. (1981). Classical conditioning in a simple withdrawal reflex in Aplysia californica. *The Journal of Neuroscience*, *1*(12), 1426-1437.

Chalmers, D. J. (1990). The evolution of learning: An experiment in genetic connectionism. In *Proceedings of the 1990 Connectionist Models Summer School* (pp. 81–90). Morgan Kaufmann.

Darwen, P., & Yao, Y. (1996). Every niching method has its niche: Fitness sharing and implicit sharing compared. *Parallel Problem Solving from Nature (PPSN IV)*, 398–407.

De Jong, E. D. (2004). The incremental pareto-coevolution archive. In *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO 2004)* (p. 525-536). Springer.

Dürr, P., Mattiussi, C., Soltoggio, A., & Floreano, D. (2008). Evolvability of neuromodulated learning for robots. In *The 2008 ECSIS Symposium on Learning and Adaptive Behavior*

in Robotic Systems (pp. 41–46). Los Alamitos, CA: IEEE Computer Society.

Dybowski, R., Collins, T. D., Hall, W., & Weller, P. R. (1996). Visualization of binary string convergence by sammon mapping. In *Proceedings of The Fifth Annual Conference on Evolutionary Programming.* MIT Press.

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

Floreano, D., & Urzelai, J. (2000). Evolutionary robots with online self-organization and behavioral fitness. *Neural Networks*, *13*, 431-443.

Glover, F., & Laguna, M. (1997). Boston: Kluwer Academic Publishers.

Goldberg, D. E. (2007). Simple genetic algorithms and the minimal deceptive problem. In L. D. Davis (Ed.), *Genetic Algorithms and Simulated Annealing, Research Notes in Artificial Intelligence.* Morgan Kaufmann.

Goldberg, D. E., & Richardson, J. (1987). Genetic algorithms with sharing for multimodal function optimization. In *Proceedings of the Second International Conference on Genetic Algorithms and their Application* (pp. 41–49). Hillsdale, NJ, USA: L. Erlbaum Associates Inc.

Gomez, F. J., & Miikkulainen, R. (1999). Solving non-markovian control tasks with neuroevolution. In *Proceedings of the 16th International Joint Conference on Artificial Intelligence* (pp. 1356–1361). Morgan Kaufmann.

Gruau, F., Whitley, D., & Pyeatt, L. (1996). A comparison between cellular encoding and direct encoding for genetic neural networks. In *Genetic Programming 1996: Proceedings of the First Annual Conference* (pp. 81–89). MIT Press.

Harvey, I. (1993). *The artificial evolution of adaptive behavior.* Unpublished doctoral dissertation, School of Cognitive and Computing Sciences, University of Sussex, Sussex.

Hase, K., & Yamazaki, N. (1999). Computational evolution of human bipedal walking by a neuro-musculo-skeletal model. *Artificial Life and Robotics*, *3*, 133-138.

Hinton, G. E., & Nowlan, S. J. (1987). How learning can guide evolution. *Complex Systems*, *1*.

Hornby, G. S. (2006). ALPS: The age-layered population structure for reducing the problem of premature convergence. In *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO 2006)* (pp. 815–822). New York, NY, USA: ACM.

Hu, J., Goodman, E., Seo, K., Fan, Z., & Rosenberg, R. (2005). The hierarchical fair competition

(HFC) framework for sustainable evolutionary algorithms. *Evolutionary Computation*, *13*(2), 241-277. (PMID: 15969902)

Hutter, M., & Legg, S. (2006). Fitness uniform optimization. *IEEE Transactions on Evolutionary Computation*, *10*, 568–589.

Ishiguro, A., Fujii, A., & Hotz, P. E. (2003). Neuromodulated control of bipedal locomotion using a polymorphic CPG circuit. *Adaptive Behavior*, *11*(1), 7-17.

Kim, Y.-H., & Moon, B.-R. (2003). New Usage of Sammon's Mapping for Genetic Visualization. In *Proceedings Genetic and Evolutionary Computation (GECCO 2003)* (Vol. 2723 of LNCS, p. 1136-1147). Springer-Verlag.

Kittler, J., & Young, P. C. (1973). A new approach to feature selection based on the Karhunen-Loeve expansion. *Pattern Recognition*, *5*, 335–352.

Lehman, J., & Stanley, K. O. (2008). Exploiting open-endedness to solve problems through the search for novelty. In *Proceedings of the Eleventh International Conference on Artificial Life.* Cambridge, MA: MIT Press.

Lehman, J., & Stanley, K. O. (2010a). Abandoning objectives: Evolution through the search for novelty alone. *Evolutionary Computation.* (To appear)

Lehman, J., & Stanley, K. O. (2010b). Efficiently evolving programs through the search for novelty. In *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO 2010).* New York, NY, USA: ACM. (To appear)

Lehman, J., & Stanley, K. O. (2010c). Revising the evolutionary computation abstraction: Minimal criteria novelty search. In *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO 2010).* New York, NY, USA: ACM. (To appear)

Mahfoud, S. W. (1995). *Niching methods for genetic algorithms.* Unpublished doctoral dissertation, Champaign, IL, USA.

Martin, A. P. (1999). Increasing genomic complexity by gene duplication and the origin of vertebrates. *The American Naturalist*, *154*(2), 111-128.

Mayley, G. (1997). Guiding or hiding: Explorations into the effects of learning on the rate of evolution. In *Fourth European Conference on Artificial Life* (pp. 135–144). MIT Press.

McHale, G., & Husbands, P. (2004). Gasnets and other evolvable neural networks applied to bipedal locomotion. *From Animals to Animats 8*.

McQuesten, P., & Miikkulainen, R. (1997). Culling and teaching in neuro-evolution. In *Proceedings of the Seventh International Conference on Genetic Algorithms.* San Francisco:

Kaufmann.

Mitchell, M., Forrest, S., & Holland, J. H. (1991). The royal road for genetic algorithms: Fitness landscapes and GA performance. In *Proceedings of the First European Conference on Artificial Life* (pp. 245–254). MIT Press.

Mouret, J. (2009). Novelty-based multiobjectivization. In *Proceedings of iros workshop on exploring new horizons in the evolutionary design of robots.*

Niv, Y., Joel, D., Meilijson, I., & Ruppin, E. (2002). Evolution of reinforcement learning in uncertain environments: A simple explanation for complex foraging behaviors. *Adaptive Behavior*, *10*(1), 5–24.

Nolfi, S., & Floreano, D. (1999, July). Learning and evolution. *Autonomous Robots*, *7*(1), 89-113.

Nolfi, S., & Parisi, D. (1993). Auto-teaching: Networks that develop their own teaching input. In B. H. G. S. N. G. Deneubourg J. L. & R. Dagonnier (Eds.), *Proceedings of the Second European Conference on Artificial Life* (pp. 845–862.).

Nolfi, S., & Parisi, D. (1996). Learning to adapt to changing environments in evolving neural networks. *Adaptive Behavior*, *5*, 75–98.

Nolfi, S., Parisi, D., & Elman, J. L. (1994). Learning and evolution in neural networks. *Adaptive Behavior*, *3*, 5-28.

Panne, M. van de, & Lamouret, A. (1995, september). Guided optimization for balanced locomotion. In *6th Eurographics Workshop on Animation and Simulation.*

Reil, T., & Husbands, P. (2002). Evolution of central pattern generators for bipedal walking in a real-time physics environment. *IEEE Trans. Evolutionary Computation*, *6*(2), 159–168.

Sammon, J. W. (1969). A nonlinear mapping for data structure analysis. *IEEE Trans. Comput.*, *18*(5), 401–409.

Saravanan, N., & Fogel, D. B. (1995). Evolving neural control systems. *IEEE Expert: Intelligent Systems and Their Applications*, *10*(3), 23–27.

Schmidhuber, J. (2003). Exploring the predictable. In S. Ghosh & S. Tsutsui (Eds.), *Advances in evolutionary computing: theory and applications* (pp. 579–612). New York, NY, USA: Springer-Verlag New York, Inc.

Schmidhuber, J. (2006). Developmental robotics, optimal artificial curiosity, creativity, music, and the fine arts. *Connection Science*, *18*(2), 173–187.

Soltoggio, A., Bullinaria, J. A., Mattiussi, C., Dürr, P., & Floreano, D. (2008). Evolutionary

advantages of neuromodulated plasticity in dynamic, reward-based scenarios. In *Artificial Life XI* (pp. 569–576). Cambridge, MA: MIT Press.

Soltoggio, A., Dürr, P., Mattiussi, C., & Floreano, D. (2007). Evolving neuromodulatory topologies for reinforcement learning-like problems. In *Proceedings of the IEEE Congress on Evolutionary Computation.*

Stanley, K. O. (2006-2008). rtNEAT C++ software homepage: **www.cs.utexas.edu/users/nn/keyword?rtneat**.

Stanley, K. O., Bryant, B. D., & Miikkulainen, R. (2003). Evolving adaptive neural networks with and without adaptive synapses. In *Proceedings of the 2003 IEEE Congress on Evolutionary Computation (CEC-2003).* Canberra, Australia: IEEE Press.

Stanley, K. O., Bryant, B. D., & Miikkulainen, R. (2005, December). Real-time neuroevolution in the NERO video game. *IEEE Transactions on Evolutionary Computation*, *9*(6), 653-668.

Stanley, K. O., & Miikkulainen, R. (2002). Evolving neural networks through augmenting topologies. *Evolutionary Computation*, *10*, 99-127.

Stanley, K. O., & Miikkulainen, R. (2004). Competitive coevolution through evolutionary complexification. *Journal of Artificial Intelligence Research*, *21*, 63–100.

Stewart, T. (2001). Extrema selection: Accelerated evolution on neutral networks. In *Proceedings of the 2001 ieee international conference on evolutionary computation* (Vol. 1). IEEE Press.

Taylor, M. E., Whiteson, S., & Stone, P. (2006, July). Comparing evolutionary and temporal difference methods in a reinforcement learning domain. In *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO 2006)* (p. 1321-1328). New York, NY, USA: ACM.

Vassilev, V. K., Fogarty, T. C., & Miller, J. F. (2000). Information characteristics and the structure of landscapes. *Evol. Comput.*, *8*(1), 31–60.

Watson, J. D., Hopkins, N. H., Roberts, J. W., Steitz, J. A., & Weiner, A. M. (1987). *Molecular biology of the gene fourth edition.* Menlo Park, CA: The Benjamin Cummings Publishing Company, Inc.

Whiteson, S., & Stone, P. (2006). Evolutionary function approximation for reinforcement learning. *J. Mach. Learn. Res.*, *7*, 877–917.

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