Efficiently Evolving Programs through the Search for Novelty

In: *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO 2010).* New York, NY:ACM

Joel Lehman
University of Central Florida
4000 Central Florida Blvd.
Orlando, FL 32816-2362 USA
jlehman@eecs.ucf.edu

Kenneth O. Stanley
University of Central Florida
4000 Central Florida Blvd.
Orlando, Ohio 32816-2362 USA
kstanley@eecs.ucf.edu

ABSTRACT

A significant challenge in genetic programming is premature convergence to local optima, which often prevents evolution from solving problems. This paper introduces to genetic programming a method that originated in neuroevolution (i.e. the evolution of artificial neural networks) that circumvents the problem of deceptive local optima. The main idea is to search *only* for behavioral novelty instead of for higher fitness values. Although such *novelty search* abandons following the gradient of the fitness function, if such gradients are deceptive they may actually *occlude* paths through the search space towards the objective. Because there are only so many ways to behave, the search for behavioral novelty is often computationally feasible and differs significantly from random search. Counterintuitively, in both a deceptive maze navigation task and the artificial ant benchmark task, genetic programming with novelty search, which ignores the objective, outperforms traditional genetic programming that directly searches for optimal behavior. Additionally, novelty search evolves smaller program trees in every variation of the test domains. Novelty search thus appears less susceptible to bloat, another significant problem in genetic programming. The conclusion is that novelty search is a viable new tool for efficiently solving some deceptive problems in genetic programming.

Categories and Subject Descriptors

I.2.6 [**Artificial Intelligence**]: Learning

General Terms

Algorithms

Keywords

Genetic programming, Premature convergence, Program bloat, Novelty search

1. INTRODUCTION

A prominent problem in genetic programming (GP) is premature convergence to local optima [6, 12, 14, 22], wherein genetic diversity is extinguished before the objective is discovered, leaving the search trapped in a dead end. Such premature convergence is caused by individuals that appear promising yet have only a weak relationship to the ultimate objective of the search. Prior experiments in neuroevolution (i.e. the evolution of artificial neural networks) have shown that sometimes searching only for behavioral novelty can effectively circumvent such deception [13, 19, 21, 25]. The idea in this paper is to introduce novelty search to GP as a new tool for solving deceptive genetic programming problems.

Fitness functions that quantify and encourage progress towards the goal of an evolutionary search are ubiquitous in evolutionary computation (EC) and GP. Such fitness functions are instances of *objective functions*, which estimate the distance to an objective and are pervasive throughout machine learning [18, pp. 97–100,188–190,194,236–238,255–266,370–373]. However, by optimizing an objective function, search can be deceived into converging to local optima from which no local step in the search space can improve the value of the objective function. The problem is that the objective function does not necessarily reward the *stepping stones* in the search space that ultimately lead to the objective.

In this way, guiding search by the compass of fitness may often prune stepping stones from the search space; individuals that are stepping stones but have low fitness may be discarded in favor of high-fitness individuals that represent local optima in the search space. The effect of such pruning is to progressively *close* the search space by concentrating on increasingly specific areas. In contrast, natural evolution, which has no final objective, is more *open-ended* and maintains an increasing diversity of individuals [3, 27].

An important property of natural evolution is that it continually produces novel forms [3, 27]; throughout evolution, an increasing variety of such novel forms have been maintained, which act as stepping stones to further novelty. Thus, instead of funneling search towards local optima as in objective-based search, natural evolution tends to *diverge*, continually finding new organisms that live in different ways.

This perspective leads to the key idea: Instead of searching directly for the objective, which through deception may prevent the objective from actually being reached, searching only for novelty may itself be a viable approach to search. Thus this paper introduces to GP the *novelty search* algorithm, which searches with no objective other than continually finding novel behaviors in the search space. Such an algorithm is immune to deception because it searches for nothing in particular, i.e. only what is different from what has been seen before. Furthermore, because there are only

GECCO'10 July 7–11, 2010, Portland, Oregon USA.

so many ways to behave, a search for novelty may discover the solution to a problem without any direct selection pressure towards that solution. Finally, novelty search avoids premature convergence because it is a *divergent* algorithm; the novelty of a behavior is inversely proportional to the amount of individuals that display that behavior.

To demonstrate the effectiveness of novelty search, in this paper it is compared to traditional objective-based search and random search in a deceptive robot maze navigation task [13, 19] and the artificial ant benchmark task [4, 9, 12, 20]. Counterintuitively, novelty search, which ignores the objective, evolves successful maze navigators and food-gathering ants more consistently.

Yet this paper goes beyond confirming that novelty search works in GP as as in neuroevolution. Another important discovery is that the solutions evolved by novelty search have consistently smaller program trees. An examination of program growth dynamics reveals that novelty search does not seem to be susceptible to continuous program bloat in the same way that fitness-based search is. This result is important because bloat is a significant problem in GP [11, 23, 26].

The conclusion is that by abstracting the process through which natural evolution discovers novelty, it is possible to derive an open-ended search algorithm that solves some GP problems more consistently and with more succinct program trees than a method that directly searches for the objective. Novelty search overcomes the problems of deception and local optima inherent in objective optimization by ignoring the objective, and may be a promising new tool for efficiently solving deceptive GP problems.

2. BACKGROUND

This section reviews deception in GP, examines how the search space can be made more open in a principled way, and explains novelty search.

2.1 Deception in Genetic Programming

The original definition of deception by Goldberg [7] is based on the building blocks hypothesis, in which small genetic building blocks may be integrated by crossover to form larger blocks [8]. In the original conception, a problem is deceptive if lower-order building blocks, when combined, do not lead to a global optima. The basic idea is that search may be *deceived* if there is no path through the search space from a seemingly-promising preliminary individual to an individual that solves the problem.

Deception is related to premature convergence, a challenging problem in GP [1, 2, 6, 12, 14, 15, 22]. GP and other search methods are often seen as operating in two phases, an exploration phase followed by an exploitation phase. In the exploration phase, a diversity of potential solutions are examined, while in the exploitation phase, diversity is sacrificed to allow more intensive search in promising areas of the search space. Premature convergence results when none of the exploited areas of the search space lead to a solution.

The relationship between premature convergence and deception is that a deceptive fitness function will create many local optima to which search may converge, making premature convergence more likely. The pathologies of deception and premature convergence have been documented for GP applied to various domains, such as the artificial ant problem [12], the MAX problem [6], RoboCup soccer [14], and minimal sorting networks [22].

Because premature convergence is an acknowledged problem in GP, there exist a variety of methods to mitigate such convergence [1, 2, 15, 22]. Researchers may combat premature convergence by adjusting the fitness function through trial and error [14]; higher mutation rates may increase diversity [1], as may larger population sizes [15]. Other approaches actively encourage diversity by replacing the most similar programs [15], maintaining a variety of genetic lineages [2], or maintaining a variety of behaviors [28].

However, in all of these methods, the radical step of eschewing the fitness function is not taken. Yet the problem is that the fitness function is the root cause of the deceptive local optima that act as attractors to objective-based search; if the stepping stones needed to solve a problem do not lie conveniently along the gradient of the fitness function, search may fail. Thus, given a sufficiently uninformative fitness function, these methods are still vulnerable to early convergence because they still rely on the potentially misleading compass of fitness.

The next section describes ways in which the search space can be opened up to allow more points in the search space to be reachable by the search algorithm.

2.2 Opening Up the Search Space

Most EC algorithms are *convergent*; selection pressure in such algorithms prunes an increasing amount of the search space to focus resources on highly-specific high-fitness areas. That is, selection pressure *restricts* search by mandating that only high-fitness individuals and their high-fitness offspring will be considered for future evolution [16]. Thus search can fail when selection pressure prunes areas of the search space that contain the stepping stones necessary to reach the ultimate objective.

The problem is that the fitness function is only a heuristic measure of progress to the goal; sometimes improving fitness requires taking steps in the search space that at first appear deleterious [24]. That is, sometimes *relaxing* the pressure to perform is beneficial to search because it *opens up* the search space (i.e. more points in the search space are reachable by the algorithm), which may allow it to escape from local optima. However, when such relaxation is maximized and every point in the search space is equally viable, the result is a random search: The search space is completely open because every point is reachable, but the search is inefficient and without principle. In other words, there is a trade-off in objective-based search between exploration of the search space and exploitation of promising areas.

However, in deceptive problems, the compass of fitness fails to reward the stepping stones needed to ultimately solve the problem. If these stepping stones lie *far* from the paths through the search space defined by the gradients of the fitness function, then it may take an unreasonably long time for *any* amount of exploration to happen upon them by chance. In these cases, the most logical step, although it may at first appear radical, is to discard the flawed guidance of objective fitness.

What is needed to replace objective fitness is a principled way to guide search that respects the stepping stones by *opening up* the search space. In this spirit, it is instructive to consider natural evolution, the prototypical example of a search process without a final objective.

In contrast to most EC models, natural evolution is *divergent*, finding a seemingly boundless variety of living or-

ganisms; that is, evolution continually produces novel forms [27]. While selection driven by a traditional fitness function *closes* the search space by always funneling search towards the increasingly fewer individuals that have better fitness, the way by which natural evolution discovers novelty is open-ended; each novel organism is a potential jumping-off point for discovering further novelty. In this way, the search space is more *open* because there are more potential stepping stones; each novel organism allows for new opportunities. These considerations led Lehman and Stanley [13] to conclude that searching for novelty may itself be a powerful search technique.

The next section explains such novelty search, which will be combined with GP in the experiments in this paper.

2.3 The Search for Novelty

Recall that the problem identified with the objective function 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.

The suggested approach is to identify behavioral novelty as a *proxy* for stepping stones [13]. That is, instead of searching for a final objective, the learning method is rewarded for finding any instance 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. If natural evolution is abstracted as a process that passively accumulates novel forms, novelty search is an explicit *acceleration* of such accumulation.

For example, in a maze navigation domain, initial attempts might run into a wall and stop. In contrast to an objective function, the novelty metric would reward simply running into a *different* wall regardless of whether it is closer to the goal or not. In this kind of search, individuals are maintained that represent the most novel discoveries. Further search then jumps off from these behaviors. After a few ways to run into walls are discovered, the only way to be rewarded is to find a behavior that does not hit a wall right away. Eventually, to do something new, a navigator will have to successfully navigate the maze even though it is not an objective.

Novelty search succeeds where objective-based search fails by rewarding the stepping stones. That is, anything that is genuinely different is rewarded and promoted as a jumping-off point for further evolution. While we cannot know which stepping stones are the right ones, if we accept that the primary pathology in objective-based search is that it cannot detect the stepping stones at all, then that pathology is remedied.

In fact, there have been several successful applications of novelty search in neuroevolution [13, 19, 21]. Novelty search was introduced in Lehman and Stanley [13] and applied to a deceptive maze task in a continuous world with real-valued sensors and effectors (in contrast to the discrete version of the maze task introduced here); these results were replicated in Mouret [19] and combined with a multi-objective evolutionary algorithm. Two independent works have also demonstrated that behavioral novelty is useful in evolving adaptive neural networks (i.e. neural networks that learn during an evaluation) [21, 25].

The next section introduces the novelty search algorithm in the context of GP by replacing the objective function with the novelty metric and formalizing the concept of novelty.

3. THE NOVELTY SEARCH ALGORITHM FOR GP

Evolutionary algorithms such as GP in general are well-suited to novelty search because the population that is central to such algorithms naturally covers a range of behaviors. In fact, tracking novelty requires little change to any evolutionary algorithm aside from replacing the fitness function with a *novelty metric*. In the past this idea was applied to neuroevolution, which evolves the topologies and weights of neural networks, suggesting that novelty search should also work when evolving the structure of programs.

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, novelty search rewards diverging from prior behaviors. Therefore, novelty needs to be *measured*. It is important to note that this conception of novelty is the same as it was in neuroevolution (i.e. rewarding novel behaviors, *not* novel genotypes [13]).

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 *behaviors* (i.e. *not* the programs themselves) of an *archive* of past individuals and the current population. The archive is initially empty, and individuals' behaviors are added to it probabilistically to penalize future individuals that exhibit previously explored behaviors.

The novelty metric is designed to characterize how far away the new individual is from the rest of the population and its predecessors in *behavior space*, i.e. the space of unique behaviors. A good metric should thus compute the *sparseness* at any point in the behavior 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

$$\rho(x) = \frac{1}{k}\sum_{i=0}^{k} \text{dist}(x, \mu_i), \tag{1}$$

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 previous work, individuals were added to the permanent archive either deterministically if their novelty score was sufficiently high or nondeterministically with a small probability [13, 21]. However, adding a highly novel individual to the archive has the disadvantage that it penalizes an area of the behavior space that may merit further exploration. Thus, in the experiments in this paper, *every* individual is eligible to be added to the archive with a fixed probability. The effect of such probabilistic sampling is that the permanent archive approximately characterizes the distribution of prior solutions in behavior space without pushing search away from newly discovered areas. 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.

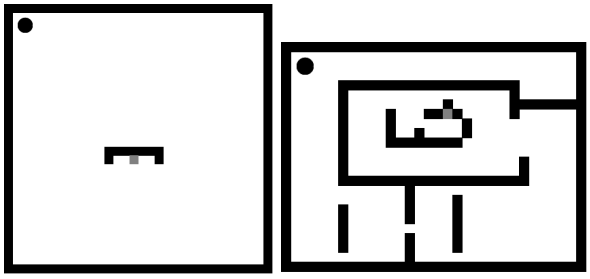
The novelty search approach in general allows any behavior characterization and any novelty metric. Although generally applicable, novelty search is best suited to domains with deceptive fitness landscapes, domain constraints on possible behaviors, and intuitive behavioral characterization. Changing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

One might argue that this approach simply replaces one objective function with another, because novelty is being maximized just as fitness was. However, conceptually the two are different; novelty creates a gradient of behavioral difference, whereas fitness creates a gradient towards the objective. Maximizing fitness is done with the intent of bringing the search towards the objective (i.e. a specific region of the search space), while maximizing novelty is done without any concept of where the search should terminate or even what general direction it should take within the search space. In other words, novelty search does not describe a *point or set of points* (i.e. the objective) in the search space towards which the search is aimed.

Once objective-based fitness is replaced with novelty, the underlying GP algorithm operates as normal, selecting the most novel individuals to reproduce. Over generations, the population spreads out across the space of possible behaviors. In effect, because novelty search is a generic algorithm that does not operate on the underlying representation, it is easily applied to GP without significant modification.

4. EXPERIMENTS

Good domains for testing novelty search should have deceptive fitness landscapes. In such domains, search algorithms following the gradient of fitness may perform worse than the search for novelty because novelty search cannot be deceived; it ignores the objective entirely. Furthermore, to examine whether the performance of novelty search in GP is different from that of a random search, as was demonstrated in prior neuroevolution experiments [13], these experiments also test GP with an alternate reward scheme that assigns a random fitness value to each individual.



(a) Medium Map (b) Hard Map

Figure 1: Maze Navigation Maps. In both maps, the circle represents the starting position of the robot and the grey square represents the goal that the robot attempts to reach. Black squares are walls. Cul-de-sacs and occlusions in both maps that prevent direct passage to the goal create the potential for deception.

An easily-visualized domain with the potential for deception is a two-dimensional maze navigation task. A reasonable fitness function for such a domain is how close the maze navigator is to the goal at the end of the evaluation. Thus, dead ends or occlusions that are near to the goal are local optima to which an objective-based algorithm may converge, which makes a good model for deceptive problems in general. This domain is similar to the maze problem described in Soule [26], and was also tested successfully with novelty search in neuroevolution [13].

Another deceptive domain is the artificial ant problem, in which an ant attempts to navigate a trail while collecting food. It is a common GP benchmark domain [4, 9, 12, 20], and is so deceptive that GP often performs little better than random search [12].

The experiments in this paper were conducted with a version of the lilGP genetic programming distribution [30] extended with an implementation of novelty search that is generational as in Mouret [19], as opposed to the steady-state implementation in Lehman and Stanley [13].

4.1 Maze Experiment

The maze domain works as follows. A robot controlled by a genetic program must navigate from a starting point to an end point within a fixed amount of time. The task is complicated by occlusions and cul-de-sacs that prevent a direct route and create local optima in the fitness landscape. The robot can move forward, turn, and act conditionally based on whether there is a wall directly in front of it or not. A robot is successful in the task if it reaches the goal location. This setup is similar to the original novelty search neuroevolution experiment in Lehman and Stanley [13]. Table 1 describes the parameters of the experiment.

Two maps are designed to compare the performance of GP with fitness-based search and GP with novelty search. The first (figure 1a) has an occlusion that blocks the most direct path to the goal. To reach the goal, the robot must learn a behavior unrelated to the locally optimal behavior of simply crashing into the occlusion. The second maze (figure 1b) provides a more deceptive fitness landscape that requires the search algorithm to explore areas of significantly lower fitness before finding the global optimum (which is a network that reaches the goal).

Fitness-based GP, which will be compared to novelty search, requires a fitness function to reward maze-navigating robots.

Objective:	Find a robot that navigates the maze
Terminal set:	Left (turn left), Right (turn right), Move (move forward one square)
Functions set:	IfWallAhead (execute one of two child instructions based on whether there is a wall directly ahead), Prog2 (sequentially execute the two child instructions)
Fitness cases:	Medium Maze and Hard Maze
Wrapper:	Program repeatedly executed for 100 time steps for the medium maze or 400 time steps for the hard maze
Population Size:	$1,000$
Termination:	Maximum number of generations $= 1,000$

Table 1: Parameters for the Maze Problem.

Because the objective is to reach the goal, the fitness f is defined as the distance from the robot to the goal at the end of an evaluation: $f = b_f - d_g$, where b_f is the maximum distance possible and d_g is the distance from the robot to the goal. Given a maze with no deceptive obstacles, this fitness function defines a monotonic gradient for search. The constant b_f ensures all individuals will have positive fitness.

GP with novelty search, on the other hand, requires a novelty metric to distinguish between maze-navigating robots. Defining the novelty metric requires careful consideration because it biases the search in a fundamentally different way than the fitness function. The novelty metric determines the behavior-space through which search will proceed. It is important that the type of behaviors that one hopes to distinguish are recognized by the metric.

Thus, for the maze domain, the behavior of a navigator is defined as its ending position. The novelty metric is then the Euclidean distance between the ending positions of two individuals. For example, two robots stuck in the same corner appear similar, while one robot that simply sits at the start position looks very different from one that reaches the goal, though they are both equally viable to the novelty metric.

This novelty metric rewards the robot for ending in a place where none have ended before; the method of traversal is ignored. This measure reflects that what is important in a maze is reaching a certain location (i.e. the goal) rather than the method of locomotion. Thus, although the novelty metric has no knowledge of the final goal, a solution that reaches the goal will appear novel. Furthermore, the comparison between fitness-based and novelty-based search is fair because both scores are computed only based on the distance of the final position of the robot from other points.

4.2 Artificial Ant Experiment

In the artificial ant problem, a simulated ant embedded in a toroidal two-dimensional grid attempts to collect as much food as possible [4, 9, 12, 20]. The food is laid out in a trail in the grid, but there are gaps in the trail such that the agent must *infer* the missing steps of the trail. The problem is deceptive because policies with no principle can nonetheless capture many units of food by chance although they are not ultimately precursors to a policy able to capture *all* the food. The ant is controlled by a genetic program that allows it to turn, move, and conditionally act based upon whether food is in front of it or not. The parameters of the artificial ant experiment are described in table 2.

The artificial ant experiment in this paper follows the formulation in Koza [9] and is attempted on both the Sante Fe and Los Altos trails. The Santa Fe trail is shorter, while the Los Altos trail is longer and harder to follow. In both maps, a successful individual is one that collects all of the food.

Fitness-based GP requires a fitness function to reward ants. Because the objective is to collect all of the food in the trail, the fitness function is the amount of food that the ant collects, which is customary on this problem [9].

GP with novelty search requires a novelty metric to distinguish between ant behaviors so that novelty can be quantified. A simple way to characterize behavior in this domain is to sample how much food has been collected by the ant at several evenly-spaced times during an evaluation. This measure frames behavior in terms of collecting food, which is the action of interest; a novel individual must do something *different* with respect to collecting food.

Sampling this value over time allows search to differentiate behaviors that ultimately collect the same amount of food by different means. It is important to note that characterizing behavior as the amount of food collected does *not* mean that novelty search will necessarily seek higher values of food collected, but rather *different* temporal sequences of food collected during an evaluation. The novelty metric is the same as in the maze domain (i.e. the Euclidean distance between two behavioral characterization vectors).

4.3 Experimental Parameters

All experiments were run with a modified version of the lilGP package with the standard settings according to Koza [10], including tournament selection and half and half initialization. The number of generations was significantly extended from the formulation in Koza [9] from 50 to $1,000$ so that long-term program growth patterns could be observed. The population size in all experiments was $1,000$.

Because GP with novelty search differs from traditional GP only in substituting a novelty metric for a fitness function, they both share the same parameters from Koza [9] except that novelty search is given a smaller tournament size (two), which reduces selection pressure so that what is novel does not drastically change from generation to generation. It is important to note that when traditional fitness-based GP was given a smaller tournament size in preliminary experiments, overall performance was worse and the dynamics of program growth were unaffected.

The number of neighbors checked in novelty search, k, was set to 25, and is robust to moderate variation. An individual has a 0.05% chance to be added to the archive, which means a new archive point is added every two generations on average.

Objective:	Find an ant that follows food trails
Terminal set:	Left (turn left), Right (turn right), Move (move forward one square)
Functions set:	IfFoodAhead (execute one of two child instructions based on if there is food directly ahead), Prog2 (sequentially execute the two children instructions)
Fitness cases:	Santa Fe Trail and Los Altos Trail
Wrapper:	Program repeatedly executed for 400 time steps for Santa Fe Trail or 3,000 time steps for Los Altos Trail
Population Size:	1,000
Termination:	Maximum number of generations = 1,000

Table 2: Parameters for the Artificial Ant Problem.

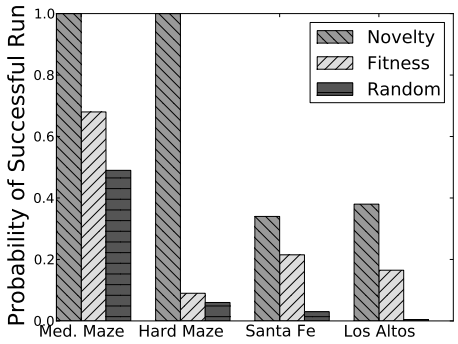


Figure 2: Performance Comparison. For each variation of the test domains, the proportion of the 200 runs that are successful are shown for fitness-based GP, GP with novelty search, and random-fitness GP. The main result is that GP with novelty search evolves solutions significantly more consistently than either fitness-based GP or random-fitness GP ($p < 0.01$; Fisher's exact test).

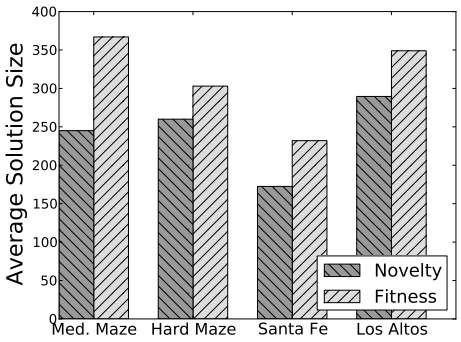


Figure 3: Bloat Comparison. For each variation of the test domains, the average size (number of nodes) of the first solutions in runs evolved by GP with novelty search and fitness-based GP is compared (lower is better). In every variation, novelty search evolves significantly smaller program trees ($p < 0.01$; Student's t-test).

In the artificial ant problem, the amount of food the ant collected was sampled eight times during an evaluation to construct the behavioral characterization vector, which proved robust to moderate variation during initial experiments.

5. RESULTS

200 trials were run for each variation of the maze and artificial ant problems with both fitness-based GP, GP with novelty search, and random-fitness GP. As illustrated by figure 2, in each task novelty search solves the task significantly more often than fitness-based GP and random-fitness GP ($p < 0.01$; Fisher's exact test).

Interestingly, an analysis of evolved solutions shows that for each of the domains, novelty search tends to evolve smaller program trees than fitness-based search, as shown in figure 3. Note that only the first successful individual from each run that solves the problem is included in these averages. The differences are significant for each domain ($p < 0.01$; Student's t-test).

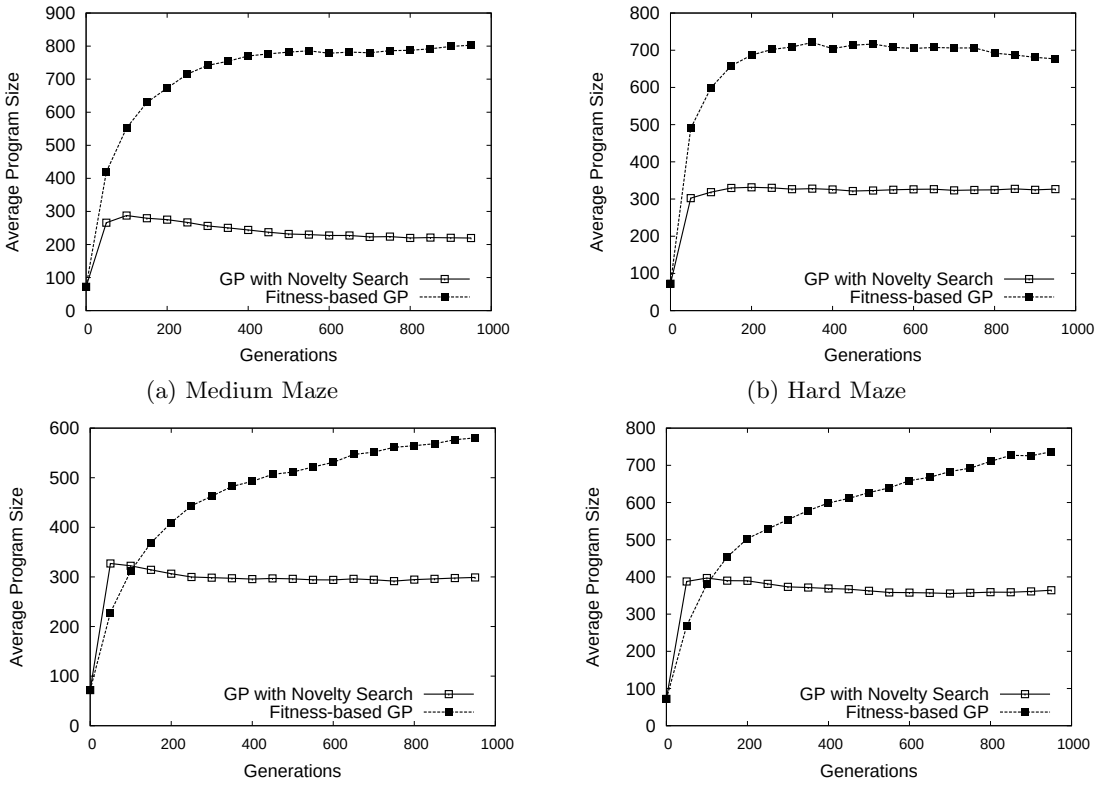
The dynamics of program growth during evolution are shown in figure 4. Interestingly, instead of growing larger as in fitness-based GP, program length stagnates and sometimes even *declines* in runs of GP with novelty search.

6. DISCUSSION

The failure of fitness-based GP on the harder maze might lead one to conclude that the Koza-style GP algorithm is at fault; however, the problem is solved consistently when all is kept the same except that behavioral novelty is rewarded instead of estimated progress to the objective. This result reconfirms the conclusion of Lehman and Stanley [13]: The problem is not in the search algorithm itself but in *how the search is guided*.

Precisely because objective fitness is a heuristic, there is no guarantee that following its gradient will lead to the objective. As John Stuart Mill once said, it is a logical fallacy to assume a priori that the "conditions of a phenomenon must, or at least probably will, resemble the phenomenon itself." [17, p. 470]; that is, the steps to the objective may not always resemble the objective. Novelty search makes no assumption about which behaviors will lead to the objective. In this way, search can progress through various stepping stones that superficially appear unrelated to the objective yet may be crucial in ultimately reaching it.

The claim is not that novelty search will always be better than an objective-based search; if the gradients of the objective function are not deceptive, then a search focused on the objective will likely find a solution faster than a



(a) Medium Maze (b) Hard Maze

(c) Santa Fe Trail (d) Los Altos Trail

Figure 4: Program Growth During Search. The average size (number of nodes) of a program tree in the population is shown as evolution progresses for GP with novelty search and fitness-based GP on the medium (a) and hard (b) mazes, and the Santa Fe (c) and Los Altos (d) ant trails, averaged over all 200 runs of each approach. The main result is that populations do not continually bloat with novelty search.

broader search for novelty. However, as problems increase in complexity and difficulty, it becomes increasingly difficult to craft a sufficiently accurate heuristic that does not deceive search by leading it to locally optimal dead ends [5, 29].

An interesting discovery in this investigation is that novelty search evolves smaller solutions than fitness-based search in every domain. Because bloat is a prominent problem in GP [11, 23, 26], this result further recommends novelty search for GP. Furthermore, the growth curves in figure 4 demonstrate that the reduced bloat in novelty search is not a simple consequence of it solving the problems faster. Rather, novelty search effectively contains bloat throughout the runs in every domain. A potential hypothesis to explain this phenomenon is that while static fitness functions promote bloat to protect program trees from destructive crossover [11], in novelty search this "protective" bloat would actually be *maladaptive*. That is, bloat buffers *against* behavioral change, while in the search for novelty, behavioral change is rewarded. Tentative evidence for this hypothesis is found in Schmidt and Lipson [23], where a coevolutionary GP algorithm (which also changes what is rewarded as search progresses) also evolved smaller programs when compared to GP with a static fitness function.

Though giving up the false comfort found in pursuing the objective goes against common assumptions about what makes search work, novelty search demonstrates that sometimes abandoning the objective can paradoxically lead to it. The successful application of novelty search to GP, following successes in neuroevolution [13, 19, 21], begins to establish the generality of the search technique itself and to suggest the limitations of the currently ubiquitous objective-based paradigm.

7. CONCLUSIONS

This paper introduced novelty search, an established method from neuroevolution, to genetic programming. Motivated by the problem of deceptive gradients in objective-based search, novelty search ignores the objective and instead searches only for individuals with novel behaviors. Counterintuitively, experiments in a deceptive maze navigation task and the artificial ant task showed that novelty search can significantly outperform objective-based search and evolve smaller programs. Thus novelty search is a new tool for solving difficult genetic programming problems.

8. ACKNOWLEDGEMENTS

This research was supported by DARPA under grant HR0011-09-1-0045 (Computer Science Study Group Phase II).

References

[1] Wolfgang Banzhaf, Frank D. Francone, and Peter Nordin. The effect of extensive use of the mutation operator on generalization in genetic programming using sparse data sets. In *PPSN IV: Proceedings of the*

4th International Conference on Parallel Problem Solving from Nature, pages 300–309, London, UK, 1996. Springer-Verlag.

[2] Edmund K. Burke, Steven Gustafson, Graham Kendall, and Natalio Krasnogor. Is increased diversity in genetic programming beneficial? An analysis of lineage selection. In *Congress on Evolutionary Computation*, pages 1398–1405. IEEE Press, 2003.

[3] Sean B. Carroll. Chance and necessity: the evolution of morphological complexity and diversity. *Nature*, 409 (6823):1102–1109, February 2001.

[4] Steffen Christensen and Franz Oppacher. Solving the artificial ant on the santa fe trail problem in 20,696 fitness evaluations. In *GECCO '07: Proceedings of the 9th annual conference on Genetic and evolutionary computation*, pages 1574–1579, New York, NY, USA, 2007. ACM.

[5] Sevan Ficici and Jordan B. Pollack. Challenges in coevolutionary learning: Arms-race dynamics, open-endedness, and mediocre stable states. In *Proceedings of the Sixth International Conference on Artificial Life*, pages 238–247. MIT Press, 1998.

[6] Chris Gathercole and Peter Ross. An adverse interaction between crossover and restricted tree depth in genetic programming. In *GECCO '96: Proceedings of the First Annual Conference on Genetic Programming*, pages 291–296, Cambridge, MA, USA, 1996. MIT Press.

[7] David E. Goldberg. Simple genetic algorithms and the minimal deceptive problem. In L. D. Davis, editor, *Genetic Algorithms and Simulated Annealing, Research Notes in Artificial Intelligence*. Morgan Kaufmann, 1987.

[8] John H. Holland. *Adaptation in Natural and Artificial Systems: An Introductory Analysis with Applications to Biology, Control and Artificial Intelligence*. University of Michigan Press, Ann Arbor, MI, 1975.

[9] J. R. Koza. *Genetic Programming: On the Programming of Computers by Means of Natural Selection*. MIT Press, Cambridge, MA, 1992.

[10] John R. Koza. *Genetic Programming II: Automatic Discovery of Reusable Programs*. MIT Press, Cambridge Massachusetts, May 1994.

[11] W. B. Langdon and R. Poli. Fitness causes bloat. In *Soft Computing in Engineering Design and Manufacturing*, pages 23–27. Springer-Verlag, 1997.

[12] W. B. Langdon and R. Poli. Why ants are hard. In *Genetic Programming 1998: Proceedings of the Third Annual Conference*, pages 193–201, University of Wisconsin, Madison, Wisconsin, USA, 22-25 1998. Morgan Kaufmann.

[13] Joel Lehman and Kenneth O. Stanley. Exploiting open-endedness to solve problems through the search for novelty. In *Proceedings of the Eleventh International Conference on Artificial Life (ALIFE XI)*, Cambridge, MA, 2008. MIT Press.

[14] Sean Luke. Genetic programming produced competitive soccer softbot teams for RoboCup97. In *Genetic Programming 1998: Proceedings of the Third Annual Conference*, pages 214–222, University of Wisconsin, Madison, Wisconsin, USA, 1998. Morgan Kaufmann.

[15] Dylan Mawhinney and Vic Ciesielski. Preventing early convergence in genetic programming by replacing similar programs. In *Proceedings of the 2002 Congress on Evolutionary Computation*, pages 67–72. IEEE, 2000.

[16] Thomas Miconi. Evolution and complexity: The double-edged sword. *Artificial Life: Special Issue on the Evolution of Complexity*, 2007.

[17] John Stuart Mill. *A System of Logic, Ratiocinative and Inductive*. 1846.

[18] Tom M. Mitchell. *Machine Learning*. McGraw-Hill, New York, 1997.

[19] Jean-Baptiste Mouret. Novelty-based multiobjectivization. In *Proceedings of the Workshop on Exploring New Horizons in Evolutionary Design of Robots,2009 IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2009.

[20] Esteban Ricalde and Katya Rodríguez Vázquez. A GP neutral function for the artificial ant problem. In *GECCO '07: Proceedings of the 9th conference companion on Genetic and evolutionary computation*, pages 2565–2571, New York, NY, USA, 2007. ACM.

[21] Sebastian Risi, Sandy D. Vanderbleek, Charles E. Hughes, and Kenneth O. Stanley. How novelty search escapes the deceptive trap of learning to learn. In *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO-2009)*, New York, NY, 2009. ACM.

[22] Conor Ryan. Pygmies and civil servants. In Kenneth E. Kinnear, Jr., editor, *Advances in Genetic Programming*, chapter 11, pages 243–263. MIT Press, 1994.

[23] M.D. Schmidt and H. Lipson. Coevolution of fitness predictors. *Evolutionary Computation, IEEE Transactions on*, 12(6):736–749, Dec. 2008.

[24] Karl Sigmund. *Games of Life: Explorations in Ecology, Evolution and Behaviour*. Oxford University Press, Inc., New York, NY, USA, 1993.

[25] Andrea Soltoggio and Ben Jones. Novelty of behaviour as a basis for the neuro-evolution of operant reward learning. In *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO-2009)*, New York, NY, 2009. ACM.

[26] Terence Soule. *Code Growth in Genetic Programming*. PhD thesis, University of Idaho, Moscow, Idaho, USA, 15 May 1998.

[27] Russell Standish. Open-ended artificial evolution. *International Journal of Computational Intelligence and Applications*, 3(167), 2003.

[28] Wei Yan and Christopher D. Clack. Behavioural GP diversity for adaptive stock selection. In *GECCO '09: Proceedings of the 11th Annual conference on Genetic and evolutionary computation*, pages 1641–1648, New York, NY, USA, 2009. ACM.

[29] N. Zaera, D. Cliff, and J. Bruten. (Not) evolving collective behaviours in synthetic fish. In *From Animals to Animats 4: Proceedings of the Fourth International Conference on Simulation of Adaptive Behavior*. MIT Press Bradford Books., 1996.

[30] Douglas Zongker and Bill Punch. lilgp 1.01 user's manual. Technical report, Michigan State University, USA, 26 March 1996.