NOVELTY-BASED FITNESS MEASURES IN GENETIC PROGRAMMING

by

John A. Doucette

Submitted in partial fulfillment of the
requirements for the degree of
Bachelor of Computer Science, Honours

at

Dalhousie University
Halifax, Nova Scotia
April 2010

DALHOUSIE UNIVERSITY

FACULTY OF COMPUTER SCIENCE

The undersigned hereby certify that they have read and recommend to the Faculty of Computer Science for acceptance a thesis entitled "NOVELTY-BASED FITNESS MEASURES IN GENETIC PROGRAMMING " by John A. Doucette in partial fulfillment of the requirements for the degree of Bachelor of Computer Science, Honours.

Dated: April 14, 2010

Supervisor: ______________________
Dr. Malcolm I. Heywood

Reader: ______________________
Dr. Andrew R. McIntyre

Table of Contents

List of Tables

List of Figures

List of Algorithms

Abstract

The utility of the class of non-qualitative fitness measures known as “novelty-based” measures is considered with respect to Genetic Programming (GP). Previously used benchmarks from GP and from other work on novelty-based search heuristics are used. The resulting data suggest that novelty-based measures may be useful when solution robustness is essential, but that overall, they may not be as powerful in the context of GP as previous work suggests they are in other areas of machine learning. The introduction of factors found in other machine learning models for which novelty-based measures preformed well previously did not improve the performance of GP when novelty-based measures were used, or produced inconclusive results, depending on the benchmark problem used.

Acknowledgements

I wish to acknowledge and thank the National Science and Engineering Research Council for providing funding for much of my undergraduate research.

Thanks to Dr. Malcolm Heywood for his guidance and instruction over the past three years, and for accepting me as a student so early in my academic studies.

Thanks to my reader, Dr. Andy McIntyre, and the chair of my thesis committee, Dr. Thomas Trappenberg.

Finally, thanks to anyone about to take the time to read this rather long and esoteric honours thesis. You'll never get these minutes back.

Chapter 1

Introduction and Background

This thesis examines the relationship between Genetic Programming (GP) and Novelty-based fitness functions. Although both topics have been explored extensively and there has been some preliminary work examining them in consort, this study concerns previously unexplored questions regarding their interaction. The effects of several novelty-based fitness measures on GP performance are examined in two simple agent control problems, to determine the effect of novelty on model generalization and GP's ability to solve problems efficiently. The interaction between teaming approaches and novelty-based measures are also considered. The primary conclusion is that novelty-based measures may not be useful when a good qualitative measure is available, but may still be of interest under certain circumstances.

1.1 Artificial Evolution

Genetic Programming is a method of using evolution to solve problems presented by John Koza in 1992 [11]. While other methods of GP have been created, Koza's remains among the most widely used, and so is often termed "canonical GP". GP works by using the properties of evolution by artificial selection to automatically generate an appropriate computer program to solve a problem of interest. To accomplish this, a set of allowable instructions is defined, and a population of computer programs is generated from random combinations of those instructions. The programs are evaluated according to a "fitness function" which describes how good a candidate solution is according to an evaluation criteria, often how closely the candidate solution matches a desired input/output mapping. Candidate solutions which receive higher values of the fitness function are allowed to "reproduce", typically by making stochastically modified clones of themselves or by recombination of their instructions with a second candidate solution. By iteratively applying this procedure and by allowing individuals which more closely approximate the desired solution to produce more offspring in the following generation, high quality solutions to many problems may be obtained. [11]

1.2 GP As Search

One possible way to view GP is as a search through the space of all possible computer programs that can be built using a specified set of functions. In this metaphor the fitness function takes on the role of a search heuristic, guiding the search toward areas that are (hopefully) closer to an ideal solution to the problem. There are many heuristic measures available for search, and likewise, many possible fitness functions. Commonly used measures like error minimization, sensitivity/specificity, euclidian distance, and the like are similar insofar as they all define the quality of a candidate solution relative to its distance from an ideal solution. The term “qualitative measures” will be used as a catchall phrase for these types of heuristics throughout this document.

Qualitative measures often make excellent, simple, and intuitive fitness functions for GP. For example, in classification tasks, an ideal solution will correctly map all input data to the correct class labels. A candidate solution’s quality may be expressed as the number of inputs for which the candidate’s class label differs from the class label of an ideal solution, or any number of similar measures. Think of this as an artificial selection pressure on the population of candidate solutions, much like an artificial selection pressure in real-world evolution.

A second class of search heuristics are known as ”novelty based measures”. Novelty based measures differ from qualitative measures in that a candidate solution’s fitness is not defined relative to an idealized solution, but relative to all solutions previously encountered by the search. For example, in classification a novelty-based solution might compare the outputs of a candidate solution to those of all previously encountered candidates. The fitness function would reward outputs which were unique (i.e. different from those of previously seen outputs), but not necessarily correct. Novelty-based measures are excellent ways of rapidly exploring a search space, but tend to have difficulty exploiting good behaviors they find in the process.

While novelty based functions have been used extensively in some domains (e.g.

Genetic Algorithms) they have only recently begun to be applied to genetic programming(GP), primarily in real-valued control systems evolved using various methods of evolutionary computation [15, 21]. The primary goal of this work is to examine the effects of applying novelty based functions to GP, and to determine what effects they have on the properties of GP solutions. In particular, the effects of novelty on important properties like generalization capabilities remain unknown, as do the effects of novelty-based measures interacting with the unique properties of non-cannonical GP formulations, such as speciation or teaming.

1.3 Novelty and No Free Lunch

In recent years, numerous works utilizing novelty have appeared in the GP literature. While several implicit rewards for novelty (e.g. [17, 19]) have been examined, only recently have explicit rewards come into play (e.g. [15, 8, 25, 21]) . While implicit rewards were often combined with explicit qualitative measures, these new explicitly novel fitness schemes aim to have no qualitative information in them at all. In spite of this, they often solve complex problems as (or more) efficiently than methods using qualitative fitness functions. This seems highly counterintuitive. By definition, an explicitly novel fitness scheme offers no reward for producing a good solution to a problem, only for producing new and different solution candidates, yet novelty based measures often seem to find ideal solutions more quickly than qualitative measures.

There is a theoretical basis for explaining why novelty based measures can perform well, known as the "No Free Lunch" Theorem [28]. No Free Lunch is often explained in terms of a restaurant metaphor. In the metaphor, a set of identical restaurants each produces a menu offering the same set of food choices. Each restaurant's menu offers the same set of prices for items, but associates each price with a different item. Thus, while restaurant X might offer a cheeseburger for \$1 and a salad for \$5, restaurant Y might charge \$5 and \$1 for the same items. A consumer who picks a random restaurant and a random item when going to lunch will expect to pay an average amount for lunch (i.e. the expected value of the price list). In contrast, a consumer with discriminating taste, a vegetarian say, will pay a much higher price at some restaurants than at others, but over all possible arrangements of items and prices will pay the same average price as our random consumer (again, the expected value). Similarly, averaged over the set of all possible search spaces (and thus, all possible problems investigable with GP), any strategy for picking an area of the space to examine next cannot outperform a completely random method. This is because for any given search space in which a strategy's choice to examine next exceeds that of a random strategy in quality, a *reciprocal* search space exists, in principle, in which exactly the opposite is true. Absurdly enough, this means that the *average* performance of hill climbing and hill descending are, in theory, over all possible search spaces, identical.

This goes against conventional wisdom. If hill climbing were truly no better than hill descending in practice this fact would have been noticed. Crucially, No Free Lunch holds only over the set of all possible permutations of a problem space. If only search spaces that have specific properties are considered, then relevant heuristics can still out perform random guessing. For example, consider only a finite set of gaussian search spaces. Here, hill climbing will certainly defeat hill descending. This is the vital ingredient in the theoretical justification of novelty based searching. Novelty should not be expected to outperform quality in all problems spaces, but rather in those spaces in which qualitative measures may mislead the search. That is, problems in which an exploitative strategy will be frequently deceived. An example problem is depicted in figure 1.1.

1.4 Deceptive Problems

The class of search spaces that novelty based fitness measures might perform well in is "deceptive" search spaces. A deceptive search space is one in which a candidate solution's distance from an idealized solution in the search space is not correlated with its distance from an idealized solution in "behavioral" or "phenotypic" space. As users of GP typically have no idea what a sequence of instructions will produce the desired result, a qualitative fitness measure working in such a domain will invariably reward individuals which are close to an ideal behavior, but very far from the code necessary to produce such a behavior. For example, maze navigation is a deceptive problem domain. A good solution is one which exits the maze, but any measure of distance other than tracing a path from a candidate solution to the exit (and thus making the problem absurdly easy) is likely to be deceptive. A navigator which is

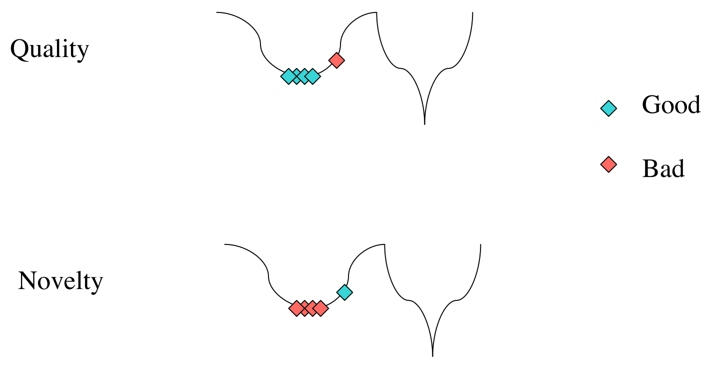

Figure 1.1: Example function minimization problem where novelty-based fitness measures could be useful. The upper graph shows how a qualitative measure rewards solutions clustered around local minima. In contrast, the novelty-based measure depicted in the lower graph offers a higher reward to a solution closer to escaping the local optima.

close to the maze exit in euclidian space may be extremely distant from an ideal solution in the search space of program code. Novelty based fitness measures may be useful on such a problem because they could reward a navigator which reached previously unseen parts of the maze. While a typical quality-based measure would cause navigators to cluster around dead-ends near the exit, a novelty based solution would encourage the navigators to spread out and discover new parts of the maze. This behavior might allow the novelty based measure to locate the maze exit quickly while the quality based search remains stuck near a local optima [15]. Some deceptive problems are shown in figure 1.2.

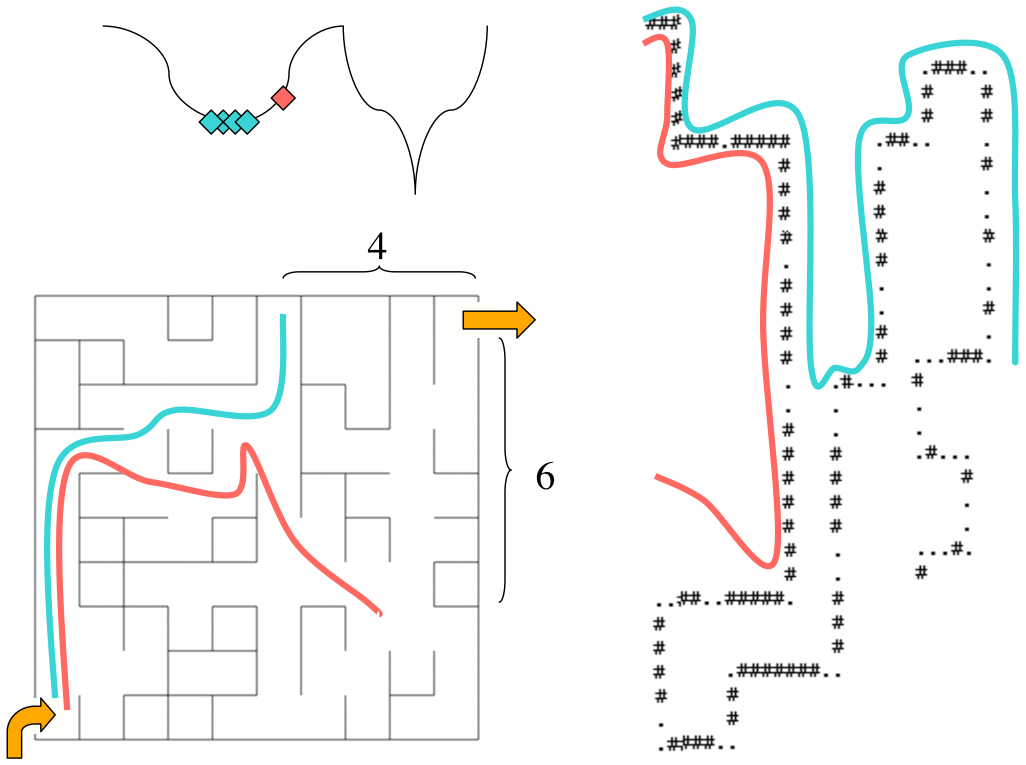

Figure 1.2: Some visual representations of deceptive problems. Red solutions are outperformed by blue solutions according to a qualitative metric (food eaten, distance from global minima, euclidian distance from maze exit), but are actually closer to solving the problem of interest.

Chapter 2

Novelty Measures

As described in the introduction, novelty-based fitness measures are those which explicitly reward finding locations in a search space that differ from previously explored locations, while qualitative fitness measures reward behavior which approaches a prespecified ideal. While both methods require a problem dependent distance metric (to compute the novelty of pairs of candidate solutions, and the distance from the ideal solution respectively), the novelty based methods also require some method of determining what constitutes a "novel" solution. The distinctions between these methods is analogous to those between different qualitative measures (e.g. accuracy, area under the ROC curve, F-measure, etc.) in that all methods may use the same problem specific measure of performance, but offer different interpretations of what a particular distance means for an individual's fitness. Several methods of comparison were examined, and are presented below.

2.1 Euclidian Phenotypic Distance

Perhaps the simplist measure of novelty is the Euclidian Phenotypic Distance. Given two candidate solutions, this distance measure returns the euclidian distance between the solutions' behaviors. It is essentially identical to the commonly used qualitative measure, but compares arbitrary individuals, rather than comparing all candidate solutions to an idealized solution. For example, in maze navigation, an individual's phenotype could be represented by the position they have reached after a fixed number of time steps. In a traditional qualitative fitness measure such an individual would be rewarded based on the distance from their final position to the maze exit (i.e. the final position of an idealized solution). Under the equlivient novelty-based measure, the individual is rewarded based on their distance from other individuals of interest. Typically, candidate solutions will be compared to all other individuals currently in the GP population, or to some select subset of individuals.

Algorithm 1: delta(i,j) : Returns the Euclidian distance between two phenotypes, where *maxdist* is the maximum possible difference between two phenotypes.

```
    Input: Phenotypic Representations $i$ and $j$
    Output: The normalized distance from $i$ to $j$
1   $dist = \frac{\sqrt{(i.x-j.x)^2+(i.y-j.y)^2}}{maxdist}$
    Result: $dist$
```

Algorithm 2: Euclidian Phenotypic Distance(S,I) : Computes EPD novelty measure of I w.r.t. S

```
    Input: Set of Phenotypic Representations S, Target Phenotypic
           Representations I
    Output: Euclidian Phenotypic Distance of I w.r.t. S
1   $min$=1;
2   foreach Individual $j$ in $S$ do
3       if $delta(j,I) < min$ then
4           $min$=delta($j$,$I$);
5       end
6   end
    Result: $min$
```

2.2 Phenotypic Hamming Distance

One drawback of Euclidian Phenotypic Distance is that it is “consequentialist”. The only factor that matters to an individual’s fitness is its endpoint, but the path an individual takes can also be important. Phenotypic Hamming Distance addresses this concern by allowing for comparison of many datapoints. An individual is represented by a list of points, and the distance between two individuals is simply their hamming distance[9]. For example, in maze navigation, a candidate solution’s representation under phenotypic hamming distance might be a list of the junctions in the maze which the navigator has passed. Two individuals are said to have similar behavior if many of the junctions in their lists occur at the same positions, and novel behavior if the opposite is true. Algorithm 3 describes this process in greater detail.

Algorithm 3: Phenotypic Hamming Distance(S,I) : Computes the PHD novelty measure of I w.r.t. S

```
Input: Set of Phenotypic Representations S, Target Phenotypic
       Representations I
Output: Phenotypic Hamming Distance of I w.r.t. S
min=1;
foreach Individual j in S do
    sum=0;
    foreach Allele a in j do
        if delta(j,I)$\neq$ 0 then
            $sum = sum + 1$;
        end
    end
    $total = I.alleles.size()$;
    if sum/total ¡ min then
        min=total;
    end
end
Result: min
```

Obviously, given such a vector representation of an individual’s behavior, hamming distance is just one of many possible distance measures. In this research, hamming distance was utilized primarily for efficiency reasons. Other research examining this issue [8] suggests small gains may be achieved by using certain compression algorithms for comparison instead of hamming distance. Biological and linguistic sequence alignment algorithms might also be applicable. However, both of these alternatives are highly computationally intensive, which precludes their usage in a large-scale study with the available resources. For this reason, neither of these alternative measures were considered here.

2.3 Archetype Archiving

While either distance measure may be used alone, as in [15, 8], they may also be utilized in conjunction with an "Archetype Archive", [6]. Archetype Archiving is similar to, though not the same as, the "Sentinals" concept advanced by Morrison [20], in that it attempts to foster diversity within a population via comparisons with an existing and diverse set of candidate solutions. Rather than comparing individuals to other members of the current population to determine novelty, an archetype archive stores archetypical individuals and uses them as a comparison population to determine novelty (parameter S in algorithms 3 and 2). Thus, individuals which rediscover behaviors which were seen several generations ago, but have since vanished from the population, will not be rewarded. This may be a drawback if such behaviors are necessary intermediate steps toward a correct solution, but archetype archiving should also prevent the population from becoming stuck in cycles of useless behaviors. At the end of training, the archive will contain representatives of a broad range of behaviors, some of which may be useful for solving the problem at hand.

Archetype Archives necessitate a method of determining what constitutes an archetype. Archetypes should be candidate solutions which can act as a placeholder for a broad range of potential solutions which exhibit similar behavior. Two possible methods of selecting archetypes are considered. In the first method, known as "infinite" or "fixed threshold" archiving, an archive population of variable size is created, and assigned an entrance threshold Δ_{min}. During the evaluation of the primary GP population, individuals which differ by at least Δ_{min} (according to the present novelty-based measure) from all members of the archetype archive are added to the archive as well. While this method has some advantages in its simplicity, the selection of Δ_{min} forces a tradeoff between computational requirements and the minimum behavioral differences required for admittance into the archive. As shown in algorithm 4 , the number of comparisons needed to perform each insertion into the archive grows linearly with the number of archive members. Thus, infinite archiving is better suited to problems which utilize a scaler measure of fitness (and thus, euclidian phenotypic distance), than those which utilize a vector measure (and thus, phenotypic hamming distance), since the cost per comparison tends to be dramatically lower in the former case. This reward system is very similar to those used in [15, 25], but uses only the archive when computing fitness, and not the current population as well.

The other method of archetype selection considered is “finite” or “variable threshold” archiving. In finite archiving, an archive of fixed size is created. When a new individual is added to this archive, pairwise comparisons are done between all archive members. The pair with the lowest novelty value (i.e. the most similar behaviors) is stored for reference. The novelty value achieved by this pair is used as the Δ_{min} value for the archive until the next time an individual is added to the archive. At that time, one randomly selected member of this “lowest novelty pair” is replaced by the candidate solution under examination. The advantage of this method is that provides a natural gradient for the GP population, increasing the minimum standard of novelty over time. The drawback is that a number of comparisons quadratic in the size of the archive needs to be done for each archive insertion. Thus, the tradeoff between computational workload and solution quality is even more pronounced than in infinite archiving. Furthermore, the archive size parameter appears to be somewhat more sensitive to modification than the Δ_{min} parameter for infinite archiving is. Finite archiving is presented in greater detail in algorithm 5.

Algorithm 4: Infinite Archetype Archive Admittance Criteria(A,I,N): Determines whether an individual meets the admittance requirements for an archive.

```
Input: Infinite Archetype Archive A, Target Phenotypic Representatoin I,
       Novelty Based Measurement Function N(S,I)
Output: true if I is to be added to A, false otherwise
1 $result$=$false$;
2 if $N(A, I) \geq \Delta_{min}$ then
3     $result$=$true$;
4 end
Result: $result$
```

Algorithm 5: Finite Archetype Archive Admittance Criteria(A,I,N) : Determines whether an individual meets the admittance requirements for an archive.

```
Input: Finite Archetype Archive A, Target Phenotypic Representation I,
       Novelty Based Measurement Function N(S,I)
Output: true if I is to be added to A, replacing the worst individual, false
        otherwise
1 $result$=$false$;
2 $worst\_indiv$=$find\_worst(A)$;
3 if $N(A, I) \geq N(A, worst\_indiv$ then
4     $result$=$true$;
5 end
Result: $result$
```

Chapter 3

Experiment 1

The first experiment consisted of evolving solutions to a classic deceptive problem using both qualitative and novelty-based fitness measures. The primary motive for this initial experiment was investigating whether earlier work using novelty to evolve artificial neural networks [15] could be directly extended to canonical GP [11]. Earlier work on artificial neural networks did not examine the effects on novelty on solution brittleness, and the data from experiment 1 was thus used to investigate this question as well. The results for experiment 1 were originally presented at EuroGP 2010 and are discussed in greater detail in the proceedings of that conference [6].

3.1 The Santa Fe Trail

The Santa Fe Trail is a classic GP benchmark, in which a control program for a simulated ant is evolved. Traditionally, the ant is placed on a 32x32 toroidal grid populated with 89 pieces of ”food”. Food squares are arranged to form a rough trail which the ant controller will ideally learn to follow. Ants on the Santa Fe Trail are allowed to view only the contents of the grid square immediately ahead of them, and use a built-in function to detect whether it is empty or contains food. At each time step, the ant controller may either move the ant directly forward, or rotate in place 90° left or right. An ant is said to eat a piece of food, removing it from the grid, by walking over it. The objective is to evolve an ant controlling program which can eat all the largest amount of food on the trail in a fixed number of time steps, and thus the traditionally used qualitative fitness measure is the number of food pieces eaten by the ant. The problem was first used as a GP benchmark by Koza [11], but continues to be widely used and studied [14, 22, 27, 7, 12, 23].

The Santa Fe Trail is an excellent benchmark for the two questions experiment 1 aims to answer. First, the trail is an example of a deceptive problem under the traditionally used fitness measure. This is discussed at some length in [14], but is essentially due the tendency of evolved ants to deviate from the trail in order to consume nearby pieces of food. Since control programs exhibiting this behavior consume the trail prior to deviating, there will typically not be enough information to systematically return to the trail after deviating from it. Thus control programs which deviate from the trail represent a local maxima. They may eat quite large amounts of food compared with ants which follow the trail with greater devotion, but there is no easy way to move from such a solution to one which consumes the *entire* trail. As there are many locations on the trail where this behavior is possible, the trail has many local maxima and is deceptive. As discussed above, novelty-based fitness functions are thought to be better suited to deceptive problems. This property makes the Santa Fe Trail ideal for determining the effects of novelty-based fitness measures in canonical GP. The second useful property of the trail is an established procedure for testing generalization capabilities [13, 12], which allow for the generation of "Santa Fe-Like" trails. These trails share the properties which define the Santa Fe Trail, including maximum length of gaps in the trail, types of trail segments and the probability distribution of trail segments. The ability to produce other trails allows the models evolved on the Santa Fe Trail to be tested for robustness to variation in their environment, an issue which can be non-trivial to resolve in other deceptive problem domains. Combined, these features make the trail an ideal initial problem to determine the feasibility of novelty-based fitness measures in canonical GP, and the effects of novelty on the robustness of solutions.

3.2 Phenotype Definitions

In order to measure novelty in GP it is necessary to have a phenotypic representation of GP individuals. This is because genotypic representations for GP (that is, the source code of an evolved program) may differ radically from one another and yet produce nearly identical results. For example, an ant program

$(prog2(prog2(left, left), prog2(left, left)), forward)$

is genetically dissimilar to

$(forward, prog3(prog3(right, right, right), prog3(right, right, right), prog2(right, right)))$

but in fact produces essentially the same behavior. After executing the code, both ants will be one square ahead and facing in the same direction they started in after rotating 360° left and 720° right respectively. Thus, a novelty measure on genotypic representation rewards interesting (or needlessly complex) source code rather than the desired novel *behaviors*.

The intuitive phenotypic representation of a control program in the Santa Fe Trail is simply the sequence of actions performed by the ant when the program is run. This would entail storing all 400-600 actions taken by the ant in the course of a typical navigation of the trail. While simple, the excessive computational cost incurred by the novelty based measures discussed in Section 2 make this representation undesirable. The method also suffers the same problems as a genotypic representation, in that the mapping from phenotypes of this form to behavioral effects is not injective. Thus, an alternative phenotypic representation is considered. Control programs are represented as vectors of length 89 containing the order in which the ant ate various pieces of food, or a null value for food pieces not eaten. When an ant consumes its nth piece of food during evaluation the nth cell of the vector is set to the location of the food on the grid. This allows phenotypes to represent the most important information in the problem relatively concisely, which allows much more extensive experimentation. The representation also creates an injective mapping from phenotypes to behavior, meaning two ants with different phenotypes will differ in at least *some* respect. Figure 3.1 shows how similar phenotypes could be constructed for a small grid.

3.3 Novelty Fitness Measures

Experiment 1 compared two possible novelty-based fitness measures with the classically used qualitative measure for the Santa Fe Trail. The qualitative measure was defined as the percentage of food an ant consumes when directed by a particular control program. Both novelty-based fitness measures used phenotypic hamming distance to determine the relative novelty of two individuals, as discussed in algorithm 3. This process is presented diagrammatically in figure 3.2. Although phenotypic hamming distance has some drawbacks under this representation (most notably that identical behaviors offset from on another by a single position will receive large novelty rewards), the alternative techniques were of considerable computational expense (e.g. sequence alignment), or suffered from other comparable drawbacks.

Novelty method 1 used the finite or fixed archiving scheme from algorithm 5. The

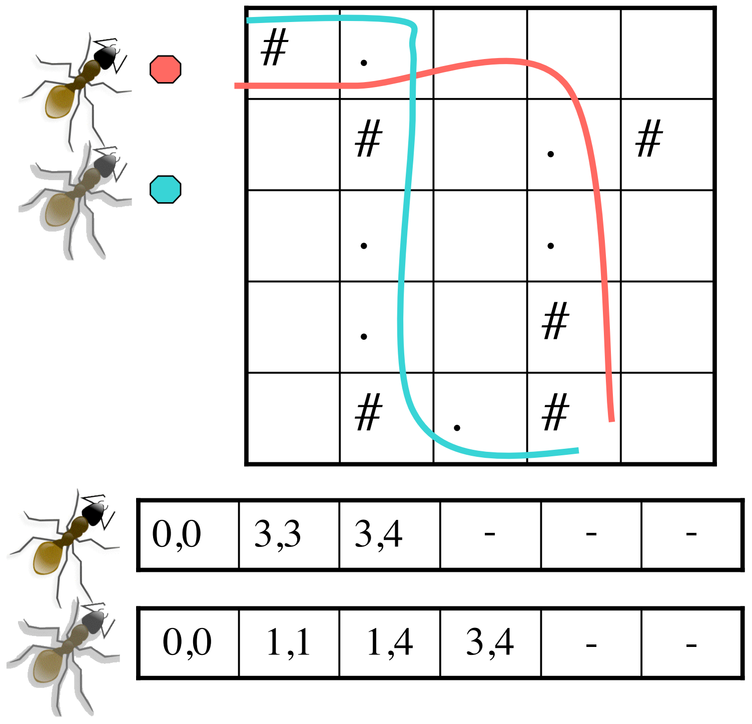

Figure 3.1: Visual Representation of Phenotype Construction for the Santa Fe Trail. An ant's path through the grid is represented as a vector of the location of food pieces eaten.

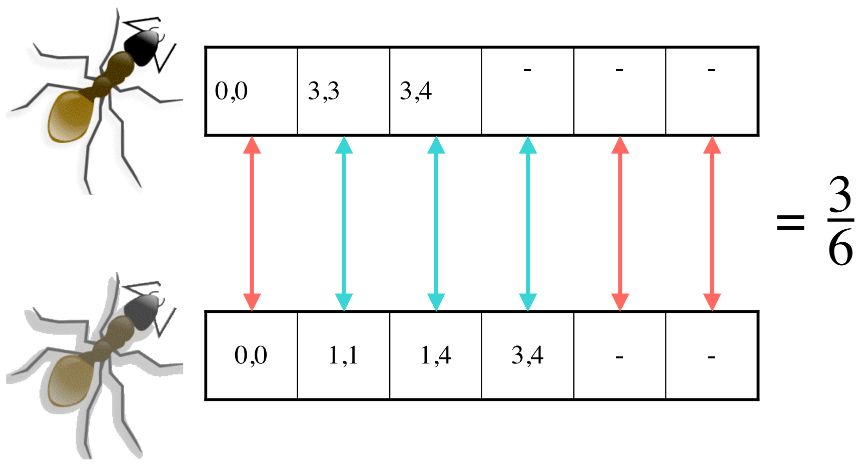

Figure 3.2: Visual Representation of Phenotype Comparison for the Santa Fe Trail. Red arrows denote areas of similarity between the phenotypes, while blue arrows denote differences. The normalized sum of the differences is used as a measure of novelty.

archive size was selected by trying 10 different values over the interval (5, 200) on a single random seed, and using the value which produced the best results. An archive size of 100 was used for all subsequent data gathering in experiment 1. Novelty method 2 used the infinite archiving scheme from algorithm 4. The Δ_{min} parameter was selected using the same procedure as the archive size for method 1, but over the interval (5, 40). A value of 5 was used in subsequent work. A third method was also considered, using the average of the finite archiving scheme from method one and the traditionally used qualitative measure to produce a "combination" fitness measure. Finally, the traditional qualitative measure was also benchmarked for comparison purposes.

3.4 Implementation and Experiment Parameters

All three new novelty evaluation methods were implemented in the lil-gp framework [29], and benchmarked along with the included version of the Santa Fe Trail, which uses the traditional qualitative measure. The parameters used during benchmarking runs were those provided with lil-gp. The only exception was a change from crossover to mutation in the reproduction operators, as suggested in [14]. The full list of parameters is presented in table 3.1. Experiment 1 consisted of running each method on 500 unique random seeds. During a run, populations of 1000 control programs were allowed to reproduce in 50 generations while being evaluated on the Santa Fe Trail according to one of the four fitness measures under examination. At the end of the run, all members of a population were evaluated according to the traditional qualitative measure, and the best individual was designated the "champion". Champions were evaluated on 100 randomly generated test trails, created using an implementation of Kuscu's Santa Fe-like Trail generation strategy [13, 12]. Some example trails are shown in figure 3.3.

3.5 Results

All results for experiment 1 were analyzed using rigorous quantitative methods. All data was tested for normality using a Jarque-Bera test [2]. Data which were normally distributed were evaluated with one-way ANOVA tests, and post-hoc analysis was

```
.##..####..#
.          #
#   #.##.  .
#   #   ..##
   ..
   #
   #
   .
.##.
```

```
#
#
#
.
..
 #
 #
 #
 ##
  #
  #
  #####
      #
      #                #
      #                .
      ###.             .
         ..##.         #
             ..##.     #
                 ..##...
```

```
    #
    #
    #
    .
#..##
#
.
.
.##.#
    #
    ..
     #
     #
     #
     ### #
         #
  ###    .
  #....##.
```

Figure 3.3: Some “Santa Fe-Like Trails” generated using Kuscu’s strategy. The trails are somewhat smaller than the full Santa Fe Trail, but retain many of the same characteristics.

Table 3.1: The parameters used in benchmarking novelty-based fitness measures on the Santa Fe Trail. For meanings of the parameters, see [11].

Parameter	Value
Terminal Set	Left, Right, Move_Ahead
Function Set	If_Food_Ahead, Prog2, Prog3
Selection Method	Stochastic Elitism
Max Time Steps	400
Max Program Depth	17
Initialization	Ramped half and half, max depth 6
Reproduction Operators	90% Mutation, 10% Reproduction
Population Size	1000
Maximum Generations	50

performed with pairwise student t-tests. Data which were not normally distributed were evaluated with the non-parametric Kruskal-Wallis test, and post-hoc analysis was performed with pairwise Wilcoxon Rank-Sum tests. The term “statistically significant” refers to significance at confidence levels of 95% throughout, and p-values were corrected with the Bonferroni procedure when required for post-hoc analysis. The traditional qualitative measure (% food eaten) was used as a performance measure in all data presented below.

3.5.1 Training Performance

The performance of each run’s champion on the true Santa Fe Trail was tracked to form the “training” data set. The results are summarized in figure 3.4. All four fitness measures had data distributions which were not significantly different from normal distributions. A one-way ANOVA test found a significant difference between the means of the different fitness measures. Student t-tests found significant differences between all pairs of means. The traditional fitness measure outperformed all others on the training data, with an observed mean 7%, 10%, and 13% higher than those for the combination, fixed archive, and infinite archive measures respectively.

3.5.2 Test Performance

Each run’s champion was evaluated on the same set of 100 randomly-generated “Santa Fe-like” trails. The median performance of these 100 trials was used as a measure to compare the champions. The results are summarized in figure 3.5. All four fitness measures produced data consistent with the hypothesis that it was drawn from a normal distribution. A one-way ANOVA test found a significant difference between the means of the median performance of the different fitness measures. Pairwise t-tests indicated that there was no significant difference between the quality and combination measures, but significant differences were found between all other pairs of means. The observed mean of the fixed archive method was 3% lower than the quality and combination measures, and the infinite archiving method’s observed mean was 6% lower.

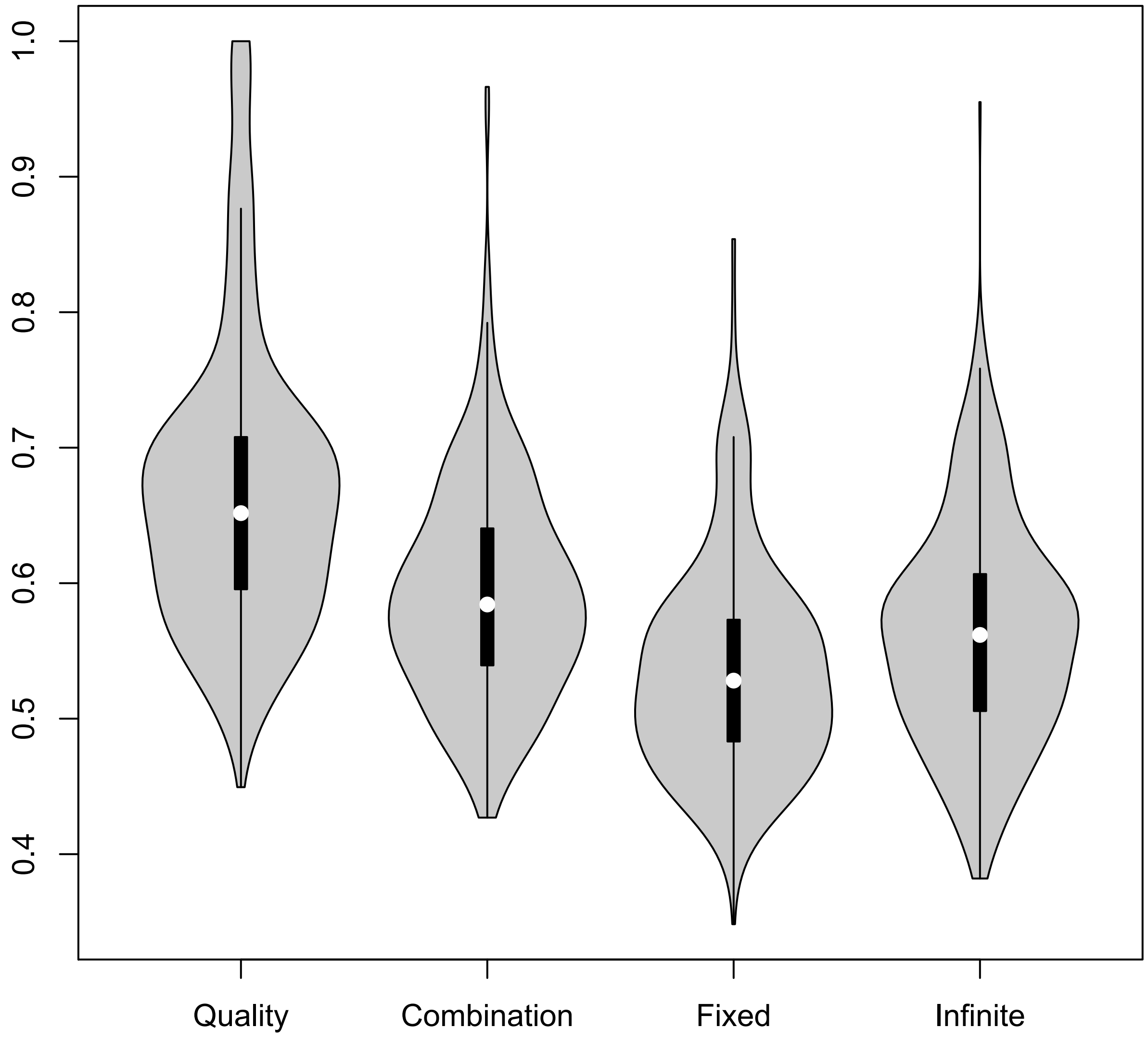

Figure 3.4: Violin plot of each fitness measure's training performance. White dots denote the median, while the edges of black boxes denote the first and third quartiles. The full distribution is presented using a rotated kernel density plot surrounding the boxes [10]. The y-axes shows the ratio of food eaten to total food present. Built using R [24, 1]

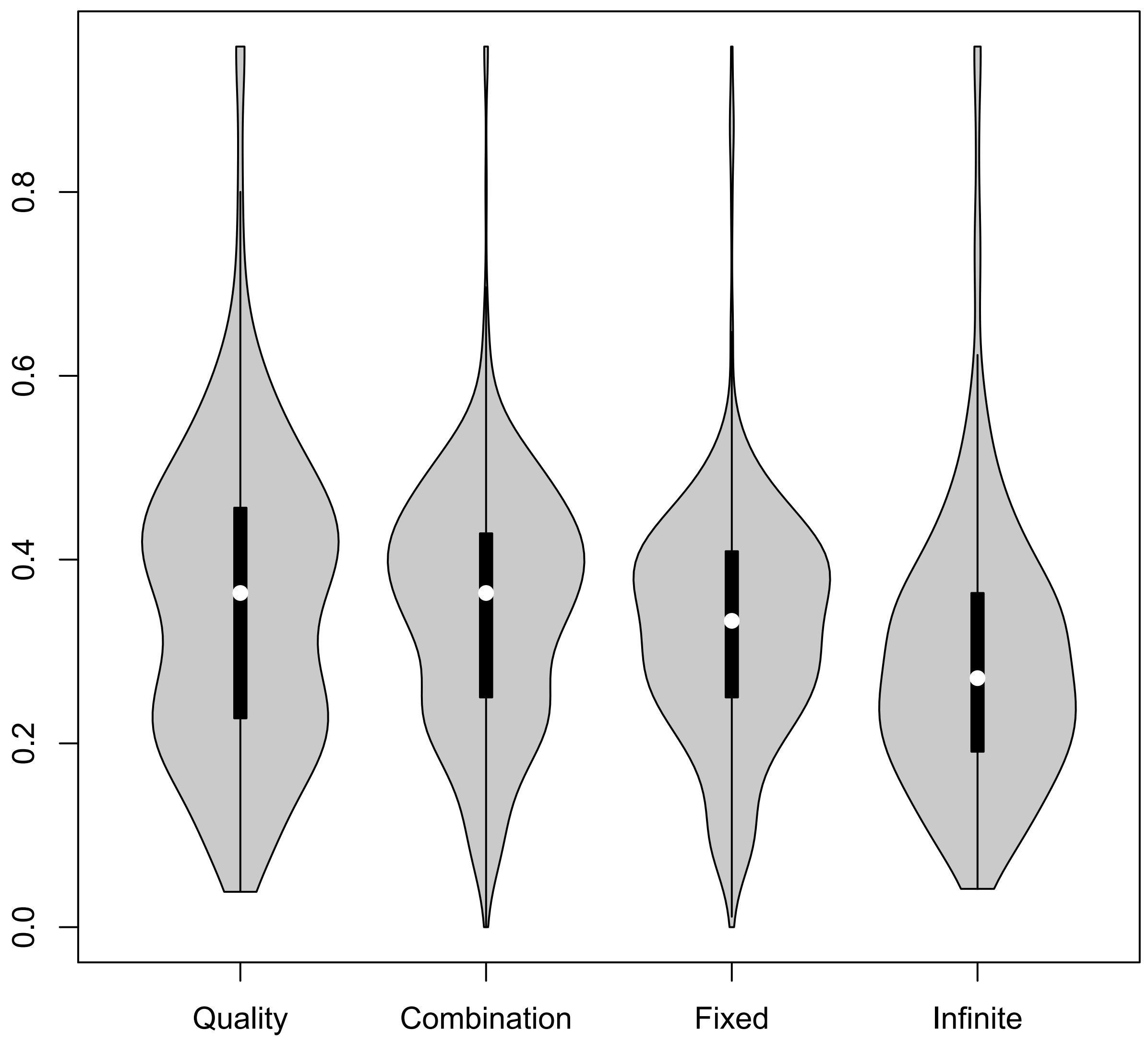

Figure 3.5: Violin plot of each fitness measure's median test performance. See Fig. 3.4 for explanation of the violin plot.

3.5.3 Generalization Performance

The final performance measure considered was paired generalization error, computed as the difference between normalized training and normalized test performance of the same champion across all 100 test trails. Although a certain amount of generalization error is usually expected, a large difference may suggest overfitting of the training data. The results are summarized in figure 3.6. Again, the median performance of champions is used in comparisons between methods. All four methods produced data consistent with the hypothesis that it had been drawn from a normal distribution. A one-way ANOVA test confirmed differences between the means of the various fitness measures. Pairwise t-tests found significant differences between all pairs of means, with observed means of 21%, 24%, 27%, and 31% for the fixed archive, combination method, infinite archive, and qualitative measure respectively.

3.6 Conclusions

The data gathered in experiment 1 provide answers to both questions asked at the start of the chapter. First, the failure of novelty-based measures to match or exceed the performance of the measures incorporating quality is an important result. It appears that the novelty-based measures used previously in other domains [15, 25] may not be directly applicable to canonical GP. This suggests some factor may be present in these other domains, but not in canonical GP, which facilitates the use of novelty based fitness functions. An alternative interpretation of the results suggests that novelty-based measures produce worse *intermediate* solutions than qualitative measures, but might find ideal solutions sooner, since no method tested produced an ideal solution in the training environment. Given the theorized propensity of novelty measures for avoiding local optima, this seems plausible, but further study is needed in either case.

Second, the data suggest that novelty may reduce solution brittleness and overfitting. The generalization error for the combination method in particular is much lower than that for the qualitative method, while retaining statistically identical test performance. The bimodal distribution appearing in the plots of test and generalization error performance for the qualitative measure offer support to this notion,

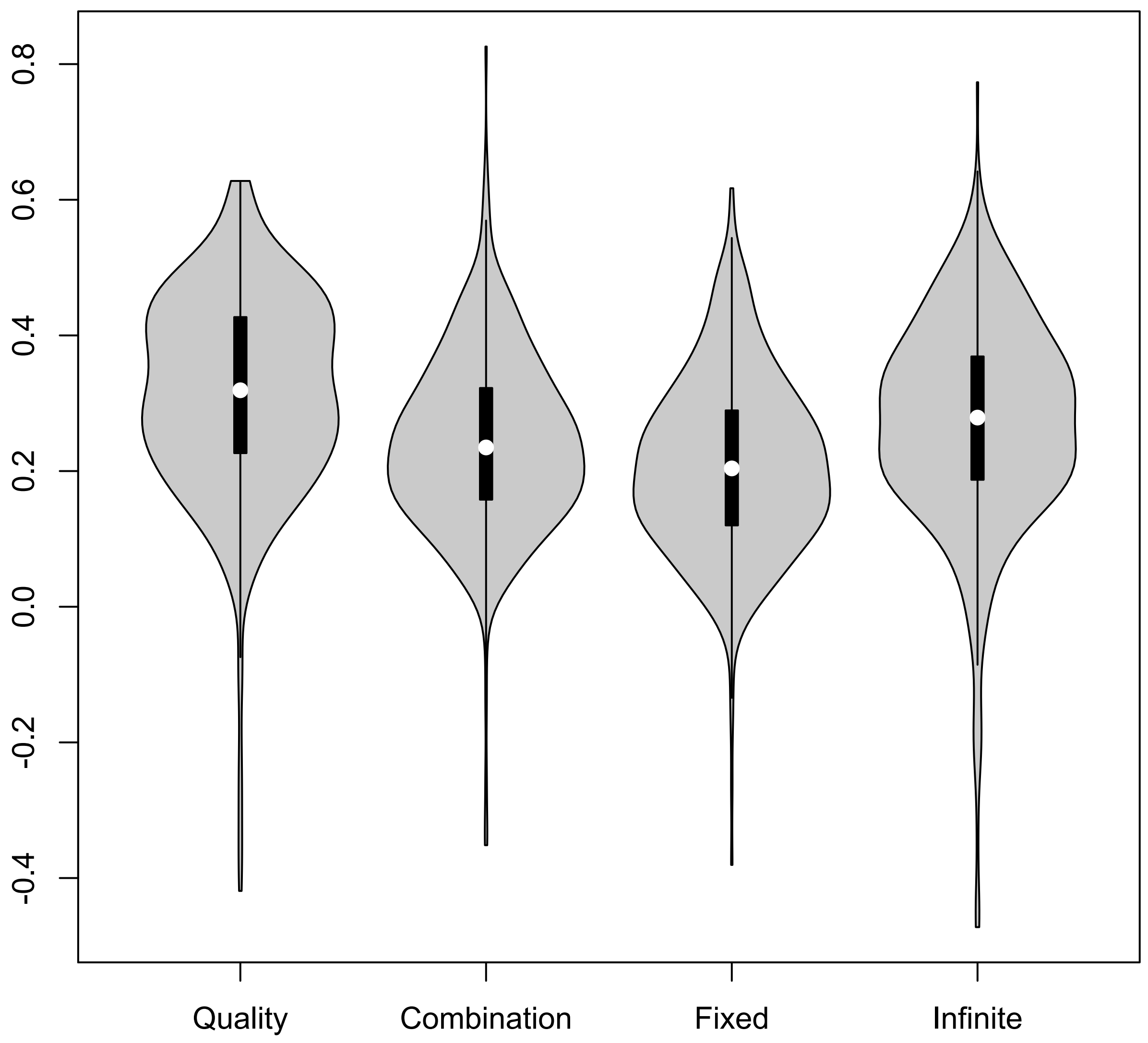

Figure 3.6: Violin plot of each fitness measure's median paired generalization error. See Fig. 3.4 for explanation of the violin plot.

suggesting that a substantial proportion of the individuals created using the qualitative measure may have over-fit the training trail. Thus, novelty-based measures may be considered for use in situations where a good estimate of model performance during training is especially important, such as control of autonomous robots in poorly understood environmental conditions. However, more study is needed before drawing firm conclusions in this area.

Chapter 4

Experiment 2

Experiment 2 was devised to answer some of the questions raised by experiment 1. The primary goal was to test whether automatic problem decomposition was the missing factor in canonical GP's failure in using novelty-based measures. This feature, inspired by the biological property of niching, is found in systems which have previously met with success using novelty-based measures [15]. The secondary goal was to collect data on the interaction of GP and novelty from a larger range of problems. While a variant of the Santa Fe Trail problem was used again, a real-valued control problem based around navigating a robot through a maze, similar to that used in an earlier study [15] was also considered.

4.1 Symbiotic Bid-Based Genetic Programming

Symbiotic Bid-Based Genetic Programming (SBB-GP) is a variant of GP in which solutions are composed of teams of many short programs instead of a single long one, as in canonical GP. These teams of small programs collectively decide which action to take for a given input via a "bidding" procedure. To decide which action to take, an SBB-GP team runs each small program separately on a given input and stores their output or "bid" value. The small program which produces the highest bid value is allowed to perform its assigned action [17]. Thus, there is a symbiotic relationship between teams and team members which is used to drive the evolution of both populations. The bidding behavior also allows automatic problem decomposition, which can potentially produce much simpler solutions than in traditional GP models [5, 4]. An additional feature which differentiates SBB-GP from canonical models is a competitive co-evolutionary interaction with its environment. The subset of environmental test cases on which SBB-GP individuals are evaluated is modified over time in a host-parasite relationship, which helps avoid becoming stuck in local maxima or plateaus of the search space. Figure 4.1 provides a visual representation of the algorithm, while a full description of the SBB-GP algorithm may be found in [17]. Novelty may be effective in conjunction with this model because it shares the niching features found in other algorithms for which novelty has been shown to be effective [15, 25].

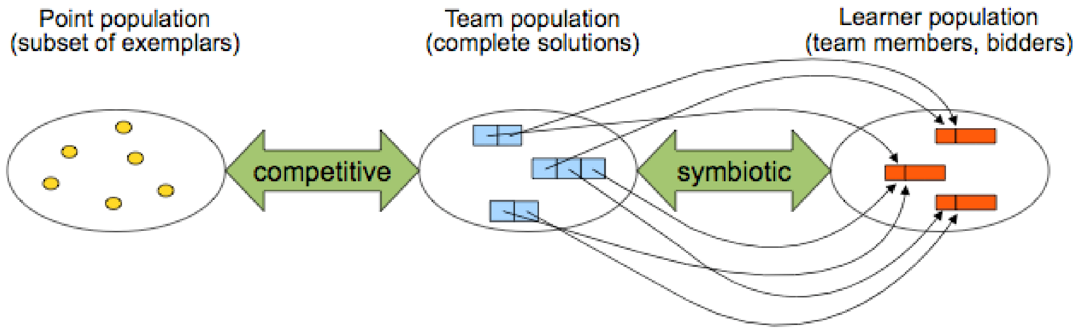

Figure 4.1: Visual depiction of the SBB-GP algorithm, originally appearing in [18], reprinted here with the author's permission.

4.2 The Santa Fe Trail, Part II

The first benchmark considered in experiment 2 was a modified version of the Santa Fe Trail problem from experiment 1. Since SBB-GP does not lend itself to direct modification of its function sets, the "if-food-ahead" function commonly used in the original benchmark could not be directly implemented. Similarly, the fact the SBB-GP's individual team members do not produce actions with their code, but rather bids, precluded the inclusion of the traditional "prog2" and "prog3" operators. As this greatly reduced the information available to the algorithm, eliminating the implicit memory capabilities supplied by the prog2 and prog3 functions, and requiring the evolution of an if-food-ahead function *in silico*, additional information was granted to the algorithm. Control programs were given both the contents of the square directly ahead of them, and the absolute facing of the ant as input. The absolute facing was computed by assigning the directions "North", "East", "South", and "West" to their corresponding directions on the 2d plane created by "unrolling" the toroidal ant world with the ant's starting position in the upper left corner, as shown in figure 4.2.

Much of SBB-GP's success depends on coevolutionary interactions between it the set of training environments with which it is provided. This necessitated creating a set of training environments that was larger than the original Santa Fe Trail alone. The 100 ant trails generated for use as test data in experiment 1 were split into two groups of 50. Each trail was then copied 4 times, creating versions in which the ant starts the simulation facing north, east, south, or west respectively, creating 200 trails in each group. Ants were trained on one group and evaluated on the other.

4.3 Mazes

The second benchmark considered in experiment 2 was a robot maze navigation task, in the spirit of [15]. A simulated circular robot 0.25 units in diameter was placed in a maze with passages of width 1 unit, and 90° angle corners. Control programs were given 13 inputs, and asked to produce one of 6 possible outputs at each time step of the simulation. Inputs 1-8 corresponded to the distance to the nearest maze wall in each of the 8 cardinal directions, relative to the robot's current heading. Inputs 9-12 were binary, set to 1 if the maze exit lay in the corresponding quadrant relative to the robot's current facing, and 0 otherwise. Input 13 was the robot's current speed, and was allowed to range between -1 and +1 units per time step. Figure 4.3 offers a visual representation of the inputs. The 6 available outputs modified the robot's velocity. Output 1 left the velocity unchanged during the current time step. Outputs 2 and 3 accelerated and decelerated respectively by 0.1 units. Using outputs 2 or 3 while at the maximum or minimum speed produced the same result as using output 1. Outputs 4, 5, and 6 caused the angle of the robot's velocity vector to change by -90°, +90°, and 180° respectively. At each time step, the simulation updated the robot's position after resolving the output action's effect. If the robot collided with a maze wall, it bounced back a short distance (0.05 units) along the impact vector, and the simulation set its speed to 0. The robot could rotate in place when moving at a speed of zero. Control programs solved a maze by successfully guiding the robot from a designated start location to a designated exit location within a specified number of time steps.

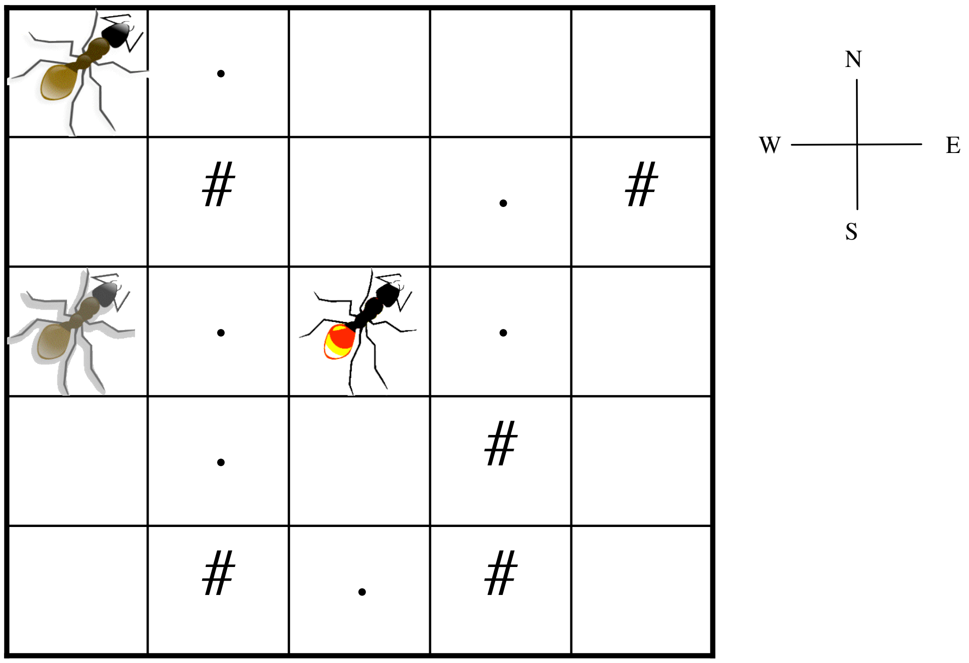

Figure 4.2: An “unrolled” ant world. The ant’s starting position is always in the top left (0,0). Thus, an ant moving to the right on this 2d grid would be moving “east”, while an ant moving down would be moving “south”. For example, the shortest path from the red ant (2,2) to the brown ant (0,0) involves moving “north” and “west”.

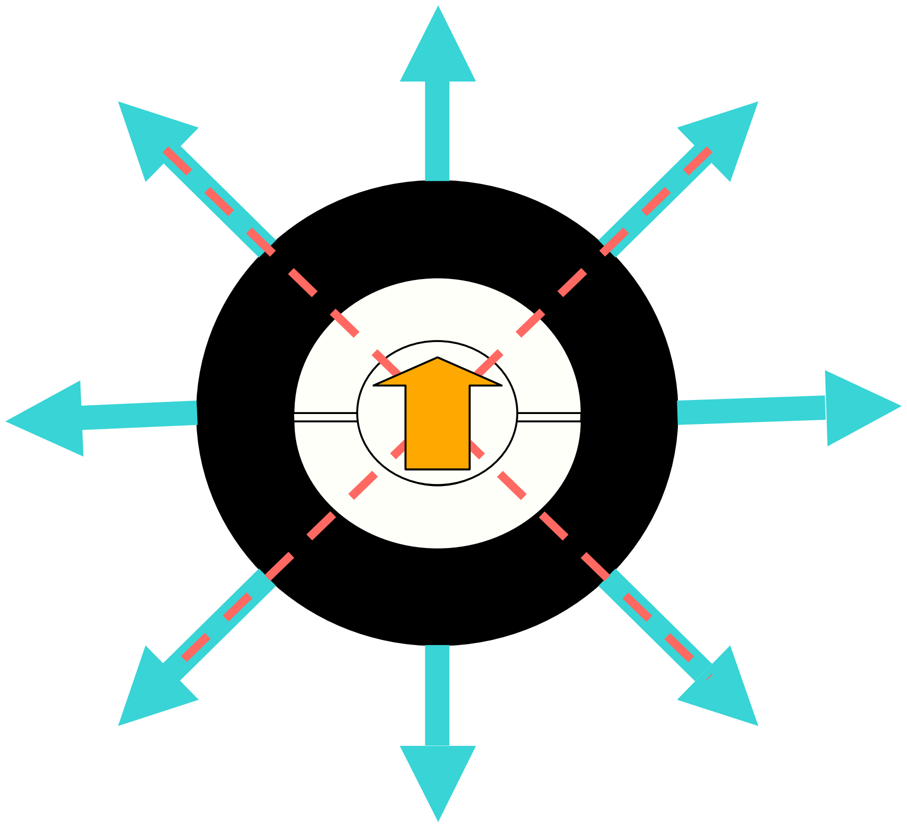

Figure 4.3: Visualization of the maze robot's sensors. Sensors 1-8 are portrayed as rangefinders in blue. The detection zones corresponding to sensors 9-12 are segmented by dashed red lines. The orange arrow in the centre indicates the robot's current heading.

4.3.1 Maze Generation

Two sets of mazes were generated for use in experiment 2, one for training, and one for evaluation. Each set was comprised of 25 “base mazes” which were randomly generated according to algorithm 6 with parameters S = 10 and G = 5. The algorithm creates a square with a specified side-length S. At each integer value along the x-axis, a vertical line segment is added, sectioning off a piece of the square. Into this line segment a randomly placed gap of width G is inserted. A random start place for the robot is assigned between the lowest vertical line and the left wall of the bounding square. A random ending place is similarly assigned between the rightmost vertical line and the right wall of the bounding square. A sample maze produced by this method is shown in figure 4.4.

To provide greater variation in the mazes, each of these 25 base mazes was rotated by 90° increments about the upper left corner of the maze, to generate four related mazes. These mazes shared the same structure, but involved making different initial rotations of the robot, which always began facing directly along the positive X axis. Thus both the training and test data sets contained 100 mazes in total, but only 25 unique patterns.

4.3.2 Phenotype Definitions

A euclidian phenotypic distance measure, as defined in algorithm 2 was used to determine novelty in the maze problem. Two individuals were said to exhibit novel performance relative to one another in a particular maze if their ending points were at least Δ_{min} units apart. This measure was used to replace the expression for $d_{tk}[ni + j]$ in the SBB-GP algorithm, as described in equation 3 of [16]. The fitness sharing system built into SBB was thus used in place of an archiving system.

4.4 Experimental setups

The parameters selected for SBB-GP were based on those from [16], and are summarized in table 4.1. A full description of each parameter can be found in [18]. The maximum number of generations was significantly reduced due to the high computational requirements of the novelty-based evaluation scheme.

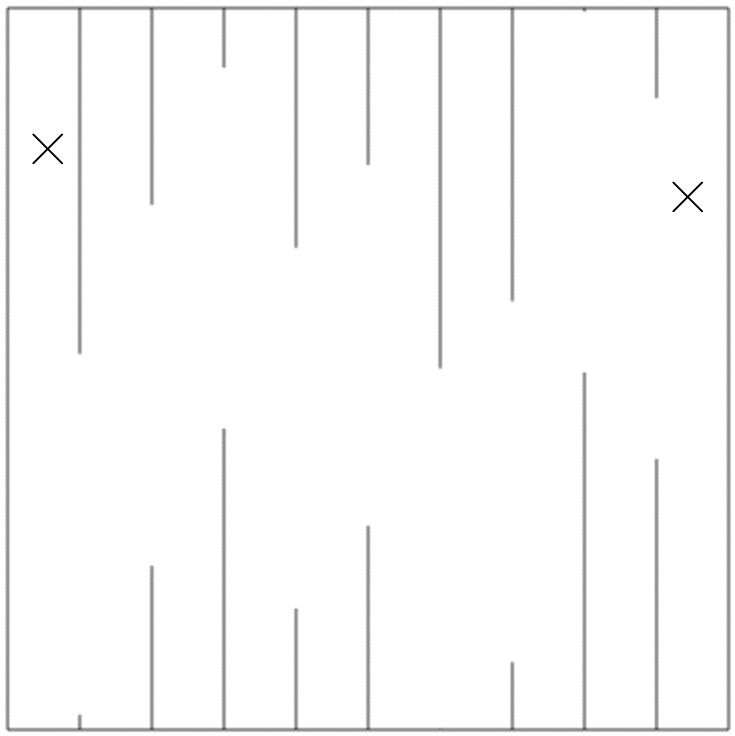

Figure 4.4: A sample maze built using algorithm 6 with G = 5 and S = 10. A maze navigation robot would be tasked with starting at one X, and reaching the other within a fixed number of time steps.

Algorithm 6: Maze Generation(G,S) : Generates a new random maze based on parameters G and S

```
   Input: G: an integer gapsize. S: the side length of a square maze > G+2.
   Output: A list of line segments which comprise a randomly generated maze.
1  Let segs be a list containing line segments defining a square of side length S,
   with a bottom left corner at the origin.
2  for $i = 1$ to $S - 1$ do
3      $x_1 = i$;
4      $y_1 = 0$;
5      $x_2 = i$;
6      $y_2 = random(1, S - (G + 1))$;
7      $segs.add(new\_line\_segment(x_1, y_1, x_2, y_2))$;
8      $segs.add(new\_line\_segment(x_2, y_2, i, y_2 + G))$;
9  end
   Result: segs
```

Table 4.1: The parameters used in benchmarking novelty-based fitness measures in SBB-GP

Parameter	Value
Function Set	+, x, -, ÷, cos, sin, exp, log, if_lt
Max Time Steps	400
Min Program Length	1
Bid Add/Delete/Mutate/Swap Probability	0.5
Team Add/Delete/Mutate/New	0.9/03/0.1/0.1
Number of Registers	8
T_{pop} size	1000
T_{pop} gap	500
P_{pop} size	30
P_{pop} size	15
Maximum Generations	40

Experiment 2's goal was to test the *interaction* of two factors, novelty and symbiotic teaming. This required a two-way test of interaction, such as 2-way analysis of varriance (2-way ANOVA). The computational requirements of the novelty-based evaluation schemes and prolonged simulations times (particularly in the maze problem) prevented the attainment of the very large sample sizes used in experiment 1. Smaller sample sizes in GP often produce non-normal distributions of data, and there is no accepted general-purpose non-parametric test of interaction [26]. While specific tests, such as the Friedmann Test, allow for non-parametric analysis in block designs, GP experiments do not conform to the requirements of commonality within blocks in this case. One suggested method [3] is to perform both a standard 2-way ANOVA on the raw data and also on the data ranks. Identical results are indicative of 2-way ANOVA being unaffected by the violation of the normality assumption, and thus suitable for analysis in this situation. [26] suggests that 2-way ANOVA is most robust to violations of the normality assumption when the number of factors is small, as in a 2x2 layout. For this reason, only one novelty metric was compared for each problem, rather than the three metrics considered in experiment 1.

The experimental setup was comprised of 4 treatment groups for each problem. Groups 1 and 2 were evaluated with maximum team sizes of 9 and maximum program sizes of 48, while groups 3 and 4 were evaluated with teams comprised of one "monolithic" individual with up to 432 ($9 * 48$) instructions. Groups 1 and 3 were evaluated with a problem-specific novelty-based fitness measure, while groups 2 and 4 were evaluated with a qualitative measure. Table 4.2 offers a pictorial representation of the experimental design. Each of the four treatment groups contained 100 sample runs with different random seeds.

Table 4.2: Experimental design for experiment 2. Each group contains 100 data points.

	Qualitative	Novelty-Based
Teaming	Group 1	Group 2
Monolithic	Group 3	Group 4

4.4.1 The Santa Fe Trail

The additional parameters used in the Santa Fe Trail are specified in table 4.3. A pair of individuals was rated “novel” with respect to a particular trail if they differed in at least 30% of the cells of their phenotypic representations.

Table 4.3: The specific SBB-GP parameters used for the Santa Fe Trail benchmark.

Parameter	Value
Action Set	Left, Right, Move_Ahead
Terminal Set	Ant_Facing, Food_Ahead
Phenotypic Novelty Radius	30% of maximum food on a particular trail

4.4.2 Maze

The additional parameters used in the maze navigation problem are specified in table 4.4. A simple density function was used to determine fitness. If two members of the population ended their simulation more than 3 units of Euclidian distance apart, they received a fitness of 1 for that maze. Otherwise, their fitness was proportionate to the distance between them divided by 3.

Table 4.4: The specific SBB-GP parameters used for the maze navigation benchmark.

Parameter	Value
Action Set	Left, Right, Reverse, Faster, Slower, No_Op
Terminal Set	Values of the 8 Radial Rangefinders and 4 ”Compass” Sensors, Current Speed
Phenotypic Radius	3

4.5 Results

4.5.1 The Santa Fe Trail

The data for the observed mean test performance of champions in all four treatment groups on the Santa Fe Trail is shown in figure 4.5 . Several of the distributions shown in figure 4.5 appear skewed or strongly multi-modal, a notion supported by quantitative measures. A Jarque-Bera test at a 95% confidence level found that the distribution of data from three of the four treatment groups was inconsistent with a normal distribution.

As no suitable non-parametric test of interaction was available, the data were analyzed using a 2-way ANOVA test on both the raw and ranked data, as suggested by [3]. The analysis of raw data indicated no interaction was present between the novelty vs. quality and teaming vs. monolithic factors. The analysis of ranked data disagreed, producing a very strong indication that interaction was present ($p-value < 1.0e-6$). Disagreement between the two tests suggests that the violation of the normality assumption for 2-way ANOVA has a significant effect in this case, and thus, the analysis is potentially misleading. No useful information about the interaction effects could be drawn from this data.

Additional analysis proceeded under the assumption that there was no significant interaction. A 2-way ANOVA test without interaction was performed on the ranked and raw data, and followed with post-hoc analysis using Wilcoxon rank sum tests and the Bonferroni correction procedure. The Novelty/Quality factor had a significant effect in both the raw and ranked data in the Santa Fe Trail problem, producing a difference between sample means of 33% in favor of the qualitative measure. The Teaming/Monolithic factor also produced a significant effect in both the raw and ranked data in the Santa Fe Trail problem, producing a difference in sample means of 12% in favour of the monolithic program design. The differences in means were significant at a 95% confidence level.

4.5.2 Maze Navigation

The data for the observed mean test performance of champions in all four treatment groups on the maze navigation problem is shown in figure 4.6 . While most of the treatment groups in this problem produced normally distributed data, the skew in the Quality/Monolithic treatment group was large enough to prevent direct parametric analysis. This was confirmed using a Jarque-Bera test at a 95% confidence level, as in the Santa Fe Trail problem. The 2-way ANOVA test on both the raw and ranked data used for the Santa Fe Trail problem above was applied here as well. Both the ranked and raw data analysis indicated a lack of interaction between the two factors.

Additional analysis proceeded under the assumption that there was no significant interaction. A 2-way ANOVA test without interaction was performed on the ranked and raw data, and followed with post-hoc analysis using Wilcoxon rank sum tests and the Bonferroni correction procedure. The Novelty/Quality factor had no significant effect in both the raw and ranked data in the Santa Fe Trail problem, and so no analysis was performed for this factor. There was a significant effect from the Teaming/Monolithic factor, producing a difference of 4% in sample means. The difference in means was significant at a 95% confidence level.

4.6 Conclusions

The results of experiment 2 suggest that the effect from the interaction between teaming and novelty is too small to be detected with the reasonably large sample sizes used, and thus likely too small to be a significant factor in the performance of other methods utilizing novelty-based fitness measures. The post-hoc analysis from experiment 2 also offers conflicting answers regarding the usefulness of novelty-based measures to GP in general. Novelty-based measures performed very well on the maze problem, agreeing with the results from [15], and also demonstrating there is no significant difference in test performance on that problem between qualitative and novelty-based fitness measures. In contrast, the Santa Fe Trail problem offers the opposite result, agreeing with the results of experiment 1 (see chapter 3), which show novelty alone to be ineffective. One possible explanation for this is the much larger variety of possible solutions in the Santa Fe Trail's behavioral representation. While the maze navigation problem allows comparison across only two dimensions (the X and Y axis of the maze), the Santa Fe Trail problem has 89 dimensions to compare across. Thus, the number of possible solutions to the Santa Fe Trail which could be rewarded for novel behavior will be enormous (very roughly, proportionate to the number of permutations of pieces of food). In contrast, the number of possible solutions rewarded in maze navigation will be considerably smaller, on the order of square of the side length of the maze. This very large difference in the size of the search spaces could be addressed by choosing a different behavioral representation for the Santa Fe Trail in future work.

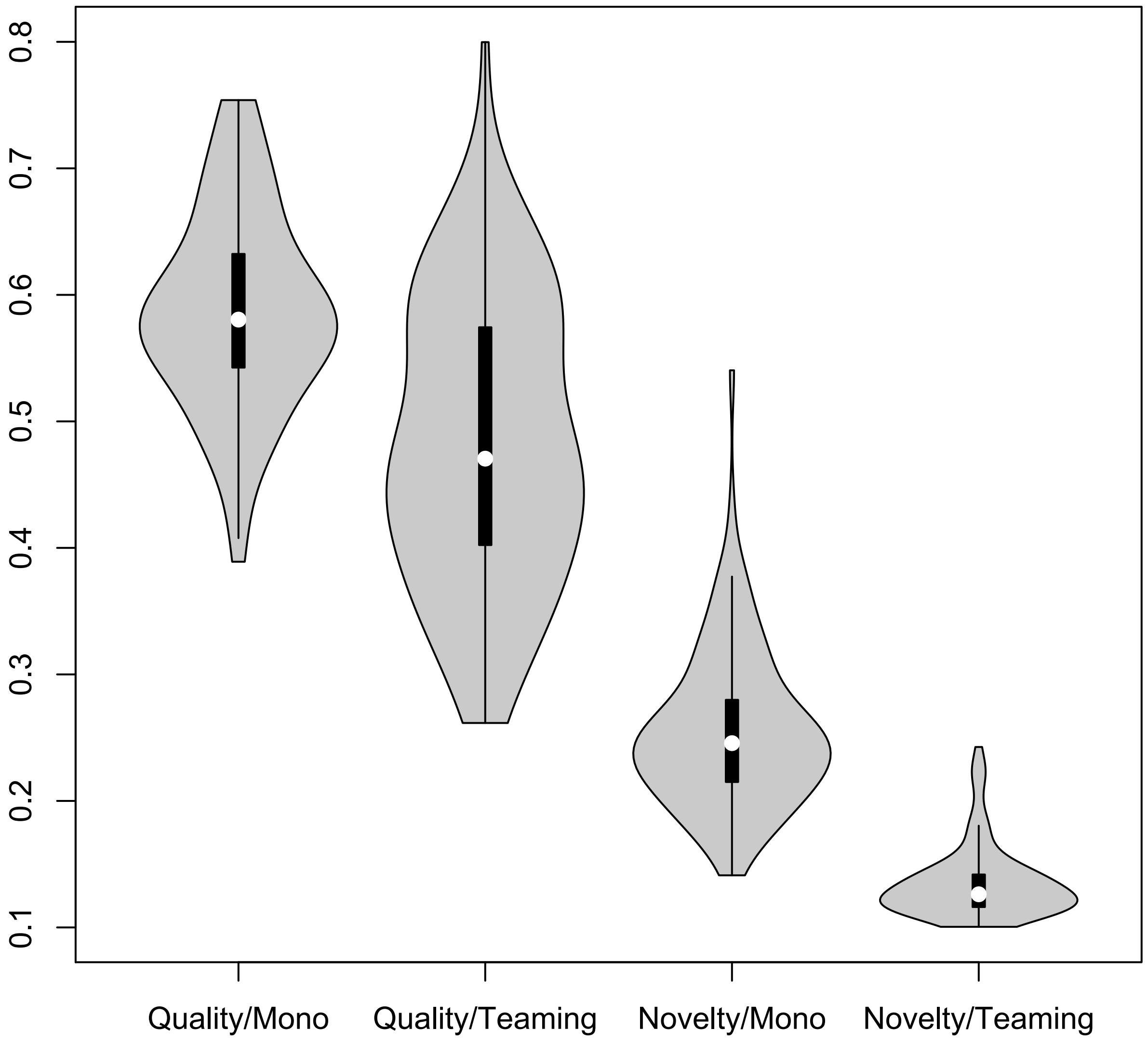

Figure 4.5: Violin plot showing the results of experiment 2 on the Santa Fe Trail problem. See figure 3.4 caption for interpretation of violin plots. The two factors listed below each group on the x-axis indicate the treatments applied, indicating the use of a qualitative or novelty-based fitness measure, and a symbiotic teaming or monolithic program design.

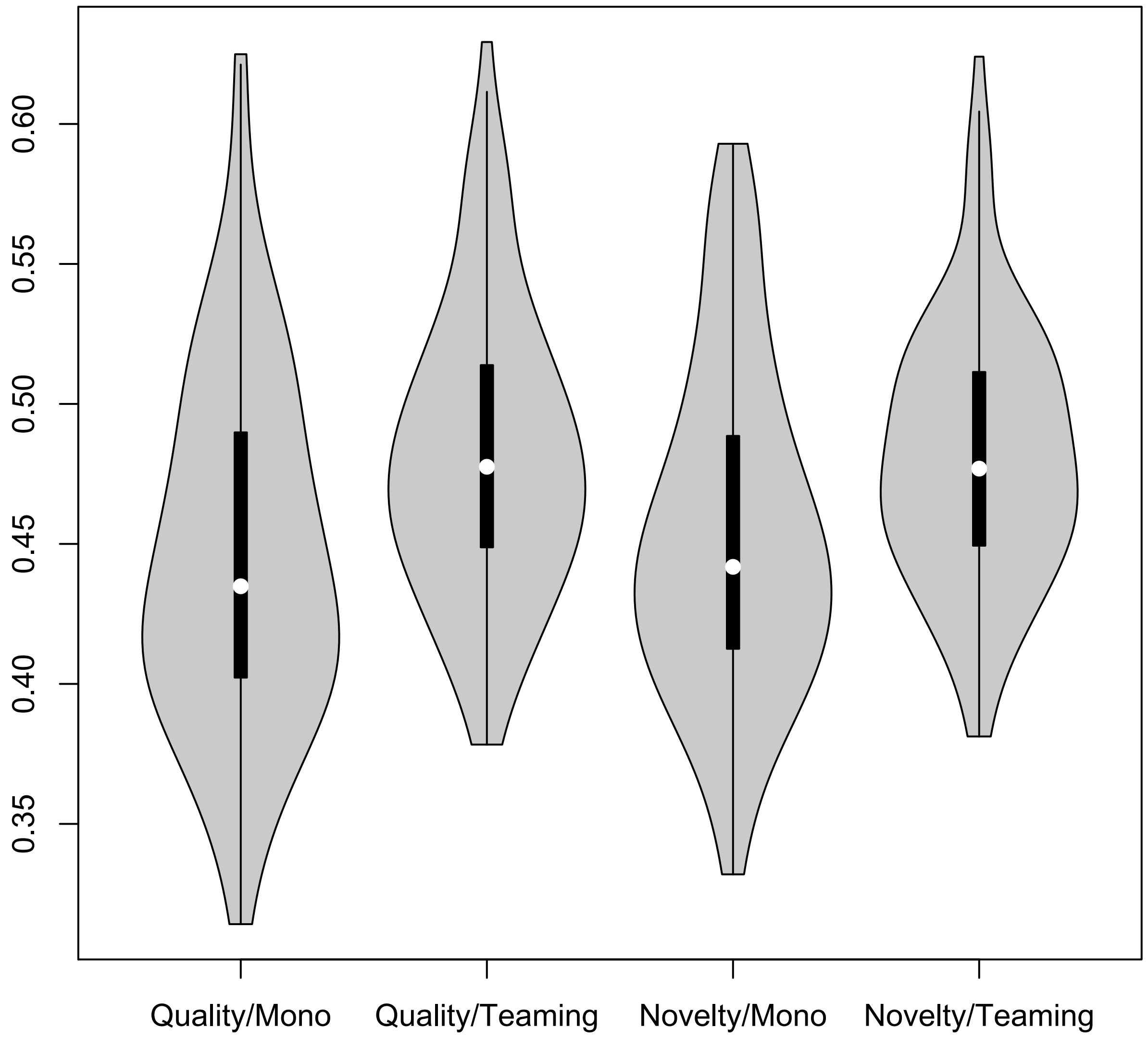

Figure 4.6: Violin plot showing the results of experiment 2 on the maze navigation problem. See figure 3.4 caption for interpretation of violin plots. The two factors listed below each group on the x-axis indicate the treatments applied, indicating the use of a qualitative or novelty-based fitness measure, and a symbiotic teaming or monolithic program design.

Chapter 5

Discussion and Future Work

This research attempted to examine the effects of novelty-based fitness functions in genetic programming, both primary (performance), and secondary (solution generalization, interaction with properties of specific models).

The primary finding of this research is that novelty-based fitness measures may not be effective when a good qualitative measure is available, producing results that are at best, on par with those for simpler, faster qualitative measures of fitness on several traditional GP benchmark problems. At least some of this performance difference may be attributed to novelty-based measures attempting to escape local maxima, potentially leading to earlier discovery of global maxima than a qualitative search would allow [15]. In spite of this, many real world applications of genetic programming feature search spaces large enough or complex enough to make the discovery of truly global maxima unlikely or impossible. In such problem domains, over a limited search time, novelty-based measures should not be expected to produce useful results when compared to qualitative measures.

That said, there are still problem domains for which novelty-based measures may be used to great effect. In search spaces that are bereft of substantiative gradients, novelty-based measures can be used to rapidly locate more acceptable starting positions for a qualitative search than random initializations [21]. As this research shows (chapter 3), in domains where generalization error is important and expensive to determine, novelty-based measures may also be preferable to qualitative ones. In problem domains where the search space is small (as in maze navigation), novelty may find optimal solutions more quickly than quality [15], though in practice problems of this type are not always of great interest. Finally, novelty-based measures provide a way to solve problems where qualitative measures may be ineffective, or where qualitative evaluations may be extremely expensive.

The secondary finding of this research is that the efficacy of novelty-based measures observed in previous research [15, 25] is probably not due to the interaction between the effects of novelty and features of previously evaluated algorithms. In fact, experiment 2 suggests that the choice of problem domain is a much more likely candidate in explaining the discrepancy between earlier results and the findings in experiment 1. This suggests that novelty-based measures may be very sensitive to the dimensionality of the behavioral representation used in a particular problem, recommending their use only when the range of possible behaviors is comparatively small.

These results open many avenues of future research involving novelty and genetic programming. Simple, if time intensive, experiments could be devised to confirm the effects of novelty-based fitness measures on the quality of solutions throughout a prolonged GP run. Such experiments could confirm that the novelty-based measures in this work perform poorly as a result of attempts to escape local maxima, theoretically resulting in faster convergence on global maxima. Such a result would be in agreement with prior work [15, 25]. Experiments with a longer timeframe could also allow further investigation of the benefits of novelty-based measures in reducing generalization error. If the effects observed in experiment 1 held during longer training intervals, novelty-based measures could be of considerable utility in certain problem domains. Future work could also examine the utility of a "layered" evolution, in which an alternating schedule of quality and novelty-based fitness measures is used to drive the evolution. This could be an effective method of escaping plateaus [21]. Finally, novelty measures inspired by biological sequence alignment algorithms might be very applicable to GP in spite of their considerable computational costs, and should be investigated in future.

Bibliography

[1] Daniel Adler. *vioplot: Violin plot*, 2005. R package version 0.2.

[2] Anil K. Bera and Carlos M. Jarque. Efficient tests for normality, homoscedasticity and serial independence of regression residuals : Monte carlo evidence. *Economics Letters*, 7(4):313–318, 1981.

[3] W.J. Conover. *Practical Nonparametric Statistics.* Wiley series in probability and mathematical statistics. Wiley and Sons Inc., 3rd edition, 1999.

[4] John Doucette, Peter Lichodzijewski, and Malcolm Heywood. Benchmarking coevolutionary teaming under classification problems with large attribute spaces. In *Proceedings of the Genetic and Evolutionary Computation Confrence (GECCO)*, pages 1901–1902. ACM New York, 2009.

[5] John Doucette, Peter Lichodzijewski, and Malcolm Heywood. *Genetic Programming Theory and Practice VII*, chapter 3, pages 37–54. Springer US, 2010.

[6] John A. Doucette and Malcolm I. Heywood. Novelty-based fitness: An evaluation under the santa fe trail. To be presented at EuroGP 2010, April 2010.

[7] Marco Tomassini Francisco Fernandez and Leonardo Vanneschi. An empirical study of multipopulation genetic programming. *Genetic Programming and Evolvable Machines*, 4:21–51, 2003.

[8] F. J. Gomez. Sustaining diversity using behavioral information distance. In *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO)*, pages 113–120. ACM, 2009.

[9] R. W. Hamming. Error detecting and error correcting codes. *Bell System Technical Journal*, 29(2):147–160, 1950.

[10] J. L. Hintze and R. D. Nelson. Violin plots: a box plot-density trace synergism. *The American Statistician*, 52(2):181–4, 1998.

[11] John Koza. *Genetic Programming: On the Programming of Computers by Means of Natural Selection.* MIT Press, 1992.

[12] Ibrahim Kuscu. Evolving a generalised behavior: Artificial ant problem revisited. In *Evolutionary Programming VII*, pages 799–808. Springer Berlin, 1998.

[13] I. Kushchu. Genetic programming and evolutionary generalization. *IEEE Transactions on Evolutionary Computation*, 6(5):431–442, 2002.

[14] W. B. Langdon and R. Poli. *Foundations of Genetic Programming.* Springer, 2002.

[15] J. Lehman and K. O. Stanley. Exploiting open-endedness to solve problems through the search for novelty. In *Proceedings of the International Conference on Artificial Life(ALIFE XI)*, pages 363–370. MIT Press, 2008.

[16] P. Lichodzijewski and M. I. Heywood. Pareto-coevolutionary genetic programming for problem deco,position in multi-class classification. In *Proceedings of the Genetic and Evolutionary Computation Confrence (GECCO)*, pages 464–471. ACM New York, 2007.

[17] P. Lichodzijewski and M. I. Heywood. Managing team-based problem solving with symbiotic bid-based genetic programming. In *Proceedings of the Genetic and Evolutionary Computation Conference*, pages 262–270, 2008.

[18] Peter Lichodzijewski and Malcolm I. Heywood. Symbiosis, complexification and simplicity under gp. In *Proceedings of the Genetic and Evolutionary Computation Confrence (GECCO).* ACM New York, 2010.

[19] A. R. McIntyre and M. I. Heywood. Cooperative problem decomposition in pareto competitive classifier models of coevolution. In *Proceedings of the European Conference on Genetic Programming (EuroGP)*, pages 289–300. Springer, 2008.

[20] Ronald W. Morrison. *Designing Evolutionary Algorithms for Dynamic Enviroments.* Springer -Verlag Berlin, 2004.

[21] J. B. Mouret and S. Doncieux. Overcoming the bootstrap problem in evolutionary robotics using behavioral diversity. In *IEEE Congress on Evolutionary Computation*, pages 1161–1168, 2009.

[22] Michael O'Neill and Conor Ryan. Evolving multi-line compilable c programs. In *Second European Workshop, EuroGP'99*, pages 651–660. Springer Berlin, 1999.

[23] Zuzana Oplatkova and Ivan Zelinka. Investigation on artificial ant using analytic programming. In *Proceedings of the 8th annual conference on Genetic and evolutionary computation*, pages 949–950. ACM New York, 2006.

[24] R Development Core Team. *R: A Language and Environment for Statistical Computing.* R Foundation for Statistical Computing, Vienna, Austria, 2009. ISBN 3-900051-07-0.

[25] S. Risi, S. D. Vanderbleek, C. E. Hughes, and K. O. Stanley. How novelty search escapes the deceptive trap of learning to learn. In *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO)*, pages 153–160. ACM, 2009.

[26] Shlomo S. Sawilowsky. Nonparametric tests of interaction in experimental design. *Review of Educational Research*, 60(91):91–126, 1990.

[27] Terence Soule and Robert Heckendown. An analysis of the causes of code growth in genetic programming. *Genetic Programming and Evolvable Machines*, 3:283–309, 2002.

[28] David H. Wolpert and William G. Macready. No free lunch theorms for search. Technical report, The Santa Fe Institute, 1996.

[29] D. Zongker and B. Punch. *lil-gp 1.0 User's Manual*. Michigan State University, 1995.