A Behavior-based Analysis of Modal Problems

Leonardo Trujillo
Enrique Naredo
Yuliana Martínez
Instituto Tecnológico de Tijuana, México
leonardo.trujillo@tectijuana.edu.mx
{enriquenaredo,ysaraimr}@gmail.com

Lee Spector
Cognitive Science
Hampshire College
Amherst, MA 01002
lspector@hampshire.edu

ABSTRACT

Genetic programming (GP) has proven to be a powerful tool for (semi)automated problem solving in various domains. However, while the algorithmic aspects of GP have been a primary object of study, there is a need to enhance the understanding of the problems where GP is applied. One particular goal is to categorize problems in a meaningful way, in order to select the best tools that can possibly be used to solve them. This paper studies modal problems, a conceptual class of problems recently proposed by Spector at GECCO 2012. Modal problems are those for which a solution program requires different modes of operation for different contexts. The thesis of this paper is that modality, in this sense, is better understood by analyzing program performance in behavioral space. The behavior-based perspective is seen as part of a scale of different forms of analyzing performance; with a coarse view given by a global fitness value and a highly detailed view provided by the semantics approach. On the other hand, behavioral analysis is seen as a flexible approach where the context of a program's performance is considered at in a domain-specific manner. The experimental evidence presented here suggests that behavior-based search could allow a GP to find programs with disjoint behavioral structures, that can satisfy the requirements of each mode of operation of a modal problem.

Categories and Subject Descriptors

I.2.2 [**Artificial Intelligence**]: Automatic Programming—*program synthesis*

General Terms

Performance,Experimentation,Theory

Keywords

Genetic Programming, Modal Problems, Behaviors, Novelty Search, Semantics

1. INTRODUCTION

Evolutionary computation research has generated a diverse set of algorithmic tools for search and optimization that are robust and applicable to many domains. Nonetheless, it is also clear that each algorithm is well suited for some problems and fails in others. Therefore, there is a need to analyze, understand and categorize problems in a meaningful way [11], to define problem classes and identify what type of algorithms perform well on each problem class.

Spector [26] describes a new conceptual class of problems for genetic programming (GP). To understand it, we must assess the state of current GP research. While the original goal of GP, the automatic synthesis of computer programs, still remains as a guiding principle, current systems have hit a more realistic plateau. GP has proven to be a tool for the automatic generation of *small* computer functions or mathematical expressions that perform very specific tasks.

There are many reasons why synthesizing complete software solution still lies beyond current algorithms. In particular, Spector proposes the concept of modality to describe problems where a software system must produce qualitatively different actions for different types of inputs. He refers to such input-action pairs as a program's mode of operation; i.e., to solve a modal problem a program must exhibit different modes of operation for each context or circumstance. Spector points out that, in general, GP systems "are generally limited to problems for which solution programs can perform similar actions for all possible inputs" [26]. Here, we refer to these type of problems as *modal problems* and the general concept as *problem modality*; stressing the difference of the nomenclature with other uses of the term. For instance, the term *modal* is used to describe *multimodal* landscapes [30], or *multimodal* information processing systems [25].

Nevertheless, the concept of modality is somewhat ambiguous. For instance, it might not be clear how modality can be used to further problem understanding. It could seem that all problems are modal, since *performing a different action* for *different inputs* can be considered as a tautology. However, it is important to consider that different actions are not implied by different outputs, just consider a simple function $y = x + 2$. Another issue with problem modality is that it overlaps with other concepts in the field, particularly multi-objective and decomposable problems. Spector [26] outlines arguments to distinguish problem modality, which are here recounted in Section 2. Spector also proposes a measure of modality; however, it leads to a circular definition that depends on the solution strategy. Nonetheless, thus far it is the only quantitative measure proposed for problem modality, and the issue is not addressed here.

In this paper, the goal is to provide a different, hopefully clarified, picture of modal problems using the concept of behavioral space. Instead of focusing on an analysis of input-output pairs, the proposal is to move towards an understanding of program behaviors. Behaviors are a description of what a solution (GP individual) does within a given environment. The behavior of an individual is expressed in a domain-specific manner, using domain knowledge to describe how an individual performs. In this way, under some reasonable assumptions, each individual is mapped to a single point in behavioral space. This introduces a higher level of abstraction, compared to the more traditional fitness, genotypic or phenotypic spaces considered in evolutionary algorithms. From this high-level view, it is possible to pose the search within a behavioral space that is intuitive and natural to the problem domain. Then, a modal problem can be described as one that requires complex or disjoint behavioral solutions; i.e., solutions that exhibit two or more distinct behavioral patterns, one for each context in which the solution can be expected to operate. For instance, a robot controller might induce a robot to exhibit a wall following pattern and an obstacle avoidance pattern as part of a navigation behavior, where each pattern is activated by different environmental contexts. Concretely, Section 5 proposes two domains where modal problems could be expected: evolutionary robotics and pattern analysis. This work argues that by leveraging the idea of behavioral-space, problem modality can be defined and understood more clearly. Moreover, the paper suggests that searching for solutions of modal problems is easily posed by considering behavioral space. In particular, the paper reports recent results of applying novelty search (NS) [8, 10, 17, 16], a behavior-based search that focuses selective pressure on finding novel solutions.

The remainder of this paper proceeds as follows. Section 2 defines the concept of modal problems [26]. Section 3 describes behavioral space and behavior-based search. Section 4 analyzes modal problems from the behavioral space perspective and proposes novelty search as a plausible solution strategy. Section 5 proposes two domains where modal problems are common and shows how behavior-based search has been used to solve them. Conclusions are given in Section 6.

2. PROBLEM MODALITY

As stated above, Spector [26] defines a problem to be modal if a solution has to perform qualitatively different actions depending on the inputs. Moreover, it is important to consider that we may or may not know in advance how many modes of operation are required to solve a problem.

Modal problems are similar to multi-objective problems [3], since in both cases a solution has to exhibit acceptable performance from various perspectives. It might be possible to pose modal problems as multi-objective, by considering performance in each mode of operation as a separate objective. However, this would only be possible if the different modes were known in advance. Moreover, multi-objective problems are not necessarily modal, since a single mode could be evaluated using different performance criteria.

Another conceptual class related to modal problems is decomposable or modular problems. A decomposable problem can be partitioned, or decomposed, into a set of smaller sub-problems, or modules, following a divide-and-conquer approach. It is normally expected that sub-problems will be easier then the original one, and that solutions will be easier to evolve. The complete solution for a decomposable problem can then be constructed from an aggregation of sub-problem solutions. In the more general and difficult case, the number and form of these sub-problems is not known in advance. This topic has been extensively studied from different perspectives and following different approaches [4, 22, 31]. In this sense, each mode of operation of a modal problem could be solved by a different specialized module. However, such a strategy might only increase complexity and decrease code optimization, since sub-problems can conceivably require the same, or similar, portions of code. Moreover, if non-linear interdependencies exist between each sub-problem, then concurrently decomposing and optimizing individual solutions will not be a trivial endeavor [4].

For these reasons, Spector proposes a novel selection strategy for modal problems called lexicase selection. During evolution, lexicase selection selects parents by starting with a pool of candidates and filtering it on the basis of performance on single fitness cases, considered one after another. In the basic implementation, the initial pool of candidates is all the population and the fitness cases are ordered randomly. Moreover, only the best individuals are selected at each iteration, an elitist approach. Spector presented initial experiments on a symbolic regression problem for GP with good results. However, lexicase selection suffers from two shortcomings. First, in its most basic form it is a computationally expensive process. Second, it is not only a proposal to solve modal problems but it is also proposed as a measure to determine if a problem is modal or not; a logical problem discussed at length by Spector [26]. However, instead of focusing on the circularity of the original definition (which is not addressed or resolved in this work), this paper argues that the concept of problem modality is troublesome because of the level of abstraction at which program performance is analyzed. Therefore, the proposal is to use the concept of behavioral space to define, and hopefully clarify problem modality. Based on this definition, a plausible solution strategy is proposed based on a behavior-based open-ended search. This paper does not suggest that lexicase selection should be abandoned as a plausible approach to solve modal problems; indeed, it has recently been demonstrated to have utility on other problems as well [6]. The goal here is to propose a different conceptual framework with which to describe problem modality and explore alternative solution strategies.

3. BEHAVIOR-BASED SEARCH

In general, an evolutionary algorithm concurrently samples three different spaces during a search. First, genotypic or search space, which corresponds to the encoding used to represent each valid solution to a problem. Second, phenotypic or decision space, which represents the domain specific space where solutions are expressed, such as program space. Finally, objective or fitness space, that corresponds to the space constructed by the set of performance criteria.

While these three spaces have been the focus of most research in GP, other spaces have recently entered discourse in the field. Consider that, in most applications, the fitness of a GP individual gives a coarse global evaluation of its performance, usually averaging out differences in program quality when it is evaluated on several fitness cases. Considering this, some researchers have proposed a finer grained approach to analyze an evolved program's performance.

One approach is known as semantics in GP literature [12, 13, 29], with its corresponding semantic space. Semantic space in GP corresponds to the space of possible program outputs [13]; i.e., given a set of n fitness cases the semantics of a program K is the corresponding set of outputs it produces, normally expressed as a vector $\mathbf{y} \in \mathbb{R}^n$. Semantics is an important concept in GP because (vastly) many genetically different programs can produce the same semantics. However, strictly analyzing program outputs might not be the best approach in some domains. For instance, consider the GP classifier based on static range selection (SRS) described in [32]. For a two class problem and real-valued GP outputs, the SRS classifier is simple: if the program output for input pattern $\mathbf{x}$ is greater than zero then the pattern is labeled as belonging to class A, otherwise it is labeled as a class B pattern. In this case, while the semantics (as defined above) of two programs might be strictly different, they can still produce the same high-level classification (to illustrate this, consider any two outputs $\mathbf{y}_1, \mathbf{y}_2 \in (0, \infty)$ with $\mathbf{y}_1 \neq \mathbf{y}_2$).

Now consider a second example from another domain. In evolutionary robotics (ER), evolutionary algorithms are used to search for robust controllers of autonomous systems [20]. What is of importance in the ER approach, is to find high quality solutions introducing as little prior knowledge as possible into the objective function; i.e, the evolutionary process should perform a search based on a very high-level definition of the robotic task [18]. In this scenario, the correspondence between program inputs, outputs and induced actions is much less clear cut. Moreover, evolution in an ER system is performed within real or simulated environments, where noisy sensors and the physical coupling between actuators and the real world can produce a non-injective and non-surjective relation between program output and the actions performed by the robot.

Therefore, researchers in ER have proposed another approach towards performance analysis, focused on the idea of behavioral space [14, 27]. The concept of behaviors in robotics dates back to the seminal works of Brooks from the 1980's [1], making it easy to include the concept of behaviors in ER research. A behavior can be understood as a description β of the way an agent K (program in the GP case) acts in response to a series of stimuli within a particular context $\mathcal{C}$. A context $\mathcal{C}$ includes the internal state of the robot, external environmental conditions that are not sensed or given as input, and the coupling process between the outputs and the actions performed. Stated another way, a behavior β is produced by the interactions of agent K, output $\mathbf{y}$ and context $\mathcal{C}$. In behavior-based robotics, for instance, behaviors were described at a very high level of abstraction by the system designer. In ER, on the other hand, researchers have recently proposed domain-specific numerical descriptors that describe each behavior, allowing them to explicitly consider behavioral space during evolution [14]. The justification for this is evident, given that the objective function is stated as a high-level goal, then population management should take in to account the behavioral aspect of the solutions that are evolved. Following this approach, researchers have been able to develop diversity preservation techniques [27, 28] and open-ended search algorithms [8, 10]; for a comprehensive review on the topic see [14].

Care must be taken to understand the suggested difference

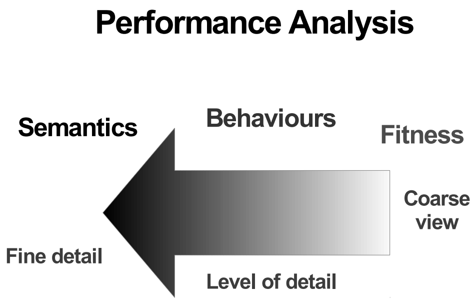


Figure 1: Conceptual view of how the performance of a program can be analyzed. At one extreme we have fitness-based analysis, a coarse view of performance. Semantics lies at another extreme, where a high level of detail is sought. Finally, behavioral analysis provides a variable scale based on how context is considered.

between semantics and behaviors. A behavior, as defined above, is a higher-level description of the semantics of a program. An individual's behavior is described at a higher level of abstraction, accounting not only for program output but the context in which the output was produced. While semantics can imply an injective or non-injective relation between input and output, behaviors are more general, allowing for multi-valued functions or many-to-many relations, if only input is considered and context is not. For instance, for the SRS GP classifier described above, context is given by the SRS heuristic rule used to assign class label. This is more evident in the ER examples, where context includes those parts of the surrounding environment that are not sensed or considered by the controller and that can effect behavior, what is known as the embodiment problem [1].

A final comment is pertinent. In essence, fitness, semantics, and behaviors are different levels of abstraction of a program's performance, see Figure 1. At one extreme form of analysis, fitness provides a coarse grained look at performance, a single value (for each criterion) that attempts to capture a global description of performance. At the other end of the analysis scale, semantics describe program performance in great detail, considering the raw outputs. On the other hand, behavioral descriptors move between fitness and semantics, providing a finer or coarser level of description depending on how behaviors are meaningfully characterized within a particular domain. Therefore, the behavior based approach might provide an advantage in two scenarios: (a) when fitness does not give sufficient detail to differentiate between individuals, given their performance and the current state of a search; or when (b) semantics gives to much detail and you cannot see the forest for the trees, and a higher-level view of performance might be required.

4. MODAL PROBLEMS AND BEHAVIOR-BASED SEARCH

The main thesis of this paper is that modal problems may be productively described and understood in behavioral space. The main argument is that we can equate each mode of program operation with a different context in which the program can be applied. Since a program that behaves differently in different contexts will exhibit a disjoint behavioral descriptor, then each unique behavioral pattern can be considered as a different mode of operation. Therefore, a solution program should exhibit a unique behavioral sequence based on the sequence of contexts it encounters, and modes it must exhibit, to solve a modal problem.

Therefore, a plausible strategy to solve modal problems is to describe performance behaviorally. However, for modal problems, contexts and behaviors cannot be defined generally, they are domain specific and care should be taken to design them. For instance, for some problems a fitness-based description of performance might be sufficient, and for others a detailed semantics approach might be best. Nonetheless, as argued in the previous section, fitness, semantics and behaviors can be interpreted as just different abstractions of performance, and thus the correct level of abstraction must be chosen based on the problem and desired results.

From this perspective, problem modality might only appear to be a conceptual taxonomical tool, that maybe simplifies our understanding but does not allow us to build real algorithms that can explicitly address problem modality. However, such an observation would be incorrect, since it is similar to the original critiques given to Brooks' behavior-based robotics [1], a research program that has produced many promising technological advances [21]. On the contrary, a behavior-based analysis of modal problems might help clarify the object of discourse and maybe lead to a plausible solution strategy. In particular, if a specific behavioral pattern is required, then the evolutionary algorithm should explicitly consider how behavioral space is sampled during the search.

4.1 An overview of Novelty Search

The main idea behind novelty search (NS) is to eliminate an explicit objective function from an evolutionary search [8, 9, 10]. In other words, evolution is not guided by the measured *quality* of each individual, instead it is guided by a measure of *uniqueness*; i.e., how novel each individual is with respect to what has been found by the search in earlier generations. A known limitation of the traditional objective-based search is a tendency to converge and stagnate on local optima, particularly in multi-modal problems with irregular fitness landscapes. Diversity preservation techniques are usually incorporated within an EA to overcome the above shortcoming. However, most proposals are ad-hoc solutions that attempt to balance exploration and exploitation during the search. Conversely, through the search for novelty alone, diversity preservation introduces the sole selective pressure.

In NS, individuals are described in behavioral space using a domain-specific descriptor that captures the main traits of an individual's actions [27, 28]. Since many genetically different individuals can express the same behavior (mapped to the same point in behavioral space), the search for novelty is often feasible. Moreover, if the number of simple behaviors is relatively small, then the search for novelty could lead to "complex" solutions. To clarify, since complexity, in general, can be an ambiguous term, complex solutions are here equated with solutions that exhibit disjoint behaviors, or behaviors that exhibit distinct patterns on different types of inputs. For instance, in ER a disjoint navigation behavior exhibited by a mobile robot might include a wall following pattern and goal homing pattern. Therefore, the thesis of this paper is that problems that require solutions with disjoint behaviors can be regarded as being modal.

Summarizing, NS uses a measure of *novelty* to characterize each individual. The sparseness of each individual within behavioral space is measured, with respect to other individuals in the population and novel solutions from previous generations. NS measures the sparseness ρ around each individual K, described by its behavioral descriptor $\boldsymbol{\beta}$, using the average distance to the k-nearest neighbors in behavioral space, with k an algorithm parameter, given by

$$\rho(\boldsymbol{\beta}) = \frac{1}{k}\sum_{i=0}^{k} dist(\boldsymbol{\beta}, \boldsymbol{\alpha}_i) \;, \tag{1}$$

where $\boldsymbol{\alpha}_i$ is the behavioral descriptor of the ith-nearest neighbor of K in behavioral space with respect to distance metric $dist$, a domain-dependent measure that depends on how descriptors are represented. If the average distance is large then the individual lies within a sparse region of behavioral space and it lies in a dense region when the measure is small.

Sparseness is computed based on the contents of the current population and an archive of individuals that at one moment were considered to be novel. Therefore, an individual is added to the archive if its sparseness is above a minimal threshold ρ_{min}, the second parameter of the NS algorithm. The archive can also mitigate backtracking by the search process. The archive growth can be seen as a shortcoming, because if the archive grows then a higher computational cost is incurred when sparseness is measured. Therefore, [9] implements the archive as a fixed size FIFO queue.

5. DOMAINS WITH MODAL PROBLEMS

The goal of this section is to propose specific domains where modal problems are common. In particular, two domains are considered: robotics and pattern analysis. In [26], Spector analyzes the most common application domain for GP, symbolic regression. The modal problem he presents is a piece-wise function and experiments show that lexicase selection provides improved performance compared with a canonical GP. Recent work has also extended the application of lexicase selection to boolean regression problems [6]. However, providing a behavioral descriptor for symbolic regression is not straightforward and has been proposed only recently [15]; the point is discussed further in Section 6.

5.1 Modal problems in robotics

The first plausible domain where modal problems arise is ER. In ER researchers start with a high-level description of a problem, such as developing a gait [2], autonomous navigation [20] or other complex tasks [19]. The strategy is to use evolution to search for robot behaviors that perform the desired task while introducing the smallest amount of prior knowledge into the search. In most problems, the robot has to perform different types of actions in different contexts. For instance, for navigation a robot must learn to move in a straight line in open spaces or evade obstacles in crowded areas. Each of these can represent the different modes of operation of a modal problem. In fact, considering ER as an extension of the behavior-based approach to robotics, it is easy to see that the original formulation by Brooks explicitly contemplates several modes of operation that are dynamically combined by the subsumption architecture [1].

Moreover, there is strong evidence that corroborates the second claim of this work; that modal problems are effectively solved by explicitly considering behavioral-space during a search. For instance, almost all successful applications of the NS algorithm have been on ER problems, including navigation with obstacle avoidance and gait control [10]. Moreover, as stated above, behavioral space has successfully been used to promote diversity and find a diverse set of solutions for multimodal search spaces [14, 27, 28]. In fact, this has been the case because posing robotic problems in terms of behaviors is a common conceptual tool that continues to play a big role in state-of-the art robotics research [21].

5.2 Modal problems in pattern analysis

A second domain where modal problems can be identified is pattern analysis, particularly in data classification and clustering. By definition, in these problems the goal is to find a function, or program from the GP perspective, that is able to identify underlying regularities within a set of sample patterns. In each case, the program will have multiple modes of operation, depending on the number of classes, or clusters, in which sample patterns are organized. Hence, depending on the input, a program must exhibit distinct modes of operation, deriving the correct mode from the values of the input patterns themselves. In what follows, two recent behavior-based approaches are reviewed, for supervised classification [17] and unsupervised clustering [16].

In both cases, a domain specific behavioral descriptor is proposed and the NS algorithm is used to search within behavioral space. Besides the specifics of the NS algorithm, a traditional Koza-style GP was used for program representation and genetic operators. The parameters of the GP systems are given in Table 1 [1]. The NS-based GP is implemented with an archive limit of 400 individuals, denoted hereafter as NS-GP. The NS parameters are set to $\rho_{min} = 40$ and $k = 15$. In both test cases the algorithm was run 30 times, and implemented using Matlab 2009a and the GPLAB toolbox [23].

5.2.1 Supervised classification

This work uses the static range selection GP classifier (SRS-GPC) described by Zhang and Smart [32]. In a classification problem, a pattern $\mathbf{x} \in \mathbb{R}^p$ has to be classified as belonging to a single class from $\Omega = \{\omega_1, ..., \omega_M\}$, where each ω_i represents a distinct class label. Then, in a supervised learning approach the goal is to build a mapping $g(\mathbf{x}) : \mathbb{R}^p \to \Omega$, that assigns each pattern $\mathbf{x}$ to a corresponding class ω_i, where g is derived based on evidence provided by a training set $\mathcal{T}$ of N p-dimensional patterns with a known classification. In this work, only two-class classification problems are considered. In SRS-GPC, $\mathbb{R}$ is divided into M non-overlapping regions, one for each class. Then, GP evolves a mapping $g(\mathbf{x}) : \mathbb{R}^p \to \mathbb{R}$, such that the region in $\mathbb{R}$ where pattern $\mathbf{x}$ is mapped to, determines the class to which it belongs. For a two-class problem, if $g(\mathbf{x}) > 0$ then $\mathbf{x}$ belongs to class ω_1, and belongs to ω_2 otherwise. The fitness function consists on minimizing the classification error. To apply NS, the fitness function is substituted by the sparseness measure of Equation 1, that requires a domain specific behavioral descriptor.

Accuracy Descriptor $\boldsymbol{\beta}^A$: The training set $\mathcal{T}$ used by SRS-GPC contains sample patterns from each class. For a two-class problem with $\Omega = \{\omega_1, \omega_2\}$, if $\mathcal{T} = \{\mathbf{y}_1, \mathbf{y}_2, ..., \mathbf{y}_L\}$, then the behavioral descriptor for each GP classifier K_i is a binary vector $\boldsymbol{\beta}^{A_i} = (\beta_1, \beta_2, ..., \beta_L)$ of size L, where each vector element β_j is set to 1 if classifier K_i correctly classifies sample $\mathbf{y}_j$, and is set to 0 otherwise. Obviously, the descriptor considers classifier accuracy at a finer level of detail compared with a global fitness score. Now, let $\mathbf{x}$ represent a binary vector, function $u(\mathbf{x})$ return the number of 1s in $\mathbf{x}$, and let K_O be the *optimal* classifier that achieves a perfect accuracy on the training set.

[1]The IF function has arity 3, $IF(a, b, c)$; it evaluates the sub-tree rooted at node c if $a = 0$ and evaluates the sub-tree rooted at b otherwise.

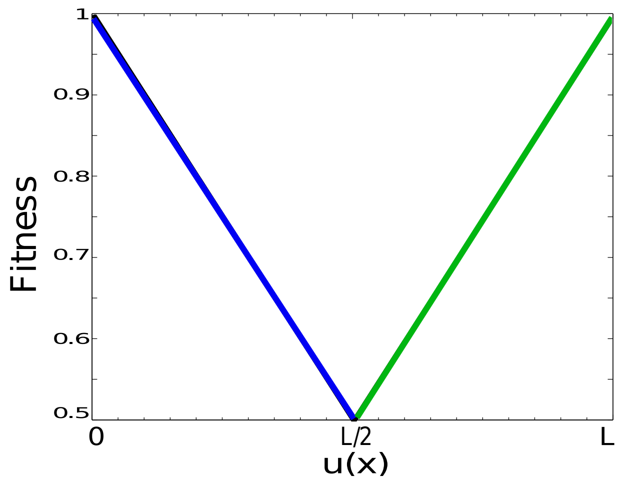


Figure 2: Fitness landscape in behavioral space for the accuracy descriptor.

Then, the descriptor of K_O is $\boldsymbol{\beta}^{A_O}$ where $u(\boldsymbol{\beta}^{A_O}) = L$. For a two-class problem, an equally useful solution is to take the opposite (complement) behavior and invert the classification, such that a 1 is converted to a 0 and vice-versa. This is a similar strategy to what is done to solve deceptive problems in genetic algorithms. The complement behavior is $\boldsymbol{\beta}^{A_*}$ with $u(\boldsymbol{\beta}^{A_*}) = 0$. Figure 2 shows the fitness landscapes of the accuracy descriptor in behavioral space [17]. The landscape has two global optima, at $\boldsymbol{\beta}^{A_O}$ and $\boldsymbol{\beta}^{A_*}$. Notice that the worst performance is given by a random classifier with 50% accuracy, depicted in the middle valley of Figure 2. On difficult problems, worst case performance can be expected from random individuals in the initial generation. However, NS does not search directly within this landscape; indeed an algorithm that could, would be the best possible solution strategy given the smooth gradient of the landscape towards each global optimum, but no such algorithm exists. However, when the *gradient* for novelty is positively correlated with the gradient for *fitness*, then NS can exploit its exploration for novel individuals to find high quality solutions, indirectly climbing towards either of the optimal behaviors.

To illustrate the performance of NS with the accuracy descriptor, a synthetic classification problem is used. A random Gaussian mixture model is used to generate a two-class problem within the $\mathbb{R}^2$ plane, generating 200 random sample points for each class; following the strategy reported in [28, 17]. The graphical representation of the two class problem is shown in Figure 3. Table 2 presents the results of NS-GP and the baseline SRS-GPC method on the test data averaged over all thirty runs. Results indicate that both algorithms achieve similar results on this hard problem, with NS exhibiting a slightly better average classification error.

Table 1: Parameters for the GP-based search.

Parameter	**Description**
Population size	200 individuals.
Generations	200 generations.
Initialization	*Ramped Half-and-Half*, with 6 levels of maximum depth.
Operator probabilities	Crossover $p_c = 0.8$, Mutation $p_\mu = 0.2$.
Function set	$\{ +, -, \times, \div, \lvert\cdot\rvert, x^2, \sqrt{x}, log, sin, cos, IF \}$
Terminal set	$\{x_1, ..., x_i, ..., x_p\}$, where x_i is a dimension of the data patterns $\mathbf{x} \in \mathbb{R}^n$.
Bloat control	Dynamic depth control [24].
Initial dynamic depth	6 levels.
Hard maximum depth	20 levels.
Selection	Tournament.
Training/Testing partition	70%/30%.

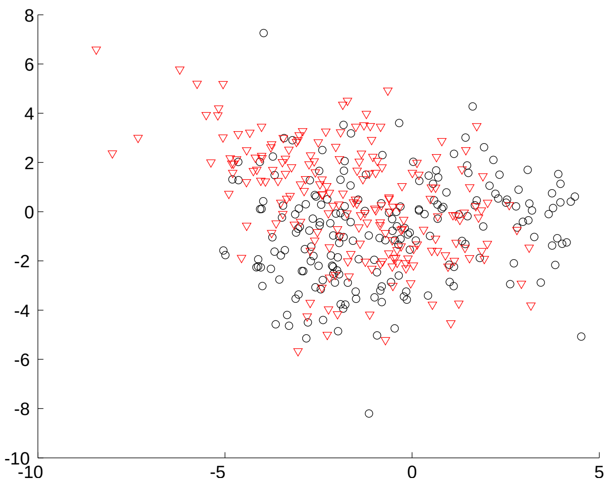


Figure 3: Synthetic 2-class problem for supervised classification.

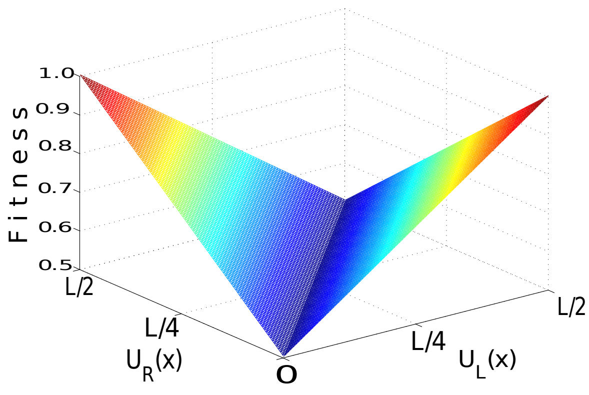


Figure 4: Fitness landscape in behavioral space for the cluster descriptor.

Table 2: Average classification error of the best solution found by each algorithm on the classification problem.

Algorithm	**SRS-GPC**	**NS-GP**
Performance	0.374	0.365

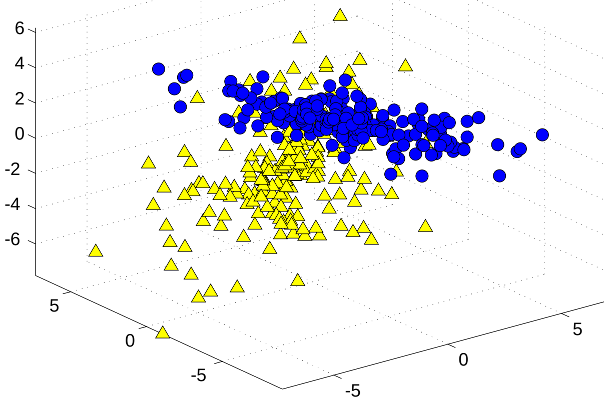
5.2.2 *Unsupervised clustering*

In clustering, a set of patterns is partitioned into disjoint groups, or clusters, such that patterns that belong to the same cluster are similar, and patterns from different clusters are dissimilar. The GP approach implemented here uses the same heuristic as the SRS classifier to assign cluster labels. Performance is measured using the class distance ratio (CDR), that compares the dispersion within the clusters to the gap between the clusters [7, 16].

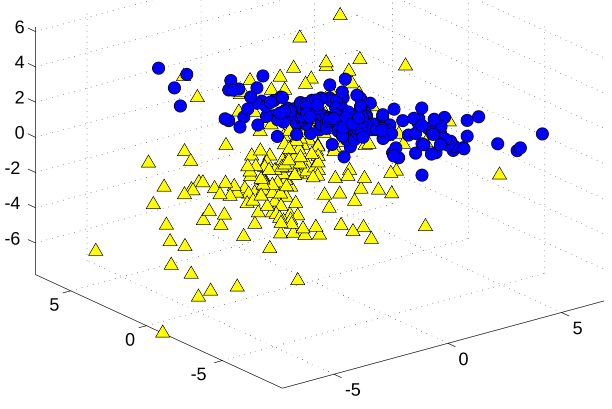
Unlike the supervised classification case, in a clustering problem no prior-knowledge of the correct grouping is available. Therefore, there is no training phase, since all data is test data; and a clustering algorithm must infer class membership based on the properties of the data itself. Hence, the accuracy descriptor cannot be used in this case, so a different descriptor is required [16].

Cluster Descriptor $\boldsymbol{\beta}^C$: For a problem where two clusters are assumed $\{\omega_1, \omega_2\}$ (most clustering algorithms require this information as input) the cluster descriptor is constructed in the following way [16]. If $\mathcal{T} = \{\mathbf{y}_1, \mathbf{y}_2, ..., \mathbf{y}_L\}$ represents the data samples, then the behavioral descriptor for each GP clustering function K_i is a binary vector $\boldsymbol{\beta}^{C_i} = (\beta_1, \beta_2, ..., \beta_L)$ of size L, where each vector element β_j is set to 1 if K_i assigns label ω_1 to pattern $\mathbf{y}_j$ and is set to 0 otherwise. Then, the cluster descriptor of the optimal clustering function K_O is given by $\boldsymbol{\beta}^{C_0} = (1_1, 1_2, ..., 1_{\frac{L}{2}}, 0_{\frac{L}{2}+1}, ..., 0_L)$. As in the accuracy descriptor case, an equally useful solution is to take the opposite (complement) behavior $\boldsymbol{\beta}^{C_*} = (0_1, 0_2, ..., 0_{\frac{L}{2}}, 1_{\frac{L}{2}+1}, ..., 1_L)$. Then, suppose that the number of samples from each cluster is $\frac{L}{2}$, and that the first $\frac{L}{2}$ elements in $\mathcal{T}$ have a ground truth label of ω_1. The fitness landscapes of $\boldsymbol{\beta}^C$ in behavioral space is depicted in Figure 4, where u_L measures the number of ones in the first $\frac{L}{2}$ bits of a behavior descriptor $\boldsymbol{\beta}^C$, and u_R does the same for the remaining $\frac{L}{2}$ bits; see [16] for a detailed explanation.

To illustrate the performance of NS with the cluster descriptor, a synthetic clustering problem is constructed using



(a) *Ground Truth*



(b) *NS-GP*

Figure 5: (a) Ground truth data; (b) clustering achieved by the NS-GP search.

the same strategy described above, except that the data samples are set in $\mathbb{R}^3$ [16]. The graphical representation of the two-cluster problem is shown in Figure 5a. In this case, the parameters of the GP search are the same as in Table 1 except that all of the data is used during evolution to construct the descriptor of each individual. Table 2 presents results of NS-GP compared with two common clustering methods, the k-means (KM) and Fuzzy C-means (FC) algorithms provided with Matlab toolboxes. The table uses the ground truth data to compute the clustering error for each algorithm, computed as the percentage of samples placed in the wrong cluster. Notice the strong performance achieved by behavior-based search on this problem. Finally, Figure 5b shows the clustering of an example run obtained of the NS-GP clustering.

6. CONCLUSIONS

This paper presents a behavior-based analysis of modal problems; problems that require different modes of operation when applied in different contexts. The main thesis is that the concept of problem modality can be clearly understood when behavioral space is explicitly considered. Literature and experimental work is presented as evidence that modal problems can be solved when the search incorporates a behavior-based analysis of GP individuals. Moreover, an extreme approach is considered, where behavioral diversity is the sole source of selective pressure during evolution, using the novelty search algorithm. First, evolutionary robotics is identified as a plausible domain where modal problems appear; current literature seems to support this claim, by illustrating that posing problems in behavioral space can be useful and sometimes required. Then, pattern recognition is considered as a second domain where modal problems are common; in particular data classification and clustering problems are considered, and a behavior descriptor is described for each. In pattern recognition, it seems that the requirement of different modes of program action is built into how these problems are posed. Experimental results show that behavior-based search can perform well in this domain, relative to control methods.

Table 3: Average classification error for each algorithm on the clustering problem.

Algorithm	**KM**	**FC**	**NS-GP**
Performance	0.360	0.355	0.335

A final note on lexicase selection approach proposed by Spector in [26] is also relevant. In that work, modality is studied in a symbolic regression problem, achieving encouraging initial results. An interesting question is to determine if lexicase selection is equivalent to the behavior-based search proposed here. While a complete answer is left as future work, an outline of a possible strategy to tackle it is given here. Consider the nature of a symbolic regression problem, where the goal is to fit program output to each fitness case. In this sense, a proper description of what the program does might be the output vector, what amounts to a semantics-based analysis of a program. The actions of a program might not be context dependent, and therefore its performance might be better described using great detail, an extreme within the conceptual scale of Figure 1. In fact, this is exactly how lexicase selection operates, by comparing solutions based on their raw output; also see [6]. If this is the case, it is possible to argue that while lexicase selection and behavior-based search focus on different details of program performance, the approaches are more similar to each other than each is to a global fitness-based selection. Therefore, it seems that selecting the proper level of analysis will surely depend upon the nature of the problem domain. However, future work should sharpen this notion, since recent results have shown that NS might also be applicable to symbolic regression [15]. Moreover, NS and lexicase selection seem to overlap in another way. NS promotes diversity above all else, achieving good results when an evolutionary road that leads towards novelty also leads towards quality; this is particularly the case when a problem is difficult and random individuals perform poorly [10, 15, 16, 17]. On the other hand, lexicase selection is biased towards solutions that perform well on fitness cases where the rest of the population performs poorly. In a sense, lexicase selection seems to combine diversity preservation and fitness based selection. For now, the evidence provided in this work should help strengthen the concept of problem modality within the GP community and allow us to explore novel solution strategies.

Acknowledgments

Funding for this work provided by CONACYT (Mexico) Basic Science Research Project No. 178323 and DGEST (Mexico) Research Project No. TIJ-ING-2012-110. This material is based upon work supported by the National Science Foundation under Grant No. 1017817. Authors E. Naredo and Y. Martínez supported by CONACYT scholarships, respectively No. 232288 and No. 226981.

References

[1] R. A. Brooks. *Cambrian intelligence: the early history of the new AI*. MIT Press, Cambridge, MA, USA, 1999.

[2] J. Clune, B. E. Beckmann, C. Ofria, and R. T. Pennock. Evolving coordinated quadruped gaits with the hyperneat generative encoding. In *Proceedings of the 11th Congress on Evolutionary Computation*, CEC'09, pages 2764–2771, IEEE Press, 2009.

[3] C. A. C. Coello, G. B. Lamont, and D. A. V. Veldhuizen. *Evolutionary Algorithms for Solving Multi-Objective Problems (Genetic and Evolutionary Computation)*. Springer-Verlag New York, Inc., Secaucus, NJ, USA, 2006.

[4] E. Dunn, G. Olague, and E. Lutton. Parisian camera placement for vision metrology. *Pattern Recogn. Lett.*, 27(11):1209–1219, 2006.

[5] S. Forrest, T. Nguyen, W. Weimer, and C. Le Goues. A genetic programming approach to automated software repair. In *Proceedings of the 11th annual Genetic and evolutionary computation*, GECCO '09, pages 947–954, ACM. 2009.

[6] T. Helmuth and L. Spector. Evolving a Digital Multiplier with the PushGP Genetic Programming System. To appear in *Proceeding from the 15th annual Genetic and evolutionary computation conference companion*, GECCO Companion '13, ACM, 2013.

[7] T. K. Ho and M. Basu. Complexity measures of supervised classification problems. *IEEE Trans. Pattern Anal. Mach. Intell.*, 24(3):289–300, 2002.

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

[9] J. Lehman and K. O. Stanley. Efficiently evolving programs through the search for novelty. In *Proceedings of the 12th annual Genetic and evolutionary computation*, GECCO '10, pages 837–844, ACM, 2010.

[10] J. Lehman and K. O. Stanley. Abandoning objectives: Evolution through the search for novelty alone. *Evol. Comput.*, 19(2):189–223, 2011.

[11] K. McClymont, D. Walker, and M. Dupenois. The lay of the land: a brief survey of problem understanding. In *Proceedings of the 14th annual Genetic and evolutionary computation conference*, GECCO Companion '12, pages 425–432, ACM, 2012.

[12] N. F. McPhee, B. Ohs, and T. Hutchison. Semantic building blocks in genetic programming. In *Proceedings of the 11th European conference on Genetic programming*, EuroGP'08, pages 134–145, Berlin, Heidelberg, 2008. Springer-Verlag.

[13] A. Moraglio, K. Krawiec, and C. G. Johnson. Geometric semantic genetic programming. In *Proceedings of the 12th international conference on Parallel Problem Solving from Nature - Volume Part I*, PPSN'12, pages 21–31, Berlin, Heidelberg, 2012. Springer-Verlag.

[14] J. B. Mouret and S. Doncieux. Encouraging behavioral diversity in evolutionary robotics: An empirical study. *Evol. Comput.*, 20(1):91–133, 2012.

[15] E. Naredo, Y. Martínez and L. Trujillo. Searching for novel regression functions. To appear in *Proceedings of the 15th Congress on Evolutionary Computation*, CEC'13, IEEE Press, 2013.

[16] E. Naredo and L. Trujillo. Searching for novel clustering programs. To appear in *Proceeding from the 15th annual Genetic and evolutionary computation conference*, GECCO '13, ACM, 2013.

[17] E. Naredo, L. Trujillo, and Y. Martínez. Searching for novel classifiers. In *Proceedings from the 16th European Conference on Genetic Programming, EuroGP 2013*, volume 7831 of *LNCS*, pages 145–156. Springer-Verlag, 2013.

[18] A. L. Nelson, G. J. Barlow, and L. Doitsidis. Fitness functions in evolutionary robotics: A survey and analysis. *Robot. Auton. Syst.*, 57(4):345–370, 2009.

[19] A. L. Nelson and E. Grant. Using direct competition to select for competent controllers in evolutionary robotics. *Robotics and Autonomous Systems*, 54(10):840–857, 2006.

[20] S. Nolfi and D. Floreano. *Evolutionary Robotics: The Biology,Intelligence,and Technology*. MIT Press, Cambridge, MA, USA, 2000.

[21] R. Pfeifer, M. Lungarella, and F. Iida. The challenges ahead for bio-inspired 'soft' robotics. *Commun. ACM*, 55(11):76–87, 2012.

[22] J. Rosca. *Hierarchical learning with procedural abstraction mechanisms*. PhD thesis, University of Rochester, Rochester, NY, USA, 1997.

[23] S. Silva and J. Almeida. Gplab–a genetic programming toolbox for matlab. In L. Gregersen, editor, *Proceedings of the Nordic MATLAB conference*, pages 273–278, 2003.

[24] S. Silva and E. Costa. Dynamic limits for bloat control in genetic programming and a review of past and current bloat theories. *Genetic Programming and Evolvable Machines*, 10(2):141–179, 2009.

[25] M. Soleymani, J. Lichtenauer, T. Pun, and M. Pantic. A multimodal database for affect recognition and implicit tagging. *IEEE Trans. Affect. Comput.*, 3(1):42–55, 2012.

[26] L. Spector. Assessment of problem modality by differential performance of lexicase selection in genetic programming: a preliminary report. In *Proceedings of the 14th annual Genetic and evolutionary computation conference companion*, GECCO Companion '12, pages 401–408, ACM, 2012.

[27] L. Trujillo, G. Olague, E. Lutton, and F. Fernández de Vega Discovering several robot behaviors through speciation. In *Proceedings of the 2008 conference on Applications of evolutionary computing*, Evo'08, pages 164–174. Springer-Verlag, 2008.

[28] L. Trujillo, G. Olague, E. Lutton, F. Fernández de Vega, L. Dozal, and E. Clemente. Speciation in behavioral space for evolutionary robotics. *Journal of Intelligent & Robotic Systems*, 64(3-4):323–351, 2011.

[29] N. Q. Uy, N. X. Hoai, M. O'Neill, R. I. Mckay, and E. Galván-López. Semantically-based crossover in genetic programming: application to real-valued symbolic regression. *Genetic Programming and Evolvable Machines*, 12(2):91–119, 2011.

[30] S. Verel. Fitness landscapes and graphs: multimodularity, ruggedness and neutrality. In *Proceedings of the 11th annual Genetic and evolutionary computation conference: Late Breaking Papers*, GECCO '09, pages 3593–3656, ACM, 2009.

[31] S. X. Wu and W. Banzhaf. Rethinking multilevel selection in genetic programming. In *Proceedings of the 13th annual Genetic and evolutionary computation conference*, GECCO '11, pages 1403–1410, ACM, 2009.

[32] M. Zhang and W. Smart. Using gaussian distribution to construct fitness functions in genetic programming for multiclass object classification. *Pattern Recogn. Lett.*, 27(11):1266–1274, 2006.