Searching for Novel Clustering Programs

Enrique Naredo
Leonardo Trujillo*
Doctorado en Ciencias de la Ingeniería
Instituto Tecnológico de Tijuana
Tijuana, B.C., México
enriquenaredo@gmail.com
leonardo.trujillo@tectijuana.edu.mx

ABSTRACT

Novelty search (NS) is an open-ended evolutionary algorithm that eliminates the need for an explicit objective function. Instead, NS focuses selective pressure on the search for novel solutions. NS has produced intriguing results in specialized domains, but has not been applied in most machine learning areas. The key component of NS is that each individual is described by the behavior it exhibits, and this description is used to determine how novel each individual is with respect to what the search has produced thus far. However, describing individuals in behavioral space is not trivial, and care must be taken to properly define a descriptor for a particular domain. This paper applies NS to a mainstream pattern analysis area: data clustering. To do so, a descriptor of clustering performance is proposed and tested on several problems, and compared with two control methods, Fuzzy C-means and K-means. Results show that NS can effectively be applied to data clustering in some circumstances. NS performance is quite poor on simple or easy problems, achieving basically random performance. Conversely, as the problems get harder NS performs better, and outperforming the control methods. It seems that the search space exploration induced by NS is fully exploited only when generating good solutions is more challenging.

Categories and Subject Descriptors

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

General Terms

Performance,Theory,Experimentation

Keywords

Novelty Search, Genetic Programming, Clustering

*Corresponding Author

GECCO'13, July 6–10, 2013, Amsterdam, The Netherlands.

1. INTRODUCTION

Natural evolution, as an open-ended process, is different from the conventional engineering approach towards search and optimization. Nevertheless, evolutionary algorithms (EAs) are abstractions of Neo-Darwinian evolution that are guided by an objective function and mostly lack the open-ended feature of their natural inspiration. Nonetheless, many EAs have distinguished themselves as innovative search techniques that frequently solve difficult problems in diverse domains [10]. Moreover, since the search is guided by fitness, most of the traditional algorithms tend to converge on local optima unless proper heuristic methods are integrated into the search for diversity preservation.

On the other hand, it is illustrative to consider that some of the earliest EAs were open-ended techniques [4]. Recently, other EAs have been developed that share this feature, but are mostly aimed at specialized domains, such as artificial life environments [19] and interactive search [9]. In particular, this paper studies the algorithm proposed by Lehman and Stanley called novelty search (NS) [11, 12, 13], an EA that abandons an explicit fitness function. Instead, NS focuses the selective pressure on finding unique solutions; i.e., individuals that introduce novel information into the search process with respect to the current search progress.

The core requirement in NS is that instead of relying on the genotype, phenotype or fitness to describe individuals, they are described based on their behavior. Therefore, a domain dependent behavior descriptor must be proposed to apply NS successfully. Most of the published works on NS have been in evolutionary robotics (ER), where the concept of robot behavior has a strong history in the field [3, 18, 23, 24, 16]. Conversely, the concept of behaviors is not common in other domains. In GP, for instance, resent research has incorporated the concept of semantics [15, 25]; however, behaviors are a broader concept, incorporating contextual information, as discussed in the following section.

Therefore, the goal of this paper is to present an application of NS on a common problem in machine learning and pattern recognition. In particular, unsupervised data clustering is solved with a NS-based GP system, using a domain specific behavioral descriptor. This work is an extension of recent research where NS was applied to supervised data classification [17]. Results are encouraging, the NS algorithm is indeed capable of solving clustering problems and in some cases outperforming standard clustering techniques. In particular, NS seems better suited for diffi-

cult cases, where the explorative abilities of the algorithm can be fully leveraged.

The remainder of this paper is organized as follows. Section 2 describes the concept of behavioral space and the NS algorithm. In Section 3 the clustering problem is defined and related work is discussed. Afterwards, Section 4 presents the proposed NS-based GP algorithm for data clustering and the proposed behavioral descriptor for evolved clustering programs. Then, Section 5 presents the experiments and an analysis of the results. Finally, a summary of the paper and concluding remarks are given in Section 6.

2. BEHAVIOR-BASED SEARCH

It is well understood that an EA concurrently samples three different spaces during a search. First, genotypic space, which corresponds to the encoding used to represent each valid solution. Second, phenotypic space, which represents the problem space where solutions are expressed. Finally, objective or fitness space, that corresponds to the space of performance criteria. In some cases these spaces are clearly distinct, but for some representations it is not [6].

While these three spaces have been the focus of most research, other spaces have recently been considered within the field. First consider that fitness gives a coarse and global evaluation of an individual's performance, averaging out differences in program quality on different fitness cases. Considering this, some researchers have proposed a finer grained approach to analyze the performance of each individual.

One approach is known as semantics in GP literature [15, 25], with its corresponding semantic space, which corresponds to the space of possible program outputs. Semantics is an important concept in GP because many genetically different programs can share the same semantic representation. However, only analyzing program outputs might not be the best approach to analyze performance in some domains. For instance, consider ER problems, where evolutionary algorithms are used to search for robust controllers of autonomous robots [18]. The goal of the ER approach, is to find high quality solutions introducing as little prior knowledge as possible into the objective function. In this scenario, the correspondence between program inputs, outputs and induced actions is much less clear. 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 (simulated) world can produce a non-injective and non-surjective relation between program output and the actions performed by a robot.

Therefore, another approach towards describing performance in ER research is to focus on the behavioral space of a problem [23, 16]. The concept of behaviors in robotics dates back to the seminal works of Brooks from the 1980's [3]. A behavior is a description β of the way an agent K (robot in ER and program in the GP case) acts in response to a particular stimulus within a particular context $\mathcal{C}$. A context $\mathcal{C}$ includes the internal description the agent has of its environment and its own internal state [1]. Stated in another way, a behavior β is produced by the interactions of an agent K, the output $\mathbf{y}$ and a context $\mathcal{C}$. In behavior-based robotics, for instance, behaviors are 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[16]. The justification for this is evident, given that the objective function is stated as a high-level goal, then population management should take into account the behavioral aspect of the solutions.

A behavior, defined in this way, is a higher-level description of the semantics of a program. An individual's behavior is described in more general terms, accounting not only for the program output but the context in which the output was produced. If contextual information is not relevant, then semantics and behaviors could be regarded as equivalent. Therefore, in essence, fitness, program semantics, and behavior provide different levels of abstraction of a program's performance. At one extreme of this scale of analysis, fitness provides a coarse grained look at performance, a single value (for each criteria) that attempts to provide a global evaluation. At the other end of the analysis scale, semantics describe program performance in great detail. On the other hand, behavioral descriptors lie in between fitness and semantics, providing either a finer or coarser level of description, depending on how behaviors are meaningfully characterized within a particular domain.

2.1 Novelty Search

Lehman and Stanley proposed NS, an EA that eliminates the need for an explicit objective function [11, 12, 13]. Evolution is not guided by a measure of *quality*, but by the a measure of *uniqueness*. To measure the novelty an individual introduces into the evolutionary process, the NS algorithm describes each individual within behavioral space.

A limitation of most EAs is their tendency to converge on local optima, a common result in many real-world problems with multimodal and irregular fitness landscapes. Within EC, diversity preservation is usually incorporated into an EA to overcome this shortcoming. However, most proposals can be regarded as ad-hoc solutions that must continuously attempt to balance exploration and exploitation during the search. Conversely, through the search for novelty alone, diversity preservation introduces the sole selective pressure during evolution. The core assumption behind NS, that suggests why it can find high-fitness solutions, is that at the beginning of a search, most random solutions will exhibit behaviors with bad fitness values. Therefore, as NS progresses and it leads the search towards novel regions in the search space, this should also lead towards better behaviors and thus better fitness values; i.e., when initial random solutions have low performance then the search for novelty could lead towards quality during evolution.

In summary, instead of using fitness to guide the search, NS uses a measure of *novelty* to characterize each individual. More precisely, a sparseness measure establishes how novel an individual is within behavioral space, with respect to the current population and novel solutions generated in previous generations. Such a measure of novelty is dynamic; i.e., it can produce different results for the same individual depending on the population state and search progress. In NS, 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, is given by

[1] McPhee et al. [15] describe context as the root node of a tree. This might be appropriate for some problems, but here context is given by external conditions of the problem.

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

where $\boldsymbol{\alpha}_i$ is the behavioral descriptor of the ith-nearest neighbor of K in behavioral space with respect to distance measure *dist*, a domain-dependent measure that depends on how descriptors are expressed. 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. Fitness is only considered at the end of the search, when the individual with the best fitness score is selected as the final solution produced by the NS algorithm.

An important aspect is to determine which individuals should be considered when sparseness in computed, since the population changes with every generation, and the sparseness value for the same individual can vary over time. In NS, the proposal is to consider individuals in the current population as well as individuals that were considered to be novel in previous generations; hence, an individual has an intra and inter-generational neighborhood. Therefore, individuals with a sparseness value above a minimal threshold ρ_{min}, the second parameter of NS, are added to a population archive. The archive stores novel individuals, and the individuals in the archive are used along with the population to compute the sparseness value. An advantage of using the archive is that it can mitigate backtracking during the search, serving as a memory of the search progress.

Since its original proposal [11], and in later works [12, 13], most applications of NS have focused on ER, such as navigation [11, 12, 13], morphology design [14] and gait control [13]. As stated before, search algorithms that explicitly contemplate behaviors seem well suited for robotics, since most high-level robotic tasks can usually be solved in structurally different ways, thus guaranteeing a multimodal search.

In spite of these good results, NS has still not been used in most other domains. One exception is found in [26], where NS is integrated with an interactive evolutionary system, combining the strengths of both strategies. Another recent example is found in [17], where NS is used for supervised classification. However, in general applying NS to mainstream problems, to the authors' knowledge, is not yet commonly done. Thus, the present work proposes the use of NS for a common machine learning problem: data clustering, extending the proposal in [17] to a more difficult domain.

3. CLUSTERING WITH GP

Data classification is one of the most common tasks in machine learning. Posed as a learning problem, classification can be either supervised or unsupervised. Unsupervised classification is also referred to as data clustering, where the goal is to organize a set of unlabeled elements into distinct groups using little or no prior knowledge [27].

Clustering has been studied in various domains, such as computer vision and web development [8]. Given the variety of areas where clustering is applied, a multitude of clustering methods have been developed. Most of the clustering techniques are objective driven, such as the well known K-means (KM) that uses a hard partitioning of feature space [27] or the the Fuzzy C-means (FC) algorithm that instead uses a soft partitioning [1].

It is commonly expected that a cluster must present internal homogeneity and external separation from other clusters; i.e., patterns in the same cluster should be similar to each other, while patterns from different clusters should be different (in some sense). Generally in a clustering problem is given a set of patterns $\mathcal{T} = \{\mathbf{x}_1, ..., \mathbf{x}_j, .., \mathbf{x}_m\}$, where each pattern is described by a vector of features $\mathbf{x}_j = (x_{j,1}, x_{j,2}, ..., x_{j,n}) \in \mathcal{R}^n$. Then, a clustering algorithm must assign each pattern to a unique cluster; where the number of clusters is normally set a priori.

GP has been used extensively for data classification, see for instance [28]. In particular, several GP-based clustering systems have been proposed, using grammar based evolution [5], theoretic probabilistic interpretations [2] and hybrid multiobjective algorithms [21]. However, one underlying characteristic that previous methods share, is that they all rely on an explicit fitness function, just like any evolutionary search. Instead, in this paper fitness is substituted by a behavior-based search based on the novelty criteria. Nevertheless, a measure of clustering performance is presented next, since it is required to compare different results and to select the final solution returned by the NS algorithm.

3.1 Cluster Distance Ratio

The cluster distance ration (CDR) compares the within cluster dispersion with the gap between clusters [7]. Here, only problems with two clusters will be considered. The intra-cluster dispersion is computed using the Euclidean distance of each pattern $\mathbf{x} \in \mathbb{R}^n$ to its nearest neighbor within the cluster. The average of this distance, for elements in the data set, measures the intra-cluster dispersion, given by

$$IntraCluster = \frac{1}{N}\sum_{i=1}^{K}\sum_{x \in C_i} ||\mathbf{x} - \mathbf{u}_i|| \ , \quad (2)$$

where N is the number of data elements, K is the number of clusters, and $\mathbf{u}_i$ is the nearest neighbor within the same cluster of $\mathbf{x}$. Then, the inter-cluster separation is computed similarly, but now considering the closest neighbor from the other cluster, given by

$$InterCluster = \frac{1}{N}\sum_{i=1}^{K}\sum_{x \in C_i} ||\mathbf{x} - \mathbf{v}_i|| \ , \quad (3)$$

where $\mathbf{v}_i$ is the closest neighbor of $\mathbf{x}$ from the other cluster. Finally, the CDR is defined as

$$CDR = \frac{IntraCluster}{InterCluster} \ . \quad (4)$$

The CDR value provides a good measure of clustering quality, where the goal is to achieve the highest value.

4. CLUSTERING WITH NS-GP

The proposal of this paper is to develop a NS-based GP system for unsupervised data clustering, hereafter referred to as NS-GP. The algorithm is a modification of the GP classifier that uses a static range selection [28] that functions as follows. Let us consider two clusters ω_1 and ω_2, a set of patterns $\mathbf{x} \in \mathbb{R}^n$, and a GP functions (programs) $K : \mathbb{R}^n \mapsto \mathbb{R}$. Then, data pattern $\mathbf{x}$ is assigned to cluster ω_1 if $K(\mathbf{x}) > 0$ and it is assigned to cluster ω_2 otherwise. To apply NS to clustering, a behavioral descriptor for each program K is required; the proposed descriptor is presented below.

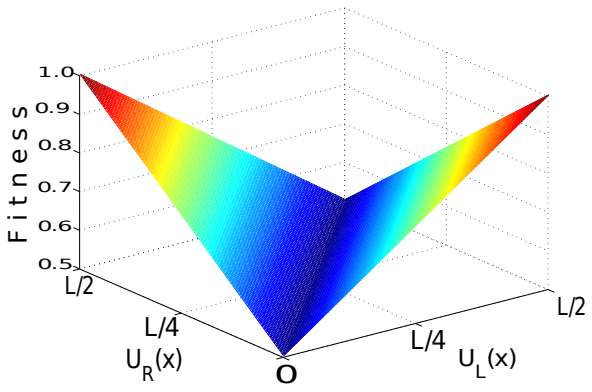


Figure 1: Fitness landscape in behavioral space for the proposed clustering descriptor.

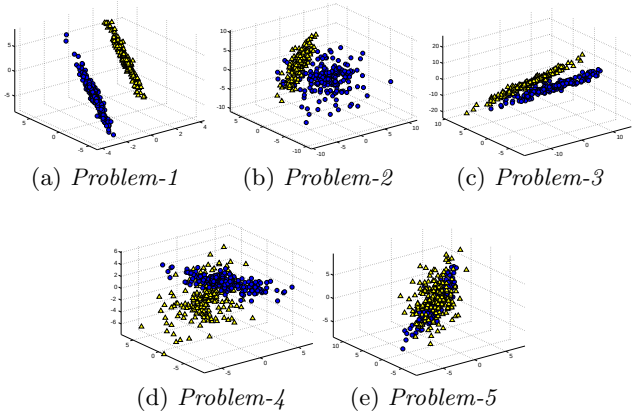


Figure 2: Five synthetic clustering problems; the observed clusters represent the ground truth data.

4.1 Clustering Descriptor

For a clustering problem a training set is not given since it is unsupervised, so the entire dataset $\mathcal{T}$ is treated as testing data, which contains sample patterns from each cluster. For a two-cluster problem with data set $\mathcal{T} = \{\mathbf{y}_1, \mathbf{y}_2, ..., \mathbf{y}_L\}$, and clusters ω_1 and ω_2, the cluster descriptor $\boldsymbol{\beta}^C$ is proposed, where for each GP clustering program K_i the descriptor 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 clustering function K_i assigns label ω_1 to pattern $\mathbf{y}_j$ and is set to 0 otherwise.

4.2 Behavioral Landscape

For a better explanation lets consider a synthetic problem where a ground truth clustering is provided. Then, suppose that the number of samples from each cluster is $\frac{L}{2}$, and that they are ordered in such a way that the first $\frac{L}{2}$ elements in $\mathcal{T}$ have a ground truth label of ω_1. Let $\mathbf{x}$ represents a binary vector, and function $u(\mathbf{x})$ returns the number of 1s in $\mathbf{x}$. Moreover, let K_O be the *optimal* clustering program that achieves a perfect accuracy on the data set based on the ground truth labeling. Then, the descriptor of K_O is given by $\boldsymbol{\beta}^{C_0} = (1_1, 1_2, ..., 1_{\frac{L}{2}}, 0_{\frac{L}{2}+1}, ..., 0_L)$. Moreover, for a two-cluster problem, an equally useful solution is to take the opposite (complement) behavior and invert the clustering, such that a 1 is converted to a 0 and vice-versa. This mirror behavior is defined by the descriptor $\boldsymbol{\beta}^{C_*} = (0_1, 0_2, ...0_{\frac{L}{2}}, 1_{\frac{L}{2}+1}, ..., 1_L)$. The complete fitness landscape in behavioral space, with fitness given by clustering accuracy, is depicted in Figure 1.

For a two-cluster problem with a reasonable degree of difficulty, the initial generations of a GP search should be expected to contain random clustering programs. For the cluster descriptor, behavioral space is organized on a two dimensional surface, such that one axis u_L considers the number of ones on the left hand side, first $\frac{L}{2}$ bits, of a behavior descriptor $\boldsymbol{\beta}^C$, and the other axis u_R considers the remaining $\frac{L}{2}$ bits; see Figure 1. Notice that the middle valley of the fitness landscape corresponds to basically random clustering functions, with worst case scenario performance. Hence, NS will push the search towards either of the two global optima, $\boldsymbol{\beta}^{C_0}$ and $\boldsymbol{\beta}^{C_*}$ Finally, given the above binary descriptor, a natural $dist()$ function for Equation 1 is the Hamming distance, which counts the number of bits that differ between two binary sequences.

Table 1: Parameters for the NS-GP.

Parameter	Description
Population size	200 individuals.
Generations	100 generations.
Initialization	*Ramped Half-and-Half*, with 6 levels of max. 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	$\{x_1, ..., x_i, ..., x_n\}$,where x_i is a dimension of the data patterns $\mathbf{x} \in \mathbb{R}^n$.
Bloat control	Dynamic depth control.
Initial dynamic depth	6 levels.
Hard maximum depth	20 levels.
Selection	Tournament.
Archive	No Limit on archive size.

5. EXPERIMENTS

The performance of the NS-GP clustering algorithm is compared against two well known clustering methods; K-means (KM) and Fuzzy C-means (FC). The NS-GP is tested in two different versions, each with a different behavioral threshold ρ_{min}: NS-GP-20 with $\rho_{min} = 20$ and NS-GP-40 with $\rho_{min} = 40$; in both cases parameter the number of neighbors k is set to 15 to compute sparseness.

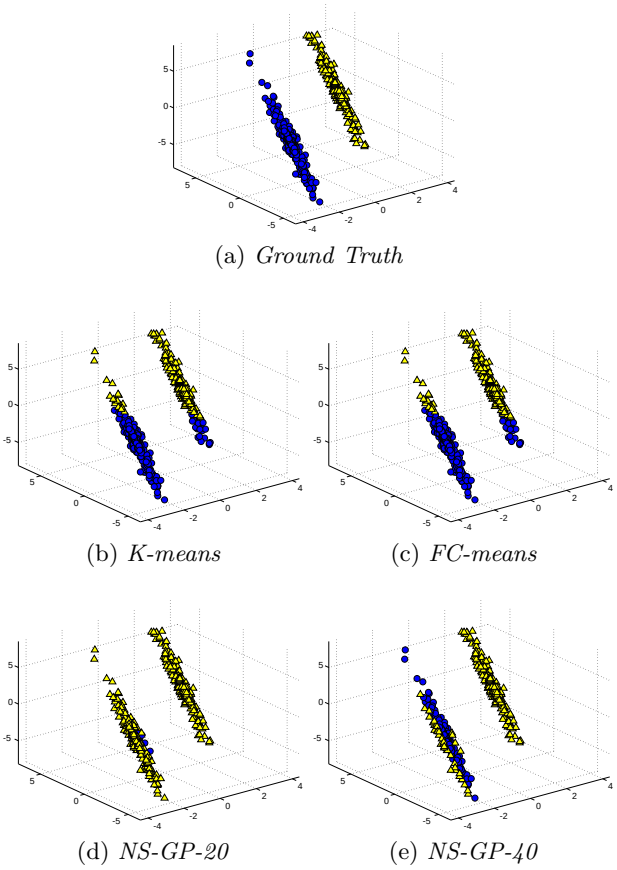
5.1 Problems and GP Parameters

Synthetic clustering problems were generated using two randomly generated Gaussian functions in $\mathbb{R}^3$ for each problem, following a similar strategy to [22]. From each function, 160 data points were randomly generated. Then, an additional 40 data points were generated by triplicating the standard deviation of each, to get disperse clusters with some outliers. Table 2 shows the five selected test problems of increased difficulty, where each figure represents the ground truth data.

The NS-GP algorithms use a Koza style GP with subtree crossover and subtree mutation. Table 1 summarizes the

Table 2: A comparison of all of the clustering methods on each problem. The data shows the CDR values and the clustering error score; for NS these values are averages over all runs.

Problem	Clustering Distance Ratio					Clustering Error			
	GT	KM	FC	NS-GP-20	NS-GP-40	KM	FC	NS-GP-20	NS-GP-40
Problem-1	9.17	10.22	10.25	9.20	9.21	0.10	0.10	0.45	0.46
Problem-2	4.46	5.51	5.46	5.38	4.47	0.07	0.07	0.24	0.25
Problem-3	5.55	5.37	5.29	4.51	4.53	0.30	0.26	0.20	0.22
Problem-4	4.10	6.21	6.11	5.51	5.45	0.36	0.35	0.10	0.19
Problem-5	1.59	4.54	4.52	3.64	3.22	0.45	0.45	0.19	0.27



(a) *Ground Truth*

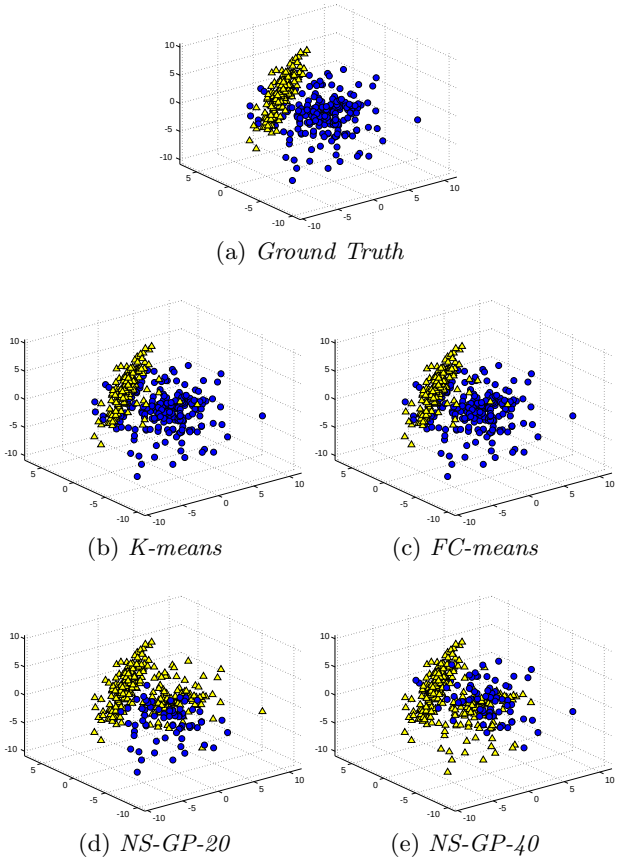
(b) *K-means*

(c) *FC-means*

(d) *NS-GP-20*

(e) *NS-GP-40*

Figure 3: Comparison of clustering performance on Problem No.1.



(a) *Ground Truth*

(b) *K-means*

(c) *FC-means*

(d) *NS-GP-20*

(e) *NS-GP-40*

Figure 4: Comparison of clustering performance on Problem No.2.

parameters of the algorithm, which was coded using Matlab 2012b and the GPLAB toolbox [20].

5.2 Results

This section presents an experimental evaluation of NS-GP clustering. All of the clustering algorithms were executed 30 times to find statistically significant results. Table 2 compare the performance of NS-GP with two baseline methods, KM and FC. The table presents two comparative views of average performance over all runs. First, the algorithms are compared based on their CDR score, and the CDR of the ground truth of each problem is also presented. Additionally, using the ground truth, a classification error was computed, based on the ordering suggested by each clustering method. In general, the results indicate two noteworthy trends. First, NS performs much worse on the simpler problems, it seems like it is basically doing a random search. On the other hand, NS noticeably outperforms the control methods on the harder problems, this is especially true for the hardest, Problem 5. Second, it seems that a lower ρ_{min} encourages better performance in most cases. A detailed view of how the data is being clustered can provide a different analysis of the results. Figures 3 - 7 present a graphical illustration of the clustering output achieved by each method. All figures show the ground truth clusters for visual comparison, along with a typical clustering output from each method. These figures confirm the data presented in Table 2, NS-GP performs worse on the easy problems and performs better on the difficult ones.

Figures 8 and 9 examine how sparseness evolves during the NS-GP search, for NS-GP-20 and NS-GP-40 respec-

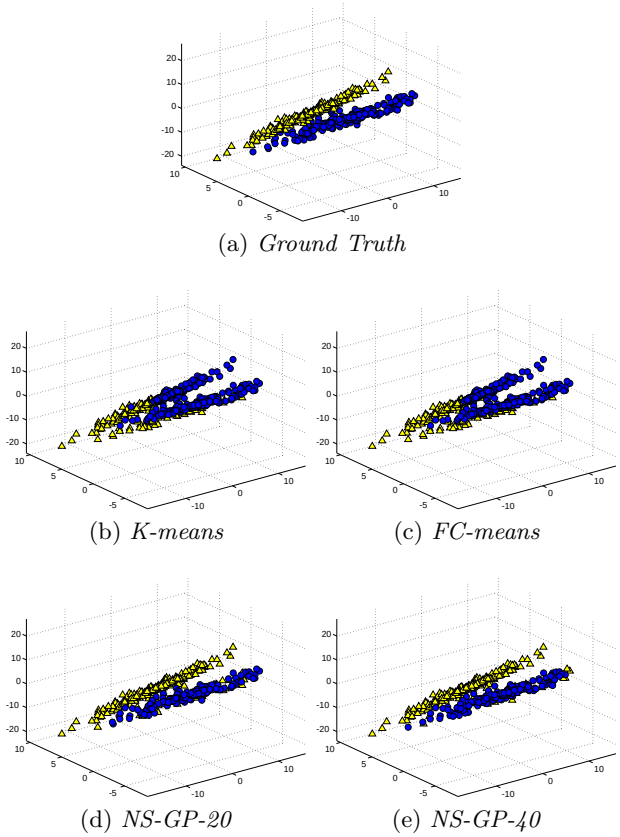


Figure 5: Comparison of clustering performance on Problem No.3.

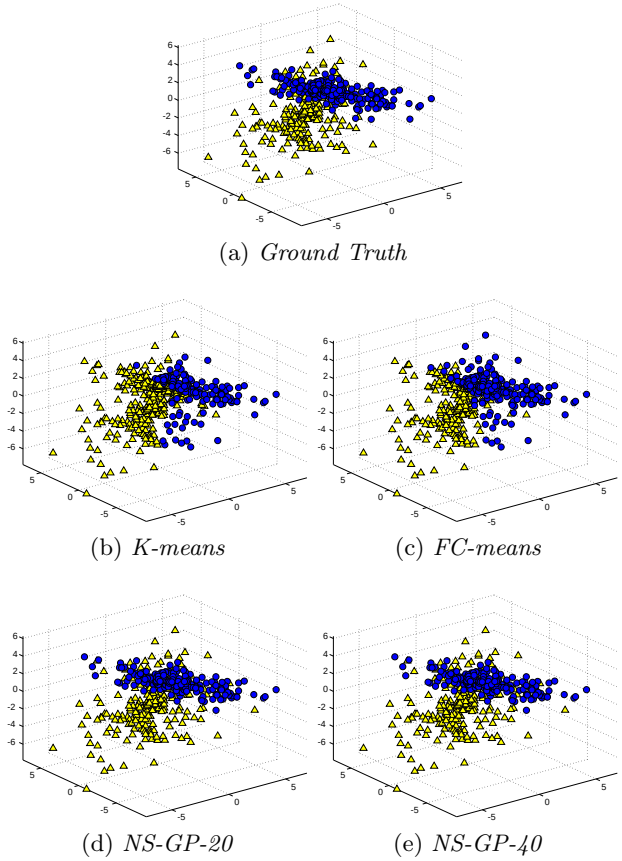


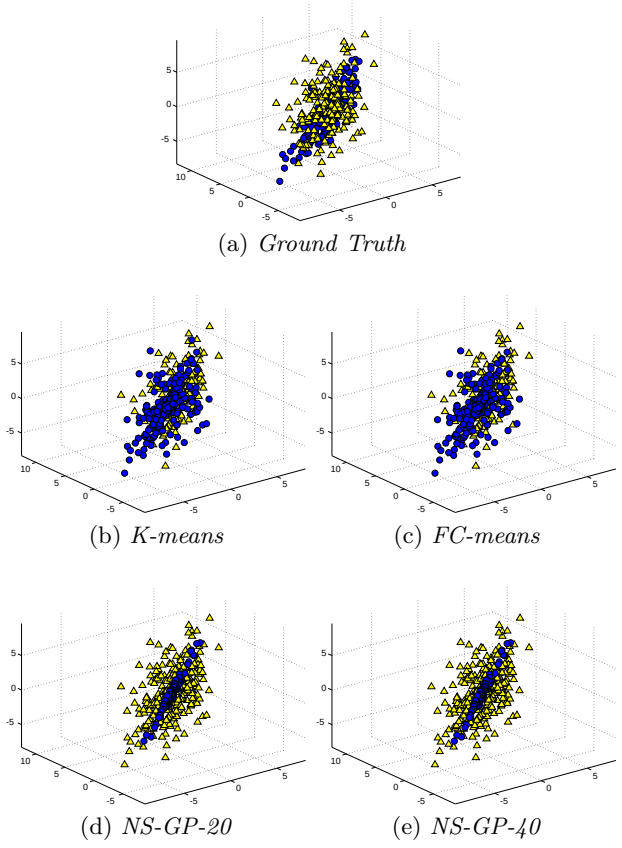
Figure 6: Comparison of clustering performance on Problem No.4.

tively. Each figure presents a similar plot that shows how the sparseness of the best individual (based on fitness) evolves over each generation. The plots are averages of the 30 runs of each experiment and present a curve for each problem. A horizontal line shows the corresponding threshold value for each configuration, set to 20 in Figure 8 and set to 40 in Figure 9. In both cases it is possible to observe a similar pattern. For the easy problems (1, 2 and 3) the sparseness value of the best individual at each generation does not reach the threshold, and is therefore not included in the population archive. This means that the individual is lost, and maybe it is never found again. This explains the reason why NS fails to produce good results on the easy problems. It appears that since the problems are not difficult, then novelty does not necessarily lead to quality, and NS is not influencing the search as desired. On the other hand, for the more difficult problems (4 and 5) it is clear that good solutions almost always correspond with novel solutions; i.e., the solutions that are being introduced into the archive of novel solutions also exhibit good fitness values. In these cases, the search for novelty is indeed guiding evolution towards solutions that perform better. This is reasonable, since for difficult problems the initial (random) programs will mostly exhibit bad performance, and novelty can lead towards quality.

Finally, to illustrate how evolution progresses with the NS-GP clustering algorithm, Figure 10 presents four snapshots of NS-GP-40 applied to Problem 5. Each plot shown in the figure represents the best NS solution (taken from the current population and population archive) based on the CDR value at four different generations. For this difficult problem, it is clear that NS is progressively exploring the search space, and finding better solutions.

6. CONCLUSIONS

This work presents a NS-based GP algorithm for automatic data clustering. To the authors knowledge, the work represents the first attempt to leverage NS to solve this common problem in pattern recognition and machine learning, since previous applications of NS were primarily focused on robotics. This paper develops the idea of program behaviors and how they relate to semantics, and suggests that behavior based search, such as the NS algorithm, can be applicable to many problem domains. For the clustering problem addressed here, a domain-specific behavioral descriptor was proposed and its fitness landscape was analyzed. The NS-GP algorithm was also compared with two standard clustering methods, K-means and Fuzzy C-means. Initial results are informative and encouraging. The experiments suggest that NS is not well suited to solve easy problems, failing drastically on the examples given here. However, this is not discouraging for researchers from real-world domains, easy problems such as these rarely come up. Conversely, the NS-based GP exhibits very good results on the difficult test cases, outperforming the control methods. It seems that for easy problems, the exploration capacity of NS is mostly



(a) *Ground Truth*

(b) *K-means* (c) *FC-means*

(d) *NS-GP-20* (e) *NS-GP-40*

Figure 7: Comparison of clustering performance on Problem No.5.

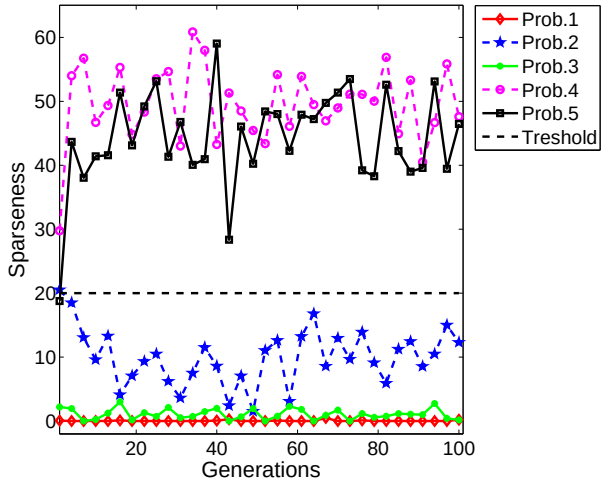


Figure 8: Evolution of sparseness for NS-GP-20, showing the average sparseness of he best individual at each generation.

unexploited or maybe even limits the search; i.e., if random solutions have a high fitness then novelty could easily lead the search towards worse results. On the other hand, since randomly generating a high-performance solution is less probable for difficult problems, then the incentive for behavioral exploration is incremented and the search for novelty can indeed lead towards quality during evolution. Future work will center on exploring further experiments in this domain, comprehensively testing different parametrizations of the NS-GP search, evaluating performance on real problems, and comparing the algorithm with other GP systems for data clustering.

Acknowledgments.

This research is funded by CONACYT (Mexico) Basic Science Research Grant No. 178323; and the first author supported by doctoral scholarship No. 232288 from CONACYT.

7. REFERENCES

[1] J. Bezdek, R. Ehrlich, and W. Full. Using direct competition to select for competent controllers in evolutionary robotics. *Fcm: The fuzzy c-means clustering algorithm*, 10(2–3):191–203, 1984.

[2] N. Boric and P. Estevez. Genetic programming-based clustering using an information theoretic fitness measure. In *Proceedings of the 2007 IEEE Congress on Evolutionary Computation*, pages 31–38. IEEE Press, 2007.

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

[4] R. Dawkins. *Climbing Mount Improbable*. W.W. Norton & Company, 1996.

[5] I. De Falco, E. Tarantino, A. D. Cioppa, and F. Fontanella. A novel grammar-based genetic programming approach to clustering. In *Proceedings of the 2005 ACM symposium on Applied computing*, SAC '05, pages 928–932, New York, NY, USA, 2005. ACM.

[6] E. Galván-López, J. Mcdermott, M. O'Neill, and A. Brabazon. Defining locality as a problem difficulty measure in genetic programming. *Genetic Programming and Evolvable Machines*, 12(4):365–401, 2011.

[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] A. Jain, M. Murty, and P. Flynn. Data clustering: a review. *ACM Computing Surveys*, 31(3):264?323, 1999.

[9] T. Kowaliw, A. Dorin, and J. McCormack. Promoting creative design in interactive evolutionary computation. *Evolutionary Computation, IEEE Transactions on*, 16(4):523 –536, 2012.

[10] J. Koza. Human-competitive results produced by genetic programming. *Genetic Programming and Evolvable Machines*, 11(3):251–284, 2010.

[11] 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.

[12] J. Lehman and K. O. Stanley. Efficiently evolving programs through the search for novelty. In *Proceedings of the 12th annual conference on Genetic*

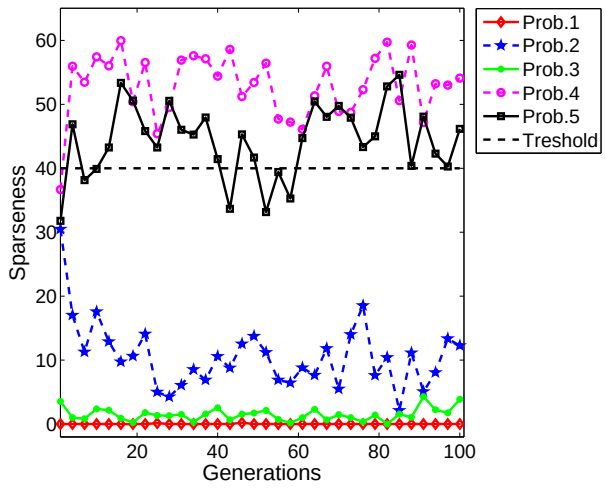
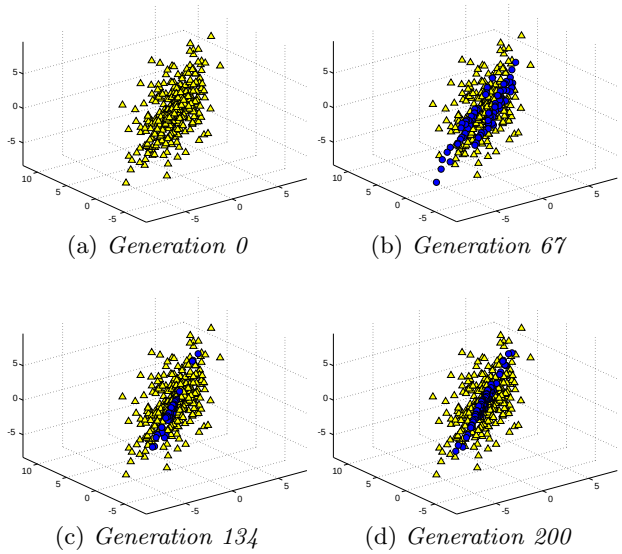


Figure 9: Evolution of sparseness for NS-GP-40, showing the average sparseness of he best individual at each generation.



(a) *Generation 0* (b) *Generation 67*

(c) *Generation 134* (d) *Generation 200*

Figure 10: Evolution of the best solutions found at progressive generations for Problem 5 with NS-GP-40.

and evolutionary computation, GECCO '10, pages 837–844. ACM, 2010.

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

[14] J. Lehman and K. O. Stanley. Evolving a diversity of virtual creatures through novelty search and local competition. In *Proceedings of the 13th annual conference on Genetic and evolutionary computation*, GECCO '11, pages 211–218. ACM, 2011.

[15] 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.

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

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

[19] C. Ofria and C. O. Wilke. Avida: a software platform for research in computational evolutionary biology. *Artif. Life*, 10(2):191–229, 2004.

[20] 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.

[21] J. Sun, W. Sverdlik, and S. Tout. Parallel hybrid clustering using genetic programming and multi-objective fitness with density (pyramid). In *Proceedings of the 2006 International Conference on Data Mining, DMIN 2006*, pages 197–203, 2006.

[22] L. Trujillo, Y. Martínez, E. Galván-López, and P. Legrand. Predicting problem difficulty for genetic programming applied to data classification. In *Proceedings of the 13th annual conference on Genetic and evolutionary computation*, GECCO '11, pages 1355–1362, New York, NY, USA, 2011. ACM.

[23] L. Trujillo, G. Olague, E. Lutton, and F. F. 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.

[24] 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.

[25] 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.

[26] B. G. Woolley and K. O. Stanley. Exploring promising stepping stones by combining novelty search with interactive evolution. *CoRR*, abs/1207.6682, 2012.

[27] R. Xu and I. Wunsch, Donald. Survey of clustering algorithms. *Neural Networks, IEEE Transactions on*, 16(3):645–678, may 2005.

[28] 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.