Supervised and Unsupervised Discretization Methods for Evolutionary Algorithms

Erick Cantú-Paz *
Lawrence Livermore National Laboratory
cantupaz@llnl.gov

Abstract

This paper introduces simple model-building evolutionary algorithms (EAs) that operate on continuous domains. The algorithms are based on supervised and unsupervised discretization methods that have been used as preprocessing steps in machine learning. The basic idea is to discretize the continuous variables and use the discretization as a simple model of the solutions under consideration. The model is then used to generate new solutions directly, instead of using the usual operators based on sexual recombination and mutation. The algorithms are tested with several functions and the results suggest that combining discretizers with EAs may be an interesting path for future developments.

1 INTRODUCTION

Probabilistic model-building evolutionary algorithms (EAs) are a promising path to construct reliable optimization algorithms that can solve difficult problems in reasonable times. However, most of these algorithms use discrete representations, which may not be natural to the problem at hand.

This paper presents simple model-building EAs that work on continuous domains. The algorithms are based on discretization methods that are used in machine learning. We distinguish between supervised discretizers that use a class label to find intervals and unsupervised methods that do not use labels. In EAs the labels are determined by the selection method and correspond to whether an individual survives to the next generation or not. In essence, the algorithms proposed here discretize the continuous variables and use the discretization as a simple model of the solutions under consideration. The model is used to generate new solutions directly; the EA omits the usual operators based on sexual recombination and mutation.

Typical discretization algorithms consider only one variable at a time. However, to succeed in problems where there are significant correlations between several variables, EAs must consider the related variables simultaneously or use exponential time. For space reasons, this paper considers only univariate methods, but it discusses a natural extension of one method to multivariate problems.

The next section briefly reviews model-building EAs. Section 3 describes some supervised and unsupervised discretization algorithms and the corresponding EAs. Section 4 presents experimental results. Finally, section 5 summarizes the paper and offers some recommendations for future research.

2 BUILDING MODELS TO GUIDE THE SEARCH

The idea of building probabilistic models and using them to guide the search of EAs has been around for some time. The complexity of the models has increased over time as the methods of building models from data mature and more powerful computers become available. This section briefly reviews previous work. Interested readers should consult the reviews by Pelikan et al. (1999) and Larrañaga et al. (1999).

The simplest model-building EAs use a product of univariate and independent probability distributions as the model of solutions. Baluja (1994) introduced the PBIL algorithm that uses a binary alphabet and a variation of the Hebbian learning rule to update its model. The compact GA (Harik et al., 1998) and the UMDA (Mühlenbein, 1998) are other examples of algorithms that use univariate models and operate on binary alphabets. Servais et al. (1997) extended PBIL to dis-

*UCRL-JC-142243. Work performed under the auspices of the US DOE by UC LLNL contract W-7405-Eng-48.

crete alphabets of higher cardinality and Sebag and Ducoulombier (1998) extended it to continuous variables. The model used by Sebag and Ducoulombier is a product of normal densities, and is similar to the model used by Rudlof and Köppen (1996). Gallagher et al. (1999) used a mixture of normal distributions.

More sophisticated algorithms such as MIMIC (de Bonet et al., 1996) and the BMDA (Pelikan & Mühlebein, 1999) capture relations between pairs of variables. Pelikan et al. (1999), Etxeberria and Larrañaga (1999), and Mühlenbein and Mahnig (1999) introduced algorithms that learn Bayesian networks, which can represent relations among arbitrary number of variables. However, these algorithms consider only discrete variables, and it is inefficient to discretize the domain variables as a preprocessing step because if b bins are used for each variable, a node in the network with $n-1$ parents would need to store b^n bins.

Larrañaga et al. (1999) proposed several EAs for continuous domains. Their first algorithm uses univariate distributions, then they adapted the MIMIC algorithm, and finally they proposed a method based that starts with a complete Gaussian network and tests all possible edge exclusion to identify conditional independences among the variables.

Sophisticated model-building EAs can consistently solve problems with complex interactions among many variables, but constructing probabilistic networks from data is costly. Bosman and Thierens (1999) discussed several ways to estimate the density of the selected solutions and determined that the algorithmic complexities vary from $O(n^2)$ to $O(n^3)$.

There have been other approaches to learn a model to guide the search of EAs. For example, Michalski (2000) introduced a system that induces rules that classify individuals into three groups depending on their fitness. The rules are used to generate new individuals. Michalski's method can operate on continuous variables, but they have to be discretized first.

3 DISCRETIZATION-BASED EAs

Analogously to machine learning algorithms, discretizers that use information about the label of the solutions are called supervised, and those that do not are called unsupervised.

3.1 UNSUPERVISED METHODS

The simplest unsupervised discretization method is to divide the range of observed values into b bins of equal width. Although it is easy to implement, this method is very sensitive to outliers, and may not adequately represent the distribution of each variable.

A related unsupervised method is to divide the range into b bins of equal frequency. This method is less susceptible to outliers, and the intervals would be closer to each other in regions where there are more elements and farther apart in sparsely-populated regions, which represents the distribution of each variable better than the equal-width method. In both methods the user must specify the number of bins b.

The corresponding EA is the same for these two unsupervised methods. In each iteration, the algorithm selects n_s promising solutions, discretizes each variable of these solutions independently, and for each variable generates n/b uniformly-distributed random numbers in each bin. These random numbers correspond to the values of the variable in the new population. Since the discretization captures the distribution of the *selected* individuals, we expect that the new individuals will have similar univariate marginal distributions than the selected individuals. This does not guarantee that the fitnesses of the new individuals will be as good as the selected ones, unless the hypothesis of the variables being independent holds.

This algorithm monotonically reduces the range of values for each variable, which is desired as we want to narrow the search. But if the algorithm narrows the search too fast, it may impede a proper exploration of the search space, which might impact negatively the quality of the solutions. One solution would be to use a large population to sample the search space better, but larger populations represent higher computational costs. Another solution would be to slightly enlarge the range of values generated for the new population. One possibility would be to add additional bins with a few elements at the extremes of the discretized range, but this creates a few additional design decisions (e.g., what is the range of the additional bins and how many elements should we put in them?). Another possibility is to adopt a mutation mechanism.

Possibly the major problem with the methods described in this section (and other algorithms that depend on univariate models) is their inability to represent accurately disjoint regions of promising solutions. The problem is that the algorithm discretizes the *entire* range of the selected individuals, which may include large areas of low performance. The equal-width bins method is the most sensitive to this problem since a large fraction of the bins may be used to represent regions of low performance. Certainly, it may be useful to generate a few points in the regions where the observed performance of a variable is low, because the apparent low-performing region may have not been sampled adequately missing good solutions. It may also be possible to reach good results when the observed low variable is combined with other variables in regions that have not been sampled yet. However, if the low-performance region dominates the range of a variable, it may be wasteful to use these methods.

3.2 SUPERVISED METHODS

Supervised discretizers find intervals where all or most of the data instances have the same label and the labels are different across consecutive intervals. Supervised discretizers assume that the data instances are labeled with the class to which they belong. In our case, the individuals have binary labels corresponding to whether they were selected to survive or not.

Holte (1993) proposed a simple algorithm that consists on sorting the observed values of a variable and greedily dividing the domain into bins that contain instances with the same label. To avoid having one bin for each observed value, each bin is required to have a minimum number of elements. Fayyad and Irani (1993) proposed a recursive method that finds intervals that minimize the class information entropy. The optimal boundary T^* that minimizes the entropy is chosen to partition the range, and the algorithm is applied recursively to the sets to the left and right of T^*. The resulting intervals will be closer to each other in the regions with high entropy, and far apart in the uniform regions where the entropy is low. In a sense, supervised discretizers create a rough model that predicts the class label based on the intervals. This model may be too inaccurate to be a practical classification algorithm, but it may be sufficient to guide the search of an EA. Recognizing that Fayyad and Irani's method can be regarded as building a tree, Kohavi and Sahami (1996) used C4.5 to discretize continuous variables. The next section shows some experiments using decision trees as supervised discretizers.

The corresponding EA is the same for these two supervised methods, and is similar to the EA for unsupervised methods. The core loop is to apply the selection method to label the individuals, discretize each variable using all the individuals (not just the selected), and generate uniform random numbers only in the intervals that correspond to the selected individuals. Note that this algorithm can easily represent disjoint regions of promising solutions, and the ranges of promising solutions also decrease monotonically.

As mentioned before, it may be useful to generate a few points in the regions of low performance. It is not necessary to modify the sampling procedure described above, because all the probabilistic selection methods assign a non-zero probability of selection to all individuals, including those with low fitness.

The natural extension of these algorithms is to consider multivariate discretization. This, of course, is much more complex than discretizing one variable at a time, but we can continue to borrow from the machine learning field and use inductive learning algorithms to build models of the individuals. For example, we may use decision trees to build a multivariate model of the current solutions in a EA. Decision trees are appealing in classification tasks because they are fast to create, reasonably accurate, and easy to interpret. Also, decision trees ignore variables that do not seem related to the class label, which can be an advantage in our case because it permits the algorithm to focus on the variables that have the greatest influence on the fitness at a particular stage of the search. In a sense, this reduces the dimensionality of the problem over time, which may result in algorithms that scale up better to the dimensionality than other model-building EAs. However, using trees introduces some difficulties on the generation of new individuals. For space reasons, we do not explore this option further. Of course, we are not limited to use decision trees to build multivariate models: we could use other learning algorithms that work on continuous domains or other discretization algorithms such as the algorithm of Kozlov and Koller (1997) and vector quantization.

4 EXPERIMENTS

The experiments test four univariate model-building EAs: two unsupervised algorithms based on equal-width and equal-frequency histograms, and two supervised algorithms based on Holte's discretization and decision trees. The test functions are commonly used to evaluate EAs and other optimization algorithms.

Most algorithms used a population size of 100 individuals for all problems, the exception was the EA based on Holte's discretization which used 400. Four-way tournaments were used to select/label the individuals. No mutation was used, which may explain some of the poor results. The histogram used only five bins. The experiments were halted after 200 generations.

Table 1 presents the average and the standard deviation (over 100 trials) of the best value found by each algorithm. The table specifies the number of dimensions of each problem and the initial range of each variable. The results suggest that the simple algorithms perform well, but on individual runs, the EAs based on supervised discretizers found competitive solutions.

Function	Dim.	Range	Equal Width	Equal Freq.	Holte's	Trees
Sphere	10	[-10,10]	**0.00011(0.00039)**	0.00082(0.002)	0.00421(0.0052)	0.082(0.014)
Griewank	10	[-600,600]	**6.5e-6(1.29e-6)**	0.00015(0.00033)	0.07843(0.0221)	0.02379(0.0252)
Schwefel	10	[-10,10]	0.085(0.010)	**0.00309(0.00345)**	0.0145(0.0089)	0.0331(0.0154)

Table 1: Final objective function values of different supervised and unsupervised discretization-based EAs.

5 SUMMARY

This paper proposed several EAs based on univariate supervised and unsupervised discretizers. The algorithms were tested on a few functions, and the results appear to favor the simpler unsupervised methods, but the results are not appear conclusive. In the future, the algorithms will be tested on other continuous optimization problems and compared against traditional and model-building EAs. Extensions to multivariate models are necessary.

References

Baluja, S. (1994). *Population-based incremental learning: A method for integrating genetic search based function optimization and competitive learning* (Tech. Rep. No. CMU-CS-94-163). Pittsburgh, PA: Carnegie Mellon University.

Bosman, P., & Thierens, D. (1999). *An algorithmic framework for density estimation based evolutionary algorithms.* Utrecht University Technical Report UU–CS–1999–46.

de Bonet, J., Isabell, C., & Viola, P. (1996). Mimic: Finding optima by estimating probability densities. In Jordan, M. et al. (Eds.), *Advances in Neural Information Processing Systems*, Volume 9. Cambridge, MA: MIT Press.

Etxeberria, R., & Larrañaga, P. (1999). Global optimization with Bayesian networks. In *II Symposium on Artificial Intelligence (CIMAF99).* (pp. 332–339).

Fayyad, U. M., & Irani, K. B. (1993). Multi-interval discretization of continuous-valued attributes for classification learning. In *Thirdteenth International Joint Conference on Artificial Intelligence* (pp. 1022–1027). San Francisco, CA: Morgan Kaufmann.

Gallagher, M., Frean, M., & Downs, T. (1999). Real-valued evolutionary optimization using a flexible probability density estimator. In Banzhaf, W. et al. (Eds.), *Genetic and Evolutionary Computation Conference* (pp. 840–846). San Francisco, CA: Morgan Kaufmann Publishers.

Harik, G. R., Lobo, F. G., & Goldberg, D. E. (1998). The compact genetic algorithm. In *IEEE Iternational Conference on Evolutionary Computation* (pp. 523–528). Piscataway, NJ: IEEE Service Center.

Holte, R. C. (1993). Very simple classification rules perform well on most commonly used datasets. *Machine Learning*, *11*, 63–90.

Kohavi, R., & Sahami, M. (1996). Error-based and entropy-based discretization of continuous features. In Simoudis, E. et al. (Eds.), *Second International Conference on Knowledge Discovery in Databases* (pp. 114–199). San Mateo, CA: AAAI Press.

Kozlov, A. V., & Koller, D. (1997). Nonuniform dynamic discretization in hybrid networks. In *Thirteenth Conference on Uncertainty in Artificial Intelligence* (pp. 314–325). San Mateo, CA: Morgan Kaufmann.

Larrañaga, P., Etxeberria, R., Lozano, J. A., & Peña, J. M. (1999). *Optimization by learning and simulation of Bayesian and Gaussian networks* (Tech Report No. EHU-KZAA-IK-4/99). Conostia-San Sebastian, Spain: University of the Basque Country.

Michalski, R. S. (2000). Learnable evolution process: Evolutionary process guided by machine learning. *Machine Learning*, *38*(1-2), 9–40.

Mühlenbein, H. (1998). The equation for the response to selection and its use for prediction. *Evolutionary Computation*, *5*(3), 303–346.

Mühlenbein, H., & Mahnig, T. (1999). FDA-A scalable evolutionary algorithm for the optimization of additively decomposed functions. *Evolutionary Computation*, *7*(4), 353–376.

Pelikan, M., Goldberg, D. E., & Cantú-Paz, E. (1999). BOA: The bayesian optimization algorithm. In Banzhaf, W. et al. (Eds.), *Genetic and Evolutionary Computation Conference* (pp. 525–532). San Francisco, CA: Morgan Kaufmann Publishers.

Pelikan, M., Goldberg, D. E., & Lobo, F. (1999). *A survey of optimization by building and using probabilistic models* (IlliGAL Report No. 99018). Urbana, IL: University of Illinois at Urbana-Champaign.

Pelikan, M., & Mühlebein, H. (1999). The bivariate marginal distribution algorithm. In Roy, R. et al. (Eds.), *Advances in Soft Computing - Engineering Design and Manufacturing* (pp. 521–535). London: Springer-Verlag.

Rudlof, S., & Köppen, M. (1996). Stochastic hill climbing with learning by vectors of normal distribution. In *First Online Workshop on Soft Computing* (pp. 60–70). Nagoya, Japan.

Sebag, M., & Ducoulombier, A. (1998). Extending population-based incremental learning to continuous search spaces. In Eiben, A. E. et al. (Eds.), *PPSN V* (pp. 418–427). Berlin: Springer-Verlag.

Servais, M., de Jager, G., & Greene, J. (1997). Function optimization using multiple-base population based incremental learning. In *Eighth Annual South African Workshop on Pattern Recognition* (pp. 6–11).