angle every 1.5 s such that this approach can deal in reasonable response time with common obstacles that might cause collisions in indoor environments.

## VI. Conclusions

In this study, a vision-based obstacle avoidance approach for ALV navigation has been proposed. The vehicle can detect obstacles, including walls and objects in the way, in an unknown indoor environment and safe collision-free paths can be generated from quadratic classifier design in real time. According to the collision-free path, the vehicle can modify the turning angle of the wheels to achieve the purpose of collision avoidance. Besides, a systematic method has been proposed for generating input patterns for classifier design to compute safe quadratic paths.

The use of quadratic paths instead of linear ones produces smoother paths and prevents dead-reckoning navigation to increase the flexibility of ALV applications in unknown complex environments with obstacles. Additionally, quadratic paths also match the ALV trajectory better than linear ones. A method for computing the optimal turning angle to avoid collisions in real time has also been proposed. The proposed approach has been implemented on a real ALV and a lot of successful navigations confirm the feasibility of the approach.

## References

[1] J. C. Hyland and S. R. Fox, "A comparison of two obstacle avoidance path plannings for autonomous underwater vehicles," in *Proc. Symp. Autonomous Underwater Vehicle Technology*, Washington, DC, June 1990, pp. 216–222.

[2] J. Cesarone and K. F. Eman, "Mobile robot routing with dynamic programming," *J. Manufact. Syst.*, vol. 8, no. 4, pp. 257–266, 1989.

[3] C. Acosta and R. G. Moras, "Path planning simulator for a mobile robot," *Comput. Indust. Eng.*, vol. 19, no. 1–4, pp. 346–350, 1990.

[4] J. O. Kim and K. Khosla, "Real-time obstacle avoidance using hormontic potential functions," *IEEE Trans. Robot. Automat.*, vol. 8, pp. 338–349, June 1992.

[5] K. Onoguchi, M. Watanabe, Y. Okamoto, Y. Kuno, and H. Asada, "A visual navigation system using a multi-infomation local map," in *Proc. 1990 IEEE Int. Conf. Robotics Automation*, Cincinnati, OH, vol. 2, pp. 767–774, May 1990.

[6] D. C. H. Yang, "Collision-free path planning by using nonperiodic B-spline curves," *J. Mech. Design*, vol. 115, pp. 679–684, Sept. 1993.

[7] L. L. Wang and W. H. Tsai, "Collision avoidance by a modified least-mean-square-error classification scheme for indoor autonomous land vehicle navigation," *J. Robot. Syst.*, vol. 8, no. 5, pp. 677–798, Oct. 1991.

[8] J. Borenstein and Y. Koren, "Real-time obstacle avoidance for fast mobile robots," *IEEE Trans. Syst., Man, Cybern.*, vol. 19, pp. 1179–1187, Sep./Oct. 1989.

[9] T. Skewis and V. Lumelsky, "Experiments with a mobile robot operating in a cluttered unknown environment," in *Proc. 1992 IEEE Int. Conf. Robotics Automation*, Nice, France, vol. 2, pp. 1482–1487, May 12–14, 1992.

[10] R. Bauer, W. Feitern, and G. Lawitzky, "Steer angle fields: An approach to robust manoeuvring in cluttered, unknown environments," *Robot. Auton. Systems*, vol. 12, pp. 209–212, 1994.

[11] K. Fukunaga, *Introduction to Statistical Pattern Recognition*, 2nd ed. San Diego, CA: Academic, 1990.

[12] C. C. Lai, "Outdoor autonomous land vehicle guidance by road information using computer vision and fuzzy wheel adjustment techniques," M.S. thesis, Inst. Comput. Inf. Sci., National Chiao Tung Univ., Hsinchu, Taiwan, R.O.C., June 1993.

[13] S. D. Cheng and W. H. Tsai, "Model-based guidance of autonomous land vehicles in indoor environments by structured light using vertical line information," *J. Elect. Eng.*, vol. 34, pp. 441–452, Dec. 1991.

[14] A. Ohya, A. Kosaka, and A. Kak, "Vision-based navigation of mobile robot with obstacle avoidance by single camera vision and ultrasonic sensing," in *Proc. 1997 IEEE/RSJ Int. Conf. Intelligent Robot Systems*, Grenoble, France, Sept. 1997, vol. 2, pp. 704–711.

[15] L. M. Lorigo, R. A. Brooks, and W. E. L. Grimsou, "Visually-guided obstacle avoidance in unstructured environments," in *Proc. 1997 IEEE/RSJ Int. Conf. Intelligent Robot .Systems*, Grenoble, France, vol. 1, pp. 373–379, Sept. 1997.

[16] Y. G. Yang and G. K. Lee, "Path planning using an adaptive-network-based fuzzy classifier algorithm," *13th Int. Conf. Computers Applications*, Honolulu, HI, Mar. 1998, pp. 326–329.

[17] R. Biewald, "Real-time navigation and obstacle avoidance for nonholonomic mobile robots using a human-like conception and neural parallel computing," in *Int. Workshop Parallel Processing Cellular Automata and Array*, Berlin, Germany, Sept. 1996, pp. 232–240.

# Dynamic Fuzzy Control of Genetic Algorithm Parameter Coding

Robert J. Streifel, Robert J. Marks, II, Russell Reed, Jai J. Choi, and Michael Healy

***Abstract*—An algorithm for adaptively controlling genetic algorithm parameter (GAP) coding using fuzzy rules is presented. The fuzzy GAP coding algorithm is compared to the dynamic parameter encoding scheme proposed by Schraudolph and Belew. The performance of the algorithm on a hydraulic brake emulator parameter identification problem is investigated. Fuzzy GAP coding control is shown to dramatically increase the rate of convergence and accuracy of genetic algorithms.**

## I. Introduction

Genetic algorithms are powerful search techniques which have been applied to many practical problems. However, the accuracy of the final solution found by binary coded genetic algorithms is limited by the number of bits used to code search parameters into strings. The low resolution of binary coding does not seriously affect the solution for many problems (e.g., integer and combinatorial searches). Accuracy becomes a more important consideration when

1) the search space consists of floating point parameters;
2) the parameters have a large dynamic range;
3) a relatively small number of bits are used to code the parameters.

The standard genetic algorithm uses no problem specific information except the relative fitness of the coded binary strings. Lack of gradient information can cause slow progress in search regions where the objective function has nearly zero gradient. The combination of low slope areas and low resolution binary coding can cause slow convergence on many practical problems.

The fuzzy genetic algorithm parameter (GAP) coding methodology presented in this paper is specifically designed to improve the search performance on a parameter identification problem. Conventional genetic algorithm parameter coding is static, the coding is constant for the entire search. This results in slow convergence. Greater accuracy

Manuscript received April 15, 1996; revised July 5, 1997. This paper was recommended by Associate Editor L. O. Hall.

R. J. Streifel, R. J. Marks, II, and R. Reed are with the University of Washington, Seattle, WA 98195 USA (e-mail: robert.j.streifel@boeing.com).

J. J. Choi and M. Healy are with the Boeing Information and Support Services, Seattle, WA 98195 USA.

Publisher Item Identifier S 1083-4419(99)00903-6.

in the final solution is obtained and convergence speed is increased by dynamically controlling the coding of the search space. The addition of fuzzy rules for control of coding changes provides more uniform performance in genetic algorithm searches. Similar fuzzy control of learning and optimization parameters based on implementation heuristics have been applied to error back propagation training of multilayered perceptrons [1], [2], [4], [5], random optimization [2], [4], ART neural networks [2], [4], and Kohonen neural networks [12].

The following section describes previous work on improving the performance of genetic searches by dynamically controlling the representation of the search parameters in the coding. The use of fuzzy rules to dynamically control GAP coding is described in Section III. Section IV contains results of fuzzy GAP coding experiments and compares the performance to the standard genetic algorithm with static coding and the genetic algorithm parameter coding method proposed by Schraudolph and Belew [10].

## II. Background

Several approaches, including parameter coding control, have been proposed to improve the performance of genetic algorithm search. A brief overview is appropriate.

### *A. Adaptive Representation Genetic Optimizer Technique (ARGOT)*

Schaefer [9] proposed an algorithm for modifying the representation of search parameters by controlling coding. The coding is adapted as the genetic algorithm searches by means of an intermediate mapping. The intermediate mapping converts the genetic strings to the search parameters. A standard genetic algorithm is used to provide the search using numerous triggered operators to adapt the parameter coding. These operators are applied to the coding transformation when certain conditions are detected.

A variety of population measurements are used to trigger the operators. Measurements such as the degree of convergence, the position of the cluster of strings, and the variance of the cluster are used. The measurements dictate the change in the number of bits in the parameter coding as well as the shift, contraction, or expansion of the search region.

Although effective, the algorithm requires the setting of many algorithm parameters and thresholds. These algorithm parameters can be difficult to define and make implementation rather difficult. Schaefer's heuristics for dynamic coding, roving search regions, region centering, and expansion and contraction of the search region, can be implemented using fuzzy GAP control.

### *B. Evolution Programs*

Evolution programming is eloquently presented in the book by Michalewicz [7]. These search algorithms are similar to genetic algorithms but do not require the search parameters to be coded into binary strings. The parameters, rather, are represented as arrays of floating point numbers. The coding effects of binary strings such as course resolution are thus not present.

The evolutionary operators are problem dependent. Many of the proposed operators are geometrically intuitive. The ability to include complex constraints in the search is a significant advantage. However, the large search space and specialized operators required by these algorithms can often make convergence very slow. If constraints are required in the search, the use of an evolutionary program rather than a genetic algorithm is warranted. However, experience indicates that many problems can be solved to a satisfactory accuracy using the faster convergence of genetic algorithms and specifically by the fuzzy GAP coding methodology proposed in this paper.

### *C. Delta Coding*

The delta coding algorithm proposed in [13] also motivates components of the fuzzy GAP coding algorithm. The delta coding algorithm begins by performing a standard genetic search until the population of strings has converged.

After convergence, the best solution found by the genetic algorithm is saved. The genetic algorithm is restarted with the search parameters being offsets from the previous best solution rather than the actual parameter values. This has been shown to improve search since the search space is advantageously altered in this step.

After each delta iteration, a specified number of bits are removed from the delta representation. The delta values are reinitialized and the genetic search is restarted if the delta values are not all zero. If all delta values are zero after an iteration, the number of bits is increased. This process is repeated until a suitable solution is found.

The use of delta values in this method is similar to both the ARGOT technique and the fuzzy GAP coding adaptation method. However, an intermediate mapping is used in the latter two techniques to change the representation rather than merely changing the number of bits, the strategy used in the delta coding method. Mathias and Whitley present a thorough study of the algorithm in [6] and discuss the benefits of using Gray coding in genetic algorithms.

### *D. Dynamic Parameter Encoding*

The dynamic parameter encoding (DPE) algorithm proposed by Schraudolph and Belew [10] has also provided significant inspiration for fuzzy GAP coding. In the encoding procedure proposed by Schraudolph and Belew, the search space for each parameter is defined by an offset and a range. The genetic string can then be thought to represent numbers in the interval [0.0, 1.0].

The DPE algorithm considers only a few of the most significant bits of the search. When the population has converged, the search space is reduced by half using a histogram filtering approach. The convergence of each parameter is considered independently and no initialization is performed between the genetic iterations. The search regions are not moved and are only reduced in size.

The independent parameter convergence criteria causes difficulty with some problems. The experiment on the eight dimensional Rosenbrock's function described in Section IV is a good example. Due to the random initialization, some parameters will converge before others. But the converged parameters may not be in the correct region due to the shape of the objective function. The small number of bits used in parameter coding also causes convergence problems. If the true objective function minimum is not represented to sufficient precision by the small number of bits, the region will potentially be moved to the wrong location.

## III. Fuzzy GAP Coding

The fuzzy GAP coding proposed in this paper for controlling the genetic algorithm parameter coding shares many of the characteristics of the algorithms described in the previous section. The use of an intermediate mapping between the genetic strings and the search space parameters is also used in the fuzzy GAP coding—similar to both the ARGOT and Dynamic Parameter Encoding schemes. The genetic string contains parameters which represent delta values. Each search space parameter is specified by the following equation:

$$p_s = \left(\frac{p_g}{2^l - 1}\right)R + O \tag{1}$$

where $p_s$ is the search space parameter, $p_g$ is the genetic parameter, $l$ is the number of bits in the genetic parameter, $R$ is a specified parameter range, and $O$ is a specified offset. This coding is illustrated

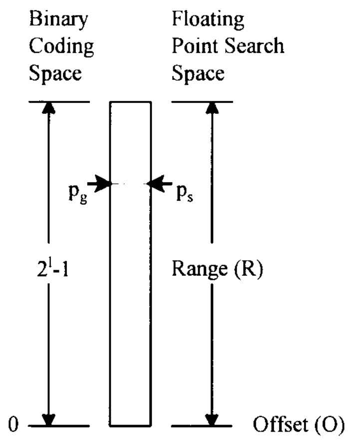


Fig. 1. Correspondence between a binary genetic string parameter and the floating point search parameter. The range $(R)$, offset $(O)$, and number of binary digits $(l)$ are used to specified the floating point parameter. By controlling the offset and range, more accurate solutions are obtained using the same number of binary bits.

in Fig. 1. The search space parameter is controlled by specifying an appropriate range and offset. The offset is the minimum search space parameter value and the range specifies the interval to be searched. By adjusting the offset and reducing the range, increasingly more accurate solutions can be coded into the binary string.

### *A. Convergence Criteria*

To investigate convergence of fuzzy GAP coding, the criteria used by Whitley *et al.* [13] is used. Convergence is measured by evaluating the average number of bits which differ between all the genetic strings. Each string is compared to every other string and the number of different bits are counted. If the average number of differing bits per string pair is less than a threshold, the genetic algorithm has converged. This convergence criterion requires a large computational effort for searches with many bits or strings.

### *B. Position Measurement*

After the genetic strings have converged, the new range and offset for the search parameters are determined by measuring the distance between the center of the current range and the best solution found in the search. The measurement is performed for each parameter independently. The distance measure is relative to the range of the parameter and thus lies in the interval [0.0, 1.0]

$$d(x, O, R) = \left| 2\left(\frac{x - O}{R}\right) - 1 \right|. \tag{2}$$

This distance scaling function is shown in Fig. 2. A distance value of 0.0 indicates that the best solution was exactly in the center of the range. A value of 1.0 indicates the best solution was either at the lower limit or upper limit of the range.

Heuristic rules are easily developed given this position measurement. For example, if the best solution is near the center of the range, it makes sense that the range should be reduced in size. The best solution in the center of the range indicates that previous range adjustments were correct and the true solution is near the center. Range adjustments in previous generations center the range on the best solution. So, if the best solution is near one of the limits, the best solution is moving and the search space should be adjusted to include more of the space about the best solution. Thus, increasing the size and centering the range is reasonable. The use of fuzzy rules allows easy and straightforward implementation of heuristic rules.

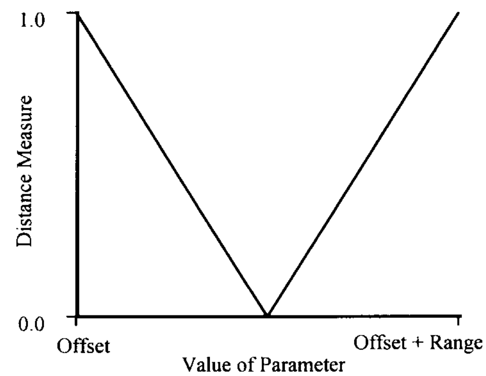


Fig. 2. Shape of the distance function. The distance measure is scaled relative to the parameter range. A parameter in the center of the range will have a distance measure of zero. A parameter value at the extremes of the range will have a distance measure of one.

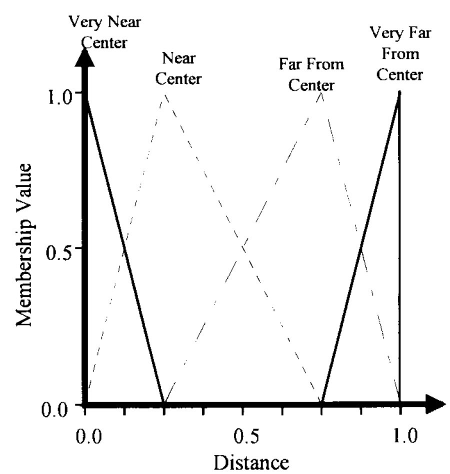


Fig. 3. Membership functions of the distance classes are constructed using triangles. A parameter near the center of the search range will have a distance measure which is near zero and thus will have membership in the "very near center" and "near center" classes.

### *C. Fuzzy Rules*

The distance measure is divided into four fuzzy classes: *very near center, near center, far from center*, and *very far from center*. The change in the range is also divided into four fuzzy classes: *decrease greatly, decrease slightly, increase slightly*, and *increase greatly*. The membership functions used for the distance measure classes are shown in Fig. 3. The membership functions are positioned such that only two have a nonzero membership value at any value of the distance measure. The "change in range" membership functions are constructed in a similar manner and are shown in Fig. 4.

The fuzzy If-Then rules have the form

$$\text{If } (antecedent) \text{ then } (consequent)$$

where the antecedent applies to the distance classes and the consequent applies to the range change classes. The following fuzzy rules describe the changes in the range of the search parameter using these classes.

- If the distance is (*very near center*) then the range change is (*decreased greatly*).
- If the distance is (*near center*) then the range change is (*decreased slightly*).
- If the distance is (*far from center*) then the range change is (*increased slightly*).

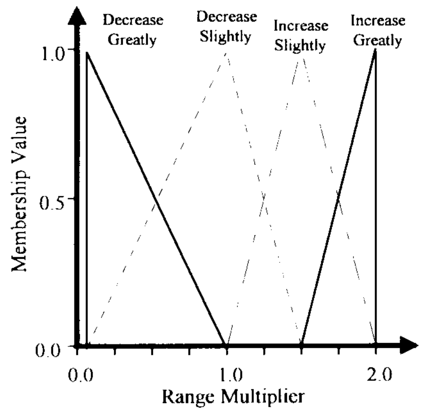


Fig. 4. Range multiplier class membership functions are constructed using triangles. The lower and upper limits are vertical to control the minimum and maximum values of the range multiplier result.

- If the distance is (*very far from center*) then the range change is (*increased greatly*).

Each of the search space parameters may have absolute limits on allowable values. The range of allowable values is often known for many practical search problems. However, difficulty results if only the above rules are used when the true solution is close to the absolute range limits. If the best solution is near the range limit, the range will be increased and will typically result in a new range which exceeds the hard limits of the search. The range will then be reset to the original value and no progress will be made. To over come this difficulty, an additional (crisp) rule is used.

- If (*hard limit is exceeded*) then (*center range on best solution with maximum range*).

Application of this rule will result in a much smaller search region than determined by the fuzzy rules. Since the range is allowed to shift and grow as better solutions are found, the reduction in range is not a problem. An incorrect range reduction will result only if the problem is not sufficiently smooth and the reduced range does not include the global minimum. Since the genetic algorithm is allowed to converge before changing the range, the static coding algorithm would not find a better solution.

The result of evaluating the fuzzy rules is a set of membership values in the output fuzzy classes. Since the range multiplier must be crisp (a single value), the fuzzy decision must be defuzzified. First, each output class membership function is multiplied by the corresponding rule consequent membership value. For example, let the distance class membership be {06, 0.4, 0.0, 0.0}, where the set of numbers represents the membership of a parameter in the distance classes $\{$*very near center, near center, far from center, very far from center*$\}$. The consequent membership values would then be $\{0.6, 0.4, 0.0, 0.0\}$, where the set represents membership in the range change classes $\{$*decrease greatly, decrease slightly, increase slightly, increase greatly*$\}$. The *decrease greatly* membership function is multiplied by 0.6 and the *decrease slightly* membership function is multiplied by 0.4. The other output membership functions are set to zero. The defuzzification is then performed by computing the center of mass of the weighted and aggregated output membership functions.

## IV. Experiments

The performance of the algorithm described above is now illustrated. The performance is compared to the standard genetic algorithm with static coding and to the related Dynamic Parameter Encoding (DPE) method proposed by Schraudolph and Belew [10].

TABLE I
Genetic Algorithm Parameters for Solution of the Ten Dimensional Optimization Problem

| | |
|---|---|
| Parameters Common to All Three Genetic Algorithms | Number of Strings = 100<br>Probability of Cross Over = 0.95<br>Probability of Mutation = 0.001 |
| Static Algorithm Convergence | Average Number of Different Bits = 0.10 |
| Fuzzy GAP Coding Algorithm Convergence | Average Number of Different Bits = 20.0 |
| DPE Parameters | Filter Time Constant = 50<br>Convergence Threshold = 0.95 |

The standard genetic algorithm is designed for large parameter identification problems (for example, the hydraulic brake simulation problem below) and seems to function quite well. Good solutions were obtained in a small number of generations. The basic algorithm is derived from the simple genetic algorithm described by Goldberg [3].

### A. Quadratic

The ten-dimensional quadratic function is a simplistic toy optimization problem. However, the problem illustrates the advantage of dynamic coding of genetic parameters and provides a convenient comparison of algorithm performance. The objective function to be minimized is given by

$$E(\vec{x}) = \sum_{i=1}^{10} x_i^2. \tag{3}$$

The minimum error occurs when all elements of the vector $\vec{x}$ are zero. The minimum value of the error is also zero.

The parameters of the algorithms used for comparison are listed in Table I. The basic genetic algorithm parameters were common to all the algorithms. The convergence parameters for the fuzzy controlled coding algorithm and the DPE method were selected to produce good results. The effect of the convergence parameter on the fuzzy controlled coding algorithm is discussed below. Since the static coding algorithm does not adjust the parameter coding, a high degree of convergence was required to achieve good results. The same convergence algorithm was used for the static algorithm as for the fuzzy GAP coding algorithm.

The search with each algorithm was repeated 20 times and the error of the best solution at each generation was averaged to produce the curves shown in Fig. 5. The solid line shows the performance of the static genetic algorithm. The search converges well before 500 generations and typically finds the best possible solution given the precision of the parameter coding. The DPE algorithm does slightly better than the static algorithm on average. However, the best solution found by the DPE is significantly better than the static algorithm. The fuzzy algorithm makes continuous improvement on average. The performance of the fuzzy GAP coding algorithm is superior to the other algorithms in terms of the average final solution. The DPE method found solutions which were closer to the optimal values but convergence to incorrect regions caused less reliable convergence on average.

Additional experiments were also performed to determine the effect of the number of bits used to encode the parameters. The reader should refer to Section IV-F for details.

The additional time required by the fuzzy control of the genetic algorithm over the conventional algorithm is very small. The param-

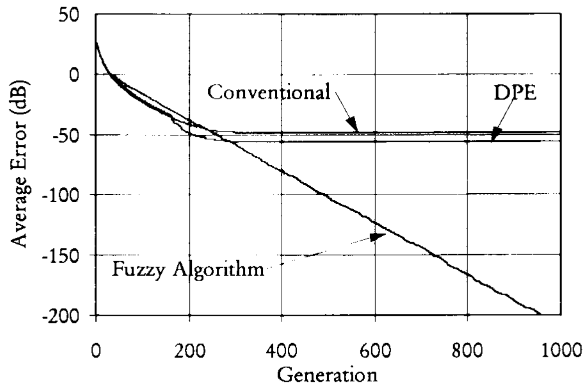


Fig. 5. Comparison of the three algorithms on the ten-dimensional quadratic function. Note that the average error is displayed on a decibel (20 $\log_{10}$ ERROR) scale. The average error of 100 trials is given in decibels.

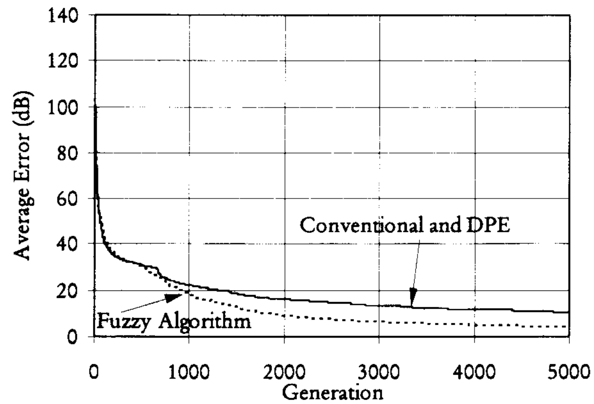


Fig. 6. Performance of the three searches on the Rosenbrock function. The objective function value is plotted in a decibel scale. The solid line shows the average performance of the static genetic algorithm over 100 trials and 5000 generations. The dotted line shows the performance of the fuzzy GAP controlled coding and the dashed line shows the performance of the DPE method. The static method and the DPE method produce equivalent results and the results are graphically indistinguishable. The difference between the fuzzy controlled coding algorithm and the static genetic algorithm is about 6 dB corresponding to a factor of about 4.

TABLE II
GENETIC ALGORITHM PARAMETERS USED FOR OPTIMIZATION OF $N = 8$ ROSENBROCK FUNCTION

| | |
|---|---|
| Parameters Common to All Three Genetic Algorithms | Number of Strings = 100<br>Probability of Cross Over = 0.95<br>Probability of Mutation = 0.001 |
| Static Algorithm Convergence | Average number of different bits = 0.2 |
| Fuzzy GAP Coding Algorithm Convergence | Average number of different bits = 4.0 |
| DPE parameters | Filter Time Constant = 100<br>Convergence Threshold = 0.99 |

eter adjustment is only performed after the convergence has been detected. The parameters were set such that convergence was rarely detected and it was found that both the fuzzy gap coding algorithm and the conventional algorithm took the same time to complete. When the parameters shown in Table I were used, the fuzzy gap coding algorithm required approximately 454.2 s and the conventional algorithm required 453.4 s, an increase of 0.2%. The problem is very easily computed in this case. The percent increase in execution time will be far less as the complexity of the objective function increases.

### *B. Rosenbrock's Function*

To provide a more difficult test of the coding algorithm, Rosenbrock's function [8] is used. This function provides a shallow slope toward the minimum in some regions and a steep slope in others. This problem is badly conditioned and has significant curvature variation. The problem is a challenging optimization problem even for gradient techniques. Several gradient techniques have been developed to solved this problem. However, the function provides a convenient comparison of algorithm performance. The function is

$$E(\vec{x}) = 100 \sum_{i=0}^{N-2} (x_{i+1} - x_i^2)^2 + (1 - x_i)^2 \tag{4}$$

where $N = 8$ was used. The error function is minimum when all elements of the vector have a value of 1.0.

The parameters required to achieve good search results are different from those used for the quadratic function. The convergence parameters need adjustment to prevent the search from converging on an incorrect region. The parameters used are listed in Table II. The parameters were not tuned but merely adjusted slightly to improve the performance of the static and GPE algorithms.

The performance of the three searches are shown in Fig. 6. The fuzzy controlled coding shows the best average performance with a 6 dB improvement over the static algorithm. The DPE performance is similar to the static algorithm. The DPE algorithm converges on solutions one parameter at a time. Since the DPE coding adjustments do not allow widening the search region once it has been reduced, errors made in the early part of the search are not corrected later when other parameters allow location of better solutions. Unlike the quadratic function, the DPE algorithm did not find the best solution of the three methods. The poor performance of the three search methods demonstrates the difficulty of this problem.

### *C. Bessel Function*

One of the greatest advantages of genetic algorithm search is the ability to avoid local minima. Any change in the algorithm should not interfere with the location of global minima. To test both the ability to find global minima and improve performance, the following error function was used:

$$E(\vec{x}) = \frac{1}{2}\left[2 - J_0\left(\frac{x_0 - 1}{4}\right) - J_0\left(\frac{x_1 - 1}{8}\right)\right] \tag{5}$$

where $J_0$ is a zero order Bessel function of the first kind. This two dimensional error function has a large number of local minima and a global minimum at the coordinates (1.0, 1.0). The search space extended from $-65.536$–65.536. This large range prevents the location of adequate solutions using a static genetic algorithm when only eight bits are used for each parameter.

The plot in Fig. 7 shows the performance of the three algorithms for the multimodal Bessel function. The parameters used were the same as the parameters used for the quadratic function solution. The DPE method and the static method had essentially equal performance. Both methods failed to find adequate solutions due to the course quantization but reached the best solution very quickly. The dotted line shows the performance of the fuzzy controlled coding. This performance plot was clipped at $-200$ dB and actually exceeded that level on average. The fuzzy control of parameter coding proved to be vastly superior in this case.

### *D. Hydraulic Brake Simulation*

The fuzzy GAP coding algorithm was initially developed to solve a difficult applied parameter identification problem. The objective was

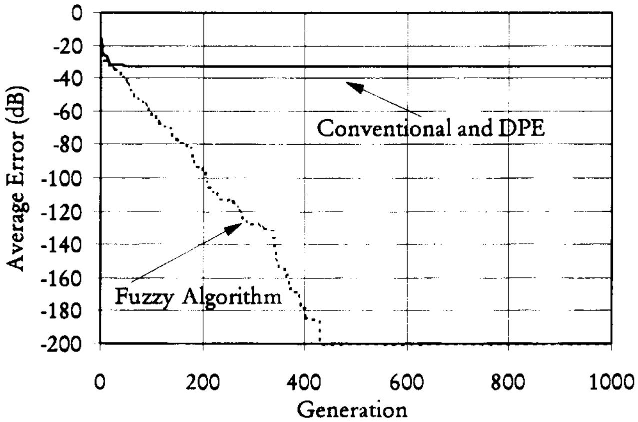


Fig. 7. Performance of the three genetic algorithm coding algorithms on the multimodal Bessel function. The error is the average of 20 independent trials. The DPE method and the static method exhibit essentially equal performance and are graphically indistinguishable. Both methods failed to find adequate solutions due to the course quantization. The dotted line shows the performance of the fuzzy GAP coding algorithm.

to develop an emulation model of the Boeing Commercial Airplane Group hydraulic brake system to replace hardware with simulation software. The use of hydraulic hardware in a simulation system loop can be prohibitively expensive and time consuming. Replacement of the hardware with a computer model allows faster studies in systems using the hydraulic system as a component.

A rough model of the brake system was developed with the help of a brake system engineer. The model is nonlinear, dynamic, and required optimization over both continuous and discrete parameters. The discrete parameters were removed from the search since the same values were always found. In order to facilitate optimization of the model, the hydraulic hardware output was acquired corresponding to numerous inputs under exhaustive conditions. The fitness function used for the genetic algorithm was derived directly from the rms error between the model and hardware output.

The parameters of the model were found using the static genetic and fuzzy GAP coding algorithms described above. The model consists of 33 floating point parameters coded into 16 bits each. Detailed discussion of the details of the model are beyond the scope of this paper and are described in [11].

Using static genetic algorithms, the model error decreases rapidly during the early portion of the search but progress slows considerably later. The slow progress later in the search indicates that reducing the search space will provide additional improvements in the hydraulic system model. The parameters have a rather large range of allowable values and some have a much larger effect on the error than others. Early attempts to reduce the search space were performed manually. A new search space was specified around the best solution found by previous trials. This method improved the results in the genetic search but was much too time consuming for practical applications. The use of fuzzy rules to control the coding is a natural means of automating manual tuning.

Since the processing time is extensive for this problem, a modified convergence criterion is used to provide more updates for the dynamic coding algorithm. The genetic search continues until all the strings are similar as before and, additionally, if a specified number of generations have been performed without reduction in the error, the algorithm is considered to have converged. Typically, a maximum of 5000 generations are performed and if no improvement is made in 2500 generations, the search is terminated.

TABLE III
PERFORMANCE COMPARISON OF THE HYDRAULIC BRAKE EMULATION OPTIMIZATION USING GENETIC ALGORITHMS

| | Static Coding Final Error | Fuzzy Controlled Coding Final Error |
|---|---|---|
| Full Hard Limits | 1.797 | 0.402 |
| Hand Reduced Limits | 0.357 | 0.266 |

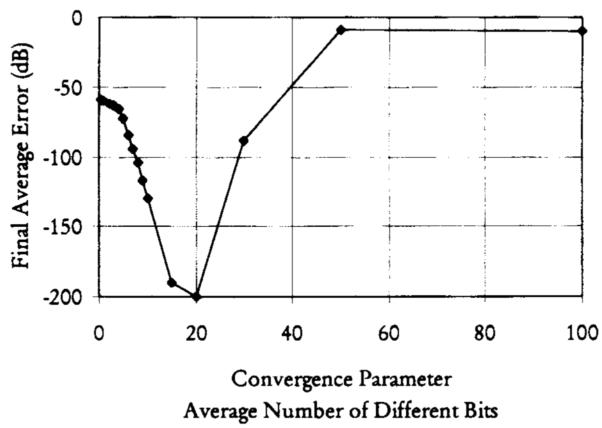


Fig. 8. Effect of the average number of different bits on the average final error. The minimum error achieved for each trial is averaged over 100 trials and the final value is plotted as a function of the convergence criterion parameter. The average final error is plotted on a decibel scale.

Table III shows the results of the static coding and the fuzzy GAP coding control. The first row shows the final scaled sum of squared errors using the initial full range for all parameters. The fuzzy control of coding algorithm produced a final error which was reduced by more than 77%. The fuzzy GAP control algorithm was still making improvements when the program terminated but the accuracy at the stopping point was judged sufficient. The second line shows the performance of both algorithms using the reduced limits. Even though the limits were reduced by hand over the span of many different trials, the fuzzy controlled coding algorithm still reduced the error by approximately 25%. The fuzzy GAP control algorithm achieved approximately the same performance level using the initial parameter ranges in a single trial. Further processing further reduces the error but the performance of the model was considered adequate for emulation of the brake hardware and processing was halted.

### *E. Convergence Criteria Experiment*

The only variable parameter available in the fuzzy GAP control algorithm is the convergence criterion. If fewer than a specified average number of bits are different between the genetic strings, the genetic algorithm is considered to have converged. The effect of the convergence criterion is now examined.

The ten-dimensional quadratic problem is used for this study. The final average error achieved by the genetic search is plotted in Fig. 8 as a function of the convergence parameter, the average number of different bits. Each search is executed 100 times using the same parameters as used for the fuzzy controlled genetic search in Section IV-A. The average error is seen to have a minimum at a value of 20. Using a value of 20 for other problems, however, did not result in accurate search results. The same study was performed on the Rosenbrock function but the final average error was very flat through the lower region and large values of the criterion resulted in incorrect solutions. It is reasonable to assume that a relatively small value is appropriate for most search problems. At very small values of the

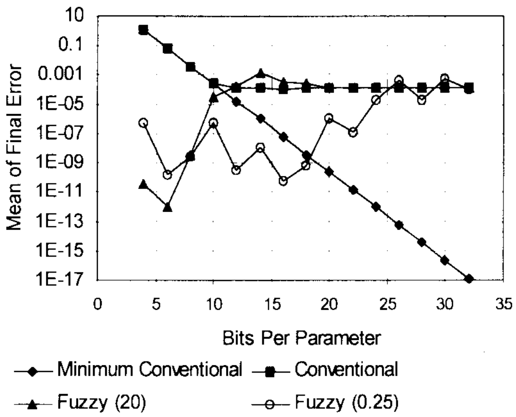


Fig. 9. Plot of the average final error achieved by genetic search as a function of the number of bits used to code the search parameters. Each point in the plot represents the average of 100 independent trials. The minimum possible error using a static coding genetic algorithm is shown for comparison. The final average error for the static algorithm, the fuzzy GAP coding algorithm with a constant convergence parameter of 20, and the fuzzy GAP coding algorithm with a variable convergence parameter is shown.

convergence criterion, little adjustment of the parameter coding will be allowed and therefore little advantage will be gained by using the fuzzy controlled coding. If the convergence parameter is too large, the coding will be changed before proper convergence is reached. The premature coding changes will result in searches in the wrong areas and thus poor search performance. A moderate value is always suggested unless multiple searches can be performed.

### *F. Parameter Size Experiment*

Another variable not considered in the studies above is the effect of the number of bits used to code each parameter in the search. Eight bits were used for each parameter in all of the experiments described above. If genetic algorithm search performance were not a function of the number of bits used to code the parameters, arbitrary accuracy could be achieved by increasing the number of bits used to code each floating point search parameter. However, increasing the number of bits for each parameter results in an increase in the dimension of the search space and thus effects the search.

The quadratic problem is again used for comparison. Fig. 9 shows the average final error obtained for 100 trials. The algorithm parameters are the same as used in Section IV-A and are shown in Table I. Due to the effects of quantization, the static algorithm may obtain a minimum error determined by the number of bits used to code the parameters. The minimum possible error is shown for comparison. The static algorithm search achieves the minimum possible error when the number of bits is less than 10. As the number of bits is increased, the static algorithm does not improve the search accuracy. The search is slowed due to the increase in the dimension of the problem but only 1000 generations were allowed before termination of the search. The fuzzy GAP coding algorithm shows that when a constant convergence parameter is used (indicated by the line labeled "Fuzzy (20)"), the smaller the number of bits, the more accurate the final solution. The fuzzy GAP coding algorithm dynamically adjusts the resolution to locate more accurate solutions. However, as the number of bits increases, the dimension also increases and convergence is more difficult to obtain. Thus, the search is much slower and performance approaches that of the static algorithm.

The convergence parameter has been shown in the previous section to have an impact on the search performance. Constraining the convergence parameter to a constant does not seem reasonable as the number of bits is changed. The original problem allowed an average of 20 different bits in each string before convergence was declared. Eight bits were used for each of the ten parameters. Therefore, 25% of the total number of bits in a string were allowed to be different. The above experiment was repeated but the convergence parameter was constrained to represent a fixed fraction of 0.25 rather than a constant number of different bits. For example, a convergence parameter of 10 was used when only four bits were used for each parameter and a convergence parameter of 45 was used when each parameter was coded with 18 bits. The performance identified with "Fuzzy (0.25)" in Fig. 9 indicates that a small number of bits provides more accurate solutions with less work. The fuzzy GAP coding algorithm exceeds the performance of the static algorithm even when as many as 24 bits are used to code each parameter.

## V. Conclusions and Future Work

The fuzzy GAP control of genetic algorithm parameter coding is shown to be an effective method for improving the resolution of genetic searches. The algorithm has also been shown to be more reliable than other dynamic coding algorithms providing more accurate solutions in fewer generations. This occurred despite the use of many of the concepts of these other algorithms in fuzzy GAP coding. The algorithm was shown to converge even when the objective function had numerous local minima.

The fuzzy rules used to control the genetic search in this paper were derived heuristically. Though the rule performed very well, techniques for automatic generation of fuzzy rules could be used to provide more robust rules. The rules used in this work were developed, coded, and applied to the test problems without modification. Since the performance of the genetic algorithm under control of the rules was significantly better than without the fuzzy control, no effort was made to improve the rules.

The parameter coding is modified after the genetic algorithm has converged and thus improvements (using the same number of bits for each parameter) are only available after an additional genetic search. Search time is increased to provide the increase in solution accuracy.

## Acknowledgment

The authors would like to acknowledge Boeing Phantom Works and the Boeing Commercial Airplane Group for support of this work. The contribution of R. von Doenhoff of the Boeing Commercial Airplane Group was critical to the development of the hydraulic brake system model. The review of this work and subsequent suggestions by Prof. J. N. Hwang of the University of Washington are greatly appreciated.

## References


[1] P. Arabshahi, J. J. Choi, R. J. Marks, II, and T. P. Caudell, "Fuzzy control of backpropagation," in *Proc. 1st IEEE Int. Conf. Fuzzy Systems (FUZZ-IEEE '92)*, San Diego, CA, Mar. 1992, pp. 967–972.

[2] P. Arabshahi, J. J. Choi, R. J. Marks, II, and T. P. Caudell, "Fuzzy parameter adaptation in optimization: Some neural net training examples," *IEEE Computat. Sci. Eng.*, 1996.

[3] D. E. Goldberg, *Genetic Algorithms in Search, Optimization and Machine Learning*. New York: Addison-Wesley, 1989.

[4] J. J. Choi, P. Arabshahi, R. J. Marks II, and T. P. Caudell, "Fuzzy parameter adaptation in neural systems," in *IJCNN Int. Joint Conf. Neural Networks*, Baltimore, MD, June 7–11, 1992, pp. 232–238, vol. 1.

[5] S. Haykin, *Neural Networks: A Comprehensive Foundation*. New York: IEEE Press, 1994, pp. 201–205.

[6] K. Mathias and L. D. Whitley, "Changing representations during search: A comparative study of delta coding," *Evol. Computat.*, vol. 2, no. 3, pp. 249–278, 1995.

[7] Z. Michalewicz, *Genetic Algorithms + Data Structures = Evolution Programs*. New York: Springer-Verlag, 1992.

[8] H. H. Rosenbrock, "An automatic method for finding the greatest or least value of a function," *Comput. J.*, vol. 3, pp. 175–184, 1960.
[9] C. G. Schaefer, "The ARGOT strategy: Adaptive representation genetic optimizer technique," in *Genetic Algorithms Applications: Proc. 2nd Int. Conf.*, 1987, pp. 50–58.
[10] N. N. Schraudolph and R. K. Belew, "Dynamic parameter encoding for genetic algorithms," *Mach. Learn.*, vol. 9, no. 1, pp. 9–21, 1992.
[11] R. J. Streifel, R. von Doenhoff, R. J. Marks II, J. J. Choi, and M. Healy, "Application of genetic algorithms to hydraulic brake system parameter identification," The Boeing Company, Seattle, WA, Doc. D6-81795TN.
[12] E. C. K. Tsao, J. C. Bezdek, and N. R. Pal, "Fuzzy Kohonen clustering networks," *Pattern Recognit.*, vol. 27, no. 5, pp. 757–764, 1994.
[13] D. Whitley, K. Mathias, and P. Fitzhorn, "Delta coding: An iterative search strategy for genetic algorithms," in *Proc. 4th Int. Conf. Genetic Algorithms*, L. Booker and R. Belew, Eds. San Mateo, CA: Morgan Kauffman, 1991, pp. 77–84.
[14] L. A. Zadeh, "Fuzzy sets," *Inf. Contr.*, vol. 8, pp. 338–353, 1965.

# Genetic K-Means Algorithm

K. Krishna and M. Narasimha Murty

***Abstract*—In this paper, we propose a novel hybrid genetic algorithm (GA) that finds a globally optimal partition of a given data into a specified number of clusters. GA's used earlier in clustering employ either an expensive crossover operator to generate valid child chromosomes from parent chromosomes or a costly fitness function or both. To circumvent these expensive operations, we hybridize GA with a classical gradient descent algorithm used in clustering viz., K-means algorithm. Hence, the name genetic K-means algorithm (GKA). We define K-means operator, one-step of K-means algorithm, and use it in GKA as a search operator instead of crossover. We also define a biased mutation operator specific to clustering called distance-based-mutation. Using finite Markov chain theory, we prove that the GKA converges to the global optimum. It is observed in the simulations that GKA converges to the best known optimum corresponding to the given data in concurrence with the convergence result. It is also observed that GKA searches faster than some of the other evolutionary algorithms used for clustering.**

***Index Terms*— Clustering, genetic algorithms, global optimization, K-means algorithm, unsupervised learning.**

## I. Introduction

Evolutionary algorithms are stochastic optimization algorithms based on the mechanism of natural selection and natural genetics [1]. They perform parallel search in complex search spaces. Evolutionary algorithms include genetic algorithms, evolution strategies and evolutionary programming. We deal with genetic algorithms in this paper. Genetic algorithms (GA's) were originally proposed by Holland [2]. GA's have been applied to many function optimization problems and are shown to be good in finding optimal and near optimal solutions. Their robustness of search in large search spaces and their domain independent nature motivated their applications in various fields like pattern recognition, machine learning, VLSI design, etc. In this paper, we propose an algorithm, that is a modification of GA, for clustering application.

Manuscript received September 4, 1995; revised August 27, 1997 and March 10, 1998.

K. Krishna is with the Department of Electrical Engineering, Indian Institute of Science, Bangalore 560012, India (e-mail: kkrishna@ee.iisc.ernet.in).

M. N. Murthy is with the Department of Computer Science and Automation, Indian Institute of Science, Bangalore 560012, India (e-mail: mnm@csa.iisc.ernet.in).

Publisher Item Identifier S 1083-4419(99)00770-0.

Clustering has been effectively applied in a variety of engineering and scientific disciplines such as psychology, biology, medicine, computer vision, communications, and remote sensing. Cluster analysis organizes data (a set of patterns, each pattern could be a vector measurements) by abstracting underlying structure. The grouping is done such that patterns within a group (cluster) are more similar to each other than patterns belonging to different groups. Thus, organization of data using cluster analysis employs some dissimilarity measure among the set of patterns. The dissimilarity measure is defined based on the data under analysis and the purpose of the analysis. Various types of clustering algorithms have been proposed to suit different requirements. Clustering algorithms can be broadly classified into hierarchical and partitional algorithms based on the structure of abstraction. Hierarchical clustering algorithms construct a hierarchy of partitions, represented as a *dendrogram* in which each partition is nested within the partition at the next level in the hierarchy. Partitional clustering algorithms generate a single partition, with a specified or estimated number of nonoverlapping clusters, of the data in an attempt to recover natural groups present in the data. In this paper, we confine our attention to partitional clustering of a given set of real-valued vectors, where the dissimilarity measure between two vectors is the Euclidean distance between them.

One of the important problems in partitional clustering is to find a partition of the given data, with a specified number of clusters, that minimizes the total within cluster variation (TWCV) (which is defined below). We address this problem, viz., minimization of TWCV, in the present paper. In general, partitional clustering algorithms are iterative and hill climbing and usually they converge to a local minimum. Further, the associated objective functions are highly nonlinear and multimodal. As a consequence, it is very difficult to find an optimal partition of the data using hill climbing techniques. The algorithms based on combinatorial optimization such as integer programming, dynamic programming and, branch and bound methods are expensive ever for moderate number of data points and moderate number of clusters. A detailed discussion on clustering algorithms can be found in [3].

The simplest and most popular among iterative and hill climbing clustering algorithms is the K-means algorithm (KMA). As mentioned above, this algorithm may converge to a suboptimal partition. Since stochastic optimization approaches are good at avoiding convergence to a locally optimal solution, these approaches could be used to find a globally optimal solution. The stochastic approaches used in clustering include those based on simulated annealing, genetic algorithms, evolution strategies and evolutionary programming [4]–[11]. Typically, these stochastic approaches take a large amount of time to converge to a globally optimal partition. In this paper, we propose an algorithm based on GA, prove that it converges to the global optimum with probability one and compare its performance with that of some of these algorithms.

Genetic algorithms (GA's) work on a coding of the parameter set over which the search has to be performed, rather than the parameters themselves. These encoded parameters are called *solutions* or *chromosomes* and the objective function value at a solution is the objective function value at the corresponding parameters. GA's solve optimization problems using a population of a fixed number, called the *population size*, of solutions. A solution consists of a string of symbols, typically binary symbols. GA's evolve