Designing Fuzzy Net Controllers Using Genetic Algorithms

Jinwoo Kim, Yoonkeon Moon, and Bernard P. Zeigler

As control system tasks become more demanding, more robust controller design methodologies are needed. A Genetic Algorithm (GA) optimizer, which utilizes natural evolution strategies, offers a promising technology that supports optimization of the parameters of fuzzy logic and other parameterized non-linear controllers. This article shows how GAs can effectively and efficiently optimize the performance of fuzzy net controllers employing high performance simulation to reduce the design cycle time from hours to minutes. Our results demonstrate the robustness of a GA-based Computer-Aided System Design methodology for rapid prototyping of control systems.

Introduction

Computer-Aided System Design (CASD) should support designing various functions of high autonomy systems [10], such as normal operation control, fault-tolerance, communication, planning, and scheduling. Since conventional control schemes are limited in their range of practical applications, fuzzy logic control and neural net methods are receiving increased attention for intelligent control applications. Such parameterized controllers can only work once their parameters have been set to satisfy the required control performance. Well-known successes have been reported with manual tuning of parameters (or equivalently, specification of the rules) for fuzzy logic controllers. However, a robust CASD methodology cannot rely on the availability of experts for all applications. On the other hand, neural nets require long periods of automatic training to determine their parameters (synaptic weights).

In the CASD methodology investigated here, the setting of parameters is performed by an optimization process. There are many classical optimization algorithms but they cannot be relied upon to find the optimal (global) solution in multi-parameter search spaces. These spaces are typically subject to discontinuities, multi-modality, and non-linear constraints which wreak havoc on gradient-based direct search methods. A genetic algorithm (GA) is a parallel, global search technique that emulates natural genetic operations[12]. Because it simultaneously evaluates many points in the parameter space, it is more likely to converge toward the global solution. It need not assume that the search space is differentiable or continuous and can also iterate several times on each datum received.

This article shows how GAs can effectively and efficiently optimize the performance of parameterized non-linear controllers based on fuzzy logic algorithms in high performance simulation environments. A brief review of the fuzzy net controllers and genetic algorithms is first presented. To show the versatility of the methodology, two examples are discussed where GA-optimized fuzzy logic controllers are applied to simulated non-linear plants.

Fuzzy Net Control Systems

The basic idea of fuzzy control centers around the labeling process, in which the reading of a sensor is translated into a label as done by human expert controllers [5]. With expert supplied membership functions for labels, sensor readings can be fuzzified and through fuzzy logic eventually defuzzified to generate analog control commands. It is important to note that the transitions between labels are not abrupt and a given reading might belong to several label regions.

The fuzzification, inferencing and defuzzification processes can be parallelized. For example, an input signal can be fuzzified by matching all membership functions simultaneously against the incoming value. In this way, fuzzy control processing can be viewed as a parallel neural network where each neuron represents a fuzzy membership function and each link represents the weight of a fuzzy rule. See the references [7,8] for more detail explanation of fuzzy net controller. In our application, we employ a bell-shaped membership function with a maximum of 1 and minimum of 0, such as

$$\mu_{A_i}(x) = \frac{1}{1 + \left(\frac{x - c_i}{a_i}\right)^p}$$

where $a_i$ : width, $b_i$ : steepness, $c_i$ : mean

Fig. 1(a) shows the structure of the Fuzzy Net Controller (FNC) and its fuzzy subspace (Fig. 1(b)) [7]. In this figure, five fuzzy regions are defined for the inputs and output. However, unlike ANFIS [7], the links between layers 3 and 4 in our fuzzy net controller are not fixed. These links represent consequents of fuzzy rules which could be optimized by GA.

While an earlier Fuzzy Logic Controller [3] was implemented in rule-based (if-then) form, the current FNC employs a parallel inferencing network structure. Due to such parallelization, the FNC can provide better real-time performance. Moreover, the
parallel scheme affords a straightforward basis for GA optimization.

Recently there has been research in developing well-performing fuzzy membership functions without help of human expertise [1]. To do this, it is necessary to employ computer-aided optimization. Tuning the membership functions requires adjusting many parameters simultaneously and is difficult to do manually. As indicated before, gradient-based, hill-climbing search methods cannot deal with the complexity of such multi-parameter search spaces. Instead, a probabilistic optimization method utilizing evolution strategies, such as the Genetic Algorithm (GA), can be employed to reliably find optimal membership functions. However, optimizing multi-parameter systems where each experiment requires a simulation can be very time consuming. Therefore, we developed new forms of GA which are especially oriented to simulation-based optimization on parallel computers [2,11].

Asynchronous Genetic Algorithms in High-Performance Simulation Environments

The GA is a probabilistic algorithm which maintains a population of individuals, \( P(t) = x_1(t), \ldots, x_d(t) \) for iteration \( t \). Each individual represents a potential solution to the problem at hand, and is implemented as some (possibly complex) data structure \( S \). Each solution \( x_d(t) \) is evaluated to give some measure of its fitness. Then new population (iteration \( t+1 \)) is formed by selecting the more fit individuals (select step). Some members of the new population undergo transformation (recombine step) by means of “genetic” operators to form new solutions. There are higher order transformations \( c_j \) (crossover type as shown in Figure 2(a-b), which create new individuals by combining parts from several (two or more) individuals and unary transformation \( m_i \) (mutation type as shown in Figure 2(c-d), which create new individuals by a small change in a single individual (\( m : S \rightarrow S \)) [12]. After some number of generations the search converges and

---

**Fig. 1.** Fuzzy inference network and fuzzy subspaces.

**Fig. 2.** Genetic operator: crossover and mutation.
is successful if the best individual represents the global optimum solution.

Parallel GAs (PGAs) were investigated to reduce search time in real applications [11]. We improved upon PGAs by employing Asynchronous Genetic Algorithm (AGA) which is especially oriented to massively parallel processing and does not need to be synchronized by generations to create successive populations [2]. In such a multiprocessor environment, individuals in the AGA are evaluated concurrently, and a controller updates the genetic population continuously as the evaluation results become known. This is important in our CASD methodology, since evaluations are performed by simulations and can be highly variable in their execution times. The optimization scheme was first simulated in the Chez-Scheme environment on a RISC workstation (Motorola MultiPersonal Computer 200) and its performance was tested with various test functions [2]. Recently, the scheme was ported to the CM-5 massively parallel computer. The performance benefits predicted by simulation were remarkably well corroborated [4].

Fig. 3 shows the operation of Asynchronous Genetic Algorithms in Connection Machine CM-5. Complex evaluation functions such as the FNC evaluation module can be implemented inside CM-5 processing nodes (GA-agent). When execution starts, genetic parameters, such as crossover and mutation probability, string size and population size, are initialized in the control processor (GA-controller). Whenever there are idle processing nodes available, the control processor requests new individuals from the gene-pool and sends them to the processing node. When an individual is returned to the control processor after evaluation, it updates the gene-pool. If other individuals arrive at the processor during the update operation, they are stored in the message queue until the current update is completed. The newly evaluated individuals replace its parent individual if fitness is higher than that of its parent. The selection for new individual is performed based on roulette wheel whose slot is sized according to fitness [6].

Interaction of the FNC Module with GA Optimizer

Fig. 4 shows the interconnection of the FNC, simulation model, and GA optimizer. The FNC operates the simulation model of the target system to be controlled. An individual of the GA population represents one trial set of fuzzy membership functions. A bell-shaped membership function needs three parameters (a,b,c, as shown before), so 45 parameters (15 membership functions x 3) are implemented in a single individual. Each parameter is coded by an eight-bit binary number and total length of an individual is a 360-bit binary code. Using a binary representation scheme makes it easy to apply genetic operations (mutation and crossover) to chromosome populations.

Fig. 5 provides more details of the interaction. The GA optimizer sends a parameter assignment (an individual) to the FNC which determines its fuzzy membership functions. The model is reset to its initial conditions. The operational specifications such as the target trajectory of each state variable are set inside the controller. During control operation, the values of state variables are sampled with some sampling time, while the FNC issues control commands to the simulation model.

CASD Application Examples

Our primary objective is to come up with optimal fuzzy membership functions and rules that perform well with given operational specifications while utilizing minimal human expertise. Two examples will now be discussed, where this design is performed by GA-based optimizers. Example 1 demonstrates how the GA optimized fuzzy membership functions to meet various operational specifications, for a space-based Oxygen Production System (OPS) [10]. Example 2 demonstrates the design of a FNC for the inverted pendulum system, an emerging benchmark problem for intelligent control. Both experiments employ the same genetic parameters: population size = 200, total number of evaluation = 40000, Pc = 0.8, Pm = 0.002.
For these design problems, we want to minimize the total amount of error, \( E = \text{desired-control-output} - \text{actual-control-output} \). Therefore, a smaller \( E \) represents a higher fitness (GA maximizes performance). There are two ways to convert the \( E \) to a fitness value of a GA.

1. \( \text{Fitness} = \text{Offset} - E \).
2. \( \text{Fitness} = \frac{1}{E} \).

In real-world problems, sometimes it is difficult to select an appropriate \( \text{Offset} \) value. If an inappropriate value is selected, the performance of the GA will be degraded. In order to strengthen the relative fitness differences between individuals in later GA operations, we choose scheme (2) to compute the fitness. The performance index of our GA experiments is

\[
\text{Performance Index} = \left( \frac{C_1}{E} \right) C_2
\]

where \( C_1 \) and \( C_2 \) are heuristically chosen to adjust the fitness difference.

The OPS, as shown in Figure 6, includes Zirconia tubes located symmetrically inside a cylinder [10]. A radiation heater is wrapped around the outer surface. With this configuration, the majority of heat transfer between the outer surface and the oxygen gas inside the system is due to radiation. Applying the one-dimensional heat equation with lumped temperature distributions for the surface and oxygen temperatures we obtained two first-order differential equations as provided below. \( T_p \) represents the pipe temperature and \( T_z \) is Zirconia tube temperature. A variable SW is either 0 or 1 to control the heat source (heater). The objective of the FNC is to supply heat to the Zirconia tubes so as to bring it to a goal operating temperature at a safe, constant rate.

\[
\frac{dT_p}{dt} = 2.75 \times SW - 4.42 \times 10^{-12} (T_p^4 - T_z^4) - 8.65 \times 10^{-4} (T_p - 278.0)
\]

\[
\frac{dT_z}{dt} = 4.42 \times 10^{-12} (T_p^4 - T_z^4)
\]

As shown in Fig. 1, there are two input signals to the FNC, e.g., temperature increase error rate (input1) and the derivative of this error rate (input2). Based on two inputs, the FNC produces output commands which control the on/off duty cycle of the heater. As shown in Fig. 1, we employed five membership functions for each input signal and five membership functions for the output signal.

The FNC controlled the plant with three different starting temperatures (278, 578, 878) and eight different temperature increase rates (4, 6, 8, 10, 12, 14, 16, 18). These operational specifications were prepared to meet possible scenarios where the plant was temporarily shut down, say, after the detection of an abnormal situation. Recovery to standard operating temperature would then have to start from temperatures other than that of cold start.

For each trial individual, the FNC evaluation module executes 24 cases (three different starting temperatures \( \times \) eight different temperature increase rates). The error in each case was normalized by the total rise time and aggregated to give the fitness of the trial individual. With this scheme for measuring fitness we compute the performance for each of the 24 cases using the same fuzzy membership functions. (An alternative, but more costly, scheme is to employ different membership functions and rules for different initial conditions.)

Fig. 7 shows a set of fuzzy membership functions and fuzzy rules found by the GA-optimizer. Fig. 8(a) illustrates the performance of this optimized FNC with different starting temperatures and increase rates. Among the 24 cases, the performance of the best case is provided in Fig. 8(a) and worst case is shown in Fig. 8(c). Note that in the latter case, the error is out of tolerance in the early period. The temperature profiles for nominal rate = 10"/min appear in Fig. 8(b).
Fig. 8. Performance of optimized FNC: (a) initial temperature: 278K, increase rate = 12K/min, (b) initial temperature: 278K, increase rate = 10K/min, (c) initial temperature: 878K, increase rate = 4K/min.

Summarizing, in this example, 24 different operational specifications have to be considered at the same time for a single adjustment of the FNC. This task is beyond human reasoning capability. Furthermore, the time taken for manual tuning—an experiment we performed suggests this to be several days—is impractical for timely design cycles. Even on a state-of-the-art conventional workstation, the time taken for the GA-based optimization was in the neighborhood of seven hours. However, our recent implementation on the CM-5 employing 512 processors reduced this time to seven minutes. Thus, a CASD environment which includes a GA optimizer and high-performance simulation capability enables the timely design of fuzzy logic controllers to meet demanding specifications.

Fig. 9 shows the schematic diagram of an inverted pendulum system. In this example, GA finds the membership functions as well as fuzzy rules for control of the Inverted Pendulum. The structure of the inverted pendulum system is composed of a rigid pole and a cart onto which the pole is hinged. The cart moves either right or left, depending on the force exerted on the cart. The pole is hinged to the cart through a frictionless free joint so that there exists only one degree freedom. The control objective is to balance the pole starting from non-zero conditions by supplying the appropriate forces to the cart.

If we let $x_1(t) = 0(t)$ and $x_2(t) = \dot{\theta}(t)$, then this system can be defined by the following differential equations [7]:

$$
\begin{align*}
\dot{x}_1 &= x_2 \\
\dot{x}_2 &= \frac{g \sin(x_1) + \cos(x_1) \left(-F - mlx_2^2 \sin(x_1)\right)}{l \left(\frac{4}{3} m \cos^2(x_1) + \frac{m_c}{m} + m\right)} \\
&= H_2(x_1, x_2, F)
\end{align*}
$$

where $g$ is 9.8 meter/sec$^2$, $m_c$ (mass of cart) is 1.0 kg, $m$ (mass of pole) is 0.1 kg, $l$ (half length of pole) is 0.5 meter, and $F$ is the applied force in Newtons. The objective of the GA in this experiment is to find the optimal set of membership functions and fuzzy rules for a FNC. Our fuzzy controller has the following specifications:

- **Input signals of FNC**: angle of pole (degrees), angular velocity of pole (degrees/sec.)
- **Output signal of FNC**: force (Newtons)
- **Type of membership function**: bell-shaped (requires parameters $a$, $b$, $c$ for the fuzzy membership function).
- **Fuzzy region**: NE (negative), ZE (zero), PO (positive)
- **Number of fuzzy rules**: 9

In this example, an individual contains parameter information of fuzzy membership functions (bell-shaped) and fuzzy rules. Twenty-seven parameters (three for each of nine membership functions) are required for fuzzy membership functions and 9 parameters are employed for fuzzy rules. We assigned 8 bits to each parameter so that the total size of an individual is 288 bits.

The GA searches for an optimal FNC with a single training data set: an initial angle of 10 deg, an initial angular velocity of 0 deg/sec, and a length of 0.5 m. With the FNC optimized by the conditions given above, we also test its ability to control other initial conditions, such as:

- **Initial angle (deg)**: ±10, ±20, ±30, ±40, ±50, ±60, ±70, ±80
- **Initial angular velocity (deg/sec)**: ±10, ±20, ±30, ±40, ±50, ±60, ±70, ±80
- **Pole length (m)**: 0.25, 0.5, 1.0

We have tested the applicability of the optimized FNC to control of the pendulum under different initial conditions. The success rate shows the number of successes in balancing the pole when the optimized FNC is applied to different initial conditions (total 768 cases). Notice that GA-optimizer utilizes only a single training data set. Fig. 10 shows an example of the performance of the optimized FNC with an initial angle of 10 degrees. The FNC provides approximately a 90.0% success rate and balances
the pole within 3.5 sec. in most cases. Fig. 11 shows the optimized fuzzy membership functions and fuzzy rules.

**Conclusions**

We have presented a Computer-Aided System Design environment which imitates natural evolutionary design. As control tasks become ever more demanding, more robust controller design methodologies are needed. A GA optimizer, which employs naive, but robust, search strategies, offers a promising technology that supports optimization of the parameters of fuzzy logic and other parameterized non-linear controllers. The results reported here show how the GA optimizer for the FNC affords more reliability in global optimization than does an adaptive neural net approach [7].

Our subsequent work [9] has developed new techniques for GA/simulation-based design including (a) the intentional use of noise to more efficiently sample continuous parameter spaces, and (b) a hierarchical architecture that enables search based on increasing levels of abstraction. Still, we are only on the threshold of understanding what are the ultimate limits of GA-based optimization. Nevertheless, results of executing this CASD methodology on the CM-5 supercomputer suggest that the use of GAs in control system design is feasible right now. With the advent of high performance parallel computing accessible to all control engineers, such application will become widely adoptable.

**References**


Jinwoo Kim received the B.S. degree in electronics engineering in 1989 from Sogang University, Seoul, Korea, and the M.S. and Ph.D. degrees in electrical and computer engineering in 1991 and 1994, respectively, from the University of Arizona, Tucson. From 1990 to 1993, he was a research assistant in the University of Arizona/NASA Space Engineering Center, where he worked on developing an intelligent real-time control system. After his graduation, he joined the NSF HPCC Grand Challenge Application Project as a research associate in the electrical and computer engineering department at the University of Arizona. His research interests include parallel/distributed processing, artificial intelligence, genetic algorithms, fuzzy logic, neural networks, hierarchical real-time control, object-oriented modeling and simulation, and intelligent information systems.

Yoonkeon Moon received the B.S. and M.S. degrees in control and instrumentation engineering from Seoul National University, Seoul, Korea, in 1982 and 1984, respectively. From 1984 to 1989, he worked at the Central Research Lab of Goldstar Industrial Systems Company in Korea as a research engineer. He is pursuing the Ph.D. and working for the NSF HPCC Grand Challenge Application Project as a research assistant in the Electrical and Computer Engineering Department at the University of Arizona, Tucson. His research interests include genetic algorithms, fuzzy rule-based systems, parallel/distributed processing, and discrete event system modeling and simulation.

Bernard P. Zeigler is professor of electrical and computer engineering at the University of Arizona, Tucson. He received his B. Eng. Phys from McGill University in 1962, his M.S.E.E. from MIT in 1964, and his Ph.D. from the University of Michigan in 1969. He has published more than 200 journal and conference articles in modeling and simulation, knowledge-based systems, and high-autonomy systems. His first book, Theory of Modeling and Simulation (Wiley, 1976), is regarded as one of the foundational works in the field. A second book, Multifaceted Modeling and Discrete Event Simulation (Academic Press, 1984), was given the outstanding simulation publication award by TIMS College on Simulation in 1988. Concepts developed in earlier work are implemented in the DEVS simulation environment and applied to high-autonomy issues in the latest book, Object-Oriented Simulation with Hierarchical, Modular Models; Intelligent Agents and Endomorphic Systems, published by Academic Press, Boston, 1990. Zeigler's research has been supported by federal agencies including NSF, NASA, USAF, and the U.S. Army, as well as industrial sponsors including Siemens, McDonnell Douglas, and Motorola. He is currently heading a multidisciplinary team to demonstrate an innovative approach to massively parallel simulation of large-scale ecosystem models within NSF's HPCC Grand Challenge Initiative.