Evolution of Fuzzy Controllers with Multi–Chromosomal Representation of Membership Functions

Markus Spitzlinger

Department of Computer Science
University of Salzburg
A–5020 Salzburg, Austria
Email: mspitzl@cosy.sbg.ac.at

Helmut A. Mayer

Department of Computer Science
University of Salzburg
A–5020 Salzburg, Austria
Email: helmut@cosy.sbg.ac.at

***Abstract*— In this work we present experiments evolving the membership functions of a fuzzy controller for the inverted pendulum problem. Specifically, the conventional representation of membership functions on a single chromosome is compared to a genotype encoding with multiple chromosomes in an evolutionary algorithm. In this context we present chromosome shuffling, a genetic operator recombining complete chromosomes based on biological evidence. The hypothesis that the multi–chromosomal representation increases the generalization capability of the fuzzy controller, is tested by measuring the controllers' performance on test cases they have not seen during their evolutionary construction.**

I. Introduction

The concepts of fuzzy logic and fuzzy reasoning allow the construction of controllers for complex control tasks being hard to solve for conventional controllers. The use of linguistic variables and simple `if-then` rules enables humans to easily understand the mechanics of the fuzzy controller. However, for the construction of rule bases and membership functions precise knowledge of the problem domain supplied by human experts is of utmost importance, as the fuzzy system cannot learn from data describing the problem. Automatic construction of fuzzy controllers (capable of learning from data) has been mainly achieved by integrating fuzzy and neural systems (*Neuro–Fuzzy Systems*) [1], or fuzzy and evolutionary systems (*Genetic–Fuzzy Systems*) [2], [3].

In this paper we investigate the evolution of membership functions for a fuzzy controller for the well–known *Inverted Pendulum* task. Specifically, we are interested in the multi–chromosomal representation of the triangular membership functions. We compare representations, where all membership functions of all linguistic variables are encoded (conventionally) on a single chromosome, with a genotype consisting of multiple chromosomes, each of them representing the membership functions of a single linguistic variable. In both cases, no expert knowledge is needed for the construction of the membership functions. The survival of the solutions depends only on their phenotypic fitness based on the controller's performance.

The biological motivation for the use of multiple chromosomes comes from *Meiosis*, a complicated cell division process involving sexual reproduction. A maternal and a paternal set of chromosomes (humans have 23 arranged in *Diploid Sets*, i.e., each chromosome occurs in two homologous variants) is combined into one cell. Homologous chromosomes of father and mother align in a phase called *Synapsis*, and parts of the genetic code can be exchanged by crossing over at certain sites. Usually, 1 to 8 crossover points can be identified on one chromosome [4]. This process is the model for the recombination operator in EAs; however, a very interesting step occurs after crossover. The two recombined chromosomes (actually, four, because of diploidy) separate randomly to different areas of the cell. Thus, there is an additional shuffling of genetic material at the level of chromosomes. In humans, this process allows for $2^{23} \approx 8.4 \times 10^6$ different chromosome combinations.

In the realm of artificial evolution we term this process *Chromosome Shuffling* and model it simply by exchanging the chromosomes of the two parents with a probability of $p_s = 0.5$. As is the case in nature, only homologous chromosomes (describing the same variables of the solution with identical representation) are shuffled, which allows the co–existence of different chromosomes within an individual. E.g., a solution could be encoded by a bitstring, an integer and a real chromosome. However, in this work we are concerned with multiple chromosomes having identical structure (real values encoded as bitstrings).

From above statements potential benefits of multi-chromosomal representations can be identified. First, a complex problem can be decomposed into sub–problems being encoded in corresponding chromosomes. Complete solutions of a sub–problem can be exchanged by means of chromosome shuffling without the disruptive effects of crossover. Second, each chromosome can be encoded using a representation adapted to the specific sub–problem. Third, chromosome shuffling could induce improved generalization capabilities of solutions, as a "specialist" chromosome that contributes to good solutions in a few individuals only will be quickly weeded out by evolution. A more general solution of the sub–problem being successful in a great variety of individuals will have increased chances of survival.

A. Related Work

Pierrot and Hinterding (1997) presented an investigation of the use of multi–chromosomes to solve an allocation problem by means of an EA. 500 goods are to be produced on three machines, where each machine has specific fixed and variable costs. The fixed cost is incurred only, if the machine is utilized for the production of goods. The variable costs are defined per good produced on a specific machine. The main idea is to specify the usage of machines on one chromosome, while the variables on the second chromosome encode the number of goods to be produced on the corresponding machine. It has been found that a multi–chromosomal representation has the potential to improve solutions, but it should be noted that no attempt has been made to adjust mutation and crossover rates in the experiments with single and multiple chromosomes. As a consequence, the different mutation and crossover rates have been identified as main source of the improvements of the solution encoded on multiple chromosomes [5].

II. Fuzzy Controller Evolution

In this work we are concerned with the evolution of membership functions of triangular shape.The test bed for evolutionary experiments using single and multiple chromosomes to represent the membership functions is the well–examined inverted pendulum problem [7].

A. The Inverted Pendulum Problem

The inverted pendulum (Figure 1, left) can be described by the following nonlinear differential equation [7] (pp. 104–107):

$$M\frac{d^2}{dt^2}x + m\frac{d^2}{dt^2}(x + l\,\sin\phi) = f, \tag{1}$$

where M is the mass of the cart, m is the mass of the pendulum, l is half the length of the pendulum, x is the cart position, ϕ is the angle and f is the force. For the case of small angles ϕ Equation 1 can be approximated by

$$Ml\frac{d^2\phi}{dt^2} = (M + m)g\phi - f, \tag{2}$$

and

$$M\frac{d^2x}{dt^2} = f - mg\phi, \tag{3}$$

where g is *Newton's* gravitation constant. For small simulation cycle times t_s the differentials can be substituted by differences allowing fast computation of $\phi(t)$ and $x(t)$.

The state variables of the system are ϕ, the angular velocity $\omega = \frac{d\phi}{dt}$, and the control force f. Depending on the system state $(\phi(t), \omega(t))$ a controller has to apply an appropriate force so as to keep the pendulum in a (nearly) vertical position. We now take a closer look at the details of a fuzzy controller for the problem at hand.

B. Expert Rules and Membership Functions

The inverted pendulum problem is an excellent example for both, the conceptual simplicity of a fuzzy controller, and the difficult task of determining the exact values of its parameters. For comparisons with the evolved controllers we employed the linguistic rules and membership functions of an expert controller given in [8]. It should be noted that for the inverted pendulum problem basic problem knowledge is sufficient to formulate reasonable fuzzy rules. Hence, the term “expert controller” in this case simply indicates its construction by humans. The linguistic term set is defined as {Negative Medium (NM), Negative Small (NS), Zero (ZR), Positive Small (PS), Positive Medium (PM)}. The trapezoidally shaped membership functions (Figure 1, right) are scaled to an interval specific to each variable as follows: $\phi \in [-1, 1]$ radians, $\omega \in [-6, 6]$ radians per second, and $f \in [-21, 21]$ Newton.

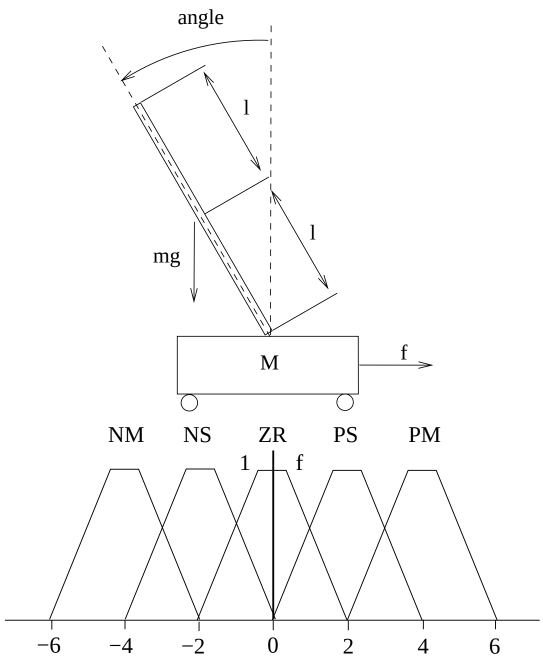

Fig. 1. Inverted Pendulum (top) and the domain partition of the angular velocity ω (bottom).

The seven expert rules are given by

(IF ϕ is NM AND ω is ZR THEN f is NM)
(IF ϕ is NS AND ω is NS THEN f is NS)
(IF ϕ is PS AND ω is NS THEN f is ZR)
(IF ϕ is NS AND ω is PS THEN f is ZR)
(IF ϕ is ZR AND ω is ZR THEN f is ZR)
(IF ϕ is PS AND ω is PS THEN f is PS)
(IF ϕ is PM AND ω is ZR THEN f is PM)

When evolving a fuzzy controller, we use the same set of expert rules and let evolution determine the domain partitions of the linguistic variables within the given intervals.

C. The Fitness Function

In artificial evolution the fitness function is the only problem–specific feedback to the EA. Hence, the assessment of the controller's quality should exactly focus on the goal of the control system, which in our case is to keep the pendulum at an angle of $\phi = 0$, i.e., to minimize the average of the absolute value of ϕ for a number of test cases differing in initial states of the pendulum. The fitness function of a fuzzy controller k is defined as

$$F_k = \frac{1}{mn} \sum_{i=1}^{m} \sum_{j=1}^{n} |\phi_{k,i,j}|, \tag{4}$$

where m the number of test cases, and n the number of simulation cycles for each test case.

D. Membership Function Genotype

In contrast to the expert controller's membership functions being of trapezoidal shape, we set up evolution to generate triangular membership functions, as test runs evolving trapezoidal shapes showed that the resulting shapes often were trapezoids degenerated to triangles. However, we can easily evolve any polygonal shape with the presented system. The triangles are encoded as bitstrings in a straight–forward manner. We only encode the x–value of the triangles and fix the y-values (degree of membership) to 0.0 for the triangle's base points, and to 1.0 for the top. In order to avoid illegal solutions, the three x–values have to be sorted, when decoding the membership functions.

Figure 2 shows the general organization of a population of individuals described by multiple chromosomes encoding the membership functions. Each individual of the population is a complete fuzzy controller.

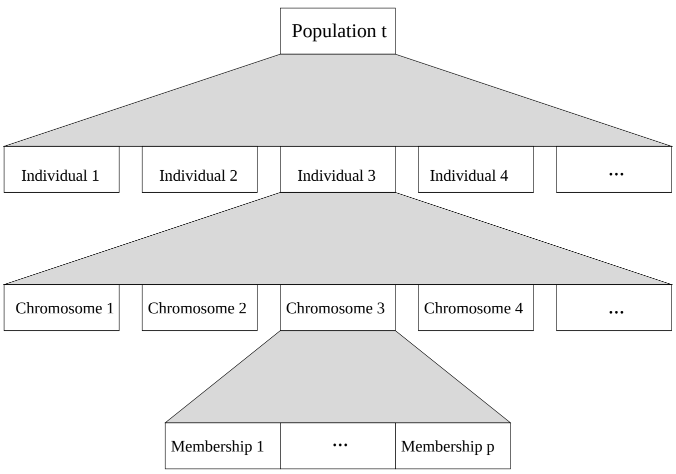

Fig. 2. Details of a population of fuzzy controllers evolving membership functions with a multi–chromosomal organization.

In our test problem the five membership functions of each variable are encoded on a separate chromosome, hence, the number of chromosomes is equal to the number of variables (three). With single chromosome representation we simply concatenate all chromosomes to a single bitstring. During evolution the main difference of the two representations is induced by chromosome shuffling, which alters an individual only in the case of multiple chromosomes. Moreover, crossover is acting more locally, when using the multi–chromosomal representation. In order to only investigate the influence of above differences, we adjusted mutation and crossover rates to the given representation as described in the next section.

III. Experimental Setup

The technical platform for the evolution of fuzzy controllers is the *fuzJEN* system, a Java framework for evolutionary construction of fuzzy controllers, developed by the authors. The following parameters have been used with all the experiments in this paper.

Each point (positive or negative x–value) of the membership functions is encoded using eight bases. This results in a chromosome length of $5 * 3 * 8 = 120$ bases in case of multiple–chromosomes, and a single chromosome length of $3 * 120 = 360$ bases. The mutation rate $p_m = \frac{1}{l}$, based on the length l of the single chromosome, is identical for both representations.

EA Parameters: Population size = 50, Generations = 100, Runs = 50, Mutation rate $p_m = 0.0028$, Crossover = 2–point, Crossover rate = adjusted, Shuffle rate $p_s = 0.5/0.0$, Selection method = Binary tournament.

The adjustment of the crossover rate starts with the observation that a unique crossover rate for both representations results in different numbers of crossover operations. Thus, we adjusted the crossover rate to each representation by calculating the survival probability p_{NA} that an individual is not altered by crossover including the impact of chromosome shuffling.

In the most general case of multiple chromosomes with shuffling and crossover the survival probability is

$$p_{NA} = [p_s^c + (1 - p_s)^c](1 - p_c)^c, \tag{5}$$

where p_s is the shuffle rate, p_c is the crossover rate, and c is the number of chromosomes. The first term takes into account the probabilities of shuffling exchanging all or no chromosomes, while the second term describes the probability of crossover not to occur.

In the experiments (Section IV) we investigate the following variants of representation and operators: single chromosome with crossover S, multiple chromosomes with crossover M_C, multiple chromosomes with shuffling M_S, and multiple chromosomes with crossover and shuffling M_{CS}. If shuffling is used, the shuffle rate is fixed at a biologically plausible rate of $p_s = 0.5$. The crossover rates for the specific encodings are adjusted based on the survival probability p_{NA}. As M_{CS} cannot be properly adjusted to the set S_1, M_C, and M_S, we run a second single chromosome S_2 adjusted to M_{CS}.

Inverted Pendulum Parameters: $M = 0.5\,kg$, $m = 0.2\,kg$, $l = 0.3\,m$, Simulation cycle time $t_c = 150\,ms$, Simulation cycles $n = 30$, Training cases m = 6 with initial angles ϕ

in $\{6°, -6°, 13°, -13°, 25°, -25°\}$, Controller activated after first cycle.

Each experiment is repeated 50 times using a different random seed for each run. The experiments have been devised in order two answer two main questions. The first one is simply, if the multi–chromosomal representation allows the evolutionary process to find well–performing fuzzy controllers at all. The second question is concerned with the possible phenotypical differences of multi–chromosome controllers.

IV. Experimental Results

In Table I the fitness of the evolved fuzzy controllers utilizing single and multiple chromosomes are compared.

TABLE I

STATISTICAL PARAMETERS OF THE FITNESS OF FUZZY CONTROLLERS EVOLVED WITH DIFFERENT GENETIC REPRESENTATIONS AND OPERATORS (AVERAGED ON 50 RUNS).

Genotype/Operators	Avg	StdDev	Best	Worst
S_1 ($p_c = 0.75$)	2.84	0.52	1.88	4.59
M_C ($p_c = 0.37$)	2.84	0.47	2.07	4.60
M_S ($p_c = 0.0$)	4.19	3.54	2.07	16.74
S_2 ($p_c = 0.94$)	2.72	0.58	1.86	5.18
M_{CS} ($p_c = 0.37$)	2.89	0.56	2.04	4.79

It can be seen that most of the representations with adjusted crossover operators generate very similar results. The only exception are the runs using multiple chromosomes with shuffling without crossover (M_S), which sometimes fails to evolve a well–performing controller, as can be seen with the worst fitness of an average angle of 16.7. Though, mere shuffling is in effect a crossover operation (at fixed sites), it is not sufficient to achieve the results of the compared encodings, as optimization of each chromosome only relies on mutation.

The results presented in Table I are based on the controller's performance on the specific training cases. However, in order to be applicable in the real–world the evolved controller should be able to generalize on test cases not seen during evolution. Thus, we computed the fitness of the best controllers found for each representation with initial angles ϕ in the interval $[-30.0° \dots 30.0°]$ (sampled at $1°$ steps), and compared their performance to the expert controller's (Table II).

TABLE II

FITNESS OF BEST CONTROLLERS ON UNSEEN TEST CASES.

	Expert	S_1	M_C	M_S	S_2	M_{CS}
Fitness	12.42	3.42	2.44	2.41	6.06	3.55

Clearly, all evolved controllers outperform the expert controller, but even more interesting is the improved generalization of controllers evolved with multi–chromosomal representation of membership functions. For a more thorough analysis of generalization capabilities we compare the test fitness of all 50 controllers evolved in each evolutionary run in Table III.

Though, S_1 shows a slightly better test performance than the multi–chromosomal variant M_C not employing chromosome shuffling, the comparison of S_2 with M_{CS} reveals that the addition of shuffling improves the test performance of the evolved controllers. Specifically, the worst controller found in the 50 M_{CS} runs is considerably superior to the respective S_2 counterpart. These results are further evidence that recombination of multiple chromosomes enforces the formation of more general solutions of a sub–problem encoded in a chromosome.

TABLE III

FITNESS OF EVOLVED CONTROLLERS ON UNSEEN TEST CASES (AVERAGED ON 50 RUNS).

Genotype/Operators	Average	StdDev	Best	Worst
S_1 ($p_c = 0.75$)	4.69	2.54	2.34	14.52
M_C ($p_c = 0.37$)	5.42	2.69	2.43	11.70
M_S ($p_c = 0.0$)	6.76	5.00	2.41	23.17
S_2 ($p_c = 0.94$)	4.61	3.35	2.34	18.01
M_{CS} ($p_c = 0.37$)	4.47	1.72	2.25	8.12

V. Summary

We have presented experiments evolving the membership functions of a fuzzy controller for the inverted pendulum problem employing multi–chromosomal representations in the evolutionary process. It has been found that multiple chromosomes enable evolution of controllers of similar fitness, when compared to evolution employing standard single chromosome representations. Another result supports our hypothesis that recombination of genotypes with multiple chromosomes leads to preferred selection of more general solutions of a sub–problem encoded in a chromosome. Improved generalization could be observed, when comparing the performance of evolved controllers with both representations on test cases not seen during evolution. Continuing work will explore the use of multi–chromosomal representations for the evolution of more complex fuzzy controllers, e.g., for a robot soccer player, and other problem domains.

References

[1] D. Nauck, F. Klawonn, and R. Kruse, *Foundations of Neuro–Fuzzy Systems*. Chichester: John Wiley & Sons, 1997.

[2] O. Cordón, F. Herrera, and M. Lozano, "A Classified Review on the Combination Fuzzy Logic–Genetic Algorithms Bibliography," Department of Computer Science and A.I., University of Granada, 18071 Granada, Spain, Tech. Rep. DECSAI-95129, December 1996.

[3] J. T. Alander, "An indexed bibliography of genetic algorithms with fuzzy logic," in *Fuzzy Evolutionary Computation*, W. Pedrycz, Ed. Kluwer Academic, 1997, pp. 299–318.

[4] H. Lodish, D. Baltimore, A. Berk, S. L. Zipursky, P. Matsudaira, and J. Darnell, *Molecular Cell Biology*, 3rd ed. Scientific American Books, 1995, ISBN 07167-2380-8.

[5] H. J. Pierrot and R. Hinterding, "Using Multi-chromosomes to Solve a Simple Mixed Integer Problem," in *Proceedings of the Tenth Australian Joint Conference on Artificial Intelligence*. Springer, 1997, pp. 137–146.

[6] F. Herrera, M. Lozano, and J. Verdegay, "A Learning Process for Fuzzy Control Rules using Genetic Algorithms," Department of Computer Science and A.I., University of Granada, 18071 Granada, Spain, Tech. Rep. DECSAI-95108, February 1995.

[7] K. Ogata, *Modern Control Engineering*, 2nd ed. Englewood Cliffs, NJ: Prentice–Hall, 1990.

[8] O. Cordón, F. Herrera, and A. Peregrín, "Applicability of the Fuzzy Operators in the Design of Fuzzy Logic Controllers," *Fuzzy Sets and Systems*, vol. 86, no. 1, pp. 15–41, 1997.