Computer Assisted Mechanics and Engineering Sciences, 14: 317–329, 2007.

Fuzzy evolutionary algorithms and neural networks in uncertain optimization problems

Piotr Orantek
Department for Strength of Materials and Computational Mechanics
Silesian University of Technology, Konarskiego 18a, 44-100 Gliwice, Poland

(Received in the final form March 20, 2007)

This paper is devoted to the application of the evolutionary algorithms and artificial neural networks to uncertain optimization problems in which some parameters are described by fuzzy numbers. The special method of global optimization: Two-Stages Fuzzy Strategy (TSFS) for structures in uncertain conditions is proposed. As the first stage of the TSFS the fuzzy evolutionary algorithm is used. As the second stage the local optimization method with neuro-computing is proposed. The presented approach is applied in the identification problems of mechanical structures, in which material parameters and loadings are uncertain. To solve the direct problem the fuzzy boundary element method (FBEM) is used. Several numerical tests and examples are presented.

1. Introduction

In the majority engineering cases it is not possible to determinate exactly all parameters of the system. It is necessary to introduce some uncertain parameters which describe granular character of data [15].

The representation of uncertain values may have different forms. It depends on the physical meaning of the problem. One of the possibilities is to model numbers as "soft numbers" called fuzzy numbers [9, 10, 14, 15]. There are many kinds of fuzzy numbers depending on the level of the fuzzy: (i) interval numbers, (ii) trapezoid numbers, (iii) L-R numbers and many others.

The evolutionary algorithms [1, 16], as the global optimization technique for searching uncertain values, can be applied in finding the fuzzy models [2], fuzzy controllers [11], fuzzy rules [8] and others [9, 13–15]. In such algorithms, the chromosome consists of uncertain genes, which are represented by fuzzy numbers. Therefore, the evolutionary operators are modified for fuzzy types of data.

The artificial neural networks, as the tool of approximation can works with the uncertain values.

In the paper the new conception of connection FEA and neural network is presented. The Two-Stages Fuzzy Strategy (TSFS) as the optimization method is shown. The main idea of creating the TSFS is the coupling of the advantages of FEA and gradient optimization methods aided by neuro-computing.

The TSFS is applied to the identification problems of mechanical structure.

2. The fuzzy evolutionary algorithm

The paper concerns the fuzzy evolutionary algorithm (FEA) with fuzzy operators and fuzzy representation of the data is presented. The chromosomes contain fuzzy genes. Each gene decides about the heredity of one or a few characteristics. The individuals can be modified by means of the fuzzy operators. The evolutionary operators generate new chromosomes. The next step is the operator of the selection. It creates a new generation, which contains better chromosomes. All steps are repeated

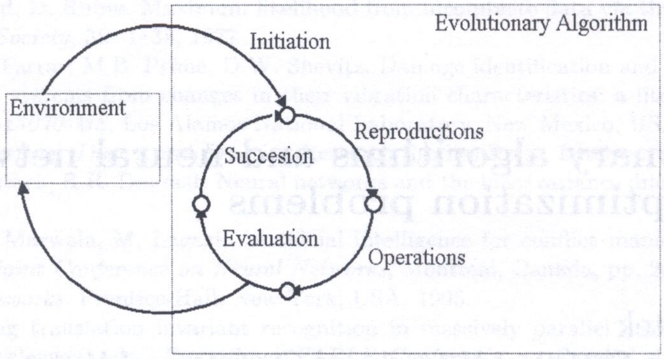

Fig. 1. The flow chart of the evolutionary algorithm

until the stop condition is fulfilled (Fig. 1). In the fuzzy evolutionary algorithm an individual expresses a fuzzy solution. In each generation the fuzzy evolutionary algorithm contains a population of fuzzy solutions. Each solution is evaluated, and as the result a fuzzy value of the fitness function is obtained. The next generation is constructed on the basis of better fuzzy chromosomes of the previous generation. In this case the special types of fuzzy relations are defined. Also the fuzzy types of operators are constructed. Two types of fuzzy mutations can change the selected values of the fuzzy chromosome. Two kinds of crossovers are applied to exchange the selected values between fuzzy chromosomes. As the result of fuzzy operators, fuzzy chromosomes are obtained. One can observe that the next population in the fuzzy evolutionary algorithm is better than the previous one.

2.1. The fuzzy representation of the data

2.1.1. The fuzzy gene

In most cases the evolutionary algorithm has the genes as the real values. The fuzzy algorithm works on the fuzzy data, so the gene should be modified to fuzzy representation. The fuzzy values are particular cases of the fuzzy sets, which are convex and have the continuous membership functions [10]. Some basic arithmetic operators are defined for that representation of the fuzzy values. They do not assure result as the fuzzy value. Other form of the fuzzy values is the L-R form. In this case only the addition and subtraction are defined as the exact solution. Multiplication and division

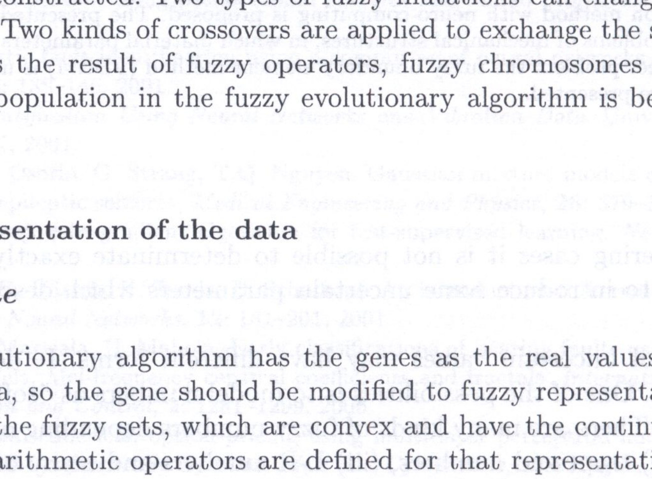

Fig. 2. The fuzzy value and corresponded intervals

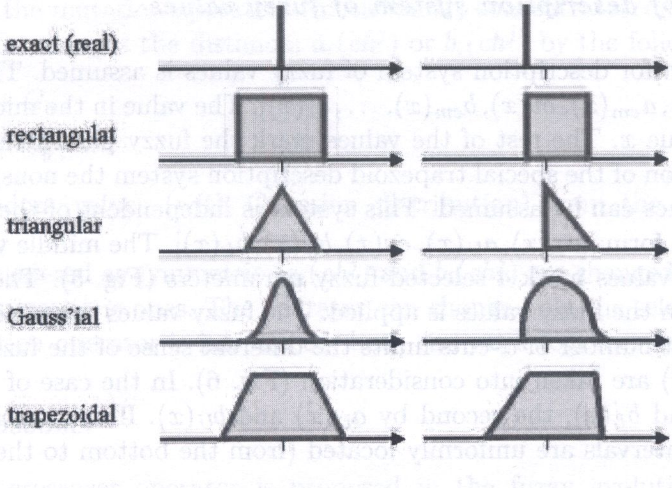

Fig. 3. The selected symmetrical and non-symmetrical forms of the fuzzy values

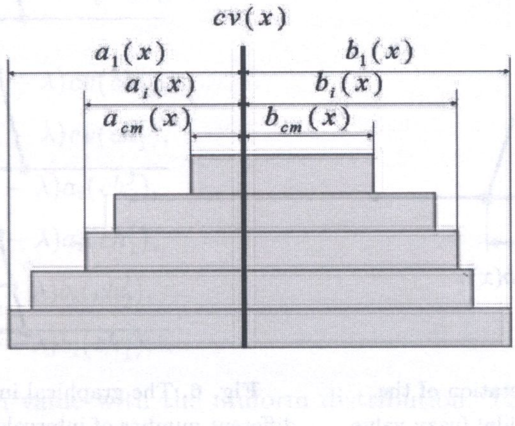

Fig. 4. The gene in the fuzzy evolutionary algorithm

have the approximate character. Also the fuzzy value x can be considered as the set of the interval values, which are stretched on the adequate levels (α-cuts) of the fuzzy value. The number of the α-cuts can be arbitrary. Figure 2 shows an example of the replacement of the fuzzy value using the 5 interval values.

This approach gives some advantages. For each α-cut the very good known interval arithmetic operators are used. It is possible to obtain different forms of the fuzzy values (Fig. 3) due to generation of a few α-cuts and corresponded them interval values $[\underline{x}, \overline{x}]$.

The forms can be symmetric or not symmetric. They describe some characteristic forms of the fuzzy values, and permit to build a new form of the fuzzy value too. Finally, each gene x is expressed as the real value: the central value $cv(x)$ (Fig. 4) and a set of parameters $a_i(x)$ and $b_i(x)$ which define distances between $cv(x)$ and the boundaries of the intervals (Fig. 4) [19]. It is possible to introduce the constraints on the $cv(x)$ and non-symmetric constraints on the widths of the intervals using the central value $cv(x)$. The constraints on the central value $cv(x)$ and the fuzzy parameters $a_i(x)$ and $b_i(x)$ are defined as follows,

$$cv^{\min}(x) \leq cv(x) \leq cv^{\max}(x) \tag{1}$$

$$a_i^{\min}(x) \leq a_i(x) \leq a_i^{\max}(x) \tag{2}$$

$$b_i^{\min}(x) \leq b_i(x) \leq b_i^{\max}(x) \tag{3}$$

where indexes 'min' and 'max' mean the maximum and minimum values.

2.1.2. *Convention of description system of fuzzy values*

The special convention for description system of fuzzy values is assumed. The system contains the parameters: $[a_1(x), \ldots, a_{cm}(x), cv(x), b_{cm}(x), \ldots, b_1(x)]$. The value in the middle is the central value $cv(x)$ of the fuzzy value x. The rest of the values mark the fuzzy parameters of the fuzzy value x (Fig. 4). By introduction of the special trapezoid description system the nonsymmetrical constraints imposed for fuzzy values can be assumed. This system is independent of the number of the α-cuts.

The system has the form: $[a_L(x), a_U(x), cv(x), b_U(x), b_L(x)]$. The middle value means the central value $cv(x)$. The rest values mark 4 selected fuzzy parameters (Fig. 5). The trapezoid description system for construction the fuzzy values is applied. The fuzzy values can consist of different number of α-cuts. The different number of α-cuts limits the different sense of the fuzzy parameters. For one α-cut, $a_L(x)$ and $b_L(x)$ are taken into consideration (Fig. 6). In the case of two α-cuts, the first is described by $a_L(x)$ and $b_L(x)$, the second by $a_U(x)$ and $b_U(x)$. If the number of α-cuts is bigger than three, the next intervals are uniformly located (from the bottom to the top) (Fig. 6).

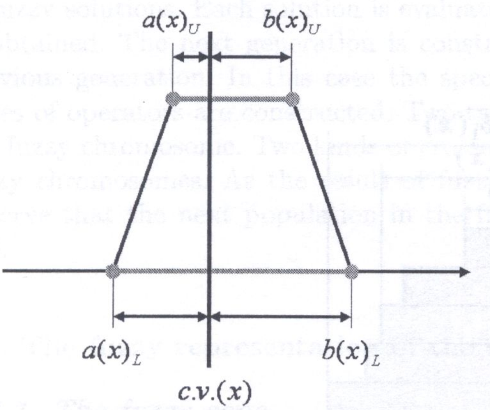

Fig. 5. The graphical interpretation of the description system of the trapezoidal fuzzy value

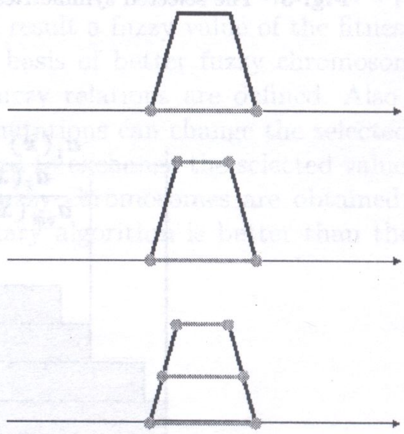

Fig. 6. The graphical interpretation of generating the different number of intervals on basis the trapezoidal system

2.1.3. *The fuzzy chromosome*

The main difference between the fuzzy evolutionary algorithm and the real coded evolutionary algorithm consists in the representation of the data. The chromosome consists of fuzzy genes and can be considered as the fuzzy vector and is considered as potential fuzzy solution of the problem.

2.2. The fuzzy evolutionary operators

2.2.1. *The fuzzy mutation*

Two new types of the mutation operators in the presented algorithm are applied. In both cases the modified gene ch^j is randomly selected from the chromosome $ch = [ch^1, ch^2, \ldots, ch^j \ldots, ch^n]$. In the first type of the mutation (mutation I) the central value $cv(ch^j)$ of the j-th fuzzy value ch^j is modified. The operator is expressed by the following equation,

$$cv(ch^{j*}) = cv(ch^j) + G_{cv}\,, \tag{4}$$

where G_{cv} – random value (with Gaussian distribution) from the range given by Eq. (1), $j = 1, \ldots, n$, is the number of the gene.

The second type of the mutation operators (mutation II) concentrates/deconcentrates the fuzzy value ch^j. The mutation changes the distances $a_i(ch^j)$ or $b_i(ch^j)$ by the following equation,

$$a_i(ch^{j*}) = a_i(ch^j) + G_a\,, \tag{5}$$

$$b_i(ch^{j*}) = b_i(ch^j) + G_b\,, \tag{6}$$

where G_a i G_b – random values (with Gaussian distribution) from the ranges given by Eqs. (2) and (3).

This operator is considered as symmetric ($a_i(ch^j)$ and $b_i(ch^j)$ are changed using by means of the same value), and non-symmetric ones. The operator can change only the selected α-cut. Therefore, two types of the mutation operator is introduced, both can work together or independently.

2.2.2. The fuzzy crossover

The fuzzy arithmetic crossover operator is proposed in the fuzzy evolutionary algorithm. The crossover creates two children individuals ch_1^* and ch_2^* on the basis of the two parent chromosomes ch_1 and ch_2.

The selected parameters of the j-th genes of the children chromosomes are expressed by the following equations,

$$cv(ch_1^{j*}) = \lambda cv(ch_1^j) + (1-\lambda)cv(ch_2^j), \tag{7}$$

$$cv(ch_2^{j*}) = \lambda cv(ch_2^j) + (1-\lambda)cv(ch_1^j), \tag{8}$$

$$a_i(ch_1^{j*}) \;=\; \lambda a_i(ch_1^j) + (1-\lambda)a_i(ch_2^j), \tag{9}$$

$$a_i(ch_2^{j*}) \;=\; \lambda a_i(ch_2^j) + (1-\lambda)a_i(ch_1^j), \tag{10}$$

$$b_i(ch_1^{j*}) \;=\; \lambda b_i(ch_1^j) + (1-\lambda)b_i(ch_2^j), \tag{11}$$

$$b_i(ch_2^{j*}) \;=\; \lambda b_i(ch_2^j) + (1-\lambda)b_i(ch_1^j), \tag{12}$$

where $\lambda \in [0,1]$ is a random value with the uniform distribution. The crossover operator like the mutation can change only the selected α-cuts.

2.2.3. The fuzzy selection

The last modified operator for the fuzzy values is the selection operator. This operator is constructed on the basis of the well known tournament selection. In this selection the fitness function values f are compared, and the better chromosome wins more often. Therefore the special strategy of comparison of two fuzzy values is proposed. The explanation of the proposed method is presented in the following example.

At the beginning consider two real values (fitness functions): $eval_1 = f(ch_1)$ and $eval_2 = f(ch_2)$. In the real value case the better one wins if the random condition is true: $\beta < \beta_{rand}$, where β – the random value with uniform distribution, β_{rand} – the win parameter, which takes the value close to 1 (for example $\beta_{rand} = 0.95$). It is easy to observe that the better individual wins more often, but it is not certain. In this work this approach is developed for the fuzzy case. Consider two fuzzy values $eval_1 = f(ch_1)$ and $eval_2 = f(ch_2)$. Both values have the same number of α-cuts. The minimization problem is considered. At the beginning the following condition is checked,

$$cv(eval_1) = cv(eval_2). \tag{13}$$

If the condition (13) is fulfilled, the width condition (a_i and b_i) is checked for each α-cut. The fuzzy value, which has the bigger width (for example: $eval_1$), takes the value $\beta_1 = 0.4$ and other fuzzy value takes value $\beta_2 = 0.6$ (values β_1 and β_2 play the role parameter β for the real value case). If both widths are identical, β_1 and β_2 take the value 0.5. This way promotes more concentrate values.

If the condition (13) is not fulfilled, there are two possibilities. Consider one of the possibilities:

$$cv(eval_1) < cv(eval_2). \tag{14}$$

In the second possibility the way is similar, but values $eval_1$ and $eval_2$ are treated on the contrary.

Therefore, if the condition (14) is fulfilled, win parameters take the next values. Parameter β_1 takes a value close to 1 (in this work β_1 is equal to 0.95), parameter β_2 takes a value close to 0 (in this work β_2 is equal to 0.05, $\beta_1 + \beta_2 = 1$). Next, the conditions are checked,

$$cv(eval_2) - a_i(eval_2) \le cv(eval_1) + b_i(eval_1), \tag{15}$$

$$cv(eval_2) - a_i(eval_2) \le cv(eval_1), \tag{16}$$

$$cv(eval_2) - a_i(eval_2) \le cv(eval_1) - a_i(eval_1), \tag{17}$$

$$cv(eval_2) \le cv(eval_1) + b_i(eval_1), \tag{18}$$

$$cv(eval_2) + b_i(eval_2) \le cv(eval_1) + b_i(eval_1). \tag{19}$$

In the case of fulfilling each condition (15)–(19), the parameter β_1 is decreasing by $\Delta\beta_1$ and the parameter β_2 is increasing by $\Delta\beta_2$. In this paper both parameters take the values: $\beta_1 = \beta_2 = 0.05$.

It is possible to observe that if no one condition is fulfilled, both intervals do not have the common part. Fulfillment of the conditions corresponds to increasing the common part of both intervals.

This process is repeated for all α-cuts. The selection based on conditions (15)–(19) generates bigger probability of the survival of a much better adapted individual.

In the proposed method the starting values of parameters and increasing and decreasing values are arbitrary. It is possible that for other values the algorithm can work better.

2.2.4. The fuzzy fitness function

One of the most important step of the evolutionary algorithm is the evaluation of the fitness function. If the design variables are the real numbers, the fitness function result is also real value. In the case of the fuzzy evolutionary algorithm, the problem of evaluating the fitness function can be more complicated. A few ways of the estimation of the results in this case are possible.

If the fitness function is the explicit function of fuzzy design parameters, its value can be computed taking into account the extension principle. Unfortunately, in many cases the fitness function evaluation can be done after solving the fuzzy boundary-value problem. The fuzzy boundary-value problems can be solved by means of the boundary element method FBEM [6, 18] or the finite element method FFEM [7].

The key problem is to solve the fuzzy algebraic equations which are obtained after the numerical discretization of the boundary-value problem using the FBEM or the FFEM. There are a few approaches to overcome these difficulties.

One of the more general approaches is based on the α-cuts and the interval representation of the solutions. More detail description of this idea can be found in [18].

3. The multilevel artificial neural network

3.1. The artificial neural network

Consider two types of the artificial neural networks (Fig. 7), with sigmoid (MLP) and radial (RBF) active functions. The fitness function is modeled close to the optimum by the parabolic function for each design variable, therefore one hidden layer in the network (MLP) is sufficient. In the case of the network (RBF) one "hidden" layer (basis functions) is always assumed.

The number of neurons in the input layer is equal to the number of design variables of the fitness function. In the output layer there is only one neuron, its output value plays the role of the fitness function.

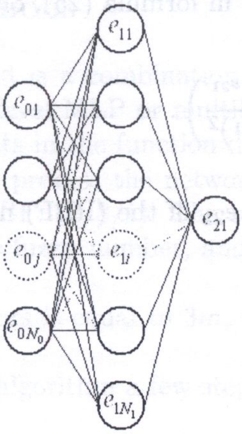

Fig. 7. The schema of neural network

The number of neurons in hidden layer depends on the degree of difficulty of the function.

The output values of neurons in the input layer take the values of design variables (in both cases MLP and RBF). The output value of neurons (McCulloch and Pitts model with sigmoid active function) in the hidden layer in this case network (MLP) is expressed by [4]

$$e_{ik} = f(s_{ik}) = \frac{1}{1 + e^{-s_{ik}}} \tag{20}$$

where

$$s_{ik} = \sum_{n=1}^{N_{i-1}} e_{i-1\,n} w_{i-1\,njk} + ww_{ik} \tag{21}$$

and where e – output values of neurons in previous layers, w – weights.

The output value of neurons in the hidden layer in the case of the network (RBF) is expressed by

$$e_{1i} = f(u_i) = e^{-0.5u_i} \tag{22}$$

where

$$u_i = \sum_{n=1}^{N} \left(\frac{e_{ik} - t_k^i}{\sigma_k^i} \right)^2 \tag{23}$$

and where t_i – the centre of i-th radial function, σ_i – width of i-th radial function.

The output value of the neuron in the output layer in network (MLP) is computed like in the previous, the hidden layer, however in the network (RBF) as follows,

$$e_{21} = \sum_{n=1}^{N} W_i e_{1i} + W_0 \,. \tag{24}$$

3.2. The sensitivity of the neural networks

The sensitivity of the output signal e_{21} of the (MLP) network in the some z-th input value e_{0z} is expressed by

$$\frac{de_{21}}{de_{0z}} = \sum_{i=1}^{I_1} \frac{ds_{1n_1}}{de_{0z}} \frac{de_{1n_1}}{ds_{1n_1}} \frac{ds_{2n_1}}{de_{1n_1}} \frac{de_{21}}{ds_{21}} \,. \tag{25}$$

Taking into account Eqs. (20) and (21) in formula (25), one obtains

$$\frac{\mathrm{d}e_{21}}{\mathrm{d}e_{0z}} = \sum_{i=1}^{I_1} \left(\frac{w_{0z1n_1} e^{-s_{1n_1}}}{(1+e^{-s_{1n_1}})^2} \frac{w_{1n_1 21} e^{-s_{21}}}{(1+e^{-s_{21}})^2} \right). \tag{26}$$

The sensitivity of the output signal e_{21} of the (RBF) network in the some z-th input value e_{0z} is expressed by

$$\frac{\mathrm{d}e_{21}}{\mathrm{d}e_{0z}} = \sum_{i=1}^{I_1} \frac{\mathrm{d}s_{1n_1}}{\mathrm{d}e_{0z}} \frac{\mathrm{d}e_{1n_1}}{\mathrm{d}s_{1n_1}} \frac{\mathrm{d}s_{2n_1}}{\mathrm{d}e_{1n_1}} \frac{\mathrm{d}e_{21}}{\mathrm{d}s_{21}}. \tag{27}$$

Taking into account Eqs. (22), (23) and (24) in formula (27) one obtains

$$\frac{\mathrm{d}e_{21}}{\mathrm{d}e_{0z}} = \sum_{i=1}^{I_1} \left(-w^{n_1} \exp\left(-\frac{1}{2} u^{n_1} \right) \frac{x_{0z} - t_{0z}^{n_1}}{(\sigma_{0z}^{n_1})^2} \right) \tag{28}$$

The formulas (26) and (28) will be used in local gradient method, which is presented in next section.

3.3. The multilevel artificial neural network

The special multilevel artificial neural network (Figs. 8 and 9) is used as the approximation method of the fuzzy problem. Each level corresponds with a selected parameter of the fuzzy number and is modeled as a MLP or RBF network.

Number of levels of multilevel neural network depends on the number of α-cuts of the fuzzy number and is equal to $1 + 2cm$, where cm – the number of α-cuts.

The central level of the multilevel neural network corresponds with the central value of the fuzzy number (red color, Fig. 9). The other levels correspond with the fuzzy parameters of the fuzzy number (the blue levels corresponds with the parameters a_i, the green levels correspond with the parameters b_i) (Fig. 9). All levels can be connected with each others (Fig. 8).

The sensitivities of the output signals of presented multilevel network is expressed by Eq. (26) for (MLP) and Eq. (28) for (RBF). If all levels are connected with each others, then formulas are extended. The sums concern all levels of network.

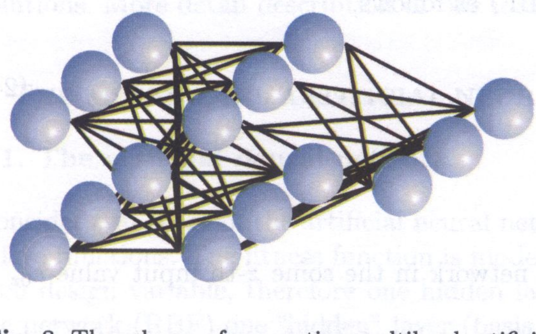

Fig. 8. The scheme of connecting multilevel artificial neural network

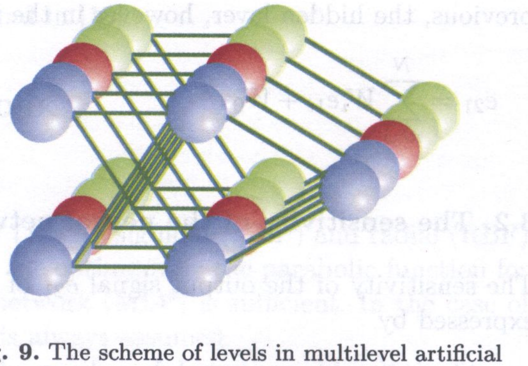

Fig. 9. The scheme of levels in multilevel artificial neural network

4. The local optimization methods

The proposed local optimization method is a combination of the classical gradient method (the steepest descent method) and the multilevel MLP or multilevel RBF network. In the first step of the algorithm a set (cloud) of fuzzy points in the function domain is generated.

In order to perform the optimization process the network is constructed. The multilevel MLP has architecture: (2/10/1), multilevel RBF: (2/8/1). The number of levels of the neural network depends on the number of α-cuts of the fuzzy number, and is equal to $1 + 2cm$, where cm – the number of α-cuts.

The starting number of training vectors is equal to $3m$, where m – the number of fuzzy design variables of the minimizing function.

In each iteration of the optimization algorithm a few steps are performed (Fig. 10).

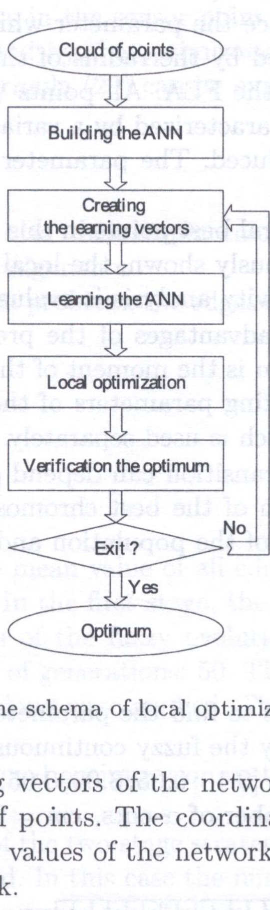

Fig. 10. The schema of local optimization method

In the first step the set of training vectors of the network is created. In the first iteration the set is created on basis of the cloud of points. The coordinates of points (central value and fuzzy parameters) play the role of the input values of the network, the fuzzy fitness values in points play the role of output value of the network.

In the second step the network is trained.

In the next, third step, the optimization process is carried out. The gradient method (the steepest descent method) of optimization is used. The network as the fitness function approximation is used. The gradient formula (26) or (28) is employed in computation. The special kind of the fuzzy gradient is introduced. For all edges of fuzzy values the real sensitivities are calculated. Each parameter can be modified on the basis of this information. If the fuzzyfication degree is fixed, only the central value is modified.

For a point, which is a result of optimization (found in step 3), the actual fitness function is computed.

In the last step the stop condition is checked. In the case, in which the condition is true, the point is treated as the result of the optimization process. If this condition is false, this point is added to the training vector set and the next iteration is carried out (go to step 1).

This method was tested numerically and results were satisfactory.

5. THE TWO STAGE FUZZY STRATEGY

The main idea of the creating the two-stage fuzzy strategy (TSFS) is the coupling of the advantages of evolutionary and gradient optimization methods aided by neuro-computing. The fuzzy evolutionary algorithms can find the global optimum, but it is very time consuming. The fuzzy gradient methods can find the optimum precisely, but they need information about the sensitivity of the objective function.

The TSFS in first stage uses some properties of the fuzzy evolutionary algorithms (FEA).

These algorithms are procedures to search the optimum in the feasible space of solutions.

The FEA generates clusters of fuzzy points. The clusters are positioned closely to the optimum. There is a great possibility that the optimum is the global optimum. There is a risk that the points are located close to the more than one optimum. In this case the second stage (local method) can work unstably. It can be solved in a few ways.

One of the possibilities is to introduce the parameter which describes the maximum size of the cluster. The parameter can be expressed by the radius of the region in domain. The center of the region is equal to the best solution of the FEA. All points which are inside the region, belong to the cloud of points. This approach is characterized by a variable number of training vectors. In this case an alternative parameter is introduced. The parameter defines the maximum number of the points in the cloud.

In the second stage of the TSFS several best points in this region are selected. Then, these points play the role of the cloud and, as previously shown, the local method begins. This method is based on the gradient method, but the sensitivity analysis is evaluated by the neuro-computing.

Therefore, the TSFS combines the advantages of the previous described methods, and avoids their disadvantages. The crucial problem is the moment of the transition from the first stage to the second one. Some experience allows taking parameters of the TSFS, for which the TSFS can find the optimum earlier than the FEA, which is used separately.

In the general case the moment of transition can depend on some parameters of the first stage: (i) – the changes of the fitness function of the best chromosome, (ii) – the size of the clusters of chromosomes, (iii) – the diversification of the population and many others.

6. EXAMPLE OF IDENTIFICATION

The aim of the identification problem is to find the parameters which define the circular defect: x, y and r (Fig. 11). The plate is loaded by the fuzzy continuous traction field $\mathbf{q}$ (Fig. 11). The actual fuzzy parameters of the defect are $x = [0.10; 0.05; 3.00; 0.05; 0.10]$, $y = [0.10; 0.05; 3.00; 0.05; 0.10]$, $r = [0.10; 0.05; 2.00; 0.05; 0.10]$ (the number of α-cuts, $cm = 2$ is assumed).

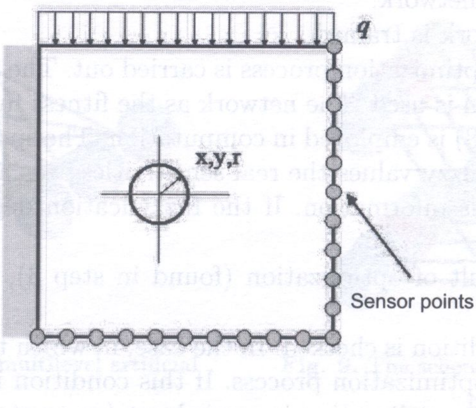

Fig. 11. The plate with the circular defect

The chromosome takes the form: $ch = [ch1, ch2, ch3] = [x, y, r]$, where x, y, r – are the fuzzy values.

The material parameter E also as the fuzzy values is assumed. The 21 sensor points the 21 points on the boundary are selected (Fig. 11). The traction $\mathbf{q}$ is described using the trapezoid value with parallel arms (Fig. 5), therefore each α-cut is determinate as the identical interval $q = [99.8; 100.2]$ (kN). The Young modules for each α-cut as the interval [2e+11 – 2% ; 2e+11 + 2%] are assumed.

In the previous works (the real-coded problem) the minimizing problem was expressed as the minimizing of the displacement functional

$$f = \sum_{i=1}^{n} \int_{\Gamma} \left(|\mathbf{u}(\underline{\mathbf{x}}) - \hat{\mathbf{u}}(\underline{\mathbf{x}})|\right)^2 \delta\left(\underline{\mathbf{x}} - \underline{\mathbf{x}}^i\right) \mathrm{d}\Gamma \tag{29}$$

where $\hat{\mathbf{u}}$ – the measured displacements in the sensor point $\underline{\mathbf{x}}^i$, $\mathbf{u}$ – the computed displacements in the same point for x, y and r generated by the optimization algorithm, n – the number of sensor points, δ – the Dirac function. The formula (29) can be expressed in a more simple form

$$f = \sum_i (u_i - \hat{u}_i)^2 \tag{30}$$

where: $\hat{u}_i$ – the measured displacement, u_i – the displacements computed for the structure with the defects generated by the evolutionary algorithm.

In the case of the fuzzy identification problem, the edges of the intervals $fc \in [\underline{fc}; \overline{fc}]$ are computed as follows,

$$\underline{fc^j} = \sum_i \min\left(\left|\underline{u_i} - \underline{\hat{u}_i}\right| ; \left|\overline{u_i} - \overline{\hat{u}_i}\right|\right) \tag{31}$$

$$\overline{fc^j} = \sum_i \max\left(\left|\underline{u_i} - \underline{\hat{u}_i}\right| ; \left|\overline{u_i} - \overline{\hat{u}_i}\right|\right) \tag{32}$$

The central value of the result, as the mean value of all edges of α-cuts is computed.

The two-stage strategy is applied. In the first stage, the region of the global optimum is finding by means FEA. The following values of the fuzzy evolutionary algorithm parameters have been assumed: population size: 40, number of generations: 50. The best 27 individuals in the whole optimization process as the points of the cloud were applied. The number of fitness function evaluations was equal to 644.

In the second stage the presented local optimization method found the optimum with use of 162 (MLP) or 87 (RBF) iterations.

In order to evaluate the efficiency of the two-stage strategy only the fuzzy evolutionary algorithm for the identification problem was used. In this case the number of fitness function computations is equal to 1435 and it is bigger than the two-stage strategy is used.

The results of identification are presented in Tables 1 and 2. The results at the beginning and after selected stages are presented in Figs. 12–14.

Table 1. The coordinates of the optimum (after 1st stage)

	a_1	a_2	cv	b_2	b_1
x	0.04	0.03	2.97	0.02	0.04
y	0.05	0.03	2.99	0.03	0.03
r	0.08	0.02	1.96	0.01	0.02
y	0.88	0.43	2.42	0.73	0.96

Table 2. The coordinates of the optimum (after 2nd stage)

	a_1	a_2	cv	b_2	b_1
x	0.10	0.05	3.00	0.05	0.10
y	0.10	0.05	3.00	0.05	0.10
r	0.10	0.05	2.00	0.05	0.10
y	0.00	0.00	0.00	0.00	0.00

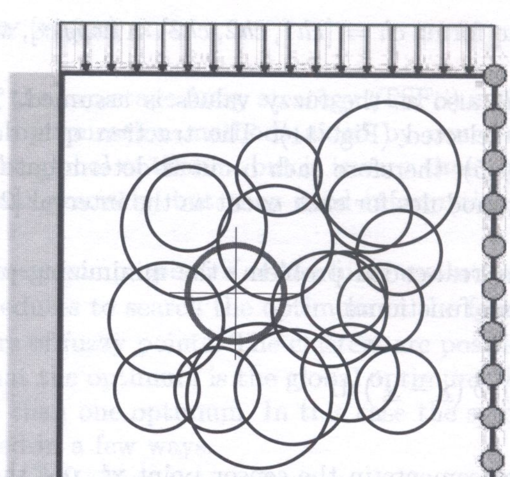

Fig. 12. The results in 1st generation of FEA

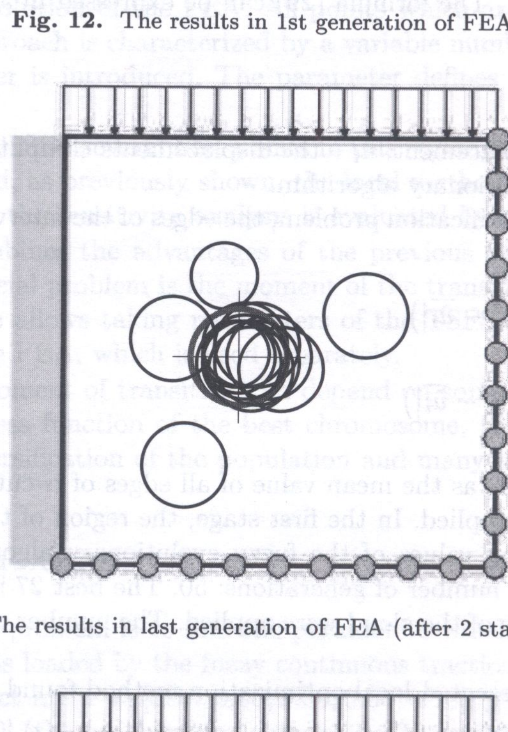

Fig. 13. The results in last generation of FEA (after 2 stage of TSFS)

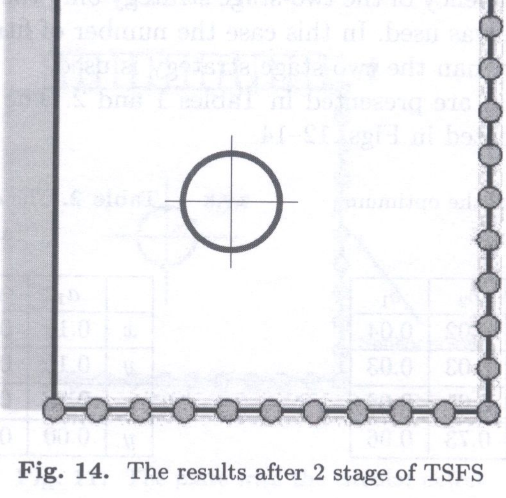

Fig. 14. The results after 2 stage of TSFS

7. Conclusions

An effective Two-Stages Fuzzy Strategy (TSFS) based on the fuzzy evolutionary algorithm and multilevel artificial neural networks has been presented.This approach can be applied in the optimization problems and identification of defects in mechanical structures.

The application of evolutionary algorithms allows avoiding local minimum.

The optimum can be found in less number of iterations due to applied the multilevel neural network for local approximation of the fitness function. In the some tests the time was decreased even to 50%.

The future task is testing the influence of the parameters on the sensitivity of the algorithm: the parameters of the fuzzy evolutionary algorithm (number of individuals, probability of the operators), control parameters [5] of the selection (probability of the comparison of two fuzzy numbers).

In the general case the uncertain conditions have the granular form [3]. The models based on random variables can be used instead of the fuzzy approach presented in this paper.

The granular evolutionary algorithm can be created as a general method for all described models.

References

[1] J. Arabas. *Wykłady z algorytmów ewolucyjnych.* WNT, 2001.

[2] A. Arslan, M. Kaya. Determination of fuzzy logic membership functions using genetic algorithms. *Fuzzy Sets and Systems,* **118**: 297–306, 2001.

[3] A. Bargiela, W. Pedrycz. *Granular Computing: An introduction.* Kluwer Academic Publishers, Boston–Dordrecht–London, 2002.

[4] C.M. Bishop. *Neural networks for pattern recognition.* Amazon, 1995.

[5] T. Burczyński, P. Orantek. Application of neural networks in controlling of evolutionary algorithms. *First Asian-Pacific Congress on Computational Mechanics, APCOM 01,* Sydney, Australia, 2001.

[6] T. Burczyński, J. Skrzypczyk. Fuzzy aspects of the boundary element method. *Engineering Analysis with Boundary Elements,* **19**: 209–216, 1997.

[7] L. Chen, S.S. Rao. Fuzzy finite element approach for vibrating analysis of imprecisely defined systems. *Finite Elements in Analysis and Design,* **27**: 69–83, 1977.

[8] O. Cordon, F. Gomide, F. Herrera, F. Homann, L. Magdalena. Ten years of genetic fuzzy systems: current framework and new trends. *Fuzzy Sets and Systems,* **141**: 5–31, 2004.

[9] E. Czogała, W. Pedrycz. *Elementy i Metody Teorii Zbiorów Rozmytych.* PWN, Warszawa 1985.

[10] J. Kacprzyk. *Zbiory Rozmyte w Analizie Systemowej.* PWN Warszawa, 1986.

[11] C.L. Karr. Design of an adaptive fuzzy logic controller using a genetic algorithm. *Proc. 4th Int. Conf. on Genetic Algorithms,* San Diego, July 13–16, 1991, pp. 450–457.

[12] M. Littman, D. Ackley. Adaptation in constant utility nonstationary environment. *Proc. ICGA'91,* 1994.

[13] W. Pedrycz. Fuzzy evolutionary computing. *Soft Computing,* **2**: 61–72, 1998.

[14] A. Piegat. *Modelowanie i Sterowanie Rozmyte.* Akademicka Oficyna Wydawnicza EXIT, Warszawa, 2003.

[15] D. Rutkowska, M. Piliński, L. Rutkowski. *Sieci Neuronowe, Algorytmy Genetyczne i Systemy Rozmyte.* PWN, Warszawa–Łódź, 1997.

[16] R. Schaefer. *Podstawy Genetycznej Optymalizacji Globalnej.* WUJ, Kraków, 2002.

[17] I. Skalna. *Zastosowanie Metod Algebry Przedziałowej w Jakościowej Analizie Układów Mechanicznych* (praca doktorska). Politechnika Śląska, Gliwice, 2002.

[18] J. Skrzypczyk, T. Burczyński. Theoretical and computational aspects of the fuzzy boundary element methods. In: T. Burczyński, ed., *Advanced Mathematical and Computational Mechanics Aspects of Boundary Element Method,* pp. 351–364. Kluwer, 2001.

[19] L.A. Zadeh. Fuzzy sets. *Information and Control,* **8**: 338–353, 1965.