An Adaptive Interactive Multi-objective Optimization Approach Based on Decision Neural Network

Alia Youssef Gebreel

Abstract—The primary aim of this paper is to integrate neural network with genetic algorithm to solve the interactive multi- objective optimization problems. This work uses the preference information during the optimization process to find one preferred efficient solution. Neural network is used to transform the multi- objective problem into single objective problem based on the initial point. Genetic algorithm starts with the output of the neural network, and applies its operators such as selection, crossover and mutation to generate future solution. If the required solution cannot be produced, the output of genetic algorithm again restarts for providing the preferred solution based on the required value or level of objectives. This approach achieves a great success in terms of both the effectiveness and accuracy for optimization process, but it needs more time with additional objectives and decision variables.

Index Terms— Interactive multi-objective optimization, genetic algorithm (GA), and neural network (NN).

1 INTRODUCTION

MANY real world problems have several criteria with a group of constraints to be optimized. The resulted solutions are called non-dominated, efficient, or Pareto- optimal solutions. Artificial intelligent optimization algorithms have been one of the most popular methods to find multiple trade-off solutions in single simulation. But when decision making is emphasized, the goal of solving a multi-objective optimization problem is referred to supporting a decision maker in finding the most preferred Pareto optimal solution according to his/her subjective preferences. So, the interactive methods use the preference information progressively during the optimization process. The main aspect in these approaches is that the decision maker provides some information during the optimization process from time to time. Such information is the direction of search, weight vector, reference points, and other factors. Since the decision maker is familiar to the optimization process, these techniques become popular in practice. Generally in an interactive method, the decision maker works together with analyst or an interactive computer program [6, 11, 12].
The methods for solving the multi- objective optimization problems can be classified according to information mode as follows:
a) Methods with a priori information:
Decision maker provides global preference information (weights, utility, goal values, . . .). Analyst solves a single objective problem.
b) Methods with progressive information—interactive methods:
Decision maker provides local preference information. Analyst solves local problems and provides current solutions.
c) Methods with a posteriori information:
Analyst provides a non-dominated set. Decision maker provides global preference information on the non-dominated set. Analyst solves a single objective problem.
There are proposed many methods from these categories. Most of the methods are based on trade-offs [15]. But, the proposed algorithm is devoted to get single solution based on three ways.
There are some earlier works that are related to this research. For example, Hong, Zhigang and Ming [2] construct the designer's preference structure model using artificial neural networks with the model parameter vectors as the input and the preference information. The desired output articulates by the designer over representative samples from the Pareto frontier. Then, the optimization problem is solved to search for improved solutions. This model is very effective in capturing different kinds of preference structures of the designer, and it can obtain the Pareto solution that satisfies the designer's preference. But in Javier, Miguel and Xavier [9], the user's information about his/ her preferences is taken into account within the search process. The preference functions convert these objective preferences into numbers. Problems found due to the multimodality nature of a generated single cost index are managed with genetic algorithm. It presented with examples showing its application in engineering design problems. Finally, Jian and Pratyush [10] designed goal programming model for capturing the decision maker's preference information. This model supports the search for the best compromise solutions in multi-objective optimization. The interactive step trade-off method is employed to generate a typical subset of efficient solutions of a multi- objective problem. It identifies and eliminates the possible inconsistent information which may exit in the preferences of decision mak-

Alia Youssef Gebreel is currently Ph.D. student in Department of Operations Research, Institute of Statistical Studies and Research, Cairo University, Egypt.

er. Also, it provides various ways to carry out post-optimality analysis for testing the robustness of the obtained best compromise solutions.
This paper presents an intelligent interactive approach that incorporates neural network with genetic algorithm to optimize multi- objective problems. It is possible to improve the efficiency of the optimization process. Where, GA has the potential of yielding many Pareto optimal solutions in a single simulation. NN is considered as a tool to produce good starting solutions for GA to obtain the required solution.
The paper consists of six sections. Section 2, and 3 present the basic ideas and architecture of genetic algorithm and neural network, respectively. In section 4, the proposed algorithm is produced. Section 5 contains some illustrated examples to evaluate the paper's algorithm. Finally, section 6 concludes the paper with some suggestions for further research.

2 GENETIC ALGORITHM

A genetic algorithm is a heuristic search that mimics the process of natural evolution concept coming from Darwin's theory of evolution.

Fig. 1. Gene stairs

GA is started with a set of solutions (chromosomes) called population. The solutions from one population are taken and used to form a new population. This is motivated by a hope; that the new population will be better than the old one. The selected solutions to form new solution (offspring) are choosen based on their fitness. This process is repeated until some condition (such as maximum number of generations or improvement of the best solution) is satisfied.
There are five phases namely initial population, fitness function, selection, crossover, and mutation.

Initial population:
It begins with randomly generated population of ***n*** chromosomes (that are satisfactory to the problem).

Fitness function:
The fitness function produces the next generation of population. A good fitness function should return better chromosomes. The fitness function gives a score to each chromosome. The probability of being chosen for reproduction is based on decision maker's fitness score.

Selection:
Two pairs are selected at random to reproduce. They are selected based on their fitness function score.

Crossover:
With a crossover probability, offspring are created by exchanges between the parents at the crossover point.

Mutation:
With a mutation probability mutate new offspring at each position in chromosome [4, 13].
The flow chart of GA is presented in **Fig. 2**.

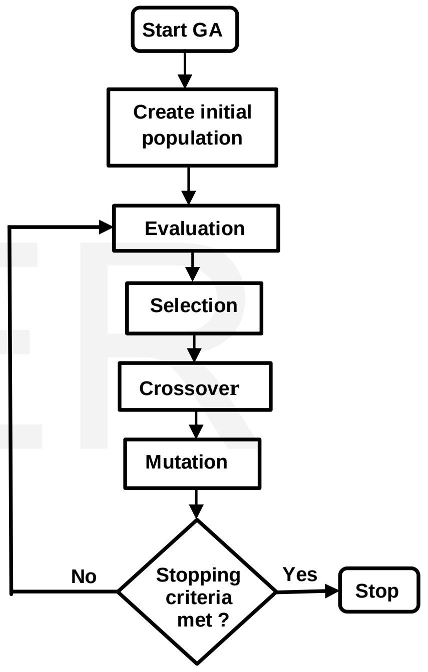

Fig. 2. Steps of a Simple Genetic Algorithm.

3 NEURAL NETWORK

A Biological Neural Network refers to the information processing elements of the nervous system, organized as a collection of neural cells, called neurons that are interconnected in networks and interact with each other using electrochemical signals.

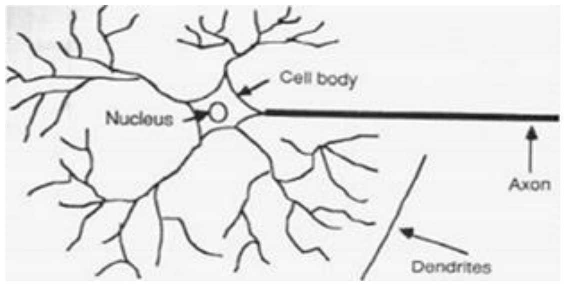

Fig. 3. Human Neuron.

As shown in **Fig. 3**, a biological neuron is generally comprised of an axon which provides the input signals and it is connected to other neurons via synapses. The neuron reacts to input signals and may produce an output signal on its output connection called the dendrites.

Artificial Neural Network (ANN) is modeled on the mechanism of the human brain. This network has the capacity to learn from examples, memorize, and create relationships amongst dada [8, 14].

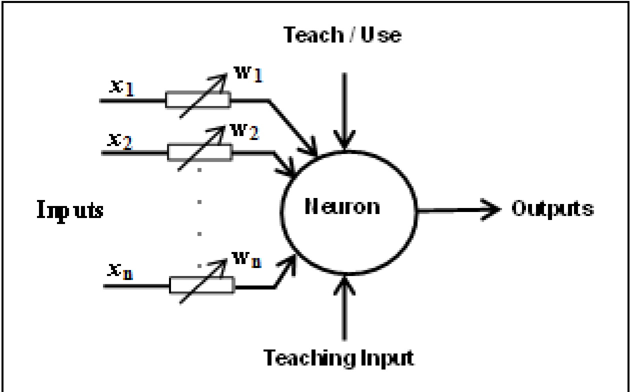

Fig. 4. Artificial Neuron

Artificial neural network life cycle:

Fig. 5 shows a typical neural network life cycle. Basically, this cycle encompasses the three main stages: network design, training, and testing.

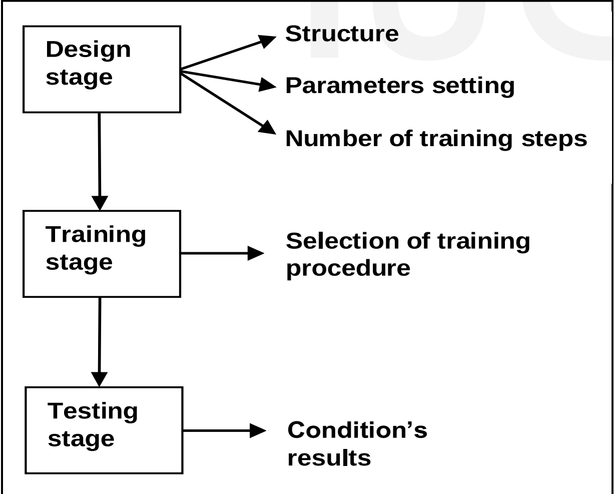

Fig. 5. Artificial Neural Network Life Cycle.

1- Artificial neural network design:

This stage involves a number of aspects: designing the network structre, selection of propagation function, selecting neuron transfer functions, selecting a method for updating the weights, designing the phyical connectivity, and selecting a training cessation scheme.

1-1 Designing the network structure:

The procedure for designing the ANN can be subdivided into three distinct categories as follows:

Input layer: The activity of the input units represents the raw information that is fed into the network.

Hidden layer: The activity of each hidden unit is determined by the activities of the input units and the weights on the connections between the input and the hidden units.

Output layer: The behaviour of the output units depends on the activity of the hidden units and the weights between the hidden and output units [3, 14].

1-2 Selection of propagation functions:

Several options of propagation function are available, such as weighted sum, cumulative sum, maximum, minimum, majority and product.

1-3 Selecting neuron transfer functions:

The behavior of an Artificial Neural Network (ANN) depends on both the weights and the input-output function (transfer function).

The most popular transfer functions as shown in **Fig. 6** are Linear, Sigmoid, Threshold, and Gaussian [5, 14].

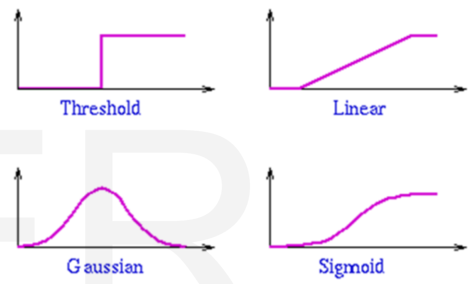

Fig. 6. Some Different Transfer Functions.

1-4 Selecting a method for update the weights:

The output of each node may be at variance with expected or required outputs. This produces an error signal that is fed back to the weighting functions and adjustments are made using an equation that reweights the original weighting functions. Back- propagation network can use either delta rule or generalized delta rule for learning purpose [14].

There are two major categories of neural networks based weights:

1. **Fixed networks** in which the weights cannot be changed according to the problem to solve.
2. **Adaptive networks** which are able to change their weights, i.e., d(weight)/d(time) ≠ 0.

1-5 Designing the physical connectivity:

The size of the network in terms of the number of units and connections should be a polynomial function of the dimension of inputs and outputs.

1-6 Selecting a training cessation scheme:

A stopping criterion needs to be chosen for determining the point at which the ANN provides a satisfactory performance and training should be ceased.

2- Training stage:

The purpose of training is to find optimal values of the weights so as to realize the desired mapping from inputs to outputs according to a criterion.

All learning methods are used for adaptive neural networks can be classified into three major categories:

1. **Supervised learning** which incorporates an external teacher, so that each output unit is told what its desired response to input signals ought to be.
2. **Unsupervised learning** uses no external teacher and it is based upon only local information. It is also referred to as self- organization.
3. **Reinforcement:** the network is provided with only a critique on the correctness of network outputs, not the correct answers [7 , 14].

Architecture of artificial neural networks:

Many identical neurons gathered in a neural network by interconnecting them according to a give graph (network topology) such as:

1) Feed-forward networks
2) Feedback networks
3) Network layers

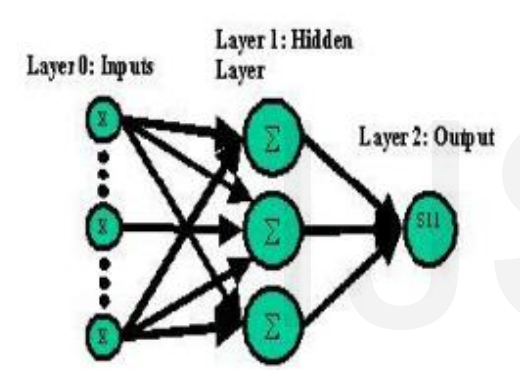

Fig. 7. Feed-forward Networks.

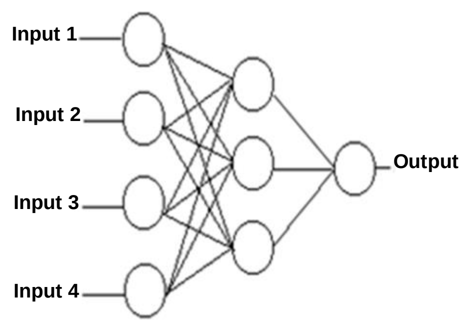

Fig. 8. Feedback Networks.

3- Testing stage:

The performance of the network after training is very important test of the network design.

4. THE PROPOSED ALGORITHM

In this section, an integration model based on decision neural network and genetic algorithm is developed to solve the interactive multi- objective optimization problems. The decision maker must specify her or his preference, hopes, and opinions within solving the multi- objective problem. Here, these preferences are the selection of initial values and stopping rule to guarantee getting the required solution. The neural - genetic model can be illustrated by **Fig. 9**.

The principle steps of this algorithm consist of:

a) The constrained problem is converted to unconstrained optimization problem.
b) Transforming the multi- objective problem into single objective problem by weighting the selected initial point based on the ANN. There are three ways to do this step on MATLAB software; the first way is to use ANN to set weights for the selected start point and this weighted point is given to the genetic algorithm as initial solution. But, the second way multiplies the ANN's weights and biases by the selected initial point. This weighted point is used as input to the GA. The third way is the integration between both the first way and the second way.
c) The initial population of the suitable solution is created for the problem; i.e., the first generation.
d) During evaluating step, the fitness of population f(x) is calculated.
e) Selecting two parent solutions from a population to reproduce itself; the solution takes place in a random way but with a probability proportional to fitness.
f) Applying different genetic operations such as crossover and mutation to generate future population from the selected set.

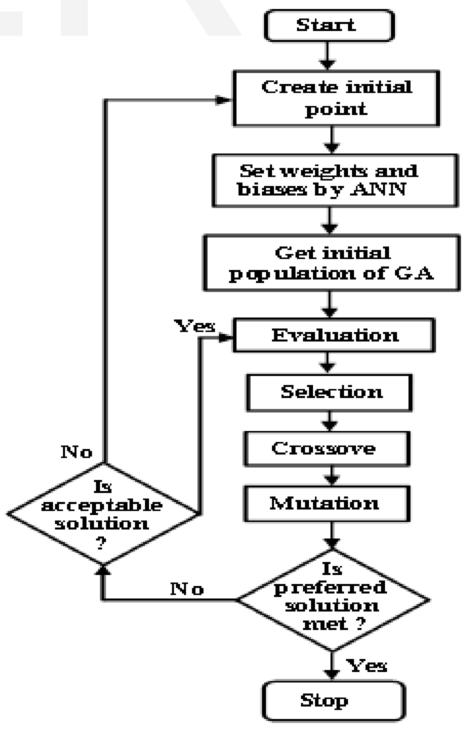

Fig. 9. The Flowchart of the Proposed Algorithm.

g) Placing new solution in a new population for a further generation.
h) If the end condition is satisfied, stop. Else, the output solution of GA is restarted to ANN (step "b") or the evaluation step (step "d").

Note that:

- This process is repeated until the required solution or near solution to it is met.
- The convergence of the genetic algorithm is determined by two ways: (i) When the GA reaches a maximum number of generations, and /or (ii) preferred value of objectives met.
- A value function such as Alia point [1] can be used to simulate the decision maker's preferences in the solution evaluation process.
- The generated solution can be created for all or some decision variables to save time.

i) This cycle continues until the preferred solution is reached by GA, and the algorithm ends based on stopping criteria (the value of objectives).

Remark:
The required solution is achieved easily based on the algorithm programming.

5. SOME ILLUSTRATED EXAMPLES

The proposed algorithm is implemented on the two different nonlinear examples using MATLAB code 7.0.

Example (1):

Consider the following multiobjectives problem:

Min $f_1 = 4x_1^2 + 4x_2^2$,
$f_2 = (x_1 - 5)^2 + (x_2 - 5)^2$
Subject to
$(x_1 - 5)^2 + (x_2 - 5)^2 \leq 25$,
$-(x_1 - 8)^2 - (x_2 + 3)^2 \leq -7.7$,
$0 \leq x_1 \leq 5, \quad 0 \leq x_2 \leq 3$.

This example has two variables constrained multi-objective optimization. Its solution has two connected regions in Pareto optimal front. First region constitutes $x_1^* = x_2^* \in [0, 3]$ and second region lies on $x_1^* \in [3, 5]$, $x_2^* = 3$ [17].

The selected initial point is the optimal solution of the second objective (5, 3). There are four different ANN's functions (newff, tansig, purelin, and trainlm) for weighting objectives. In the first way to transform the multi- objective into single objective, the weighted starting point of genetic algorithm is (x_1=1.7871, x_2= 2.4252). Here, the network's input ranges from [0 to 10] neurons. Three layers for nonlinear functions, and one layer has linear function.

By direct search toolbox, you can use GAOPTIMSET for default GA options structure. The penalty function is used to transform this constrained problem to unconstrained problem. The stopping criterion of GA is the maximum number of generations that equals to infinity. The required level of objectives or the fitness limit is 42.16.

The solution of this example is achieved by two ways of initial point calculating. The two pairs of variables x_1 and x_2 are selected by the first way to reproduce a new pairs of variables x_1 and x_2. The first way needs more time than other way to get the decision maker's solution. But in the second way, the change is not happening in all the variables at every time. As shown in the following two tables and figures, the decision maker's solution is ($x_1 = x_2$ =1.4645) with the total value of objectives is 42.1573. The value of mutation's parameter is one; the mutation function is Gaussian function. The used crossover operator is single point for all steps of the first way to create the initial point for GA with cross fraction 0.5, and population size is 10.

Table (1) The first way for initial point of example (1)

No.	x_1^*	x_2^*	No.	x_1^*	x_2^*
1	5.0	3.0	18	1.4581	1.4699
2	1.7871	2.4252	19	1.4577	1.4689
3	1.4897	2.3521	20	1.4588	1.4641
4	1.3144	1.7877	21	1.4605	1.4660
5	1.3568	1.5022	22	1.4624	1.4658
6	1.4315	1.5559	23	1.4626	1.4649
7	1.4414	1.5443	24	1.4627	1.4645
8	1.4414	1.5255	25	1.4644	1.4664
9	1.4008	1.5505	26	1.4620	1.4669
10	1.4242	1.5163	27	1.4623	1.4665
11	1.4319	1.4999	28	1.4628	1.4651
12	1.4364	1.4939	29	1.4605	1.4646
13	1.4450	1.4835	30	1.4640	1.4647
14	1.4533	1.4705	31	1.4642	1.4646
15	1.4539	1.4691	32	1.4644	1.4646
16	1.4549	1.4760	33	1.4645	1.4645
17	1.4552	1.4740			

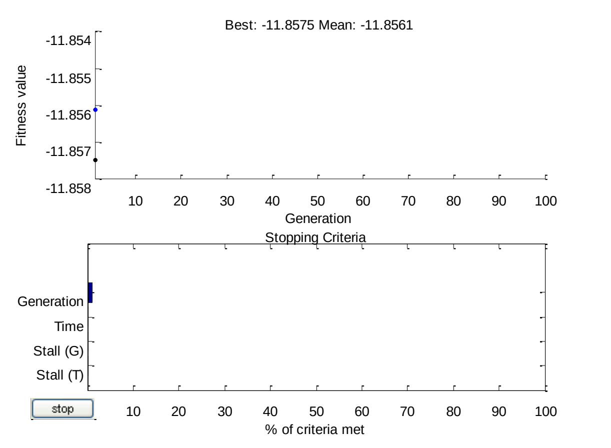

Fig. 10. The Final Generation of GA by the First Way of Example (1).

But in the second way, population sizes are 10 or/and 20 with different cross fractions (0.5, 0.7, 0.9).

Table (2) The second way for initial point of example (1)

No.	x_1^*	x_2^*	No.	x_1^*	x_2^*
1	5.0	3.0	25	1.6452	1.6928
2	0.4427	0.7553	26	1.6452	1.6853
3	2.2135	2.2659	27	1.6123	1.6329
4	2.1009	2.1744	28	1.5891	1.6285
5	1.9966	2.1744	29	1.5608	1.6219
6	1.9939	1.9986	30	1.5052	1.6219
7	1.9896	1.9979	31	1.5052	1.5615
8	1.9896	1.9948	32	1.5052	1.5270
9	1.9868	1.9948	33	1.5052	1.5206
10	1.9868	1.9928	34	1.4849	1.5028
11	1.9813	1.9916	35	1.4729	1.4979
12	1.9813	1.9912	36	1.4729	1.4797
13	1.9813	1.9897	37	1.4698	1.4786
14	1.9706	1.9897	38	1.4667	1.4786
15	1.9706	1.9825	39	1.4667	1.4770
16	1.9332	1.9733	40	1.4667	1.4692
17	1.9332	1.9716	41	1.4667	1.4683
18	1.9276	1.9598	42	1.4662	1.4680
19	1.9106	1.9477	43	1.4646	1.4677
20	1.8910	1.9386	44	1.4646	1.4665
21	1.8840	1.9239	45	1.4646	1.4648
22	1.8098	1.8835	46	1.4644	1.4647
23	1.8098	1.8677	47	1.4644	1.4646
24	1.7253	1.8196	48	1.4645	1.4645

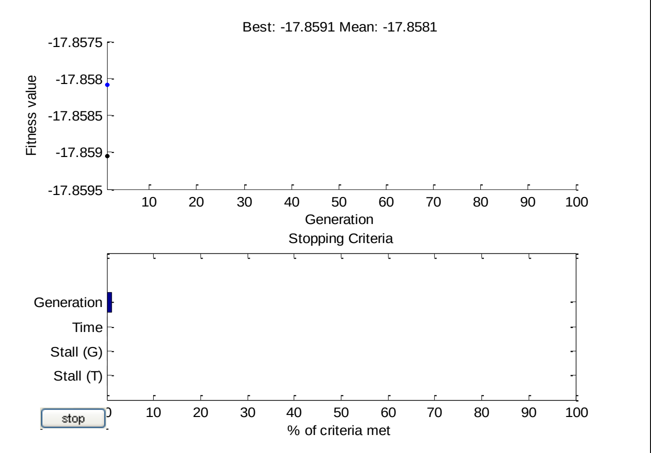

Fig. 11. The Final Generation of GA by the Second Way of Example (1).

Example (2):

Min= $((x_1-2)^2 + (x_2-1)^2 + 2)$, $(9x_1 - (x_2-1)^2)$,
S.to:

$x_1^2 + x_2^2 \leq 225$,
$x_1 - 3x_2 \leq -10$,
$x_1 \geq 0$, $x_2 \geq 0$ [16].

The individual optimal solutions of this problem are (1.1, 3.7), and (0, 15). Therefore, the ideal points of this example are f_1^*= (10.1, 2.61) and f_2^*= (202, -196). For the first way to select the initial point of the proposed algorithm, the started point to ANN is (0, 0). The output of ANN by using “newff”, “tansig”, and “purelin” functions is (0.166, 2.5977). The range of network's layers is [0 5] layers; three for nonlinear functions and one for linear function.

The first generation of GA by the first way of example two presents the point (-0.3883, 4.7639) as depicted in Fig. 12. So, the second initial point to genetic model is (0.166, 4.7639). **Table (3)** presents these results. The mutation functions are uniform and Gaussian with values 0.00001, 0.0001, 0.3, 0.5, 0.99, and 1. The range of population size is [10- 40]. The used cross fraction values are set to [0.3: 1.0].

Table (3) The integration way for initial point of example (2)

No.	x_1^*	x_2^*	No	x_1^*	x_2^*
1	**0**	**0**	20	0.0013	3.3349
INN 1	**0.1660**	**2.5977**	21	0.0003	3.3349
3	0.1660	4.7639	**22**	**0.0001**	**3.3349**
4	0.1660	3.8448	**INN 2**	**0.2540**	**0.9185**
5	0.1498	3.9099	**24**	**0.0000254**	**3.0631057**
6	0.1498	3.4110	25	0.0000254	3.3129
7	0.1498	3.3041	26	0.0000254	3.3329
8	0.0852	3.3041	**27**	**0.0000254**	**3.3343**
9	0.0852	4.3333	**INN 3**	**0.0103**	**0.7861**
10	0.0852	3.3310	**29**	**2.6162E-07**	**2.621093**
11	0.0791	3.3310	30	2.6162E-07	3.0339
12	0.0467	3.3310	31	2.6162E-07	3.3350
13	0.0248	3.3310	**32**	**2.6162E-07**	**3.3338**
14	0.0073	3.3310	**INN 4**	**0.0075**	**1.1213**
15	0.0031	3.3310	**34**	**1.96215E-09**	**3.73818994**
16	0.0027	3.3310	35	1.96215E-09	3.2334
17	0.0027	3.3489	36	1.96215E-09	3.3305
18	0.0026	3.3489	**37**	**1.96215E-09**	**3.3334**
19	0.0016	3.3349			

The fitness limit is equal to 6.0. The starting process for GA is schematically shown in **Fig. 12**. The optimization process is continued for up to 22 trials by restarting a new initial solu-

tion using ANN. But due to save time, the reminder steps apply the second way for extracting the required solution.
In another solution of this example by the second way, the weights by ANN are (0.7209, 0.6270) for the point (0, 15). Then the initial solution to GA is (0, 3.1350). The variable x_2 is only reproduced, but the value of x_1 is zero. **Table (4)** shows the list of selected initial points for this problem with different degrees of the mutation Gaussian.

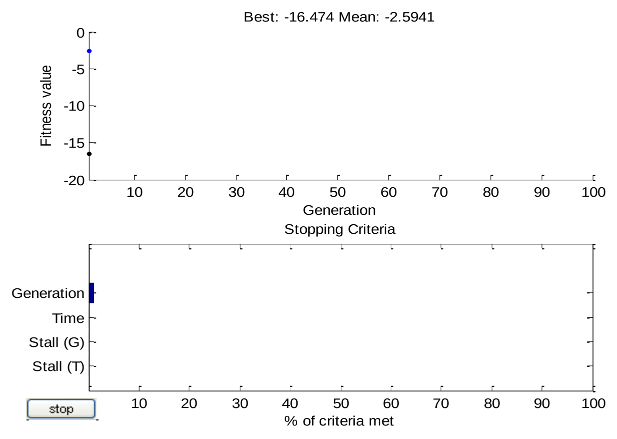

Fig. 12. The First Generation of GA by the First Way of Example (2).

Table (4) The second way for the starting points of example (2)

No.	Initial points		Population Size	Cross Fraction	Mutation
	x_1*	x_2*			
1	0	15			
INN 1	0.7209	0.6270			
2	0	3.1350	30	0.9	Gaussian 1 1
3	0	3.4588	30	0.9	Gaussian 1 1
4	0	3.2801	30	0.9	Gaussian 1 1
5	0	3.4192	30	0.9	Gaussian 1 1
6	0	3.3709	50	0.8	Gaussian 1 1
7	0	3.3411	30	0.8	Gaussian 0.99 1
8	0	3.3373	30	0.3	Gaussian 0.5 1
9	0	3.3367	20	0.3	Gaussian 0.3 1
10	0	3.3359			
INN 2	0.3648	0.9973			
12	0	3.326893	40	0.7	Gaussian 0.5 0.0001
13	0	3.3349	30	0.8	Gaussian 0.3 0.001
14	0	3.3305	40	0.7	Gaussian 0.5 0.0001
15	0	3.3334	→ The required solution		

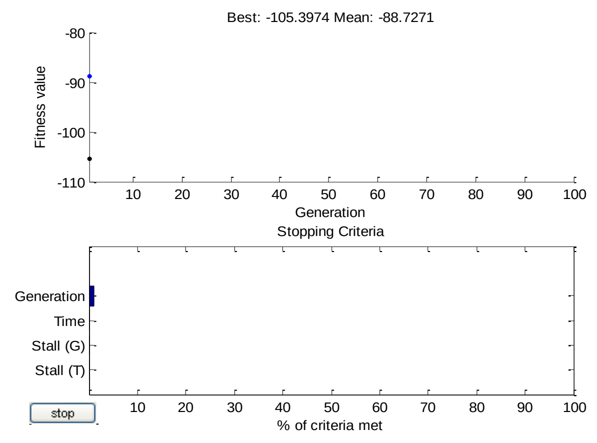

Fig. 13. The Final Generation of the First GA's Cycle by the Second Way of Example (2).

Fig. 13 shows the final generation of the first GA's cycle by the second way of example two that produces the point (-1.2094, 3.3359). Then the initial point for the second GA's cycle is (0, 3.3359) that will be weighted by ANN. This process is continued until the optimization algorithm is terminated when a pre-defined value is reached as seen in **Table (4)** and the following figure.

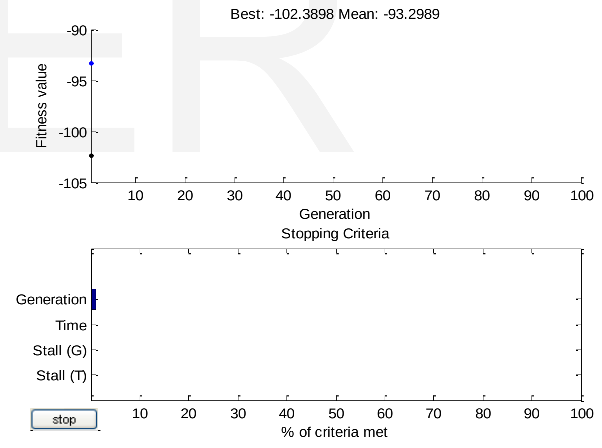

Fig. 14. The Final Generation of GA by the Second Way of Example Two.

6 CONCLUSION

This paper introduces an adaptive interactive approach based on decision neural network. The neural network is used to capture and represent the decision maker's preference as the input to search for the desirable solution. Then by applying genetic algorithm, the most desirable solution is provided. Also, a value function is used to simulate the decision maker's preferences in the solution evaluation process.
The results show that the combination of neural network model with genetic algorithm gives a clear improvement in solving the interactive multi- objective optimization problems.

For future research directions, it is valuable to incorporate the other algorithms into NN or / and GA that have a number of practical implications.

Acknowledgment

- **At first thanks be to ALLAH** for his guidance and supports in showing us the path.
- **Secondly**, I would like to thank my supervisor **Prof. Dr. Mohamed Sayed Ali Osman** (Vice President of the Tenth of Ramadan University, Egypt) for his supporting.
- **Thirdly**, I want to thank my second supervisor **Prof. Dr. Waiel Fathi Abed El Wahed** (Former Dean of the Faculty of Computers and Information, Menofia University, Egypt); who died in September 2011; for his supporting and suggesting the idea of this research.

REFERENCES

[1] Alia Y. Gebreel, "On a compromise solution for solving multi- objective convex programming problems", International Journal Scientific & Engineering Research (ISSN 2229- 5518), Vol. 7, No. 6, PP. 403-409, 2016.

[2] Hong Z. Huang, Zhigang Tian, and Ming J. Zuo, "Intelligent interactive multiobjective optimization method and its application to reliability optimization", IIE Transactions, ISSN, 0740- 817 X, 37, PP.983-993, 2005.

[3] Internet: "A seminar on neural network", www. powerpointpresentationon.blogspot.com.

[4] Internet: IV Genetic algorithm, http.

[5] Internet: Mg. Samuel Oporto Díaz, "Desarrollo de soluciones inteligentes - Introducción a las redes neuronales", soporto@wiphala.net http://www. wiphala. net/oporto.

[6] Internet: "Multi-objective optimization", Wikipedia, the free encyclopedia, 2015.

[7] Internet:Theodore B. Trafalis, " Optimization-neural networks Learning from data", School of Industrial Engineering University of Oklahoma Norman.

[8] Jason Brownlee, "Clever algorithms: Nature-inspired programming recipes", PhD at Swinburne University, 2011.

[9] Javier Sanchis, Miguel A. Martinez, and Xavier Blasco, "integrated multiobjective optimization and a priori preferences using genetic algorithms", Information Sciences, An International Journal, www. Elsevier.com/locate/ins, 2008.

[10] Jian- Bo Yang, and Pratyush Sen, "Preference modelling by estimating local utility functions for multiobjective optimization", European Journal of Operational Research, 95, PP. 115- 138, 1996.

[11] Kaisa M. Miettinen, "Nonlinear multiobjective optimization", Kluwer Academic Publishers, 1998, and Fourth printing 2004.

[12] Kalyanmoy Deb, "Multi- objective optimization using evolutionary algorithms", John Wiley & Sons, Ltd, 2001.

[13] Maridass Balasubramanian, Marissa A. Paglicawan, Zhen-Xiu Zhang, Sung Hyolee, Zhen-Xiang Xin, and Jin Kuk Kim," Prediction and Optimization of Mechanical Properties of Polypropylene/Waste Tire Powder Blends using a Hybrid Artificial Neural Network-Genetic Algorithm (GA-ANN)", Journal of Thermoplastic Composite Materials, Vol.21, PP.51-69, 2008, http://www.sagepublications.com.

[14] Mohamed H. Rasmy, Amr M. S. Goneid, and Alia Y. Gebreel, "A Tool for Local Cash Flow Forecasting Based on Fuzzy Neural Networks ", Master Degree in Operations Research, Institute of Statistical Studies and Research, Cairo University, 2001.

[15] Petr Fiala, "Multiobjective De Novo Linear Programming", Acta Univ. Palacki. Olomuc., Fac. rer. nat., Mathematica, Vol. 50, No.2, PP. 29–36, 2011.

[16] Saad M. A. Eid, Mohamed H. Rasmy, Elhamy Hasanain Elsherif, "Genetic algorithms approach to solve multi-objective decision problems", Degree of PhD. of Operations Research, Department of Computer & Information Sciences and Operations Research, Institute of Statistical Studies and Research , Cairo University,1998.

[17] Tushar Goel, and Nielen Stander, "Multi- objective optimization using LS-OPT", LS-DYNA Anwenderforum, Frankenthal, DYNA *more* GmbH, 6, 2007.