Submitted PAPER (ISEE05), HCMCity University of Technology, VietNam, October,2005.

A NOVEL APPROACH FOR EFFECTIVELY CONVERGENT OPTIMISATION OF GENETIC ALGORITHMS BY APPLYING A NEW ACCELERATION TECHNIQUE USING IN SOME ENGINEERING OPTIMAL PROBLEMS .

Ahn Kyoung Kwan (Ph.D. , Professor) **Ho Pham Huy Anh (M.Eng)**
FPCL, Mechanical and Automotive Engineering Department
University of ULSAN, Korea

ABSTRACT .

In recent years, genetic algorithm is becoming a powerful tool for researchers and engineers to find good optimal solutions in diverse hard computational mechanical and electrical engineering problems. However, classic genetic algorithms (GA) encounter several disadvantages in which this is the difficulty of choosing optimal parameter settings. Genetic algorithm literature has presented lots of empirical tricks that enable GAs to be convergent better in some way. This paper presents a novel approach for applying a new acceleration technique as well as applying the DeJong functions in testing which has enabled engineers to resolve optimal mechanical and electrical engineering problems that outperform conventional GAs. Thus, the GA with new acceleration technique is experimented in simulation for gathering the simulated data then comparing them with the results from conventional GA. Consequently, a conclusion can be settled on how efficient the new GA proves.

Keywords : genetic algorithm, acceleration technique, optimal control, accelerated genetic algorithm, DeJong function .

I.INTRODUCTION

Our real world is filled with discontinuities and noisy process. Conventional optimal control methods are too slow for finding the solution and often aren't robust enough because they are local optimal in scope. On the contrary, genetic algorithm (GA) is a efficient and robust search method demanding little data to search effectively optimal solution. With the nature that GA goes through reproduction and mutation, it is able to create offsprings that have a best chance of inheriting excellent characteristics of both parents with higher fitness value. Furthermore, GAs are computationally simple and easy to implement.

We summarize the basic advantages of GA as following [3] :

- GA requires not much mathematical basis of the controlled process. With its evolutionary nature, GA will search for optimal solution for control problem without attention to the specific inner structure of the process. GAs can revolve successfully any kind of objective functions as well as any kind of linear or non-linear constraint defined on discrete, continuous or hybrid controlled process.
- GA proves very effective in performing globally optimal search. Conventional approaches perform local search with a stepwise convergent procedure. Furthermore, global optimisation can be found successfully only if the solution possesses some convexity properties that determine it is a global optima.
- GA permits us a high flexibility to hybridize itself with other domain-dependent intelligent control such as fuzzy or neural networks to make an effective implementation in a specific controlled process.

II.ACCELERATED GA (AGA) STRUCTURE

The basic factors that affect the speed and the robustness of a genetic algorithm are dependent on the population size of a GA, the fitness and the number of generations. A new acceleration technique is applied in order to control effectively the size of the population of

chromosomes. The more increasing the population is, the more the computation time requires. But with a population of quality (elitism), the size of the population can be reduced without reducing the GA performance. Thus, new acceleration technique leads us to the virtual population for formatting the high-quality chromosomes in GA population.

The skeletal outline of the optimal searching in a process control for a AGA program is as described below [1][2] :

a. **{Begin}** : Create randomly population of n chromosomes
b. **{Fitness}** : Estimate the fitness *f(x)* for each chromosome *x* .
c. **{New Population}** : Repeating following steps until a new population is completed.
- ***[Selection]*** : Choose two parent chromosomes from the population with their high fitness level.
- ***[Crossover]*** : Crossover the parents to form a new offspring. With elitism critera, in case no crossover was realised, offspring would be an exact copy of parents.
- ***[Mutation]*** : Mutate new offsprings at well-defined bits with a mutation probability.
- ***[Acceptance]*** : Form a new population from new offspring depending on their fitness level.

d. **{Replace}** : Generate a new population (combining parents and offsprings) for a further run of algorithm.
e. **{Test}** : In case optimal criteria has been achieved, stop GA program and display results.
f. **{Loop}**: On the contrary, go to **Step b**.

III.IMPLEMENTATION :

Such AGA proves no doubt successful in achieving optimal solution. In modern engineering problems, AGA would be able to satisfy a constant demand for even faster and better optimal-searching techniques. It is also the main purpose of this paper.

Firstly, we apply a new acceleration technique which helps GA program running faster and much more robust. Secondly, we apply DeJong Functions in GA program for proving and testing GA features. Finally, we present some simulation results for optimal searching problem using conventional GA as well as new approach GA combining new acceleration technique . Both of them were written in C language.

There is a long list of DeJong Functions, and for the aim of paper only to find an effective tool for testing and proving optimal solution of GA, we choose only the first three and implement them in the GA simulation program. These first three DeJong functions were defined as following : [5][6]

Function DeJong1 : **$f(x_i)$ = sum of (x_i^2) while i ranges from 1 to 3 and $-5.12 \le x_i \le 5.12$** .

Function DeJong2 : **$f(x_i) = 100(x_1^2 - x_1^2) + (1+x_1)^2$ with $-2.048 \le x_i \le 2.048$.**

Function DeJong3 : **$f(x_i)$ = sum of integer of (x_i) while i ranges from 1 to 5 and x_i within $-5.12 \le x_i \le 5.12$**

It is also presented a new acceleration technique which was implemented and tested in the accelerated GA program. The C code for the accelerator is shown in appendix .

IV.SIMULATION RESULTS :

For each of AGA simulation program implemented with one type of Testing DeJong function, results achieved with and without the use of new acceleration technique will be presented together in order to compare easily the two's result. Furthermore, such results will be presented under graphical format for analysing them easily.

In these following figures, the vertical represents the fitness level as well as the horizontal represents the number of iterations.

The results of GA simulation program of *Function Testing Dejong1* without and with the use of accelerator will be showed in *Figure 1* and *Figure 2* :

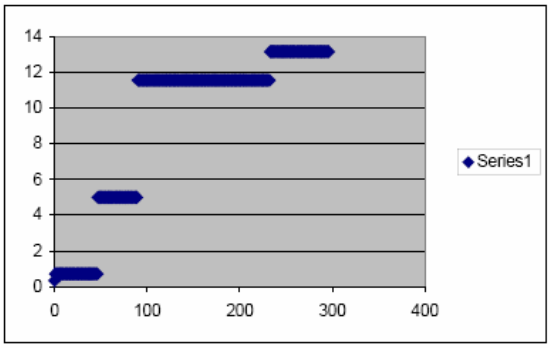

Figure1: GA fitness of DeJong1 Function without accelerator

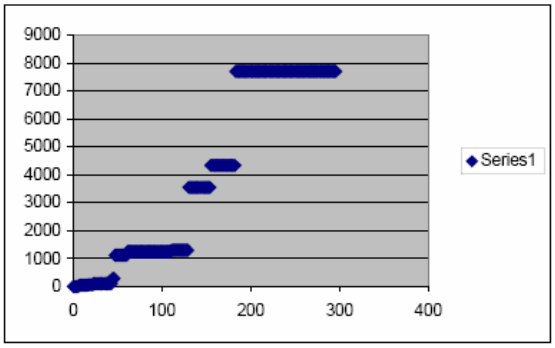

Figure 2: GA fitness of DeJong1 Function with accelerator

For this case of DeJong1 function, the higher the fitness function is, the better GA algorithm proves. In comparison the results from *Figure 2* and *Figure 1* , it is easily shown that with the effect of the acceleration technique, we obtain a much higher fitness level in a shorter time.

The results of GA simulation program of *Function Testing Dejong2* without and with the use of accelerator will be showed in *Figure 3* and *Figure 4* :

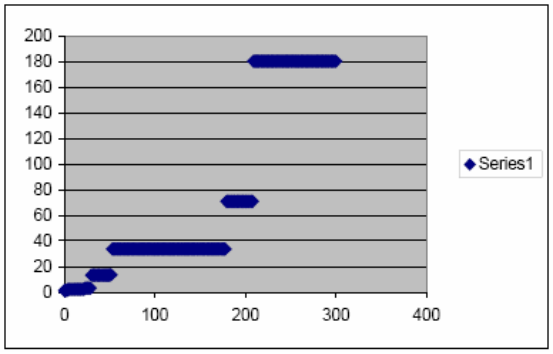

Figure 3: GA fitness of DeJong2 Function without accelerator

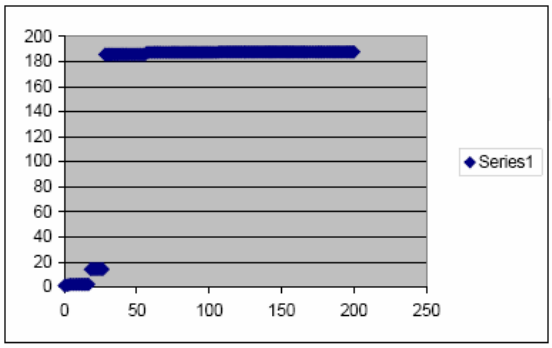

Figure 4: GA fitness of DeJong2 Function with accelerator

In comparison the results from *Figure 3* and *Figure 4* , it is easily shown that with the effect of the acceleration technique, we obtain a much higher fitness level in a shorter time. Particularly, in *Figure 4*, with the aid of the acceleration technique, optimal result was achieved in no more 50 iterations . On the contrary, in *Figure 3*, it was only obtained after 200 iterations. Consequently, it can be deduced that the new acceleration technique speeds up effectively the optimal searching process of GA program.

The results of GA simulation program of *Function Testing Dejong3* without and with the use of accelerator will be showed in *Figure 5* and *Figure 6* :

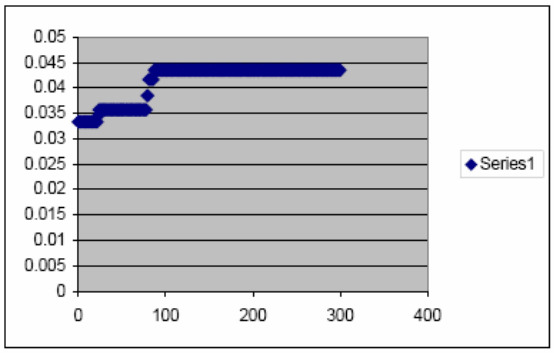

Figure 5: GA fitness of DeJong3 Function without accelerator

In comparison the results from *Figure 5* and *Figure 6*, it is easily shown that with the effect of acceleration technique, due to different coding, the result is particularly different from the rest. For this case of DeJong3 function, the lower the fitness function is, the better GA algorithm proves. The result showed that the fitness level obtained from GA without

accelerator at the end of simulation process is about 0.045 as shown in Figure 5. Whereas, the fitness level obtained from GA with accelerator at the end of simulation process is more precise about 0.0385 as shown in Figure 6. Clearly, GA with accelerator proves superior.

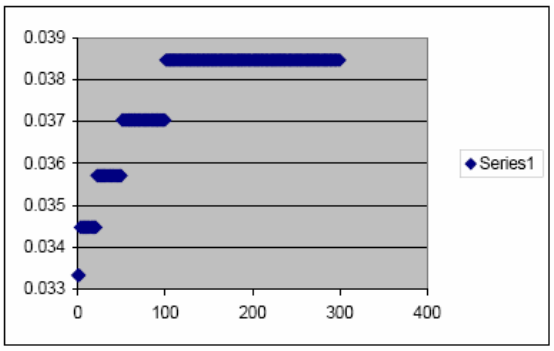

Figure 6: GA fitness of DeJong3 Function with accelerator

V.CONCLUSION :

Up to now, GA has been very popular as an efficient and robust form of optimal searching in control technique. Nevertheless, it's still limited with the drawback that GA doesn't know exactly optimisation process has been completed and thus, we need additional test functions in order to fix when optimal solution has been achieved.

The paper introduced the DeJong Functions which prove useful to test for optimal result. We also insert a new acceleration technique in GA program to speed up the optimal searching process. Simulation results were carried out with and without this new acceleration technique and were presented in graphs and in tables. We easily saw that with acceleration technique , GA program needs less iterations for the search of optimal solution.

This new acceleration technique has proved highly successful in speeding up the optimal searching process of GA. Of course , we continue to find out other available techniques. The future research in this domain would be to explore other acceleration techniques and compare all these results to conclude which will be the most efficient acceleration technique. With the most common use of GA is for machine learning as well as for search optimisation. GA can be used efficiently for the tuning of PID controllers in mechanical and electrical engineering process as well as optimisation of power control system. Results acquired from GA simulation program determine the success of applying various acceleration techniques in combination with GA as well as make GA more efficient in its prosperous applications.

REFERENCES :

[1] GEN, M. & CHENG, R., *Genetic Algorithms and Engineering Optimisation*, John Wiley & Sons, Inc. 2000 .

[2] GEN, M. & CHENG, R., *Genetic Algorithms and Engineering Design*, John Wiley & Sons, Inc. 1997 .

[3] GOLDBERG, D., *Genetic Algorithms in Search, Optimisation and Machine Learning*, Addison-Wesley, 1987 .

[4] MICHALEWICS, Z., *Genetic Algorithms + Data Structure = Evolution Programs*, Springer – Verlag, 1996 .

[5] University of Nevada, Reno, "*DeJong Function Minimisation*", website visited 2nd, September2005, http://www.cs.unr.edu/~gruber/ga/dejong.html

[6] University of Nevada, Reno, "*DeJong Function Minimisation*", website visited 2nd , September2005, http://www.cs.unr.edu/~ekasit/ga/dejong.html.

[7] MAREK OBITKO, "*Genetic Algorithms and Applications* ",website visited 2nd, September2005, http://www.cs.unr.edu/~gruber/ga/dejong.html

Submitted PAPER (ISEE05), HCMCity University of Technology, VietNam, October,2005

USING A MODIFIED GENETIC ALGORITHM FOR MULTI-OBJECTIVE MODELLING AND OPTIMISATION OF TAGAKI-SUGENO FUZZY MODEL .

Ahn Kyoung Kwan (Ph.D. , Professor) **Ho Pham Huy Anh (M.Eng)**
FPCL, Mechanical and Automotive Engineering Department
University of ULSAN, Korea

ABSTRACT .

This paper is concerned with using a modified genetic algorithm (MGA) for generating the Tagaki-Sugeno type fuzzy model. By using such a MGA, we process the input-output data and optimize the fuzzy model. This is referred to perform fuzzy identification by which we generate automatically the appropriate fuzzy if-then rules to characterize the plant .
We know that in conventional identification techniques, difficulties such as poor knowledge of the process, inaccurate process parameters determining and complexity of the resulting model, all which limit their usefulness during dealing with dynamic industrial processes. Fuzzy model identification is a rather new area in control system design as well as the field of MGA optimisation of such T-S type fuzzy model is still in its early stage.
Keywords : Tagaki-Sugeno fuzzy model, identification model, optimal control, modified genetic algorithm.

I.INTRODUCTION :

Conventionally, the fuzzy model was installed using expert human knowledge of the system and often carrying a heuristic trial and error approach. Thus, it is always desirable to develop an available good fuzzy model of the system but still restricting the complexity of this model. For the purposes of process control, a fuzzy model obtained from a training data set is available for prediction, simulation, optimization as well as for control of the unknown system plan.

This paper suggests an algorithm for the generation of the Tagaki-Sugeno type fuzzy model by using Genetic Algorithm to process the input-output data in order to optimize this fuzzy model. By this way, we generate the appropriate fuzzy if-then rules to characterize the system plant. Traditional identification techniques encounter lots of difficulties such as poor understanding of the plant, inaccurate parameters of the process as well as complex level of the resulting mathematical model. These difficulties limit their usefulness during dealing with nonlinear and dynamic processes which proved more and more common in various branches of modern engineering. This result is only in an initiative phase because fuzzy identification is still a new area in control systems design and the optimization of such fuzzy model using genetic algorithm (GA) proves a really promising field.

II. FUZZY MODEL :

Traditional mathematical models seem hard to achieve with nonlinear processes because the underlying dynamics of the system proves impossible to model [11]. The problem will be solved if the process is described as a series of if-then rules. These rules map input space to output space for modeling the process. Thus, fuzzy model possesses no strict mathematical equations and then the application range for the fuzzy model is very large.

Four main components of the fuzzy model are shown in *Figure 1* :

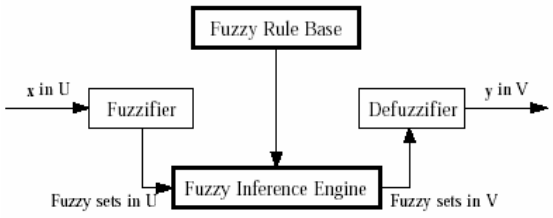

Figure 1

- ***The rule-base*** : holds the knowledge for the system control as a set of If-Then statements

- ***The inference engine*** : evaluates which rules are relevant at the current time and decides the input value to the process should be.
- ***The fuzzification interface*** : is the input interface with modifying function the inputs so that they may apply to the rules in the rule-base.
- ***The defuzzification interface*** : is the output interface with output converted from the inference mechanism may be fed into the process.

This fuzzy model provides a means for encompassing nonlinearities as well as uncertainties of the process without strict mathematical statements.

III.TAGAKI-SUGENO FUZZY MODEL :

The T-S fuzzy model implies the conclusion of a rule is a linear function of the inputs with its structure is shown in *Figure 2* :

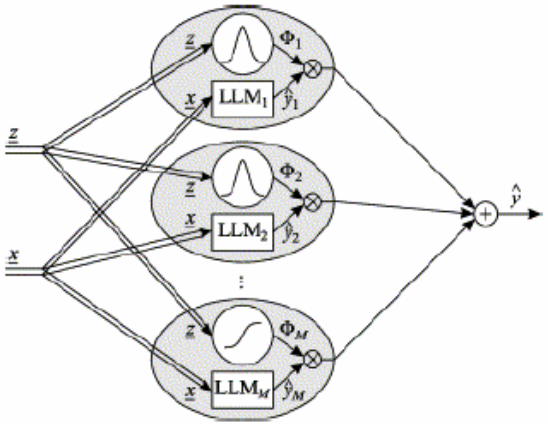

Figure 2: Tagaki-Sugeno Fuzzy Model

The rule premise inputs z and the rule consequent inputs x are subsets of the function inputs u with neuron of such network possess a fuzzy rule and activate as a Local Linear Model (LLM).[7]

The T-S fuzzy model output is calculated as a weighted sum from LLM outputs where the validity functions $\Phi_i()$ are interpolated as operating point dependant weighting factors :

IF z_1 is $A_{i,1}$ AND … AND z_{nz} is $A_{i,nz}$
THEN $y = w_{i,0} + w_{i,1} x_1 + \ldots + w_{i,nx} x_{nx}$

Validity functions are used to interpolate between different LLMs. The validity functions $\Phi_i()$ take a partition of unity in the form that :

$$\sum_{i=1}^{M} \Phi_i(\underline{z}) = 1$$

and thus the output of the T-S fuzzy model is calculated :

$$\bar{y} = \sum_{i=1}^{M} \Phi_i(\underline{z}).(w_{i,0} + w_{i,1} x_1 + \ldots + w_{i,nx} x_{nx})$$

The T-S fuzzy model, through research as well as through practical application, proves to be a suitable tool for adaptively extracting rules from the training data.

IV.GENETIC ALGOITHM OPERATOR :

For the purpose of demonstration, binary values will be considered here for Genetic Algorithm (GA) operator. The evolving GA will use three operators :

- Reproduction
- Crossover
- Mutation

GA operates after following steps :

a. Generate a population of chromosomes.
b. Each chromosome is evaluated in the population
c. Mate current chromosomes for creating new chromosomes; while mating the parent chromosomes, applying mutation and crossover.
d. Evaluate new chromosomes and insert them into the new population.
e. Create new population from the best parent and new chromosomes.
f. If time is up or the maximum number of generations is reached or the fitness level is obtained, the GA operation is stopped, then return the best chromosomes. If not, repeat the GA operation form step C .

There are an ever-increasing number of applications of GA. Genetic Algorithm has been applied to solve such following types of problems [3] :

- Multiple-objective optimization

- Constrained optimization
- Multiple-solution problems
- Problems of nonlinear / dynamic / unknowable process

V.GENETIC ALGORITHM FOR OPTIMISING T-S FUZZY MODEL PARAMETERS :

In recent research papers, GA may be used to generate a fuzzy model based in two ways . The first is off-line design of fuzzy system, including generation of fuzzy model as well as design fuzzy controller. The second is on-line tuning of fuzzy systems.

This paper focuses on the off-line generation of a fuzzy model. In paper [10], Babuska summarizes the critical areas that define the fuzzy model structure :

- The more complex system is, the more important for deciding is that which variables should be used like inputs to the model. It is also important for estimating the order for dynamic systems.
- The structure and rule choice involves the model type (Tagaki-Sugeno, Mandani, singleton,..) and the antecedent form.
- The accuracy of the fuzzy model is often traded-off against complexity. Thus, it is careful to choose the number and the type of membership functions for each variable.

From these main principes, there are various methods for modeling a fuzzy model :

- Start with one rule model, and then progressively add rules as well as refine until design accuracy is achieved .
- Description the input-output with a fuzzy relation .
- Division the input-output data into clusters. Each rule will represent one or several clusters which may be interpreted as local models.

M. Mannle [6] proposes an effective basis for creating fuzzy models. In this algorithm, fuzzy modelling process is performed iteratively with each iteration consists of two phases. Departure with a simple model, a more complex structure with more fuzzy rules is chosen and its parameters are optimized by a learning algorithm which is based on given training data. This routine is only stopped if the fuzzy model is good enough or the number of fuzzy rules is exceeded. The way for forming new fuzzy rules bases on a current rule, then sub-divide it into two more rules. Through this algorithm, it may start with a single rule and along with its progression, furthermore efficient rules are created.

Farag et al [7] apply GA for only fine tuning the membership functions. Hoffman et al [3] apply GA for only shaping the membership functions and proved that the process of modeling T-S fuzzy model using GA reveals explicitly the trade-off between model accuracy and the model number.

It exists often the common problem of GA-based fuzzy model generation is that the number of IF-THEN rules will increased exponentially with the pattern space dimensionality. Solving this problem, Ishibuchi [8] applied new approach by pre-screening the rules. In general, an optimal fuzzy model is defined based on a set of objectives :

- Minimize the total rule length.
- Minimize the number of chosen rules.
- Maximize the classification accuracy.

V. ALGORITHM DESIGN PROCEDURE :

All results have been obtained in MATLAB simulation with the help of fuzzy logic Toolbox as well as GAOT (Genetic Algorithm Optimization Toolbox) .

5.1.DESIGN ISSUES :

Principal design issues prove apparent for developing the algorithm :

- Fuzzy Rule base
- Adequate scaling
- Membership functions type
- Number of membership functions
- Implementation the GA program

GAOT in use in this system is based on three basic steps :

1.Set the bound to search for each parameter.
2.Process the search space and settle a value for each parameter.
3.Evaluate fitness for that set of parameters

5.2.MODELLING ALGORITHM :

The algorithm structure composed of five steps to follow :

1) Create a basic T-S fuzzy model with default input/output membership functions as well as default rule base.
2) Determine the bounds for all input/output membership functions as well as rule base in preparation for initializing a population of parameters.
3) Realize the GA for settling a new parameters of the fuzzy model .
4) Estimate the fuzzy model in each generation through evaluating absolute error .
5) Repeat the model with best fitness value.

5.3.MODEL INITIATION :

The user specify the number of required membership functions (MF). The value entered can be any number greater than one with no upper limit. A simple modeling problem can require only 5 membership functions whereas a dynamic complex process may need 50 to 100 membership functions. In all cases, basic T-S fuzzy model is always created in the same way with examples are illustrated in Figure 3- 4 :

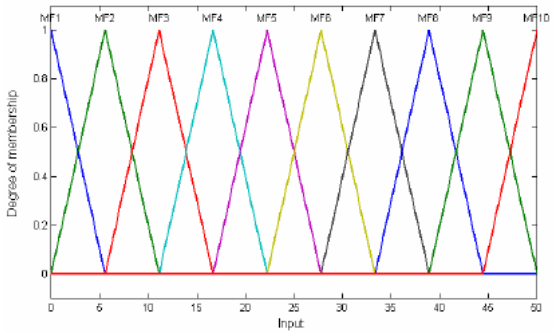

Figure 3: Ten Membership Function Input

The principe for shaping the input and output membership functions follows as :

- Settle the apex of the first MF at the lowest range value.
- Settle the apex of the last MF at the highest range value.
- Position the other MFs equally in order to fixing the crossover point between adjacent MFs at the default value of 0.5 . This satisfy that the apex of MF is always inline with the side of adjacent MFs.

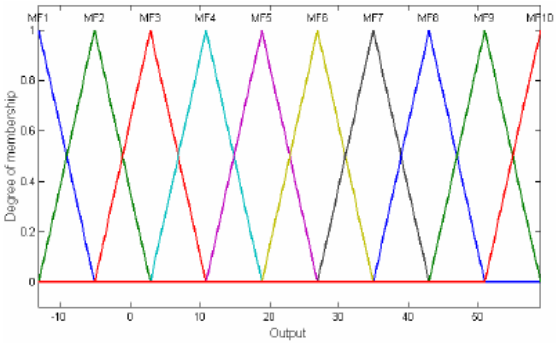

Figure 4: Ten Membership Function Output

5.4.RULE BASE INITIATION :

With a straitforward way, the rule base is settled with its number corresponding to the number of membership functions and each input membership functions will point to its corresponding output membership functions
The T-S fuzzy model initiation is settled in such a way membership functions will always spread out over the whole data set even though what range input or output data covers. Every input will be considered initially so that the most benefit could be obtained from searching the entire search space with all the data of the system is participated.

5.5.POPULATION INITIATION :

The Genetic Algorithm must be settled by setting the upper and lower bounds for each parameter, within which may search the algorithm. Each membership function possess three parameters to be optimized which are 3 vertex positions. The algorithm let each corner have the amount of freedom.

VI.TESTING RESULTS :

For testing the system, a nonlinear process is the most available for providing a sufficient challenge to the GA algorithm in optimizing

the model. For example, it is applied the nonlinear function for testing this algorithm :

$y = 10\sin(5u) + 7\cos(4u)$

For proving the performance of the algorithm, it is used two different data sets of the same function, each with a different range. Later test will be more challenging than the previous. The first test is the output of the function within the input range (0-3) with its graphic is shown in *Figure 5*:

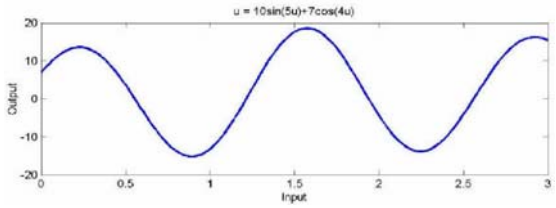

Figure 5: Testing fuction for Test1 (Input:0-3)

The second test got much more challenging , represents the output of the same function for the input range (0-25) with its graphic is shown in Figure 6 :

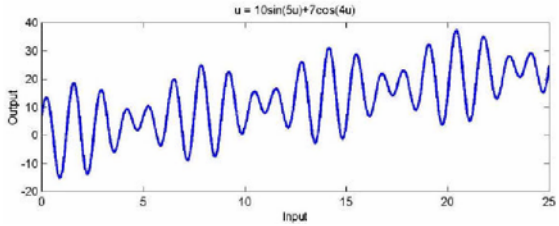

Figure 6:Testing fuction for Test2 (Input:0-25)

It is clearly that these two tests prove a conformable challenge for the optimal problem in modeling a T-S fuzzy model. Within each case, the fuzzy model is identified with different number of membership functions, with varied population sizes as well as with the different number of generations as to give a complete panorama about the capabilities of the algorithm. In each case, fitness value of the fuzzy model will be determined by the mean absolute error (MAE) of the model.

For the first case, a result summary is shown in Table I. *(see Appendix)*
It is clear that the mean absolute error reduces significantly as increasing the number of membership functions .

It also can see from TEST 1A that the fuzzy model with 5 membership functions functions well and rather matches, not including some clipping at the peaks of the response. The responding output of the fuzzy model is shown in *Figure 7*:

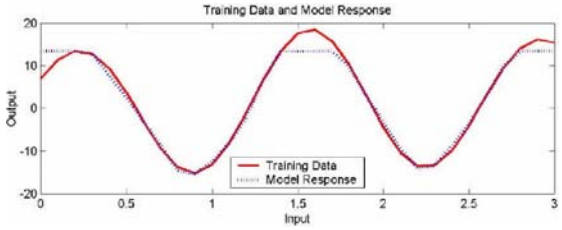

Figure 7 : Test 1A - 5 membership functions

The responding output of the fuzzy model for TEST 1C is shown in *Figure 8*, in which exploiting 20 membership functions. This response represents considerable improvement over the previous with the model error 0.2514.

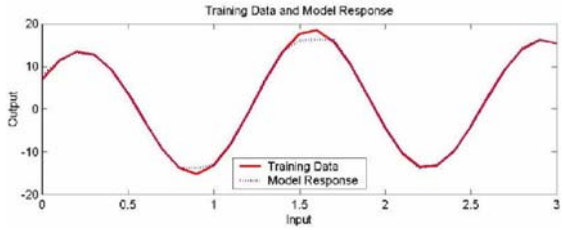

Figure 8 : Test 1C - 20 membership functions

TEST 1D with 35 membership functions reduces remarkable the model error of 0.1138 with the respond output showing in *Figure 9* .

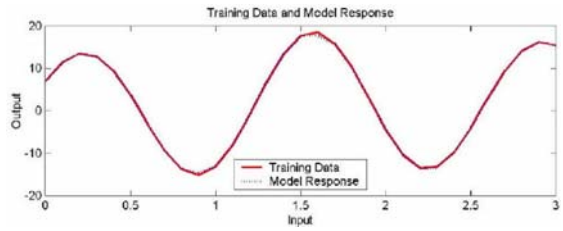

Figure 9 : Test 1D - 35 membership functions

The result for the second test which proves much more challenging is presented in Table II . *(see Appendix)*
It is easy to see that the model exploiting only 5 membership functions in TEST 2A only track simply the average value of the test function over the range. Consequently, the model error is very large. The respond output of this simple fuzzy model is illustrated in *Figure 10* .

In TEST 2C with the number of membership functions is increased to 30, the fuzzy model matches rather precisely with the dynamics of

the process. However, there are still some skipping in several sections. The respond output of TEST 2C is illustrated in Figure 11 :

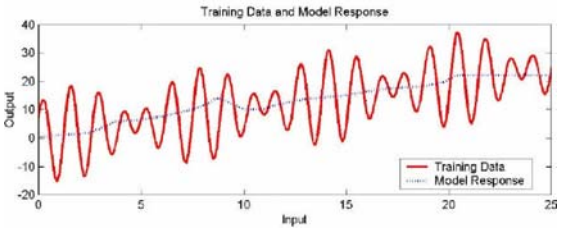

Figure10: Test 2A - 5 membership functions

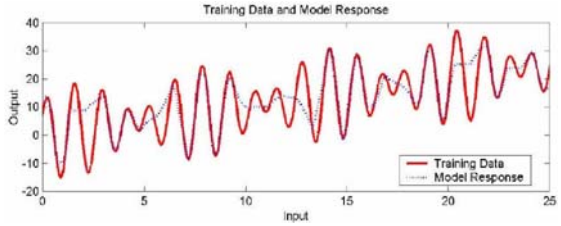

Figure11:Test 2C - 30 membership functions

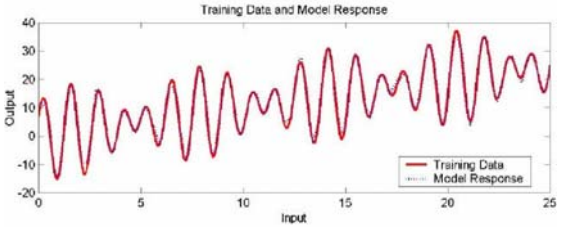

Figure 12: Test 2G - 80 membership functions

In TEST 2G using 80 membership functions, the fuzzy model is optimized with the model error being reduced to 0.5773. All the dynamics of the Test function are modeled perfectly. The respond output of this TEST 3G fuzzy model is illustrated in Figure 12.

Structure of membership function input and output in TEST 2A after being optimized by GA algorithm are showed in Figure 13 & 14 . (*see APPENDIX*)

VI. DISCUSSION :

Results proves evidently that the algorithm working well and the critical limit of the algorithm performance clearly involves the number of membership functions. The greater number of membership functions possess, the better accuracy of the fuzzy model is. An another interesting result is that, where plant requires a very simple model – for example some membership functions. In this case GA algorithm will model as posible the simplest it can. On the contrary, it is impossible to model a highly nonlinear process with just some membership functions, but it is possible to model accurately that process with a satisfied number of membership functions. All proves the GA algorithm extremely flexible.

VII. FUTURE DEVELOPMENT :

There is much scope for further development from the result of this paper .

Firstly, we can continue to apply different shaped membership functions . The paper only uses triangular membership functions for the input and output inference mechanism. It is possible that, for other plant process, Gaussian or bell-shaped membership functions types will be more efficient than triangular membership functions.

Secondly, we can allow membership functions more freedom. This GA algorithm only allows a small degree of freedom to the corner and apex of each membership function. Otherwise, it may increase the search space for the GA program.

Thirdly, we can find the way to reduce rule base. Because not all of the output membership functions will be used in the rule base, it would be useful to remove them from the model for reducing the complexity of it but still unchanging the fuzzy model performance.

Finally, we can develop this algorithm from simple SISO process to handle MIMO systems. We also intelligently change the number of membership functions to be used. It might be possible to apply an optimum number of membership functions within the GA algorithm.

VIII. CONCLUSION :

The capacity of genetic algorithm is used for optimizing a Tagaki-Sugeno fuzzy model. Such GA technique for optimizing a fuzzy model proves lots of benefits. The first is that no prior data of the process is required while modeling the fuzzy model. The second is that the algorithm is multi-objective, it optimizes not only input membership functions, output membership functions but also the rule base.

The third advantage is that the modeling procedure works well with a various and challenging data sets. The number of membership functions in use is fixed adequately at runtime permitting the best flexibility in developing a fuzzy model to adapt the plant in process. The result is in a promising trend related to the fields of system identification, system control as well as engineering for GA optimization of T-S fuzzy model. The initiative results has been obtained in this field with great hopefulness will aid further research and development in future .

REFERENCES :

[1] Passino K.M. and Yurkovich S., *Fuzzy Control* , Addison Wesley 1998

[2] Goldberg D. E., *Genetic Algorithms in Search, Optimisation and Machine Learning* , Addison Wesley 1989

[3] Hoffman F. and Nelles O. , *Genetic Programming for Modal Selection of TSK Fuzzy Systems* , Information Sciences,Vol. 136, August 2001, P. 7-28.

[4] Jennings Pamela, *Mechanical Engineering in Real-time Computer System-Fuzzy Logic*, Accessed website on 25th August 2005 : http://www.digitalbauhaus.com/fuzzylogic.html

[5] Dulay N., *Introduction to Genetic Algorithms*, Accessed website on 25th August 2005: http://www.doc.ic.acuk/~nd/journal/vol1/hmw/article1.html

[6] Mannle M., *Identifying Rule-Based TSK Fuzzy Models*, University of Karlhure, Germany, 2002

[7] Farag W.A. et al , *A Genetic-Based Neuro-Fuzzy Approach to Modelling and Control Of Dynamical Systems*, IEEE Transaction on Neural Network Vol.9, Sep.1998, p.756-767

[8] Isibuchi H. et Yamamoto T. , *Fuzzy Rule Selection by multi-objective local search Genetic Algorithm* , IEEE Transaction on System Cybernetics, Sep.1998,

[9] Papadakis S.E., Theocharis J.B. , *GA-based fuzzy modeling approach for generating TSK models*, Fuzzy Sets and Systems, Vol.131, P. 121-152 (2002)

[10] Babuska R. , *Fuzzy System, Modelling and Identification*, Delft University of Technology, 2001

[11] Maniadakis M. , Surmann H. , *A Genetic Algorithm for Structural and Parametrical Tuning of Fuzzy Systems* , European Symposium on Intelligent Techniques, ESIT99

[12] Sugeno M. , Yasukawa T. , *Lingustic Modelling based on Numerical Data*, Proceedings of IFSA91, Brussels

[13] Pedryz W. , *An Identification Algorithm in fuzzy relational systems*, Fuzzy Sets and Systems 13, p. 1-25, 1991

[14] Xu C.W., Lu Y.Z., *Fuzzy Model Identification and Self Learning for Dynamic Systems*, IEEE Transactions on Systems, Man and Cybernetics, p. 683-689, 1987

[15] Abreu Antonio et al., *Fuzzy Modelling : a Rule Based Approach* , Proceedings of the 5th IEEE Conference on Fuzzy Systems, p. 162-168,1996

APPENDIX :

Table I : TEST 1 Results

Test	Input	M.Fs.	P.size	generations	MAE
1A	0-3	5	40	1000	1.2485
1B	0-3	10	50	2000	0.6277
1C	0-3	20	60	3000	0.2514
1D	0-3	35	70	3500	0.1183

Table II : TEST 2 Results

Test	Input	M.Fs.	P.size	generations	MAE
2A	0-25	5	50	2000	7.1964
2B	0-25	10	60	2500	6.7432
2C	0-25	30	70	3000	3.7064
2D	0-25	45	70	3000	1.1861
2E	0-25	60	60	3500	0.7880
2F	0-25	70	80	4000	0.7134
2G	0-25	80	90	4000	0.5723

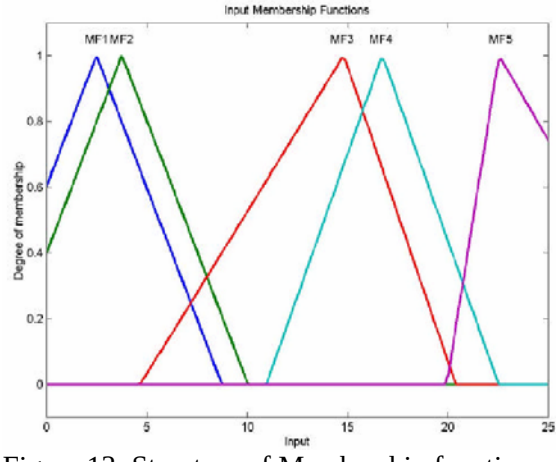

Figure 13: Structure of Membership function Input of TEST 2A

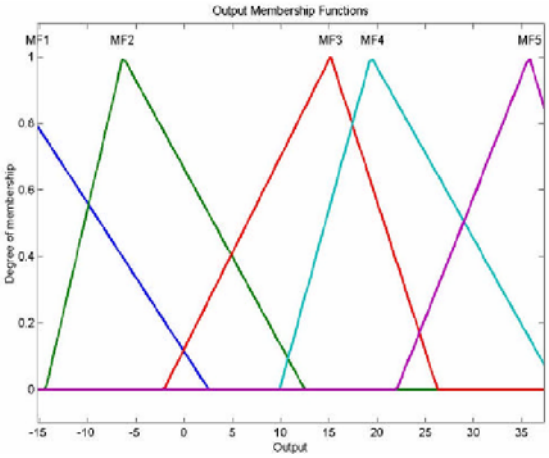

Figure 14: Structure of Membership function Output of TEST 2A