INVESTIGATIONS ON THE PERFORMANCE OF FUZZY LOGIC SYSTEM WHEN EVOLVED USING GENETIC ALGORITHM FOR DIFFERENT NUMBER OF FUZZY RULES

Shivani Kakkar[1], Satvir Singh[2], Sarabjeet Singh[3],Vijay Kumar Banga[4]
[1,2,3]SBS State Technical Campus, Ferozepur (Punjab), India,
[4]Amritsar College of Engg. & Tech., Amritsar (Punjab), India
[1]kakkarshivani47@yahoo.in, [2]Drsatvir.in@gmail.com, [3]sarabjeet_singh13@yahoo.com,
[4]v_banga@rediffmail.com

Abstract— Most of the real world problems are NP hard in nature those are difficult to be solved with traditional mathematical approaches. Traditional mathematic based search techniques are outperformed by the Evolutionary Algorithm (EA) techniques. Most of the EA have been developed on the basis of meta-heuristics of natural systems. The EA are stochastic in nature and capable of evolving near optimal solutions in reasonable amount of time. Genetic Algorithms (GA) is one of the popular EA techniques which is inspired by the process of evolution in natural systems. Fuzzy Logic System (FLS) is another paradigm that belongs to Computational Intelligence (CI) and is capable of handling uncertainty. The paper presents the detailed methodology to evolve Fuzzy Logic (FL) rules automatically using GA. Along with the methodology the paper also highlights the advantages and need of hybridization of FLS and GA. Last but not the least the scope and application of this work to other domains has been highlighted.

Index Terms— Fuzzy Logic System (FLS), Genetic Algorithm (GA), Mean Square Error (MSE), Evolutionary Algorithm (EA), Computational Intelligence (CI), Particle Swarm Optimization (PSO) .

I. INTRODUCTION

Designing a FLS includes identification of its rule base connecting the inputs and output variables. Although FLS has the ability to deal with uncertainty yet it lacks self learning and generalization of its rules [1]. So we need to adopt some optimization technique in order to search the best optimal solution to design a FLS. Optimizing a fuzzy rule based system using traditional methods have certain limitations. GA is an effective optimization technique which has the capability to search the best optimal solution from the large search space in a reasonable amount of time[11][15][16]. So in this paper we have been presenting the detailed methodology for evolving an FLS using GA for time-series forecasting problem. The brief introduction to FLS and GA has been discussed below.

A. Fuzzy Logic Systems (FLS)

Fuzzy Logic Systems was introduced by Lotfi. A. Zadeh [2] [3]. FLS is that class of computational intelligence that is capable of handling uncertain/imprecise. Unlike traditional Boolean logic which deals with only two values true or
false, i.e., 1 or 0 Fuzzy Logic is a many-valued logic which may have a truth value ranging between 0 and 1 [4][5]. FLS contains four components Rules, Fuzzifier, Inference engine, Defuzzifier. In FLS linguistic knowledge which is in the form of non-numeric statement is collected from experts. This linguistic knowledge collected from experts is represented in the form of fuzzy rules. The rules are constructed using collection of if/then statements depending upon the possible combinations of the input and output variables. The *if* part of the fuzzy rule represents the antecedent and then part

represent the consequent. Both antecedents and consequents are represented using fuzzy sets (defined by membership functions). The curve of membership function defines the degree of truth for each element in the input space. Once rules have been established an FLS can be viewed as mapping from inputs to the output and this can be expressed quantitatively as y=f(**x**) where f is highly non-linear and high order differential mapping of multiple crisp inputs with that of (multiple) crisp output(s).

1) *Type-1 FLS*-The basic architecture of Type-1 Fuzzy Logic System is shown in Fig.1.We start with fuzzification where crisp values of input data are converted to fuzzy sets using membership functions. After performing inference on the set of if-then rules the resulting output fuzzy set is used to obtain single crisp output using defuzzification. So, the Type-1 Fuzzy Logic System does not have the ability to handle uncertainty over uncertainty. In order to deal with such kind of uncertainty another type FLS known as Type-2 FLS are used [3].

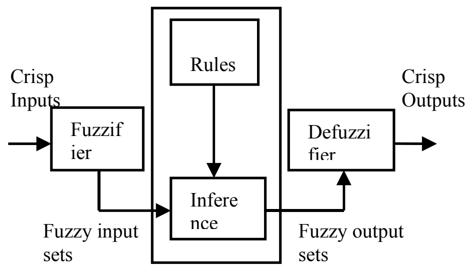

Fig. 1 Block diagram of a Type-1 FLS [**2**]

2) *Type-2 Fuzzy Logic System:* Fuzzy sets models words
that are being used in rule base and inference engine. However, word mean different thing to different people and, therefore, are uncertain. Membership degree of a Type-1 fuzzy set cannot capture uncertainties about the words. Hence, another type of fuzzy set, i.e., Type-2 fuzzy Sets, came into existence which is capable of handling such uncertainties. For such a fuzzy set membership value corresponding to some crisp input is not a crisp value rather a Type-1 fuzzy set called secondary membership [7] [6] . This concept can be extended to Type-n fuzzy sets. Computations based on Type-2 fuzzy sets are very intensive, however, when secondary membership is assumed unity the computational burden reduces drastically. This is another variant to fuzzy set representation and is known as Interval Type 2 fuzzy sets [6] [8].

B. *Genetic Algorithms (GA)*

Genetic Algorithm developed by John Holland in 1970 [9] is a global search algorithm inspired by natural mechanism of genetical improvement in biological species [10], described by Darwinian Theory of Survival of Fittest [11]. GA is widely used as an optimization tools in various fields such as medical, engineering and finance [19] and can also be used for the purpose of automatic evolution using crossover, mutation and survival of the fittest [11]. GA starts with initialization of random population consisting of vectors of chromosomes [12]. After that the fitness for each chromosome in the population is calculated using some standard fitness function. A new population is generated by applying crossover and mutation over selected chromosomes. Selection is most commonly performed using a ”Roulette wheel” mechanism. The next step crossover involves the interchanging of some values of 2 parent chromosomes depending upon the crossover probability [12]. Mutation is performed which stands for changing the values of the elements of the population randomly using mutation probability. The all new population is generated now which is copied back to the initial population in order to calculate the new fitness values. The new population will yield the improved values of fitness. The whole algorithm repeats until some required condition is not met. The detailed flowchart depicting the Genetic Algorithm is shown in Fig.2.

C. *Mackey-Glass Time-Series*

Mackey-Glass time series forecasting is a benchmark problem which is used to study the behavior of dynamically varying systems like weather and climate. The Mackey- Glass Time Series is generated using the following delay differential equation:-

$$dx(t)/dt = (0.2x(t-\tau)/1 + x^{10}(t-\tau) - 0.1x(t)$$
(1)

The application is used to predict the value at time period t depending upon the values at previous time periods (t-1), t-2), (t-3), (t-4). Therefore the FLS system needs to be designed with 4-inputs and 1-output value . Each input value has 4 Type-1 Gaussian membership functions. Following the different possible combinations of the input membership functions the possible number of rules become 4^4=256 with each rule comprising four antecedents membership functions. Further each Gaussian membership function is defined by 2 parameters, i.e., mean (m) and sigma (σ). Hence we have to deal with total 2×1024=2048 number of parameters in order to design our FLS.

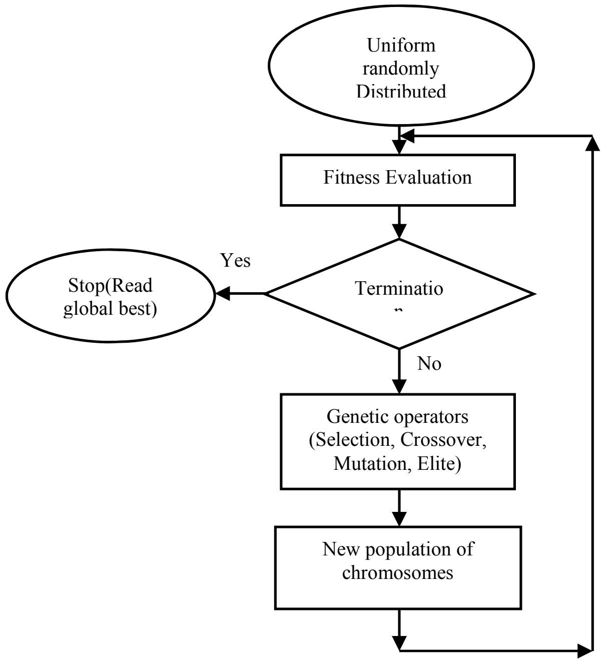
Uniform randomly Distributed
Fitness Evaluation
Yes
Stop(Read global best)
Terminatio
No
Genetic operators (Selection, Crossover, Mutation, Elite)
New population of chromosomes

Fig. 2: Genetic Algorithm Flowchart [16]

II. RELATED WORK AND OBJECTIVES

Due to the chaotic nature of the Mackey-Glass time series its prediction becomes uncertain and also we have to deal with the large number of possible rules in order to design a FLS. Hence with such a large number of possible rules the problem becomes computationally intensive. Generating fuzzy rules is the most important requirement to design a FLS. As we does not need to use all the possible rules so for the good performance of the system we need to search the best possible rules from all the possible rules. Searching for the best possible rule set from such a high dimension search space using conventional optimizations techniques becomes highly complicated. Genetic Algorithm (GA) is an effective optimization tool inspired by the process of evolution in natural systems[17]. It is a powerful technique to search suitable solutions to various optimization problems[15][16]. So instead of using conventional methods GA is one of the most popular Evolutionary Algorithm for searching optimal solution. In 1999 Yuhui Shi et. al. and Russell Eberhart et. al. [12] evolved a FLS in which fuzzy rule set including the number of rules inside it were evolved using GA. In 1991 P.Thrift et. al. [13] and in 1994 Hwang et.al. and Thompson et.al. [14] in their papers presented a methodology for combining genetic algorithms and Fuzzy Logic Systems for learning the optimal rules while fixing the shapes of the membership functions. So our objective in this paper is to investigate the performance of a FLS for time-series forecasting problem when evolved using different number of rules. Therefore along with the evolution of the best possible rules we also need to discover the maximum number of rules for which the FLS performed better.

After clarifying briefly our objectives in this paper the remaining paper is organized as follows: section III explains the implementation of evolving Fuzzy rules using Genetic Algorithms along with its detailed. Results and the performance evaluation of the system are depicted in section
IV and at last the conclusion and future work of the paper is discussed in section V.

III. IMPLEMENTATION

The implementation involves the methodology to evolve optimal design of FLS where fuzzy sets are designed using fuzzy c-mean clustering algorithm and fuzzy sets are evolved by GA. The task of evolution involves random generation of fuzzy rule base as well as optimization of fuzzy rule base from a large search space. The detailed flowchart explaining the implementation is shown in Fig. 3.

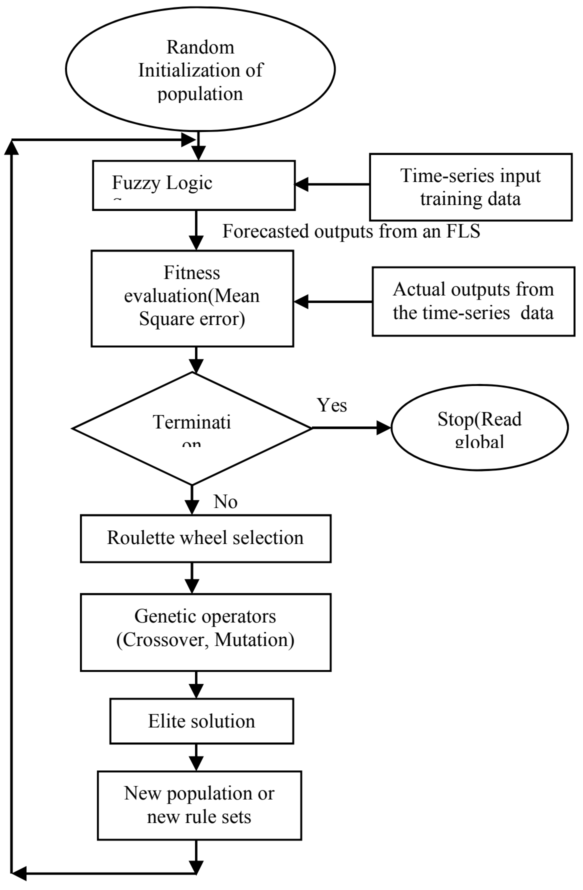
Random Initialization of population
Fuzzy Logic
Time-series input training data
Forecasted outputs from an FLS
Fitness evaluation(Mean Square error)
Actual outputs from the time-series data
Terminati on
Yes
Stop(Read global
No
Roulette wheel selection
Genetic operators (Crossover, Mutation)
Elite solution
New population or new rule sets

Fig. 3 : Flowchart of Evolutionary Fuzzy Systems using Genetic Algorithm.

A. Random initialization of population:

Initially the rule sets of an FLS to be designed are generated randomly keeping in mind the range of its antecedents and consequents. These rules are then encoded as GA population of chromosomes. At this step we have just assumed the number of fuzzy rules to be included in each rule set. Therefore we also need to evolve the number of rules to be fixed in an FLS using GA.

B. Fitness evaluation

The performance evaluation for each chromosome is done by calculating its fitness. Fitness of each chromosome is calculated using Type-1 FLS which is fed with the random rule sets from GA population and crisp inputs available from the time-series data. Antecedents and consequents of the rule sets are represented by Gaussian membership function given by:-

$$\mu(x) = \exp(-(x-m)^2 / 2\sigma^2 \quad (2)$$

where x stands for the input values, m stands for mean and σ stands for sigma[18]. The rules in each rule base are fired using input values. For implication and aggregation min and max operators are used. At last height defuzzification is performed to obtain output from an FLS[18]. Height defuzzification is given by:-

$$y = \sum_{i=1}^{M} C_i \mu(x_i) / \sum_{i=1}^{M} \mu(x_i) \quad (3)$$

where M stands for number of rules, C represents the location of singleton consequent fuzzy sets and $\mu(x_i)$ stands for the clipping level after implication. For time-series forecasting problem the function used to calculate the fitness of a particular chromosome/rule base is Mean Square Error (MSE). MSE is calculated by comparing the forecasted outputs from an FLS with that of actual output available from the time-series data. MSE is given as below:-

$$MSE = \sum_{i=1}^{N} (y_i - d_i)^2 / N \quad (4)$$

where N is the number training data, y is the forecasted output from the designed FLS and d is the actual output from the system.

C. Selection

Lesser the value of MSE, higher is the fitness of the chromosome. Selection operator is used to select the fittest chromosomes for the next generation. In order to carry out the selection Roulette Wheel concept is used. Using Roulette Wheel selection each time the wheel is spun the chromosome with higher fitness are more likely to be selected[11][12]. Therefore the chromosomes with higher fitness have the higher probability for being selected for the succeeding generations.

D. Crossover

After selection the crossover operator randomly interchanges the gene values between the parent chromosomes in order to produce better offspring's. The uniform random number (r) is generated which is compared with the crossover probability initially set as 0.09. If r is less then crossover probability the crossover between 2 chromosomes takes place otherwise no crossover takes place and original copies of parent chromosomes are produced for succeeding generations.

E. Mutation

The gene values of the chromosomes are varied using mutation operator. Mutation is carried out in order to remove the bottleneck of uniform crossover. Using one point crossover there is a possibility of 2 parent chromosomes having same string at a given gene level and the population will have the same string forever if mutation is not carried out. The mutation is carried out randomly depending upon the uniform random number generated either greater than or less than the mutation probability.

Elite solutions are applied in order to replace the worst chromosome in the population with the best one preserved earlier depending upon its fitness. After applying these GA operators the newly generated population now may have some of its elements out of range of antecedents and consequents. In order to check this we apply a validation check depending upon the range of antecedents and consequents.

Last but not the least the newly generated population is fed back to the FLS as a new improved rule set. The population again undergoes selection, crossover and mutation to produce new improved population with reduced value of MSE. So, GA iteratively improves the performance of the system by reducing the value MSE hence increasing the value of fitness level. Finally the best rule base is used to design a FLS for time-series forecasting problem.

IV. EXPERIMENTAL SETUP

The experiments were conducted on Intel(R) Core(TM) i3-2328M CPU@2.20 GHz, Window 7(Professional). The system code was written in Visual Studio C++(2012Release Mode) and was compiled on nvcc compiler. The system was evolved for different number of rules i.e. 2, 4, 6, 8, 10, 15, 20, 25, 30, i.e, the dimension size was kept 10, 20, 40, 50, 75, 100, 125, 150. The experiments were performed for 1000 and 10,000 iterations.

V. RESULT ANALYSIS

It can be observed from the simulation results shown in table 1 that the system with 10 rules, i.e, with dimension size 10×5=50 evolved with least value of MSE and hence performed better. The average value of MSE obtained was found to be 1.1475×10^{-2} using 10 rules in the rule set.

Table 1: Average minimum value of MSE obtained when FLS evolved with different number of rules for 1000 and 10,000 iterations.

Number of rules	MSE for 1000 iterations	MSE for 10,000 iterations	Average MSE
2	0.09962	0.02032	0.05997
4	0.02746	0.02764	0.02755
6	0.01852	0.01842	0.01847
8	0.01458	0.01314	0.01386
10	0.01154	0.01141	0.011475
15	0.01895	0.01907	0.01901
20	0.04457	0.04327	0.04392
25	0.07151	0.06945	0.07048
30	0.09594	0.09309	0.094515

The graph showing the comparison between the evolution of an FLS for different number of rules is shown in fig.

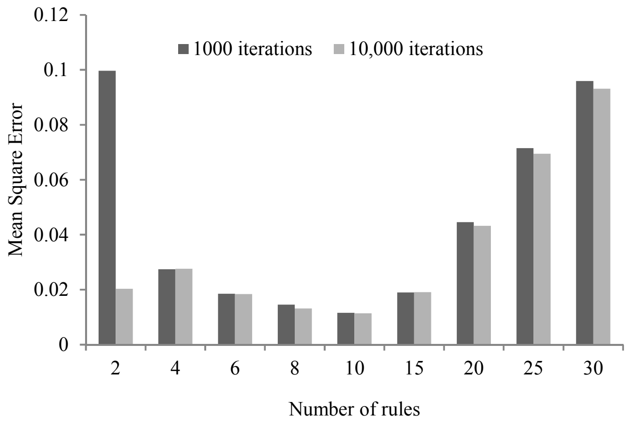

Fig. 4: Minimum value of MSE obtained when FLS evolved with different number of rules.

The graphs showing minimization of MSE using 1000 and 10,000 iterations for 10 rules are shown in Fig. 5 and 6.

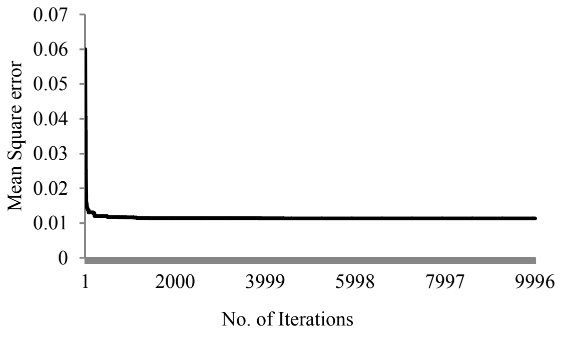

Fig. 5: Minimization of MSE for 10,000 iterations.

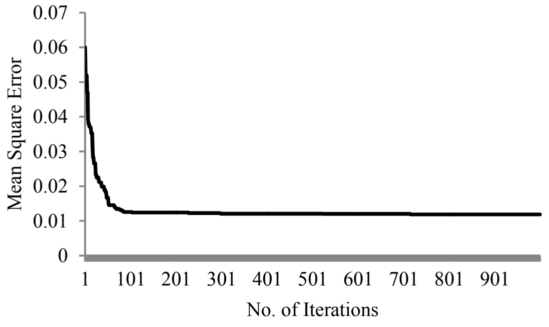

Fig. 6: Minimization of MSE for 1000 iterations.

The forecasted outputs obtained from the designed FLS using 10 rules in a Rule base were compared with the actual Time-Series outputs. The Graph of comparison is as shown in Fig. 7.

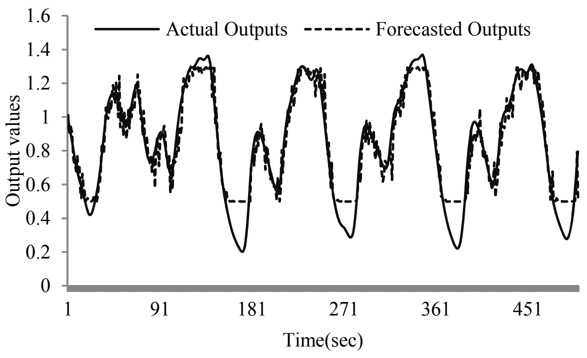

Fig. 7: Comparison between the Actual and the Forecasted time series

VI. CONCLUSION AND FUTURE WORK

The work presents the way to evolve the fuzzy if/then rules using GA for time-series forecasting problem. The designed FLS proved to be the best for forecasting outputs almost nearer to the actual outputs for 0 db SNR in a reasonable amount of time. Although the designed system for evolution of the best solution is very effective for solving time-series problem, their execution time can become a limiting factor as it involves large number of parameters that are to be determined making it computationally intensive. In future implementing the same system on Graphic Processing Unit (GPU) using Compute Unified Device Architecture (CUDA) can help in increasing the execution speed of the system. The designed fuzzy logic system can also be evolved using some different evolutionary algorithm technique like Particle Swarm Optimization (PSO) and the methodology can be applied in many other applications involving wide range of classification.

REFERENCES

[1] K. Mankad, P. S. Sajja, and R. Akerkar, “Evolving rules using genetic fuzzy approach: an educational case study,” *International Journal onSoft Computing*, vol. 2, no. 1, pp. 35–46, 2011.

[2] J. M. Mendel, “Fuzzy logic systems for engineering: a tutorial,” *Proceedings of the IEEE*, vol. 83, no. 3, pp. 345–377, 1995.

[3] L. A. Zadeh, *The concept of a linguistic variable and its application to approximate reasoning*. Springer, 1974.

[4] O. Cordon, F. Herrera, and M. Lozano, “*On the bidirectional integration of genetic algorithms and fuzzy logic*,” in Proceedings of the Second Online Workshop on Evolutionary Computation (WEC2), 1996, pp. 13–16.

[5] D. J. Dubois, Fuzzy sets and systems: theory and applications. Academic press, 1980, vol. 144.

[6] S. Singh and S. Kakkar, “Fuzzy systems using gpgpu - a survey,” in *International Conference on Communication, Computing and Systems*, Ferozepur, Punjab, India, August 2014, pp. 238–241.

[7] N. N. Karnik and J. M. Mendel, “Operations on type-2 fuzzy sets,” *Fuzzy sets and systems*, vol. 122, no. 2, pp. 327–348, 2001.

[8] O. Castillo and P. Melin, *3 Type-2 Fuzzy Logic*. Springer, 2008.

[9] J. H. Holland, “Genetic algorithms and the optimal allocation of trials,” *SIAM Journal on Computing*, vol. 2, no. 2, pp. 88–105, 1973.

[10]L. De Giovanni and F. Pezzella, “An improved genetic algorithm for thevdistributed and flexible job-shop scheduling problem,” *European journal of operational research*, vol. 200, no. 2, pp. 395–408, 2010.

[11]R. S. Sinha and S. Singh, “Optimization techniques on gpu,” in *International Multi Track Conference on Science, Engineering & Technical Innovations*, CT Group of Institutions, Jalandhar, June 2014, pp. 566–568.

[12]Y. Shi, R. Eberhart, and Y. Chen, “Implementation of evolutionary fuzzy systems,” *Fuzzy Systems, IEEE Transactions* on, vol. 7, no. 2, pp. 109– 119, 1999.

[13]P. R. Thrift, “Fuzzy logic synthesis with genetic algorithms.” in *ICGA*, 1991, pp. 509–513.

[14]W.-R. Hwang and W. E. Thompson, “Design of intelligent fuzzy logic controllers using genetic algorithms,” in *Fuzzy Systems, 1994. IEEE World Congress on Computational Intelligence*., Proceedings of the Third IEEE Conference on. IEEE, 1994, pp. 1383–1388.

[15]O. Cord´on, F. Herrera, F. Gomide, F. Hoffmann, and L. Magdalena, “Ten years of genetic fuzzy systems: current framework and new trends,” in *IFSA World Congress and 20th NAFIPS International Conference, 2001. Joint 9th*, vol. 3. IEEE, 2001, pp. 1241–1246.

[16]A. Bastian and I. Hayashi, “An anticipating hybrid genetic algorithm for fuzzy modeling,” *aNx,* vol. 7, no. 5, pp. 997–1006, 1995.

[17]Y. Yuan and H. Zhuang, “A genetic algorithm for generating fuzzy classification rules,” *Fuzzy sets and systems*, vol. 84, no. 1, pp. 1–19,1996.

[18]D. Chauhan, S. Singh, S. Singh, and V. K. Banga, “Speedup of type- 1 fuzzy logic systems on graphics processing units using cuda,” in *International Conference on Communication, Computing and Systems*, Ferozepur, Punjab, India, August 2014, pp. 267–271.

[19]A. Kumar, A. Khosla, J. S. Saini, and S. Singh, “Meta-heuristic range based node localization algorithm for wireless sensor networks,” in *Localization and GNSS (ICL-GNSS), 2012 International Conference on*. IEEE, 2012, pp. 1–7.