MULTILAYER ARCHITECTURE OF PARALLEL - GENETIC - FUZZY SYSTEM: A CASE OF EFFECTIVE TRANSPORTATION FOR CO-OPERATIVES IN INDIA

Dr. Priti S. Sajja

Abstract—The paper discusses broad architecture of parallel execution of genetic-fuzzy system by identifying limitations of the single minded traditional genetic algorithms. For additional advantages to manage uncertainty as well as other advantages related with fuzzy logic, fuzzification is also incorporated in the approach. The propose architecture of the hybrid genetic-fuzzy systems is experimented in the domain of dairy co-operatives and sample encoding, genetic operations, fitness functions and fuzzification are discussed for the case. An interface screen is also presented to demonstrate working of the prototype system. At end, advantages and future scope of the proposed work is presented.

I. INTRODUCTION

Modern industries are dealing with complex procedures and handle business at greater scope. To handle complex environment of the business and to support effective decision making process, modern technology support has become essential. Genetic algorithms deals with problem solving in large areas in its unique way of problem solving, which is inspired from mother nature. Generally genetic algorithms are dealing with multiple genetic operations on big populations, which make it slow. Further, when it is dealing with a business having wide scope in terms of number of products and geographic span of the business, the process of evolution becomes slower. Above this, genetic algorithm cannot preserve the knowledge generate for future use and explanation/reasoning purpose. To improve its working, two supporting techniques can be suggested in this paper; one is parallelization in working and another is dealing with the fuzzy linguistic values.

The paper is organized as follows. Section 2 of the paper discusses the base technologies used in the proposed architecture. These technologies include genetic algorithms and fuzzy logic. Section 3 discusses the concept of the parallel execution of the genetic algorithm as well as incorporation of the fuzzy logic in it. Section 3 also presents a generic architecture of the parallel genetic-fuzzy system. Section 4 discusses applications of the architecture in dairy industry. It discusses the broad objectives, encoding, genetic operation and fitness functions. At end of section 4, an interface screen of the prototype system is also provided to demonstrate working of the system. Section 5 concludes the paper with advantages and application of the proposed work.

II. BASE TECHNOLOGIES: GENETIC ALGORITHMS AND FUZZY LOGIC

Genetic algorithm is a technology based on the Mother Nature's principle of survival of the fittest. From very large population a few randomly selected individuals are considered as the initial population and tested for their ability to serve as solutions via a well-designed, application specific fitness test. Obviously the randomly selected candidates may not serve the purpose; hence they need to be modified. To allow possible modification, the candidates need to be encoded in a predetermined sequence of small building blocks or genes. To encode individual in genetic pattern, binary encoding scheme is generally used. One may use alphabets, number systems other than binary, trees and symbols (Akerkar R A and Sajja Priti Srinivas, 2009). Modification can be done by changing and combining required genes of the individual. The typical operations are known as (i) mutation-where a gene value is replace by other possible gen value; (ii) cross over- where strong genes (substrings) of an individual is mixed with another individual; and (iii) selection- directly selecting the stronger elements. With these typical operations, new population can be generated, which is further tested for its fitness. Number of such iterations may be carried out to get the required result. Besides, the aforementioned typical operators, application specific operators as well as generalized operators can also be designed as per need. Figure 1 describes the outline of typical genetic algorithms.

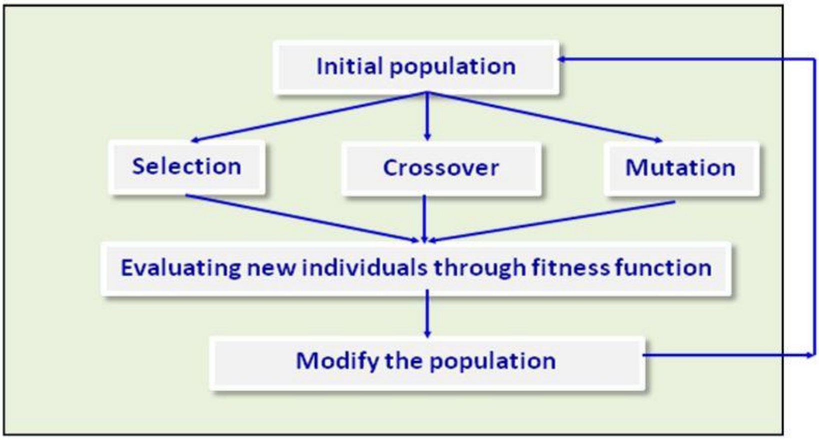

Fig. 1 Outline of the Typical Genetic Algorithms

Genetic algorithm has many advantages; the biggest among which is to be able to evolve the solution naturally in a significantly large population without bothering about the exact algorithm. Further, the genetic algorithm is applicable where there is no traditional solution available. Beside these significant advantages, the main drawback of the genetic algorithm is its unpredictable nature and no safekeeping of the knowledge generated for explanation & reasoning as well as future use. The efficiency of a genetic algorithm depends on its fitness function and application of mutation and crossover functions. If these operations and fitness function are not wisely chosen, the genetic approach may not serve the purpose. Above these, the fitness functions must not be rigid, but open enough to allow nearly fit solutions too in next generation. Observation shows that there are some poor individuals which may have a few genes very strong, which may act as key characteristics towards solution. Blindly eliminating such misfit and partly fit individuals with stronger genotype (bunch of genes) is not good practice and leads to more iterations and hence making the process of evolution slow. Here fuzzy logic can help us in overcoming the above mentioned limitations of the genetic algorithm.

Fuzzy logic (Zadeh, 1965)is based on fuzzy sets, which are sets without a boundary unlike the typical crisp sets. Typical sets, also called rigid sets have tight definition of belongingness. As per the definition of a crisp set, if an individual satisfies the definition, it is fully, completely belongs to the set; otherwise does not. There is no intermediate graded membership. Fuzzy set allows the candidate to be a partial member of the set with some degree, which is generally denoted between TRUE and FLASE or 1 and 0. This value is called membership degree to the given fuzzy set. To determine membership degree that is, the degree of partial belongingness, notion of fuzzy membership function is used. The fuzzy membership function uses linguistic variable with adjectives such as high, low, medium, etc; and maps it into a value between 0 and 1. This process is known as fuzzification. Human beings are very used to and comfortable with such linguistic variables such as high temperature, strict teacher and low speed. With the help of fuzzy membership functions, such linguistic variables can be converted into values to be further processed by machines. To convert linguistic variables into equivalent values, defuzzification needs to be done. As stated above fuzzy logic is logic based on fuzzy sets. Fuzzy logic uses the linguistic variables and provides mechanism to infer them in order to provide conclusion and reasoning. For a typical fuzzy logic based system, we need a rule base where fuzzy rules are available. To interpret a rule, repository of a fuzzy membership function is also needed. Besides, facilities such as inference mechanism, reasoning & explanation, consortium of defuzzification methods, and user interface are also needed. These components are shown in figure 2.

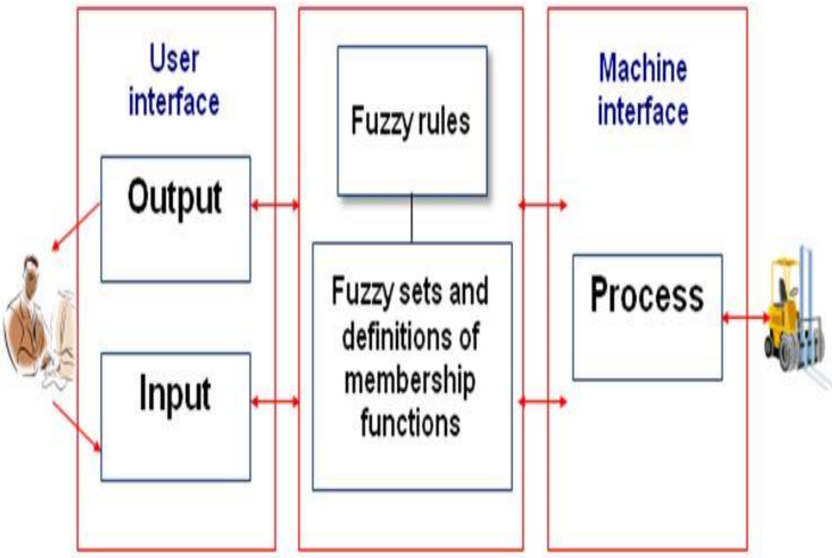

Fig. 2 Architecture of Fuzzy Logic Based System

Fuzzy logic provides advantages such as dealing with uncertainty and partial information. It also documents knowledge for future reference and explanation & reasoning procedure. Fuzzy logic can be an additional advantage to the genetic algorithms, as it can deal with not only justification of the decision taken in a form of explanation and reasoning as well as future use of knowledge but also imparting human like working mechanism. Here are the ways to collaborate genetic algorithms and fuzzy logic.

- By defining fuzzy fitness function;
- By determining loose crossover and mutation function;
- Genetic evolution of fuzzy membership functions;
- To provide explanation and reasoning for genetic algorithm;
- Fuzzy linguistic user interface supporting natural language query; etc.

Fuzzy logic and genetic algorithm can be hybridized in concurrent, co-operative or pure hybrid manner.

III. PARALLEL EXECUTION OF FUZZY GENETIC ALGORITHMS

Genetic algorithms are very applicable where the solution space is too large to handle or there is no traditional model available to get the solution. Some of the limitations of the genetic algorithms can be solved by incorporating fuzzy logic with the genetic algorithms. However, the genetic algorithm themselves are generally slow and hence costly in many ways. Incorporating fuzzy logic utility makes it further slower. Modern technology provides advance infrastructural facilities that support multitasking at operating system level and at application level. Why should not one think for parallelization of the evolution processes? Since genetic algorithms are working with various populations, which are nearly independent, parallel execution of genetic algorithm can be possible. The objective of parallelizing genetic algorithms is to make effective use of modern parallel infrastructural facilities and make the procedure faster than ever. There can be two approaches to employ parallelism using genetic algorithms namely. The first one is to divide main processes between computing resources and another one is to carry out similar processes with vectors of data (populations) using multiple processors. Since the fitness testing as well as operations are two major processes of any genetic algorithms, we propose process-wise division of work. Here many subsidiary processors can be assigned the job of performing various genetic operations such as crossover and mutation. Another highly capable machine/processor can be assigned job to verify fitness of the suggested or modified individuals. Further, these operations are aided by fuzzy logic at different capacities.

Parallel genetic algorithm was initially experimented by Bethke (Bethke, 1976) long back in the year 1976 on parallel architecture. After that, Grefenstette (Grefenstette, 1981) also suggested different parallel GA models with distributed populations. On distributed population also a model is suggested by Tanese in 1987(Tanese, 1989) and studied migration rate on a distributed population model. Some experiment of her could achieve considerable efficiency. The farmer model with a central unit for parallel execution of genetic algorithm is introduced (Cantu-Paz, 1997) later in 1997. In 2004, gradual distributed real-coded genetic algorithm outline is given by (E. Alba, F. Luna, A.J. Nebro, J.M. Troya, (2004)) using platform of local area network. Environmental intelligent transportation systems are also being popular now a days as per the claims of (Matthew Barth, Guoyuan Wu, and Kanok Boriboonsomsin , March 2016) Intelligent Transportation Systems Show Promise in Reducing Energy Consumption and GHG Emissions, Genetic algorithms in parallel manner are also being used in task scheduling at application level as well as at microprocessor level (A.S. Wu, H. Yu, S. Jin, K.-C. Lin, G. Schiavone, 2004) and (Fatma A. Omaraa, Mona M. Arafa, 2010). Some analysis of papers on the intelligent transportation system is presented in work of (José A. Moral-Muñoz, Manuel J. Cobo, Francisco Chiclana, Andrew Collop, and Enrique Herrera-Viedma, 2015). Work presented in the (Mohammadreza Ghatreh Samani, Seyyed-Mahdi Hosseini-Motlagh, vol 5, isssue 1, 2017) utilizes hybridization of simulated annealing and genetic algorithms in the domain.

Parallel genetic algorithm with fuzzy logic incorporation is proposed by (Tzung-Pei Hong, Yeong-Chyi Lee, Min-Thai Wu, 2014) for associative rule mining. However, these solutions are application dependent. In some parallel model, parallel hardware is also utilized. (Mohammadreza Ghatreh Samani, Seyyed-Mahdi Hosseini-Motlagh, vol 5, isssue 1, 2017)

In the proposed model, we have considered only a single processor to employ parallelism in genetic algorithm; however, advantage of the hardware parallelization can be achieved for better efficiency. Figure 3 illustrates the way how genetic algorithms can be paralleled along with fuzzy intervention. The architecture works in this way. First of all, through users interface request is handled and sent to the control processor. The control processor generates initial population by taking help of encoding mechanism suggested in the architecture. Here the encoding may use fuzzy linguistic variables to deal with uncertainty and ambiguity in the encoding process as well as the individual capabilities in many ways. The control processor first checks the fitness of the randomly generated individual and poorly fit individuals are discarded. Selection of the individual is proportional (Roulette wheel based) or through ranking.

Remaining individuals are sent to the subordinate processors for parallel operations such as mutation, cross over and edge recombination. The number of subordinate processors is determined in this way:

$$\text{subprocessors} = \sqrt{(\text{NI} * \text{t}/\text{r})}$$

Where subprocessors is number of subordinate processors, NI is number of individuals in undistributed population, t is the average time to perform genetic operation, and r is time taken to respond to the control processor. The crossover frequency can be set to 60 to 70% and 2 point crossover is suggested. For mutation frequency is set to 30-to 40%. Maximum generation is fixed to 1000 at this stage; however, it is determined by the user experts depending on the size of the population.

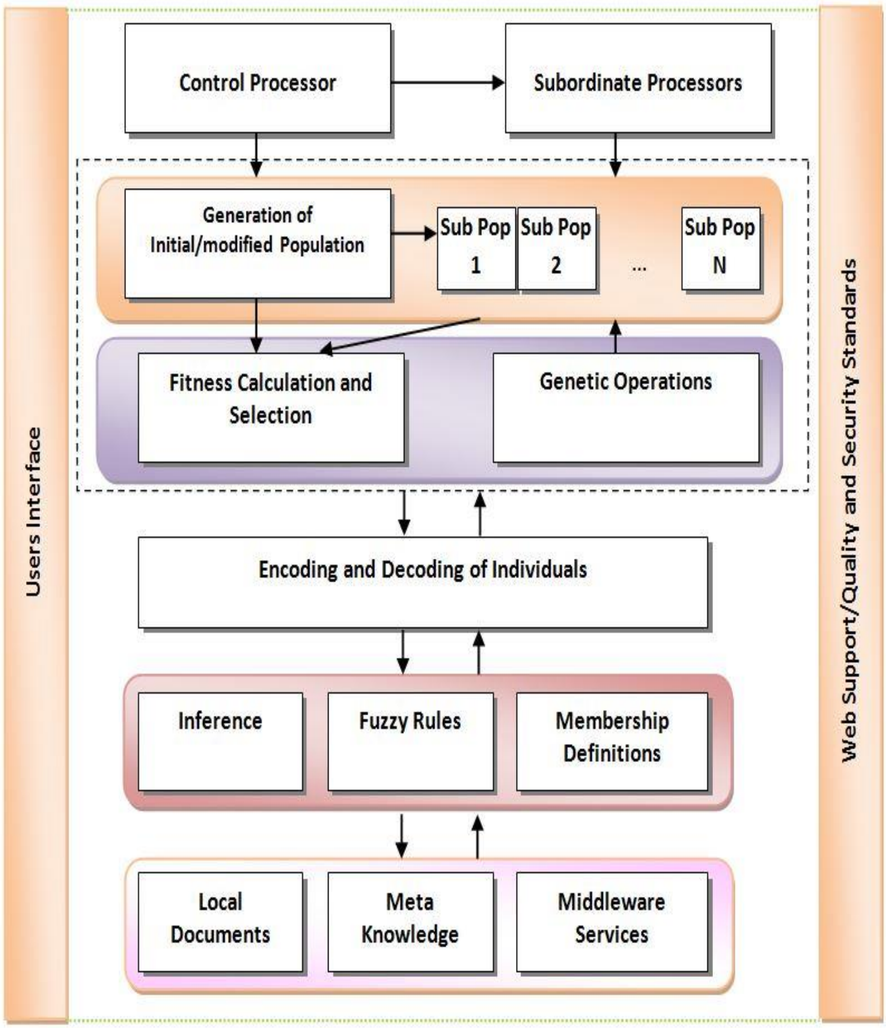

Fig. 3 Multilayer Architecture of Parallel - Genetic - Fuzzy System

This procedure is carried out in repetitive manner. We may think of application specific operators or fuzzy operators here. Second iteration onwards, population is not randomly generated, but modified population is taken into consideration.

The control processor determines (may be with help of users interactions) when to seize the process and decode the solution, as the solution may be in fuzzy form. Further, fuzzy rules can be used to fine tune the result and also provide explanation based on the inference procedure employed. Third party or middle ware services can also be used, if ready fuzzy membership functions are available. Next section illustrates use of the proposed model in domain of dairy co-operatives.

IV. APPLICATION OF THE PROPOSED ARCHITECTURE IN DAIRY INDUSTRY IN INDIA

A co-operative society is an autonomous group of users to meet their common needs in a given domain. The needs can be economical, social or cultural. Such societies are jointly owned and managed in co-operative manner. For milk related business, a cooperative society model is established by Dr. Verghese Kurien with H.M. Dalaya in the year 1946 in first in Gujarat and later India. The inspiration came from Hon Sri Sardar Vallabhbhai Patel, Hon Morarji Desai and local leader in Gujarat Shri Tribhuvandas K Patel. The main objective is to deal withunfair market practices and milk distribution as well as prices. Dr. Kurien is known for India's white revolution and Dr. Dalaya is known for his innovation of making skim milk powder from buffalo milk (for the first time in the world). Amul[www.amul.com] is an Indian dairy cooperative, based at a town called Anand in the state of Gujarat, India.

The co-operative societies work in different levels; village level – where milk is collected and farmers were given prices (at rate fixed by upper level) for the milk as per the quality and quantity of the milk at the place. The collected milk is taken to the processing units and generated products are marketed by a state level marketing agency. Each state in India has a state level marketing unit. All such state level marketing units forms a consortium. There is a national level apex body, which manage production, marketing and distribution of the milk and milk related products. It is to be noted that the established structure is also being utilized for various awareness and rural development related programmes. At present the apex body in India (National Co-operative Dairy Federation, India-NCDFI) manages 26 state level federations and 1.44 lacks village level co-operative societies. In such a large and nationwide business management of multiple dairy related products is not an easy job. This leads to effective utilization of technology for the business. As the dairy industry deals with fragile and time bounded milk related products, which expires early, the transportation of such product require effective planning and utilization of resources. Further, such a product falls under the class of the basic necessities; it must be delivered to each and every corner of the area where supply is committed. Procurement of milk and other raw material as well as distribution of the semi-finished and finished product is a challenging work. This section presents outline of the experimental prototype designed using the generic model discussed in the figure 3.

A. Encoding:

Every individual consist of information about various aspects as follows:

- Station code: Five bits are reserved for the station code. Each station can be encoded further accommodating the state information, type code, and serial number.
- Station capacity: capacity to store and handle amount of product depending on infrastructural facilities available at the station. Three digits are reserved for the same.
- Station demand: fuzzy variable in three digits showing demand and/or frequency of the product. VHG, HGH, MED, LOW, VLW etc. linguistic values can be used here.
- Feedback: previous experience about the station can be accommodated using an alphabet. The feedback value is generally collected from vendors and users dealing with the station. In absence, system administrator can also provide this value. One digit will be enough for this. This field is optional.
- Rank of the station: depending of the history and experience of the station, 3 digited fuzzy rank value can be assigned by the upper level body or supplier manager.

To accommodate all above information we suggest a significant digit subset coding and formation of genetic encoding of 15 bits. Figure 4 illustrates the gene structure.

Station Code					Capacity			Demand			Feedback	Rank		
State + Type + Srno					*Fuzzy*			*Fuzzy*				*Fuzzy*		
1	2	3	4	5	6	7	8	9	10	11	12	13	14	15

Fig. 4 Gene Structure for an Individual

Examples of encoded individual using the aforementioned scheme are as follows.

TNR03HGHHGH1AVG
GUR09LOWMED0LOW
PNG01HGHVGH2HGH

Besides the encoded individuals, there is a need of distance metrics to calculate distance between the source and destination stations in kilometers/miles. Database of the station codes is also maintained. This database will be useful to cross verify the stations information, once genetic operators generate the modified populations. The auto-generated station will be verified for its possibilities/existence via this database.

B. Genetic Operations:

Mutation is a kind of independent operation, and can be carried out in parallel manner easily. Mutation on individual is done as follows. For mutation a expert defined probability called Prob_M. Figure 5 illustrates the process of mutation on an individual. Crossover requires two individuals and can be performed by interchanging substrings representing common characteristics. Figure 5 (b) illustrates the crossover procedure on individuals.

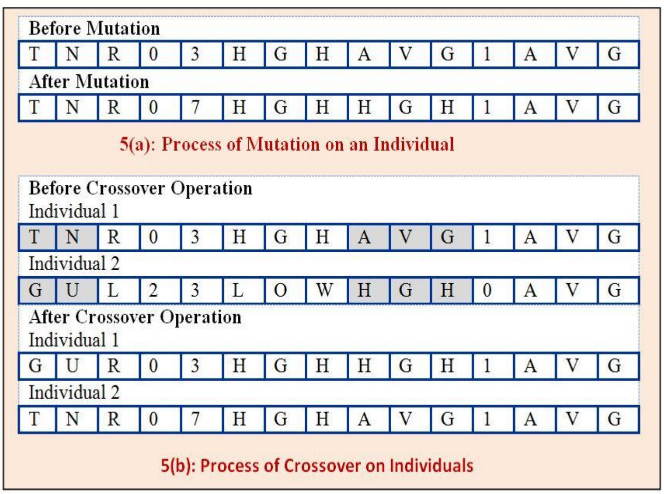

Fig. 5 Genetic Operations: Mutation and Crossover

The fuzzy values used in encoding are determined by average of various fuzzy functions at lower level. For example, to determine fuzzy value related to the capacity of a station, then capacity of the station for the various products as well as infrastructure utilities need to be considered. For multiple products, we may consider individual fuzzy functions (as the products are of different nature/ units), which further with a help of fuzzy rule can be integrated.

As the encoding of individual is done using fuzzy linguistic variables, fitness function should also incorporate fuzzy variables while checking for the suitability of an individual. The prime objective of the exercise is to minimize the routing effort and cost and to incorporate stations with high demand. Hence the fitness function can be designed as follows:

Fitness function F_I = {P_{State} *Match (Station code, Required Station) + P_{Demand}*Match(Demand, "HGH")+ Match (Feedback, '1') + P_{RANK}*Match(Rank, "HGH")}

The interface screen for the proposed system is illustrated in figure 6.

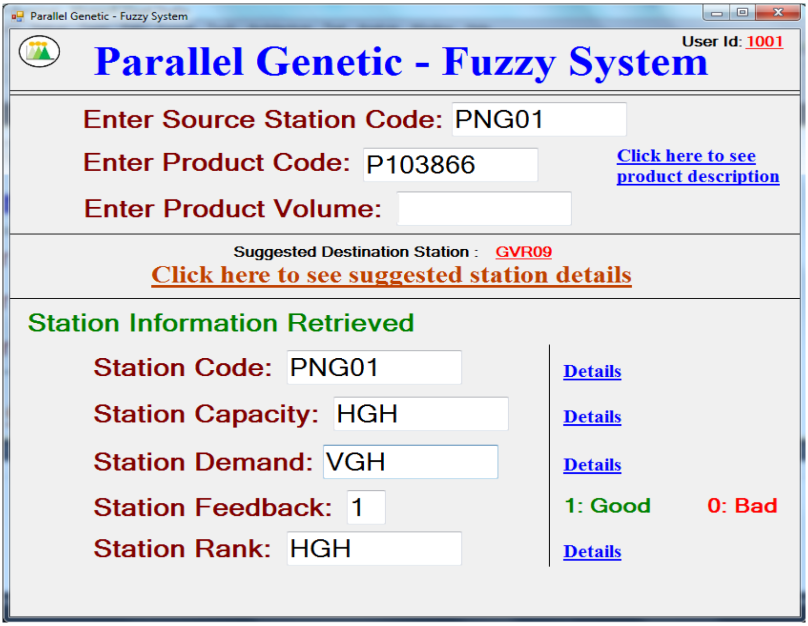

Fig. 6 Interface Screen of the Prototype System

V. CONCLUSIONS

The architecture presented in this paper discusses parallel execution of genetic algorithm supported by fuzzy logic. The parallel execution offers advantages as follows.

- Better efficiency in terms of speedy execution;
- Possibility to yield alternative solution;
- Can take advantage of parallel hardware;
- Easy co-operation with other exiting problems solving mechanism such as searches; etc.

With help of fuzzy logic embedded with the parallel genetic algorithms, advantages to handle voluminous data can be effectively reduced. Further fuzzy logic deals with the partial and uncertain information; hence it is easy to handle not only voluminous data, but also varacity of the data. Many times, fuzzy logic acts as good instrument for missing data also. The proposed architecture is generic in nature and supports applications from various domains without changing much in its design. Further, it also provides opportunities to develop application specific fuzzy logic components such as membership functions, defuzzification methods and inference mechanism. One can also think for domain specific as well as generalized fitness functions and genetic operators along with encoding schemes such as binary, tree and alphabet encoding.

REFERENCES

[1] A.S. Wu, H. Yu, S. Jin, K.-C. Lin, G. Schiavone. (2004). An incremental genetic algorithm approach to multiprocessor scheduling. *IEEE Trans. Parallel Distrib. Syst. 15* , 824-834.

[2] Akerkar R A and Sajja Priti Srinivas. (2009). *Knowledge Based Systems.* MA, USA: Jones & Bartlett Publishers.

[3] Bethke, A. (1976). Comparison of genetic algorithms and gradient-based optimizers on parallel processors:efficiency of use of processing capacity. *Tech. Rep. No. 197, University of Michigan, Logic of Computers Group, Ann Arbor, MI* .

[4] Cantu-Paz, E. (1997). Designing efficient master–slave parallel genetic algorithms. *Technical L Report No. 97004, University of Illinois at Urbana-Champaign, Illinois Genetic Algorithms Laboratory, Urbana, IL* .

[5] E. Alba, F. Luna, A.J. Nebro, J.M. Troya. ((2004)). Parallel heterogeneous genetic algorithms for continuous optimization . *Parallel Computing 30* , 699–719.

[6] Fatma A. Omaraa, Mona M. Arafa. (2010). Genetic algorithms for task scheduling problem . *Journal of Parallel Distrib. Comput. 70* , 13-22.

[7] Grefenstette, J. (1981). Parallel adaptive algorithms for function optimization. *Tech. Rep. No. CS-81-19, Vanderbilt University, Computer Science Department, Nashville, TN* .

[8] José A. Moral-Muñoz, Manuel J. Cobo, Francisco Chiclana, Andrew Collop, and Enrique Herrera-Viedma. (2015). Analyzing Highly Cited Papers in Intelligent. *IEEE TRANSACTIONS ON INTELLIGENT TRANSPORTATION SYSTEMS* , 1-9.

[9] Matthew Barth, Guoyuan Wu, and Kanok Boriboonsomsin . (March 2016). Intelligent Transportation Systems Show Promise in. *National Centre for sustainable Transportation, Center for Environmental Research and Technology (CE-CERT) University of California, Riverside,* .

[10] Mohammadreza Ghatreh Samani, Seyyed-Mahdi Hosseini-Motlagh. (vol 5, isssue 1, 2017). A Hybrid Algorithm for a Two-Echelon Location-Routing Problem with Simultaneous Pickup and Delivery under Fuzzy Demand. *International Journal of Transportation Enginering* .

[11] Tanese, R. (1989). Distributed genetic algorithm, . *Proceedings of Third International Conference on Genetic algorithms* , pp. 434–439.

[12] Tzung-Pei Hong, Yeong-Chyi Lee, Min-Thai Wu. (2014). An effective parallel approach for genetic-fuzzy data mining. *Expert Systems with Applications, Volume 41, Issue 2* , 655-662.

[13] Zadeh, L. A. (1965). Fuzzy Sets. *Information and Control*, 338-358.

AUTHORS' PROFILE

<table>
<tr>
<td></td>
<td>Dr. Priti S. Sajja

She has been working as a Professor at the Post Graduate Department of Computer Science, Sardar Patel University, India. Her research interests include knowledge-based systems, soft computing, multi-agent systems, and software engineering. She has produced more than 180 publications in books, book chapters, journals, and in the proceedings of national and international conferences out of which five publications have won best research paper awards.</td>
</tr>
</table>