I.J. Modern Education and Computer Science, **2021, 2, 23-32**
Published Online April 2021 in MECS (http://www.mecs-press.org/)
DOI: 10.5815/ijmecs.2021.02.03

Agent-based Modeling in doing Logic Programming in Fuzzy Hopfield Neural Network

Shehab Abdulhabib Saeed Alzaeemi
School of Mathematical Sciences, Universiti Sains Malaysia, 11800 USM, Penang, Malaysia
E-mail: Shehab_alzaeemi@yahoo.com

Saratha Sathasivam
School of Mathematical Sciences, Universiti Sains Malaysia, 11800 USM, Penang, Malaysia
*Corresponding author: saratha@usm.my

Muraly Velavan
School of General & Foundation Studies, AIMST University, 08100 Bedong, Kedah, Malaysia
E-mail:muraly@aimst.edu.my

Received: 28 October 2020; Revised: 06 December 2020; Accepted: 18 January 2021; Published: 08 April 2021

Abstract: This paper introduces a new approach to enhance performance in performing logic programming in the Hopfield neural network by using agent-based modeling. Hopfield networks have been broadly utilized to solve problems of combinatorial optimization. However, this network yielded a satisfiability problem because the network has grown larger, and it is more complex. Therefore, an improved algorithm has been proposed to enhance the Hopfield network's capability by using the technique of fuzzy logic to provide more efficient energy relaxation and to avoid the local minimum solutions. Agent-based modeling has been introduced in this paper to conduct computer simulations, which aim at verifying and validating the introduced approach. By applying the technique of fuzzy Hopfield neural network clustering in the system, better quality solutions are produced, and the network is handled better despite the increasing complexity. Also, the solutions converged faster by the system. Accordingly, this technique of the fuzzy Hopfield neural network clustering in the system has produced better-quality solutions.

Index Terms: Fuzzy Hopfield neural network, logic programming, agent-based modeling.

1. Introduction

The neural network is a parallel processing network, generated by the simulation of the intuitive thinking of the human's image drawing upon exploring the biological neural network with regard to the biological neurons, as well as the features of the neural network through simplifying, summarizing, in addition to refining [1]. Moreover, the non-linear mapping idea is utilized, the parallel processing method, in addition to the neural network's structure for expressing the related knowledge of input, as well as the output [2]. The Hopfield model [3, 4] can be defined as a sole layer recursive neural network, whereby each neuron output can be associated with every other neuron input. Hopfield neural network's energy function is a quadratic function, which is joined with the objective function so that the problems of optimization are minimized [5].

A method is introduced by Wan Abdullah [6] of performing a logic program in the Hopfield network. The network conducted a logical inconsistency minimization in a program after defining the connection strengths by a logic program, that is, through equating cost function, in addition to energy function. The network, after that, relaxes to the neural states, which are the models (i.e., viable logical interpretations) of the corresponding logic program. The direct method involves introduced learning by Wan Abdullah [7].

In this paper, the technique of the fuzzy Hopfield neural network clustering, combined with a modifying activation function, has been introduced with the aim of solving combinatorial optimization issues. Thus, the fuzzy Hopfield neural network involves a specific numerical procedure. It can minimize an energy function so that the membership grade is located [8]. Combinatorial optimization involves locating a set of options among a discrete set to provide the most optimum value of the corresponding cost function [9]. According to fuzzy Hopfield neural network, the algorithm of fuzzy *c*-means clustering has been imposed to activate the states of the neuron without exception in a Hopfield neural network. Accordingly, using the method of fuzzy Hopfield neural network clustering has been analyzed with the modification of an activation function to fast move computation to perform the logic programming in the given Hopfield network.

Furthermore, agent-based modeling for the proposed method is introduced in this paper. In FCMHNN, a unique Hopfield network was modified with the addition of fuzzy c-means clustering approach. The energy function, then, could be swiftly converged to local minimum. Compared to the conventional technique of HNN, the introduced FCMHNN fundamental strength showed more computation efficiency because of integral corresponding structures. Based on the simulation study, FCMHNN showed that it is capable of yielding promising results compared to HNN.

In the subsequent section of the paper, the focus is placed on the Hopfield neural network. Next, the paper investigates how logic learning has been carried out in the Hopfield network followed by explaining how fuzzy logic is integrated into the Hopfield network. After that, a discussion about agent-based modeling is provided. Finally, the conclusion of the study is provided in the last section of this paper.

2. Hopfield Neural Network

The Hopfield Neural Network (HNN) provides a model that simulates human memory. It has a wide range of applications in artificial intelligence, such as machine learning, associative memory, pattern recognition, optimized calculation, VLSI and parallel realization of optical devices. The Hopfield model can be defined as a sole layer recursive neural network, whereby each neuron output can be associated with every other neuron input. The Hopfield network's energy function, which is a specific quadratic function, can be connected to an objective function to reduce optimization issues.

The Hopfield network, involving *n* units, is having a couple of versions. These involve binary, in addition to the continuously valued. Consider v_i is a state or an output of i th unit. Regarding the binary networks, v_i denotes +1 or -1, however, regarding continuous networks, v_i denotes a specific value between 0 and 1. Consider w_{ij} is synapse weight on the connections from units i to j . According to Hopfield networks, $w_{ij} = w_{ji}$, $\forall i, j$ (symmetric networks), as well as $w_{ii} = 0$, $\forall i$ (i.e., no self-feedback connections). The dynamics of the network of the given binary Hopfield network involve:

$$v_i = \operatorname{sgn}\left(\sum_j w_{ij} v_j - threshold\right) \tag{1}$$

The energy function of the discrete Hopfield network can be provided by:

$$E = -\frac{1}{2}\sum_i \sum_j w_{ij} v_i v_j - \sum_j w_i v_i \tag{2}$$

Fig.1 shows the discrete Hopfield network as the extended form of the common representation of the given Hopfield network. In Hopfield and Tank's work [3], the network can be usefully utilized to solve problems of combinatorial optimization as a content addressable memory or as an analog computer. Thus, combinatorial optimization is to locate a set of options from a given distinct set that can produce the best value of several connected cost function.

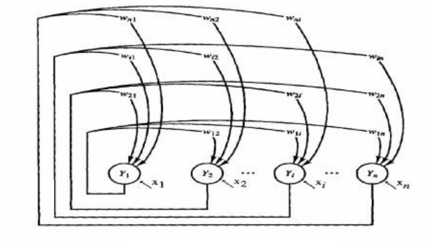

Fig. 1. Discrete Hopfield neural network

Because the network progresses based on the dynamics, a given energy E can be decreased only or remained unaffected at each update, which can be attributed to changing ΔE because of the change of the output is either zero or negative. Thus, the network converged to a (local) minimum energy state. This is due to *E*, bounded below. The local minimum points in each energy landscape corresponded to prototype patterns stored at a given storage phase. This indicates that the retrieval of Hopfield network is approximately equal to the given target patterns.

3. Logic Programming in Hopfield Network

This section reviews the Little-Hopfield model [4], involving a content addressable memory standard model. The Little-Hopfield neural network [4, 6] minimizes a Lyapunov function, also known as the energy function due to obvious similarities with a physical spin network. Thus, it is useful as a content addressable memory or an analog computer for solving combinatorial type optimization problems because it always evolves in the direction that leads to lower network energy. This implies that if a combinatorial optimization problem can be formulated as minimizing the network energy, then the network can be used to find optimal (or suboptimal) solutions by letting the network evolve freely. Little dynamics are asynchronous, whereby every neuron individually updates its state. Thus, the system comprises *N* formal neurons, represented via the Ising variable $S_i(t),(i=1,2,....N)$. The Bipolar neurons, $S_i \in \{-1,1\}$ following the dynamics $S_i \rightarrow \text{sgn}(h_i)$, whereby a local field, $h_i = \sum_j J_{ij}^{(2)} V_j + J_i^{(1)}$, *i* as well as *j* connect the entire neurons *N*, $J_{ij}^{(2)}$ signifies the connection (synaptic) strength from the given neuron *j* to the given neuron *i*, and $-J_i$ signifies the threshold of the given neuron *i*. These are symmetric connections; they are zero-diagonal, $J_{ij}^{(2)} = J_{ji}^{(2)}$, $J_{ii}^{(2)} = 0$.

This can monotonically decrease with energy relaxation. A further generalization of the two-neuron model contains higher-order connections, which changes “field” into:

$$h_i = ...+\sum_j \sum_k J_{ijk}^{(3)} S_j S_k + \sum_j J_{ij}^{(2)} S_j + J_i^{(1)} \tag{3}$$

whereby “…..” refers to the higher orders; the Lyapunov energy function is written as in the following:

$$E = ...-\frac{1}{3}\sum_i \sum_j \sum_k J_{ijk}^{(3)} S_i S_j S_k - \frac{1}{2}\sum_i \sum_j J_{ij}^{(2)} S_i S_j - \sum_i J_i^{(1)} S_i \tag{4}$$

If the $J_{ijk}^{(3)} = J_{[ijk]}^{(3)}$ of the *i, j, k* distinct, with […] signifying the permutations in a given cyclic arrangement; the $J_{ijk}^{(3)} = 0$ of any *i, j, k* equal; identical symmetry conditions are met for higher-order connections. Thus, the updating rule is provided as in the following equation:

$$S_i(t+1) = \text{sgn}[h_i(t)] \tag{5}$$

Logic programming has been performed in the Hopfield network following the method by Wan Abdullah.

i) The entire clauses in the logic program are translated to the basic form of Boolean algebraic.
ii) The neuron is initialized to each ground neuron.
iii) The values of the synaptic strengths are initialized to zero.
iv) The cost function is calculated, which is linked to the clauses negation like $\frac{1}{2}(1+S_x)$ signifying the neuron’s logical value *X*, whereby S_x is a neuron that is associated with *X*. Thus, the S_x value can be defined as it can carry values of 1 when *X* can be true, but -1 in case *X* can be false. A negation (i.e., the neuron *X* does not occur) is shown by $\frac{1}{2}(1-S_x)$ and, hence, the conjunction logical connective can be obtained via multiplication. However, the disjunction connective can be obtained via addition.
v) The synaptic strengths values are obtained via equating cost function, in addition to energy, *H*.
vi) In the end, neural networks evolve until locating minima energy and computing whether this is the wanted global solution.

4. Fuzzy Logic

This work introduces the method of fuzzy Hopfield neural network clustering. The study aims to tackle logic programming in the Hopfield network. The utilized technique can be defined as a specific numerical procedure, which is used for minimizing energy function so that the membership grade is found. This technique has been widely utilized based on the Lyapunov energy function. This technique can be usefully implemented to tackle combinatorial optimization issues as an analog computer or as a content addressable memory. Combinatorial optimization is to locate a set of options from a distinct set that can produce the best value for several associated cost functions. According to the fuzzy Hopfield neural network, the fuzzy *c*-means clustering algorithm was imposed so that the states of the neuron are updated in the Hopfield neural network [10, 11].

A. Fuzzy Clustering Technique

Different algorithms are available according to the criterion of the least mean squares for the problem of clustering [12, 13]. The fuzzy *c*-means (FCM) clustering algorithms are famous techniques with the best performance among the entire fuzzy clustering algorithms. The techniques of fuzzy clustering represent mathematical gears so that resemblances are identified amid members of the collection samples. The fuzzy clustering technique has undergone extensive studies and analyses by researchers. The algorithm of fuzzy *c*-means clustering has been utilized to minimize the corresponding criterion function; originally proposed by Dunn [14]. However, it has been extended later to a relevant formulation, including the advanced algorithm by Bezdek [15] to an unlimited family of objective functions.

Clustering involves partitioning or classifying the collection of samples (or objects) to detached subsets or separate clusters. It has been assumed based on the fuzzy *c*-means clustering that there are specific predefined *c* clusters in the data sets. The minimization of an objective function can be carried out so that the ideal set of clusters is located. The objective function denotes the Euclidean distance amid data samples, as well as the cluster center. The fuzzy *c*-means method aims at minimizing the criterion in the least squared error sense. Concerning $c \geq 2$ and value of the fuzzification parameter, *m* is chosen from a real number above 1. Thus, the algorithm will be degenerating to a crisp clustering algorithm in case *m=1*. Following Bezdek [15], the technique of fuzzy *c*-means clustering can be provided as in the following:

Consider $Z = \{z_1, z_2, ..., z_n\}$ is an unlimited unclassified data set, whereby z_x signifies the sample of training. The fuzzy *c*-partition of *Z* signifies the fuzzy subsets' family of *Z*, represented by $P = \{A_1, A_2, ...A_z\}$, whereby *c* represents a predetermined number of the cluster number. Thus, the membership grade demonstrated by μ_{xi} shows the potential that z_x belongs to *i*th fuzzy cluster.

The membership grade μ_{xi} represents the value between zero and the one that can satisfy [15]:

$$\sum_{i=1}^{c} \mu_{xi} = 1, \quad for \quad x = 1, 2, 3, \ldots, n \tag{6}$$

$$0 < \sum_{i=1}^{c} \mu_{xi} < n, \quad for \;\; i = 1, 2, 3, \ldots, c \tag{7}$$

Using equation (7), the advanced fuzzy *c*-means algorithm has updated the membership grade as follows:

$$\mu_{x_i} = \frac{\left(\dfrac{1}{(d_{x,i})^2}\right)^{\frac{1}{m-1}}}{\sum_i^c \left(\dfrac{1}{(d_{x,i})^2}\right)^{\frac{1}{m-1}}} \tag{8}$$

where $d_{x,i}$ signifies the Euclidean distance amid the training pattern z_x , with a center of class, v_i provided in the following equation (9):

$$d_{x,i} = \left\| z_x - v_i \right\| \tag{9}$$

Based on equation (9), *m* can be characterized as the parameter of fuzzification or the exponential weight. Thus, the fuzzification parameter, *m* has been selected at any value between 1 and ∞. It can be utilized for dominating the membership grade influence. Hence, the cluster centers can reduce noise sensitivity in the class centers' computation. Therefore, the influence of μ_{xi} draws upon the value of *m*, whereby the greater the value of *m*, the higher fuzziness is provided. When *m=1* , it cannot be selected because the algorithm will be degenerating to the crisp clustering algorithm.

B. Fuzzy C-Means (FCM) Hopfield Neural Network

Hopfield neural network can be referred to as a familiar technique utilized for solving the optimization problem depending on Lyapunov energy function as provided in equation (4).

Consider S_i signifies the binary states of the neuron *i*, and J_{ij} signifies the synaptic weight between the neurons *i* and *j*. In conformity with the traditional Hopfield neural network, the neuron in firing state, e.g., $S_i = 1$ shows that the sample z_x belongs to the class *i* . The neuron state has been set to one or zero, and it signifies that the neuron state *i* is either firing or not firing, respectively. However, in the fuzzy Hopfield neural network, the fuzzy state neuron shows that the sample z_x , belongs to class *i* to a specific uncertainty degree described by the membership function. Thus, the Lyapunov energy function in equation (5) has been modified as follows:

$$H = -\frac{1}{3}\sum_i \sum_j \sum_k J^{(3)}_{[ijk]} S_i S_j S_k \mu_{xi} - \frac{1}{2} J^{(2)}_{[ij]} S_i S_j - \mu_{xi}(NN) \tag{10}$$

whereby μ_{xi} denotes membership grade, whereas *NN* denotes the number of neurons. The fuzzy Hopfield neural network has introduced the concept of the fuzzy set into the Lyapunov energy function. The algorithm can be summarized as follows:

Step 1: Initially, the cluster centers are defined, $V_i (2 \le i \le c)$, with the parameter of fuzzification m $(1 \le m \le \infty)$.

Step 2: The membership matrix, μ_{xi} is calculated using equation (8).

Step 3: After achieving the value of the fuzziness, the energy function has been modified in equation (10).

5. The Methodology of Neural Networks Simulator Model

Users are usually familiar with using the system; however, they are not familiar with why the simulator needs to be built, as well as the advantage of accurate results. Therefore, further description is provided in this section. The simulator of the Hopfield networks' simulator, which utilizes a traditional computer, is formulated rather than continue building a new design of the network or storing a novel memories' set. By doing so, a lot of time and energy can be saved for programmers, who wish to start rebuilding a novel system from scratch. Accordingly, the exactly matching computer program with the user's demand is constructed so that for the Hopfield Network action is simulated. Thus, the programmer can easily perform program modification, as well as storing a new data set. Agent-based modeling is designed so that users can activate the simulator. The agent-based modeling, which implemented fuzziness has been created in this study, thereby providing an idea for the users to apply the NetLogo model [16].

Moreover, agent-based modeling (ABM) [17], which is also referred to as individual-based, modeling, involves a novel paradigm of computational modeling. Thus, it is described as a system for analysis, representing 'agents' that involve and simulate the interactions of agents [18]. The attributes, as well as behaviors, are grouped by these interactions to become a scale. Programmers can, therefore, design ABM in Netlogo using the button, with the input, and the output, in addition to the slides, along with other functions, which can make ABM easily comprehended and used. ABM has also revealed the system appearance starting from low outcomes to high outcomes. Moreover, improvements can be made by disabling traditional limitations of modeling like permitting agent learning, as well as adaption and limited knowledge, along with information access because the paradigm of the agent-based modeling has been extensively utilized in dynamics, as well as intricate communities, e.g., telecommunications, health care, and other services, which involve large populations that use explicitly capture social networks.

Generally, when an ABM is built for simulating a specific phenomenon, the actors need to be identified first (i.e., the agents). After that, the processes (i.e., the rules), which govern the interactions between the agents, need to be considered.

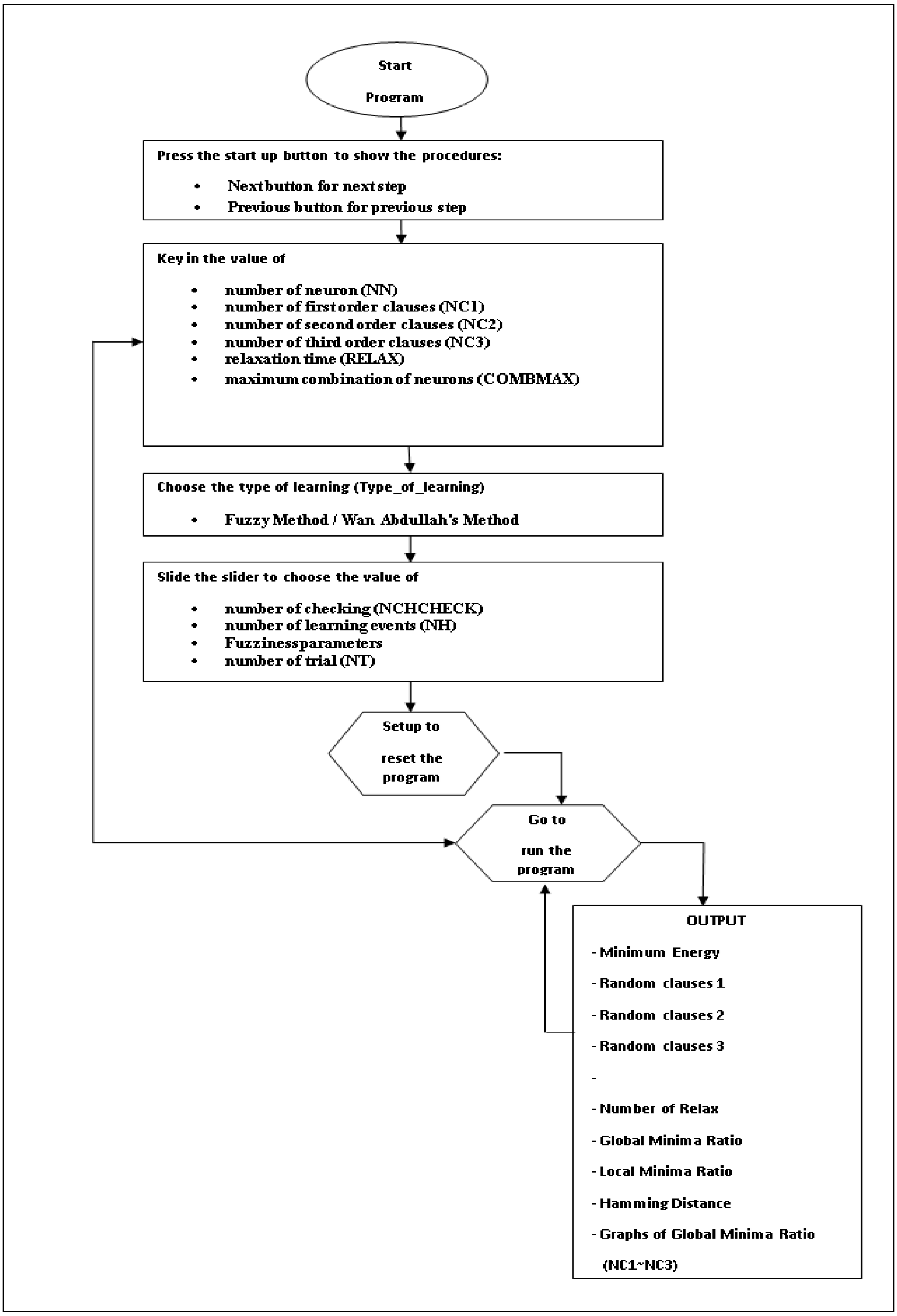
Start
Program
Press the start up button to show the procedures:
Next button for next step
Previous button for previous step
Key in the value of
number of neuron (NN)
number of first order clauses (NC1)
number of second order clauses (NC2)
number of third order clauses (NC3)
relaxation time (RELAX)
maximum combination of neurons (COMBMAX)
Choose the type of learning (Type_of_learning)
Fuzzy Method / Wan Abdullah's Method
Slide the slider to choose the value of
number of checking (NCHCHECK)
number of learning events (NH)
Fuzziness parameters
number of trial (NT)
Setup to reset the program
Go to run the program
OUTPUT
- Minimum Energy
- Random clauses 1
- Random clauses 2
- Random clauses 3
-
- Number of Relax
- Global Minima Ratio
- Local Minima Ratio
- Hamming Distance
- Graphs of Global Minima Ratio (NC1~NC3)

Fig.2. Flow Chart for integrating fuzziness logic programming in the Hopfield neural network

Description of the flow chart [16]:

PHASE 1: Enter Values

1) Press start-up / Reset Quick-Start button for a novel user.
2) The user either presses the subsequent button so that he/she advances to the subsequent step or presses the preceding button so that the user retreats to the earlier step.
3) Subsequently, the user can key in NN, NC1, NC2, RELAX, and COMBMAX.

4) There is quite a small difference in higher orders as the NN maximum is 30. NC1, NC2, NC3 relate to the value of NN, which has been keyed in. The maximum RELAX and COMBMAX numbers are 100. The try and error technique has been used to select the entire values.
5) Select the learning type that is the Hebb's rule or Wan Abdullah's approach.
6) Next, the user slides the slider so that NCHCHECK, NH, TOL, as well as NT are selected.
7) Maximum NCHCHECK, NH, as well as NT is 100, while the TOL value is selected as 0.5 after applying the technique of trial and error.
8) After setting the entire values, press the setup button for fixing and setting the program values.
9) After that, the user presses the 'go' button to activate the program.
10) It will produce random program clauses, e.g., when the user declares NC1 as 3, the 3 first-order clauses are created.

PHASE 2: Training

1) The neurons' initial states are set in clauses.
2) In accordance with the Hopfield network's idea, which was originated from the behaviour of neurons (i.e., the particles) in a certain magnetic field, the particles are 'communicating' with each other and in a completely linked form with each particle striving to accomplish the energetically promising state, in other words, with a minimum of the energy function, called activation. All neurons in this state rotate to encourage each other to go on with rotation, allowing the network to evolve till minimum energy is reached. Consequently, the minimum energy corresponds to the wanted energy so that the global minima value is reached. Other programs that fuzzification integrated into Wan Abdullah's method, are explained in section 4.
3) Testing of the final state (i.e., the accomplished state after the neurons' relaxation). The final state is identified by the system as an unchanging state when it remains stable for 5 runs.
4) The stable state's corresponding final energy can be calculated.
5) If there is a difference amongst the final energy, as well as the global minimum energy within the value of tolerance, the obtained solution will be global, or the user should return to step 1.
6) The user computes the global solution, along with the global solutions' ratio (i.e., the ratio= Number of global solutions/ Number of iterations). Relaxation is run for 1000 trials; it is also run for 100 neurons' combinations.
7) To conclude, the output of each run is printed out by the system.

6. Simulations and Discussion

The cluster center is set, as well as the parameter of fuzzification $v_i = 4$ and $m = 2$. The technique of trial/error will generate all the values. No theoretical basis can be found to locate the most optimum choice of the parameter of fuzzification m . According to Bezdek [15], the algorithm is implemented for one of these values $m > 1$, which is selected as it can produce the ideal network performance. The pattern of training z_x is set to an initial state, including 1 or -1. The influence of μ_{xi} is, thus, reliant on the value of m as the greater the m , the higher fuzziness value can be obtained.

The program has been operated for 100 trials, as well as 100 neurons' combinations. The clauses number (NC) for every number of the neurons has been identified by the ratio between 0.1 and 1.0. A comparison is held for the global minima ratio (i.e., global solutions' number/the runs' number), in addition to the running time (that is, the needed time for the neurons' relaxation toward final states) between direct method, as well as the FCM algorithm for logic programming to be performed in Hopfield network.

A. Hamming Distance

Hamming distance has been originally perceived for detecting and correcting errors [9, 19]. Simply, it is a differing bits' number amid the two-bit vectors. Describing closeness of a certain bit pattern can be done easily to a differing bit pattern by determining the bit positions' number, where both compared patterns differ. Therefore, the hamming distance has been measured between both states: the final state, as well as the global state.

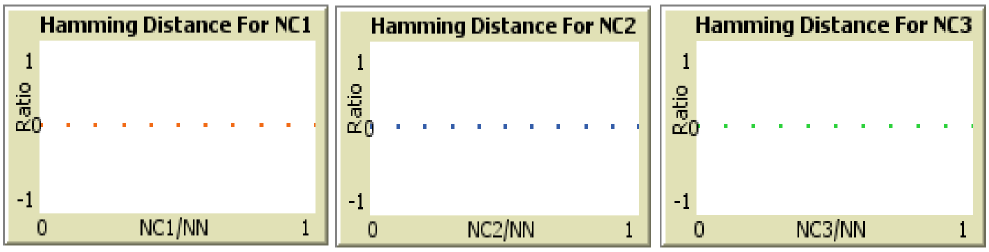

Fig.3. Hamming distances for NC1, NC2, and NC3

Fig. 3 displays the graphs of the hamming distance of each level of the clauses like NC1, NC2, and NC3. The Fig. 3 demonstrates that the hamming distance is almost close to zero, including all the cases if the neurons number (NN), with the number of the literals per clause (NC1, NC2, and NC3) amplified after applying the trial/error method. The hamming distance value is almost close to zero because neurons have reached a stable state. Therefore, the distance is approximately zero amid the given stable state, as well as the given global state. An identical statistical error is, therefore, obtained in the entire cases.

B. Global Minima

Based on the global minima graphs, when it is close to 1, it is assumed to approach the best solution [20].

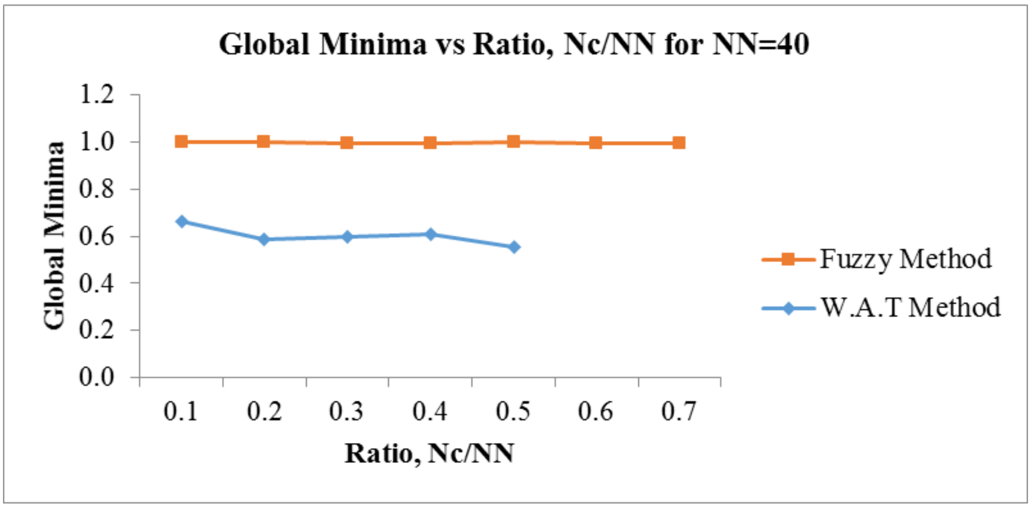

Fig.4. Global Minima for W.A.T method and fuzzy method, NN=40

Based on the conducted analysis, the achieved global minima ratio (i.e., the global minima values' number/the number of the runs) of the FCM has been better compared to the direct method based on the figures above. The neurons have not yielded or trapped in any of the sub-optimal solutions. It can be observed that as the network's complexity increased (i.e., the number of neurons grew larger), FCM performed more efficiently. This can be attributed to the modification and restriction of the cluster centers, as well as the energy function, thereby controlling the energy relaxation of every neuron. This, in turn, has enhanced the competence of controlling the energy relaxation process although the network has become larger. Therefore, the FCM method's global minima ratio has outperformed compared to the given direct method. When the network can be complex, the direct method's processing time has become bigger compared with FCM. The solutions also appear to oscillate; they are more easily stuck more with the given direct method.

C. Processing Time

Processing time is defined as a time acquired by a model to complete the training phase and retrieval phase[21].

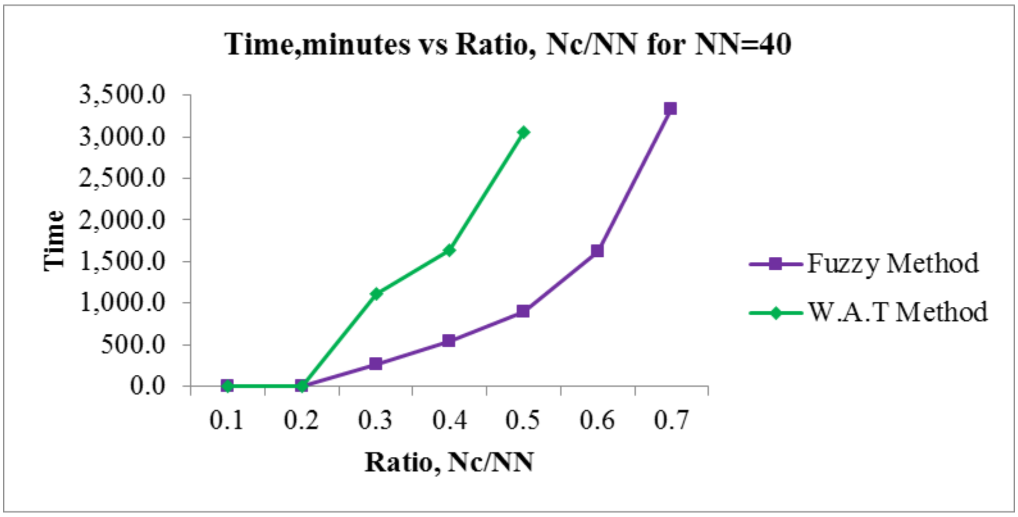

Fig.5. Processing Time for W.A.T method and fuzzy method, NN=40

As shown in Fig. 5, the technique of the fuzzy Hopfield neural network clustering can still maintain its consistency between 1.0 and 0.7 despite the increased network's complexity. However, the values of global minimum solutions for the W.A.T method decreased up to 0.4. This proves that the technique of the fuzzy Hopfield neural network clustering is better than the W.A.T method. From Fig. 5, the processing time for the fuzzy Hopfield neural network clustering technique is better than the W.A.T method. Therefore, by utilizing the FCM method for enhancing the energy function, performing logic programming in the neural network can be enhanced.

7. Conclusion

Agent-based modeling has been developed in this paper to perform logic programming in the Hopfield network by utilizing the NETLOGO platform. Based on the findings of the study, the FCM's capability of performing logic programming on HNN has been more efficient compared to the method proposed by Wan Abdullah (only McCulloch pits updating rule applied to HNN). A better result has been provided regarding the global minima ratio and Hamming distance, as well as the CPU time. The results showed that FCMHNN enhanced the performing competence of logic programming if the neurons' number is larger. Also, FCMHNN possesses a higher memory capacity; it did not undergo a substantial capacity loss if the input clauses are larger. Based on the experimental results, the introduced FCMHNN produced more promising results compared to HNN. Moreover, the designed FCMHNN represents a self-organized structure. It is enormously interconnected, used in an equivalent method, and easily designed for hardware gadgets to accomplish the high-speed application. However, finding the objective function's global minimum by utilizing the Hopfield neural network remains a problem. Therefore, improving the FCMHNN training by using other algorithms such as ABC and AIS is recommended for further studies.

Acknowledgment

This research is supported by Research University Grant (RUI) (1001/PMATHS/8011131) by Universiti Sains Malaysia.

References

[1] A. S. A. Garcez, & G. Zaverucha, (1999). The connectionist inductive learning and logic programming system. *Applied Intelligence*, *11*(1), 59-77.

[2] S. Sathasivam, S. A. Alzaeemi, & M. Velavan, (2020). Mean-Field Theory in Hopfield Neural Network for Doing 2 Satisfiability Logic Programming. *International Journal of Modern Education & Computer Science*, 12(4).

[3] J. J. Hopfield, & D. W. Tank, (1985). "Neural" computation of decisions in optimization problems. *Biological cybernetics*, *52*(3), 141-152

[4] S. Sathasivam, M. Mansor, M. S. M. Kasihmuddin, & H. Abubakar, (2020). Election Algorithm for Random k Satisfiability in the Hopfield Neural Network. *Processes*, *8*(5), 568.

[5] H. Abubakar, & S. Sathasivam, (2020, October). Developing random satisfiability logic programming in Hopfield neural network. In AIP Conference Proceedings (Vol. 2266, No. 1, p. 040001). AIP Publishing LLC.

[6] W. A. T. W. Abdullah, (1992). Logic programming on a neural network. *International journal of intelligent systems*, *7*(6), 513-519.

[7] S. Sathasivam, (2010). Upgrading logic programming in Hopfield network. *Sains Malaysiana*, *39*(1), 115-118.

[8] S. Sathasivam, (2012). Applying Fuzzy Logic in Neuro Symbolic Integration. World Applied Sciences Journal 17 (Special Issue of Applied Math), pp. 79-86.

[9] J. Wang, X. Liu, J. Bai, & Y. Chen, (2019). A new stability condition for uncertain fuzzy Hopfield neural networks with time-varying delays. *International Journal of Control, Automation and Systems*, 17(5), 1322-1329.

[10] G. Bologna, (2004). Is it worth generating rules from neural network ensembles?. *Journal of Applied Logic*, *2*(3), 325-348.

[11] H. Huang, D. W. Ho, & J. Lam, (2005). Stochastic stability analysis of fuzzy Hopfield neural networks with time-varying delays. *IEEE Transactions on Circuits and Systems* II: Express Briefs, 52(5), 251-255.

[12] J. S. Lin, (1999). Fuzzy clustering using a compensated fuzzy Hopfield network. *Neural Processing Letters*, *10*(1), 35-48.

[13] L. A. Zadeh, & R. A. Live, (2019). Fuzzy Logic Theory and Applications. https://doi.org/10.1142/10936.

[14] K. Singh, (2016). Fuzzy logic based modified adaptive modulation implementation for performance enhancement in ofdm systems. *International Journal of Intelligent Systems and Applications(IJISA)*, *8*(5), 49.

[15] J. C. Bezdek, (2013). *Pattern recognition with fuzzy objective function algorithms*. Springer Science & Business Media.

[16] S. A. Alzaeemi, & S.Sathasivam, (2018). Hopfield neural network in agent based modeling. *MOJ App Bio Biomech*, 2(6), 334-341.

[17] S. A. Alzaeemi, S. Sathasivam, M. Velavan, M. Mamat. Agent-based Modeling for Activation Function in Enhancement Logic Programming in Hopfield Neural Network. *International Journal of Engineering and Advanced Technology (IJEAT),* 9 (4), pp. 1872-1879.

[18] E. Bonabeau, (2002). Adaptive agents, intelligence, and emergent human organization: capturing complexity through agent-based modeling: methods and techniques for simulating human systems. *Proceedings of the National Academy of Sciences USA*, *99*, 7280-7287.

[19] M. S. M. Kasihmuddin, M. A. Mansor, & S. Sathasivam, (2016). Bezier Curves Satisfiability Model in Enhanced Hopfield Network. *International Journal of Intelligent Systems & Applications (IJISA)*, *8*(12).

[20] M. A. Mansor, & S. Sathasivam, (2016). Accelerating activation function for 3-satisfiability logic programming. *International Journal of Intelligent Systems and Applications (IJISA)*, *8*(10), 44.

[21] F. Boufera, F. Debbat, N. Monmarché, M. Slimane, & M. F. Khelfi, (2018). Fuzzy inference system optimization by evolutionary approach for mobile robot navigation. *International Journal of Intelligent Systems and Applications (IJISA)*, *10*(2), 85.

Authors’ Profiles

Shehab Abdulhabib Alzaeemi received his Bachelor's Degree in Mathematics Science from Taiz University in 2003-2004 and his Master of Applied Mathematics Science from Universiti Sains Malaysia in 2016-2017 and now I continue in Ph.D. research in Universiti Sains Malaysia. He is fellow under the Academic Staff System in Sana'a Community College from 2005-2014. His research interests mainly focus on data mining, logic programming and neural network.

Saratha Sathasivam is an Associate Professor in the School of Mathematical Sciences, Universiti Sains Malaysia. She received her MSc and BSc(Ed) from Universiti Sains Malaysia. She received her Ph.D at Universiti Malaya, Malaysia. Her current research interest are neural networks, agent based modeling and constrained optimization problem.

Muraly Velavan is a senior lecturer and also the Deputy Director in the School of General & Foundation Studies, AIMST University. He received his MSc at Universiti Malaysia Perlis. His current research interests are neural networks and agent based modeling.

How to cite this paper: Shehab Abdulhabib Saeed Alzaeemi, Saratha Sathasivam, Muraly Velavan, " Agent-based Modeling in doing Logic Programming in Fuzzy Hopfield Neural Network", International Journal of Modern Education and Computer Science(IJMECS), Vol.13, No.2, pp. 23-32, 2021.DOI: 10.5815/ijmecs.2021.02.03