MedCrave
Step into the World of Research

MOJ Applied Bionics and Biomechanics

Review Article

Open Access

CrossMark

Hopfield neural network in agent based modeling

Abstract

In recent years, most of the researchers in the field of artificial intelligence focused on developing the Agent Based Modelling in order to simplified their learning algorithms. This version of simplification is useful to find a provable criterion for convergence in a dynamic system. In this paper, an agent based modelling (ABM) was developed by using NETLOGO as a platform for activation functions to carry out logic programming in Hopfield network. The developed model seems to illustrate the task of doing logic programming in a simple, flexible and user friendly manner.

Keywords: hopfield network, netlogo, agent based modelling, logic program

Volume 2 Issue 6 - 2018

Shehab Abdulhabib Saeed Alzaeemi, Saratha Sathasivam
School of Mathematical Sciences, Universiti Sains Malaysia, Malaysia

Correspondence: Shehab Abdulhabib Saeed Alzaeemi, School of Mathematical Sciences, Universiti Sains Malaysia, 11800 USM, Penang Malaysia, Email shehab_alzaemi@yahoo.com

Received: January 30, 2018 | **Published:** November 23, 2018

Introduction

An Artificial Neural Network (ANN), otherwise known as connectionist system, neural net, parallel distributed processing models, neuromorphic systems to mention but few, is an information processing system which mimics the biological nervous system especially the brain. Neural network can be seen as a step towards understanding of intelligence. There are many neural network architectures which provide different performance for different applications. There are many neural network architectures which provide different performance for different applications. Common neural network architectures include Radial Basis network, Single layer network, Multilayer network, Competitive network and Hopfield network. Hopfield network is a recurrent neural network invented by John Hopfield in 1982 that consist of a set of N interconnected neurons which all neurons are connected to each others in both directions. It consist of synaptic connection pattern that are the building blocks of Lyapunov function, E (energy function) for dynamic activities.[1,2] The knowledge presented in Hopfield network must be well founded in order to optimize Hopfield's architecture. With that in mind, logic program will be embedded to Hopfield network. Logic program provides a natural way for problem-solving.[3] Logic programming can be demarcated as an optimization problem,[4,5] that is similar to other optimization problem such as travelling salesman problem. Compared to neural network which is a black box model, logic program is easier to understand, easier to verify and also easier to change.[6] The assimilation between both paradigm (Logic programming and Hopfield network) was presented by Wan Abdullah and revolve around propositional Horn clauses.[7,8] Gadi Pinkas and Wan Abdullah,[7,9] proposed a bi-directional mapping between logic and energy function in a symmetric neural network. The aim of the mapping is to find the optimal assignments that corresponds to global minimum energy of the system. This optimal assignment will be the model of the corresponding logic program. The first agent based modelling to carry out logic programming in Hopfield network by using NETLOGO as platform was proposed by Sathasivam.[10]

In this paper, we develop agent based modeling (ABM) for activation functions to do logic programming in Hopfield network. Agent-based modelling (ABM) alternatively called individual-based modelling or multi-agent based modelling. It is a powerful simulation modelling technique that have gained recognition in a number of applications in the last few years, including applications to real-world business problems. In agent-based modelling (ABM), a system is modelled as a collection of autonomous decision-making agents. Agent-based model (ABM) is kind of microscale model that simulate the simultaneous operations and interactions of multiple agents in an attempt to re-create and predict the appearance of complex phenomena. The process is one of emergence from the lower (micro) level of systems to a higher (macro) level. As such, a key notion is that simple behavioral rules generate complex behavior. Another central tenet is that the whole is greater than the sum of the parts. Individual agents are typically characterized as boundedly rational, presumed to be acting in what they perceive as their own interests, such as reproduction, economic benefit, or social status, using heuristics or simple decision-making rules.[11] ABM agents may experience "learning", adaptation, and reproduction. As these non-linear, adaptive interactions are mostly too complex to be captures by analytical expression, computer simulations are most often use, idea of such simulation is to specify the rules of behaviour of individual entitles, as well as the rules of their interaction in a multitude of the individual entitles using a computer model and to explore the consequences of the specified individual- level rules on the level of population as a whole, using results of simulation of their behaviour and interaction are known as agent-based stimulation, the properties of individual agents describing their behaviour and interaction are known as elementary properties, and the properties emerging on the higher, collective level are known as emerging properties. Agent- based modelling is appealing and interesting because of the collective level are often neither obvious nor expectable, even in many cases when the assumption on individual agent properties are very simple, the capability of generation complex and intriguing emergent properties arises not so much that the in-built rules of individual agent behaviour, as from the complexity of the network of interaction among the agents. Precisely this multitude of agents, as well as the multitude and complexity of their interaction are the main reasons why in most cases formal mathematical deduction of results of an agent based model is not possible ABMs are also called individual-based models (IBMs), and individuals within IBMs may be simpler than fully autonomous agents within ABMs[12] A review of recent literature on individual-based models, agent-based models, and multiagent systems shows

that ABMs are also useful in non-computing related scientific domains including biology, ecology and social science. Agent-based modelling is related to, but distinct from, the concept of multi-agent systems or multi-agent simulation. ABM is also use to search for explanatory insight into the collective behaviour of agents obeying simple rules, typically in natural systems, rather than in designing agents or solving specific practical or engineering problems. It combines elements of game theory, complex systems, emergence, computational sociology, multi-agent systems, and evolutionary programming. Monte Carlo methods are used to introduce randomness.

The rest of the paper is organized as follows. Section II Contains Hopfield Neural Network. In Section III, We briefly discussed Logic Programming in Hopfield Network. IV. We talk about Activation Function Section V we discuss the significance and the usage of NETLOGO. Section VI describes Agent Based Modelling (ABM) in Hopfield network. Section VII discuss the implementation of Agent Based Modelling in logic programming. Section VIII discussion and section IX to conclude the study.

Hopfield neural network

Hopfield neural network is easier to be integrated with any paradigms to solve satisfiability problem,[4,13,14] A Hopfield network is a network of N interconnected artificial neurons, which are fully interconnected.[1,15] The connection weight from neuron j to neuron i is represented and denoted by the number W_{ij}, in general, the number W_{ij} is symmetric, that is $W_{ij} = W_{ji}$ and in Hopfield network $W_{ij} = W_{ji} = 0$ (no self connection), the set of all such numbers is represented by the connection weight matrix W, whose elements are . The local field of the connection is given by

$$S_i(t+1) = \mathrm{sgn}\left[h_i(t)\right] \tag{1}$$

Where … denotes the higher order connection. The updating rule reads

$$S_i(t+1) = \mathrm{sgn}\left[h_i(t)\right] \tag{2}$$

In order to check the correctness of the final state, total energy of the neurons will be evaluated. An energy function for the Hopfield network is given by [6]:

$$E = -\frac{1}{2}\sum_i \sum_j W_{ij}^{(2)} S_i S_j - \sum_i W_i^{(1)} S_j \tag{3}$$

Where … denoted the higher order connection of the network.

Hopfield network has N interconnected neurons that update their activation values asynchronously and independently of other neurons. Each neuron has both input and output which make it different from the classical feed-forward neural network, where the feed-forward network consists of input layers, hidden layers and output layers.[2] Each layer consists of certain number of neurons. Hopfield network is being utilized in various area such as image processing, signal processing, data de-convolution, pattern matching, solving equations and optimizing functions, travelling salesman, scheduling and resource allocation to mention nut few.

The Hopfield network has demonstrated the interesting features.[4,5,16,17]

Distributed representation: A memory is stored as pattern of activation across a set of processing elements, memories can be superimposed upon one another, and different memories can be represented by different patterns over the same set of processing elements.

Distributed asynchronous control: Each processing element makes decisions based only on its own local situation. All of these local actions add up to a global solution.

Content addressable memory: The network can store a number of patterns. It can retrieve a pattern if only a portion of the pattern is specified. The network will automatically find the closest match.

Fault tolerance: The network can still function properly, even if few of the processing elements misbehave or fail completely.

Logic program hopfield neural network

A logic program consists of program clauses and is activated by an initial goal statement. Logic program provides a natural way for problem-solving.[3] The fact that logic programming has gained ground as a vital application in computer science and artificial Intelligence (AI) is undisputable, it provides helpful programming languages for computer to express facts in which intelligent system can comprehend. Logic programming is easy to understand, easier to verify and change compare to other programming where deeper knowledge of understanding matters a lot. Thus, an inexperienced users and database users might find it very easy to comprehend. Moreover, logic programming is a high level language, easy to create prototype, and offer shorter and more readable programs that suits many AI applications. Logic programming is a programming paradigm based on formal logic. A program written in a logic programming language is a set of sentences in logical form, expressing facts and rules about some problem domain.[18] It is made up a set of axioms and a goal statement. The program allows user to state a collection of axioms from which a theorem or goal can be proven.[19] When the goal is stated by the user, the language implementation will search a collection of axioms and inference steps that together imply the goal. In almost all logic languages, axioms are written in a standard form known as Horn clause. A Horn clause consists of a head and a body:

$$H \leftarrow B_1, B_2, B_3, \ldots B_n \tag{4}$$

Where H is the head and B_i are the body. Horn clause is a clause of disjunction of literals that have at most one positive literal. The below expression is a typical example of Horn clause:

$$\neg A \vee \neg B \vee \ldots \vee \neg E \vee H \tag{5}$$

Where the positive literal in the expression is H, whereas the others literals are negative since the symbol (¬) is negation of the literal that follows it. A Horn clause that consist of exactly one positive literal is called a definite Horn clause.

Equation (5) above can be rewritten equivalently in a simplified form as shown below.

$$(A \wedge B \wedge \ldots \wedge E) \rightarrow H \tag{6}$$

The lexicon (non-logical symbol) of propositional logic consists of a set of proposition symbols. In equation (5) above, the proposition symbols are A, B, ..., E and H. The logical symbols of propositional logic are: ¬, ↔, and →. Following are some terminologies of propositional logic in logic programming.

A sentence of the form $A \vee B$ is a disjunction.

Citation: Alzaeemi SAS, Sathasivam S. Hopfield neural network in agent based modeling. *MOJ App Bio Biomech.* 2018;2(6):334–341. DOI: 10.15406/mojabb.2018.02.00089

A sentence of the form $A \wedge B$ is a conjunction

A statement of the form $A \rightarrow B$ is an implication, where B is refers to as antecedent and B is termed consequent.

A statement of the form¬A means negation of A (Not A) this will change the state of A from whatever state it is earlier

Based on Wan Abdullah's method, the following algorithm summarizes how a logic program can be done in a Hopfield network proposal by Wan Abdullah (Figure 1).[20,21]

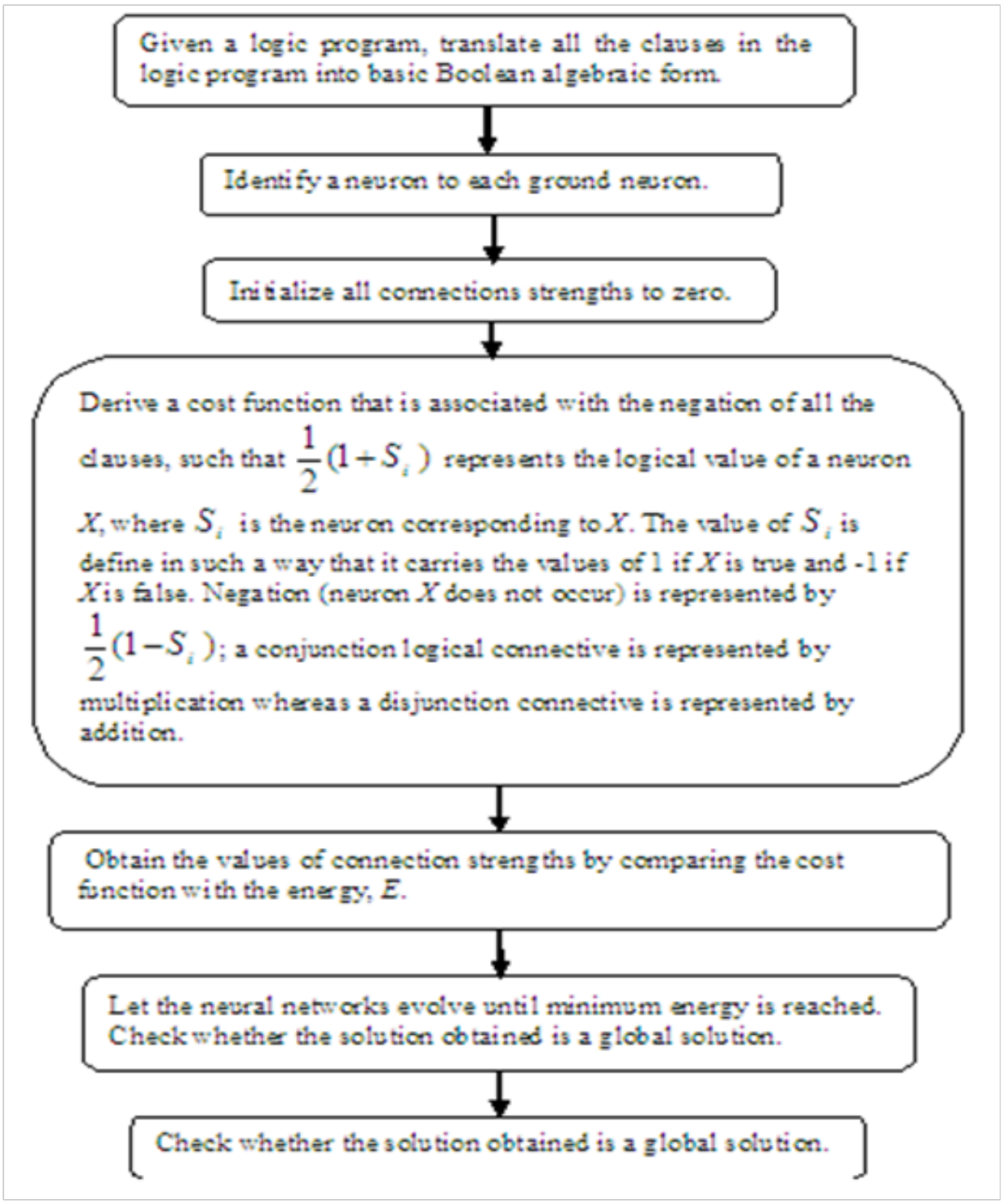

Figure 1 Flow chart on algorithm how a logic program can be done in a Hopfield network.

Citation: Alzaeemi SAS, Sathasivam S. Hopfield neural network in agent based modeling. *MOJ App Bio Biomech.* 2018;2(6):334–341. DOI: 10.15406/mojabb.2018.02.00089

Activation function

The first activation function implemented in logic programming in Hopfield neural network was the sign function by McCulloch-Pitts (ideal model) proposed by Walter and Pitts.[22] Although McCulloch-Pitts Activation Function helps the network to find global solution, this function is prone to few weaknesses such as computational burdening and lack of efficiency in producing desired results. McCulloch-Pitts neuron has been generalized in many ways and one of the obvious generalizations is to use activation functions other than threshold function. Figure 2 below shows different types of activation functions:

In computational networks, the activation function defines the output of the neuron by the given input. The activation function for the original McCulloch-Pitts neuron was the unit step function. However, the artificial neuron model has since been expanded to include other functions such as the sigmoid, piecewise linear and transfer function. The activation function is sometimes called a "transfer". The most commonly used transfer functions are logistic sigmoid function and hyperbolic tangent function. According to Kalveram,[23] the logistic sigmoid function was being more frequently used than the hyperbolic tangent function in Hopfield neural network. Numerous researchers proved Hyperbolic activation function outperform most of the activation function.[24,25]

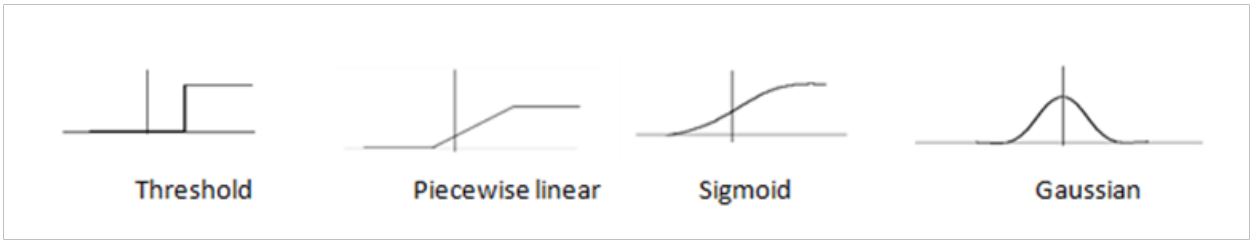

Figure 2 Various types of activation function.

Netlogo

To assist in the development of agent-based models (ABM), a number of different platforms have been developed.[26] These platforms vary in how much support they provide. Some factors in the program are represented by Swarm and Mason. ABM offers a set of software libraries to be used in programming a model. Repast, in its Symphony (sic) version, Netlogo offers a few more tools for quick construction of agent-based models. Some other agent-based modelling platforms provide fixed sets of rules that can be used with some chosen parameters, but these are often too restricted to capture the wide range of phenomena that one might want to model. NetLogo consists of a programming language (derived from the earlier Logo language) and a set of libraries, as well as a programming environment. Like Repast and Swarm it provides a set of programming facilities, however NetLogo also provides a graphical tool for quickly constructing interfaces for running agent-based models. One of the benefits of using NetLogo is its interface. More information on the interface can be found in the NetLogo Interface Guide.[19,27] More information on linking these interface elements with the program can be found in the following subsection. To engage people who are experienced in the real-world system to participate in your model and provide you with feedback,[27] NetLogo supports participatory agent-based modelling through its HubNet facility. More information can be found in the NetLogo HubNet Guide,[19] and HubNet Authoring Guide.[27] NetLogo enables users to open simulations and "play" with them, exploring their behavior under various conditions. NetLogo is also an authoring environment that is simple enough to enable students and researchers to create their own models, even if they are not professional programmers.[28] own models, even if they are not professional programmers. We designed NetLogo for both education and research. There has been considerable research on the use of multi-agent modeling in K-12 settings e.g.[29–32] In this paper, however, we focus on NetLogo as a powerful research tool that is also suitable for learners at the undergraduate level and above. NetLogo has a simple interface, models can be set up and run with only the push of one or two buttons; to produce the model such as the one illustrated takes just 50 lines of simple code. NetLogo is good for setting up simple simulations very quickly.

Agent based modelling

After knowing the efficiency of doing logic programming in Hopfield network, we will used NETLOGO as a platform to develop agent based modelling (ABM). We will design an agent based modelling to implement the Hopfield network in doing logic programming. Netlogo was designed and authored by Uri Wilensky, director of Northwestern University's Center for Connected Learning and Computer-Based Modelling.[33] Agent-based modeling (ABM) is a technique increasingly used in a broad range of social sciences. It involves building a computational model consisting of "agents," each of which represents an actor in the social world, and an environment.[34] Netlogo is an agent based programming language and integrated modeling environment. Netlogo was designed, in the spirit of the Logo programming language, to be low threshold and no ceiling. It teaches programming concepts using agents in the form of turtles, patches, links and the observer. Netlogo was designed for multiple audiences in mind, in particular: teaching children in the education community and domain experts without a programming background to model related phenomena. Many scientific articles have been published using Netlogo. While for the link agents, they connect the mobile agents to make networks, graphs and aggregates which can let the users get more understanding on output of system. Moreover, its runs are exactly reproducible cross-platform. The model can be viewed in either 2D or 3D form. Programmer can choose any interesting agent shapes to design the agent based modelling, and some interface builders like buttons, sliders, switches, choosers, monitors, notes, output area in the agent based modelling can be developed too. Those interface builders are ready to use and programmer do not need to write more programming language from them. Later in the next section will carry out the definition and more explanation of simulator and benefits of agent based modelling in NETLOGO.

Citation: Alzaeemi SAS, Sathasivam S. Hopfield neural network in agent based modeling. *MOJ App Bio Biomech.* 2018;2(6):334–341.
DOI: 10.15406/mojabb.2018.02.00089

Implementation of agent based modelling in logic program

Sathasivam[35] developed the first reverse analysis in Hopfield network. With that in mind, Agent Based Modelling (ABM) doing logic programming in Hopfield network will be presented (Figure 3). Netlogo will be used as a platform to develop agent based modelling (ABM). Agent based modelling designed to implement Hopfield network in doing logic programming. Netlogo is a multi-agent programming language and integrated modelling environment. There has been continuous development for connected learning and computer-based modelling. Furthermore, it is a well suited method for modelling complex systems that can give instructions to hundreds or thousands of agents to operate independently. It is because it is fully programmable and formed by simple language structure. It can instruct mobile agents move over a grid of stationary agents. While for the link agents, they connect the mobile agents to make networks, graphs and aggregates which can let the users get more understanding on output of system. Moreover, its runs are exactly reproducible cross-platform. The model can be viewed in either 2D or 3D form.

Programmer can choose any interesting agents shapes to design the agent based modelling, and some interface builders like buttons, sliders, switches, choosers, monitors, notes, output area in the agent based modelling can be developed too. Those interface builders are ready to use and programmer do not need to write more programming language from them. A simulator of Hopfield networks that use a conventional computer had created but not for building up a new network design every time or to store a new set of memories. Hence, Netlogo saves lots of energies and times for the programmer to rebuild new system from time to time. Thus, a computer program which emulates exactly what the user want needs to construct in order to simulate the action of Hopfield Network. It is easier for the programmer to modify the program and store a new set of data. Thus, an agent based modelling had designed for the user to run the simulator. Moreover, agent based Modelling which also called individual-based modelling is a new computational modelling paradigm which is an analyzing systems that representing the agents that involving and simulating of their interactions. Their attributes and behaviours will be group together through their interactions to become a scale. Programmer can design ABM in Netlogo by using button, input, output, slides and other functions that make ABM easy to understand and use. In addition, ABM reveals the appearance of the systems from low to high level outcomes and it make improvement by surpassing the traditional modelling limitations such as allowing agent learning and adaption, limited knowledge and access to information. This is because, the agent based modelling paradigm are commonly used in dynamics and complex communities such as telecommunication, health care and others that involving large populations which used explicitly capture social networks. In this section the steps involved in developing ABM for the logic programming method will be considered. Firstly, look into the steps involved in developing for activation functions to do logic programming in Hopfield Network as Figure 4 below:

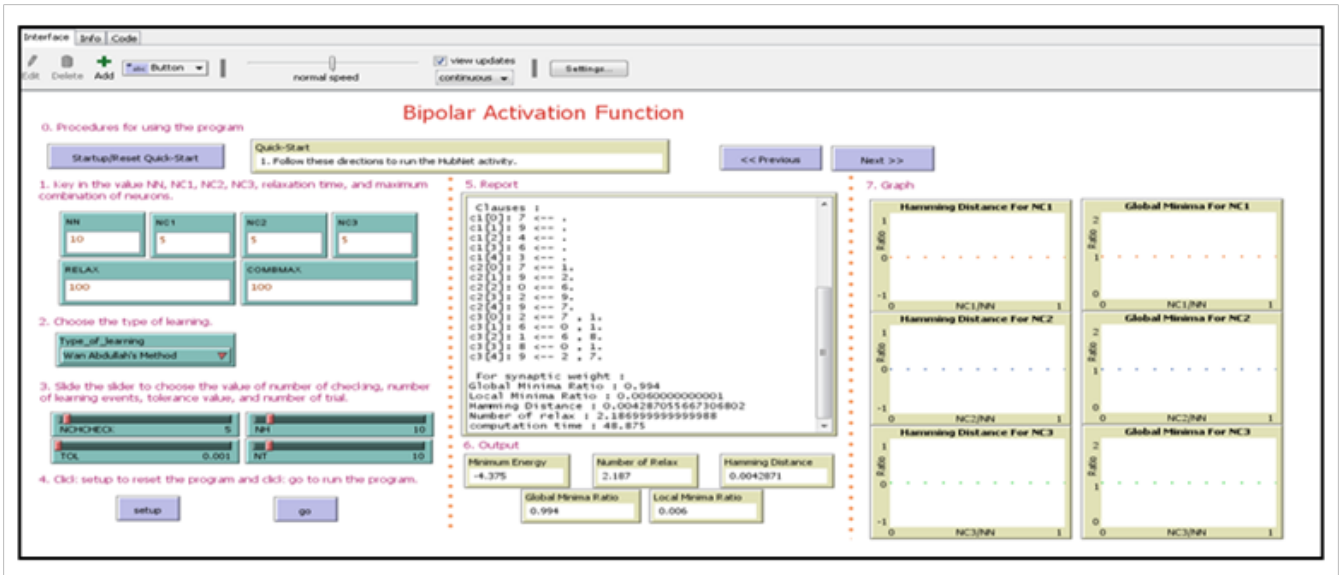

Figure 3 Layout ofAgent Based Modelling using Netlogo.

Explanation for the Figure 4:-

Phase 1: Entering Values

A. Choose the type of activation function which is either Hyperbolic tangent activation function, Unipolar activation function, Bipolar activation function, McCulloch-Pitts function, Gaussian Activation Function, others……

B. Press the start-up / Reset Quick-Start button for the new user.

C. User can press next button to go for the next step and previous button to go for the previous step.

D. Later, key in NN, NC1, NC2, NC3, RELAX and COMBMAX.

E. The maximum of Number of neurons (NN) is 80.

Number of first order clauses (NC_1), Number of second order clauses (NC_2), Number of third clauses (NC_3) is related to the value of the NN that had entered. The maximum number of relaxation time (RELAX), and maximum combination for neurons (COMBMAX) is 100. All these values are chosen by try and error technique.

a) Choose type of learning which is either Wan Abdullah's Method or Hebb's Rule.

Citation: Alzaeemi SAS, Sathasivam S. Hopfield neural network in agent based modeling. *MOJ App Bio Biomech.* 2018;2(6):334–341. DOI: 10.15406/mojabb.2018.02.00089

b) Then, slide the slider to choose number of checking (NCHCHECK), number of learning event (NH), tolerance value (TOL) and number of trial (NT).

c) The maximum of NCHCHECK, NH and NT are 100 while the maximum of TOL is 0.001.

d) After all the value had been set, press the setup button to fix and set those values in the program.

e) Then, press go button to run the program.

f) The program will generate random program clauses. Example, if user declared NC1 as 3, then 3 first order clauses will be generated.

Phase 2: Training

A. Initialize initial states for the neurons in the clauses

B. Next, based on the idea for the Hopfield network that originated from the behaviour of neurons (particles) in a magnetic field, every particles will 'communicates' to each other to a completely linked forms with each of them are trying to reach an energetically favourable state that means a minimum of the energy function. This state is known as activation. Thus, all neurons in this state will rotate and thereby encourage each other to continue the rotation. So, let the network evolves until minimum energy is reached. The minimum energy corresponds to the energy needed to reach global minima values.

C. Test the final state (state obtained after the neurons relaxed). The system will determine the final state as a stable state if the state remains unchanged for more than five runs.

D. Following this, calculate corresponding final energy for the stable state.

E. If the different between the final energy and the global minimum energy is within tolerance value, then consider the solution as global solution or else go to step 1.

Finally, calculate global solution and also calculate ratio of global minima ratio

$$Global\ minima\ ratio = \frac{Number\ of\ global\ solutions}{Number\ of\ iterations} \quad (7)$$

The relaxation will run for 100 trials and 100 combinations of neurons.

Lastly, the system will print out the output for each run.

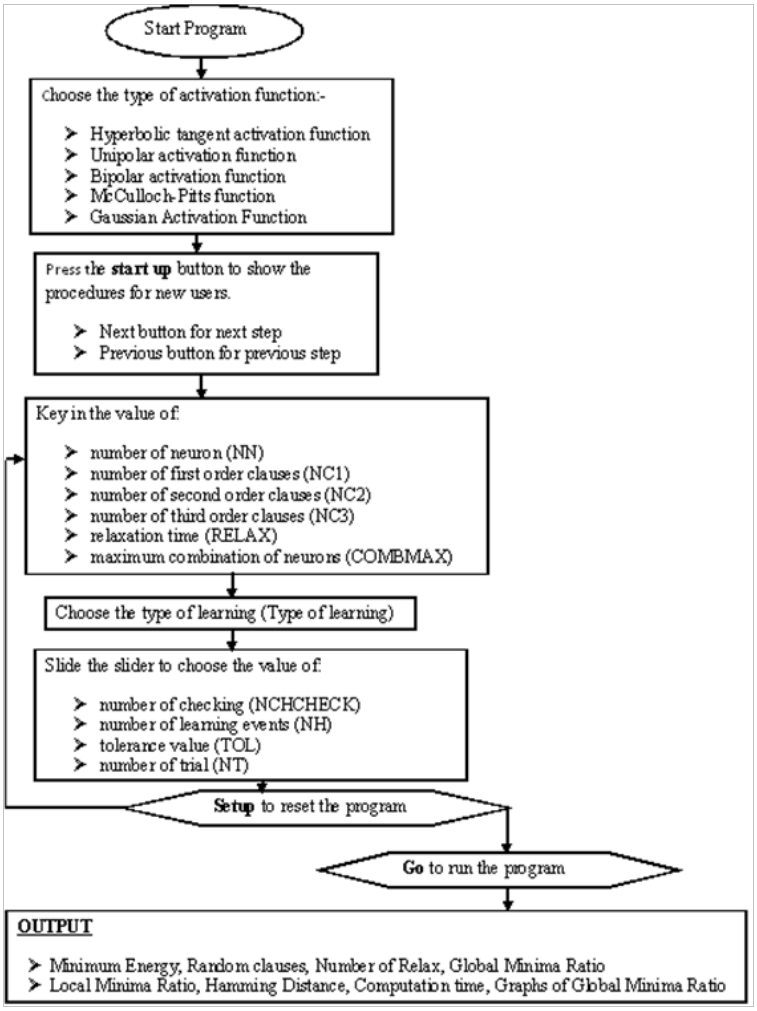

Figure 4 Flow chart of agent based modelling.

Citation: Alzaeemi SAS, Sathasivam S. Hopfield neural network in agent based modeling. *MOJ App Bio Biomech.* 2018;2(6):334–341. DOI: 10.15406/mojabb.2018.02.00089

Discussion

We test the agent based modelling with computer simulation. We generate a set of 3 random clauses with 4 variables and subject 10 neurons network through Wan Abdullah's Method from events satisfying the generated clauses. The experiment was run for various numbers of neurons (NN) from 10 up to 80 and number of literals per clause (NC1, NC2, NC3). The network complexity increases as the number of neuron (NN) increases. The development of agent based modeling was done using Microsoft Window 7 professional 64-bit, with the following specification (500GB hard disk, 4096MB RAM and processor 3.40GHz) since the computer specification play a significant role in the performance of the Agent Based Modelling (ABM). The developed ABM was design in such a way that latest version of NETLOGO (5.3.1), tools and techniques were utilized. The interface of the ABM was designed in such a way the programmer/ user see the series of procedures and stages involves so he/she has flexibility to adjust input parameters at the beginning of the programme and see the results of the global minima, hamming distance, computational time to mention but few at the end of the simulation. By using agent based modelling for activation functions to do logic programming in Hopfield Network computer we verified that we can obtains models for the logic program. This however shows that we can implement activation functions to do logic programming in Hopfield Network model through a specific procedure. Furthermore, using ABM, user can analyze the graphical design of the links more efficiently and systematically.

Conclusion

In this paper, we had developed agent based modeling for activation functions to do logic programming in Hopfield Network by using NETLOGO as platform. Agent Based Modelling (ABM)is easy to handle but designed in different ways that give the users know more in Netlogo such as the user input message, read data systems that ease the users to keys in and 2D animation that had carried out. They were very user friendly. Besides, the benefits of ABM can be summarized as below:

A. ABM is able to integrate or link logic program and Hopfield Network. It offers a natural description of a system.

B. ABM is able to produce models for the set of logic program because it captures emergent phenomena.

C. ABM is flexible. Users can change the training parameters.

D. Although, ABM develops model, the process for activation functions to do logic programming in Hopfield Network are quite efficient and the system still faces oscillation problem when involving in high complexity of system. Thus, to improve it a future work will carry out.

Acknowledgments

None.

Conflicts of interest

The author(s) declares that there is no conflicts of interest.

References

1. Hopfield JJ. Neural Networks and Physical System with Emergent Collective Computational abilities. *Proc Natl Acad Sci USA*. 1982;79(8):2554–2558.
2. Hopfield JJ. Neurons with Graded Response Have Collective Computational Properties like Those of Two-State Neurons. Proc *Natl Acad Sci USA*. 1985;81(10):3088–3092.
3. Kowalski RA. Logic Programming for Problem Solving, New York: Elesvier Science Publishing Co; 1979.
4. Sathasivam S. Upgrading Logic Programming in Hopfield Network. *Sains Malaysiana*. 2010;39:115–118.
5. Sathasivam S. Learning in the Recurrent Hopfield Network, Proceedings of the Fifth International Conference on Computer Graphics, Imaging and Visualisation, 2008. p. 323–328.
6. Sathasivam S. Energy Relaxation for Hopfield Network with the New Learning Rule. International Conference on Power and Optimazation, Bali, 2009.
7. Pinkas G, Dechter R. Improving energy connectionist energy minimization. *Journal of Artificial Intelligence Research*. 1995;3:223–15.
8. Rojas R. Neural Networks: A Systematic Introduction. Berlin: Springer; 1996.
9. Wan Abdullah WAT. Logic Programming on a Neural Network. *Malaysian Journal of computer Science*. 1993;9(1):1–5.
10. Sathasivam S, Fen NP. Developing agent based modeling for doing logic programming in hopfield network. *Applied Mathematical Sciences*. 2013;7(1):23–35.
11. Heckbert, Scott, Tim Baynes. Agent-based modeling in ecological economics. *Ann NY Acad Sci*. 2010;1185(1):39–53.
12. Macal Charles M, Michael J. Tutorial on agent-based modeling and simulation. In: Proceedings of the 37th conference on Winter simulation. Winter Simulation Conference, 2005. p. 2–15.
13. Vilhelm D, Peter J, Magnus W. Counting models for 2SAT and 3SAT formulae. *Theoretical Computer Science*. 2005;332(1): 265-291.
14. Mansor, Mohd Asyraf, Mohd Shareduwan M. Kasihmuddin, and Saratha Sathasivam. "VLSI Circuit Configuration Using Satisfiability Logic in Hopfield Network. *International Journal of Intelligent Systems & Applications*. 2016;8(9).
15. Hopfield JJ. Neurons with Graded Response Have Collective Computational Properties like Those of Two-State Neurons. Proc *Natl Acad Sci USA*. 1984:81(10):3088–3092.
16. Sathasivam S. Logic Mining in Neural Networks. PhD. Thesis. Malaysia: University of Malaya; 2006.
17. August Mayer, Gerald W, Markus S. Applications of Hopfield Networks. University of Salzburg-Institute for Computer Science, Austria, 1999.
18. Michael LS. Programming Language Pragmatics. United States: Morgan Kauffman Publications; 2008.
19. Wilensky U. NetLogo Simulation Software northwestern. edu/netlogo Center for Connected Learning and Computer-Based Modeling. Northwestern University; 2008.
20. Wan Abdullah WAT. Logic Programming in Neural Networks. *Malaysian Journal of Computer Science*. 1996;9(1):1–5.
21. Wan Abdullah WAT. Neural Network logic. In O. Benhar, editor. Neural Networks: From Biology to High Energy Physics. Pisa: ETS Editrice. 1991. p. 135–142.
22. McCulloch WS, Pitts W. A logical calculus of the ideas immanent in nervous activity. *The bulletin of mathematical biophysics*. 1943;5(4):115–33.
23. Kalveram KH. A Neural Network enabling sensorimotor learning. *Neurocomputing*. 1992;55(1):299–314.

24. Kasihmuddin MS, Sathasivam S. Accelerating activation function in higher order logic programming. In Advances In Industrial And Applied Mathematics: Proceedings of 23rd Malaysian National Symposium of Mathematical Sciences (SKSM23) AIP Publishing. 2016;1750(1):030006.
25. Mansor MA, Sathasivam S. Performance analysis of activation function in higher order logic programming. In Advances In Industrial And Applied Mathematics: Proceedings of 23rd Malaysian National Symposium of Mathematical Sciences (SKSM23) AIP Publishing. 2016;1750(1):030007.
26. Berryman MJ. Review of software platforms for agent based models. Technical Report DSTO-GD-0532, Defence Science and Technology Organisation, Australia: Edinburgh; 2008.
27. Ramanath M, Gilbert N. The design of participatory agent-based social simulations. *Journal of Artificial Societies and Social Simulation*. 2004;7(4).
28. Tisue Seth, Uri Wilensky. Netlogo: A simple environment for modeling complexity. *International conference on complex systems*. 2004. p. 16–21.
29. Wilensky U. Paradox, programming, and learning probability: A case study in a connected mathematics framework. *The Journal of Mathematical Behavior*. 1995;14(2):253–280.
30. Resnick M. Beyond the centralized mindset. *The journal of the learning sciences*. 1995;5(1):1-22.
31. Wilensky U. Statistical mechanics for secondary school: The GasLab multi-agent modeling toolkit. *International Journal of Computers for Mathematical Learning*. 2003;8(1):1–41.
32. Rader C. Making constructionism work in the classroom. *International Journal of Computers for Mathematical Learning*. 2003;8(1):63–108.
33. Wilensky U, Rand W, Kornhauser D. Visualization tools for agent-based modeling in NetLogo. Chicago, 2007. 15–17.
34. Gilbert GN. Agent-based models (No. 153). Sage, 2008.
35. Sathasivam S, Wan Abdullah WAT. Logic Learning in the Hopfield Networks. *Modern Applied Science*. 2008;2(2):57–62.

Citation: Alzaeemi SAS, Sathasivam S. Hopfield neural network in agent based modeling. *MOJ App Bio Biomech.* 2018;2(6):334–341.
DOI: 10.15406/mojabb.2018.02.00089