DEVELOPING AGENT BASED MODELING FOR LOGIC PROGRAMMING AND REVERSE ANALYSIS FOR HOPFIELD NETWORK

NG PEI FEN

UNIVERSITI SAINS MALAYSIA

2013

DEVELOPING AGENT BASED MODELING FOR LOGIC PROGRAMMING AND REVERSE ANALYSIS FOR HOPFIELD NETWORK

by

NG PEI FEN

Thesis submitted in fulfillment of the requirements
for the degree of
Master of Science

Jun 2013

ACKNOWLEDGEMENTS

This thesis would not have been successful without the support of many people that I'm really grateful especially my supervisor Dr. Saratha A/p Sathasivam. Without her guidance, I would not have gone that far. When I am faced with problems, she was the one that lends her helping hand to me and show me the way to solve. I sincerely thank her for her continuous help and encouragement.

MyBrain 15 sponsored me the scholarship for tuition fees for 2 years. Universiti Sains Malaysia mathematical school staffs and lecturers provided me the facilities and documents.

Moreover, my sister, Pei Fang and family members provided me love, care and financial support for me to study master in Universiti Sains Malaysia.

Last but not least, I would like to take this opportunity to thank my friend, Xin Ying, who helped me out with my grammar corrections after I had done my thesis writing and Peh Sang, who guided me in my thesis format.

In a nutshell, I would take this opportunity to express my gratitude towards all lecturers, family members and friends whom had given me continuous support and encouragement for me to complete my thesis writing and my studies.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF ABBREVIATIONS

ABM	Agent based modeling
ANN	Artificial neural networks
BCI	Brain-computer interface
CNF	Conjunctive normal form
COMBMAX	Maximum combination of neurons
CPU	Central processing unit
HOHN	Higher order Hopfield networks
NC1	Number of first order clause
NC2	Number of second order clause
NC3	Number of third order clause
NC4	Number of fourth order clause
NC5	Number of fifth order clause
NCHCHECK	Number of checking
NH	Number of learning events
NN	Number of neuron
NP	Nondeterministic polynomial time
NT	Number of trial
RELAX	Relaxation time
TOL	Tolerance value

LIST OF PUBLICATIONS

i. Saratha Sathasivam and Ng Pei Fen. (2013) Developing Agent Based Modeling for Doing Logic Programming in Hopfield Network, Applied Mathematical Sciences, 7(1), pp 23-35 (indexed in scopus)

ii. Saratha Sathasivam and Ng Pei Fen. (2013) Application of Higher Order Hopfield Network, accepted For 6th Engineering Conference, Sarawak, 3-4 Julai 2013.

iii. Saratha Sathasivam and Ng Pei Fen. (2013). Assimilating Boltzmann Machine and Reverse Analysis Method, accepted for International Conference of Mathematics in Kerala, India, 9-10 August 2013.

iv. Saratha Sathasivam, Ng Pei Fen and Nawaf Noor. (2013) Developing Agent Based Modeling For Reverse Analysis Method, accepted for publication in Research Journal of Applied Sciences and Technology. (indexed in scopus)

MEMBANGUNKAN EJEN PEMODELAN BAGI PENGATURCARAAN LOGIK DAN ANALISIS SONGSANG BAGI RANGKAIAN HOPFIELD

ABSTRAK

Untuk pengaturcaraan aras tinggi, seni bina rangkaian peringkat yang lebih tinggi adalah diperlukan seperti rangkaian neural aras tinggi untuk mempunyai kadar penumpuan lebih cepat, kapasiti penyimpanan yang lebih besar, sifat anggaran yang lebih kuat, dan toleransi kesalahan yang lebih tinggi berbanding rangkaian neural aras rendah. Demikian, peringkat yang lebih tinggi rangkaian Hopfield telah membawa kepada tesis ini dengan menggunakan pengaturcaraan logik dan analisis songsang dalam rangkaian Hopfield. Matlamat melaksanakan pengaturcaraan logik berdasarkan skim pengurangan tenaga adalah untuk mencapai minimum global terbaik. Walau bagaimanapun, tidak ada jaminan untuk mencari sekurang-kurangnya yang terbaik dalam rangkaian. Oleh itu, Mesin Boltzmann dan fungsi pengaktifan tangent hiperbolik diperkenalkan untuk mengatasi masalah ini. Untuk memilih kaedah terbaik dan berkesan untuk mendapatkan minima global antara kaedah Wan Abdullah (menggunakan McCulloch-Pitts mengemaskini peraturan dalam Hopfield bersih), mesin Boltzmann dan fungsi pengaktifan tangent hiperbolik, jadual perbandingan akan dibangunkan dalam tesis ini. Untuk menjalankan kerja itu, model berasaskan ejen diwujudkan. NetLogo digunakan untuk menjalankan pengaturcaraan logik dan analisis songsang. Ejen Pemodelan boleh membenarkan pembangunan yang pesat model, ciri mudah ditambahkan dan mesra pengguna mengendalikan dan pengkodan. Dalam sistem pengaturcaraan logik, hasil dalam tempoh minimum global bukan sahaja akan dianalisis manakala dalam aspek jarak Hamming dan masa bagi unit pemprosesan pusat (CPU) juga akan dijalankan. Dalam sistem analisis songsang, hubungan yang wujud antara data dengan mengeluarkan corak biasa yang wujud

dalam set data akan dibelajar. Kita boleh mencari ketidakpastian dan hubungan yang tidak diduga. Dengan ini, apikasi masalah kehidupan sebenar akan dijalankan dengan menggunakan ejen pemodelan untuk menjalankan simulasi komputer dalam tesis ini. Apikasi masalah kehidupan sebenar yang melibatkan markah peperiksaan pelajar dalam peperiksaan ujian satu dan kes kehidupan sebenar mengenai sifat-sifat haiwan di zoo akan dijalankan. Kemudian untuk keputusan selepas menggunakan ejen pemodelan, penyelesaian global telah dicapai dalam pengaturcaraan logik aras rendah (peringkat ketiga, kedua dan pertama) manakala bagi peringkat lebih tinggi (peringkat keempat dan kelima) adalah sukar untuk mencapai kerana penyimpanan memori yang tinggi untuk maklumat neuron. Dalam sistem analisis songsang, keputusan yang baik telah diperolehi dalam aras rendah untuk ramalan data manakala bagi aras tinggi disebabkan isu kerumitan yang tinggi, masa yang diperlukan untuk mendapat keputusan adalah amat lama. Oleh itu, sebagai tanda aras, bagi aras rendah yang terhad sehingga tiga atom telah memberi keputusan yang baik dan merupakan peringkat yang sesuai dalam menggunakan ejen pemodelan berbanding dengan peringkat lebih tinggi. Sebagai kesimpulan, ejen pemodelan telah berjaya berfungsi dan mencapai matlamat objektif yang dinyatakan di dalam tesis ini.

DEVELOPING AGENT BASED MODELING FOR LOGIC PROGRAMMING AND REVERSE ANALYSIS FOR HOPFIELD NETWORK

ABSTRACT

For higher-order programming, higher order network architecture is necessary as high order neural networks have faster convergence rate, greater storage capacity, stronger approximation property, and higher fault tolerance than lower-order neural networks. So, higher order Hopfield network is brought into this thesis by using logic programming and reverse analysis in Hopfield network. The goal of performing logic programming based on the energy minimization scheme is to achieve the best global minimum. However, there is no guarantee to find the best minimum in the network. Thus, Boltzmann Machines and Hyperbolic Tangent activation function are being introduced to overcome this problem. To choose the best and efficient method to obtain the global minima among Wan Abdullah method (use McCulloch-Pitts updating rule in Hopfield net), Boltzmann machine and Hyperbolic Tangent activation functions, a comparison table will be created in this thesis. To carry out such work, agent based modeling (ABM) is created. NetLogo as the platform to carry out logic programming and reverse analysis. ABM can allow rapid development of models, easy addition of features and a user friendly handling and coding. In logic programming systems, not only the result in terms of global minimum will be analyzed but in the aspect of hamming distance and central processing unit (CPU) times will also be carried out. In reverse analysis systems, the inherent relationships among the data can be learned by extracting common patterns that exist in data sets. The unknown and unexpected relation can be seek. As a result, real life cases will be carried out by using ABM to run computer simulation in this thesis. The real cases involves a real case that is concern about the examination mark

of students during first examination and a real life case regarding the attributes of animals in zoo. Later for the results after the runs of ABMs, the global solutions were achieved in lower order (third, second and first order) logic programming while for higher order (fourth and fifth order) was hard to achieve due to the high memory storages for neuron informations. In reverse analysis method, lower order had generated good results for data predictions while for higher order due to high complexity issue, the result took a long time. From here as a benchmark, lower order that restricted up to three atoms had given great result and was a suitable order to run for ABMs compare to higher order. As a conclusion, the ABMs that builded in this thesis had successfully functioned and achieve the goal of objectives stated in this thesis.

CHAPTER 1

INTRODUCTION

1.1 Introduction to Neural Networks

A large variety of applications of neural network has been widely in use in various areas in the last two decades. A neural network is a mathematical (or computational) model that is inspired by the structure and function of biological neural networks in the brain. A technical neural network consists of simple processing units, the neurons, and directed, weighted connections between those neurons (Gershenson, 2003). Here, the strength of a connection (or the connecting weight) between two neurons *i* and *j* is referred to as w_{ij} . A neural network formed by a group of artificial neurons (i.e., nonlinear processing units) which are connected to each other via synaptic weights (interconnected) use a connectionist move towards to compute and process information. This is because neural network can solve and analyze sophisticated recognition problem via learning from raw data. During learning process from raw data, neural networks gain expert knowledge from problem field and have the capability to simplify this knowledge. This ability often outperforms the human professionals. Artificial neural networks (ANN) have been successfully used in various applications such as biological, medical, industrial, control engendering, software engineering, environmental, economical, and social applications in image processing, pattern recognition, optimization solvers, fixed-point computation and other engineering areas (Balasubramaniam & Rakkiyappan, 2008).

1.1.1 Artificial Neural Networks

Artificial neuron is a necessary building block of every artificial neural network (Krenker et al, 2011). Its functionalities and design are based on the observation of biological neurons that formed biological neural networks. It can be seen in Figure 1.1 and Figure 1.2.

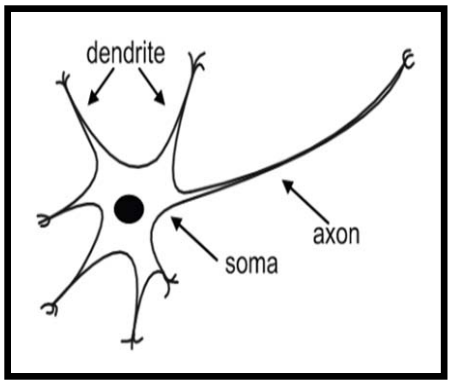

Figure 1.1: A biological neuron

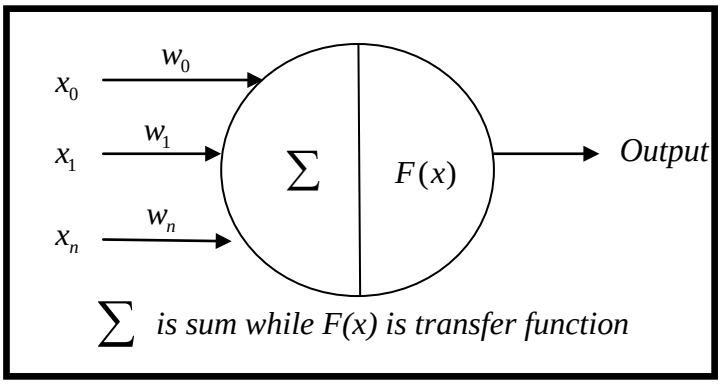

Figure 1.2: An artificial neuron

A typical neural network is consists of n input units $(x_0 \ldots x_n)$, weights of inputs $(w_0 \ldots w_n)$ and one output axon *y(x)* (Krose & Smagt, 1996). *F(x) is* an activation function that weight how leading the output should be from the neuron derived from the sum of the input.

The activation function, *F(x)* is a sigmoid form. It can be a Hyperbolic Tangent or logistic function. Logistic function is the standard sigmoid function that is the most frequently used function in neural networks. The McCulloch-Pitts updating rule is one of generalization that uses activation functions other than the threshold function. The logistic function, defined by

$$y_i = \frac{1}{1+\exp(-\beta v_i)} \tag{1.1}$$

where v_i is the induced local field of neuron *i*, and y_i is the output of the neuron. The slope, β (also called gain) of the sigmoid function can be changed. The larger the β, the steeper the slope, the more closely it approximates the threshold function

μ_i . It maps to the range of values (0, 1). Hyperbolic Tangent activation function, will be applied in doing logic programming in Hopfield networks and will be compared with the McCulloch-Pitts updating rule to enhance the better global minima value.

After neurons go through the training set, in order to improve the values of the weights (connection strengths) to make error smaller, a gradient descent method is required. It is able to adjust the weights till the error reaches acceptable level. Although it does not guarantee convergence to a global minimum, it gets close to one if a sufficient number of hidden units are used. Thus, to reduce the potential of being stuck in the local minima, Boltzmann machine, a type of gradient descent method will be introduced in chapter 3 to get through the problem.

1.1.2 Network Architectures

Based on the connection pattern (architecture), ANN can be grouped into two major categories as feed-forward networks and feedback networks (recurrent network). Figure 1.3 shows typical networks for each category.

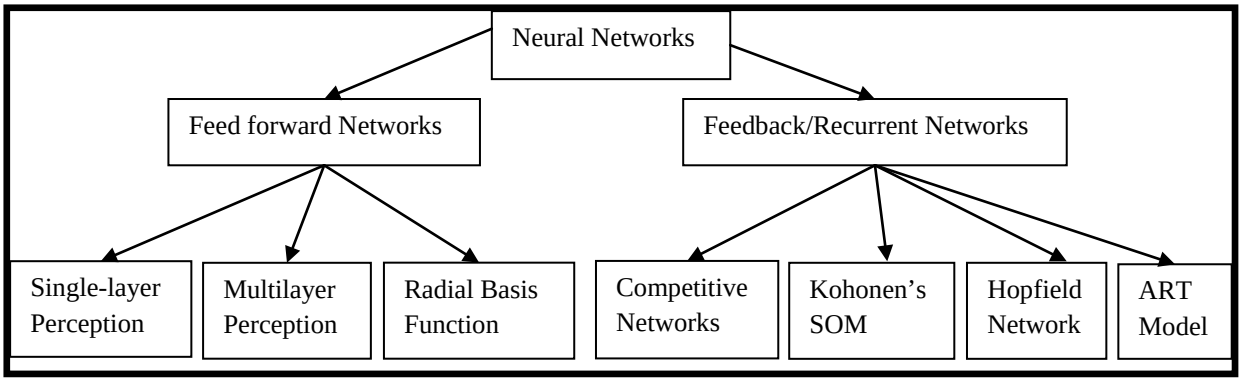

Figure 1.3: A taxonomy of network architectures

However, discrete Hopfield network only will be focusing in this thesis,

1.2 Introduction to Hopfield Network

A Hopfield network is one of the recurrent networks that is used to store network memory. These units have binary values 0 and 1 or 1 and -1 to determine whether the units input exceed their threshold or not. The network has symmetric weights with no self-connections. The existence of Lyapunov function (energy function) in the Hopfield network enables the network to converge to a stable set of activations, rather than oscillating. This is because energy exists, and it is local minima or global minima that are attractors towards which the system flows until it gets stuck in one of them. This was associated by Hopfield as associative memory.

The Hopfield Networks with the order = *n-1* is stated as below. The energy function (Ding et al, 2010a) is

$$\begin{aligned} E = & -\frac{1}{n}\sum_{i_1}\sum_{i_2}\sum_{i_3}\cdots\sum_{i_n} J_{i_1 i_2 i_3 \ldots i_n} S_{i_1} S_{i_2} S_{i_3} \ldots S_{i_n} \\ & -\frac{1}{n-1}\sum_{i_1}\sum_{i_2}\sum_{i_3}\cdots\sum_{i_{n-1}} J_{i_1 i_2 i_3 \ldots i_{n-1}} S_{i_1} S_{i_2} S_{i_3} \ldots S_{i_{n-1}} - \ldots - \frac{1}{2}\sum_{i_1}\sum_{i_2} J_{i_1 i_2} S_{i_1} S_{i_2} - \sum_i J_i S_i \end{aligned} \tag{1.2}$$

$J_{i_1 i_2 i_3 \ldots i_n}$ defines the connection weights of the *n*th order connection from neurons $i_1 i_2 i_3 \ldots i_n$ to neuron *i*, h_i is the input potential to neuron *i* and S_i is the state of neuron *i*. In the high-order model each node is assigned a sigma-pi unit that updates its activation value by first computing the partial derivative of the energy function. The dynamic equation or the updating rule of the network is

$$S_i(t) = \operatorname{sgn}(h_i(t)),$$

$$\begin{aligned} h_i = & \sum_{i_1}\sum_{i_2}\sum_{i_3}\cdots\sum_{i_n} J_{i_1 i_2 i_3 \ldots i_n} S_{i_1} S_{i_2} S_{i_3} \ldots S_{i_n} + \\ & \sum_{i_1}\sum_{i_2}\sum_{i_3}\cdots\sum_{i_{n-1}} J_{i_1 i_2 i_3 \ldots i_{n-1}} S_{i_1} S_{i_2} S_{i_3} \ldots S_{i_{n-1}} + \ldots + \sum_{i_1}\sum_{i_2} J_{i_1 i_2} S_{i_1} S_{i_2} + \sum_i J_i S_i \end{aligned} \tag{1.3}$$

where sgn is signum function. The connection weight of Hopfield networks is symmetrical. That means that the value of $J_{i_1 i_2 i_3 \ldots i_n}$ is independent of the ordering of

index like $J_{123} = J_{132} = J_{213}...$. This condition is analogical to the symmetric requirement of the Hopfield Networks connection weight matrix. Below is a standard energy function for discrete Hopfield network.

$$H = -\frac{1}{2}\sum_i \sum_j J_{ij} S_i S_j - \sum_i J_i S_i \qquad i = 1, 2, \ldots N \; j = 1, 2, \ldots N \qquad (1.4)$$

Hopfield network is practical in use as an analog computer or as a content addressable memory for solving combinatorial optimization problems (Ding, 2013). Combinatorial optimization involves looking for the combination of choices from a discrete set which yields an optimum value for some associated cost function. If neurons were mapped to choices and equate *H* to the cost function concerned, combinatorial optimization on the neural network can be carried out. The values of the connections are obtained from the equation for *H* and the network with these connections are relaxed with the appropriate dynamics to arrive at the minimum of *H* and thus the cost function. The resulting configuration then yields the optimal choices.

The strength of the connections between neurons and the parameters of the activation function (thresholds) will be modified to present the learning in this field. The neurons are interconnected. Hence, the network structure is built in a way like Figure 1.4. Hopfield (1982) presented a neural network model as a theory of associative memory and brought his skills in physics to the world of neurobiology which was a larger contribution to enhance a better understanding for people to know how the brain thinks.

A Hopfield network was connected by a symmetric network structure with binary neurons. Binary can be the active (”firing”, 1) or inactive (”not firing”, 0) neurons. It is consists of *n* input units $(x_1 \ldots x_n)$, weights of inputs $(w_1 \ldots w_n)$ and

outputs axon $(Y_1 \ldots Y_n)$ The connections are weighted, and depending on the sign of the weight they can be activated.

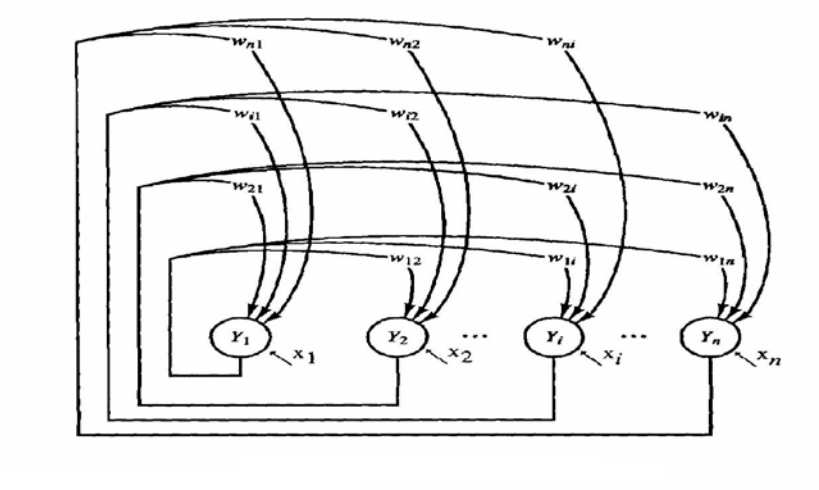

Figure 1.4: A simple diagram of Hopfield Net

At first, the neuron's activation matches the pattern to recognize at the beginning of the calculation of the network output. Then the state of the neurons will calculated until the network is stable, which mean the network state does not change any more. This can be seen as the minimization of the energy in the net, so that the final state is a minimum of an energy function called attractor.

1.2.1 The Hebb rule

Hebb (1949) formulated the **Hebbian rule** which is the basis for most of the more complicated learning rules that will discuss in this thesis.

According to Donald Hebb :*When an axon of cell A is near enough to excite a cell B and repeatedly or persistently takes part in firing it, some growth process or metabolic changes take place in one or both cells such that A's efficiency as one of the cell firing B, is increased.*

This statement obtained from a neurobiological context. Two-part rules may be expanded and rephrased:

i. The connection strength is selectively increased if two neurons on any side of a connection are activated at the same time.

ii. The connection strength is selectively eliminated or weakened if two neurons on any side of a connection are activated asynchronously.

The simplest form of Hebbian learning formula for the modification of synaptic strengths, J_{ij} when memorizing a pattern $\{\xi\}$ is described by:

$$J_{ij}(new) = \lambda \xi_i \xi_j + J_{ij}(old) \tag{1.5}$$

The coefficient λ can be called the rate of learning.

If learning is started from zero connection strengths or *tabula rasa*, equation (1.5) can be rewritten as:

$$J_{ij} = \lambda \xi_i \xi_j \tag{1.6}$$

1.3 Logic Program

A logic program is formed by a set of axioms in order to fulfil the goal statement. It is because the implementation of a logic program corresponds to the goal statement that is fulfilled from the axioms. According to J. A. Robinson, a program is a theory and computation is deduction from the theory. When a program was started to be create, it is all just starting to begin a theory, while when the program started to be run which mean started to compute data, it is just an implication of the theory. Thus, the process of software engineering becomes a problem description or describes the control component of the program. It makes reader closer to problem domain thus higher programmer productivity. Besides, logic program has simple declarative semantics and referential transparency. It is suitable

for prototyping and exploratory programming. Further explanation for logic program will carry out in chapter 2.

1.4 Reverse Analysis

Logical knowledge exists in the synaptic connections of a network. Reverse analysis is a method that when examining the connection strengths obtained during the learning process, logical rules that have been acquired may be deduced. This step will repeat until it gets the value that is wanted. From the reverse analysis given the values of the connections of a network, let the readers to know what logical rules are entrenched in it. Besides, the inherent relationships among the data by extracting common patterns that exist in dataset can be learnt. Readers can look for unknown and unexpected relations, which can be uncovered by exploring the data in the data sets. Moreover, reverse analysis can be used as a data mining method. Data mining is an automatic induction of knowledge from data sets. Nowadays, manual methods of gaining knowledge from data sets have been statistical methods. Thus, new methods beside statistical ones are used. For further explanation will be done in chapter 2 in the reverse analysis section.

1.5 Agent Based Modeling

Agent Based Modeling (ABM) is a paradigm where a simulation is built in a bottom-up manner by specifying the individual components of a system and their interactions. Aggregate behavior is observed as an emergent property of the interactions of the agents and their environment. An emergent property of a system is a property that arises from the interactions of entities in the system. ABM is often

contrasted with traditional, equation based approaches that model system in a top-down manner.

According to an article wrote by Bonabeau (2002), he had listed 5 criteria for situations that benefit from the use of Agent Based Modeling techniques.

- The interactions are discrete, nonlinear, discontinuous or complex among the agents.
- The agents' locations are not fixed and space is essential.
- Each agent is different and the population is heterogeneous.
- The topology of interactions is complex and heterogeneous.
- The agents show complex behaviour such as adaptation and learning.

Clearly Hopfield networks meet most, if not all, of these criteria. Due to the size of Hopfield network that is involved, Agent Based Models are well-suited to this domain. Further explanation will carry out in chapter 4.

1.6 Literature review

In this review lots of information regarding artificial neural networks, Hopfield network, logic program on Hopfield network, reverse analysis of Hopfield network, Boltzmann machine, Hyperbolic Tangent activation function and Agent Based Modeling were explored.

According to Krenker et al (2011), artificial neural network is a mathematical model that is inspired from the functionalities and structure of biological neural networks. It is based on the assumptions that neuron is a component that processes and transmit information. Each connection link has a concentrated weight and in each neuron there is an activation function (threshold) which is implemented to its

network input to determine its output signal. More details about neural network can be obtained from any neural network books (Akyol & Bayhan, 2007).

Hopfield network consists of linked neurons with binary activation with the weights being symmetric between the individual neurons and without any neuron being directly connected to itself (Kriesel, 2005). The output of each neuron is reaction to each other neurons in the network so that it is a dynamic network. It is practical in use as an analog computer or as a content addressable memory for solving combinatorial optimization problems.

Traditional Hopfield networking was widely used to solve combinatorial optimization problems. However, higher order Hopfield networks (HOHN), as an expansion of traditional Hopfield networks, are seldom used to solve combinatorial optimization problems. It is because it needed lots of memory spaces and consumed times. In theory, compared with low order networks, higher order networks have better properties, such as stronger approximations and faster convergence rates. In other words, the higher order Hopfield network has the ability to overcome the difficulty to find suitable parameters to guarantee convergence and explore a new path for artificial intelligence and intellectual computer. HOHN can also be applied in data mining such as forecasting and simulating data. Other than that, HOHN can solve non-linear and discontinuous data in larger field and connections. For example, it performs well in nonlinear statistical modelling and it can provide a new alternative to logistic regression in bigger state and numbers like prediction of a wholesale buyer tendency to purchase the products or not. Furthermore, HOHN is able to detect all possible interactions between predictor variables such as detect complex nonlinear relationships between dependent and independent variables. For example, user may want to compute the relationship between the dose of a drug and

its effectiveness, the relationship between the price of a shop lot and the time it takes to sell it, etc. Lastly, it can be used as a research tool like neurobiologists use for interpretation of neurological phenomena such as endogenous-based BCI systems. It can select and use a combination of some of interpretation of neurological phenomena to improve the performance of the system. From here, reader know that Hopfield network can be used to minimize a configurationally energy function and thus can solve the combinatorial optimization problem (Cheung & Lee, 1993a). This is a reason why there are better solutions that can be found. Lots of benefits by using HOHN directly had proved that it is an essential method in solving NP-complete optimization problem (Cooper, 1995, Ding et al, 2010b, Cheung & Lee, 1993a, Cheung & Lee, 1993b, Shen et al, 2005) such as Travelling salesman problem whereby positive solutions would be produced by HOHN.

Few papers had been carried out that applied higher order Hopfield networks such as using HOHN (Ding et al, 2010b) to solve N-queens problem and construction method of energy function and neural computing method. Besides, compared with the first order Hopfield network and the method how to speed the convergence and escape from the local minima are also discussed in those papers. In Cheung and Lee, (1993a) the convergence property had been restudied before being put in real life application. It had extended the reshaping idea to generalized Higher-order Hopfield network and the corresponding reshaping strategy to guarantee convergence is derived. Moreover, the Ising spin problem (Cooper, 1995) also had carried out on it. It provides to demonstrate the limitations of the Hopfield network and the manner wherein HONNs may overcome these limitations in Ising spin problem. The formation of seed points is encouraged in locally optimal segments due to the tendency of the Hopfield network's dynamics. The most important paper that

affects the main backbone of this thesis is in Joya et al (2002). It is a study of the different dynamics in HOHN and problem affecting the practical application of these networks is brought to light such as incoherence between the network dynamics and the associated energy function, error due to discrete simulation on a digital computer, existence of local minima and convergence depends on the coefficients weighting the cost function terms. However, in this thesis only local minima and convergence are considered. For higher-order programming, higher-order network architecture is necessary as high order neural networks have faster convergence rate, greater storage capacity, stronger approximation property, and higher fault tolerance than lower order neural networks. Thus, the HOHN are brought into the case as the algorithm for higher order neural network of doing logic programming in Hopfield network based on the energy minimization scheme, restricting ourselves to program clauses of up to five atoms. If more than five atoms, system will oscillating.

One paper that reviews satisfiability aspect of logic program in Hopfield network is entitled “The Satisfiabilty Aspect of Logic on Little Hopfield Network” written by Sathasivam and Wan Abdullah (2010). They reviewed a method of doing logic programming in Hopfield network based on the energy minimization scheme, restricting ourselves to program clauses of up to three atoms. Computer simulations to demonstrate the ability of doing logic programming in Hopfield network was carried out. Besides that, energy landscapes of network programmed by program clauses in the aspects of satisfiability and complexity are also being analyzed.

In order to avoid or escape from local minimum and search for the global minimum is to utilize simulated annealing. In the paper entitled “Fundamental properties of Hopfield Networks and Boltzmann Machines for Associative Memories” written by Meyder and Kiderlen (2008), Hopfield model is a dynamic associative-

memory model, and the Boltzmann machines as well as some other stochastic models are proposed as its generalization. They show the method of approximate numerical simulation at a finite temperature using Boltzmann method which will bring the techniques of statistical mechanics to tolerate on optimization.

The major advantage of Boltzmann machine technique is that the procedure or network need not get stuck by permitting uphill moves. The transitions out of a local minimum are always likely at nonzero temperature. The overview of the method promise that simulated annealing will be a very good and widely applicable on logic programming in Hopfield neural network.

Hinton (2007) described the type and stochastic dynamics of a Boltzmann machine. The learning of Boltzmann machine and its ability to solve computation problems also will be carried out.

Besides, Hyperbolic Tangent activation function is introduced to perform better performance on obtaining global solution. Hyperbolic Tangent activation function is one of an example of transfer function that is well known in neural network.

Karlik and Olgac (2007) had described the various types of activation functions that is performed in neural network. It had shown that Hyperbolic Tangent activation functions had performed the best performance among the activation functions. This is because by using hyperbolic, the states are being updated more systematically. In this thesis, Hyperbolic Tangent activation function, Boltzmann machine and compare the proposed method with the McCulloch-Pitts function in logic programming will be carried out.

One paper that reviews about reverse analysis method in Hopfield network is entitled “Application of Neural Networks in predictive data mining” written by

Sathasivam (2011). It describes the use of reverse analysis in data mining such as sweep through database and identifies previously hidden patterns and enables to discover the trends in the dataset. A simple simulation example implemented Reverse analysis method is carried out.

This paper had inspired us to further on the reverse analysis that involves higher order clauses that means imply higher order connections. In this thesis fifth order connections of clauses will be performed for prediction. Furthermore, some real life cases that involve higher order clauses also will be carried out.

Moreover, nowadays multi-agent systems are becoming more popular and widely in use. It is able to eliminate much of human workload, saves resources and time consume. A simple integrator system is the starting point of the development of distributed control. Later, more and more complex nonlinear systems will be adopted by different control methodologies (Das & Lewis, 2013). The dynamics of the agents as caught up with networked control construction might not be the same from a practical viewpoint. Therefore, an ideal distributed control should be introduced for nonlinear systems related with unknown dynamics to accommodate multiple agents. Agent based modeling is starting to crack problems that have resisted treatment by analytical methods. Many of these are in the physical and biological sciences, such as the growth of viruses in organisms, flocking and migration patterns, and models of neural interaction. In the social sciences, agent-based models have had success in such areas as modeling epidemics, traffic patterns, and the dynamics of battlefields. In recent years, the methodology has begun to be applied to economics, simulating such phenomena as energy markets and the design of auctions (Epstein, 2011). Thus, in this thesis, agent based modeling had been created for the logic programming and reverse analysis in Hopfield network.

1.7 Research methodology

In the thesis, agent based modeling in NetLogo will be created to compare methods of doing logic programming in a Hopfield network based on the energy minimization scheme by restricting up to three-atoms, two-atoms and one-atom which are considered as lower order while four-atoms and five-atoms are considered as higher order. The comparison will based on the ability of Boltzmann Machine, Hyperbolic Tangent activation function and Wan Abdullah method (using McCulloch-Pitts updating rule in Hopfield net) doing logic program on a Hopfield network by observing the ratio of global solutions, final hamming distance and central processing unit (CPU) time. Besides, agent based modeling will be created by deriving lower clauses and higher clauses in reverse analysis in Hopfield networks that is also restricted to third, second, first order connections and fourth, fifth order connections. The comparisons will be in the aspect of central processing unit (CPU) time and complexity. Real life simulations will be carried out to test the validness and robustness of Reverse analysis method.

1.8 Objective

The objectives of the thesis are:

i. To develop agent based modeling (ABM) for carrying out logic programming in Hopfield network focusing on lower order connections (up to three atoms)

ii. To develop agent based modeling (ABM) for carrying out logic programming in Hopfield network focusing on higher order connections (up to five atoms)

iii. To design coding for higher order logic programming in Hopfield network

iv. To design coding for Boltzmann machine and Hyperbolic Tangent activation function to enhance the capability of doing logic programming in Hopfield network

v. To create and implement ABM for higher order network for Reverse analysis method

1.9 Research contribution

Following are the research contributions of this thesis:

i. To develop agent based modeling (ABM) for doing lower order and higher order logic programming in Hopfield network

ii. To develop agent based modeling (ABM) for doing lower order and higher order in Reverse analysis method

iii. Introducing methods (Boltzmann machine, Hyperbolic Tangent activation function) to upgrade the performance of (i) and (ii).

iv. Creating new ABM to solve real life problems and other data mining problems

1.10 Overview

This thesis is organized as follows. Chapter 1 introduces the topic to be studied in this thesis, methodology, objectives and aims to achieve. In Chapter 2, explanation of logic programming in Hopfield network and reverse analysis method in Hopfield network will be carried out. In Chapter 3, Boltzmann machine and Hyperbolic Tangent activation function are described. In Chapter 4, introduction to NetLogo and agent based modeling. Besides, flow chart of agent based modeling in logic programming and reverse analysis in Hopfield network is discussed.

Meanwhile in Chapter 5 encloses the simulation results and discussions. Next in Chapter 6, all concern about real life cases will be described. Finally in Chapter 7, the conclusion and future work regarding this thesis.

CHAPTER 2

LOGIC PROGRAMMING AND REVERSE ANALYSIS IN HOPFIELD NETWORK

2.1 Logic Programming on Hopfield Network

In this chapter, the previously proposed general framework for integrating logic program into Hopfield network will be studied.

According Kowalski (1979), logic program provides an easy way for problem solving. A logic program is easy to understand, prove and modified relational with neural network especially used by new users and programmers. It is because in the logic programming model, programmer is expressing the basic logical relationships by combining the logic which is formed with the information that is used in problem solving and with the control over the manner that information had keyed in to build up algorithm and programs. So the relationships can perform in

Logic + control = algorithms

A proposition is a statement that can be either true or false. Thus, propositional logic is the logical form that builds based on Boolean variables and connectors. The main connectors are disjunctive (V), conjunctive (Λ), implication and negation. V and Λ will be formed in a truth table. For example, p V q and p Λ q. Let T as true and F as false.

Table 2.1: Truth table for p V q and p Λ q

p	q	p V q	P Λ q
T	T	T	T
F	F	F	F
T	F	T	F
F	T	T	F

In many applications of logic, it is adequate to restrict the form of clauses to those at most one conclusion which is known as Horn clauses. A logic program P is said to be formed by a finite set of horn clauses, in the form of $H \leftarrow L_1, L_2, \ldots\ldots L_n$

with $n \geq 0$ where all L_i $(0 \leq i \leq n)$ are literals and H is an atom. H is called as head of the clause while $L_1, L_2, \ldots\ldots L_n$ is the body of clause.

A bidirectional mapping of symmetric neural network had been defined by Gadi Pinkas (Pinkas, 1991a, Pinkas, 1991b,) and Wan Abdullah, (1991, 1992) which is involving energy functions and propositional logic formulas in order to find the solutions obtained are models in the related logic program. Both researchers are interested in Hopfield network. Moreover, both approaches can handle non-monotonically. Meanwhile, Wan Abdullah's method shows learning capability of Hopfield network by revolving around propositional Horn clauses. It is able to search for the best solutions and generate the clauses in the logic program. When new clauses are added, the corresponding solutions may change. The interpretation with least logical inconsistent … can be obtained even when the clauses in the logic program are not consistent.

Following are the algorithms for Wan Abdullah's method (Wan Abdullah, 1993).

i. Given a logic program, translate all the clauses in the logic program into the basic Boolean algebraic form. It's like $A \leftarrow B, C$ as $A \vee \neg (B \wedge C) = A \vee \neg B \vee \neg C$

ii. Identify a neuron to each ground neuron and initialize all connection strengths to zero. It assumed the connection with A, B and C is zero value.

iii. A cost function derived based on associated of negation of all the clauses, like $\frac{1}{2}(1+S_A)$ represents the logical value of a neuron A, where S_A is the state of neuron corresponding to A. The value of S_A is defined as 1 if A is true and -1 if A is false. Negation (neuron A does not occur) is represented by $\frac{1}{2}(1-S_A)$;

Based on $E_p = \frac{1}{2}(1-S_A)\frac{1}{2}(1+S_B)\frac{1}{2}(1+S_C)+ \ldots\ldots$ a conjunction logical connective 'and' is represented by multiplication while a disjunction connective 'or' is addition.

iv. Comparing the cost function with the energy, *H* which in the chapter 1 that had recognized in Hopfield network to obtain the values of connection strengths.

v. Let the network progress until minimum energy is obtained. The neural states then provide a solution interpretation for the logic program, and the truth of ground atom may be checked then consider whether the solution obtained is a global solution or not.

2.1.1 Logic programming in lower order Hopfield network

A logic program contains program clauses and it is activated by an initial goal. It is easy to understand, modify and verify. For example in a simple propositional case, logic clauses had formed as $A \leftarrow B_1, B_2, B_3, \ldots\ldots, B_n$ where the arrow can be read as 'if' while the comma can be read as 'and' for the purpose of interpretation the clauses by using truth value. Thus, a model or pattern can be found in the given logic program and it can be a way to solve the combinational optimization problem. Consequently, to carry out a logic program, a simulator needed to build up to run it. In the lowest order logic program, will only restricted up to three atoms. This is because according to Hopcroft and Ullmann (1979), a clause involving more than 3 atoms (spins) can be reduced to conjunctive normal form (CNF) of atomistic clauses C_i containing only 3 atoms. So, three atoms is sufficient for analyses. Further connections will carry out in higher orders.

For an example, consider the logic program below:

$$P = A \leftarrow B, D$$

$$\Lambda \; C \leftarrow B$$

$$\Lambda \; D \leftarrow \qquad (2.1)$$

Given the goal $\leftarrow B$ where B is a conjunction of neurons. The $P \Lambda \neg B$ required shown as inconsistent so that the goal can be proven.

First of all, translate all clauses and the negation of it into Boolean algebraic form (Zhang, et al, 2011):

$$P = (A \vee \neg B \vee \neg D) \wedge (C \vee \neg B) \wedge D$$

$$\neg P = (\neg A \wedge B \wedge D) \vee (\neg C \wedge B) \vee (\neg D) \qquad (2.2)$$

From this, write a cost function for bipolar neurons that is minimized when all the clauses are fulfilled as follow:

$$\begin{aligned} E_p &= \frac{1}{2}(1-S_A)\frac{1}{2}(1+S_B)\frac{1}{2}(1+S_D)+\frac{1}{2}(1-S_C)\frac{1}{2}(1+S_B)+\frac{1}{2}(1-S_D) \\ &= \frac{1}{8}(1-S_A)(1+S_B)(1+S_D)+\frac{1}{4}(1-S_C)(1+S_B)+\frac{1}{2}(1-S_D) \end{aligned} \qquad (2.3)$$

where the state of neurons S_A represent the truth values of A. Multiplication operation represents the relationship "AND" and addition operation is "OR".

Here, the minimum value for E_p is 0 when the fact had shown that all the clauses are satisfied. The value for E_p is proportional to the number of clauses unsatisfied. Table 2.2 is the truth table for $P = \{A \leftarrow B, D, C \leftarrow B, D \leftarrow .\}$.

Table 2.2: Truth table, the number of unsatisfied clauses and E_p for $P = \{A \leftarrow B, D, C \leftarrow B, D \leftarrow .\}$ in lower order clauses.

S_A	S_B	S_D	S_C	$A \leftarrow B, D$	$C \leftarrow B$	$D \leftarrow$	Num. of unsatisfied clauses	E_p
-1	-1	-1	-1	1	1	-1	1	1
-1	-1	-1	1	1	1	-1	1	1
-1	-1	1	-1	1	1	1	0	0
-1	-1	1	1	1	1	1	0	0
-1	1	-1	-1	1	-1	-1	2	2

Table 2.2-2 Continued.

-1	1	-1	1	1	1	-1	1	1
-1	1	1	-1	-1	-1	1	2	2
-1	1	1	1	-1	1	1	1	1
1	-1	-1	-1	1	1	-1	1	1
1	-1	-1	1	1	1	-1	1	1
1	-1	1	-1	1	1	1	0	0
1	-1	1	1	1	1	1	0	0
1	1	-1	-1	1	-1	-1	2	2
1	1	-1	1	1	1	-1	1	1
1	1	1	-1	1	-1	1	1	1
1	1	1	1	1	1	1	0	0

An energy function is defined as:

$$H = -\frac{1}{3}\sum_i\sum_j\sum_k J_{[ijk]}S_iS_jS_k - \frac{1}{2}\sum_i\sum_j J_{[ij]}S_iS_j - \sum_i J_{[i]}S_i \tag{2.4}$$

$$\begin{aligned} H = &-\frac{1}{3}(6J_{[ABD]}S_AS_BS_D + 6J_{[ABC]}S_AS_BS_C + 6J_{[ADC]}S_AS_DS_C + 6J_{[BDC]}S_BS_DS_C) \\ &-\frac{1}{2}(2J_{[AB]}S_AS_B + 2J_{[AD]}S_AS_D + 2J_{[AC]}S_AS_C + 2J_{[BD]}S_BS_D + 2J_{[BC]}S_BS_C \\ &+2J_{[DC]}S_DS_C) - J_{[A]}S_A - J_{[B]}S_B - J_{[D]}S_D - J_{[C]}S_C \end{aligned} \tag{2.5}$$

The updating rule reads

$$S_i(t+1) = \mathrm{sgn}[h_i(t)] \tag{2.6}$$

where sgn is signum function., h_i is the input potential to neuron *i* and S_i is the state of neuron *i*. The local field is given by equation (2.7),

$$h_i(t) = \sum_j\sum_k J_{[ijk]}S_j(t)S_k(t) + \sum_j J_{[ij]}S_j(t) - J_{[i]} \tag{2.7}$$

Comparing cost function E_p with the energy function H yields the value of synaptic strengths as shown in Table 2.3.

Table 2.3: The clauses and corresponding synaptic strengths (Wan Abdullah's method) in lower order clauses.

Synaptic Strengths	Clauses			
	$A \leftarrow B, D$	$C \leftarrow B$	$D \leftarrow$	$A \leftarrow B, D \wedge C \leftarrow B \wedge D \leftarrow$
$J_{[ABD]}$	1/16	0	0	1/16
$J_{[ABC]}$	0	0	0	0
$J_{[ADC]}$	0	0	0	0

Table 2.3-2 Continued.

$J_{[BDC]}$	0	0	0	0
$J_{[AB]}$	1/8	0	0	1/8
$J_{[AD]}$	1/8	0	0	1/8
$J_{[AC]}$	0	0	0	0
$J_{[BD]}$	-1/8	0	0	-1/8
$J_{[BC]}$	0	1/4	0	1/4
$J_{[DC]}$	0	0	0	0
$J_{[A]}$	1/8	0	0	1/8
$J_{[B]}$	-1/8	-1/4	0	-3/8
$J_{[D]}$	-1/8	0	1/2	3/8
$J_{[C]}$	0	1/4	0	1/4

In order to find the global minimum energy, use equation (2.5) and substitute the value of connection strengths into the equation. Any interpretation will substituted, which is a model for the corresponding logic program into the equation.

For example, take $S_A = S_B = S_D = S_C = 1$ and substitute into the equation.

$$H_{\min} = -\frac{7}{8}$$

So, global minimum energy for this example is -7/8.

2.1.2 Logic programming in higher order Hopfield network

However, higher order Hopfield network need to be carried out for further works to solve the optimization problems. Thus a work must be carried out by using higher order Hopfield network. It is similar to lower order logic programming that restricted to three order clauses (Saratha, 2006). For higher order, will restricted up to fifth order clauses only because the system is not stable and oscillates (providing more local minimum values) for order higher than fifth order.

For an example for higher order clause, consider the logic program below:

$$P = A \leftarrow B, C, D, E$$

$$\Lambda \; A \leftarrow B,\, C,\, D$$

$$\Lambda \; A \leftarrow B,\, C$$

$$\Lambda \; A \leftarrow B$$

$$\Lambda \; A \leftarrow \qquad (2.8)$$

where those clauses can be translated as $A \vee \neg B \vee \neg C \vee \neg D \vee \neg E$, $A \vee \neg B \vee \neg C \vee \neg D$, $A \vee \neg B \vee \neg C$, $A \vee \neg B$ and A. By using Wan Abdullah's method, a cost function will derived based on associated of the negation of all clauses like $\frac{1}{2}(1+S_A)$ representing the logic value of neuron A, while S_A is the state of neuron corresponding to A. S_A is defined as value 1 if is true, -1 if is false. If the neuron A does not occur (negation), $\frac{1}{2}(1-S_A)$ is the clause that represented it. A conjunction connection is represented by multiplication while disjunction connection is addition.

$$E_p = \frac{1}{2}(1-S_A)\frac{1}{2}(1+S_B)\frac{1}{2}(1+S_C)\frac{1}{2}(1+S_D)\frac{1}{2}(1+S_E)+\frac{1}{2}(1-S_A)\frac{1}{2}(1+S_B)\frac{1}{2}(1+S_C)\frac{1}{2}(1+S_D) + \frac{1}{2}(1-S_A)\frac{1}{2}(1+S_B)\frac{1}{2}(1+S_C)+\frac{1}{2}(1-S_A)\frac{1}{2}(1+S_B)+\frac{1}{2}(1-S_A) \qquad (2.9)$$

Table 2.4. Truth table, the number of unsatisfied clauses and E_p for P = { A ← B, C, D, E, A ← B, C, D, A ← B, C, A ← B, A ← } in higher order clauses. It shows some parts of truth table.

S_A	S_B	S_C	S_D	S_E	A←B,C ,D, E	A←B,C, D	A←B , C	A←B	A ←	Num. of unsatisfied clauses	E_p
-1	-1	-1	-1	-1	1	1	1	1	-1	1	1
-1	-1	-1	-1	1	1	1	1	1	-1	1	1
-1	-1	-1	1	-1	1	1	1	1	-1	1	1
-1	-1	1	-1	-1	1	1	1	1	-1	1	1
-1	1	-1	-1	-1	1	1	1	-1	-1	2	2
1	-1	-1	-1	-1	1	1	1	1	1	0	0
-1	-1	-1	1	1	1	1	1	1	-1	1	1
-1	-1	1	1	-1	1	1	1	1	-1	1	1
-1	1	1	-1	-1	1	1	-1	-1	-1	3	3
1	1	-1	-1	-1	1	1	1	1	1	0	0
-1	-1	1	1	1	1	1	1	1	-1	1	1
-1	1	1	1	-1	1	-1	-1	-1	-1	4	4
1	1	1	-1	-1	1	1	1	1	1	0	0