Pertanika J. Sci. & Technol. 28 (2): 545 - 564 (2020)

SCIENCE & TECHNOLOGY

Journal homepage: http://www.pertanika.upm.edu.my/

Metaheuristics Approach for Maximum *k* Satisfiability in Restricted Neural Symbolic Integration

Saratha Sathasivam[1], Mustafa Mamat[2], Mohd Shareduwan Mohd Kasihmuddin[1]* and Mohd. Asyraf Mansor[3]

[1]School of Mathematical Sciences, Universiti Sains Malaysia, 11800 USM, Penang, Malaysia
[2]Faculty of Informatics and Computing, Universiti Sultan Zainal Abidin, 21300 UniSZA, Kuala Terengganu, Terengganu, Malaysia
[3]School of Distance Education, Universiti Sains Malaysia, 11800 USM, Penang, Malaysia

ABSTRACT

Maximum *k* Satisfiability logical rule (MAX-*k*SAT) is a language that bridges real life application to neural network optimization. MAX-*k*SAT is an interesting paradigm because the outcome of this logical rule is always negative/false. Hopfield Neural Network (HNN) is a type of neural network that finds the solution based on energy minimization. Interesting intelligent behavior has been observed when the logical rule is embedded in HNN. Increasing the storage capacity during the learning phase of HNN has been a challenging problem for most neural network researchers. Development of Metaheuristics algorithms has been crucial in optimizing the learning phase of Neural Network. The most celebrated metaheuristics model is Genetic Algorithm (GA). GA consists of several important operators that emphasize on solution improvement. Although GA has been reported to optimize logic programming in HNN, the learning complexity increases as the number of clauses increases. GA is more likely to be trapped in suboptimal fitness as the number of clauses increases. In this paper, metaheuristic algorithm namely Artificial Bee Colony (ABC) were proposed in learning MAX-*k*SAT programming. ABC is swarm-based metaheuristics that capitalized the capability of Employed Bee, Onlooker Bee, and Scout Bee. To this end, all the learning models were tested in a new restricted learning environment. Experimental results obtained from the computer simulation demonstrate the effectiveness of ABC in modelling MAX-*k*SAT.

Keywords: Artificial bee colony, exhaustive search method, genetic algorithm, Hopfield neural network, maximum *k* satisfiability

ARTICLE INFO

Article history:
Received: 16 April 2019
Accepted: 19 July 2019
Published: 15 April 2020

E-mail addresses:
saratha@usm.my (Saratha Sathasivam)
must@unisza.edu.my (Mustafa Mamat)
shareduwan@usm.my (Mohd Shareduwan Mohd Kasihmuddin)
asyrafman@usm.my (Mohd. Asyraf Mansor)
* Corresponding author

ISSN: 0128-7680
e-ISSN: 2231-8526

© Universiti Putra Malaysia Press

INTRODUCTION

In the past decades, Boolean Satisfiability (SAT) has become a popular subject in artificial intelligence (AI) that attracted researchers from various field of studies. There are basically two reasons. First, the SAT is a direct transformation from real life application to mathematical formulation. In that sense, SAT serves as a foundation for more real-life applications such as Very Large Scale Integrated (VLSI) system (Mansor et al., 2016a), neural network (Kasihmuddin et al., 2018a), pattern recognition, logic mining and knowledge based paradigm. Second, SAT is a foundation to the various algorithm because interestingly, there are no efficient algorithms to comply with NP problem compared to P problem (Rojas, 2013). Hence, researchers in this field always find the approximation algorithm to comprehend SAT problem without the need of mathematical complexity. The mentioned reasons motivate the researcher (Kasihmuddin et al., 2017) to incorporate SAT with other AI applications. Applications pertaining to the hybrid SAT structure can be used to solve various on-demand applications such as scheduling and optimization problem. Such realization leads to the creation of a more effective algorithm to satisfy more variant of SAT program (Asirelli et al., 1985). Inspired by the extended version of Boolean SAT, Maximum k satisfiability (MAX-kSAT) starts to gain popularity in the heart of researcher because MAX-kSAT utilized false/negative output compared to other SAT representation (Poloczek et al., 2017). MAX-kSAT is commonly known as a logical rule that allocates symbolic binary/bipolar value to a Boolean variable with k literals for each neuron that satisfies the maximum number of clauses (Lynce et al., 2018).

Execution of the Artificial neural network (ANN) in AI is to acquire knowledge and use that information to model the intelligent system that can solve important problems. Hopfield Neural Network (HNN) is a dynamical neural system which possesses a memory that is associative and consists of interconnected neurons (Layeb, 2012). All neurons in HNN work in a dynamical manner with pre-defined threshold to mimic the actual human brain mechanism (Hopfield, 1982). The vital characteristics of HNN are the energy minimization via Lyapunov energy. Several NP problems such as travelling salesman problem (Mérida-Casermeiro et al., 2001), scheduling problem (Liang & Hsu, 1996), N Queen (Ohta, 2002) represent the state of a neuron as a possible optimal solution. In this case, the neuron state will be "excited" via pre-determined local field and synaptic weight will be updated via Hebbian learning. Interestingly, the neurons will iterate until HNN converges to minimum energy (possible desired solution). Abdullah (1992) and Sathasivam (2006) introduced the merger between two different disciplines by implementing HornSAT in HNN. In these studies, HornSAT was converted to Boolean Algebra and the synaptic weight of the network was obtained by comparing cost function and Lypunov energy function. The most interesting insight from these mergers was, there was a fixed energy value for every satisfied clause. This is due to the property of HornSAT that is always satisfiable. More specifically, the sum of all Lyapunov energy value becomes the absolute minimum energy for logic programming in HNN. Based on this paradigm, several researchers extend the usage of logic programming in the neural network. Hamadneh et al. (2012) proposed logic programming in Radial Basis Function Neural Network (RBFNN). The proposed network utilized HornSAT and satisfiable clauses in RBFNN. The usage of different SAT representation had been extended to *k*SAT (Mansor et al., 2016b). The primary motivation of this extension is the number of variables inside any clause is always $k \leq 3$ (Kullmann, 1999). After the introduction of *k*SAT, this representation has been a prominent logical rule in HNN. Several recent studies indicate that *k*SAT is compatible in doing pattern Satisfiability (Mansor et al., 2016a), and very large-scale integration circuit modelling (Mansor et al, 2018). The usage of *k*SAT in HNN also has been extended to another hybrid HNN model such as kernel HNN (Alzaeemi & Sathasivam, 2018). All the mentioned HNN-*k*SAT model only focus on satisfiable logic programming. Recently, the first attempt in representing non-satisfiable logic programming in HNN has been studied by Kasihmuddin et al. (2018b). The proposed model utilized Maximum *k* Satisfiability in doing HNN. The proposed merger created a new horizon in finding the global minimum solution although the final logical outcome was negative.

In another development, a variant of logic programming in HNN has been explored by several paradigms such as pattern reconstruction and circuit verifications. This suggests an obvious question: could learning phase of HNN be a learning environment with predetermined constraints, so that, under suitable condition, the learning model of HNN must fulfill the certain learning constraints? This question has been positively discussed when Sathasivam and Ng (2013) proposed agent-based modelling (ABM) to simulate the environment of logic programming in HNN. Each factor that affects the interaction among "agents" is examined by using ABM. Mansor et al. (2016b) proposed VLSI circuit configuration by using *k*SAT in HNN. The proposed method created SAT environment based on the circuit configuration which consisted of millions of transistors. Mansor et al. (2016a) proposed pattern SAT by embedding *k*SAT inside some square matrices. This finding led to a solid foundation for pattern recognition via *k*SAT. By introducing environmental constraints, the various model could be constructed or tested during the learning phase of HNN.

Artificial bee colony (ABC) has been increasingly viewed as an optimization technique for continuous problem (Karaboga, 2005). Karaboga and Basturk (2007) conducted a comprehensive study to compare the effectiveness of ABC with other existing metaheuristics algorithm such as Genetic Algorithm (GA), Particle Swarm optimization (PSO) and Differential Evolution (DE). The simulation results illustrated that ABC had the best performance compared to other metaheuristic algorithms. Several advancements were implemented to improve the accuracy of ABC (Karaboga, 2009; Banitalebi et al., 2015). In another perspective, the usage of binary ABC has been prominent in solving the constraint optimization problem. Ozturk et al. (2015) proposed binary ABC for solving 0-1 knapsack problem by intelligently adopting genetic operators. Kashan et al. (2012) proposed a binary ABC by replacing vector subtraction operator from original ABC algorithm with differential expression (Pamparà & Engelbrecht, 2011). The proposed expression employed a measure of dissimilarity between binary vectors. Another important study in binary ABC was done by Jia et al. (2014). This study capitalized bitwise operation to portray the movement of employed bee and onlooker bee. The proposed ABC algorithm has been extended to HNN-*k*SAT (Kasihmuddin et al., 2016) where the hybrid network was able to achieve more than 95% of global minima ratio with reasonable computation time. Unfortunately, global minimum ratio and computational time show very little, the effectiveness of ABC in HNN-*k*SAT. In this study, ABC was a learning model for clausal checking in HNN-MAX*k*SAT in a new simulated learning environment. The results showed that ABC displayed the best performance for all performance metric in the restricted learning environment.

MATERIALS AND METHOD

Maximum *k* Satisfiability

Karp (1972) had elaborated the concept of MAX-*k*SAT as the generalized variant of Boolean satisfiability logical rule structure as compared to the satisfiable logic, namely *k* Satisfiability (*k*SAT). According to Chu et al. (2019), MAX-*k*SAT is a complex and well-known variant of NP-hard problem, commonly being leveraged in various applications such as in digital circuit fault detection and encoding various engineering problems. Thus, MAX-*k*SAT has diversified the propositional Boolean Satisfiability logic variant in term of finding the optimal interpretation that contributes into negative outcome (Mansor et al., 2017). Therefore, the general definition of MAX-*k*SAT is given as follows:

Definition 1.1 (Maximum *k* Satisfiability)

Given a Boolean conjunctive normal form (CNF), the Maximum *k* Satisfiability problem can be demarcated as searching the interpretation that maximizes the number of satisfied unit of clauses for a particular Boolean MAX-*k*SAT formula.

Similarly, the MAX-*k*SAT is constructed as a logical rule in CNF with n clauses and k variable each. Zhang et al. (2003) had defined the structure of MAX-*k*SAT as a pair of (η, α) given α is the combination of the possible bipolar interpretation, $\{1, -1\}^n$. In addition, α is a mapping $\eta \to Z$ which refers to the score of the interpretations where Z is scored depending on a particular satisfied clauses. Hence, MAX-*k*SAT representation comprises identifying the best bipolar string assignments in $P_{MAX\text{-}kSAT}$ that at the same time satisfied at least h clauses out of m clauses. In the case of MAX-*k*SAT, the condition will be strictly $h < m$. The modified MAX-*k*SAT formula for $k = 2$ has been coined by Kasihmuddin et al. (2018a):

$$P_{MAX-kSAT} = (A \vee B) \wedge (A \vee \neg B) \wedge (\neg A \vee B) \wedge (\neg A \vee \neg B) \wedge (C \vee D) \quad (1)$$

where $k = 2$ denotes the number of literals strictly in a particular clause.

According to Zhang et al. (2003), there are 2^n possible bipolar interpretations for a particular MAX-*k*SAT problem, whereby n denotes the number of literals. Specifically, Equation (1) has no complete interpretation that make $P_{MAX\text{-}kSAT}$ to become true or fully satisfiable. The computation of the fitness for $P_{MAX\text{-}kSAT}$ can be done by using Equation (2).

$$f_{P_{MAX-kSAT}} = \sum_{i=1}^{NC} C_i \quad (2)$$

where NC denotes the number of the clause and C_i is the number of satisfied MAX-*k*SAT clause. A point to ponder for $P_{MAX\text{-}kSAT}$ is the fitness value that will never attain the maximum number of the clause due to existence of the falsified clauses. Henceforth, MAX-*k*SAT will consider the minimum number of falsified clauses in a complete interpretation. In this paper, the MAX-*k*SAT logic programming is carried out in restricted learning in HNN. Since, the MAX-*k*SAT works exceptionally well with the conventional learning in HNN (Kasihmuddin et al., 2018b), the impact of restricted learning will be investigated extensively in this work. In fact, the MAX-*k*SAT logic programming is chosen due to the negative outcome produced as compared with the *k*SAT programming. Therefore, the real-life problem involving the negative outcomes can be encoded in the form of MAX-*k*SAT to be further extracted by the data mining algorithm.

Hopfield Neural Network

HNN is broadly employed to store and process the patterns due to the capability of its content addressable memory (CAM). In particular, HNN is a class of dynamic recurrent network with symmetrical connected weight corresponds to the interconnected units emulated the biological human brain. The HNN is considered due to a few edges over the other variant of recurrent or feedforward neural network. It comprises good characteristics namely parallel computation, fast convergence and acceptable capacity of the CAM (Hopfield, 1982). Based on the architecture of HNN standpoint, HNN comprises nterconnected units called neurons. Hence, the neuron state in HNN is denoted as $S_i(t)$ where $i = 1, 2, ...N$. Consequently, the bipolar neuron combinations in HNN is well represented as $S_i \in \{-1,1\}$. In this work, the state will updated asynchronously per execution.

The excitation of the neuron in HNN can be represented mathematically as in S_i.

$$S_i = \begin{cases} 1 & if \sum_j W_{ij} S_j \geq \xi \\ -1 & Otherwise \end{cases} \tag{3}$$

where W_{ij} is the weight for unit j to i and ξ refers to the threshold of the HNN. The implementation of MAX-*k*SAT in HNN is denoted as HNN-MAX*k*SAT. In this case, HNN-MAX*k*SAT will consider the *k* neurons per clause. The local field is prominent to properly squash the retrieved output before generating the final state. Moreover, the local field formulation for $k = 3$ is shown in Equation (4), whereas for $k = 2$ is given in Equation (5) (Sathasivam et al., 2011).

$$h_i = \sum_{k=1, i \neq j \neq k}^{N} W_{ijk}^{(3)} S_j S_k + \sum_{j=1, i \neq j \neq k}^{N} W_{ij}^{(2)} S_j + W_i^{(1)}, \quad k = 3 \tag{4}$$

$$h_i = \sum_{j=1, i \neq j \neq k}^{N} W_{ij}^{(2)} S_j + W_i^{(1)}, \quad k = 2 \tag{5}$$

where *i* and *j* are corresponded to neurons *N*. These local fields determine the effectiveness and variability of the final states obtained by HNN. Thus, the generated final interpretation classifies whether the solution is overfit or not. Precisely, the updating rule is given as

$$S_i(t+1) = sgn[h_i(t)] \tag{6}$$

The relation is limited to be symmetric and zero diagonal $W_{ij}^{(2)} = W_{ji}^{(2)}$, $W_{ii}^{(2)} = W_{jj}^{(2)} = W_{kk}^{(2)} = W_{iii}^{(3)} = W_{jjj}^{(3)} = W_{kkk}^{(3)} = 0$ which further derive and formulate the final energy of respective variant of HNN-MAX*k*SAT as given:

$$H_{P_{MAX2SAT}} = -\frac{1}{2} \sum_{i=1, i \neq j}^{N} \sum_{j=1, i \neq j}^{N} W_{ij}^{(2)} S_i S_j - \sum_{i=1}^{N} W_i^{(1)} S_j, \quad k = 2 \tag{7}$$

$$H_{P_{MAX3SAT}} = -\frac{1}{3} \sum_{i=1, i \neq j \neq k}^{N} \sum_{j=1, i \neq j \neq k}^{N} \sum_{k=1, i \neq j \neq k}^{N} W_{ijk}^{(3)} S_i S_j S_k - \frac{1}{2} \sum_{i=1, i \neq j}^{N} \sum_{j=1, i \neq j}^{N} W_{ij}^{(2)} S_i S_j - \sum_{i=1}^{N} W_{(i=1)}^{(1)} S_j, \quad k = 3 \tag{8}$$

Therefore, for the cumulative cases, all permutations that involves i , j, and k for MAX*k*SAT clauses are considered. Ultimately, the final energy recorded by HNN-MAX*k*SAT is always stable (Zhang et al., 2016) and reduces with the dynamics (Sathasivam, 2008).

Restricted Learning in HNN-MAX*k*SAT

The ability of HNN-MAX*k*SAT to adapt to change in its environment provide vital insight into the effectiveness of the learning model. In this section, restricted learning paradigm was implemented to HNN-MAX*k*SAT for the first time. During learning phase, the initial neuron state of S_i that represents the variable in MAX*k*SAT is given by

$$S_i = \{S_1, S_2, S_3, S_4, \ldots, S_N\} \tag{9}$$

By examining the inconsistencies of the MAX-*k*SAT logical rule, the learned neuron assignment must minimize the cost function $E_{P_{MAXkSAT}}$.

$$min[E_{P_{MAXkSAT}}] \tag{10}$$

where $E_{P_{MAXkSAT}} \neq 0$ for all MAX-*k*SAT clauses. The neuron state will be updated based on the following condition

$$S_i = \begin{cases} S_i : S_i \in \{-1,1\}, i \in I\}, & NH \leq \Omega \\ S_i^{new}, & NH > \Omega \end{cases} \tag{11}$$

where I is an index set and *NH*, $\Omega \in I$. *NH* and Ω are defined as learning iteration and maximum iteration reprectively (Xu et al., 2019). In other words, the proposed HNN-MAX*k*SAT models will search the correct interpretation until $NH > \Omega$. The new state of S_i^{new} emerges and proceeds to retrieval phase of HNN. This simulated learning environment is completely different than HNN proposed in Sathasivam (2010) and Mansor et al. (2017). The learning iteration of HNN in the mentioned work is increased indefnitely $\Omega \rightarrow \infty$ until $E_{P_{MAXkSAT}}$ reached the desired minimum value. The restricted learning paradigm of HNN-MAX*k*SAT is defined as RHNN-MAX*k*SAT models. Figure 1 shows the implementation of *k*SAT programming in HNN in restricted learning environment.

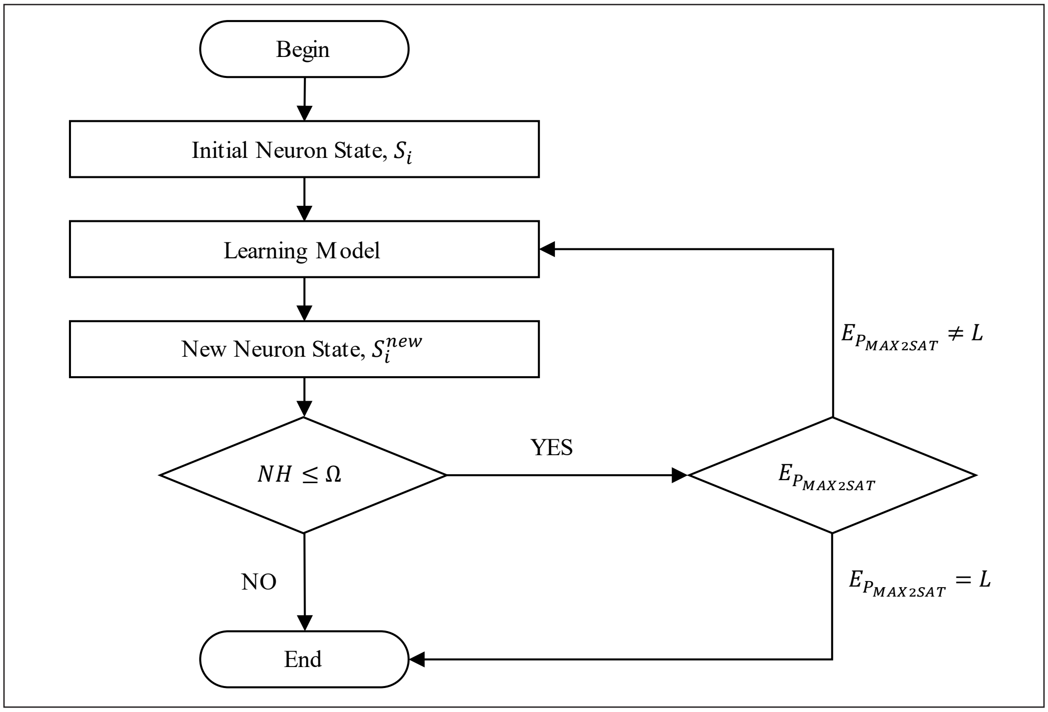

Figure 1. Restricted Learning Environment

Genetic Algorithm

Genetic algorithm (GA) is popular state of the art metaheuristic algorithm that reduce the burden of the computation in optimization problem without usage of complex Mathematical equation. It started with Golberg and Holland (1988) that had meticulously proven the idea of solution improvement in every iteration. Kasihmuddin et al. (2016) proposed an HNN embedded with *k*SAT system integrated with GA during the learning phase. The implementation of GA during learning phase of HNN-MAX*k*SAT is defined as RHNN-MAX*k*SATGA. Bipolar strings in this particular case is representing the possible satisfied assignments of RHNN-MAX*k*SATGA. The stages involved in RHNN-MAX*k*SATGA are as follows:

Stage 1: Initialization. 100 bipolar string will be generated where each element of $\{1, -1\}$ is denoted by True and False.

Stage 2: Fitness Evaluation. Bipolar string from stage 1 will be evaluated based on the following equation:

$$f_{MAX-kSAT} = max[C_1(x) + C_2(x) + C_3(x) \ldots\ldots + C_N(x)] \qquad (12)$$

where $C_1, C_2, C_3 \ldots\ldots C_N$ are the clause verified using by GA and N represents the clause number depicted in the formula. The choice of fitness function in Equation (12) is crucial to avoid the possible local maxima due the floating number produced during iteration.

Stage 3: Selection. Ten (10) bipolar strings that acquire the highest number of satisfied clauses will be selected. The selection process will dismiss potential non-fit MAX-*k*SAT solution.

Stage 4: Bipolar Crossover. The exchange of information between two sub-structure of bipolar string occurred randomly. The position of the crossover will be selected randomly. The main purpose of crossover is to diversify the potential fitness of the offspring.

Prior to crossover
Bipolar string X = -1 1 -1 -1 -1 -1 -1 -1
Bipolar string Y = 1 -1 -1 -1 1 1 1 -1

Post crossover
Bipolar string X = 1 1 -1 -1 -1 -1 -1 -1
Bipolar string Y = 1 -1 -1 -1 1 -1 1 -1

Stage 5: Mutation. Mutation includes state exchange from 1 to -1 or -1 to 1. As a general point, mutation will potentially increase the average fitness of the whole solution and reduce the fitness of the low fit bipolar string. Stage 1 to 5 is repeated for until it reaches predetermined number of generations.

Artificial Bee Colony

ABC is a well-known swarm optimization method in finding near optimal solution. Since inconsistencies of bipolar string can be easily described as a cost function of MAX-*k*SAT logic, the perspective of ABC shifted toward bipolar optimization (Ning et al., 2018). In this case, the integration of ABC during the learning phase of HNN is abbreviated as RHNN-MAX*k*SATABC. The bipolar string is represented as a food source and bees were entrusted to locate the optimal food source (Karaboga & Basturk, 2008). The three optimization layers in ABC namely employed bees, onlooker bees and scout bee will explore the global solution of the search space (Zhang & Zhang, 2017). During the learning phase, the fittest bee is the one with the highest fitness value. The main stages of ABC in RHNN-MAX*k*SATABC is as follows:

Stage 1: Initialization. 50 employed bees, 50 onlooker bees and 1 scout bee are initialized. Each bee carries bipolar string of MAX-*k*SAT which is denoted by True and False.

Stage 2: Verification of Fitness. The fitness of each bee in Stage 1 (except for scout bee) will be evaluated based on the Equation (2).

Stage 3: Employed Bee Stage. In this stage, employed bee will identify new food position for $v_{ij}^{Employed}$ (bipolar string) in a given neighbourhood. The location of the food is given as follows

$$v_{ij}^{Employed} = y_{ij} \vee \left(\phi_{ij} \otimes (y_{ij} \wedge y_{kj})\right) \tag{13}$$

where

y_{ij} food source at initial stage

y_{kj} food source that is observed

ϕ_{ij} variables where,

$$\phi_{ij} = \begin{cases} 1, rand(0,1) < 0.5 \\ -1, rand(0,1) \geq 0.5 \end{cases}$$

$\otimes$ is a 'XOR' operator

$\wedge$ is an 'AND' operator

$\vee$ is an 'OR' operator

Stage 4: Onlooker Bee Stage. Onlooker bees were selected food source based on the fitness of the employed bees in stage 3. The new position of the food source is based on roulette wheel selection (RWS) (Goldberg & Deb, 1991). The probability model for information exchange is given as

$$p_i^{Onlooker} = \frac{f_{v_{ij}^{Employed}}}{\sum_{i=1}^{SN} f_{v_{ij}^{Employed}}} \tag{14}$$

where $\sum_{i=1}^{SN} f_{v_{ij}^{Employed}}$ portrays targeted RHNN-MAX*k*SAT fitness and *SN* shows the bee's group count. Similar to Equation (13), onlooker bees are seeked for the closest food origin by using the following equation

$$v_{ij}^{Onlooker} = y_{ij} \vee \left(\phi_{ij} \otimes \left(y_{ij} \wedge y_{kj}\right)\right) \tag{15}$$

where all the variable in Equation (15) are similar with the information given in Equation (13). Stage 1 until Stage 4 is repeated until the pre-determined trials.

Stage 5: Scout Bee Stage. If the position of the food for employed bees cannot be improved through the number of trials, scout bee abandons the current food source. Bipolar state in scout bee will be randomly generated. If the food source obtained $f_{v_{ij}^{\text{Onlooker}}} = f_{NC}$ or $f_{v_{ij}^{Employed}} = f_{NC}$, the best bipolar assignment is outputted.

Implementation RHNN-MAX*k*SAT Model

The robustness of the learning method is a very critical criterion of any given network. Worth mentioning that, the earliest celebrated optimization learning model in HNN was proposed by Sathasivam (2006) and Sathasivam (2010). The mentioned paper proposed Exhaustive Search (ES) in finding the correct HornSAT interpretation during the learning phase of HNN. In this case, ES is a conventional method during the learning phase of RHNN-MAX*k*SAT. The learning phase of all RHNN-MAX*k*SAT models is used to derive the optimal cost function by maximizing the number of satisfied clauses in MAX*k*SAT. Hence, the main task of the proposed network is to create a “model” that behave according to MAX*k*SAT logical rule. The following algorithm shows the implementation of RHNN-MAX*k*SAT models:

1. Transform MAX-*k*SAT clauses to Boolean algebra (if applicable).
2. Neurons is assigned to respective variable in MAX-*k*SAT clauses.
3. By defining the inconsistencies of MAX-*k*SAT, derive the cost function by assigning $X = \frac{1}{2}(1 + S_X)$ and $\bar{X} = \frac{1}{2}(1 - S_X)$. The neuron’s state shows true if $S_X = 1$ and false if $S_X = -1$. In this case, variable inside each clause is connected with addition $\vee$ and the overall clause is connected by multiplication ($\wedge$).
4. Bipolar assignment that minimizes the cost function will be obtained via ES, GA and ABC. The proposed learning model will exit the learning loop if $NH \geq \Omega$.
5. Obtain the synaptic weight matrix of the HNN model corresponds –MAX*k*SAT logical rule.
6. Compute the lowest minimum energy of –MAX*k*SAT by using Equation (7) and Equation (8).
7. Compute the final neuron state via Equation (4) and (5).
8. By using Equation (6) and Equation (7), calculate the final energy of the neuron state in step 7.

In order to obtain a fair comparison among all RHNN-MAX*k*SAT models, all source code is implemented via Microsoft Visual Basic C++ 2013 for Windows 10. Similar device is used in every simulation to avoid the possible bad sector. Table 1 to Table 3 show all the parameters involved in each RHNN-MAX*k*SAT models.

Table 1
List of Parameters in RHNN-MAXkSATES

Parameter	Parameter value
Neuron Combination	100
Tolerance Value (*Tol*)	0.001
Ω	10^3 and 10^5
No_String	100
Selection_Rate	0.1

Table 2
List of Parameters in RHNN-MAXkSATGA

Parameter	Parameter value
Neuron Combination	100
Tolerance Value (*Tol*)	0.001
Ω	10^3 and 10^5
No_Chromosomes	100
Selection_Rate	0.1
Mutation_rate	0.01
Generation	1000

Table 3
List of Parameters in RHNN-MAXkSATABC

Parameter	Parameter value
Neuron Combination	100
Tolerance Value (*Tol*)	0.001
Ω	10^3 and 10^5
No_Employed	50
No_Onlooker	50
No_Scout	1
Trial	10
Limit	100

RESULT AND DISCUSSION

Compared to previous HNN model such as Kasihmuddin et al. (2017), this experiment has been proposed in a restricted learning environment. In relation with several studies done by Cai et al (2016), all the proposed RHNN-MAX*k*SAT model were tested up to 400 variables. The learning iteration for all proposed models had been restricted to iterate up to $\Omega = 10^3$ and $\Omega = 10^5$.

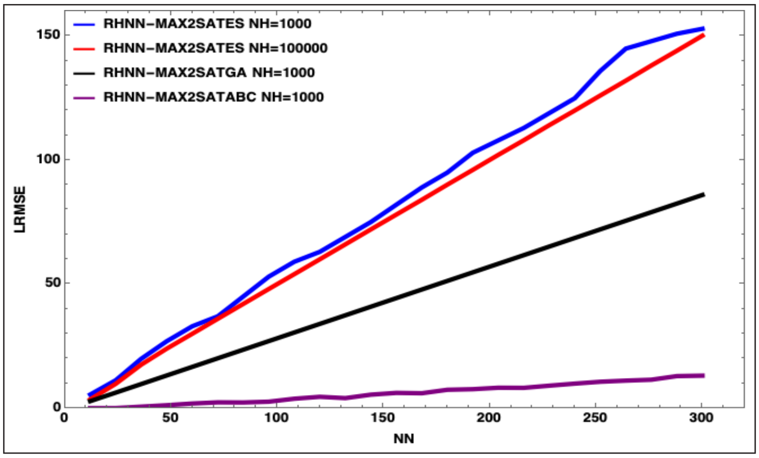

Figure 2. Root Mean Square Error (RMSE) of RHNN-MAX*2*SAT models

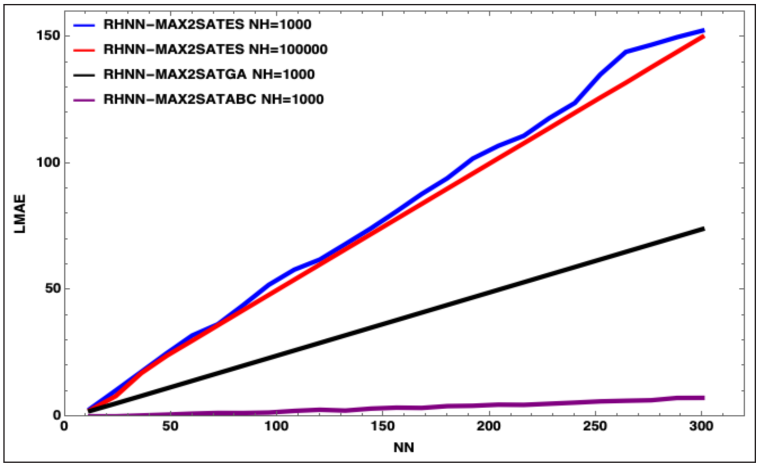

Figure 4. Mean Absolute Error (MAE) of RHNN-MAX*2*SAT models

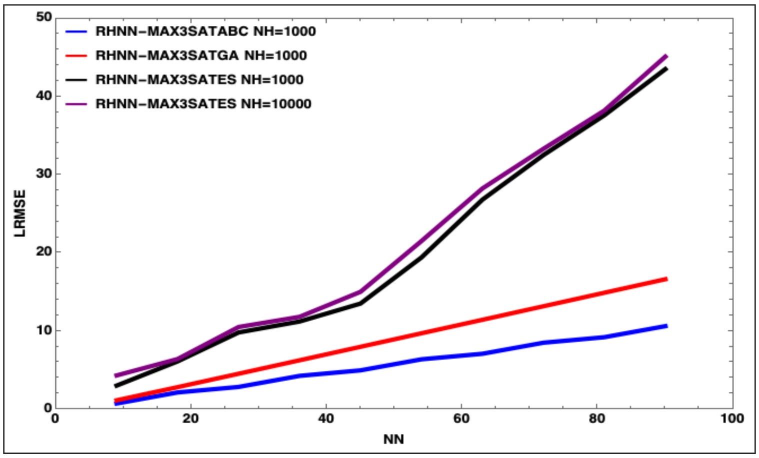

Figure 3. Root Mean Square Error (RMSE) of RHNN-MAX*3*SAT models

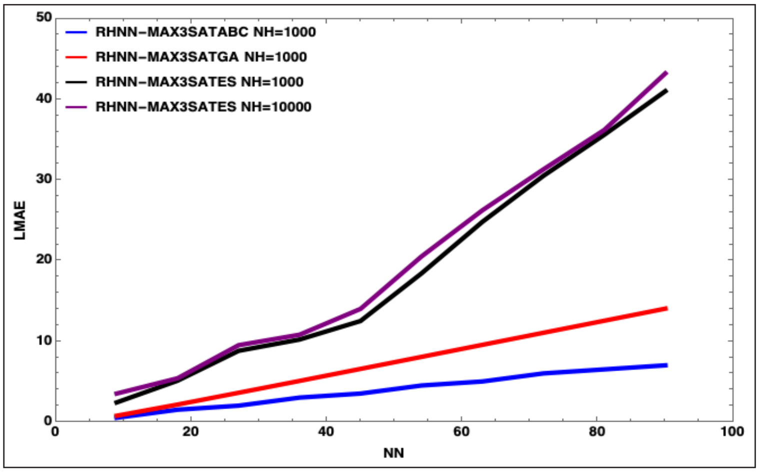

Figure 5. Mean Absolute Error (MAE) of RHNN-MAX*3*SAT models

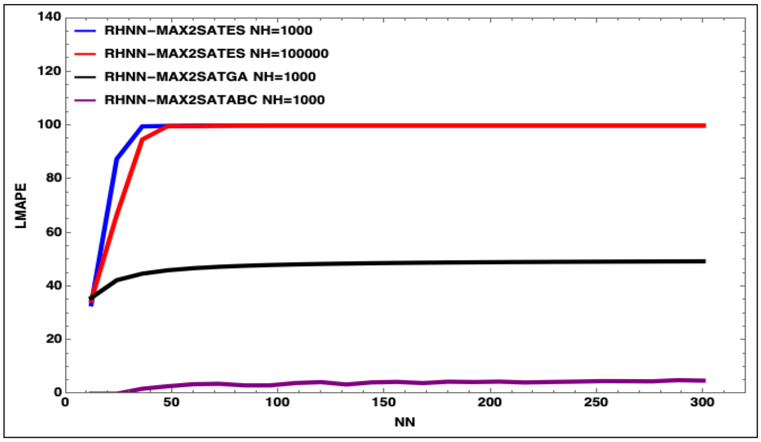

Figure 6. Mean Absolute Percentage Error (MAPE) of RHNN-MAX*2*SAT models

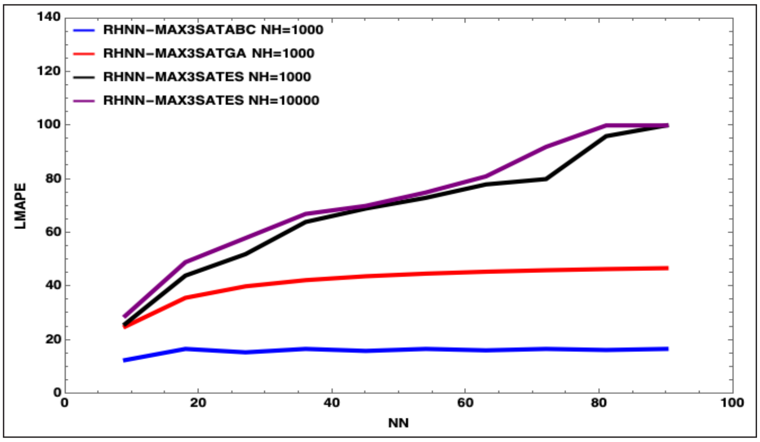

Figure 7. Mean Absolute Percentage Error (MAPE) of RHNN-MAX*2*SAT models

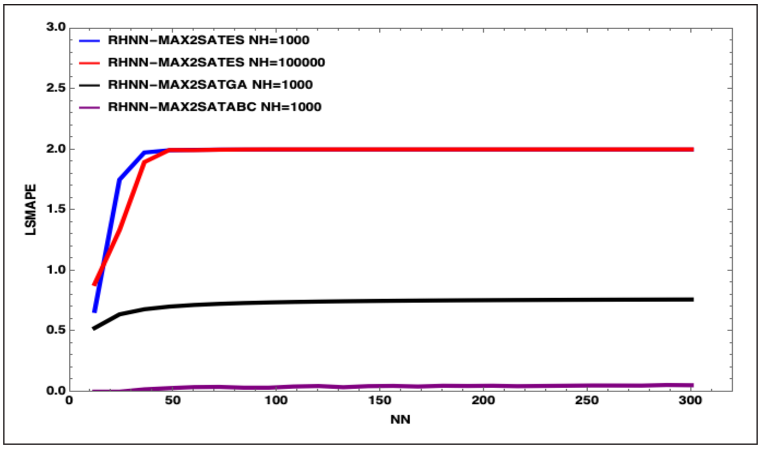

Figure 8. Symmetric Mean Absolute Percentage Error (SMAPE) of RHNN-MAX*2*SAT models

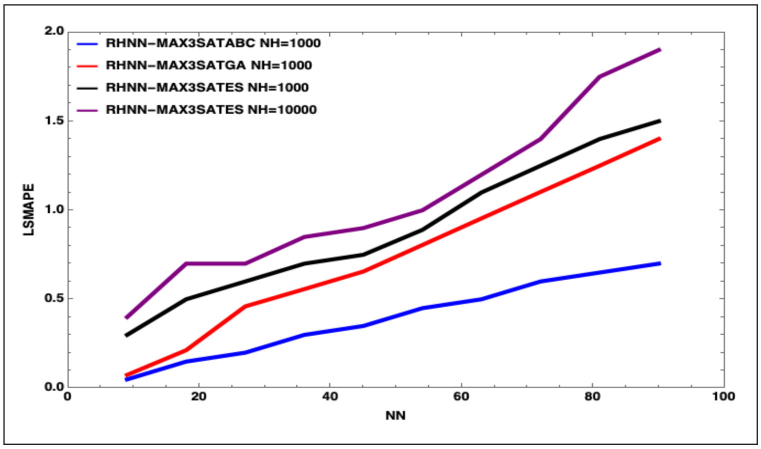

Figure 9. Symmetric Mean Absolute Percentage Error (SMAPE) of RHNN-MAX*2*SAT models

Table 4
Zm of RHNN-MAX2SAT models

NN	RHNN-MAX2SATES	RHNN-MAX2SATGA	RHNN-MAX2SATABC
12	1	1	1
24	0.99	1	1
36	0.9925	1	1
48	0.812	1	1
60	0.657	1	1
72	0.3158	1	1
84	0.007	1	1
96	0	1	1
108	0	1	1
120	0	1	1

Table 5
Zm of RHNN-MAX3SAT models

NN	RHNN-MAX3SATES	RHNN-MAX3SATGA	RHNN-MAX3SATABC
9	0.9984	1	1
18	0.7551	1	1
27	0.6458	1	1
36	0.5626	1	1
45	0.3503	1	1
54	0.1867	1	1
63	0.1414	1	1
72	0.0322	1	1
81	0.0065	0.9999	1
90	0	0.9985	1

Figure 2, Figure 3, Figure 4, Figure 5 and Table 4 demonstrate the value of RMSE, MAE, MAPE, SMAPE and *Zm* respectively for all RHNN-MAX*k*SAT models. Learning errors (RMSE, MAE, SMAPE and *Zm*) are the benchmark for accuracy and *Zm* is a benchmark for the feasibility of the RHNN-MAX*k*SAT model. The result is very significant because the successful implementation of RHNN-MAX*k*SAT model shows the HNN system is adaptable to –MAX*k*SAT logical rule. Worth mentioning that, the final outcome of the network is negative and the induced final state were expected to achieve the maximum number of satisfied clauses. The result in Figure 2 to Figure 9 and Table 4 and Table 5 allow the following observations:

1. RHNN-MAX*k*SATABC provides the best result in terms of RMSE, MAE, MAPE, and SMAPE. RHNN-MAX*k*SATES is only capable of managing a smaller number of clauses.
2. RHNN-MAX*k*SATGA requires more iteration to develop the fitness of bipolar string before effective crossover could take place. This is due to a large number of the non-fit bipolar string during an early stage of RHNN-MAX*k*SATGA.
3. RHNN-MAX*k*SATABC is reported to obtain the most consistent bipolar string during the learning phase. Interaction and exchange of information between employed bee and onlooker bee by using Equation (13) to Equation (15) reduce the possibility of the network to reach the scout bee phase.
4. After *NN* = 20, the final neuron state in RHNN-MAX*k*SATES is approaching maximum metric error. In this case, the learning phase was trapped in trial and error state.

RHNN-MAX*k*SATABC has the best value of *Zm* (approaching 1) compared to the other learning model. It was observed (refer Table 4 and 5) that more than 95% of the final state of the neuron in RHNN-MAX*k*SATABC and RHNN-MAX*k*SATGA achieved the global minimum solution. It is likely due to the learning rate acquired by both models increase the storage capacity of RHNN-MAX*k*SAT. The observation can be further explained in Sathasivam (2010) where higher relaxation time during the learning phase will increase the value of *Zm*. In another perspective, the value for the ratio of satisfied clauses (RSC) for RHNN-MAX*k*SATGA and RHNN-MAX*k*SATABC are consistently 0.857143 for $k = 2$ and 0.9091 for $k = 3$ restricted learning environment. All RSC values have good agreement with the analytical study done by Paul et al. (2016).

CONCLUSION

In this paper, three hybrid learning models in doing –MAX*k*SAT were proposed. All the proposed hybrid networks were tested in a restricted environment where $NH \leq \Omega$. On the basis of results obtained by simulation, RHNN-MAX*k*SATABC is the best network compared to other RHNN-MAX*k*SAT models. On the other hand, the integrated approaches proposed here provide a few options that can help the neural network deal with a false or negative outcome. It suggests that there are countless real-life applications that give significance to the negative result. The proposed method is a solid foundation to other SAT representation such as Majority Satisfiability, Minimum Satisfiability, and Weighted Satisfiability.

ACKNOWLEDGEMENT

This research was supported by Short Term Grant (304/PMATHS/6315226) by Universiti Sains Malaysia.

REFERENCES

Abdullah, W. A. T. W. (1992). Logic programming on a neural network. *International Journal of Intelligent Systems, 7*(6), 513-519.

Alzaeemi, S. A., & Sathasivam, S. (2018). Linear kernel Hopfield neural network approach in horn clause programming. In D. Mohamad, A. B. Akbarally, H. Maidinsah, M. M. Jaffar, M. Mohamed, S. R. Sharif & W. E. Z. W. A. Rahman (Eds.), *AIP Conference Proceedings* (Vol. 1974, No. 1, p. 020107). Melville, NY: AIP Publishing.

Asirelli, P., De Santis, M., & Martelli, M. (1985). Integrity constraints in logic databases. *The Journal of Logic Programming, 2*(3), 221-232.

Banitalebi, A., Aziz, M. I. A., Bahar, A., & Aziz, Z. A. (2015). Enhanced compact artificial bee colony. *Information Sciences, 298,* 491-511.

Cai, S., Luo, C., Lin, J., & Su, K. (2016). New local search methods for partial MaxSAT. *Artificial Intelligence,* 240, 1-18.

Chu, Y., Luo, C., Cai, S., & You, H. (2019). Empirical investigation of stochastic local search for maximum satisfiability. *Frontiers of Computer Science, 13*(1), 86-98.

Goldberg, D. E., & Deb, K. (1991). A comparative analysis of selection schemes used in genetic algorithms. In G. J. E. Rawlins (Ed.), *Foundations of genetic algorithms* (Vol. 1, pp. 69-93). San Mateo, CA: Morgan Kaufmann Publishers, Inc.

Goldberg, D. E., & Holland, J. H. (1988). Genetic algorithms and machine learning. *Machine learning, 3*(2), 95-99.

Hamadneh, N., Sathasivam, S., Tilahun, S. L., & Choon, O. H. (2012). Learning logic programming in radial basis function network via genetic algorithm. *Journal of Applied Sciences (Faisalabad), 12*(9), 840-847.

Hopfield, J. J. (1982). Neural networks and physical systems with emergent collective computational abilities. *Proceedings of the national academy of sciences, 79*(8), 2554-2558.

Jia, D., Duan, X., & Khan, M. K. (2014). Binary artificial bee colony optimization using bitwise operation. *Computers and Industrial Engineering, 76*, 360-365.

Karaboga, D. (2005). *An Idea Based on Honey Bee Swarm for Numerical Optimization* (Vol. 200) (Technical report-tr06). Erciyes University, Turkey.

Karaboga, D., & Basturk, B. (2007). A powerful and efficient algorithm for numerical function optimization: artificial bee colony (ABC) algorithm. *Journal of Global Optimization, 39*(3), 459-471.

Karaboga, D., & Basturk, B. (2008). On the performance of artificial bee colony (ABC) algorithm. *Applied Soft Computing, 8*(1), 687-697.

Karaboga, N. (2009). A new design method based on artificial bee colony algorithm for digital IIR filters. *Journal of the Franklin Institute, 346*(4), 328-348.

Karp, R. M. (1972). Reducibility among combinatorial problems. In R. E. Miller, J. W. Thatcher & J. D. Bohlinger (Eds.), *Complexity of Computer Computations* (pp. 85-103). Boston, MA: Springer.

Kashan, M. H., Nahavandi, N., & Kashan, A. H. (2012). DisABC: A new artificial bee colony algorithm for binary optimization. *Applied Soft Computing, 12*(1), 342-352.

Kasihmuddin, M. S. M., Mansor, M. A., & Sathasivam, S. (2016). Artificial bee colony in the Hopfield network for maximum k-satisfiability problem. *Journal of Informatics and Mathematical Sciences, 8*(5), 317-334.

Kasihmuddin, M. S. M., Mansor, M. A., & Sathasivam, S. (2017). Hybrid genetic algorithm in the Hopfield network for logic satisfiability problem. *Pertanika Journal of Science and Technology, 25*(1), 139-152.

Kasihmuddin, M. S. M., Mansor, M. A., & Sathasivam, S. (2018a). Discrete Hopfield neural network in restricted maximum k-satisfiability logic programming. *Sains Malaysiana, 47*(6), 1327-1335.

Kasihmuddin, M. S. M., Mansor, M. A., & Sathasivam, S. (2018b). Satisfiability based reverse analysis method in diabetes detection. In D. Mohamad, A. B. Akbarally, H. Maidinsah, M. M. Jaffar, M. Mohamed, S. R. Sharif & W. E. Z. W. A. Rahman (Eds.), *AIP Conference Proceedings* (Vol. 1974, No. 1, p. 020020). Melville, NY: AIP Publishing.

Kullmann, O. (1999). New methods for 3-SAT decision and worst-case analysis. *Theoretical Computer Science, 223*(1-2), 1-72.

Layeb, A. (2012). A clonal selection algorithm based tabu search for satisfiability problems. *Journal of Advances in Information Technology, 3*(2), 138-146.

Liang, R. H., & Hsu, Y. Y. (1996). Short-term hydro-scheduling using Hopfield neural network. IEE Proceedings-Generation, *Transmission and Distribution, 143*(3), 269-275.

Lynce, I., Manquinho, V., & Martins, R. (2018). Parallel maximum satisfiability. In Y. Hamadi & L. Sais (Eds.), *Handbook of Parallel Constraint Reasoning* (pp. 61-99). Cham, Switzerland: Springer.

Mansor, M. A. B., Kasihmuddin, M. S. B. M., & Sathasivam, S. (2017). Robust Artificial immune system in the Hopfield network for maximum k-satisfiability. *International Journal of Interactive Multimedia and Artificial Intelligence, 4*(4), 63-71.

Mansor, M. A., Kasihmuddin, M. S. M., & Sathasivam, S. (2016a). Enhanced Hopfield network for pattern satisfiability optimization. *International Journal of Intelligent Systems and Applications, 8*(11), 27-33.

Mansor, M. A., Kasihmuddin, M. S. M., & Sathasivam, S. (2016b). VLSI circuit configuration using satisfiability logic in Hopfield network. *International Journal of Intelligent Systems and Applications (IJISA), 8*(9), 22-29.

Mansor, M. A., Sathasivam, S., & Kasihmuddin, M. S. (2018). 3-satisfiability logic programming approach for cardiovascular diseases diagnosis. In D. Mohamad, A. B. Akbarally, H. Maidinsah, M. M. Jaffar, M. Mohamed, S. R. Sharif & W. E. Z. W. A. Rahman (Eds.), *AIP Conference Proceedings* (Vol. 1974, No. 1, p. 020022). Melville, NY: AIP Publishing.

Mérida-Casermeiro, E., Galán-Marín, G., & Munoz-Perez, J. (2001). An efficient multivalued Hopfield network for the traveling salesman problem. *Neural Processing Letters, 14*(3), 203-216.

Ning, J., Liu, T., Zhang, C., & Zhang, B. (2018). A food source-updating information-guided artificial bee colony algorithm. *Neural Computing and Applications, 30*(3), 775-787.

Ohta, M. (2002). Chaotic neural networks with reinforced self-feedbacks and its application to N-Queen problem. *Mathematics and computers in simulation, 59*(4), 305-317.

Ozturk, C., Hancer, E., & Karaboga, D. (2015). A novel binary artificial bee colony algorithm based on genetic operators. *Information Sciences, 297*, 154-170.

Pampará, G., & Engelbrecht, A. P. (2011, April 11-15). Binary artificial bee colony optimization. In 2011 IEEE *Symposium on Swarm Intelligence* (pp. 1-8). Paris, France.

Poloczek, M., Schnitger, G., Williamson, D. P., & Van Zuylen, A. (2017). Greedy algorithms for the maximum satisfiability problem: Simple algorithms and inapproximability bounds. *SIAM Journal on Computing, 46*(3), 1029-1061.

Rojas, R. (2013). *Neural networks: A systematic introduction*. Heidelberg, Germany: Springer-Verlag.

Sathasivam, S. (2006). *Logic Mining in Neural Networks* (Doctoral thesis). University of Malaya, Kuala Lumpur, Malaysia.

Sathasivam, S. (2008, April 2-4). Energy landscapes for Hopfield network programmed with program clauses. In 4th IASTED *International Conference in Advances Computer Science and Technology* (pp. 179-182). Langkawi, Malaysia.

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

Sathasivam, S., & Ng, P. F. (2013). Developing agent based modeling for doing logic programming in Hopfield network. *Applied Mathematical Sciences, 7*(1), 23-35.

Sathasivam, S., Hamadneh, N., & Choon, O. H. (2011). Comparing neural networks: Hopfield network and RBF network. *Applied Mathematical Sciences, 5*(69), 3439-3452.

Xu, Z., He, K., & Li, C. M. (2019). An iterative Path-Breaking approach with mutation and restart strategies for the MAX-SAT problem. *Computers and Operations Research, 104*, 49-58.

Zhang, H., Shen, H., & Manya, F. (2003). Exact algorithms for MAX-SAT. *Electronic Notes in Theoretical Computer Science, 86*(1), 190-203.

Zhang, S., Yu, Y., & Wang, Q. (2016). Stability analysis of fractional-order Hopfield neural networks with discontinuous activation functions. *Neurocomputing, 171*, 1075-1084.

Zhang, X., & Zhang, X. (2017). A binary artificial bee colony algorithm for constructing spanning trees in vehicular ad hoc networks. *Ad Hoc Networks, 58,* 198-204.