A Hybrid Hopfield Network-Imperialist Competitive Algorithm for Solving the SAT Problem

Marjan Abdechiri
Electronic, Computer. & IT Department, Qazvin Azad University, Qazvin, Iran,
marjan.abdechiri@qiau.ac.ir

Mohmmad Reza Meybodi
Computer Engineering and Information Technology Department, Amirkabir University of Technology, Tehran, Iran,
mmeybodi@aut.ac.ir

***Abstract*— The novel Imperialist Competitive Algorithm (ICA) was recently introduced has a good performance in some optimization problems. The ICA inspired by socio-political process of imperialistic competition of human being in the real world. In this paper is proposed two algorithm for Solving SAT problems. First, a new algorithm that combines ICA and LR to solve SAT problem has been proposed. Secondly, this paper presents a hybrid Hopfield network (HNN)-Imperialist Competitive Algorithm (ICA) to solve the SAT problem. In this proposed algorithm, Hopfield neural network (HNN) manages the problem's constraints and ICA algorithm searches for high quality solutions and minimum cost. At first, the SAT problem is formulated to Lagrange multipliers, which represent the weights of the clauses of the SAT and then HNNICA algorithm is used for solving Lagrange multipliers. Some famous benchmark sets containing instances from SAT problems from various domains used to test the HNNICA performance Also we present a detailed comparative analysis of the proposed algorithm's performance. Simulation results show this strategy can improve the performance of the HNN algorithm significantly. Experimental results show that the HNNICA finds solutions faster than existing methods.**

Keywords*: *Imperialist Competitive Algorithm; Hopfield Neural Network; SAT problem.

I. INTRODUCTION

Many EAs, has been proposed for the problem of global optimization. So far, different evolutionary algorithms have been proposed for optimization. Recently, a new algorithm has been proposed by Atashpaz-Gargari and Lucas [1], in 2007, which has inspired not natural phenomenon, but of course from a socio-human from phenomenon. This algorithm has looked at imperialism process as a stage of human's socio-political evolution. The Imperialist Competitive Algorithm makes relation between humans and social sciences on one hand, and technical and mathematical sciences on the other hand, having a completely new viewpoint about the optimization topic. In the ICA algorithm, the colonies move towards the imperialist country with a random radius of movement. In [2] CICA algorithm has been proposed that improved performance of ICA algorithm by the chaotic maps are used to adapt the angle of colonies movement towards imperialist's position to enhance the escaping capability from a local optima trap. The ICA algorithm is used for Neural Network Learning based on Chaotic Imperialist Competitive Algorithm [3].

Binary HNN is a Hopfield neural network used for solving constraints in optimization problems [4]. In [5] the HNN algorithm is hybridized with genetic algorithm. In [6] the HNN algorithm is hybridized with simulated annealing.

The aim of this paper is twofold: First we apply ICA algorithm to solving SAT problem and second we present a novel hybrid Hopfield network and ICA for solving SAT problems that constraint are managed by the HNN and the quality of the solution obtained is improved by the ICA algorithm.

The proposed algorithm (HNNICA) has a good performance for solving SAT problems. The HNN algorithm reduces the search space of the ICA algorithm. The performance of HNNICA has been evaluated in several SAT problems and it compared with the ICALR proposed and LPPH with very good results in all test instances considered. The empirical results obtained using the benchmarks indicate that the speed of convergence and the quality of solutions are better than LPPH. The HNNICA and ICALR algorithms have been tested on randomly generated problems and some classes of problems from the DIMACS test set.

The rest of this paper organized as follows: Section 2, provides an introduction of SAT problem. In section 3, Imperialist Competitive Algorithms (ICA) are reviewed. In section 4, The ICA algorithm proposed for solving SAT. In section 5, the proposed algorithm HNNICA is introduces. In section 6, is devoted to simulation results and finally section 7, conclusion and related work the paper.

II. SATISFIABILITY PROBLEM

The Satisfiability (SAT) problems belong to an important class of discrete constraint satisfaction problems (CSP). The SAT problem is a set of m clauses $\{C_1, C_2, \ldots, C_m\}$ on n variables $x = (x_1, x_2, \ldots, x_n)$, $x_i \in \{0,1\}$ and a Boolean formula in conjunctive normal form (CNF).

$$C_1 \wedge C_2 \wedge \ldots \wedge C_m \quad (1)$$

A clause is a combination of m literals where a literal is a Boolean variable x_i or its negation $\bar{x}_i$. If the number of literals is n for all clauses, then SAT is referred to as n-SAT. The goal of the SAT problem is to determine whether there exists an assignment of truth values to variables that makes the CNF formula satisfiable. The propositional satisfiability (or SAT) problem is one of researched problems in computer science and artificial intelligence, mathematics, machine vision, robotics and computer-aided manufacturing. In this paper, we transform the fitness function to Lagrange Multipliers representation. So objective in continuous space

978-1-4244-9386-9/11/$26.00

may "smooth out" some infeasible solutions, leading to a smaller number of local minima explored.

$$\min_{y\in E^n} f(y) = \sum_{i=1}^{m} c_i(y) \quad (2)$$

Where

$$c_i(y) = \prod_{j=1}^{n} q_{i,j}(y_j) \quad (3)$$

$$q_{i,j}(y_j) = \begin{cases} y_j & \text{if } x_j \text{ in } c_i \\ 1-y_j & \text{if } \overline{x}_j \text{ in } c_i \\ 1 & \text{otherwise} \end{cases} \quad (4)$$

Fitness function SAT is

$$\text{(SAT)} \quad \text{find x} \quad (5)$$
$$\text{such that} \quad \text{x satisfies } c_r \quad r = 1,2,\dots,m$$
$$x \in \{0,1\}^n.$$

Fitness function CONSAT is

$$\text{(CONSAT)} \quad \text{findx} \quad (6)$$
$$\text{such that} \quad h_r(x) = 0, \quad r = 1,2,\dots,m,$$
$$x \in [0,1]^n.$$

Where

$$h_r(x) = \prod_{i=1}^{n} g_{ir}(x) \quad (7)$$

The above problem can be reformulated using Lagrange multipliers into the following unconstrained problem Lagrange function for SAT is defined as follow:

$$F(x,w) = \sum_{r=1}^{m} w_r h_r(x),$$
$$x \in [0,1]^n, w \in (0,\infty)^m. \quad (8)$$

The last decade, several algorithms have been developed for solving the SAT problem. These algorithms divided into two main classes: complete (Systematic) and incomplete algorithms (non-systematic). An algorithm is *complete,* if return a solution for a problem if one exists, and prove it unsolvable otherwise. The most efficient complete algorithms are based on the Davis-Putnam-Loveland procedure [7]. This algorithm is a complete, backtracking-based algorithm for deciding the satisfiability of propositional logic formulae in conjunctive normal form. Several algorithmic variations of the Davis-Putnam procedure have been proposed [8].

The SAT problems are hard to solve using complete algorithms. One way to overcome for this problem is Local search algorithms [9,10] and evolutionary algorithms [11-13]. These include techniques Such as Simulated Annealing [14], WSAT algorithm [15], GSAT [16], Tabu Search [17].

The GSAT starts with a randomly generated truth assignment. It then changes the variable assignment that leads to the largest increase in the total number of clause satisfied. GSAT is greedy because it always tries to increase the number of true clauses.

In [18] is introduce the Lagrange programming neural network methods for SAT problem and showed that the LPPH algorithm outperforms the DPL and the BM algorithms. In [19] analyzed the stability and the local convergence properties LPPH algorithm.

Dynamical differential equations for LPPH are defined as follows: LPPH algorithm is a lagrangian programming neural network with polarized high-order connections. In this algorithm neuron i correspond to variable x_i. The potential of neuron i is u_i Each neuron has a polarity for each connection. If literal in clause is negative so polarity is negative. The input to neuron i via connection r is $\prod_{j\neq i} g_{jr}(x)$ or $\left(-\prod_{j\neq i} g_{jr}(x)\right)$ and dynamic differential equations for LPPH are

$$\frac{du_i}{dt} = -\sum_{r=1}^{m} w_r \frac{\partial h_r(x)}{\partial x_i}, \quad i = 1,2,\dots,n. \quad (9)$$

$$x_i = \frac{1}{1+e^{-u_i}}, \qquad i = 1,2,\dots,n. \quad (10)$$

$$\frac{dw_r}{dt} = h_r(x), \qquad r = 1,2,\dots,m \quad (11)$$

$$\frac{dx_i}{dt} = -x_i(1-x_i)\frac{\delta F(x,w)}{\delta x_i} =$$

$$x_i(1-x_i)\sum_{i=1}^{m} w_r \frac{\partial h_r(x)}{\partial x_i} \qquad i = 1,2,\dots,n \quad (12)$$

$$\frac{dw_r}{dt} = -\alpha w_r + \frac{\delta F(x,w)}{\delta w_r} = -\alpha w_r + h_r(x)$$
$$r = 1,2,\dots,m \quad (13)$$

Here, $F(x,w) \geq 0$, and $F(x,w) = 0$ iff x is a solution of the CONSAT. That α is the attenuation coefficient, which influences the performance of the LPPH. Each weight w_r increases according to the degree of unsatisfiability of clause *Cr*. This causes changes in the energy landscape of the Lagrangian function. The values of the variables change using the gradient descent. The instances Benchmark of SAT are used for comparing results are shown in Table1, 2.

TABLEI. CNF'S USED IN THE EXPERIMENT

Name	n	m	
EXPD1	50	100	3-SAT. Unique solution.
EXPD2	100	430	
EXPF1	319	941	3-SAT
EXPF2	318	940	
EXPH1	100	762	Test pattern generation of combinational circuits
EXPH2	225	2799	
EXPU1	50	120	Randomly generated 3-SAT unique solution
EXPU2	50	130	

TABLE II. BENCHMARKS FOR SIMULATION.

Benchmark	Instances	Variables	Clauses
Uf50-218/uuf50-218	2*1000	50	218
Uf75-325/uuf75-325	2*100	75	325
Uf100-430/uuf30-430	2*1000	100	430
Uf125-538/uuf125-538	2*100	125	538
Uf150-645/uuf150-645	2*100	150	645
Uf175-753/uuf175-753	2*100	175	753
Uf200-860/uuf200-860	2*100	200	860
Uf225-960/uuf225-960	2*100	225	960
Uf250-1065/uuf250-1065	2*100	250	1065

III. INTRODUCTION OF IMPERIALIST COMPETITIVE ALGORITHMS (ICA)

Imperialist Competitive Algorithm (ICA) is a new evolutionary algorithm in the Evolutionary Computation field based on the human's socio-political evolution. The algorithm starts with an initial random population called countries. Some of the best countries in the population selected to be the imperialists and the rest form the colonies of these imperialists. In an N dimensional optimization problem, a country is a $1 \times N$ array. This array defined as below

$$country = [p_1, p_2, \dots, p_N] \quad (14)$$

The cost of a country is found by evaluating the cost function (f) at the variables $(p_1, p_2, p_3, \dots, p_N)$. Then

$$c_i = f(country_i) = f(p_{i1}, p_{i2}, \dots, p_{iN}) \quad (15)$$

The algorithm starts with *N* initial countries and the N_{imp} best of them (countries with minimum cost) chosen as the imperialists. The remaining countries are colonies that each belong to an empire. The initial colonies belong to imperialists in convenience with their powers.

The imperialist countries absorb the colonies towards themselves using the absorption policy. The absorption policy shown in Fig.1, makes the main core of this algorithm and causes the countries move towards to their minimum optima. The imperialists absorb these colonies towards themselves with respect to their power that described in (16). The total power of each imperialist is determined by the power of its both parts, the empire power plus percents of its average colonies power.

$$TC_n = cost(imperialist_n) + \xi mean\{cost(colonies\ of\ empire_n\} \quad (16)$$

Where TC_n is the total cost of the *n*th empire and ξ is a positive number which is considered to be less than one.

$$x \sim U(0, \beta \times d) \quad (17)$$

In the absorption policy, the colony moves towards the imperialist by *x* unit. The direction of movement is the vector from colony to imperialist, as shown in Fig.1, in this figure, the distance between the imperialist and colony shown by d and *x* is a random variable with uniform distribution. Where β is greater than 1 and is near to 2. So, a proper choice can be $\beta = 2$. In our implementation γ is $\pi/4$ (*Rad*) respectively.

$$\theta \sim U(-\gamma, \gamma) \quad (18)$$

In ICA algorithm, to search different points around the imperialist, a random amount of deviation is added to the direction of colony movement towards the imperialist. In Fig. 1, this deflection angle is shown as θ, which is chosen randomly and with an uniform distribution. While moving toward the imperialist countries, a colony may reach to a better position, so the colony position changes according to position of the imperialist. In this algorithm, the imperialistic competition has an important role. During the imperialistic competition, the weak empires will lose their power and their colonies.

after a while all the empires except the most powerful one will collapse and all the colonies will be under the control of this unique empire.

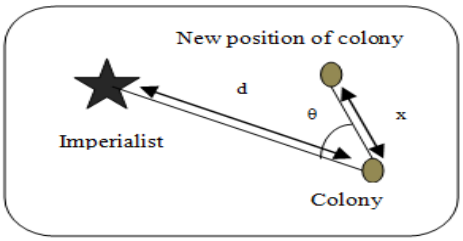

Figure1. Moving colonies toward their imperialist.

IV. THE ICA ALGORITHM FOR SOLVING SAT

In this section, the ICALR to solve the SAT is explained. The ICA approach is used to search for the Lagrange multipliers w that represent the weights of the clauses of the SAT. Firstly, the solving SAT problem is performed using ICA algorithm. This algorithm is called ICALR. The ICALR is used for solving Lagrange Multipliers of SAT problem. Lagrange Multipliers and the variables of SAT problem are solutions and any individual is introduced with variables and multipliers are shown with Matrix w and x. The number of Lagrange Multipliers is m.

$$w = \begin{matrix} w_{11} w_{12} \dots\dots w_{1m} \\ w_{21} w_{22} \dots\dots w_{2m} \\ \dots\dots\dots\dots\dots\dots \\ w_{p1} w_{p2} \dots\dots w_{pm} \end{matrix}$$

The number of variables of is n.

$$x = \begin{matrix} x_{11} x_{12} \dots\dots x_{1n} \\ x_{21} x_{22} \dots\dots x_{2n} \\ \dots\dots\dots\dots\dots\dots \\ x_{p1} x_{p2} \dots\dots x_{pn} \end{matrix}$$

A country, one row in matrix w, represents a set of the Lagrange multipliers corresponding to each clause. Each colony has a structure with two matrices w,x. The fitness of a country is defined in eq. (8). In the ICA approach, the fitness of a country is interpreted as the worth of its location in the search space. The matrices w and x are updated in each decade.

V. A PROPOSED ALGORITHM (HNNICA)

In This paper, we propose to combine the HNN with genetic algorithm to solving SAT problem. The HNNICA algorithm is formed using the HNN and the ICA. In the proposed algorithm, HNN runs before calculating the fitness function for each individual in the ICA. The HNN reduce the search space of ICA algorithm and HNN discard some solutions that are not feasible for SAT problem. In the HNNICA, HNN do local search and ICA is global search.

The HNNICA is proposed for solving Lagrange multipliers. In Fig. 4, the Diagram of the HNN algorithm is shown.

The procedure of the proposed algorithm is shown as follows:

Step1: The SAT problem is formulated to a Lagrange function that this function is unconstraint and continues.

Step2: Initialize countries of ICA (random).

Step3: For every individual run the HNN to obtain a feasible P. (In Fig. 2, Flowchart of HNN for solving the SAT problem is shown.).

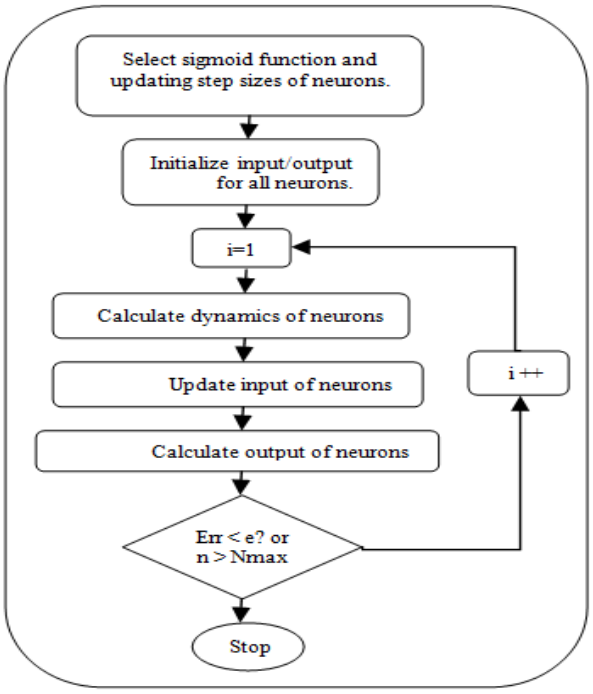

Figure2. Flowchart of HNN for solving the SAT problem.

Step4: Calculate the fitness value of the individual.

Step5: Create new individual.

Step6: Moving colonies toward imperialists and continues.

Step 7: While (Number of decades).

In Hopfield is described as follows: the output of neurons represents value of lagrange multipliers and variables of SAT. The weights are chosen such that an energy function can be defined that it is lagrangian function and lowest energy state is best solution. Hopfield evolve energy function in time to decrease monotonically until a stable steady state is reached. This steady state is correspond to the optimal solution. First SAT encoded to Lagrangian function and secondly an energy function and the corresponding weights have to be determined. In Figure 5 show diagram of Hopfield network is used in HNNICA. In HNNICA runs HNN algorithm for 20 iterations.

Figure 6, show details of implementation HNNICA algorithm.

VI. EXPERIMENTAL RESULTS

In this paper, first a new algorithm that combines ICA and LR to solve SAT problem has been proposed. The results obtained after solving some instances of the SAT showed that ICALR is computationally efficient in solving these problems. ICALR was faster than ICA and PSOLR it provided solutions that are comparable to these approaches. In terms of the solution quality, the ICALR provided a "best solution" with a lower cost than ICA and PSOLR for problem sizes larger than 200 variables. In some of instances ICALR may trap into local optima that it show in Fig. 3 and Table 3. The results are in 1000 iteration of each algorithms. The parameters for ICA algorithm are NumOfCountries = 80, NumOfInitialImperialists = 8, NumOfDecades = 1000, RevolutionRate = 0.3, AssimilationCoefficient= 2, Zeta = 0.02.

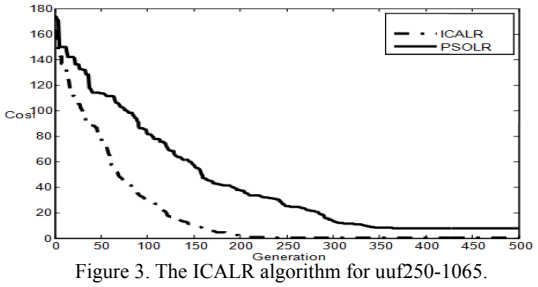

Figure 3. The ICALR algorithm for uuf250-1065.

TABLE III. COMPARING THE BEST COST.

Benchmark	PSOLR	ICALR
Uf50-218/uuf50-218	-120.06	-130.76
Uf75-325/uuf75-325	-94.01	-99.93
Uf100-430/uuf30-430	-90.00	-93.17
Uf125-538/uuf125-538	-85.42	-89.50
Uf150-645/uuf150-645	-76.20	-77.44
Uf175-753/uuf175-753	-60.00	-72.55
Uf200-860/uuf200-860	-57.77	-66.34
Uf225-960/uuf225-960	-52.51	-63.7
Uf250-1065/uuf250-1065	-50.11	-59.00

Secondly, a new hybrid Hopfield neural network-Imperialist competitive algorithm (HNNICA) for solving SAT problem has been presented. The algorithm consists of a Hopfield neural network which manages the problem's constraints, hybridized with ICA which improves the solution obtained from the network. Simulations in a set of SAT benchmark problems have shown very good performance of the algorithm.

TABLE IV. COMPARING THE BEST COST.

Best cost	HRNN	ICALR	HNNICA
SAT50	-45	-130	**-135**
SAT100	-30	-93	**-123**
SAT125	-26	-89	**-116**
SAT150	-20	-77	**-109**
SAT200	-18	-66	**-97**
SAT250	-10	-63	**-91**
SAT300	-8	-36	**-80**

TABLE V. COMPARING THE TIME.

Time(second)	HRNN	ICALR	HNNICA
SAT50	10.50	20.45	**18.12**
SAT100	37.21	45.70	**36.91**
SAT125	35.64	43.55	**30.34**
SAT150	82.23	82.61	**73,62**
SAT200	163.01	150.12	**89.00**
SAT250	257.51	264.65	**120.55**
SAT300	321.87	289.22	**158.22**

HNNICA algorithm, enhance the global search capability of the HNN algorithm. This idea balances the exploration and exploitation abilities of the proposed algorithm, using ICA and HNN. We examined the proposed algorithm in several SAT problems. From the results of experimentations on SAT problems, we observed that the proposed ICALR and HNNICA algorithms, especially HNNICA is superior to HNN.

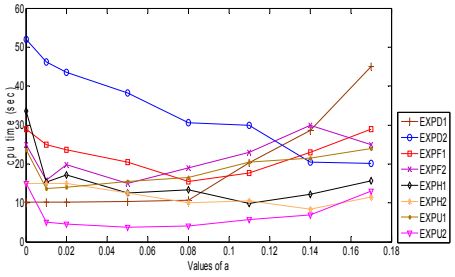

Figure 4. CPU Time versus α (LPPH).

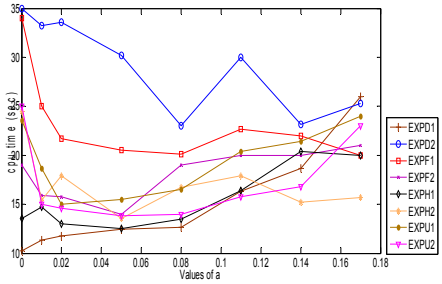

Figure 5. CPU Time versus α (HNNICA).

Fig.4,5 show the results for 8 problems. These problems are solved by the LPPH. In the LPPH, α parameter is attenuation coefficient of weights. The performance of LPPH for solving the SAT strongly depends the value of α. The good value for α depends the problem. In figure 1, show CPU time using changing the value of α. Fig2 shows the results of HNNLR for selecting good value for α is somewhat relaxed than the LPPH.

VII. CONCLUSION AND FUTURE WORK

In this paper is proposed two algorithm for Solving SAT problems. First, a new algorithm that combines ICA and LR to solve SAT problem has been proposed. Secondly, this paper presents a hybrid Hopfield network (HNN)-ICA to solve the SAT problem. In this proposed algorithm, Hopfield neural network (HNN) manages the problem's constraints and ICA algorithm searches for high quality solutions and minimum cost. HNN hybridized with the ICA which improves the solution obtained from the network. Simulations in a set of SAT benchmark problems have shown very good performance of the algorithm. HNNICA algorithm, enhance the global search capability of the HNN algorithm. This idea balances the exploration and exploitation abilities of the proposed algorithm, using ICA and HNN. Simulations are observed that the ICALR and HNNICA algorithms have a good performances, especially HNNICA is superior to HNN. In the future, we will work on the affect of the PSO algorithm on the performance of the ICA algorithm. The results show the HNNLR for selecting good value for α is somewhat relaxed than the LPPH.

REFERENCES

[1] E. Atashpaz-Gargari and C. Lucas, "Imperialist Competitive Algorithm: An Algorithm for Optimization Inspired by Imperialistic Competition", IEEE Congress on Evolutionary Computation (CEC 2007), pp 4661-4667, 2007.

[2] H. Bahrami, K. Faez and M. Abdechiri, "Imperialist Competitive Algorithm using Chaos Theory for Optimization", UKSim-AMSS 12th International Conference on Computer Modeling and Simulation Cambridge, UK, pp. 98-103, 2010, ISBN: 978-0-7695-4016-0.

[3] M. Abdechiri, K. Faez and H. Bahrami, "Neural Network Learning based on Chaotic Imperialist Competitive Algorithm", The 2nd International Workshop on Intelligent System and Applications (ISA2010), 2010.

[4] J. Hopfield, "Neurons and physical systems with emergent collective computational abilities", Proc. Natl. Acad. Sci. USA, 79:pp.2554–8, 1982.

[5] J.Balicki, A. Stateczny, B. Zak, "Genetic algorithms and Hopfield neural networks to solve combinatorial optimization problems", Applied Mathematics and Computer Science, 10(3):pp.568–92, 1997.

[6] S. Salcedo-Sanz and J. Portilla-Figueras, "Optimal solution to crossbar packet-switch problems using a sequential Hopfield neural network", Neurocomputing, 70: pp.2603–2607, 2007.

[7] M. Davis, G. Logemann and D. Loveland, "A Machine Program for Theorem-Proving", Communications of the ACM, 5, pp. 394-397, 1962.

[8] D. W. Loveland, "Automated Theorem Proving: A Logical Basis", North-Holland, 1978.

[9] M. Bertran, S. Lakhdar and G. Eric, "Tabu search for SAT", In Proc. of the 14th National Conference on Artificial Intelligence and 9th Innovative application of Artificial Intelligence Conference (AAAI-97/IAAI-97), pp. 281-285, 1997.

[10] W. M. Spears, "Simulated annealing for hard satisfiability problems", In second DIMACS implementation challenge: cliques, coloring an dsatisfiability, volume 26 of DIMACS Series in Discrete Mathematics and Theoretical Computer Science, pp.533-558, 1996.

[11] A. Kenneth, P. Andre and W. M. Spears, "Using genetic algorithm to solve NP-complete problems", In Proc. of the Third Int. Conf. on Genetic Algorithms, pp. 124-132, 1989.

[12] J. K. Hao and R. Dorne, "A new population-based method for satisfiability problems", In Proc. of the 1th Eropean conf. on Artificial Intelligence, pp. 135-139, Amsterdam, 1994.

[13] C. Fleurent and J. A. Ferland, "Genetic and hybrid algorithms for graph coloring", Annals of Operations Research, 3:pp.437-461, 1997.

[14] S. Kirkpatrick, C.D. Gelatt, and M.P. Vecchi, "Optimization by simulated annealing", Science, 4598: pp. 671–680, 1983.

[15] B. Selman, H. Kautz, and B. Cohen, "Noise strategies for improving local search", in 22th Nat. Conf. on Artificial Intelligence, California, pp. 337–343, AAAI Press, 1994.

[16] B.Selman, H. Levesque and D. Mitchell, "A New Method for Solving Hard Satisfiability Problems", Proceedings of the 1992 AAAI Conference, San Jose, CA, pp. 440 – 446, 1992.

[17] Bertrand Mazure, Lakhdar Sais, and Eric Gr´egoire, "Tabu search for SAT", In Proc. of the AAAI-97/IAAI-97, pp. 281–285, P1997.

[18] M. Nagamatu, and T. Yanaru, "Lagrange programming neural networks with polarized high-order connections for satisfiability problems of propositional calculus," IIZUKA'94. The 3rd International Conference on Fuzzy Logic, NeuraI Nets & Soft Computing, pp. 233-235, 1994.

[19] M. Nagamatu. and T. Yanaru, "On the stability of Lagrange programming neural networks for satisfiability problems of propositional calculus," to appear in Neurocomputing, 13(2-4):pp.440-6, 1992.