Indonesian Journal of Electrical Engineering and Computer Science
Vol. 10, No. 3, June 2018, pp. 1319~1330
ISSN: 2502-4752, DOI: 10.11591/ijeecs.v10.i3.pp1319-1330

Neural Network And Local Search To Solve Binary CSP

Adil Bouhouch*[1], Hamid Bennis[2], Chakir Loqman[3], Abderrahim El Qadi[4]
[1,2]Team TIM, High School of Technology - Moulay Ismail University, Meknes, Morocco
[3]department of informatics, Sciences Faculty, Dhar Mehraz, Sidi Mohammed Ben Abdellah University, Fez, Morocco
[4]LASTIMI, High School of Technology - Mohammed V University of Rabat Morocco
*Corresponding author, e-mail: bouhouch.adil@gmail.com

Article Info

Article history:

Received Jan 15, 2018
Revised Mar 12, 2018
Accepted Mar 28, 2018

Keywords:

Constraint Satisfaction Problem
Continuous Hopfield Neural Network
Meeting Scheduling
Quadratic Problem

ABSTRACT

Continuous Hopfield neural Network (CHN) is one of the effective approaches to solve Constrain Satisfaction Problems (CSPs). However, the main problem with CHN is that it can reach stabilisation with outputs in real values, which means an inconsistent solution or an incomplete assignment of CSP variables. In this paper, we propose a new hybrid approach combining CHN and min-conflict heuristic to mitigate these problems. The obtained results show an improvement in terms of solution quality, either our approach achieves feasible soluion with a high rate of convergence, furthermore, this approach can also enhance theperformance more than conventional CHN in some cases, particularly, when the network crashes.

Corresponding Author:

Bouhouch Adil,
Team TIM, High School of Technology,
Moulay Ismail University, Meknes, Morocco.
Email: bouhouch.adil@gmail.com

1. INTRODUCTION

In general, a Constraint Satisfaction Problem (CSP) consists of a finite set of variables; each one has a finite domain of values and a set of constraints which imposes a variable domain restriction. And the goal is finding a complete assignment of variables which satisfies all constraints. Many problems from the real world can be reformulated as CSP. For example, qualitative and symbolic reasoning, diagnosis, scheduling, spatial and temporal planning, hardware design and verification, real-time systems and robot planning, etc. It is known that solving a CSP with a finite domain is an NP-complete problem, which requires a combination of heuristics [1] and combinatory search methods, in order to be solved in a reasonable time [2]. Mainly, approaches to solve CSPs can be classified into two categories: exact approaches and heuristic ones. As for exact approaches, most of them have the backtracking algorithm (BT) as a main algorithm for solving constraint satisfaction problems. The BT applies a Depth-First search in order to instantiate variables and a backtrack mechanism when dead-ends appear. Many works have been devoted to improve its forward and backward phases by introducing look-ahead and look-back schemes. Some other search algorithms for classical CSPs like Forward Checking, Partial Look-ahead, Full Look-ahead, and Really Full Look-ahead are introduced and their performances are studied widely in the literature [3–6]. For small problems exact method are better but for big instances of NP-Hard problem like CSP need high time cost, due to all methods have at last experiential complexity. In this case non exact approaches give an acceptable solution in reasonable times. As far as heuristic approaches are concerned, we find that, a very different approach has investigated discrete Hopfield Neural Network (HNN) to solve CSP [7]. In this neural network approach, the CSP constraints are associated with the network topology, biases, and connection strengths. Recently, many approaches introduced Continuous Hopfield network to solve CSPs [8–10] or problems which can be formulated as CSP [11]. As we can see in [10], the authors propose mapping CSP to a quadratic problem and elaborate an appropriate parameters setting of network energy, then they solve it by finding the network equilibrium point [8]. Based on the same convergence procedure, Calabuig [12] adds dynamically linear constraints to confine the network to the feasible subspace of solutions; this improvement ensures the final solution validity. Practically, there are two important problems with approaches based on conventional neural network architectures. The first problem is that HNN partially mitigates the problem of getting stuck in local optimum. The second one is due to Hopfield network dynamic which continuously explores the search space and will not always stabilize at border 0 or 1. If the same case appears, we get low solution quality or an incomplete assignment of variables problem. In order to tackle these problems, we propose to improve the CHN solution by the Min-Conflict heuristic(MNC) [13]. The MNC works as follows: it selects the value from each variable domain for which the total error in the next configuration will be minimal. Precisely, MNC is a repair-based stochastic approach, which starts with a complete but inconsistent assignment and then repairs variables implicated in constraint violations repetitively until a consistent assignment is achieved, this approach can solve large-scale problems in a practical time.

This paper is organized as follows: In section 1, we present a modelization of the binary CSP as 0-1 quadratic program, and generalizes energy function associated with the CSPs. Section 2 is devoted to giving implementation details of the proposed approach. In the last section, we show the complexity analysis and the results of the numerical experiments against other approaches like the Genetic Algorithm [14–16].

2. BINARY CSPS SOLVED BY CHN

2.1. Quadratic model of Constraint satisfaction problem

A large number of real problems such as artificial intelligence, scheduling, assignment problem can be formulated as a Constraint Satisfaction Problem. Solving a CSP requires to finding an assignment of all variables problem under constraints restriction. The CSP can be formulated as three sets:

1) Set of N variables $X=\{X_i ; 1 \leq i \leq N \}$.
2) Set of N variables domains: $D =\{ D_i ; 1 \leq i \leq d_i\}$ where each Di contains set of d_i range values for X_i.
3) Set of M constraints: $C =\{ C_i ; 1 \leq i \leq M\}$.

Each constraint C_i associates an ordered variables subset which is called the scope of C_i. The arity of a constraint is the number of involved variables. We can easily reformulate CSP as a Quadratic Problem (QP), by introducing a binary variable x_{ik} for each CSP variable x_i, where k varies over the range of x_i, given as follows:

$$x_{ik} = \begin{cases} 1, & if\ variable\ i\ takes\ value\ k \\ 0, & otherwise \end{cases} \quad (1)$$

For each binary constraint Cij , between the variables yi and yj , we associate a state function defined as:

$$S_{ij}(x) = \sum_{r=1}^{d_i}\sum_{s=1}^{d_j} x_{ir}x_{js}Q_{irjs} \quad (2)$$

Where $x = \{x_{ik}, i \in [1.N], k \in d_i\}$ a vector of QP solution and the quadratic terms Q_{irjs} defined as:

$$Q_{irjs} = \begin{cases} 1 & if\ (r,s) \notin C_{ij} \\ 0 & otherwise \end{cases} \quad (3)$$

From all the equations defined in (2), which correspond to problem constraints, we deduce the objective function of its equivalent QP:

$$f(x) = \sum_{i=1}^{N}\sum_{r=1}^{d_i}\sum_{j=1}^{N}\sum_{s=1}^{d_j} x_{ir}x_{js}Q_{irjs} \quad (4)$$

Furthermore, some strict linear constraints equations must be satisfied by the solution: $\sum_{r=1}^{d_i} x_{ir} = 1$, for $i = 1..N$ which can be written also as $Ax = b$ (A is a $N \times M$ matrix and b is a M dimension vector fully initialized to 1). So, the model is given as follows:

$$(QP) \quad \begin{cases} Min \quad f(x) = x^T Qx \\ with \\ \qquad Ax = b \\ \qquad x \in \{0,1\}^n \end{cases} \tag{5}$$

Systematically, to solve the last Quadratic Optimization Problem with Hopfield model, we need to build an energy function such as the feasible solutions of the problem corresponding to the minimal of CHN energy function.

2.2. Hopfield neural network

Hopfield neural network was introduced by Hopfield and Tank [17] [18] [19]. At the beginning, it was developed to solve combinatorial optimization problems. Then it was extended extensively to other areas of application, such as recognition and optimization. Hopfield neural network is classified as an efficient local search, which gives an acceptable solution to hard optimization problems at a reasonable time. Basically, this neural network model is a fully connected neural network and its dynamic stats are governed by the following differential equation:

$$\frac{dy}{dt} = -\frac{x}{\tau} + Tx + i^b \tag{6}$$

Where:

- x : vector of neurons input
- y : vector of output
- T : matrix of weight between each neurones pairs
- i^b : The biases which describe the neurons self-internal noises.

Hopfield proved that if T is symmetric then the network energies is a Lyapunov function:

$$E(x) = -\frac{1}{2}x^t T x - (i^b)^t x + \sum_{i=1}^{n} \frac{1}{\tau_i} \int_0^{x_i} g^{-1}(v) dv \tag{7}$$

For each neurons, the input is governed by an activation function x = g(y) which varies between 0 and 1. This function is given by:

$$g(y) = \frac{1}{2}(1 + \tanh(\frac{y}{u_0})) \tag{8}$$

We define the energy function of CHN as to be associate with the cost of the problem to be minimized; this is down by taken τ too large:

$$E(x) = x^t T x + (i^b)^t x \tag{9}$$

In this paper, we use the ability of CHN to resolve Quadratic model optimization. So we choose a CHN energy function similar to [10], which is adapted to solve the quadratic formulation of the binary CSPs:

$$E(x)=\frac{\alpha}{2}\sum_{i=1}^{N}\sum_{j=1}^{N}\sum_{r=1}^{d_i}\sum_{s=1}^{d_j} xirx_{js}Q_{ijrs}+\ \beta\sum_{i=1}^{N}\sum_{r=1}^{d_i}x_{ir}+\gamma\sum_{i=1}^{N}\sum_{r=1}^{d_i}x_{ir}(1-x_{ir}) \qquad (10)$$

The first term penalizes two values taken by two linked variables which caused a violation, the second term restricts each variable to take one value, the third term aggregates all linear constraints (Ax = b), and the last term satisfies the propriety of integrity which ensures neurons to stabilize at 0 or 1 state. By corresponding (9) and (10), we deduce the link weight and biases of the network.

$$\begin{cases} T_{irjs}=-\alpha(1-\delta_{ij})q_{irjs}-\delta_{ij}\phi+2\delta_{ij}\delta_{rs}\gamma \\ i_{ir}^{b}\ \ =-\beta-\gamma \end{cases} \qquad (11)$$

with $\delta_{ij}=\begin{cases}1 & if\ \ i=j \\ 0 & if\ i\neq j\end{cases}$ is the Kronecker delta.

To obtain an equilibrium point of the CHN, the parameters setting procedure is used [10] [8]. The Principe of this procedure is to move the network in the direction of $\frac{dy}{dt}$. This process speeds up the neural network convergence significantly. Furthermore, to assure the stability, we use the hyperplane analysis method which calculates the energie function evolution. Let $E_{ir}=\frac{\partial E(x)}{\partial x_{ir}}$, the following conditions must be imposed to ensure the convergence.

- $\alpha > 0$ to have a minimization problem
- The first one concerns avoiding the case when two values are assigned to the same variable, $\exists i,j\in D(X_k)$ with $i\neq j$ and their corresponding neurons $x_{ki}=x_{kj}=1$, so E(x) must verify:

$$E_{ir}(x)\geq 2\phi+\beta-\gamma\geq\varepsilon \qquad (12)$$

- The second ensures that all variable must be assigned and escape the case where $x_{ir}=0$ $r\in\{1,\dots,d_i\}$ so E(x) must verify also:

$$E_{ir}(x)\leq\alpha d+\beta+\gamma\leq-\varepsilon \qquad (13)$$

With:

$$d=Max\left\{\ \sum_{j=1}^{N}\sum_{s=1}^{d_j}q_{irjs}\ /\ i\in\{1,\dots,N\}\ et\ r\in\{1,\dots,d_i\}\ \right\}$$

Empirically, the best value of ε is 10^{-5} parameter setting by solving [10]

$$\begin{cases} \phi\geq 0,\ \alpha>0 \\ -\phi+2\gamma\geq 0 \\ 2\phi+\beta-\gamma=\varepsilon \\ \alpha d+\beta+\gamma=-\varepsilon \end{cases} \qquad (14)$$

The CHN parameters setting and the starting point used in this paper are similar to [10]. Furthermore, a procedure which allows each neuron of the same variable to take 0, if one converges to the boarder 1, is added. This procedure is called Moderator (Figure 1).

```
Function Moderator( $x_{ik}$ : Current neuron ) :
    If $(x_{ik} = 1)$ then
        For each $j \in Dom(V_i) \setminus k$ do
            $x_{ij} = 0$
        end For
    end If
End
```

Figure 1. Moderator

This approach, which only uses Continuous Hopfield neural Network (CHN), was extensively studied over different instances of Constraints satisfaction problem [8], but simulation proved that this approach has many disadvantages. For example, sometimes the network in continuous dynamic does not gives a valid solution, and it is attracted by the nearest local minimal to the starting point. This attracting effect remains a result of its deterministic input-output interaction of the units in the network. Consequently, the network is not able to escape from a local solution closed to the starting point. Also, for Hopfield neural network with a continuous dynamics case, each unit output can take any value between 0 and 1. So, the network can be stranded at a local minimum which contains some units that still take real values. In this case, we obtain an invalid solution for CSP. To overcome this neural network weakness, the main idea of this work is to repair the solution given by the CHN and improve it by a known Min-conflict algorithm.

3. CHN AND MIN CONFLICT HEURISTIC TO SOLVE CSPS

There are many methods which combine two or more no exacts approaches to solve a given optimization problem [11, 20–23]. In the same direction we introduce a hybrid approach based CHN and MNC. The MNC algorithm [13] is a very simple and fast local repairing method to resolve CSPs, which aims at assigning all the variables randomly. Next, it iteratively selects one variable from the set of the variables with conflicts which violates one or more constraints of the CSP. Then, it assigns a value to the selected variable, so that it can minimize the number of conflicts. MNC has demonstrated to be able to solve the queens problem in minutes [24]. MNC is widely used to construct hybrid algorithms with other optimizations [25–28]. In this way, the basic idea of our proposed approach is to use MNC to improve the solution reached by CHN. This will be done in tw o steps (see Figure 2). First, MNC visits all assigned variables; for each one, we apply Min-Conflict directly to the neural network structure, then, it returns the best assignment for the current variable (see Figure 3), the decision will be taken by the sum of all activated neurons weight. Second, we propagate this assignment to other set variables not yet assigned iteratively by applying the MNC heuristic to guarantee as much consistency as possible. The diagram of our proposed algorithm is described by (Figure 4).

```
Function CHNMNC ( CSP : Problem ) :
    $V_a = CHN(CSPs)$
    For (each cluster $x_j \in V_a$) do
        $V_a = V_a \setminus (x_j, a)$ {a is the current affected value to $x_j$ }
        $a = Min\text{-}Conf(x_j, V_a)$ { new value assigned to $x_j$ }
        $V_a = V_a \bigcup (x_j, a)$
    end For
    {propagate current sub assignment to other
    variables which not assigned }
    For (each cluster $x_j \notin V_a$) do
        $a' = Min - Conf(x_j, V_a)$
        $V_a = V_a \bigcup (x_j, a')$
    end For
    return $V_a$
End
```

Figure 2. Main function which improve solution by Min-conflict algorithm

```
Function Min-Conf ( $x_i$ : Current variable , $V_a$ : Assigned variables set ) :
    let $x^*_{ir}$: the current position which is assigned to the variable $x_i$
    For (each value $v_k \in Dom(x_i)$) do
        $w(k) = \sum_{j \in V_a} x^*_{js} T_{ikjs}$
        { $x^*_{js} = 1$ is the position assigned to variable $x_j$}
    end For
    let $W_{max}$ set of maximum output wait
    If ($x^*_{ir} \in P_{max}$ ) then
        return r
    else
        return random value from $P_{max}$
    end If
End
```

Figure 3. Selected the most coherent neuron of current cluster with other variables clusters already affected

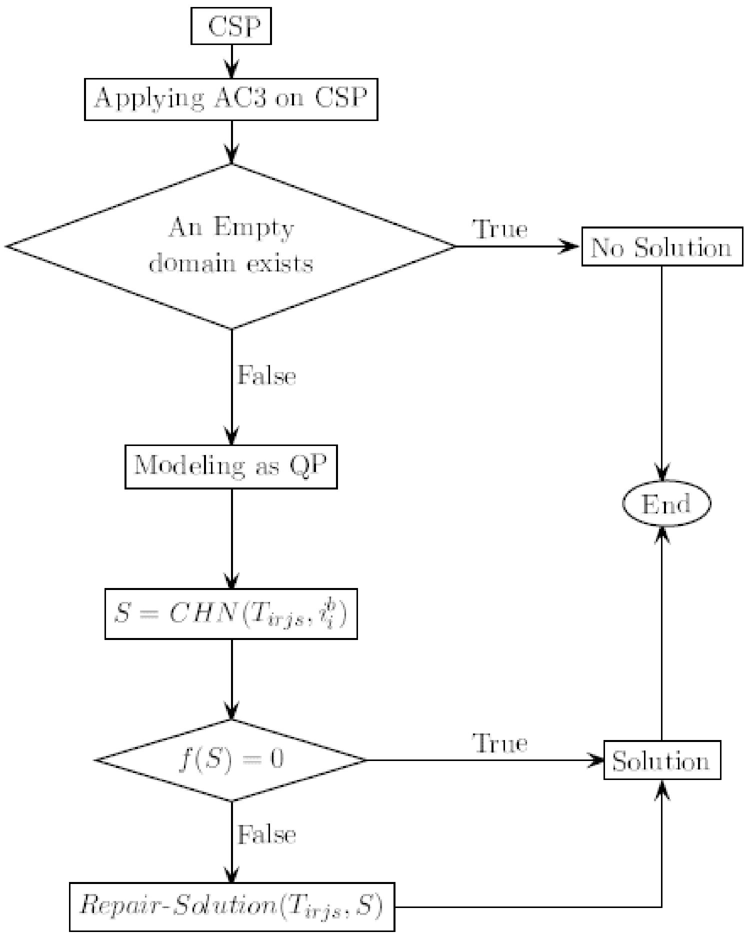
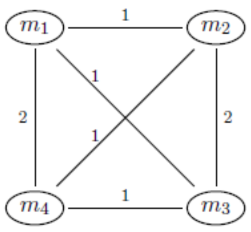


Figure 4. Diagram of the proposed algorithm

Example 1: *Meeting Scheduling Problem*

The Meeting Scheduling problem (MSP) is a decision-making process affecting several people, in which it is necessary to decide when and where several meetings could be scheduled. To show how this algorithm works, we consider the following MSP example:

- *Persons={0,1,2,3}*
- *Meetings:*
  - *m_1 attend person group: {0,1}*
  - *m_2 attend person group:{1,2}*
  - *m_3 attend person group:{0,1,2}*
  - *m_4 attend person group:{2,3}*
- *Location L={ pl_1, pl_2, pl_3, pl_4}*
- *Distances :*



- *Time Slot={ t_1, t_2, t_3, t_4, t_5}*
- *Duration of each meeting 1 time slot*

This problem can be reformulated as the following CSP:

- *Variables: V={ m_1, m_2, m_3, m_4} list of meeting*
- *Domains D=={ t_1, t_2, t_3, t_4, t_5} where*
  t_1=t_4={0,1,2,3,4} and t_2=t_3={0,4}
- *Constraints C= {C_{ij} , $1 \le i \le j \le 4$ }*

$$t_1 = t_4 = \{0,1,2,3,4\} \; and \; t_2 = t_3 = \{0,4\}$$

We supposed that CHN after running gives an incomplete solution S=V_a={{m_1=1},{m_2=4},{m_3=0}}

First pass : For each variable in V_a

- *m_1*
  - *w(0)=w(1)=w(3)=w(4)=-α*
  - *w(2)=0*
    so Min-Conf(m_1,V_a) return 2
- *m_2*
  - *w(0)= -α*
  - *w(4)=0*
    So Min-Conf(m_2,V_a) return 4
- *m_3*
  - *w(0)=0*
  - *w(4)=- α*
    so Min-Conf(m_3,V_a) return 0

The second phase will take m_4 which is not yet assigned :

- *m_4*
  - *w(0)=w(1)=w(3)=w(4)= -α*
  - *w(2)=0*
    so Min-Conf(m_4,V_a) return 2

The final solution is S={{m_1=2},{m_2=4},{m_3=0},{m_4=2},}.

4. NUMERICAL RESULTS

4.1. Generated randomly instances

To evaluate the performance of proposed resolving methods, we run some preliminary experiments on the randomly generated problems and we compare the solution quality by the number of violations. Simulation has been done with the following machine characteristics: I5 core(TM) 2,35GHz processor and limitation of memory to 2GO. For CHN parameters, the best values were founded empirically:

$$\phi = \alpha d + 2\varepsilon \qquad \gamma = \frac{\phi}{2} \qquad \beta = \varepsilon - 3\gamma$$

To generate random problems, we use a random generator based extended model B as it is described in [29-31]. This extended model which is called Model RB is able to generate a hard satisfiable instance. To summarize this generation method, a random CSP is defined by the following five input parameters:

- k: denotes the arity of each constraint, fixed at k=2, so all constraints are binary
- n : denotes the number of variables,
- α: determines the domain size $d=n^{\alpha}$ of each variable,
- r>0: determines the number $m=r\,n\ln(n)$ of constraints,
- 1>p>0: determines the number $t=p\,d^{k}$ of disallowed tuples of each relation.

In the following, a class of CSPs will be denoted by a tuple of the form <n,d,m,k,p>. For simulation, we generate 50 instances for each p value for k=2, α =0.8, r=3 and n in {20, 30, 40}, and we calculate the average of violated constraints of each instance in the same class of p value. For the model parameter used, we get <20,11,180,p>, <30,15,306,p> and <40,19,443,p>. The corresponding load curve is given in Figures 5, 6 and 7 respectively. Comparison is made between the solution obtained by the proposed algorithm and the solution given by the original approach [10]. As we can see Min-Conflict improves the quality of the solution considerably around mean 50%, and the success of network is up to 100%, but for CHN approach, we find the empirically mean value 72% over all runs.

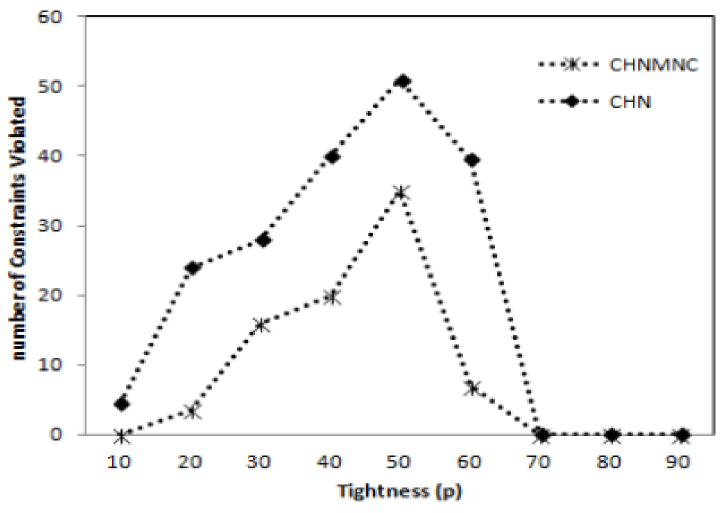


Figure 5. number of violation over classe N=20

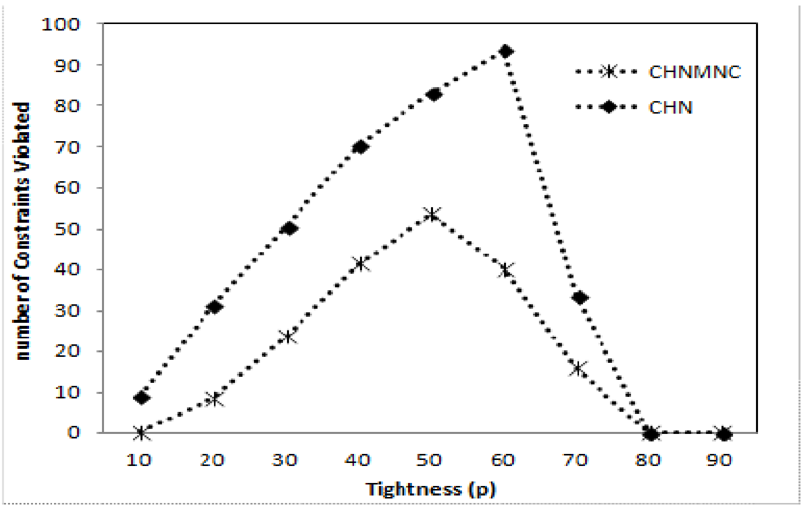


Figure 6. number of violation over classe N=30

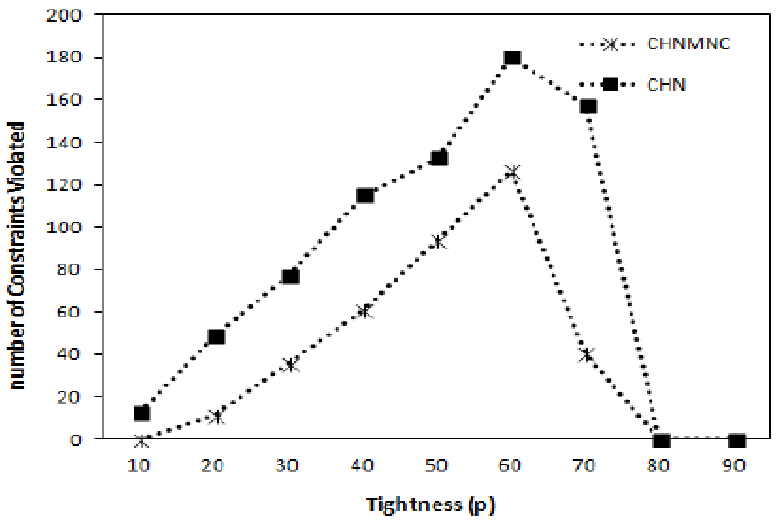


Figure 7. number of violation over classe N=40

4.2 Typical instances

For showing the practical interest of our approach, we also study its performance over problems of different natures (random, academic and real-world problems) [32]. The goal is to evaluate its performance with other evolutionary algorithms [33, 34]. Thus, we compare its efficiency with the Genetic Algorithm (GA) [35] and the Particular Swarm Optimization (PSO) [36, 37]. In practice, rather than authors settings we have empirically searching the Genetic Algorithm and PSO goods ones, adopted to CSP problem. So, for GA the population was 200 individuals, mutation rate equals to 5% and crossing rate equals to 72%, as for PSO [36, 37] we chose $\varphi_1 = \varphi_2 = 1$ and population size fixed at 100. We also run each one 200 times.

The CHN parameters setting and the starting point used in this paper are similar to [10], we have used the Variable Updating Step (VUS) technique proposed by Talaván and Yáñez in [8]. Table 1 shows the comparison between our approach CHN-MNC and original CHN. The description of table columns is the following:

- V: the number of variables.
- C: the number of constraints.
- Ratio mean: the average of the optimal value in a 200 run.
- Time: the average of the time of all runs.

Table 1. Performance of our proposed methods CHN-MNC and CHN over typical instances problems

Name of	V	C	CHN-MNC		GA		PSO	
Instance			Mean	CPU(s)	Mean	CPU(s)	Mean	CPU(s)
queens-10	10,00	45,00	1,00	0,01	2,00	0,02	13,00	0,04
queens-20	20,00	190,00	2,00	0,03	4,00	0,06	39,00	0,11
queens-5-5-5	25,00	160,00	0,00	0,03	0,00	0,06	19,00	0,10
frb30-15-5-mgd	30,00	210,00	10,00	1,07	14,00	2,46	40,00	4,08
geom-30a-5	30,00	81,00	1,00	0,06	2,00	0,13	7,00	0,21
geom-30a-6	30,00	81,00	0,00	0,05	0,00	0,12	4,00	0,20
queens-30	30,00	435,00	4,00	1,94	6,00	4,45	83,00	7,39
frb40-19-3-mgd	40,00	308,00	14,00	0,93	21,00	2,14	53,00	3,56
geom-40-2	40,00	78,00	23,00	0,01	24,00	0,03	28,00	0,04
geom-40-6	40,00	78,00	0,00	0,09	0,00	0,21	4,00	0,36
myciel-5g-3	47,00	236,00	10,00	0,22	12,00	0,50	47,00	0,83
myciel-5g-4	47,00	236,00	5,00	0,41	8,00	0,95	29,00	1,58
myciel-5g-5	47,00	236,00	1,00	0,47	3,00	1,09	12,00	1,81
myciel-5g-6	47,00	236,00	0,00	0,13	1,00	0,31	12,00	0,51
driverlogw-01c	71,00	217,00	0,00	0,08	0,00	0,17	3,00	0,29
composed*	105,00	620,00	13,00	2,93	14,00	6,75	56,00	11,20
dsjc-125-1-4	125,00	736,00	50,00	0,30	64,00	0,69	102,00	1,15
dsjc-125-1-5	125,00	736,00	19,00	0,80	29,00	1,83	85,00	3,04
Qw h-15-106-1	225,00	2324,00	20,00	1,60	23,00	3,68	66,00	6,11
qwh-15-106-4	225,00	2324,00	18,00	4,06	22,00	9,35	59,00	15,52
qwh-15-106-6	225,00	2324,00	22,00	2,31	23,00	5,31	68,00	8,82
driverlogw-04c	272,00	3876,00	3,00	8,32	5,00	19,13	11,00	31,75
driverlogw-02c	301,00	4055,00	3,00	9,17	5,00	21,09	9,00	35,01
qwh-20-166-0	400,00	5092,00	30,00	16,79	32,00	38,63	93,00	64,12
qwh-20-166-3	400,00	5092,00	29,00	2,88	31,00	6,63	89,00	11,00
qwh-20-166-6	400,00	5092,00	25,00	8,19	29,00	18,84	86,00	31,27
le-450-5a-3	450,00	5714,00	1173,00	333,69	1261,00	767,49	1566,00	1274,03
le-450-5a-4	450,00	5714,00	712,00	3,89	745,00	8,95	1066,00	14,86
le-450-5a-5t	450,00	5714,00	441,00	2,29	470,00	5,26	874,00	8,72

NB:composed* is the instance composed-25-10-20-5

Table 1 we learn that GA and CHN-MNC are close and both better than PSO, but regarding the mean time taken by all compared algorithms CHN-MNC is the short one. GA [35] have the same principle with our approach while they use GA and Minimization of conflicts, but they improve the best individual. It's not sure that improving a best solution will give the good one, it can be near the worst one in the population. For this reason, the means value of the multiple runs of our approach was the best because it improves each founded solution.

5. CONCLUSION

In this paper, we have proposed a new approach for solving binary constraint satisfaction problems. Our hybrid algorithm gives a good solution quality rather than using CHN alone, this improvement is done by adding a no importuned time computation. Furthermore the rate of network success to give a valid solution is up to 100% by repairing. Also, the results of the numerical example show that our approach is competitive with GA and PSO. The basic Hopfield neural network architecture described above includes only binary constraints, but some classes of problems are natively n-ary. The extension of this approach to solve these problems is possible if we can translate any n-ary problem to an equivalent binary one. Other studies are in progress to apply this approach to many real-world problems such as aircraft conflict, timetabling and Meeting scheduling.

REFERENCES

[1] K. Lenin, B. R. Reddy, and M. S. Kalavathi, "Wolf search algorithm for solving optimal reactive (IJEEI), vol. 3, no. 1, pp. 7–15, 2015.

[2] A. Carvalho and C. Santos, "A generator of heavy-tailed search trees," in Recent Developments in deling and Applications in Statistics. Springer, 2013, pp. 107–113.

[3] B. A. Nadel, "Tree search and arc consistency in constraint satisfaction algorithms," in Search in Artificial Intelligence. Springer, 1988, pp. 287–342.

[4] C. Bessière, P. Meseguer, E. C. Freuder, and J. Larrosa, "On forward checking for nonbinary constraint satisfaction," in Principles and Practice of Constraint Programming–CP99. Springer, 1999, pp. 88–102.

[5] R. M. Haralick and G. L. Elliott, “Increasing tree search efficiency for constraint satisfaction problems,” Artificial intelligence, vol. 14, no. 3, pp. 263–313, 1980.

[6] R. Dechter and D. Frost, “Backjump-based backtracking for constraint satisfaction problems,”Artificial Intelligence, vol. 136, no. 2, pp. 147–188, 2002.

[7] T. Nakano and M. Nagamatu, “Lagrange neural network for solving csp which includes linear inequality constraints,” in Artificial Neural Networks: Formal Models and Their Applications–ICANN 2005. Springer, 2005, pp. 943–948.

[8] P. M. Talavan and J. Yanez, “A continuous hopfield network equilibrium points algorithm,” Computers & operations research, vol. 32, no. 8, pp. 2179–2196, 2005.

[9] M. Ettaouil, C. Loqman, K. Haddouch, and Y. Hami, “Maximal constraint satisfaction problemsolved by continuous hopfield networks.” WSEAS Transactions on Computers, vol. 12, no. 2, pp. 29–40, 2013.

[10] KHaddouch, M. Ettaouil, and C. Loqman, “Continuous hopfield network and quadratic programming or solving the binary constraint satisfaction problems.” Journal of Theoretical & pplied Information Technology, vol. 56, no. 3, pp. 362–372, 2013.

[11] B. Adil, L. Chakir et al., “Chn and swap heuristic to solve the maximum independent set problem,” International Journal of Electrical and Computer Engineering (IJECE), vol. 7, no. 6, p. 3583–3592, 2017.

[12] D. Calabuig, S. Gimenez, J. E. Roman, and J. F. Monserrat, “Fast hopfield neural networks sing subspace projections,” Neurocomputing, vol. 73, no. 10, pp. 1794–1800, 2010.

[13] S. Minton, M. D. Johnston, A. B. Philips, and P. Laird, “Minimizing conflicts: a heuristic repair method for constraint satisfaction and scheduling problems,” Artificial Intelligence, vol. 58, o. 1, pp. 161–205, 1992.

[14] A. K. Yadav, O. Rahi, H. Malik, and A. Azeem, “Design optimization of high-frequency power ransformer by genetic algorithm and simulated annealing,” International Journal of Electrical nd Computer Engineering, vol. 1, no. 2, p. 102, 2011.

[15] M. Albreem, “Hybrid micro genetic algorithm assisted optimum detector for multi-carrier systems,” ndonesian Journal of Electrical Engineering and Computer Science, vol. 9, no. 2, 2018.

[16] S. Umar and G. Sridevi, “Analysis of genetic algorithm for effective power delivery and with best upsurge,” Indonesian Journal of Electrical Engineering and Informatics (IJEEI), vol. 5, no. 3, pp. 264–269, 2017.

[17] J. J. Hopfield, “Neural networks and physical systems with emergent collective computationalabilities,” Proceedings of the national academy of sciences, vol. 79, no. 8, pp. 2554–2558, 1982.

[18] J. J. Hopfield, “Neurons with graded response have collective computational properties like those of two-state neurons,” Proceedings of the national academy of sciences, vol. 81, no. 10, pp. 3088–3092, 1984.

[19] J. J. Hopfield and D. W. Tank, “neural computation of decisions in optimization problems,” Biological cybernetics, vol. 52, no. 3, pp. 141–152, 1985.

[20] R. Zhang and R. Chiong, “Solving the energy-efficient job shop scheduling problem: a multiobjective genetic algorithm with enhanced local search for minimizing the total weighted tardiness and total energy consumption,” Journal of Cleaner Production, vol. 112, pp. 3361– 3375, 2016.

[21] M. Zhang, H. Wang, Z. Cui, and J. Chen, “Hybrid multi-objective cuckoo search with dynam- ical local search,” Memetic Computing, pp. 1–10, 2017.

[22] R. Pellegrini, A. Serani, G. Liuzzi, F. Rinaldi, S. Lucidi, E. F. Campana, U. Iemma, and M. Diez, “Hybrid global/local derivative-free multi-objective optimization via deterministic particle swarm with local linesearch,” in International Workshop on Machine Learning, Optimization, and Big Data. Springer, 2017, pp. 198–209.

[23] A. Bouhouch, L. Chakir, and A. El Qadi, “Scheduling meeting solved by neural network and min-conflict heuristic,” in Information Science and Technology (CiSt), 2016 4th IEEE International Colloquium on. IEEE, 2016, pp. 773–778.

[24] S. Minton, A. Philips, M. D. Johnston, and P. Laird, “Minimizing conflicts: A heuristic repair method for constraint-satisfaction and scheduling problems,” Journal of Artificial Intelligence Research, vol. 1, pp. 1–15, 1993.

[25] H. Handa, “Hybridization of estimation of distribution algorithms with a repair method for solving constraint satisfaction problems,” in Genetic and Evolutionary ComputationGECCO 2003. Springer, 2003, pp. 991–1002.

[26] D. Hayakawa, K. Mizuno, H. Sasaki, and S. Nishihara, “Improving search efficiency adopting hill-climbing to ant colony optimization for constraint satisfaction problems,” in Knowledge and Systems Engineering (KSE), 2011 Third International Conference on. IEEE, 2011, pp. 200–204.

[27] T. Liu, M. Liu, Y.-B. Zhang, and L. Zhang, “Hybrid genetic algorithm based on synthetical level of resource conflict for complex construction project scheduling problem,” in Machine Learning and Cybernetics, 2005. Proceedings of 2005 International Conference on, vol. 9. IEEE, 2005, pp. 5699–5703.

[28] H. Zou and B. Y. Choueiry, “Characterizing the behavior of a multi-agent search by using t to solve a tight, real-world resource allocation problem,” in Workshop on Applications of Constraint Programming, Kinsale, County Cork, Ireland, 2003, pp. 81–101.

[29] K. Xu andW. Li, “Exact phase transitions in random constraint satisfaction problems,” Journal of Artificial Intelligence Research, 2000.

[30] K. Xu, F. Boussemart, F. Hemery, and C. Lecoutre, “A simple model to generate hard satisfiable instances,” arXiv preprint cs/0509032, 2005.

[31] K. Xu, F. Boussemart, F. Hemery, and C. Lecoutre, “Random constraint satisfaction: Easy generation of hard (satisfiable) instances,” Artificial Intelligence, vol. 171, no. 8, pp. 514–534, 2007.

[32] L. Christophe, “Xcsp 2.1: a format to represent csp/qcsp/wcsp instances,” http://www.cril.univ-artois.fr/ lecoutre/benchmarks.html, accessed April 4, 2008.

[33] J. Jamian, M. Mustafa, H. Mokhlis, and M. Baharudin, “A new particle swarm optimization technique in optimizing size of distributed generation,” International Journal of Electrical and Computer Engineering, v2 no. 1, p. 137, 2012.
[34] H. Omranpour, M. Ebadzadeh, S. Shiry, and S. Barzegar, “Dynamic particle swarm optimization for multimodal function,” IAES International Journal of Artificial Intelligence, vol. 1, no. 1, p. 1, 2012.
[35] H. Kanoh, M. Matsumoto, and S. Nishihara, “Genetic algorithms for constraint satisfaction problems,” in Systems, Man and Cybernetics, 1995. Intelligent Systems for the 21st Century, IEEE International Conference on, vol. 1. IEEE, 1995, pp. 626–631.
[36] L. Schoofs and B. Naudts, “Swarm intelligence on the binary constraint satisfaction problem,” in Evolutionary Computation, 2002. CEC’02. Proceedings of the 2002 Congress on, vol. 2. IEEE, 2002, pp. 1444–1449.
[37] A. T. S. Al-Obaidi, H. S. Abdullah, and Z. O. Ahmed, Meerkat clan algorithm: A new swarm intelligence algorithm,‖ Indonesian Journal of Electrical Engineering and Computer Science, vol. 10, no. 1, 2018.