Maximal Constraint Satisfaction Problems solved by Continuous Hopfield Networks

[1+]MOHAMED ETTAOUIL,[2]CHAKIR LOQMAN,[1*]KHALID HADDOUCH,[1×]YOUSSEF HAMI
[1]UFR: Scientific Computing and Computer sciences,
Engineering Sciences Modeling and Scientific Computing Laboratory,
Faculty of Science and Technology of Fez,
University Sidi Mohammed ben Abdellah Box 2202, Fez, Morocco.

[2]Department of Computer Engineering, High School of technology
Moulay Ismaïl University, B. P. 3103, 50000, Toulal, Meknes, Morocco.
E-mail: [+]mohamedettaouil@yahoo.fr, [*]haddouchk@yahoo.fr,
[2]chakirfst@yahoo.fr, [×]y.hami@hotmail.com

Abstract: - In this paper, we propose a new approach to solve the maximal constraint satisfaction problems (Max-CSP) using the continuous Hopfield network. This approach is divided into two steps: the first step involves modeling the maximal constraint satisfaction problem as 0-1 quadratic programming subject to linear constraints (QP). The second step concerns applying the continuous Hopfield network (CHN) to solve the QP problem. Therefore, the generalized energy function associated with the CHN and an appropriate parameter-setting procedure about Max-CSP problems are given in detail. Finally, the proposed algorithm and some computational experiments solving the Max-CSP are shown.

Key-Words: - Maximal constraint satisfaction problems, quadratic 0-1 programming, continuous Hopfield network, energy function

1. Introduction

A largely unexplored aspect of Constraint Satisfaction Problem (CSP) is that of over-constrained instances for which no solution exists that satisfies all the constraints or it is difficult to satisfy these constraints. Therefore, a key question is to determine a best possible assignment which maximizes the number of satisfied constraints. These problems are mentioned in the literature as Maximal Constraint Satisfaction Problems (Max-CSP). In this way, we are often looking for solutions which violate the minimum number of constraints. In this paper, we restrict ourselves to Max-CSP where all constraints are considered equally important and the goal is to find the assignment that maximizes the number of satisfied constraints. Several real word problems can be formulated as maximal constraint satisfaction problems, such as planning, scheduling, warehouse location problem, max-clique and max-cut problems. Solving the Max-CSP requires assigning a value for each variable from each domain in such a way that the maximal of constraints are satisfied. However, a Max-CSP has often a high complexity, requiring a combination of heuristics and combinatorial search methods to be solved in a reasonable time. Then, the class of Max-CSP is a subset of the class of NP-complete problems [9].

Formally speaking, a maximal constraint satisfaction problem is one of the most difficult and interesting problems for mathematicians, operational researchers and computational scientists. Thus, a number of different approaches have been developed to solve this problem such as branch and bound algorithm [8, 10], Enhancements of branch and bound methods for the maximal constraint satisfaction problem [17] and Near-Optimal Algorithms for Maximum Constraint Satisfaction Problems [1]. In this work, we propose a new model of Max-CSP which consists in minimizing the quadratic objective function subject to linear constraints (QP). To solve the QP problem, many different methods are tried and tested such as interior points, semi definite relaxations and Lagrangian relaxations [5, 20]. In this paper, we introduce the continuous Hopfield network for solving the QP problem.

Hopfield neural network was introduced by Hopfield and Tank [11, 12]. It was first applied to solve combinatorial optimization problems. It has been extensively studied, developed and has found many applications in many areas, such as pattern recognition, model identification, and optimization.

It has also demonstrated capability of finding solutions to difficult optimization problems [7].

In this paper, our main objective is to propose a new approach for solving the maximal constraint satisfaction problems using the continuous Hopfield network. This paper is organized as follows: In section 2, we propose and describe the new model of the binary Max-CSP problem. This problem is formulated as a quadratic assignment problem with linear constraints. A new theorem, which aims to define the relation between the Max-CSP and the quadratic programming, is demonstrated. In section 3, an introduction of CHN is presented, the generalized energy function associated with the Max-CSP is defined and a direct parameter setting procedure is computed. Finally, the implementation details of the proposed approach and experimental results are presented in the last section.

2. Modeling the Max-CSP

A constraint satisfaction problem refers to the problem of finding values to a set of variables, subject to constraints on the acceptable combination of values. Solving this problem requires finding values for problem variables from each domain, which satisfies all members of constraints. In some cases, it may be impossible or impractical to solve these problems completely. This case is known as a maximal constraint satisfaction problem (Max-CSP). For a Max-CSP, a relevant question, both theoretically and practically, is to determine an assignment of values to variables such that the number of satisfied constraints is maximized. Therefore, a maximal constraint satisfaction problem is a constraint satisfaction problem in which one is prepared to settle for partial solutions; these solutions may violate some constraints. In general, a maximal constraint satisfaction problem forms a class of models representing problems that have common properties, a set of variables and a set of constraints [15, 16, 21]. The variables should be instantiated from a discrete domain. The study of Max-CSP has become focused on binary maximal constraint satisfaction problem. Formally speaking, a maximal constraint satisfaction problem can be conveniently defined by means of the notion of constraint satisfaction problem (CSP). A CSP is a triplet (*Y, D, C*) where [21]:

- $Y = \{y_1, ..., y_n\}$ is a set of variables,
- $D = \{D(y_1), ..., D(y_n)\}$ is a finite collection of value domains associated with the variables, where each $D(y_i)$ is the set of d_i possible values for y_i,
- $C = \{C_1, ..., C_m\}$ is a set of constraints which restricts the values that the variables can simultaneously take.

Given such a constraint satisfaction problem *(Y,D,C)*, the Max-CSP becomes the finding of an assignment of the values of *D* to the variables of *V* such that the number of satisfied constraints is maximized. The Max-CSP can be formulated as a couple *(S, f)* where *S* is the set of all possible assignments of values of *S* to the variables of *V* and *f* is the number of unsatisfied constraints of *C* [8]. A solution of a Max-CSP is a total assignment that minimizes the number of constraint violations. In this context, we propose a new model of the Max-CSP as 0-1 quadratic programming, which consists in minimizing a quadratic function subject to linear constraints. In the following, we want to present a new formulation of the binary Max-CSP.

For each variable y_i of the Max-CSP problem, we introduce d_i binary variables x_{ir} such that:

$$x_{ir} = \begin{cases} 1 & if\ y_i = v_r \\ 0 & Otherwise \end{cases} \qquad v_r \in D(y_i) \qquad (1)$$

Where $r \in \{1, ..., d_i\}$ and $i \in \{1, ..., n\}$.

This matrix is easily converted to a N-vector:

$$x \equiv \left(x_{11} \quad \cdots \quad x_{1d_1} \quad \cdots \quad \cdots \quad x_{n1} \quad \cdots \quad x_{nd_n} \right)^T$$

With $N = \sum_{i=1}^{n} d_i$ and $d_i = |D(y_i)|$

Each variable y_i must take a unique value v_r from its domain $D(y_i)$. Then the linear constraints of Max-CSP are defined below:

$$\sum_{r=1}^{d_i} x_{ir} = 1 \quad \forall i \in \{1, ..., n\} \qquad (2)$$

Each constraint C_{ij} between the variables y_i and y_j is defined by its relation R_{ij} such that it is a subset of the Cartesian product $D(y_i) \times D(y_j)$, specifying the compatible values between y_i and y_j. For each couple (v_r, v_s), we generate a constant:

$$q_{irjs} = \begin{cases} 1 & if\ (v_r, v_s) \notin R_{ij} \\ 0 & if\ (v_r, v_s) \in R_{ij} \end{cases} \qquad (3)$$

Where $i \in \{1, ..., n\}$ and $j \in \{1, ..., n\}$.

Remark 1.

If there is no constraint between two variables y_i and y_j, $R_{ij}(v_r, v_s)$ holds for any pair (v_r, v_s) of $D(y_i) \times D(y_j)$. In this case:

$$q_{irjs} = 0 \qquad \forall r \in \{1, ..., d_i\},\ \forall s \in \{1, ..., d_j\}$$

Each constraint C_{ij} can be characterized by the following expression:

$$S_{ij} = \sum_{r=1}^{d_i} \sum_{s=1}^{d_j} q_{irjs} x_{ir} x_{js}$$

The following theorem defines the link between the constraint C_{ij} and the expression S_{ij}.

Theorem 1.

Let x_{ij} be the binary variable which is defined in the expression (1) such that $\sum_{r=1}^{d_i} x_{ir} = 1 \quad i \in \{1, ..., n\}$. For each constraint C_{ij} between two variables y_i and y_j, we have:

- $S_{ij} = 1$ if and only if the constraint C_{ij} is violated.
- $S_{ij} = 0$ if and only if the constraint C_{ij} is satisfied.

Proof.

* $S_{ij} = 1$ then the constraint C_{ij} is violated

$$S_{ij} = 1 \quad \Rightarrow \quad \sum_{r=1}^{d_i} \sum_{s=1}^{d_j} q_{irjs} x_{ir} x_{js} = 1$$

Then $\exists ! r \in \{1, ..., d_i\}$ and $\exists ! s \in \{1, ..., d_j\}$ such that $q_{irjs} x_{ir} x_{js} = 1$.

So $q_{irjs} x_{ir} x_{js} = 1 \Rightarrow q_{irjs} = 1$, $x_{ir} = 1$ and $x_{js} = 1$. In these cases, we have:

- $x_{ir} = 1 \Rightarrow y_i = v_r$
- $x_{js} = 1 \Rightarrow y_j = v_s$
- $q_{irjs} = 1 \Rightarrow (v_r, v_s) \notin R_{ij}$

Then C_{ij} is violated

* The constraint C_{ij} is violated $\Rightarrow \quad S_{ij} = 1$.

C_{ij} is violated if $\exists ! v_r \in D(y_i)$ and $\exists ! v_s \in D(y_j)$ such that $y_i = v_r$, $y_j = v_s$ and $(v_r, v_s) \notin R_{ij}$.

Then $\exists ! r \in \{1, ..., d_i\}$ and $\exists ! s \in \{1, ..., d_j\}$ such that $q_{irjs} x_{ir} x_{js} = 1$.

Therefore, we have $\sum_{r=1}^{d_i} \sum_{s=1}^{d_j} q_{irjs} x_{ir} x_{js} = 1$.

Finally $S_{ij} = 1$.

* $S_{ij} = 0 \Rightarrow$ the constraint C_{ij} is satisfied.

We have $S_{ij} = \sum_{r=1}^{d_i} \sum_{s=1}^{d_j} q_{irjs} x_{ir} x_{js} = 0$

Then $q_{irjs} x_{ir} x_{js} = 0 \quad \forall r \in \{1, ..., d_i\}$ and $\forall s \in \{1, ..., d_j\}$

Moreover, we have:

- $\sum_{r=1}^{d_i} x_{ir} = 1 \Rightarrow \exists ! r \in \{1, ..., d_i\}$ such that $x_{ir} = 1$
- $\sum_{s=1}^{d_j} x_{js} = 1 \Rightarrow \exists ! s \in \{1, ..., d_j\}$ such that $x_{js} = 1$

Since $q_{irjs} x_{ir} x_{js} = 0 \Rightarrow q_{irjs} = 0$

Then $\exists ! v_r \in D(y_i)$ and $\exists ! v_s \in D(y_j)$ such that $y_i = v_r$, $y_j = v_s$ and $(v_r, v_s) \in R_{ij}$.

Finally C_{ij} is satisfied.

* The constraint C_{ij} is satisfied $\Rightarrow \quad S_{ij} = 0$.

C_{ij} is satisfied, then we have two values $v_r \in D(y_i)$ and $v_s \in D(y_j)$ which were affected respectively to y_i and y_j such that the tuple $(v_r, v_s) \in R_{ij}$.

Therefore, we have:

- $y_i = v_r \Rightarrow \exists ! r \in \{1, ..., d_i\}$ such that $x_{ir} = 1$
- $y_j = v_s \Rightarrow \exists ! s \in \{1, ..., d_j\}$ such that $x_{js} = 1$
- $(v_r, v_s) \in R_{ij} \Rightarrow q_{irjs} = 0$

Then $S_{ij} = \sum_{r=1}^{d_i} \sum_{s=1}^{d_j} q_{irjs} x_{ir} x_{js} = 0$.

Remark 2.

- If there is no constraint between two variables y_i and y_j then $S_{ij} = 0$ (See remark 1).
- In Max-CSP problem, we cannot define the constraint C_{ii} between the variable y_i and itself. In this case, $S_{ii} = 0 \ \forall i \in \{1, ..., n\}$.

Basing on theorem (1), the objective function *f(x)* can be formulated in the following way:

$$f(x) = \sum_{i=1}^{n}\sum_{j=1}^{n} S_{ij} \qquad (4)$$

Where

$$S_{ij} = \begin{cases} 1 & \text{if } C_{ij} \text{ is violated} \\ 0 & \text{if } C_{ij} \text{ is satisfied} \end{cases}$$

The matrix form of the objective function *f(x)* is the following:

$$f(x) = x^T Q' x \qquad (5)$$

The elements of the matrix Q' are q_{irjs} where $i \in \{1,...,n\}$, $j \in \{1,...,n\}$, $r \in \{1,...,d_i\}$ and $s \in \{1,...,d_j\}$.

Recall that, we can transform the matrix Q' into symmetric matrix using the following expression:

$$Q = \frac{1}{2}\left(Q' + Q'^T\right)$$

Then, the objective function can be written in the following matrix form:

$$f(x) = \frac{1}{2} x^T Q x \qquad (6)$$

Where Q is $N \times N$ symmetric matrix. Finally, the binary Max-CSP problem is modeled as a 0-1 quadratic programming with a quadratic function subject to linear constraints:

$$(QP) \quad \begin{cases} Min & f(x) = \frac{1}{2} x^T Q x \\ Subject\ to & \\ & Ax = b \\ & x \in \{0,1\}^N \end{cases}$$

Where Q is an $N \times N$ symmetric matrix, A is an $n \times N$ matrix and b is an n vector. This new optimization model puts in interaction all constraints, and then the Max-CSP problem can be considered globally during the search. Additionally, when solving the model of this problem, we could affect each variable a value from its domain, at the same time, satisfying the maximum of constraints. The following theorem determines the relation between a binary Max-CSP problem and an optimization model QP.

Theorem 2.
Let V(QP) be an optimal value of the 0-1 quadratic programming. The number of violated constraints in maximal constraint satisfaction problem is equal to V (QP).

Proof.
Let *z* be the optimal value of the QP problem.
If V (QP) = *z*, then we have:

$$f(x) = \sum_{i=1}^{n}\sum_{j=1}^{n} S_{ij} = z$$

Recall that

$$S_{ij} = \begin{cases} 1 & \text{if } C_{ij} \text{ is violated} \\ 0 & \text{if } C_{ij} \text{ is satisfied} \end{cases}$$

Where $i \in \{1,...,n\}$ and $j \in \{1,...,n\}$

Finally, if $f(x) = \sum_{i=1}^{n}\sum_{j=1}^{n} S_{ij} = z$ then *z* is the sum of the violated constraints.

Remark 3.
The proposed model can easily detect what constraints are violated in the Max-CSP problem and what is the assignment tuple (v_r, v_s) that violate each constraint. Formally speaking, if $S_{ij} = \sum_{r=1}^{d_i}\sum_{s=1}^{d_j} q_{irjs} x_{ir} x_{js} = 1$, then the violated constraint is C_{ij} and the tuple that violates this constraint is (v_r, v_s).

Remark 4.
The proposed theorem (2) generalizes the theorem (3.1) proposed in the work [3]. The theorem (3.1) establishes the links between a binary CSP and an optimization model QP. In this way, when we have V (QP) =0, the binary Max-CSP problem has a solution which satisfy all constraints.

Example 1. (Robot clothing problem [8])
In this example, we propose an approach for solving the Robot clothing problem based on artificial intelligence concepts and especially constraint satisfaction problems. This problem is to dress a robot using the existing wardrobe: *Cordovans* and *sneakers* for *SHOES*, a *green* and a *white SHIRT*, and three pairs of *SLACKS*: *denim*, *blue*, and *gray*. This problem is described by the figure (1).

In order to simplify the representation of this problem, the elements of this latter can be coded in the following way:

- The variables *SHOES*, *SHIRT* and *SLAKS* are respectively coded by y_1, y_2 and y_3.
- The values *Cordovans*, *sneakers*, *green*, *white*, *denim*, *blue* and *gray* are respectively represented by v_1, v_2, v_3, v_4, v_5, v_6 and v_7.

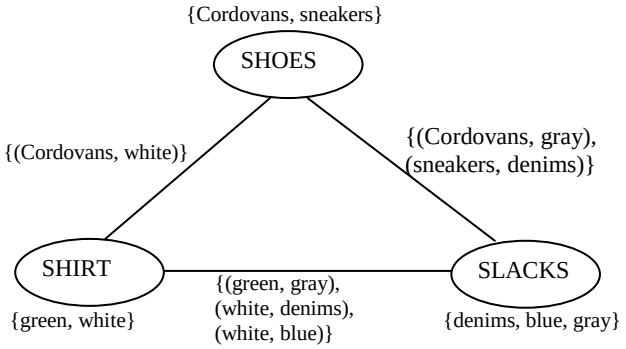

Fig.1: Robot clothing problem.

Then, the robot clothing problem can be reformulated as the following Max-CSP problem (Fig. 1):

- $Y = \{y_1, y_2, y_3\}$ is the set of three variables,
- $D = \{D(y_1), D(y_2), D(y_3)\}$ where:
  - $D(y_1) = \{v_1, v_2\}$,
  - $D(y_2) = \{v_3, v_4\}$,
  - $D(y_3) = \{v_5, v_6, v_7\}$,
- $C = \{C(y_1, y_2), C(y_1, y_3), C(y_2, y_3)\}$ is the set of three constraints where:
  - $R_{12} = \{(v_1, v_4)\}$,
  - $R_{13} = \{(v_1, v_7), (v_2, v_5)\}$,
  - $R_{23} = \{(v_3, v_7), (v_4, v_5), (v_4, v_6)\}$.

We show that, this pedagogical problem is formulated as maximal constraint satisfaction problem. In this example, the reformulation of the objective function is the following (Equation 4):

$$f(x) = S_{12} + S_{13} + S_{23}$$

$$f(x) = x_{11}x_{21} + x_{12}x_{21} + x_{12}x_{22} + x_{11}x_{31} + x_{11}x_{32} + x_{12}x_{32} + x_{12}x_{33} + x_{21}x_{31} + x_{21}x_{32} + x_{22}x_{33}$$

Then, the objective function of the robot clothing problem can be formulated by the following expression: $f(x) = \frac{1}{2} x^T Q x$

Where

$$Q = \begin{pmatrix} 0 & 0 & 1 & 0 & 1 & 1 & 0 \\ 0 & 0 & 1 & 1 & 0 & 1 & 1 \\ 1 & 1 & 0 & 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 1 \\ 1 & 0 & 1 & 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 & 0 & 0 \end{pmatrix}$$

The constraints (2) imply that:

$$\begin{cases} x_{11} + x_{12} = 1 \\ x_{21} + x_{22} = 1 \\ x_{31} + x_{32} + x_{33} = 1 \end{cases}$$

These constraints are equivalent to $Ax = b$ where

$$A = \begin{pmatrix} 1 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 & 1 \end{pmatrix}, \quad b = \begin{pmatrix} 1 & 1 & 1 \end{pmatrix}^T$$

and $x = \begin{pmatrix} x_{11} & x_{12} & x_{21} & x_{22} & x_{31} & x_{32} & x_{33} \end{pmatrix}^T$

Finally, we consider the following 0-1 quadratic program (QP):

$$(QP) \begin{cases} Min & f(x) = \frac{1}{2} x^T Q x \\ Subject\ to & \\ & Ax = b \\ & x \in \{0,1\}^7 \end{cases}$$

The main idea of this paper is to use the new model of the Max-CSP problem described in this section and to apply the continuous Hopfield network in order to give a good solution for the maximal constraint satisfaction problem.

3. Max-CSP solved by continuous Hopfield networks

As can be noticed, the maximal constraint satisfaction problem is modeled as a 0-1 quadratic programming with a quadratic function subject to linear constraints. In this section, we present a general approach to solve the Max-CSP problems using the continuous Hopfield networks.

The neural networks are efficient approaches for solving different problems in different areas [4, 6, 11, 2]. In the beginning of the 1980, Hopfield published two scientific papers, which attracted much interest. This was the starting point of the new area of neural networks, which continues today. Hopfield showed that models of physical systems could be used to solve computational problems. Moreover, Hopfield and Tank [11, 12] presented the energy function approach in order to solve several optimization problems [4]. Their results encouraged a number of researchers to apply this network to different problems. The continuous Hopfield neural network is a generalization of the discrete case. The common output functions used in the networks are hyperbolic tangent functions. Afterwards, many

researchers implemented CHN to solve the optimization problem, especially in mathematical programming problems.

The CHN is a fully connected neural network, which means that every neuron is connected to all other neurons. The connection weights between the neuron *i* and neuron *j* is represented by W_{ij} and each neuron *i* has an offset bias i_i^b [13]. The dynamics of the CHN is described by the following differential equation:

$$\frac{du}{dt} = -\frac{u}{\tau} + Wx + i^b \qquad (7)$$

Where *u*, *x* and i^b will be the vectors of neuron states, outputs and biases. The output function $x_i = g(u_i)$ is a hyperbolic tangent, which is bounded below by 0 and above by 1.

$$g(u_i) = \frac{1}{2}(1 + \tanh(\frac{u_i}{u_0})) \quad where \quad i = 1, ..., N \qquad (8)$$

Where u_0 ($u_0 \in IR^+$) is a parameter used to control the gain (or slope) of the activation function.

For solving any combinatorial problems, it is necessary to map it in the form of the energy function associated with the continuous Hopfield network. The expression of this energy function is the following:

$$E(x) = -\frac{1}{2} x^T W x - (i^b)^T x \qquad (9)$$

In this work, our main objective is to solve the QP problem, i.e., solving maximal constraint satisfaction problem using the continuous Hopfield network. Therefore, the most important step consists in representing or mapping the maximal constraint satisfaction problem in the form of an energy function associated with the continuous Hopfield network. According to the proposed model, which consists in modeling the Max-CSP into a quadratic programming QP, this step of representation becomes easy and more general. Then, the continuous Hopfield network can be used to solve the maximal constraint satisfaction problem.

We show that the model of maximal constraint satisfaction problem is a 0-1 quadratic programming with *N* variables and *n* linear constraints.

$$(QP) \begin{cases} Min & f(x) = \frac{1}{2} x^T Q x \\ Subject\ to & \\ & Ax = b \\ & x \in \{0,1\}^N \end{cases}$$

In order to represent the latter QP problem, the energy function must be defined by two expressions $E^O(x)$ and $E^C(x)$. The first one is directly proportional to the objective function of the QP problem and the second one is a quadratic function that penalizes the violated constraints of the QP problem. Therefore the energy function associated with the CHN is:

$$E(x) = E^O(x) + E^C(x) \qquad \forall\ x \in H \qquad (10)$$

Where H is set of the Hamming hypercube:

$$H \equiv \{x \in [0,1]^N\}$$

We notice that the QP problem has only the family of linear constraints:

$$e_i(x) = \sum_{r=1}^{d_i} x_{ir} = 1 \quad \forall i \in \{1, ..., n\} \qquad (11)$$

The following constraint $x_{ir} \in \{0,1\}$ can be set up in the following way:

$$x_{ir} \in \{0,1\} \Leftrightarrow x_{ir}^2 - x_{ir} = 0$$

Then, the QP problem is equivalent to:

$$(QP) \begin{cases} Min & f(x) = \frac{1}{2} x^T Q x \\ Subject\ to & \\ & e_i(x) = 1 \quad \forall i \in \{1, ..., n\} \\ & x_{ir}^2 - x_{ir} = 0 \end{cases}$$

Where $i \in \{1, ..., n\}$ and $r \in \{1, ..., d_i\}$.

In our case, the following generalized energy function for the QP problem is proposed:

$$E(x) = \alpha f(x) + \frac{1}{2}\phi \sum_{i=1}^{n} (e_i(x))^2 + \beta \sum_{i=1}^{n} e_i(x) + \gamma \sum_{i=1}^{n} \sum_{r=1}^{d_i} x_{ir}(1 - x_{ir}) \qquad (12)$$

With $\alpha \in IR^+$, $\beta \in IR$, $\gamma \in IR$, $\Phi \in IR$ and $\gamma \in IR$. The algebraic form of the generalized energy function is:

$$E(x) = \frac{\alpha}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} \sum_{r=1}^{d_i} \sum_{s=1}^{d_j} q_{irjs} x_{ir} x_{js} + \frac{1}{2}\phi \sum_{i=1}^{n} \sum_{r=1}^{d_i} \sum_{s=1}^{d_i} x_{ir} x_{is} + \beta \sum_{i=1}^{n} \sum_{r=1}^{d_i} x_{ir} + \gamma \sum_{i=1}^{n} \sum_{r=1}^{d_i} x_{ir}(1 - x_{ir}) \qquad (13)$$

To determine the weights and thresholds, we use the assimilation between equation (9) and the algebraic form of the generalized energy function. Then the weights and thresholds of the connections between *N* neurons are:

$$\begin{cases} W_{irjs} = -\alpha(1 - \delta_{ij}) q_{irjs} - \delta_{ij}\phi + 2\delta_{ij}\delta_{rs}\gamma \\ i_{ir}^b = -\beta - \gamma \end{cases} \qquad (14)$$

δ_{ij} is the Kroenecker delta.

$$\delta_{ij} = \begin{cases} 1 & if\ i = j \\ 0 & if\ i \neq j \end{cases}$$

In this way, the quadratic programming has been presented as an energy function of continuous Hopfield network. To solve an instance of the QP problem, the parameter setting procedure is used. This procedure assigns particular values for all parameters of the network, so that any equilibrium points are associated with a valid affectation of all variables when all the constraints in QP problem are satisfied. We observe that the weights and thresholds of the continuous Hopfield network depend on the parameters α, β, ϕ and γ. To solve the QP problem via the CHN, an appropriate setting of these parameters is needed. In this section, our objective is to determine these parameters. The following sets need to be defined in order to determine each state for each variable.

H_C is a set of the Hamming hypercube corners:

$$H_C \equiv \{x \in H : x_i \in \{0,1\}, \forall\ i = 1,..., N\}$$

H_F is a set of feasible solutions:

$$H_F \equiv \{x \in H_C : Ax = b\}$$

The dynamics of the CHN must ensure that any invalid solution $x \notin H_F$ cannot be a stable point. Any constraint which is imposed on the dynamics of the CHN must be translated into a set of constraints on its parameter values; this is the kernel of the parameter setting. A feasible solution is guaranteed by the CHN from a stability analysis of the Hamming hypercube corners set:

$$H_C \equiv \{x \in H : x_{ir} \in \{0,1\}, \forall i \in \{1,...,n\}, \forall\ r \in \{1,...,d_i\}\}$$

It is noted that a feasible solution is identified by a vector $x \in H_F \subset H_C \subset H$.

The parameter-setting procedure is based on the partial derivatives of the generalized energy function:

$$\frac{\partial E(x)}{\partial x_{ir}} = \alpha \sum_{j=1}^{n} \sum_{s=1}^{d_j} q_{irjs} x_{js} + \phi \sum_{s=1}^{d_i} x_{is} + \beta + \gamma(1 - 2x_{ir}) \qquad (15)$$

A point $x \in H$ will be an equilibrium point for the CHN if and only if the two following relations are satisfied:

$$\begin{cases} \dfrac{\partial E(x)}{\partial x_{ir}} \geq 0 \quad such\ that \quad x_{ir} = 0 \\ \dfrac{\partial E(x)}{\partial x_{ir}} \leq 0 \quad such\ that \quad x_{ir} = 1 \end{cases} \qquad (16)$$

This procedure uses the hyperplane method [19], so that the Hamming hypercube H is divided by a hyperplane containing all feasible solutions. To avoid the stability of any no feasible and corner solution $x \in H_C - H_F$, the following instability conditions are imposed:

$$\begin{cases} \dfrac{\partial E(x)}{\partial x_{ir}} \leq -\varepsilon \quad such\ that \quad x_{ir} = 0 \\ \dfrac{\partial E(x)}{\partial x_{ir}} \geq \varepsilon \quad such\ that \quad x_{ir} = 1 \end{cases} \qquad (17)$$

Based on this hyperplane method and the associated half-spaces, the complementary corners set of the feasible solutions for the QP problem is partitioned and a set of analytical equations of the CHN parameter is proposed. The hyperplane method is briefly explained below; however, for simplicity reasons the following parameters constrains are first assigned:

- To minimize the objective function, we impose the following constraint: $\alpha > 0$
- On the other hand, to penalize the non-feasibility of the family of linear constraints $e_i(x)$, it is natural to impose the following constraint: $\phi \geq 0$.
- In order to guarantee the instability of the interior points $x \in H - H_C$, some initial conditions are imposed on some parameters:

$$W_{irir} = -\phi + 2\gamma \geq 0 \qquad (18)$$

Where $\forall i \in \{1,...,n\}$ and $\forall\ r \in \{1,...,d_i\}$.

The partition of $H_C - H_F$ is defined as:

$$H_C - H_F = H_{1,1} \cup H_{1,2}$$

- $H_{1,1} \equiv \{\exists i : e_i(x) > 1\} \cap \{e_0(x) \geq n\}$

  Where $e_0(x) = \sum_{i=1}^{n} e_i(x)$.

In this case, one variable y_i has been assigned two different values v_r, v_s in $D(y_i)$ so $x_{ir} = x_{is} = 1$. Consequently, the following instability condition is imposed:

$$\frac{\partial E(x)}{\partial x_{ir}} \geq 2\phi + \beta - \gamma \geq \varepsilon \qquad (19)$$

- $H_{1,2} \equiv \{\exists i : e_i(x) < 1\} \cap \{e_0(x) < n\}$

In this case, one variable y_i has not been assigned any value $v_r \in D(y_i)$, such that $x_{ir} = 0$ $\forall r \in \{1,...,d_i\}$. Therefore, the following instability condition is imposed:

$$\frac{\partial E(x)}{\partial x_{ir}} \leq \alpha d + \beta + \gamma \leq -\varepsilon \qquad (20)$$

With

$$d = Max\left\{ \sum_{j=1}^{n} \sum_{s=1}^{d_j} q_{irjs} \Big/ i \in \{1,...,n\}, r \in \{1,...,d_i\} \right\}$$

Then, we can determine the parameters setting by resolving the following system:

$$\begin{cases} \alpha > 0 \quad , \phi \geq 0 & \\ -\phi + 2\gamma \geq 0 & (21.a) \\ 2\phi + \beta - \gamma = \varepsilon & (21.b) \\ \alpha d + \beta + \gamma = -\varepsilon & (21.c) \end{cases}$$

The inequality (21.a) guaranteed the satisfaction of the integrity constraints ($x_{ir} \in \{0,1\}$), but the equations (21.b) and (21.c) guaranteed the satisfaction of the linear constraints. These parameters setting are determined by fixing α , ε and computing the rest of parameters γ, β and ϕ :

$$\begin{cases} \phi = d\alpha + 2\varepsilon & \\ \gamma = \phi/2 & (22) \\ \beta = \varepsilon - 3\gamma & \end{cases}$$

Finally, the weights and thresholds of CHN (system 14) can be calculated using these parameters setting. Finally, we obtain an equilibrium point for the CHN using the algorithm depicted in [18], so compute the solution of constraint satisfaction problem.

4. Computational experiments

The algorithm of maximal constraint satisfaction problem was developed to find a solution via the continuous Hopfield network. Moreover, to understand our approach, the resolution of the robot clothing problem is described.

4.1 Description of the proposed algorithm for solving the Max-CSP problem

In order to find a solution to the maximal constraint satisfaction problem via the continuous Hopfield network, an algorithm was developed. The diagram of this solver is described by the following plan (Fig. 2). Recall that, a solution is an assignment of values to all variables so that the maximum of constraints are satisfied. The main steps of this algorithm are the following:

- The first step is to read data of the Max-CSP problem and reformulate it into XML file. This reformulation requires developing techniques to extract data (domains, variables, relationships and constraints).
- The next step concerns modeling the Max-CSP problem as 0-1 quadratic programming (determining the matrix Q, the matrix A and the vector b)

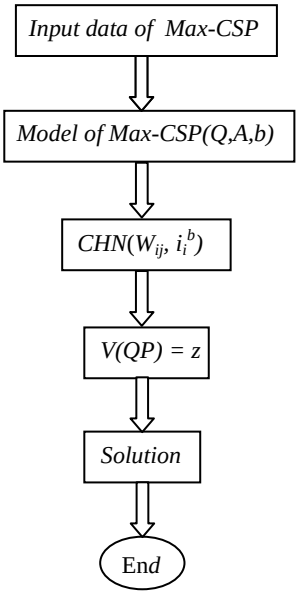

Fig.2: Diagram of the proposed algorithm.

- The last step concerns using the CHN to obtain the solution of this model, so that the number of violated constraints in Max-CSP is equal to the optimal value V(QP) obtained by the continuous Hopfield network (theorem 2).

Example 2.

As can be noticed, the robot clothing problem, described in the example 1, is modeled as the following:

$$(QP) \begin{cases} Min & f(x) = \frac{1}{2} x^T Q x \\ Subject\ to & \\ & Ax = b \\ & x \in \{0,1\}^7 \end{cases}$$

Before applying the continuous Hopfield network to solve this problem, the weights and thresholds of the CHN are computed using the parameters setting α , β , γ and ϕ . These parameters are determined by fixing the (α, ε) and solving the system (22).

Finally, the solution is given by our solver in the 0.001 second (time of resolution). The value for each variable is: $y_1 = v_1$, $y_2 = v_3$ and $y_3 = v_7$ such that this solution is coded in the following way: $SHOES = Cordovans$, $SHIRT = green$ and $SLACKS = gray$. We show that, the number of violated constraints is 1 because the tuple $(Cordovans, green)$ violate the constraint $C(SHOES, SHIRT)$ (remark 3).

4.2 Numerical results

In order to show the practical interest of the proposed approach, the experiments are achieved to solve some typical problems of different natures. These series represent a large spectrum of instances [22]. These experiments are effectuated in personal computer with a 2.79 GHz processor and 512 MB RAM. The performance has been measured in terms of the CPU time per second.

This solver is modeled by the Unified Modeling Language method and implemented by java language. The starting points are chosen in such a way that the points corresponding to the most constrained variables are the most favored in the processing. This choice allows the continuous Hopfield network to handle in a first order the variables most constrained. This process lead to an assignment of values to variables such that the number of satisfied constraints is maximized. This is guaranteed by a good affectation to the variables most constrained. Therefore, the maximum of constraints in the problem are satisfied. Finally, the starting points are randomly generated by the following expression:

$$x_{ir} = 0.8 + 0.19\frac{NVC_i}{NVMC} + 10^{-2}U$$

Where NVC_i is the number of participation for each variable y_i in the set of constraints C, $NVMC = \max\left\{ NVC_i \,/\, i \in \{1,...,n\} \right\}$ representing the more present variable in the constraints of the Max-CSP problem, and U is a random variable in the interval [-0.5, 0.5]. Recall that n is the number of variables. Based on a series of experiments, α and ε are determined by the following values:

$$\alpha = \frac{1}{n},\ \varepsilon = 10^{-4}$$

In this experimentation, some instances as benchmarks for the first round of the 2008 Max-CSPs [22] solver competition are used to test this algorithm (See Table 1).

A statistical study was represented in order to examine the quality of our approach. This study is based on the computation of performance operators. In fact, the quality of solutions obtained by our approach was evaluated in terms of performance report:

$$\rho = \frac{\text{number of violated constraints obtained by this approach}}{\text{best number of violated constraints obtained by the best solver}}$$

Among these operators performance, we have (Table 1):

- Ratio mode: the ratio between the most repetitive (mode) number of violated constraints (the optimal value obtained by CHN) in the number of run and the least number of violated constraints (best results existing in the literature) obtained by the best solver, Ratio mean: the ratio between the average number of violated constraints in a number of run and the best results existing in the benchmarks obtained by other solver,
- Ratio minimum : the ratio of the smaller number of violated constraints (better results obtained by our solver) and the best result existing in the literature obtained by other solver,
- Mean of CPU time: the average time consumed to obtain the solution in a number of run.

We show that, for each instance, if Ratio minimum is less than 1 then the proposed approach gives the best results than the others, i.e., gives a solution that violates fewer constraints compared to the best existing solver. Also Ratio minimum is superior to 1 then the proposed approach cannot give the results better than the others, i.e., gives a solution that violates more constraints compared to the best existing solver. Moreover, if Ratio minimum is equals to 1, in this case the given results is similar to the best solver existing in benchmarks (Table 1).

In comparison with other Max-CSPs solvers, the time of resolution obtained by our software is better than others; it does not exceed 1 second for the most of instances. Generally, our solver is very successful, it happens to get the solution to the maximal constraint satisfaction problems in a minimum time than the author solvers of Max-CSP [22] (See Table 1). For example, instances "kbtree-9-7-3-5-90-08", "cnf-2-40-1800-067529", "maxcut-30-400-5", "maxcut-30-340-5", etc, the proposed solver gives a solution that violates fewer constraints compared to the best existing solver.

Name of instances	number of constraint	Best result obtained on benchmark	Best CPU time obtained on benchmark	Number of run	Ratio Minimum	Ratio mean	Ratio mode	Mean of iterations	mean of CPU Time
geom-40-2	78	22	0.03s	200	1.05	1.36	1.27	58.76	0.006s
geom-30a-4	81	4	0.323 s	200	1.25	3.26	2.75	192.29	0.046s
vcsp-25-10-25-87-44	75	32	5.198s	200	1.38	1.99	1.78	254.42	0.212s
vcsp-25-10-21-85-33	63	19	0.219 s	200	1.58	2.81	2.42	261.8	0.255s
vcsp-25-10-25-87-36	75	29	1.587 s	200	1.21	2.71	1.86	244.96	0.209s
scenw-6-sub4-20	477	23	0.132 s	100	1.83	4.91	2.39	326.1	1.023s
scenw-6-sub2	353	22	0.374s	100	1.32	3.96	1.86	326.31	2.466s
scenw-6-sub2	353	22	0.374s	100	1.32	3.96	1.86	326.31	2.466s
kbtree-9-7-3-5-90-08	189	123	0.311 s	100	0.93	1.44	1.24	235.43	0.076s
kbtree-9-7-3-5-60-01	189	54	0.335s	100	1.17	2.1	1.48	214.25	0.071s
kbtree-9-7-3-5-90-11	189	123	0.338s	100	0.98	1.52	1.24	213.16	0.068s
kbtree-9-7-3-5-80-50	189	97	0.846 s	100	1.03	1.73	1.26	189.65	0.062s
kbtree-9-7-3-5-70-50	189	72	1.522s	100	1.07	1.84	1.46	217.16	0.069s
kbtree-9-5-3-5-90-45	255	163	2.297s	100	0.95	1.6	1.21	232.88	0.133s
kbtree-9-5-3-5-90-10	255	167	3.17s	100	1	1.66	1.26	183.89	0.097s
kbtree-9-5-3-5-80-47	255	129	4.969s	100	1.09	1.74	1.34	226.37	0.121s
kbtree-9-5-3-5-80-08	255	134	5.931s	100	1.13	1.51	1.31	266.14	0.141s
kbtree-9-2-3-5-90-29	309	196	461.111s	100	1.06	1.45	1.2	271.75	0.302s
kbtree-9-2-3-5-80-50	309	153	1143.39s	100	1.11	1.86	1.28	236.32	0.263s
cnf-2-80-300-186945	286	25	1.032s	100	1.04	1.3	1.28	35.83	0.015s
cnf-2-40-1800-067529	667	246	1.367s	50	0.97	1.07	1.06	30.68	0.003s
cnf-2-40-1500-890427	640	210	1.417s	50	1.01	1.1	1.06	28.7	0.002s
cnf-2-40-1600-616123	663	227	1.856s	50	0.98	1.06	1.04	27.5	0.002s
cnf-2-40-2500-147426	685	307	2.139s	50	0.99	1.06	1.05	35.94	0.003s
cnf-2-40-2600-873121	670	297	3.418s	50	0.97	1.09	1.05	42.22	0.004s
cnf-2-40-2400-421728	681	291	3.742s	50	1	1.1	1.06	36.94	0.003s
cnf-2-40-2400-421727	683	284	5.117s	50	1.02	1.12	1.07	41	0.003s
cnf-2-80-800-815450	710	102	163.366s	50	1.23	1.3	1.28	19.72	0.010s
cnf-2-80-800-815444	717	107	391.728s	50	1.13	1.25	1.22	29.26	0.013s
cnf-2-80-1100-992546	946	160	979.48s	50	1.08	1.11	1.11	22.08	0.007s
cnf-2-80-1200-718241	989	176	1261.0s	50	1.05	1.13	1.14	30.28	0.010s
maxcut-30-400-5	400	179	6.597s	100	0.97	1.12	1.06	72.73	0.003s
maxcut-40-480-6	480	193	231.657s	100	1.07	1.2	1.17	37.73	0.005s
maxcut-30-340-5	340	142	3228.99s	100	0.99	1.19	1.12	40.86	0.004s
maxcut-30-370-5	370	160	17.851s	100	1.01	1.22	1.09	48.85	0.002s
maxcut-40-420-5	420	161	20.163s	100	1.06	1.23	1.2	40.17	0.004s
maxcut-40-440-6	440	173	29.374 s	100	1.07	1.24	1.14	42.58	0.004s
maxcut-40-480-8	480	192	136.187 s	100	1.03	1.22	1.17	43.99	0.004s
maxcut-40-540-3	540	225	717.983s	50	1.07	1.17	1.12	44.1	0.004s
maxcut-40-520-1	520	210	272.491s	50	1.07	1.27	1.17	33.84	0.003s
maxcut-40-520-10	520	213	340.681s	50	1.08	1.16	1.14	60.08	0.005s
maxcut-50-580-6	580	219	940.592s	50	1.11	1.29	1.19	34.54	0.004s
maxcut-40-580-1	580	241	161.684s	50	1.05	1.22	1.14	52.26	0.005s
maxcut-50-560-10	560	212	176.564s	50	1.07	1.24	1.21	48.58	0.011s
maxcut-60-580-2	580	207	2121.51s	50	1.15	1.26	1.27	31.54	0.007s
c-fat200-2	16865	176	1.217s	25	1.22	1.22	1.22	366.36	0.963s
c-fat500-2	116111	474	283.354s	25	1.24	1.24	1.25	374.12	4.906s
c-fat500-10	78623	374	1101.83s	25	1.32	1.33	1.33	410.08	5.386s
c-fat500-5	102059	436	1151.8s	25	1.38	1.38	1.38	369.2	4.899s

Table 1: Computational results of the typical Max-CSP instances

However, in some cases as "vcsp-25-10-21-85-33", "kbtree-9-7-3-5-60-01", "maxcut-50-580-6", "c-fat500-10", etc, our approach violate plus the constraints more than author solvers. Finally, we can conclude that the best results are obtained by this approach.

5. Conclusion

In this paper, we have proposed a new approach for solving binary maximal constraint satisfaction problems. The interesting steps of this approach are: proposing the new model of maximal constraint satisfaction problem as a 0-1 quadratic program subject to linear constraints and using the continuous Hopfield network to solve this problem. The most interesting propriety of this approach is used to give the solution of the binary Max-CSP. It is also interesting to note that this method can be used with a non-binary Max-CSP after converting the latter into a binary Max-CSP [14]. The experimental results show that our method can find a good optimal solution in short time compared to other solvers. Future directions of this research are reducing the architecture of Hopfield neural network and applying this approach to get a good solution of real world problems such as warehouse location problem, max-clique and max-cut.

References:

[1] M. Charikar, and al., Near-optimal algorithms for maximum constraint satisfaction problems, *Journal ACM Transactions on Algorithms (TALG) TALG Homepage archive*, Vol. 5, No. 3, 2009, pp. 62 – 68.

[2] H. M. El-bakry and N. Mastorakis, A Modified Hopfield Neural Network for Perfect Calculation of Magnetic Resonance Spectroscopy, *WSEAS TRANSACTIONS on INFORMATION SCIENCE and APPLICATIONS*, Vol. 5, No. 12, 2008, pp. 1654-1666.

[3] M. Ettaouill, and C. Loqman, A New Optimization Model for Solving the Constraint Satisfaction Problem, *Journal of Advanced Research in Computer Science*, Vol. 1, No. 1, 2009, pp. 13-31.

[4] M. Ettaouil, C. Loqman, K. Haddouch. Job Shop Scheduling Problem solved by the continuous Hopfield networks, *Journal of Advanced Research in Computer Science (JARCS)*, Vol. 2, No. 1, 2010, pp. 31 - 47.

[5] M. Ettaouill, and C. Loqman, Constraint Satisfaction Problems Solved by Semidefinite Relaxations, WSEAS TRANSACTIONS on COMPUTERS, Vol. 7, No. 7, 2008, pp. 951-961.

[6] M. Ettaouil, K. Elmoutaouakil, Y.Ghanou. The continuous Hopfield networks (CHN) for the placement of the electronic circuits problem, WSEAS Transactions on Computer, Vol. 8, No. 12, 2009, pp. 1865-1874.

[7] D. J. Evansi, and M. N. Sulaiman, Solving optimisation problems using neucomp-a neural network compiler, *International Journal of Computer Mathematics*, Vol. 62, No. 1, 1996, pp. 1-21.

[8] E. Freuder, and R Wallace, Partial Constraint Satisfaction, *Artificial Intelligence*, Vol. 58, 1992, pp. 21-70.

[9] M.R Garey, and D.S Johnson, A Guide to the Theory of NP-Completeness, *Computers and Intractability, New York: W.H. Freeman and Company*, 1979.

[10] R. M. Haralick et G. L. Elliott, Increasing tree search efficiency for constraint satisfaction problems, *Artificial Intelligence*, Vol. 14, 1980, pp. 263-313.

[11] J. J. Hopfield, Neural networks and physical systems with emergent collective computational abilities, *Proceedings of the National Academy of Sciences of the United States of America*, Vol. 79, 1982, pp. 2554-2558.

[12] J.J. Hopfield, Neurons with graded response have collective computational properties like those of two-states neurons. *proceedings of the National academy of sciences of the USA 81*, 1984, pp. 3088-3092.

[13] J.J. Hopfield, D.W. Tank, Neural computation of decisions in optimization problems, *Biological Cybernetics*, Vol. 52, 1985, pp.1-25.

[14] N. Mamoulis, and K. Stergiou, Solving non-binary CSPs using the Hidden Variables Encoding, *In Proceedings of CP'*, 2001.

[15] V. Kumar, Algorithms for Constraint Satisfaction Problems, T A Survey, *AI Magazine*, Vol. 13, No. 1, 1992, pp. 32-44.

[16] M. Azlinah, Y. Marina, A. M. Itaza, M. Sofianita and A. R. Shuzlina, Constraint Satisfaction Problem Using Modified Branch and Bound Algorithm, *WSEAS*

TRANSACTIONS on COMPUTERS, Vol. 7, No. 1, 2008, pp. 1-7.

[17] Richard J. Wallace, Enhancements of Branch and Bound Methods for the Maximal Constraint Satisfaction Problem, *Proc. of AAAI*, 1996, pp.188-196.

[18] P.M. Talavàn and J. Yànez, A continuous Hpfield network equilibrium points algorithm. *computers and operations research*, Vol. 32, 2005, pp. 2179-2196.

[19] Talavàn and J. Yànez, The generalized quadratic knapsack problem. A neuronal network approach, *Neural Networks*, Vol. 19, 2006, pp. 416-428.

[20] Thiongan, and al., An Adapted Step Size Algorithm for a 0-1 Biknapsack Lagrangean Dual, *Annals of Operations Research*, Vol. 139, No. 1, 2005, pp. 353-373.

[21] E. Tsang, *A Foundations of Constraint Satisfaction*, Academic Press, 1993.

[22] MAX-CSP Competition: available results, Available at *http://www.cril.univ-artois.fr/CPAI08/results/results.php?idev=16, 20.*