Date of publication MAR-29, 2023, date of current version JUL-11, 2021.
www.computingonline.net / computing@computingonline.net

Print ISSN 1727-6209
Online ISSN 2312-5381
DOI 10.47839/ijc.22.1.2876

Improving Continuous Hopfield Network Stability Using Runge-Kutta Method

MOHAMMED EL ALAOUI, MOHAMED ETTAOUIL
Modelling and Scientific Computing Laboratory, University Sidi Mohammed ben Abdellah, Fez, Marocco.
(e-mail: md.elalaoui@gmail.com, mohamedettaouil@yahoo.com)

Corresponding author: Mohammed El Alaoui (e-mail: md.elalaoui@gmail.com).

ABSTRACT Continuous Hopfield network is a recurring network that has shown its ability to solve important optimization problems. Continuous Hopfield network dynamic system is characterized by a differential equation. This equation is difficult to solve, especially for large problems. This led researchers to discretize the differential equation using Euler's method. However, this method generally does not converge to a good solution because it is sensitive to the step size decision and initial conditions. In this work, we discretize the dynamic system of continuous Hopfield network by a new method of Runge-Kutta. This method is strong in terms of stability and performance in order to converge to a better solution. This new method introduces two phases for better network stability. The first phase targets to solve the dynamic equation by the Euler method, while the second phase allows refining the solution found in the first phase. Experimental results on benchmarks show that the proposed approach can effectively improve Hopfield neural network performance.

KEYWORDS Continuous Hopfield network, Runge-Kutta method, Euler method.

I. INTRODUCTION

CONTINUOUS Hopfield networks are dual-use networks, which can be used either as a way to solve optimization problems or as an associative memory in the field of image processing. In 1982, a new model named after its inventor John Hopfield enriched the field of artificial neural networks [1]. In artificial neural networks, information is one-way from input to output. While for continuous Hopfield network, the input turns into the output. This transformation is provided by a continuous transfer function to achieve network stability. For mature stability, the weight matrix should be symmetrical. The convergence of the continuous Hopfield network at a state of equilibrium is calculated by the energy function that decreases with time.

The dynamic system of this network is characterized by a differential equation. The discretization of this differential equation, with the Euler method, has been a subject of work for many researchers [2, 3]. However, this method has limitations related to the decision of the step size. To overcome this problem, we discretized the dynamic system of this network by a new second order Runge-Kutta method [4]. This method has proven its ability to bring the network back to mature stability. This method is guided by two important phases. The first phase consists in finding an approximation of the solution and the second phase refines the solution for a better approximation. The dynamic system of this network becomes more robust with the new approach by comparing itself to the classical network.

The success of Hopfield Artificial Neural Networks is due to its ability to cope with many optimization issues. The main idea of our approach is to adopt the behavior of the Runge-Kutta method with continuous Hopfield networks to ensure better convergence.

This paper presents a new approach to improve the convergence of the artificial neural network algorithm. To evaluate the proposed approach, we modeled the maximum stable problem as a quadratic problem and then as a Hopfield quadratic energy. Then we proceeded to a process of combining between the two algorithms on different instances.

The rest of paper is organized as follows. First, section 2, describes continuous Hopfield network. In section 3, a new approach is proposed for the maximum stable set problem. Section 4, presents and discusses the experimental results. Finally, we conclude the article in section 5.

II. CONTINUOUS HOPFIELD NETWORK

Hopfield Network is a fully connected neural network with a symmetrical matrix of connections. In the process of convergence, the dynamics of these networks converge to one of the equilibrium positions. These equilibrium positions are determined in advance during the learning process [5]. These are local minima of the functional energy of the network. Such a network can be used both as an associative memory and a way to solve some optimization problems [6-10]. Unlike many neural networks, which operate until a response is received after a certain number of iterations, Hopfield network operates until equilibrium is reached when the current state of the network is exactly equal to the previous state.

The neurons of Continuous Hopfield Network (CHN) are interconnected with an activation function called hyperbolic tangent used to calculate the output of each neuron. The dynamics of CHN is described by the following differential equation:

$$\frac{du}{dt} = -\frac{u}{\tau} + Tv + i^b. \qquad (1)$$

The neuron input vector $v = (v_i)$ and the neuron output $u = (u_i)$ with $1 \leq i \leq n$ and $u_i \in \{0,1\}$.

The weight matrix is given by $T = (T_{i,j})$with $1 \leq i \leq n$ and $1 \leq j \leq n$ and $i^b$is the neuron bias.

The output of each neuron is calculated by the following formula:

$$v_i = -\frac{1}{2}\left(1 + \tanh\left(\frac{u_i}{u_0}\right)\right), \qquad (2)$$

where u_0 is a parameter used to control the gain of the enable function.

The point of network stability is known as the system equilibrium point. This stability is ensured by the Lyapunov function which is a stable system decreasing over time [11]. Updating each neuron makes it p ossible to draw a path, which converges to an attractor of the network. The convergence of the Hopfield neural network is ensured by the use of the Lyapunov function, which offers the possibility of finding a local minimum. Hopfield demonstrated that the existence of the Lyapunov function is ensured by the symmetry of the weight matrix with zero diagonal. In [12], therefore, the existence of the equilibrium point is guaranteed. Then the following Lyapunov function exists:

$$E(x) = -\frac{1}{2}x^tTx - (i^b)^tx + \frac{1}{\tau}\sum_{i=1}^{n}\int g^{-1}(v)dv. \qquad (3)$$

Hopfield neural network can be applied to any combinatorial problem, which seeks to optimize an objective function:

$$E(x) = -\frac{1}{2}x^tTx - (i^b)^tx. \qquad (4)$$

Lyapunov idea is based on the energy of the system. In order for the energy of this system to converge to zero as time tends to infinity, a system could be characterized as stable. In this sense, the continuous Hopfield network can be seen as a solver of important optimization problems. We consider the following quadratic problem with n variables and m linear constraints:

$$(P)\begin{cases} Min\ \frac{1}{2}v^tQv + q^t \\ subject\ to \\ Av = b \\ v_i \in \{0,1\}, 1 \leq i \leq n \end{cases} . \qquad (5)$$

The quadratic programming presented above can be solved using the following sets:

$H = \{x \in [0,1]^n\}$ is a set of Hamming hypercube elements.

$H_C = \{x \in H : x_i \in \{0,1\} : i = 1, \ldots, n\}$ is a set of the elements of Hamming hypercube corners.

$H_F = \{x \in H_C : Ax = b\}$ is a set of feasible solutions.

Each given instance is solved by associating its equilibrium points with the local minima of the optimization problem. The energy function can also be defined by:

$$E(x) = E^o(x) + E^R(x) \ : \ \forall x \in H. \qquad (6)$$

$E^o(x)$ is proportional to the objective function of the problem.

$E^R(x)$ is a function to satisfy the constraints in order to ensure the feasibility of each solution.

The introduced energy function is intended to overcome the problem observed on the energy functions used by the authors.

In this article, our intimate goal is to solve the maximum stable set problem by a new approach. To do this, we have two important phases. In the first phase, we solve the dynamic equation of the system by a new Runge-Kutta method for better convergence. In the second phase, we represent the maximum stable problem as a $0 - 1$ quadratic problem.

III. CONTINUOUS HOPFIELD NETWORK IMPROVED FOR MAXIMUM STABLE PROBLEM

Hopfield system process dynamics consists of adopting a robust method to find a solution with a better approximation. The first step of this work is to find a quadratic model for the maximum stable problem, then to convert this model to a continuous Hopfield network. The second step is to adopt the Runge-Kutta method to discretize the Hopfield differential equation. This method goes through two important phases. The first phase is the prediction phase, which consists in finding the solution by using the Euler approach. The second phase is the most important to refine the solution found in the first phase. First, we start the presentation of the formulation of the energy function associated with this maximum stable. Next, we select a practical setting of this function [13]. Then, a search algorithm based on the Runge-Kutta is proposed.

A. MAXIMUM STABLE PROBLEM

Maximum stable problem is represented as an undirected graph

$$G = (V, E) \text{ with } V = \{v_1, v_{2,} \cdots, v_n\}.$$

A stable set of a graph $G = (V, E)$ is a subset S of V such that the subgraph generated by S does not contain an arc.

Maximum stable set problem (MSSP) consists in finding a stable set in the graph G of maximum cardinality $\alpha(G)$. Alongside its theoretical interest, the MSSP problem arises in information retrieval, experimental design and computer vision applications [13].

The stable set problem is NP-hard and even difficult to approximate [14]. The MSSP problem can be solved by using polynomial time algorithms for special classes such as perfect graphs, pie graphs and graphs with long odd cycles [15]. However, the existence of a polynomial time algorithm for arbitrary graphs seems unlikely. In the literature, several researchers focused their research on solving exactly the maximum stable problem.

Carrahan and Pardalos [16] proposed an implicit enumeration technique. Computational results for different relaxations of linear programming stable set were reported by Gruber and Rendl [17]. An effective evolution of the taboo research approach was presented in the original works of Friden, Hertz and de Werra [18].

To solve the MSSP problem using the proposed approach, it must be expressed as a linear assignment problem with a quadratic constraint. Let $S \subset V$ be a stable set of nodes. For each node v_i of the graph G, we introduce the binary variables x_i such that:

$$x_i = \begin{cases} 1 & if\ v_i \in S \\ 0 & Otherwise \end{cases}. \tag{7}$$

Two adjacent nodes v_i and v_j cannot be in the set S :

$$(v_i, v_i) \in E \Rightarrow x_i x_j = 0. \tag{8}$$

The constraints can be expressed in the following form:

$$h(x) = \sum_{i=1}^{n} \sum_{j=1}^{n} b_{ij} x_i x_j = 0. \tag{9}$$

Relationship between two neurons is defined by

$$b_{ij} = \begin{cases} 1 & if\ (v_i, v_j) \in E \\ 0 & Otherwise \end{cases}. \tag{10}$$

The objective function of the mathematical programming model is: $f(x) = -\sum_{i=1}^{n} x_i$. Consequently, the $MSSP$ problem can be expressed in the following algebraic form:

$$(QP) = \begin{cases} Min\ f(x) = -\sum_{i=1}^{n} x_i \\ subject\ to \\ h(x) = \sum_{i=1}^{n} \sum_{j=1}^{n} b_{ij}\, x_i x_j = 0 \\ x \in \{0,1\}^n \end{cases}. \tag{11}$$

The formulation of the energy function for a maximum stable problem is done as follows:

$$E(v) = \alpha \sum_{i=1}^{n} v_i + \frac{1}{2} \phi \sum_{i=1}^{n} \sum_{j=1}^{n} b_{ij} v_i v_j + \gamma \sum_{i=1}^{n} v_i (1 - v_i). \tag{12}$$

The weights of the matrix are given by the following formulation:

$$T_{ij} = -\phi b_{ij} + 2\delta_{ij}\gamma. \tag{13}$$

The Kronecker symbol is given as follows:

$$\delta_{ij} = \begin{cases} 1 & if\ i = j \\ 0 & if\ i \neq j \end{cases}. \tag{14}$$

The parameters φ, γ and α must be chosen so that the equilibrium point of the Hopfield network associated with the MSSP is achieved. The setting procedure is obtained from the partial derivative of the energy function:

$$\frac{\partial E}{\partial v_i} = -\alpha + \phi \sum_{i=1}^{n} b_{ij} v_j + \gamma (1 - 2v_i). \tag{15}$$

The parameterization is determined by the hyper plane method [9]. Before processing, certain conditions are necessary to simplify the determination of these parameters: $\varphi > 0, \gamma > 0$. To minimize the objective function, we impose the following constraint: $\alpha > 0$.

B. PROPOSED APPROACH FOR CONTINUOUS HOPFIELD NETWORK

Hopfield artificial neural network architecture is considered to be the basis of modern algorithms. It is commonly used to reconstruct degraded, noisy or incomplete data. In addition, continuous Hopfield network is interesting for solving many combinatorial problems such as constraint programming and database query optimization [19–21]. The dynamics of this network is characterized by the differential equation, which has the following form:

$$\frac{du}{dt} = f(u). \tag{16}$$

Energy function is defined by

$$f(u) = T \times \tanh(u) - I. \tag{17}$$

Euler method is often used to solve the dynamic equation of the Hopfield network. The following form defines this method:

$$u_{n+1} = u_n + h \times f(u_n). \tag{18}$$

This method suffers from initial state and step size. This leads to bad quality of local solutions when using this method. To overcome this problem, we adopt in this work a new method for better precision of network stability. This method named by Runge-Kutta of the second order [22]. The basic idea is to go through two important phases. The first phase consists in finding an approximation of the solution $\overline{u}_{n+1}$ by Euler method.

$$\overline{u}_{n+1} = u_n + hf(v_n, u_n). \tag{19}$$

In the second phase, the new solution is incorporated into the function to calculate the final solution of the function. This phase makes it possible to refine the final solution for better precision:

$$u_{n+1} = u_n + h \frac{f(v_n, u_n) + f(v_{n+1}, \overline{u}_{n+1})}{2}. \tag{20}$$

By substituting (1) for (2), we finally get a new representation of the second order Runge-Kutta method formula:

$$u_{n+1} = u_n + \frac{h}{2}\left[f(v_n, u_n) + f\big(v_n + h, u_n + hf(v_n, u_n)\big)\right]. \tag{21}$$

The Runge-Kutta method and the Euler method are evaluated with a constant step $h = 0.1$. We have two parameters, the step size h and the order used to control the size of the local error. Practical methods of solving differential equations use such estimates of the local error to determine whether the current choice of step size h is adequate. The following example represents the two methods described above. In order to show the efficiency of the Runge-Kutta method, we compare the convergence of each method for each iteration. The example was used for comparison purposes:

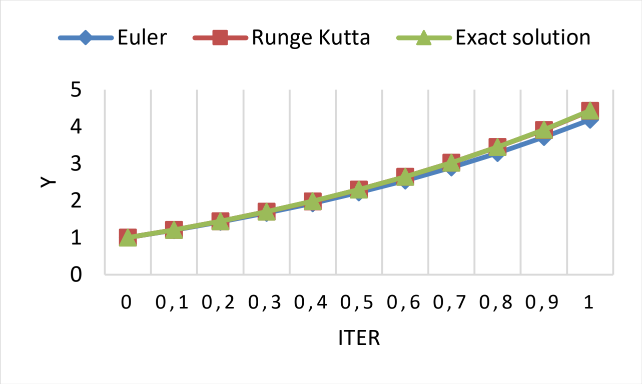

Figure 1. Runge-Kutta approximation

The Runge-Kutta method is the most accurate method to approximate the differential equation solution. In addition, the second order Runge-Kutta method requires two stages of differential equation evaluation while Eulers method requires a single evaluation of the differential equation. Therefore, one could say that the approximation of the solution generated by the Runge-Kutta method gives better solutions by comparing with the Euler method for a constant step $h = 0.1$ and $x\ (0) = 0$ and $y\ (0) = 1$. By evaluating the approximation of the following differential equation: $y' = y + 1$, we get the results shown in Fig. 1.

The use of a precise method to approximate a solution for a differential equation is necessary to solve the continuous Hopfield network. To overcome the weakness of the classical neural network we adopted the second order Runge Kutta method.

By combining this method in Hopfield continuous network we can improve the convergence of this network towards a better attractor. The following algorithm is proposed to ensure convergence.

```
Proposed algorithm
Input
-Graph $G = (V, E)$
-Weight matrix and bias vector;
-Initialize system settings.
-Use of the second order Runge Kutta method.
-$u_i \leftarrow$ Initial solution generated randomly.

Start
    1) $\Delta E = E(u_{n+1}) - E(u_n)$
    2) $\overline{u}_{n+1} = u_n + hf(v_n, u_n)$
    3) $u_{n+1} = u_n + \frac{h}{2}\left(f(v_n, u_n) + f(v_{n+1}, \overline{u}_{n+1})\right)$
    4) $v_n = 1/2[tanh\left(\frac{u_i}{u_0}\right) + 1]$
The stopping criterion is false
End
Output: Maximum stable subsystem
```

In the proposed algorithm, we have introduced a new method to refine the solution obtained by the Euler method in order to guarantee a better approximation. First, we evaluate the differential equation for an approximate solution. Secondly, we refine the solution found by a second evaluation of the differential equation. The criterion for stopping this network is to check if the previous state is the same as the current state.

IV. SIMULATION RESULTS

The theoretical results presented in this article are evaluated with a set of reference instances. A number of neurons and a number of constraints between each pair of neurons characterize each instance. To approve the effectiveness of the proposed method we used three groups of instances. The proposed algorithm was able to converge towards a point of stability using the energy function characterizing the continuous network of neurons. This algorithm has been implemented on a Core $i7$ $8G$ Ram desktop computer. The programming language used is the java language. The setting used is proposed only once for all instances to validate the capacity of the new method.

Each instance is represented as an input for our model to calculate stable points. In addition, the initial states are generated from the following formulation:

$$x_i = 0.999 + \frac{n+1-i}{n} 10^{-5} t. \qquad (22)$$

The variable i represents the index number of each neuron. The total number of neurons is n. The variable t is a random uniform variable in the interval [0.5,0.5].

To validate the proposed method, it turned out to be necessary to use multiple instances. We performed three experiments on brock, p_hat and hamming/johnson instances. The brock instance has a number of neurons ranges from 200 to 800 and the number of constraints ranges from 9876 to 208166. The instance p_hat has a number of neurons ranges from 300 to 1500 and a number of constraints is between 10933 and 568960. The hamming/johnson instance has a number of neurons between 28 and 1024 and a number of constraints is between 210 and 518656. For this experiment, we have performed 100 tests to give the average number of stable points for each instance. The following figures represent the results of the convergence of the proposed approach towards an optimal solution. The recorded result for each instance is the number of stable points using the continuous Hopfield lattice combined with a new discretization method. From the observation of all the figures, we can confirm that the proposed approach gives a better convergence with respect to its classical method antecedent. In particular, the optimal solution is well obtained with the higher complexity graphs.

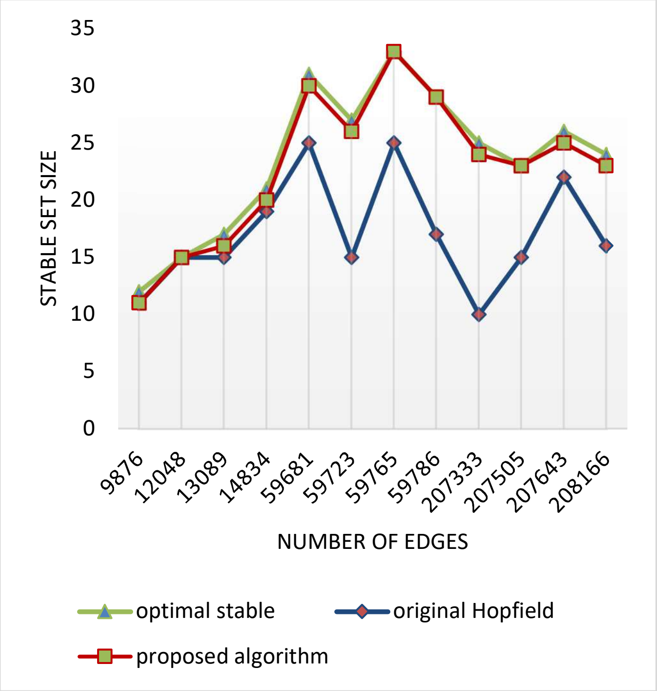

Figure 2. Brock instance with a number of neurons varies between 200 to 800

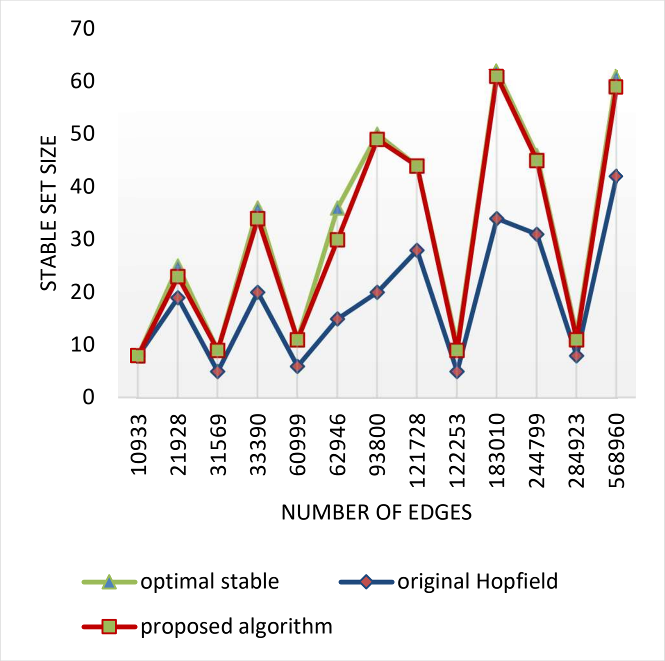

Figure 3. p_hat instance with number of neurons varies between 300 to 1500

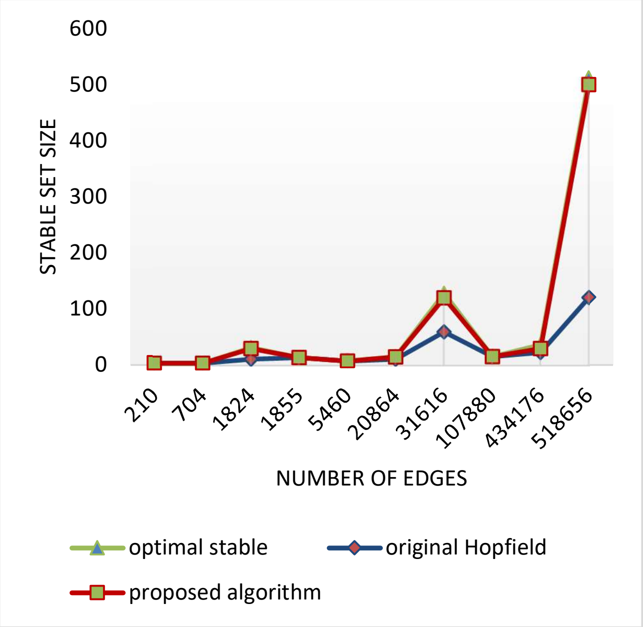

Figure 4. Hamming/Johnson instance with a number of neurons varying between 28 and 1024

V. CONCLUSION

In this work, the dynamic system of artificial neural network is discretized by a new method named Runge-Kutta to guarantee better stability of Hopfield neural network. This method is guided by two important phases. The first phase is to find an approximation of the solution and the second phase seeks to refine the solution to ensure a better approximation. In order to verify the performance of the proposed approach benchmarks are used in the experiments. The experimental results show that the proposed method can effectively help artificial neural networks to find best solutions on the majority of instances. Like a new research direction, the proposed approach can help to solve many optimization problems such as the optimization of database queries.

References

[1] J. J. Hopfield, "Neural networks and physical systems with emergent collective computational abilities," *Feynman Comput.*, vol. 79, no. 4, pp. 7–19, 2018, https://doi.org/10.1201/9780429500459.

[2] D. Calabuig, S. Gimenez, J. E. Roman, and J. F. Monserrat, "Fast hopfield neural networks using subspace projections," *Neurocomputing*, vol. 73, no. 10–12, pp. 1794–1800, 2010, https://doi.org/10.1016/j.neucom.2009.12.031.

[3] P. M. Talaván and J. Yáñez, "A continuous Hopfield network equilibrium points algorithm," *Comput. Oper. Res.*, vol. 32, no. 8, pp. 2179–2196, 2005, https://doi.org/10.1016/j.cor.2004.02.008.

[4] R. D'Ambrosio and S. Di Giovacchino, "Nonlinear stability issues for stochastic Runge-Kutta methods," *Commun. Nonlinear Sci. Numer. Simul.*, vol. 94, p. 105549, 2021, https://doi.org/10.1016/j.cnsns.2020.105549.

[5] M. Fitzsimmons and H. Kunze, "Combining Hopfield neural networks, with applications to grid-based mathematics puzzles," *Neural Networks*, vol. 118, pp. 81–89, 2019, https://doi.org/10.1016/j.neunet.2019.06.005.

[6] S. G. Hu *et al.*, "Associative memory realized by a reconfigurable memristive Hopfield neural network," *Nat. Commun.*, vol. 6, no. May, pp. 1–5, 2015, https://doi.org/10.1038/ncomms8522.

[7] D. Stauffer, A. Aharony, L. Da Fontoura Costa, and J. Adler, "Efficient Hopfield pattern recognition on a scale-free neural network," *Eur. Phys. J. B*, vol. 32, no. 3, pp. 395–399, 2003, https://doi.org/10.1140/epjb/e2003-00114-7.

[8] J. Li, J. Yang, and W. Wu, "Stability and periodicity of discrete Hopfield neural networks with column arbitrary-magnitude-dominant weight matrix," *Neurocomputing*, vol. 82, pp. 52–61, 2012, https://doi.org/10.1016/j.neucom.2011.10.025.

[9] S. Kumar and M. P. Singh, "Pattern recall analysis of the Hopfield neural network with a genetic algorithm," *Comput. Math. with Appl.*, vol. 60, no. 4, pp. 1049–1057, 2010, https://doi.org/10.1016/j.camwa.2010.03.061.

[10] X. Y. Wang and Z. M. Li, "A color image encryption algorithm based on Hopfield chaotic neural network," *Opt. Lasers Eng.*, vol. 115, no. July 2018, pp. 107–118, 2019, https://doi.org/10.1016/j.optlaseng.2018.11.010.

[11] S. Liu, W. Jiang, X. Li, and X. F. Zhou, "Lyapunov stability analysis of fractional nonlinear systems," *Appl. Math. Lett.*, vol. 51, pp. 13–19, 2016, https://doi.org/10.1016/j.aml.2015.06.018.

[12] J. Tsinias, N. Kalouptsidis, and A. Bacciotti, "Lyapunov functions and stability of dynamical polysystems," *Math. Syst. Theory*, vol. 19, no. 1, pp. 333–354, 1986, https://doi.org/10.1007/BF01704919.

[13] X. S. Yang, "Optimization and metaheuristic algorithms in engineering," *Metaheuristics Water, Geotech. Transp. Eng.*, pp. 1–23, 2013, https://doi.org/10.1016/B978-0-12-398296-4.00001-5.

[14] M. Campelo and R. Correa, "A Lagrangian relaxation for the maximum stable set problem," *CoRR*, vol. abs/0903.1, pp. 1–12, 2009.

[15] L. Zuo, B. Zhang, and S. Zhang, "The k-path vertex cover in product graphs of stars and complete graphs," *Int. J. Appl. Math.*, vol. 46, no. 1, 2016.

[16] H. Jiang, C. M. Li, and F. Manyà, "An exact algorithm for the maximum weight clique problem in large graphs," *Proceedings of the 31st AAAI Conf. Artif. Intell. AAAI 2017*, vol. 9, November 2017, pp. 830–838, https://doi.org/10.1609/aaai.v31i1.10648.

[17] G. Gruber and F. Rendl, "Computational experience with stable set relaxations," *SIAM J. Optim.*, vol. 13, no. 4, pp. 1014–1028, 2003, https://doi.org/10.1137/S1052623401394092.

[18] C. Friden, A. Hertz, and D. de Werra, "STABULUS: A technique for finding stable sets in large graphs with tabu search," *Computing*, vol. 42, no. 1, pp. 35–44, 1989, https://doi.org/10.1007/BF02243141.

[19] I. Miguel and Q. Shen, "Solution techniques for constraint satisfaction problems: Advanced approaches," *Artif. Intell. Rev.*, vol. 15, no. 4, pp. 269–293, 2000, https://doi.org/10.1023/A:1011096320004.

[20] A. Haselböck, "Exploiting interchangeabilities in constraint satisfaction problems," *Ijcai*, pp. 282–287, 1993, [Online]. Available at: http://www.cse.unl.edu/~choueiry/Documents/Haselboeck.pdf.

[21] M. Jarke and J. Koch, "Query optimization in database systems," *ACM Comput. Surv.*, vol. 16, no. 2, pp. 111–152, 1984, https://doi.org/10.1145/356924.356928.

[22] N. Zealand, "History of Runge-Kutta methods," vol. 20, pp. 247–260, 1996, https://doi.org/10.1016/0168-9274(95)00108-5.

Mohammed El Alaoui has a Ph.D. in computer science from the Faculty of Sciences and Technology of Fez, Morocco. He is a member of Modelling and Scientific Computing Laboratory at FST. His research interests are database query optimization, computer science, artificial neural networks architectures, constraint satisfaction problem, optimization, artificial intelligence.

Mohamed Ettaouil is a Doctorate Status in Operational Research and Optimization, FST University Sidi Mohamed Ben Abdellah, Fez. Ph.D. in Computer Science, University of Paris 13, Galilee Institute, Paris France. He is a professor at the Faculty of Science and technology of Fez FST, and he was responsible for research team in modelization and pattern recognition, operational research and global optimization methods.