An Adaptive Hybrid Approach: Combining Neural Networks and Simulated Annealing to Calculate the Equilibrium Point in Max-stable Problem

Mohammed El Alaoui, and Mohamed Ettaouil

***Abstract*—The simulated annealing algorithm is a stochastic optimization method for solving optimization problems associated with a large search space. The efficiency of the algorithm depends on the adaptation of the cooling model. Despite this, the main disadvantage of this algorithm is that it does not take into account the state of the system while searching. Thus, it is difficult to predict the system convergence with the simulated annealing algorithm. However, neural networks in particular continuous Hopfield networks have proven their ability in the field of machine learning to make a decision. In this paper, we introduce continuous Hopfield networks to improve the convergence of the simulated annealing algorithm. The experimental results show that the hybrid approach produces a large number of stable sets.**

***Index Terms*— Neural networks, continuous Hopfield networks, simulated annealing algorithm**

I. Introduction

Simulated annealing (SA) is a heuristic method inspired by thermodynamics, or more exactly, by static physics. It is based on an analogy to the physical annealing of solids, which involves heating a material at a high temperature and cooling it very slowly in order to let the system reach its minimum energy. If the cooling process is slow, a crystal structure is formed from a network of well-ordered nodes.

The history of the simulated annealing method goes back to the year 1953. At that time, N. Metropolis developed an algorithm to simulate the establishment of equilibrium in a system with several degrees of freedom and at a given temperature [1]. In 1983, Kirkpatrick mimicked the physical annealing behavior of solids to solve some optimization problems [2].

Currently, the annealing method is used to solve many optimization problems, like in economics, image processing and in many other combinatorial problems. Often, the simulated annealing method is used to train neural networks, specifically the multilayer optimization problem of perceptron [3].

Manuscript received August 29, 2020; revised January 27, 2021.

Mohammed El Alaoui obtained his PhD degree from University Sidi Mohammed ben Abdellah, Fez, MAROCCO. Email: (md.elalaoui@gmail.com).

Mohamed Ettaouil is a professor at the Faculty of Science and technology, University Sidi Mohammed ben Abdellah, Fez, MAROCCO. Email: (mohamedettaouil@yahoo.fr).

The advantage of the simulated annealing method lies in the possibility of avoiding the local minima of the function to be optimized and of continuing the search for the global minimum. This is achieved through the adoption of an important set of parameters that governs the convergence of SA. This set includes the cooling model, the initial and the final temperature. The cooling pattern may seem to be the most important in this set. It drives the algorithm from an initial state to another state. If the temperature drops quickly, the algorithm gets stuck in a local minimum. However, with a well-controlled temperature, the algorithm can avoid the local minimum. The cooling model is widely studied by many researchers [4]. Whereas, the main disadvantage of this algorithm is that it does not take into account the state of the system while searching. Thus, it is difficult to predict the system convergence with the simulated annealing algorithm. However, neural networks in particular continuous Hopfield networks have proven their ability in the field of machine learning to make a decision. In this paper, we introduce continuous Hopfield networks to improve the convergence of the simulated annealing algorithm.

Continuous Hopfield Networks (CHN) is a recurrent neural network. CHN are the most successful approach in many areas such as image processing and automatic recognition [5]. The success of CHN is due to their ability to adapt with many optimization issues. The main idea of our approach is to adopt the behavior of continuous Hopfield networks with simulated annealing to guarantee better convergence.

This paper presents a new approach to improve the convergence of the annealing algorithm with the use of CHN. To evaluate the hybrid approach, we model the maximum stable problem as a quadratic problem and then as a Hopfield quadratic energy. Then we proceed to a process of combining the two algorithms on different instances. The rest of this paper is organized like this. First, Section 2 describes the simulated annealing algorithm and continuous Hopfield networks. In section 3, the hybrid 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. Simulated annealing and continuous Hopfield networks

This section presents the simulated annealing approach and continuous Hopfield networks for the Max-stable problem. We also present some cooling models for the simulated annealing algorithm. Then, CHN is represented as a powerful approach because it has a dual use, either as an associative memory or as a tool to solve optimization problems. We also discuss the necessary condition to find a point of equilibrium for this network. First, we give a description of the simulated annealing approach.

A. Simulated annealing

The annealing method is used to find the global minimum of the function $f(x)$ from a certain discrete or continuous space S. The elements of the set S are states of an imaginary physical system and the value of the function f is used as the energy of the system $E = f(x)$. At each instant, the temperature of the system T is assumed to be regulated, generally decreasing with time. The system passes from one stage to another according to a generator family, generated by x and T. This passage mainly uses the Metropolis rule to decide whether a new candidate is accepted as a solution. Indeed, the system in state i is characterized by energy E_i. And a disturbance of the system generating a new energy E_{i+1} which is accepted if $E_{i+1} - E_i < 0$. However, the success of simulated annealing relies on accepting new states, even when its energy E_{i+1} does not meet the condition $E_{i+1} - E_i < 0$. In this case, the acceptance of the state $i+1$ is done only under the probability $P = \exp\left(\frac{E_i - E_{i+1}}{T^*}\right)$.where T^* is a control parameter, usually called temperature. In summary, in the simulated annealing algorithm, the acceptance of new solutions is governed by the rule:

$$P = \begin{cases} 1 & if \quad E_i - E_{i+1} < 0 \\ \exp\left(\frac{E_i - E_{i+1}}{T^*}\right) & otherwise \end{cases} \quad (1)$$

Figure 1 explains the behaviour of the simulated annealing algorithm to determine the optimal solution. In addition, the cooling models are responsible for the convergence of the algorithm so that a neighbouring solution can be accepted.

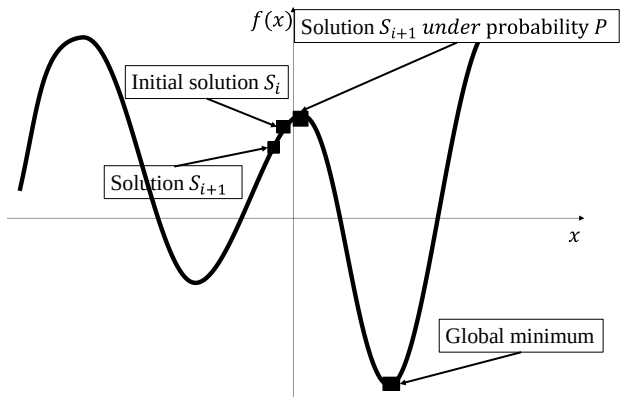

Fig. 1. Energy of the system in state i and i+1

The energy E_i and E_{i+1} specifically designates the value that the objective function takes in iteration i and $i+1$. As shown in Figure 1, significantly worse solutions are accepted than the good quality solution when the temperature is high. On the other hand, when the temperature is low, the solutions that improve the quality are the only ones accepted. Finally, when the temperature tends to zero, new solutions are only accepted if they improve on the highest quality solution found by the algorithm.

Algorithm 1 shows the general structure of the simulated annealing algorithm. In addition, the steps followed by the algorithm to achieve its thermal stability for each temperature value T before decreasing it according to a cooling model.

Algorithm 1 Simulated annealing algorithm

Input:
$T_f \leftarrow$ Final temperature.
$s \leftarrow$ Initial solution generated randomly.
$T \leftarrow$ Initial temperature.

Start:
The stopping criterion is false
$s' \leftarrow$ Generate a neighboring solution of the current solution s.
$\Delta E = E(s') - E(s)$
If $\Delta E \le 0$ **then**
$s' \leftarrow s$
End if
If random$[0,1] < \exp\left(\frac{-\Delta E}{T}\right)$ **then**
$s' \leftarrow s$
End if
$T \leftarrow f(T)$

End
Output s^*, where s^* is the best solution obtained.

The success of the simulated annealing algorithm depends in part on how quickly the system is cooled. Very rapid cooling may not allow the particles time to properly reorganize. This can generate inconsistencies in the materials. For example, a very sharp decrease in the temperature applied to a glass could generate the presence of bubbles, which would make the material more fragile and more likely to fracture. However, using such slow cooling models promotes the quality and hardness of the material. In short, materials of high quality and hardness are sought after in the shortest possible time. This combinatorial optimization process would imply that very large temperature drops prevent the algorithm from adequately exploring the space for possible solutions. Thus, the simulated annealing algorithm will return poor quality solutions, which could have been improved if the temperature had been lowered in a well-controlled manner. However, very slow cooling will result in longer run times for the algorithm. In fact, the slower the cooling time of the system, the better the chance of finding the best solutions.

One of the most important aspects of simulated annealing is cooling models. In addition, a lot of research is being carried out around this question. As already indicated, the performance of the SA algorithm strongly depends on the selected cooling model. A good model could help find good solutions for various combinatorial optimization problems. Many theoretical cooling models have been tested. Therefore,

in many cases heuristic and empirical reasoning is used to propose models to balance execution times with the quality of the solutions obtained.

The geometric model, $T_k = \alpha T_{k-1}, 0 < \alpha < 1$, is most used to perform cooling when applying the SA algorithm in various optimization problems. Simplicity and its efficiency may be the reasons for this preference [6].

A slower decrease in the geometric model ensures that the system is more likely to reach equilibrium in each of the temperature states.

The logarithmic model, $T_{k+1} = \frac{c}{\log(k+1)}$, is less frequent than the geometric model. However, it has a theoretical importance because it allows establishing rigorous proofs of the convergence of the SA algorithm [7]. Subsequently, the Lundy-Mees model is based on the idea that the temperature can be decreased from the previous temperature, according to the following model: $T_{k+1} = \frac{T_k}{1+\beta T_k}$ with the parameter $\beta = \frac{T_0 - T_f}{M.T_0.T_f}$ [8].

T_0 is the initial temperature, T_f is the final temperature and M is the total number of iterations.

The main mission of each model is to cool the system from the initial temperature T_0 to a final temperature T_f.

B. Continuous Hopfield Network

Continuous Hopfield Networks is a fully connected neural network with a symmetrical matrix of connections. In the process of convergence, the dynamics of these networks converge towards one of the equilibrium positions. These equilibrium positions are determined in advance during the learning process. These are local minima of the functional energy of the network. Such a network can be used both as an associative memory and to solve certain optimization problems [9]. Unlike many neural networks, which operate until a response is received after a certain number of iterations, Hopfield networks operate until equilibrium is reached when the next state of the network is the same as the previous one.

Continuous Hopfield Networks are made up of n interconnected neurons with an activation function called hyperbolic tangent. The dynamics of CHN is described by the following differential equation:

$$\frac{du}{dt} = -\frac{u}{\tau} + Tv + i^b \tag{2}$$

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

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

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

$$v_i = -\frac{1}{2}\left(1 + \tanh\left(\frac{u_i}{u_0}\right)\right),\ u_0 > 0 \tag{3}$$

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

Hopfield proved that the symmetry of the zero-diagonal matrix T is a sufficient condition for the existence of the Lyapunov function [10], therefore, the existence of the equilibrium point is guaranteed. Continuous Hopfield Networks will solve combinatorial problems that have an energy function. The next step is to discretize the continuous Hopfield lattice with the Euler method. In the second step, we present a modelling of the maximum stable set problem as a 0-1 quadratic programming. From this model, we use the dynamics of continuous Hopfield networks for the improvement of the convergence of the simulated annealing method.

III. Proposed hybrid approach

The hybrid approach combines CHN and simulated annealing. The hybrid approach process has two phases. The first phase corresponds to a quadratic modelization of CHN for the max-stable problem like an energy function. The second phase corresponds to incorporating the CHN in the simulated annealing method. First, we begin the presentation of the MSP problem and the formulation of the energy function associated with this problem. Then, we select a practical setting of this function. Next, a search algorithm based on the simulated annealing incorporated with the CHN is shown.

A. Quadratic program for max-stable

The max-stable problem can be represented as an undirected graph $G = (V, E)$ 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 sub graph generated by S does not contain any arc.

The problem of the maximal stable set (MSSP) consists in finding a stable set in the graph G of maximal cardinality α (G). Besides its theoretical interest, the MSSP problem arises in information retrieval, experimental design and computer vision applications [11].

The stable set problem is strong NP-hard, and even difficult to approximate [12]. The MSSP problem can be solved by using polynomial time algorithms for special classes such as perfect graphs, graphs with long odd cycles and graphs of stars [13]. But, the existence of a polynomial time algorithm for arbitrary graphs seems unlikely.

Different approaches have been discussed in the literature to exactly solve the maximum stable set problem. Carrahan and Pardalos propose an implicit enumeration technique [14]. Gruber and Rendl have reported computational results for different stable set linear programming relaxations [15]. An effective evolution of the taboo research approach is presented in the original work of Friden, Hertz and de Werra [16].

To solve the MSSP problem via 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:

$$x_i = \begin{cases} 1 & if \quad v_i \in S \\ 0 & otherwise \end{cases} \tag{4}$$

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

$$(v_i, v_j) \in E \Rightarrow x_i x_j = 0 \tag{5}$$

The constraints can be expressed in the following form:

$$h(x) = \sum_{i=1}^{n}\sum_{j=1}^{n} b_{ij} x_i x_j = 0$$
$$with \quad b_{ij} = \begin{cases} 1 & if \ (v_i, v_j) \in E \\ 0 & otherwise \end{cases} \tag{6}$$

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:

$$(QP) = \begin{cases} Min \ f(x) = -\sum_{i=1}^{n} x_i \\ sibject \ to \\ h(x) = \sum_{i=1}^{n}\sum_{j=1}^{n} b_{ij} x_i x_j \\ x \in \{0,1\}^n \end{cases} \tag{7}$$

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

$$E(v) = -\alpha \sum_{i=1}^{n} v_i + \frac{1}{2}\phi \sum_{i=1}^{n}\sum_{j=1}^{n} b_{ij} v_i v_j + \gamma \sum_{i=1}^{n} v_i (1 - v_i) \tag{8}$$

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

$$T_{ij} = -\phi b_{ij} + 2\delta_{ij}\gamma \tag{9}$$

The Kronecker symbol is given as follows:

$$\delta_{ij} = \begin{cases} 1 & if \ i = j \\ 0 & if \ i \neq j \end{cases} \tag{10}$$

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

$$\frac{\partial E}{\partial v_i} = -\alpha + \phi \sum_{i=1}^{n} b_{ij} v_j + \gamma (1 - 2v_i) \tag{11}$$

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

B. Hybrid approach

Using just one method to solve a complex problem does not always lead to success. In a hybrid architecture that combines several paradigms, the effectiveness of one approach can compensate for the weakness of another.

By combining two approaches, we can get around the drawbacks inherent in each of them. One of the promising directions for the creation of hybrid systems is the joint use of technologies such as continuous Hopfield networks and simulated annealing. We propose the following algorithm for the hybrid approach.

Algorithm 2 Proposed algorithm

Input:

→ The graph $G = (V, E)$

→ The weight matrix and bias vector

→ Use of Euler's method for discretization.

→ u_i initial solution generated randomly.

→ $f(T)$ is the cooling model that describes the process of transforming a system from an initial state to an end state.

Output:

→ Maximum stable problem set

Start:

The stopping criterion is false

$\Delta E = E(u_{i+1}) - E(u_i)$

If $\Delta E \leq 0$ **then**

$u_{i+1} = u_i - hf(u_i)$

$f(u_i) = -T \tanh(u_i) + I$

End if

If random $[0,1] < \exp\left(\frac{-\Delta E}{f(T)}\right)$ **then**

$u_{i+1} = u_i - hf(u_i)$

End if

End.

The physical analogy used to justify simulated annealing suggests that the cooling rate is low enough to distribute the probabilities of the current state of being close to thermodynamic equilibrium at all times. Unfortunately, relaxation time, the time for equilibrium to be re-established after temperature changes, is highly dependent on energy function and current temperature. In the proposed algorithm, the relaxation time also depends on the cooling model. Together these parameters are generally provided as black box functions for the proposed algorithm. Therefore, the ideal cooling rate cannot be determined in advance and must be adjusted empirically for each task. The proposed algorithm to solve the problem connects on the one hand to the cooling system according to the research progress graph. On the other hand, the proposed algorithm combines the operation of continuous Hopfield networks so that the output of each neuron is calculated based on the other neurons. This gives consistency between all the states of the system.

IV. Simulation results

In the first experiment, we evaluate the efficiency of three cooling models. Each model is associated with its own parameter. All of these parameters α, c and β are determined by experience. The best value of each parameter will be recorded to study the convergence of the simulated annealing method.

Table 1 summarizes the cooling models implemented in the simulated annealing algorithm in the context of the maximum stable problem. Likewise, it shows the parameters used in each model.

TABLE I
COOLING MODELS AND PARAMETER VALUES USED

Model	Form	parameter
Geometric	$T_{k+1} = \alpha T_k$	$\alpha = 0.99$
Logarithmic	$T_{k+1} = \frac{c}{\log(k+1)}$	$c = 3.5$
Lundy-Mees	$T_{k+1} = \frac{T_k}{1+\beta T_k}$	$\beta = \frac{T_0 - T_f}{M.T_0.T_f}$

The values of these parameters are used to study the operation of the simulated annealing method for the maximum stable problem.

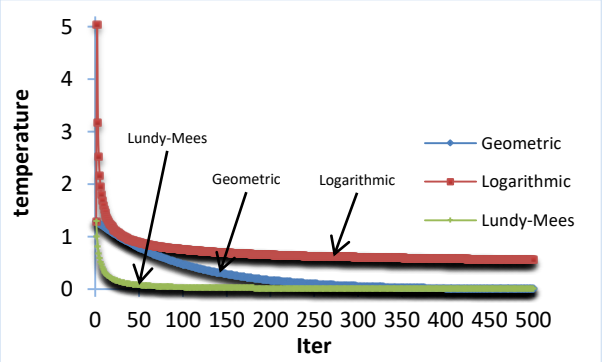

Fig. 2. The behaviour of three cooling models

Figure 2 shows the results of the Geometric, Logarithmic and Lundy-Mees model with the use of the parameters mentioned in Table 1. The Logarithmic model requires a large and impractical number of iterations for the temperature to become cold, i.e. from the initial temperature T_0 to the final temperature T_f. Despite the importance of the Geometric model to converge towards an optimal solution, but this requires a higher number of iterations. Faced with the drawbacks of other models, the Lundy-Mees model is represented as the best cooling model in terms of convergence.

In order to show the practical interest of the hybrid approach proposed in this article. We worked on a series of experiments to solve the problem of max stable sets. Most of the graphs are taken from the 2nd DIMACS Challenge [17].

These graphs have been provided as test problems to resolve the maximum click problem. We took the complement of these graphs and applied our maximum stable ensemble approach. The results are provided in table II. The result was recorded using a desktop computer (Intel Core i7, 2.9 GHz and 8 GB of RAM) running the Java programming language.

The initial states are generated randomly:

$$x_i = 0.999 + \frac{n+1-i}{n} 10^{-5} t \qquad (12)$$

Where $i = 1, \cdots, n$ and t is a random uniform variable in the interval $[0,1]$. Recall that, n is the number of the nodes.

TABLE II
INSTANCES FOR THE MAXIMUM STABLE PROBLEM.

Instance	V	$\|E\|$	$\alpha(G)$	$\alpha_1(G)$	$\alpha_2(G)$	$\alpha_3(G)$
brock200_2	200	9 876	12	10	5	10
brock200_4	200	13 089	17	17	9	17
brock400_2	400	59 786	29	20	11	25
brock400_4	400	59 765	33	33	17	33
brock800_4	800	207 643	26	26	13	26
gen200_p0.9_44	200	17 910	44	38	10	44
gen200_p0.9_55	200	17 910	55	55	22	22
gen400_p0.9_55	400	71 820	55	55	10	55
gen400_p0.9_75	400	71 820	75	75	36	75
hamming10-4	1 024	434 176	40	20	22	30
hamming8-4	256	20 864	16	10	3	10
keller4	171	9 435	11	11	6	11
keller5	776	225 990	27	23	19	27
p_hat300-1	300	10 933	8	8	2	8
p_hat300-3	300	33 390	36	36	22	36
p_hat700-1	700	60 999	11	11	4	11
p_hat700-2	700	121 728	44	20	13	26
p_hat1500-1	1 500	284 923	12	12	12	12
DSJC1000_5	1 000	499 652	15	15	2	15
DSJC500_5	500	125 248	13	13	8	13
MANN_a27	378	70 551	126	126	20	126
MANN_a45	1 035	533 115	345	311	238	345

$\alpha_1(G)$: The size of the stable set obtained by hybrid approach combined with the Geometric model.
$\alpha_2(G)$: The size of the stable set obtained by hybrid approach combined with the Logarithmic model.
$\alpha_3(G)$: The size of the stable set obtained by hybrid approach combined with the Lundy &Mees model.
To achieve these results, the machine required 500 steps with the hybrid approach (HA). This table shows that the result is better when the hybrid approach is combined with the Lundy &Mees model. In fact, the system (hybrid approach + Lundy &Mees) produces a large number of stable sets comparing with other geometric and logarithmic models. Furthermore, by comparing the convergence of the algorithm of the hybrid approach combined with the Lundy &Mees model, we can note that the system cools in the first iterations and converges quickly when comparing itself with the other models.

This reflects the effectiveness of the hybrid approach combined with Lundy &Mees.

Now we turn to the solution of the proposed approach with the use of different cooling models. For a good visualization, it is important to take into account some essential aspects, such as the number of nodes, the number of edges and the maximum number of the stable set. The diagrams, shown in figure 3 and 4, capture a comparison between the different instance types.

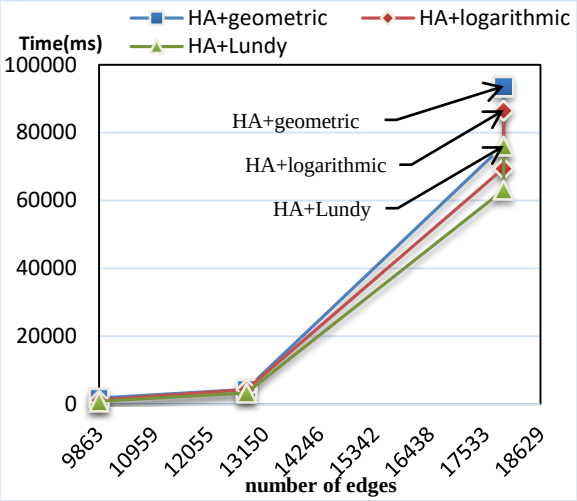

Fig. 3. Instances of 200 nodes

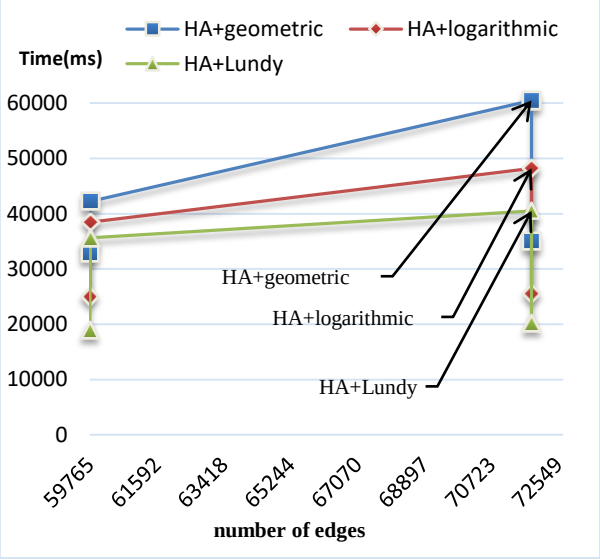

Fig. 4. Instances of 400 nodes

This new approach outperforms the results achieved by other methods in almost all instances with execution time taken into consideration. Undoubtedly, the way the elements of a graph are organized influences the execution time. For instances of 200 and 400 nodes shown in figure 3 and 4 respectively, the calculation time is strictly related to the structure of each graph. Another point that can be observed is that the calculation time is closely related to the maximum number of the stable set and the density of the instance.

V. Conclusion

In this paper, a hybrid simulated annealing approach combined with continuous Hopfield networks is presented. This approach introduced better performance than other methods' in large-scale test problems. This study has shown experimentally that in large-scale problems, optimization by the proposed approach gives good results, in addition to the fact that the proposed approach takes into account the state of the system during convergence. This is achieved with simulated annealing hybridization and continuous Hopfield gratings. This hybridization makes it possible to control the convergence towards an optimal solution. In the next perspective, this approach will be applied to the constraint programming problem and the view selection problem for query optimization in databases.

References

[1] N. Metropolis, A. W. Rosenbluth, M. N. Rosenbluth, A. H. Teller, and E. Teller, "Equation of state calculations by fast computing machines," *J. Chem. Phys.*, vol. 21, no. 6, pp. 1087–1092, 1953, doi: 10.1063/1.1699114.

[2] S. Kirkpatrick, C. D. Gelatt, and M. P. Vecchi, "Optimization by simulated annealing," *Science (80-.).*, vol. 220, no. 4598, 1983.

[3] P. A. Castillo, J. J. Merelo, J. González, V. Rivas, and G. Romero, "SA-prop: Optimization of multilayer perceptron parameters using simulated annealing," *Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics)*, vol. 1606, pp. 661–670, 1999, doi: 10.1007/BFb0098224.

[4] Y. Nourani and B. Andresen, "A comparison of simulated annealing cooling strategies," *J. Phys. A. Math. Gen.*, vol. 31, no. 41, pp. 8373–8385, 1998, doi: 10.1088/0305-4470/31/41/011.

[5] P. Campadelli, D. Medici, and R. Schettini, "Color image segmentation using Hopfield networks," *Image Vis. Comput.*, vol. 15, no. 3, pp. 161–166, 1997, doi: https://doi.org/10.1016/S0262-8856(96)01121-3.

[6] D. Henderson, S. H. Jacobson, and A. W. Johnson, "The Theory and Practice of Simulated Annealing BT - Handbook of Metaheuristics," F. Glover and G. A. Kochenberger, Eds. Boston, MA: Springer US, 2003, pp. 287–319.

[7] Y. Nourani and B. Andresen, "A comparison of simulated annealing cooling strategies," *J. Phys. A. Math. Gen.*, vol. 31, no. 41, pp. 8373–8385, 1998, doi: 10.1088/0305-4470/31/41/011.

[8] K. A. Dowsland, "Some experiments with simulated annealing techniques for packing problems," *Eur. J. Oper. Res.*, vol. 68, no. 3, pp. 389–399, 1993, doi: https://doi.org/10.1016/0377-2217(93)90195-S.

[9] J. J. Hopfield and D. W. Tank, "'Neural' computation of decisions in optimization problems.," *Biol. Cybern.*, vol. 52, no. 3, pp. 141–152, 1985, doi: 10.1007/BF00339943.

[10] F. H. Clarke, Y. S. Ledyaev, and R. J. Stern, "Asymptotic Stability and Smooth Lyapunov Functions," *J. Differ. Equ.*, vol. 149, no. 1, pp. 69–114, 1998, doi: https://doi.org/10.1006/jdeq.1998.3476.

[11] L. Babel, "Finding maximum cliques in arbitrary and in special graphs," *Computing*, vol. 46, no. 4, pp. 321–341, 1991, doi: 10.1007/BF02257777.

[12] J. Håstad, "Clique is hard to approximate within n1-ε," *Acta Math.*, vol. 182, no. 1, pp. 105–142, 1999, doi: 10.1007/BF02392825.

[13] Liancui Zuoy, Bitao Zhang, and Shaoqiang Zhang, "The k-Path Vertex Cover in Product Graphs of Stars and Complete Graphs," IAENG International Journal of Applied Mathematics, vol. 46, no. 1, pp. 97-103, 2016.

[14] R. Carraghan and P. M. Pardalos, "An exact algorithm for the maximum clique problem," *Oper. Res. Lett.*, vol. 9, no. 6, pp. 375–382, 1990, doi: https://doi.org/10.1016/0167-6377(90)90057-C.

[15] G. Gruber and F. Rendl, "Computational Experience with Stable Set Relaxations," *SIAM J. Optim.*, vol. 13, no. 4, pp. 1014–1028, Jan. 2003, doi: 10.1137/S1052623401394092.

[16] 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, doi: 10.1007/BF02243141.

[17] "Center for Discrete Mathematics and Theoretical Computer Science," 1992. http://archive.dimacs.rutgers.edu/pub/challenge/graph/benchmarks/clique/.