17th International Conference on Advanced Computational Intelligence (ICACI 2025)
Bath, UK, July 7-13, 2025

2025 17th International Conference on Advanced Computational Intelligence (ICACI) | 979-8-3315-0979-8/25/$31.00 ©2025 IEEE | DOI: 10.1109/ICACI65340.2025.11096345

A Collaborative Neurodynamic Optimization Algorithm Based on Boltzmann Machines and 2-Opt Heuristic for Solving the Traveling Salesman Problem

Hongzong Li
Department of Computer Science
City University of Hong Kong
Kowloon, Hong Kong
hongzli2-c@my.cityu.edu.hk

Jun Wang
Department of Computer Science
& Department of Data Science
City University of Hong Kong
Kowloon, Hong Kong
jwang.cs@cityu.edu.hk

***Abstract*—The traveling salesman problem is a well-known challenge in combinatorial optimization. It involves determining the shortest possible route that visits each city exactly once from a given list, returning to the starting point with the least total distance traveled. It has extensive applications in logistics, planning, and routing. It is a well-known NP-hard optimization problem. In this paper, the traveling salesman problem is formulated as a quadratic assignment problem with constraints to eliminate excessively long paths for enhancing efficiency. We introduce a collaborative neurodynamic optimization algorithm for solving traveling salesman problems with Boltzmann machines with momentum term and the 2-opt heuristic. The proposed algorithm consists of a phase with BMm's and another phase with 2-opt heuristics. It leverages multiple BMm's and 2-opt heuristics, and a particle swarm optimization update rule to re-initialize BMm's for escaping from local optima and moving toward global optimal solutions. We demonstrate its superior performance against two baselines in terms of the objective function values.**

***Index Terms*—Traveling salesman problem, combinatorial optimization, quadratic assignment problem, Boltzmann machine, 2-opt heuristic, collaborative neurodynamic optimization.**

I. Introduction

The traveling salesman problem (TSP) is a classical and extensively studied problem in combinatorial optimization and computer science. Given a list of cities and the distances between each pair of them, the objective of the TSP is to find the shortest possible route that visits each city exactly once and returns to the origin city. Despite its simple statement, the TSP is known to be NP-hard, meaning that there is no known polynomial-time algorithm to solve it [1]. The TSP has significant practical applications in various domains, such as logistics and distribution, where it models the problem of finding the most efficient route for a delivery vehicle [2]. Other applications include job sequencing in manufacturing systems [3], printed circuit boards drilling [4], wallpaper cutting [5], and warehouse order-picking [6].

Due to its computational complexity, exact algorithms for the TSP, such as the branch and bound algorithm [5] and the dynamic programming algorithm [7], are only feasible for small instances. For larger problems, heuristic and metaheuristic algorithms have been developed to find near-optimal solutions within reasonable computational times. The heuristic method includes the nearest neighbor algorithm [8], the insertion heuristic algorithm [9], and the 2-opt algorithm [10]. The metaheuristic method includes the genetic algorithm [11], the ant colony optimization algorithm [12], and the particle swarm optimization algorithm [13].

Since Hopfield's pioneering work on neural networks as computational models [14], [15], neurodynamic optimization approaches have attracted considerable attention. The discrete and continuous Hopfield networks have been applied to combinatorial optimization problems, including the TSP [16], [17]. However, single neurodynamic models often get trapped in local optima when dealing with binary variables [18]. To address this limitation, Collaborative Neurodynamic Optimization (CNO) has been proposed, where a population of neurodynamic models operates in parallel for scatter search [19]–[22]. This framework integrates neurodynamic optimization with swarm intelligence and is proven to converge almost surely to global optima [23].

In this paper, we formulate the TSP as a quadratic assignment problem with constraints designed to eliminate excessively long paths, thus enhancing efficiency. We introduce a CNO algorithm that combines Boltzmann Machines with a momentum term (BMm) and the 2-opt heuristic. The algorithm operates in two phases: the first employs multiple BM-Ms for local search, and the second applies the 2-opt heuristic for solution refinement. Additionally, a Particle Swarm Optimization (PSO) update rule is utilized to re-initialize the BMms, facilitating escape from local optima and convergence toward global optimal solutions. The experimental results indicate

This work was supported in part by the Research Grants Council of the Hong Kong Special Administrative Region of China under Grants 11202019, and 11203721; and in part by the InnoHK initiative, the Government of the Hong Kong Special Administrative Region, and Laboratory for AI-Powered Financial Technologies.

that the proposed algorithm achieves better solution quality compared to the two baseline methods. The main contributions of this work can be summarized as follows:

- We formulate the traveling salesman problem (TSP) as a constrained quadratic assignment problem (QAP) by introducing a filtering mechanism that eliminates excessively long paths during the search, thereby reducing the solution space and enhancing computational efficiency.
- We propose a novel collaborative neurodynamic optimization algorithm called CNO-CTSP/BMm, integrating multiple Boltzmann Machines with momentum (BMm), the 2-opt local search heuristic, and a particle swarm optimization (PSO)-based re-initialization strategy to escape local minima and guide the population towards global optimality.
- Extensive experiments show that the proposed method consistently outperforms baseline neurodynamic algorithms in terms of both solution quality.

The structure of the paper is organized as follows: Section II covers the foundational concepts, including the Boltzmann machine, particle swarm optimization, and mutation operations. Section III outlines the formulation and reformulation of the TSP. Section IV elaborates on the proposed CNO-CTSP algorithm. Section V presents the experimental results using six benchmark datasets. Finally, Section VI summarizes the findings of the paper.

II. Preliminaries

A. Neurodynamic Model

1) Boltzmann Machine (BM): BM is a widely recognized stochastic neural network, regarded as an extension of the Hopfield network. It is a fully connected network with binary states. Neural states in this network are updated according to the states of their neighboring neurons and the associated connections. The probability of a neural transition is given by the following equation [24]:

$$\begin{cases} P(x_i(t)=1) = 1/1+e^{-\frac{u(t)}{T}}, \\ P(x_i(t)=0) = 1-P(x_i(t)=1), \end{cases} \tag{1}$$

where T denotes a temperature parameter, and $u(t)$ is given by:

$$u(t) = Wx(t) - \theta. \tag{2}$$

The temperature follows an exponential multiplicative cooling schedule, given by:

$$T = T_0\alpha^t,$$

where T_0 denotes an starting temperature, and α is a cooling factor ranging between 0 and 1.

A BM with a momentum term (BMm) [25] is expressed as:

$$\begin{cases} u(t) = u(t) + Wx(t) - \theta, \\ p(x_i(t+1)=1) = \dfrac{1}{1+\exp(-\frac{u_i(t)}{T})}, \\ p(x_i(t+1)=0) = 1-p(x_i(t+1)=1). \end{cases} \tag{3}$$

B. Collaborative Neurodynamic Optimization

It is a swarm intelligence optimization method inspired by the cooperative dynamics seen in bird flocks. The method uses a group of particles, with individual particles modifying their positions based on their personal best and the best position within the swarm [26].

Each particle's velocity v_i and position $x_i(t)$ are updated according to the following rules:

$$\begin{cases} v_i(t+1) = & c_0v_i(t) + c_1r_1(x_i^* - x_i(t)) \\ & +c_2r_2(x^* - x_i(t)), \\ x_i(t+1) = & x_i(t) + v_i(t+1), \end{cases} \tag{4}$$

where $x_i \in \mathbb{R}^n$ represents the position of the i^{th} particle, x_i^* is the i^{th} particle's individual best position, x^* is the best position found by the entire swarm, $c_0 \in [0,1]$ is the inertia coefficient, c_1 and c_2 are cognitive and social learning factors, respectively, both within the range $[0,1]$ and r_1 and r_2 are random numbers sampled from the uniform distribution in $[0,1]$.

The mutation operation is introduced to maintain solution diversity and prevent the algorithm from getting trapped in local optima. If swarm diversity falls below a certain threshold, the mutation operation is applied to escape local optimal solutions.

The swarm's diversity is measured by [27]:

$$\delta(x) = \frac{1}{Nn}\sum_{i=1}^{N} \|x^{(i)} - x^*\|_2, \tag{5}$$

where N is the number of solutions, n is the solution dimensionality, $x^{(i)}$ is the i^{th} solution, and x^* is the best solution identified across all particles.

A commonly used mutation operator for binary variables, known as the bit-flip mutation [28], is defined as:

$$x_j = \begin{cases} \neg x_j & \text{if } \zeta_j \le \xi, \\ x_j & \text{otherwise }, \end{cases} \tag{6}$$

where $\bar{x}_j$ represents the opposite state of x_j, ζ_j is a random variable uniformly distributed within $[0,1]$, and ξ is the predefined mutation probability. The mutation helps increase exploration and maintain diversity, which improves the overall performance of the swarm-based optimization method.

C. 2-Opt Heuristic

The 2-opt heuristic is a simple and effective local search algorithm used to improve an existing tour in the TSP by iteratively eliminating crossing paths, which often leads to shorter tours [10]. Given a current tour $T = (c_1, c_2, \ldots, c_n, c_1)$, where c_i represents the city visited at position i, the 2-opt heuristic operates as follows: select two non-adjacent edges (c_i, c_{i+1}) and (c_k, c_{k+1}) with $1 \le i < k-1 \le n$, and replace them with edges (c_i, c_k) and (c_{i+1}, c_{k+1}). This exchange effectively reverses the sequence of cities between positions $i+1$ and k, resulting in a new tour. The change in tour length ΔL due to this operation is calculated by

$$\Delta L = (d_{c_ic_k} + d_{c_{i+1}c_{k+1}}) - (d_{c_ic_{i+1}} + d_{c_kc_{k+1}}), \tag{7}$$

where $d_{c_ic_j}$ denotes the distance between cities c_i and c_j. If $\Delta L < 0$, the new tour is accepted as it reduces the total tour length. This process is repeated for different pairs of edges until no further improvements can be made, yielding a tour that is locally optimal with respect to 2-opt moves.

III. Problem Formulations

A. Problem Formulation

Consider a set of n cities, and let d_{jl} represent the distance from city j and city l, with the distance matrix $D = [d_{jl}]$ being symmetric (i.e., $d_{jl} = d_{lj}$ for all $j, l = 1, \ldots, n$). We introduce binary decision variables x_{ij} for $i, j = 1, \ldots, n$, defined as $x_{ij} = 1$ if city j is visited at position i in the tour, and $x_{ij} = 0$ otherwise.

The objective is to minimize the total length of the tour, which can be expressed as [29]:

$$\min_x \sum_{i=1}^{n}\sum_{j=1}^{n}\sum_{l=1\ l\neq j}^{n} d_{jl}x_{ij}x_{(i \bmod n)+1,l}, \tag{8a}$$

$$\text{s.t.} \sum_{i=1}^{n} x_{ij} = 1, \quad \forall j = 1, \ldots, n, \tag{8b}$$

$$\sum_{j=1}^{n} x_{ij} = 1, \quad \forall i = 1, \ldots, n, \tag{8c}$$

$$x_{ij} \in \{0,1\}, \quad \forall i, j = 1, \ldots, n. \tag{8d}$$

$$\tag{8e}$$

where the objective function (8a) calculates the total distance of the tour by summing the distances between consecutive cities. The term $x_{ij}, x_{(i \bmod n)+1,l}$ ensures that the distance between city j at position i and city l at position $(i \bmod n)+1$ is included in the sum when both $x_{ij} = 1$ and $x_{(i \bmod n)+1,l} = 1$. The modulo operation $(i \bmod n) + 1$ handles the wrap-around from the last city back to the first city, ensuring the tour is closed. Constraints (8b) ensure that each city is visited exactly once. For each city j, the sum over all positions i of x_{ij} must equal 1, meaning city j appears exactly once in the tour. Constraints (8c) ensure that at each position i, exactly one city is visited. The binary constraints (8d) enforce that the decision variables x_{ij} can only take values 0 or 1. Together, these constraints guarantee that $X = [x_{ij}]$ forms a permutation matrix, representing a valid tour visiting each city exactly once.

B. Problem Reformulation

It could be efficient to solve TSP by incorporating constraints to eliminate long paths early in the search process. By introducing an additional constraint that avoids selecting city pairs with distances significantly exceeding a threshold, the search space can be reduced, focusing on promising routes. To this end, we define an index set F containing pairs of cities (j, l) to be avoided:

$$F = \{(j,l) \mid d_{jl} \geq \lambda d_{\max} + (1-\lambda)d_{\min},\ j \neq l\},$$

where d_{jl} denotes the distance between city j and city l, $d_{\max}$ and $d_{\min}$ denote the maximum and minimum distances in the distance matrix, respectively, and λ is a tuning parameter ranging from 0 to 1. The parameter λ controls the balance between the maximum and minimum distances to identify city pairs that should be avoided to prevent the selection of excessively long paths.

We reformulate the original TSP problem in (8) as follows:

$$\min \sum_{i=1}^{n}\sum_{j=1}^{n}\sum_{l=1, l\neq j}^{n} d_{jl}x_{ij}x_{(i \bmod n)+1,l}, \tag{9a}$$

$$\text{s.t.} \sum_{i=1}^{n} x_{ij} = 1, \quad \forall j = 1, \ldots, n, \tag{9b}$$

$$\sum_{j=1}^{n} x_{ij} = 1, \quad \forall i = 1, \ldots, n, \tag{9c}$$

$$x_{ij}x_{i+1,l} = 0, \quad \forall i = 1, \ldots, n,\ (j,l) \in F, \tag{9d}$$

$$x_{ij} \in \{0,1\}, \quad \forall i, j = 1, \ldots, n, \tag{9e}$$

where constraints (9d) ensure that city pairs in F are not selected consecutively in the tour, effectively preventing excessively long paths.

Transforming the TSP into a quadratic unconstrained binary optimization (QUBO) problem allows us to leverage neural network models such as BM for solving the problem efficiently. Consider the vector $x = [x_{11}, x_{12}, ..., x_{1n}, x_{21}, x_{22}, ..., x_{2n}, ..., x_{nn}]^T \in \{0,1\}^{n^2}$. The original formulation in (9) can be reformulated into:

$$\min\ x^T\hat{D}x, \tag{10a}$$

$$\text{s.t. } Ax = e, \tag{10b}$$

$$x^TBx = 0, \tag{10c}$$

where the element $\hat{d}_{(i-1)n+j,(k-1)n+l}$ equals d_{jl} if $k = i+1$ for $i = 1, 2, ..., n-1$, and $k = 1$ for $i = n$. The matrices A and e are written:

$$A = \begin{bmatrix} I_1 & I_2 & \cdots & I_n \\ I & I & \cdots & I \end{bmatrix} \in \{0,1\}^{2n \times n^2},$$
$$e = \{1\}^{2n \times 1}.$$

The matrix B is specified as:

$$B_{(i-1)n+j,(k-1)n+l} = \begin{cases} 1 & \text{if } k = i+1 \text{ and } (j,l) \in F, \text{ for } i = 1, 2, ..., n-1, \\ 1 & \text{if } k = 1 \text{ and } i = n \text{ and } (j,l) \in F, \\ 0 & \text{otherwise.} \end{cases}$$

To address the constraints in (10b), a quadratic penalty term is introduced:

$$p_1(x) = \frac{1}{2}\|Ax - e\|_2^2.$$

Additionally, a penalty term for the constraints in (10c) is defined as:

$$p_2(x) = x^TBx. \tag{11}$$

Next, we construct a penalized objective function as follows:

$$f_\rho(x) = f(x) + \rho(p_1(x) + p_2(x)),$$

where ρ is a penalty parameter.

The problem in (9) is thus transformed into a QUBO formulation:

$$\begin{aligned} &\min \ f_\rho(x), \\ &\text{s.t. } x \in \{0,1\}^{N^2}. \end{aligned} \tag{12}$$

As shown in [30], if ρ is sufficiently large, problems (8) and (12) would have the same optimal solutions.

We can further simplify the penalized objective function $f_\rho(x)$ as follows:

$$\begin{aligned} f_\rho(x) &= x^T\hat{D}x + \frac{\rho}{2}\|Ax-e\|_2^2 + \rho x^T Bx, \\ &= x^T(\hat{D}+B)x + \frac{\rho}{2}((Ax-e)^T(Ax-e)), \\ &= -\frac{1}{2}x^T(-2\hat{D} - \rho A^TA - 2B)x - \rho e^TAx + \frac{\rho}{2}e^Te. \end{aligned}$$

From this simplification, the parameters for BM are derived as:

$$\begin{aligned} W &= -2\hat{D} - \rho A^TA - 2B, \\ \theta &= -\rho e^TA. \end{aligned}$$

To ensure stability in BM, the diagonal elements of W (i.e., w_{ii}) must be set to zero. Because we have $x_i^2 = x_i$ for binary variables x_i, the diagonal elements of W can be zeroed, while adding an equivalent linear term $\text{diag}(w_{11}, \ldots, w_{nn})x$. Then, $\hat{W}$ and $\hat{\theta}$ become:

$$\hat{W} = W - \text{diag}(W), \tag{13}$$

$$\hat{\theta} = \frac{1}{2}\text{diag}(W) - \rho e^TA. \tag{14}$$

Thus, the TSP problem is transformed into the form of problem (9), using $\hat{W}$ from equation (13) and $\hat{\theta}$ from equation (14).

IV. CNO-CTSP/BMm Algorithm

Fig. 1 illustrates the framework of the proposed CNO-CTSP algorithm. In the proposed algorithm, multiple BMms are utilized for scattered local search, and the 2-opt heuristic is applied to enhance the solutions found by the BMms. Furthermore, the PSO update rule re-initializes the neuronal states of BMms. The 2-opt heuristic is a local search algorithm used to refine an existing tour in the TSP by eliminating crossings and finding shorter paths. The detailed pseudocode for the 2-opt algorithm is presented in Algorithm 1.

Algorithm 2 presents the CNO-CTSP/BMm approach for solving the TSP. In steps 2-6, multiple BMms reach their equilibrium states, and the algorithm updates the best solution for ith BMm. In step 3, neuron activation is carried out simultaneously. Step 4 applies the 2-opt heuristic to improve the solutions. Steps 7-14 involve updating the global best solution. Step 15 re-initializes the states of BMms using the PSO update rule. Step 16 measures swarm diversity, and steps 17-19 implement the bit-flip mutation.

Algorithm 1: 2-Opt Heuristic Algorithm

```
Input: An initial tour $T = [c_1, c_2, \ldots, c_n]$, where $c_i$ is
       the city at position $i$; distance matrix $D = [d_{ij}]$
Output: An improved tour $T$
1  repeat
2      improvement ← false;
3      for $i = 1$ to $n - 2$ do
4          for $k = i + 2$ to $n$ do
5              if $k = n$ and $i = 1$ then
6                  continue;
7              end
8              $i_1 \leftarrow c_i$;
9              $i_2 \leftarrow c_{i+1}$;
10             $k_1 \leftarrow c_k$;
11             $k_2 \leftarrow c_{(k \bmod n)+1}$;
12             $\Delta \leftarrow (d_{i_1k_1} + d_{i_2k_2}) - (d_{i_1i_2} + d_{k_1k_2})$;
13             if $\Delta < 0$ then
14                 $T[i+1:k] \leftarrow [c_k, \ldots, c_{i+1}]$;
15                 improvement ← true;
16             end
17         end
18     end
19 until no improvement is found;
20 return $T$;
```

V. Experimental Results

A. Experimental Setup

This section presents an assessment of the performance of three CNO-based algorithms: CNO-TSP/DHN, CNO-TSP/BM, and CNO-CTSP/BMm. These algorithms are applied to six well-known TSP benchmark datasets: Ulysses22, Berlin52, Lin105, Ch130, Gr202, and TSP225. The datasets vary in size and complexity, ranging from 22 cities to 225 cities, and are sourced from the TSPLIB benchmark suite[1].

For each dataset, we perform 50 independent runs for each algorithm to ensure statistical robustness. The parameters for all algorithms are set as follows: $\lambda = 0.8$. the number of Boltzmann Machines (population size) $N = 50$, and the termination criterion $M = 10$ iterations without improvement in the best solution. For the CNO-CTSP/BMm algorithm, the cooling rate $\alpha = 0.5$. The PSO parameters are set to $c_0 = 1$, $c_1 = 2$, and $c_2 = 2$. The bit-flip mutation probability is set to $P_m = 0.01$, with a diversity threshold of $\epsilon = 0.004$.

B. Results and Analysis

Figure 2 illustrates the optimal tours obtained by each algorithm, showing the solution quality visually. As shown in Figure 2, the CNO-CTSP/BMm algorithm consistently achieves better results than the two baseline methods across all datasets.

Table I presents the performance of each algorithm across the six datasets, including the best, worst, mean solutions,

[1]http://comopt.ifi.uni-heidelberg.de/software/TSPLIB95/index.html

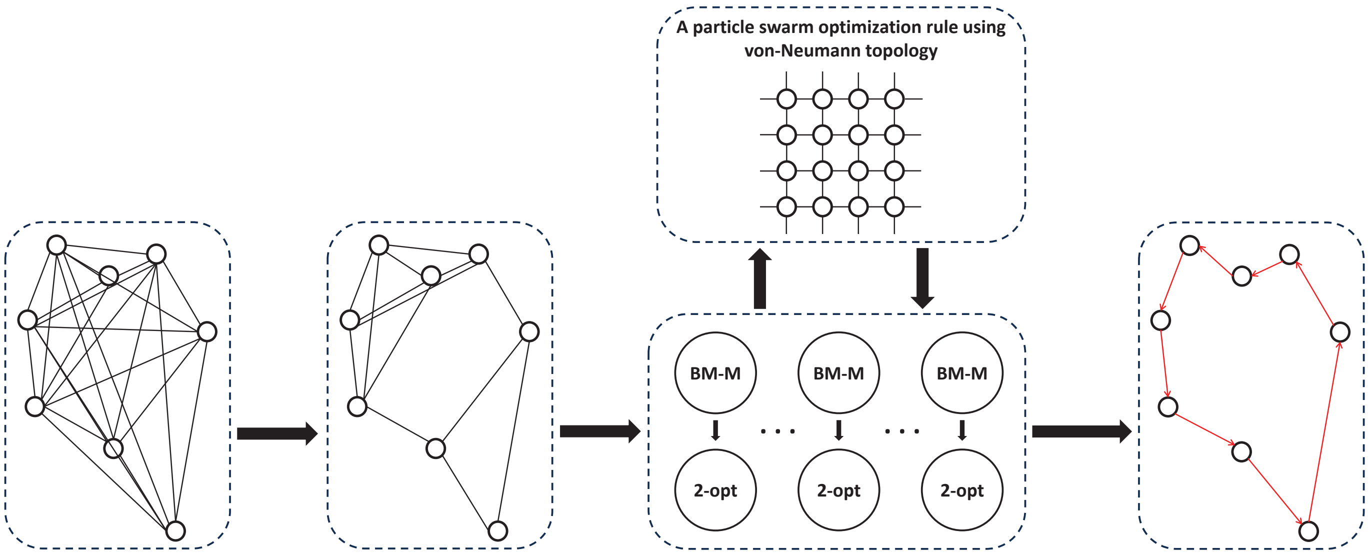

A particle swarm optimization rule using von-Neumann topology
BM-M
BM-M
BM-M
2-opt
2-opt
2-opt

Fig. 1: A flowchart of CNO-CTSP/BMm.

Algorithm 2: CNO-CTSP/BMm algorithm

```
Input: Number of BMs $N$, stopping criteria $M$,
       starting temperature $T_0$, cooling factor $\alpha$,
       $[x^{(1)}(0), ..., x^{(N)}(0)] \in 0, 1^{\{n^2 \times N\}}$,
       $[v^{(1)}, ..., v^{(N)}] \in [-1, 1]^{\{n^2 \times N\}}$, $c_0$, $c_1$, and $c_2$,
       and diversity limit $\epsilon$.
Output: $x^*$.
1  while $m \leq M$ do
2      foreach $i$ to $N$ BMs do
3          Update the neural states of the $i^{\text{th}}$ BM using
             equation (1);
4          Apply the 2-opt heuristic in Algorithm 1 to
             refine the solution $x^{(i)}$;
5          Update the objective function $f(x^{(i)})$ found by
             the $i^{\text{th}}$ BM;
6      end
7      $i^* = \arg\min_i \{f(x^{(1)}), ..., f(x^{(i)}), ..., f(x^{(N)}\}$;
8      if $f(x^{(i^*)}) < f(x^*)$ then
9          $f(x^*) \leftarrow f(x^{(i^*)})$, $x^* \leftarrow x^{(i^*)}$, $m \leftarrow 0$;
10     else
11         $m \leftarrow m + 1$;
12     end
13     Update the initial neural states of all BMs using
         the PSO rule by equation (4);
14     Calculate the diversity $\delta(x)$ using equation (5);
15     if $\delta(q) < \epsilon$ then
16         Adjust $x_i(0)$ using equation (6);
17     end
18 end
19 return $x^*$.
```

and standard deviations over 30 runs. As shown in Table I, CNO-CTSP/BMm not only reaches the optimal solution for the smaller datasets (e.g., Ulysses22 and Berlin52) but also achieves near-optimal results for larger datasets (e.g., Lin105, Ch130, Gr202, and TSP225). For instance, in the Ulysses22 dataset, CNO-CTSP/BMm consistently finds the exact optimal solution (7013) across all runs, demonstrating its ability to converge reliably to high-quality solutions. Additionally, the proposed algorithm maintains lower standard deviations across all datasets, indicating more consistent performance.

VI. Concluding Remarks

The paper introduces CNO-CTSP/BMm to address the TSP. The algorithm utilizes multiple Boltzmann machines with momentum terms, the 2-opt heuristic, and the neural states are re-initialized using the PSO update rule upon convergence. The efficiency is achieved by incorporating additional constraints to reduce computational complexity, utilizing multiple Boltzmann Machines with momentum for scattered local searches, applying the 2-opt heuristic for refinement, and leveraging the meta-heuristic rule for global re-positioning. The experimental results indicate that CNO-CTSP/BMm achieves better solution quality compared to the baseline methods. Future research could explore integrating learning methods with the CNO algorithm to enhance the performance and scalability of solving TSP.

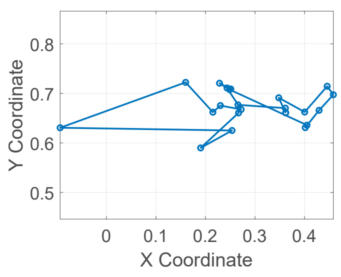
References

[1] C. H. Papadimitriou, "The euclidean travelling salesman problem is np-complete," *Theoretical Computer Science*, vol. 4, no. 3, pp. 237–244, 1977.

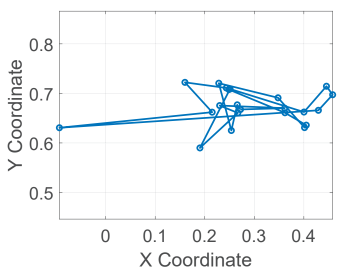
[2] J. K. Lenstra and A. R. Kan, "Some simple applications of the travelling salesman problem," *Journal of the Operational Research Society*, vol. 26, no. 4, pp. 717–733, 1975.

[3] J. Presby and M. Wolfson, "An algorithm for solving job sequencing problems," *Management Science*, vol. 13, no. 8, pp. B–454, 1967.

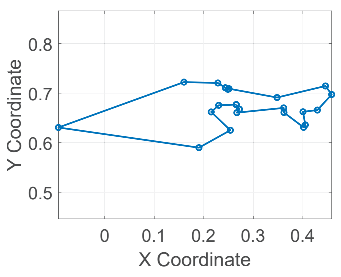
[4] M. Grotschel, M. Junger, and G. Reinelt, "Optimal control of plotting and drilling machines: a case study," *Mathematical Methods of Operations Research*, vol. 35, no. 1, pp. 61–84, 1991.



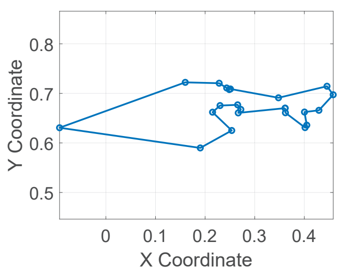
(a) CNO-TSP/DHN



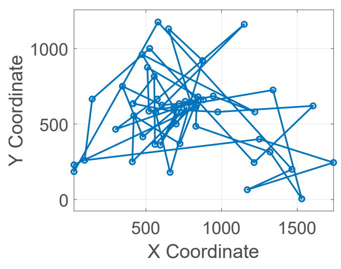
(b) CNO-TSP/BM



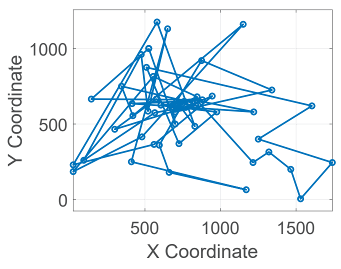
(c) CNO-CTSP/BMm



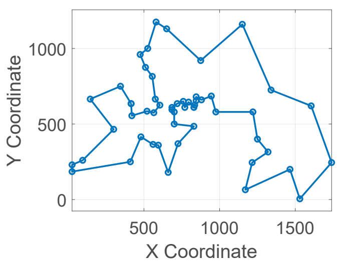
(d) Optimal solution



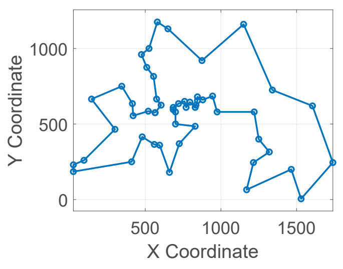
(e) CNO-TSP/DHN



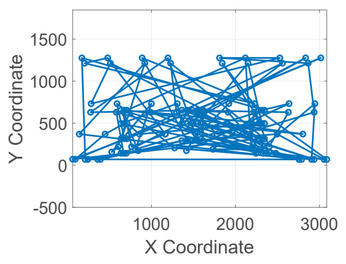
(f) CNO-TSP/BM



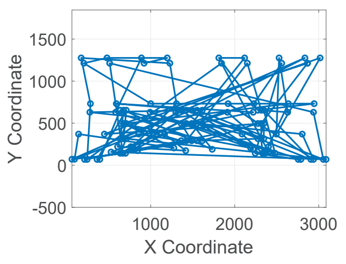
(g) CNO-CTSP/BMm



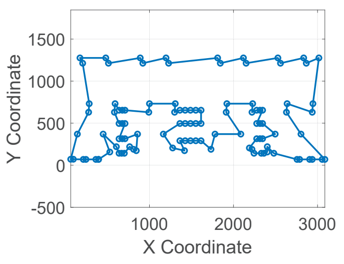
(h) Optimal solution



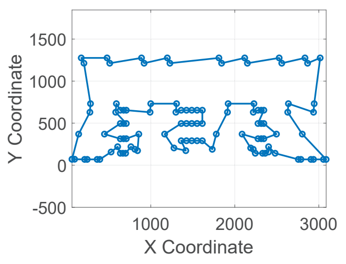
(i) CNO-TSP/DHN



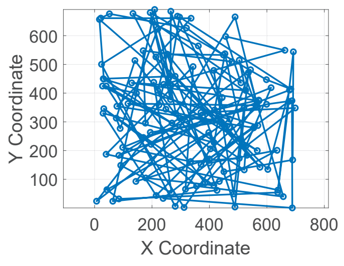
(j) CNO-TSP/BM



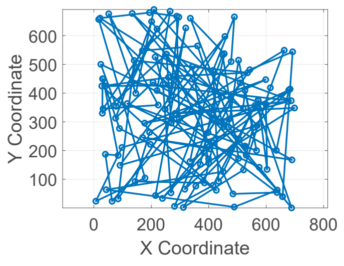
(k) CNO-CTSP/BMm



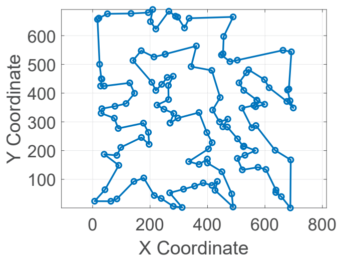
(l) Optimal solution



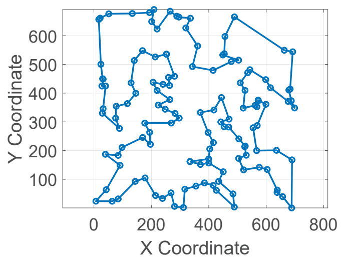
(m) CNO-TSP/DHN



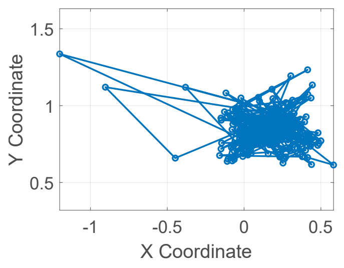
(n) CNO-TSP/BM



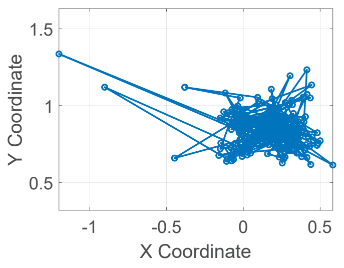
(o) CNO-CTSP/BMm



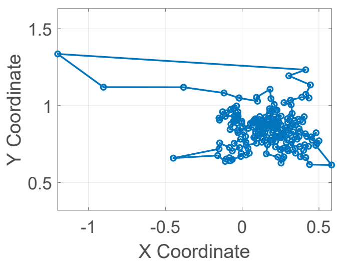
(p) Optimal solution



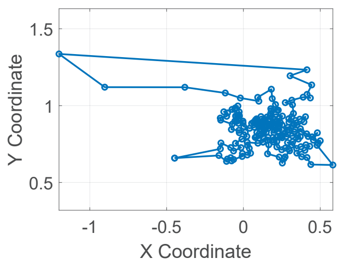
(q) CNO-TSP/DHN



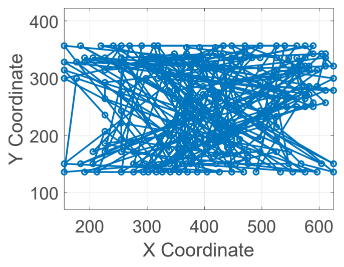
(r) CNO-TSP/BM



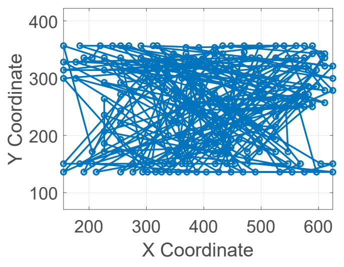
(s) CNO-CTSP/BMm



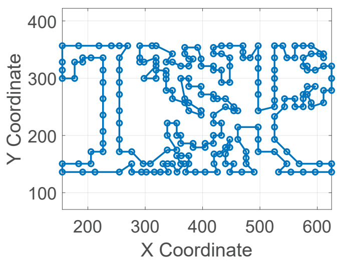
(t) Optimal solution



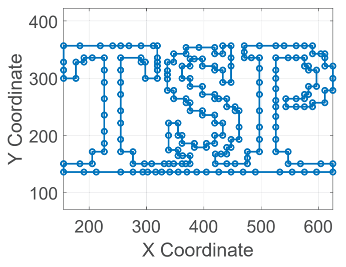
(u) CNO-TSP/DHN



(v) CNO-TSP/BM



(w) CNO-CTSP/BMm



(x) Optimal solution

Fig. 2: The optimal tour and the tour of the six datasets attained by using the baselines and the CNO-CTSP/BMm algorithm with $M = 10$ and $N = 50$.

TABLE I: Performance comparison of three CNO-based algorithms (CNO-TSP/DHN, CNO-TSP/BM, and CNO-CTSP/BMm) on six benchmark datasets for the traveling salesman problems. The table presents the number of cities, the optimal solution, and the best/worst, mean, and standard deviation of the results obtained by each algorithm, where the best results are highlighted in bold and the second-best results are underlined.

Dataset	# of cities	optimum	algorithm	best / worst	mean ± std
Ulysses22	22	7013	CNO-TSP/DHN [31]	7330.0000 / 9558.0000	8294.5600 ± 678.8170
			CNO-TSP/BM [32]	10898.0000 / 13234.0000	12033.2000 ± 552.5474
			CNO-CTSP/BMm (herein)	**7013.0000 / 7013.0000**	**7013.0000 ± 0.0000**
Berlin52	52	7544.3659	CNO-TSP/DHN [31]	22571.3816 / 25434.9371	24420.7413 ± 711.6491
			CNO-TSP/BM [32]	22066.5265 / 25244.3792	23808.1337 ± 917.5638
			CNO-CTSP/BMm (herein)	**7544.3659 / 7679.1525**	**7575.7595 ± 43.3234**
Lin105	105	14382.9959	CNO-TSP/DHN [31]	99472.1749 / 108130.2428	104229.9040 ± 2514.9779
			CNO-TSP/BM [32]	92402.6681 / 102163.5218	97989.1130 ± 2821.5035
			CNO-CTSP/BMm (herein)	**14406.1182 / 14607.8544**	**14504.5661 ± 53.1905**
Ch130	130	6110.8609	CNO-TSP/DHN [31]	39855.9005 / 41762.0722	40871.8654 ± 498.3488
			CNO-TSP/BM [32]	38866.0871 / 40717.7691	39998.8698 ± 472.4217
			CNO-CTSP/BMm (herein)	**6176.7372 / 6307.9880**	**6255.0129 ± 32.2765**
Gr202	202	40160	CNO-TSP/DHN [31]	235735.0000 / 249391.0000	244194.3600 ± 3386.9427
			CNO-TSP/BM [32]	227117.0000 / 239035.0000	235058.2000 ± 2859.7113
			CNO-CTSP/BMm (herein)	**41531.0000 / 42220.0000**	**41883.4400 ± 194.1359**
TSP225	225	3859	CNO-TSP/DHN [31]	36564.6964 / 38245.1275	37452.5363 ± 472.6754
			CNO-TSP/BM [32]	34346.3555 / 36222.5068	35495.3566 ± 440.3546
			CNO-CTSP/BMm (herein)	**4030.1605 / 4120.0306**	**4075.4680 ± 21.0441**

[5] G. Laporte, "The traveling salesman problem: An overview of exact and approximate algorithms," *European Journal of Operational Research*, vol. 59, no. 2, pp. 231–247, 1992.

[6] R. Matai, S. P. Singh, and M. L. Mittal, "Traveling salesman problem: An overview of applications, formulations, and solution approaches," *Traveling salesman problem, theory and applications*, vol. 1, 2010.

[7] P. Bouman, N. Agatz, and M. Schmidt, "Dynamic programming approaches for the traveling salesman problem with drone," *Networks*, vol. 72, no. 4, pp. 528–542, 2018.

[8] C. A. Hurkens and G. J. Woeginger, "On the nearest neighbor rule for the traveling salesman problem," *Operations Research Letters*, vol. 32, no. 1, pp. 1–4, 2004.

[9] M. Gendreau, A. Hertz, G. Laporte, and M. Stan, "A generalized insertion heuristic for the traveling salesman problem with time windows," *Operations Research*, vol. 46, no. 3, pp. 330–335, 1998.

[10] H. Okano, S. Misono, and K. Iwano, "New tsp construction heuristics and their relationships to the 2-opt," *Journal of Heuristics*, vol. 5, pp. 71–88, 1999.

[11] J. Grefenstette, R. Gopal, B. Rosmaita, and D. Van Gucht, "Genetic algorithms for the traveling salesman problem," in *Proceedings of the first International Conference on Genetic Algorithms and their Applications*, vol. 160, no. 168. Lawrence Erlbaum, 1985, pp. 160–168.

[12] T. Stützle and M. Dorigo, "ACO algorithms for the traveling salesman problem," *Evolutionary Algorithms in Engineering and Computer Science*, vol. 4, pp. 163–183, 1999.

[13] K.-P. Wang, L. Huang, C.-G. Zhou, and W. Pang, "Particle swarm optimization for traveling salesman problem," in *Proceedings of the 2003 International Conference on Machine Learning and Cybernetics*, vol. 3. IEEE, 2003, pp. 1583–1585.

[14] J. J. Hopfield, "Neural networks and physical systems with emergent collective computational abilities," *Proceedings of the National Academy of Sciences*, vol. 79, no. 8, pp. 2554–2558, 1982.

[15] ——, "Neurons with graded response have collective computational properties like those of two-state neurons," *Proceedings of the National Academy of Sciences*, vol. 81, no. 10, pp. 3088–3092, 1984.

[16] D. W. Tank and J. J. Hopfield, "Simple 'neural'optimization networks: an A/D converter, signal decision circuit, and a linear programming circuit," *IEEE Trans. Circuits and Systems*, vol. 33, no. 5, pp. 533–541, 1986.

[17] J. J. Hopfield and D. W. Tank, "Computing with neural circuits - a model," *Science*, vol. 233, no. 4764, pp. 625–633, 1986.

[18] M. Peng, N. K. Gupta, and A. F. Armitage, "An investigation into the improvement of local minima of the Hopfield network," *Neural Networks*, vol. 9, no. 7, pp. 1241–1253, 1996.

[19] H. Che and J. Wang, "A collaborative neurodynamic approach to global and combinatorial optimization," *Neural Networks*, vol. 114, pp. 15 – 27, 2019.

[20] H. Li, J. Wang, N. Zhang, and W. Zhang, "Binary matrix factorization via collaborative neurodynamic optimization," *Neural Networks*, vol. 176, p. 106348, 2024.

[21] H. Li and J. Wang, "Bi-clustering of binary data via neurodynamics-driven binary matrix factorization," in *2025 13th International Conference on Intelligent Control and Information Processing (ICICIP)*, 2025, pp. 90–97.

[22] ——, "Collaborative neurodynamic algorithms for solving sudoku puzzles," in *2022 12th International Conference on Information Science and Technology (ICIST)*, 2022, pp. 8–17.

[23] Z. Yan, J. Fan, and J. Wang, "A collective neurodynamic approach to constrained global optimization," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 28, no. 5, pp. 1206–1215, 2017.

[24] G. E. Hinton and T. J. Sejnowski, "Optimal perceptual inference," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 1983, pp. 448–453.

[25] H. Li and J. Wang, "Capacitated clustering via majorization-minimization and collaborative neurodynamic optimization," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 35, no. 6, pp. 6679–6692, Jun. 2024.

[26] ——, "A collaborative neurodynamic algorithm for quadratic unconstrained binary optimization," *IEEE Transactions on Emerging Topics in Computational Intelligence*, vol. 9, no. 1, pp. 228–239, Feb. 2025.

[27] ——, "From soft clustering to hard clustering: A collaborative annealing fuzzy *c*-means algorithm," *IEEE Transactions on Fuzzy Systems*, vol. 31, no. 3, pp. 1181–1194, 2024.

[28] Y. Zhang, S. Wang, P. Phillips, and G. Ji, "Binary PSO with mutation operator for feature selection using decision tree applied to spam detection," *Knowledge-Based Systems*, vol. 64, pp. 22–31, 2014.

[29] J. Wang, "A deterministic connectionist machine for the traveling salesman problem," in *Proc. IEEE International Conference on Systems, Man and Cybernetics*. IEEE, 1990, pp. 374–375.

[30] G. A. Kochenberger, F. Glover, B. Alidaee, and C. Rego, "A unified modeling and solution framework for combinatorial optimization problems," *OR Spectrum*, vol. 26, no. 2, pp. 237–250, 2004.

[31] H. Li, J. Wang, and J. Wang, "Solving the travelling salesman problem based on collaborative neurodynamic optimization with discrete hop-

field networks," in *2021 11th International Conference on Information Science and Technology (ICIST)*, 2021, pp. 456–465.

[32] H. Li and J. Wang, "A collaborative neurodynamic optimization algorithm based on boltzmann machines for solving the traveling salesman problem," in *2021 11th International Conference on Intelligent Control and Information Processing (ICICIP)*, 2021, pp. 325–333.