

Nearest-Better Network for Visualizing and Analyzing Combinatorial Optimization Problems: A Potential Unified Tool

Yiya Diao, Changhe Li*, Sanyou Zeng, Xinye Cai, Wenjian Luo, Shengxiang Yang, and Carlos A. Coello
Coello, *Fellow, IEEE*

Abstract—The Nearest-Better Network (NBN) is a powerful method to visualize sampled data for continuous optimization problems while preserving multiple landscape features. However, the calculation of NBN is very time-consuming, and the extension of the method to combinatorial optimization problems is challenging but very important for analyzing the algorithm’s behavior. This paper provides a straightforward theoretical derivation showing that the NBN network essentially functions as the maximum probability transition network for algorithms. This paper also presents an efficient NBN computation method with logarithmic linear time complexity to address the time-consuming issue. By applying this efficient NBN algorithm to the OneMax problem and the Traveling Salesman Problem (TSP), we have made several remarkable discoveries for the first time: The fitness landscape of OneMax exhibits neutrality, ruggedness, and modality features. The primary challenges of TSP problems are ruggedness, modality, and deception. Three state-of-the-art TSP algorithms (EAX, LKH, and NLKH) have limitations when addressing challenges related to modality and deception, respectively. LKH, based on local search operators, fails when there are deceptive solutions near global optima. EAX, which is based on a single population, can efficiently maintain diversity. However, when multiple attraction basins exist, EAX retains individuals within multiple basins simultaneously, reducing inter-basin interaction efficiency and leading to algorithm’s stagnation. NLKH improves over LKH by leveraging learned edge weights to increase the chance of reaching the global basin, but it remains vulnerable to deceptive funnels due to biased learning from underrepresented complex instances.

Index Terms—Nearest-better network, fitness landscape analysis, combinatorial optimization problem, traveling salesman

This work was supported in part by the National Natural Science Foundation of China under Grant 62476006, in part by the Hubei Provincial Natural Science Foundation of China under Grant 2023AFA049, in part by the Fundamental Research Funds of the AUST under Grant 2024JBZD0007, and in part by the National Natural Science Foundation of China under Grant U23B2058. (*Corresponding author: Changhe Li*).

Y. Diao and W. Luo are with Guangdong Provincial Key Laboratory of Novel Security Intelligence Technologies, School of Computer Science and Technology, Harbin Institute of Technology, Shenzhen, Guangdong 518055, China (email: diaoyiyacug@gmail.com).

C. Li, S. Yang, and X. Cai are with the State Key Laboratory of Digital Intelligent Technology for Unmanned Coal Mining, Anhui University of Science & Technology, Huainan 232001, China. They are also with the School of Artificial Intelligence, Anhui University of Science & Technology, Hefei 231131, China. S. Yang is also with the School of Computer Science and Informatics, De Montfort University, Leicester, LE1 9BH, U.K. (email: changhe.li@gmail.com).

S. Zeng is with the School of Mechanical Engineering and Electronic Information, China University of Geosciences, Wuhan, China.

C. A. Coello Coello is with CINVESTAV-IPN, Department of Computer Science, Mexico, D.F. 07360, Mexico. He is also (as part of a sabbatical leave) a member of the Faculty of Excellence with the School of Engineering and Sciences, Tecnológico de Monterrey, Monterrey, N.L., Mexico.

problem

I. INTRODUCTION

Combinatorial optimization problems are a crucial category of optimization problems, prevalent in various real-world applications. The Traveling Salesman Problem (TSP) is a classic example of a combinatorial optimization problem, and many practical combinatorial optimization problems can be addressed using the TSP model, such as vehicle routing problems [1], DNA sequencing [2], and computer chip layout design [3]. The study of TSP is of significant importance to both academia and industry.

There are numerous optimization algorithms for solving the TSP, from which Lin-Kernighan-Helsgaun (LKH) [4] and Edge Assembly Crossover based GA (EAX) [5] are two state-of-the-art algorithms. In recent years, neural combinatorial optimization approaches have also emerged, such as NeuroLKH (NLKH), which integrates deep learning models with the LKH heuristic [6]. However, research on TSP algorithms appears to have reached a bottleneck in recent years, with few new algorithms showing significant performance improvements [7]. But, is this really the case? As shown in experiments reported in [8], even for relatively simple TSP instances with 500 cities, none of EAX, LKH, or NLKH can achieve 100% accuracy. This indicates that certain landscape features pose challenges that these algorithms struggle to overcome. This also suggests that there is still room for improvement. What we truly lack is a robust tool for analyzing combinatorial problems and algorithms. Such tools would enable researchers to effectively identify the inherent difficulties in the problems, pinpoint the algorithms’ weaknesses, and would allow to systematically improve existing algorithms.

Fitness Landscape Analysis (FLA) provides tools for understanding problem difficulty and guiding algorithm design through the analysis and visualization of optimization landscapes. Among existing methods, uniform-sampling-based approaches are the least biased. The Nearest Better Network (NBN) [9] is currently the only visual uniform-sampling-based method applicable to combinatorial optimization problems (COPs).

Previous empirical results have shown that NBN can effectively visualize key landscape features such as modality, neutrality, and funnel structures. However, NBN lacks an efficient calculation method to handle the large number of

samples generated by combinatorial algorithms and also lacks a theoretical analysis to substantiate its mechanisms. Motivated by these challenges, this paper presents a comprehensive study of NBN in the context of COPs. The main contributions of this paper are as follows:

- This paper attempts to explain the working mechanism of NBN from the perspective of algorithm search behaviors with theoretical analysis. Our preliminary analysis shows that NBN fundamentally represents the maximum transition probability model of an algorithm. This analysis reveals that NBN statistically models the algorithm's behavior, thereby simplifying the original fitness landscape. Moreover, it explains why NBN can preserve most features of the fitness landscape, which have significant impacts on algorithm's performance.
- This paper proposes an efficient algorithm to compute NBN for COPs so that we can visualize the NBN of COPs in logarithmic linear time complexity.
- This paper uses the tunable W-Model [10] to illustrate the landscape structures of COPs. By adjusting its parameters, the NBN networks reveal key features such as ruggedness, neutrality, and multimodality.
- This paper analyzes the strengths and limitations of leading TSP algorithms: EAX, LKH, and NLKH. While LKH struggles with deceptive solutions near global optima due to its reliance on local search, EAX maintains diversity but tends to stagnate when dealing with multiple basins of attraction. NLKH guides the search toward the global optimum using a learned sparse graph, but remains vulnerable to deceptive funnels due to biased learning from underrepresented complex instances.

The remainder of this paper is organized as follows. Section II gives an overview of previous related work. Section III provides a theoretical proof of NBN. Section IV details the algorithm to calculate NBN for the given problems. Section V presents the experimental analysis of the landscape features for OneMax and TSP, as well as the behavior of algorithms applied to TSP. Finally, our conclusions and some potential paths for future research are given in Section VI.

II. PREVIOUS RELATED WORK

A general view of fitness landscapes was proposed in [11], in which the fitness landscape consists of three elements (\mathbf{X}, χ, f) :

- A set \mathbf{X} of potential solutions to the problem,
- a notion χ of neighborhood, nearness, distance, or accessibility on \mathbf{X} , and
- a fitness function $f : \mathbf{X} \rightarrow \mathbf{R}$. The fitness of a solution indicates how good the solution is, and the larger the fitness value, the better the solution.

There are several definitions related to the fitness landscape and the algorithm's behavior, including the following:

- **Search space:** The search space \mathbf{X} is the union of all possible solutions of an optimization problem.
- **Neighborhood:** The neighborhood relationship is a mapping $\chi : \mathbf{X} \rightarrow \mathbf{N}$, which associates each solution \mathbf{x} with a set of candidate solutions $\mathbf{N}(\mathbf{x})$,

- **Basin of Attraction (BoA) and local optimum:** $B(\mathbf{x}^*) = \{\mathbf{x} \in \mathbf{X} \mid \mathbf{x}^* = \text{local-search}(\mathbf{x})\}$, where the BoA $B(\mathbf{x}^*)$ of a local optimum \mathbf{x}^* is the set of solutions $B(\mathbf{x}^*)$ that approaches \mathbf{x}^* by utilizing a local search strategy among the decision variable space \mathbf{X} [12].
- **Search trajectory T :** Search trajectory is a sequence of the solutions generated by the algorithm in one run. In this paper, T represents the set of solutions of a search trajectory in one run of the algorithm.

Methods for studying fitness landscapes can be broadly categorized into two complementary directions: (1) the design of tunable benchmark problems, which embed specific landscape features to enable controlled experiments on algorithm behavior and problem difficulty; and (2) fitness landscape analysis (FLA) methods, which directly characterize landscape features using numerical indicators or visualization techniques. Among these, visualization plays a crucial role, as it enables intuitive understanding of complex, high-dimensional landscapes. Relying exclusively on numerical indicators often provides only partial insights, particularly when multiple interacting features are present. This section focuses on approaches applicable to COPs.

A. Tunable Benchmark

Several tunable combinatorial benchmark models have been proposed, including the NK [13], MNK [14], ND [15], Royal Road [16], and TOP [17] models. Although each supports certain landscape features (e.g. epistasis or neutrality), they typically lack modularity and flexibility in feature control. In contrast, the W-MODAL model [10] introduces a unified and layered framework with tunable control over ruggedness, neutrality, epistasis, deceptiveness, and multi-objectivity. Its modular design allows for flexible composition and isolation of features, making it more expressive and versatile than existing models.

B. FLA Methods

FLA methods can be broadly classified on the basis of their sampling strategies into four categories:

1) *Algorithm-Based Sampling:* These methods record solutions generated during algorithm execution to analyze algorithm behavior in a specific problem. For instance, the deceptiveness analysis proposed by Goldberg et al. [18] evaluates the misleading nature of a problem for genetic algorithms (GAs). The Search Trajectory Network (STN) [19] visualizes dynamic search paths using force-directed layouts. The IDEE method [20] employs t-SNE to project combinatorial solutions into a two-dimensional space to observe the evolution of the population.

While these methods reveal algorithm trajectories and convergence patterns, the sampling is inherently biased by the algorithm's behavior. In addition, they provide a limited explanation for search failure, as they lack a direct association with the underlying landscape features.

2) *Operator-Based Sampling*: Operator-based FLA methods define neighborhoods according to the search scope of operators and analyze landscape features through operator-induced sampling. As optimization algorithms typically consist of multiple operators, these methods enable the association of algorithmic behavior with landscape features. They are particularly effective in studying properties such as evolvability [21], which are closely related to algorithm performance. The Local Optima Network (LON) [22] is an operator-based graph method that shows local optima and their transitions. It uses a 2-opt operator to find local optima and a 4-opt operator to escape them, building the network for the TSP. However, like algorithm-based methods, these methods are also affected by operator bias.

3) *Walk-Based Sampling*: These methods sample solutions via random walks in the search space, enabling statistical analysis of landscape features without relying on specific algorithms or operators. This strategy has been applied to evaluate ruggedness [23], neutrality [24], gradient information [25], and basin structures [26]. Loss-Gradient Cloud [26] projects high-dimensional landscapes into a three-dimensional space defined by the gradient magnitude, the fitness value, and the local curvature, indirectly revealing the number, shape, and connectivity of basins.

The distribution of samples may be affected by step size: short walks may overemphasize local regions, leading to sampling bias. To alleviate this, progressive random walks have been proposed [27].

4) *Uniform Sampling*: Uniform sampling methods aim to minimize bias by evenly distributing samples across the search space. These methods support a comprehensive analysis of various landscape features.

Representative metrics include: variance for measuring gene interactions [28]; dispersion for global topological diversity [29]; and fitness-distance correlation (FDC) for global structure [30].

Nearest-Better Clustering (NBC) [31] has been extensively studied for basin identification [32] and niching [33], [34] in multimodal optimization. Building upon this idea, the Nearest-Better Network (NBN) [35] was proposed as a visual, network-based fitness landscape analysis (FLA) method. It is currently the only uniform-sampling-based FLA approach applicable to COPs. Previous experiments [9] have demonstrated that NBN can effectively capture and visualize a variety of landscape features in continuous problems. In this paper, we further extend the application of NBN to COPs.

III. NEAREST-BETTER NETWORK

The Nearest Better Network (NBN) [9] is defined as a directed graph $G = (X_N, E)$, where the vertex set X_N contains N sampled solutions, and the edge set $E = \{(x, \beta(x)) \mid x \in X_N\}$ contains the nearest better relation for each solution. Formally,

$$\beta(x) = \arg \min_{y \in \{y \mid y \in X_N, f(y) > f(x)\}} \|y - x\|, \quad (1)$$

where $\beta(x)$ denotes the nearest better solution for x . As introduced in [9], NBN (X_N, β, f) serves as a simplification

of the original fitness landscape (X, χ, f) , and has proven effective in capturing important landscape features such as modality, neutrality, and ruggedness. This section provides a concise theoretical justification for the effectiveness of NBN—namely, why the nearest-better relation β can meaningfully approximate the original neighborhood relation χ . To enhance the generality and applicability of this approach, we further provide formal proofs in the context of continuous optimization problems.

A. Theoretical Analysis of NBN in Binary Combinatorial Optimization

This subsection provides a theoretical analysis of binary COPs, establishing that, under certain conditions, the solution with the highest transition probability from a given solution x aligns with its nearest better solution.

Generally speaking, binary encoding can represent most COPs. For inherently binary problems such as OneMax and the knapsack problem, binary encoding can be directly applied. Problems typically modeled with permutations or integer encodings—such as the TSP and graph coloring—can also be represented through binary variables, for example, by encoding a TSP tour as a binary adjacency matrix. Although this may introduce redundancy and increase decoding complexity, binary encoding provides a unified framework for expressing COPs. A general binary COP can be formulated as:

$$\max_{x \in \{0,1\}^D} f(x) \quad \text{s.t.} \quad g(x) \leq 0, h(x) = 0 \quad (2)$$

where $x = (x_1, x_2, \dots, x_D)$ is a binary decision vector of dimension D , with each component $x_i \in \{0, 1\}$. The objective function $f(x)$ is assumed to be maximized. The constraint vectors $g(x) = (g_1(x), \dots, g_M(x))^T$ and $h(x) = (h_1(x), \dots, h_P(x))^T$ represent inequality and equality constraints, respectively.

The FLAs aim to help design efficient algorithms, so it is a natural way to analyze the fitness landscape from the perspective of optimization algorithms. Here we consider a simple format of an evolutionary algorithm, a (1+1) ES with bitwise mutation for binary-encoded problems [36]. For a given solution x , each bit x_i is flipped independently with probability ρ :

$$x'_i = \begin{cases} 1 - x_i, & \text{if } \mathcal{U}(0, 1) < \rho, \\ x_i, & \text{otherwise,} \end{cases} \quad (3)$$

where $\mathcal{U}(0, 1)$ denotes the uniform distribution over $[0, 1]$. This process is equivalent to performing D independent Bernoulli trials, each with success probability ρ . And the offspring is accepted if and only if it is feasible and strictly better than the parent:

$$x = \begin{cases} x', & \text{if } f(x') > f(x) \wedge g(x') \leq 0 \wedge h(x') = 0, \\ x, & \text{otherwise.} \end{cases} \quad (4)$$

Let the Hamming distance between x and x' be used as the distance metric, defined as

$$k = \|x' - x\|_H, \quad (5)$$

i.e., the number of differing bits. Since each bit flips independently with probability ρ , k follows a binomial distribution:

$$k \sim \mathcal{B}(D, \rho). \quad (6)$$

Thus, the mutation transition probability from solution \mathbf{b} to solution \mathbf{a} is given by:

$$p_m(\mathbf{a} | \mathbf{b}) = p(k) = \binom{D}{k} \rho^k (1 - \rho)^{D-k} \quad (7)$$

Similar derivations [37] have also been employed in the convergence analysis of Markov chains underlying optimization algorithms. The probability of transitioning from solution \mathbf{b} to \mathbf{a} is essentially a conditional probability density, representing the likelihood of generating \mathbf{a} given that the current solution is \mathbf{b} [37]. Throughout this paper, we denote it uniformly as $p(\mathbf{a} | \mathbf{b})$ and refer to it simply as probability for brevity. It is important to note that $p(\mathbf{a} | \mathbf{b})$ is a probability density function. Notably, in some cases—such as when the variance of a Gaussian density is very small—this transition probability density may exceed 1.

To understand how $p(k)$ varies with k , we analyze its first difference:

$$\Delta p(k) = p(k+1) - p(k) = p(k) \cdot \frac{(D+1) \cdot \rho - (k+1)}{(k+1)(1-\rho)}. \quad (8)$$

From this expression, we observe that $p(k)$ increases when $\Delta p(k) > 0$ and decreases when $\Delta p(k) < 0$. Setting $\Delta p(k) = 0$ yields the approximate mode of the distribution:

$$k^* = (D+1) \cdot \rho - 1. \quad (9)$$

$p(k)$ increases in the region $[0, (D+1) \cdot \rho - 1]$, decreases in the region $[(D+1) \cdot \rho - 1, D]$, reaches the max at the $(D+1) \cdot \rho - 1$. Therefore, when the mutation probability ρ is small, k^* is also small.

Based on the selection rule defined in Eq. (4), the selection probability from solution \mathbf{b} to solution \mathbf{a} is given by:

$$p_s(\mathbf{a} | \mathbf{b}) = \begin{cases} 1, & \text{if } f(\mathbf{a}) > f(\mathbf{b}) \wedge \mathbf{g}(\mathbf{a}) \leq \mathbf{0} \wedge \mathbf{h}(\mathbf{a}) = \mathbf{0} \\ 0, & \text{otherwise.} \end{cases} \quad (10)$$

Assuming $\rho \leq \frac{2}{D+1}$, the mutation probability $p_m(\mathbf{y} | \mathbf{x}) = p(k)$ decreases monotonically with respect to the Hamming distance $\|\mathbf{y} - \mathbf{x}\|_H = k$ within the interval $k \in [(D+1)\rho - 1, D] \subseteq [1, D]$. Under this condition, the solution with the highest transition probability from \mathbf{x} is given by:

$$\begin{aligned} \beta(\mathbf{x}) &= \arg \max_{\mathbf{y} \in \mathbf{X}_N} p_m(\mathbf{y} | \mathbf{x}) \cdot p_s(\mathbf{y} | \mathbf{x}) \\ &= \arg \min_{\mathbf{y} \in \{\mathbf{y} \in \mathbf{X}_N \mid f(\mathbf{y}) > f(\mathbf{x}), \mathbf{g}(\mathbf{y}) \leq \mathbf{0}, \mathbf{h}(\mathbf{y}) = \mathbf{0}\}} \|\mathbf{y} - \mathbf{x}\|_H, \end{aligned} \quad (11)$$

which indicates that, under the condition $\rho \leq \frac{2}{D+1}$, the nearest better solution is also the one with the highest transition probability. Generally, solutions that are close in the search space are likely to have similar fitness values. Consequently, mutation probability ρ is often chosen to be small to promote local search behavior. Under such settings, approximating the highest transition probability solution by the nearest better solution is theoretically valid in most practical scenarios.

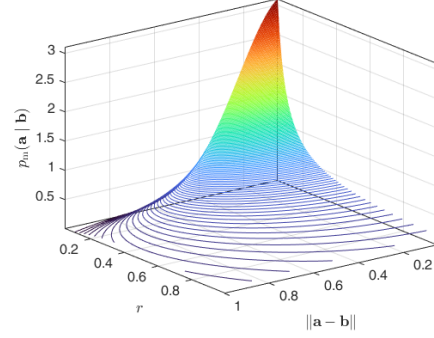


Fig. 1. Mutation probability $p_m(\mathbf{a} | \mathbf{b})$ with respect to mutation step-size r and distance $\|\mathbf{a} - \mathbf{b}\|^2$ ($D = 2$)

B. Theoretical Analysis of NBN in Continuous Optimization

This subsection presents a theoretical analysis in the context of continuous optimization. Here, we consider a $(1 + 1)$ -ES version with a Gaussian mutation operator.

$$x'_i \leftarrow x_i + \mathcal{N}(0, r), \quad (12)$$

where $\mathbf{x} = [x_1, x_2, \dots, x_D]$, D is the dimensionality of the problem, and r is the mutation step-size.

$$\mathbf{x} = \begin{cases} \mathbf{x}' & \text{if } f(\mathbf{x}') > f(\mathbf{x}) \\ \mathbf{x} & \text{otherwise} \end{cases} \quad (13)$$

Although this type of EA is very simple, most popular EAs share similarities with it. For example, the Covariance Matrix adaptation Evolution Strategy, (CMA-ES) [38] for continuous optimization problems has a similar format combined with its gradient calculation method.

Then, the transition probability between two solutions can be derived from Eqs. (12) and (13). Assuming that mutations in each dimension are independent and identically distributed, the mutation probability from \mathbf{b} to \mathbf{a} is given by

$$\begin{aligned} p_m(\mathbf{a} | \mathbf{b}) &= \prod_{i=1}^D p_m(a_i | b_i) \\ &= (2\pi r)^{-\frac{D}{2}} \exp\left(-\frac{\|\mathbf{a} - \mathbf{b}\|^2}{2r}\right), \end{aligned} \quad (14)$$

where r denotes the mutation step size. Fig. 1 illustrates how $p_m(\mathbf{a} | \mathbf{b})$ varies with respect to $\|\mathbf{a} - \mathbf{b}\|^2$ and r .

The selection probability $p_s(\mathbf{a} | \mathbf{b})$ is calculated by

$$p_s(\mathbf{a} | \mathbf{b}) = \begin{cases} 1, & \text{if } f(\mathbf{a}) > f(\mathbf{b}) \\ 0, & \text{otherwise} \end{cases}, \quad (15)$$

Finally, the transition probability between the two solutions $p(\mathbf{a} | \mathbf{b})$ is calculated by

$$\begin{aligned} p(\mathbf{a} | \mathbf{b}) &= p_m(\mathbf{a} | \mathbf{b}) p_s(\mathbf{a} | \mathbf{b}) \\ &= \begin{cases} (2\pi r)^{-\frac{D}{2}} \exp\left(-\frac{\|\mathbf{a} - \mathbf{b}\|^2}{2r}\right), & \text{if } f(\mathbf{a}) > f(\mathbf{b}) \\ 0, & \text{otherwise} \end{cases} \end{aligned} \quad (16)$$

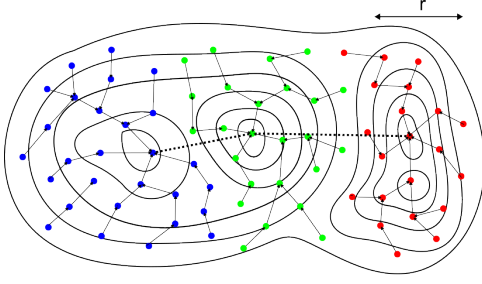


Fig. 2. Illustration of the NBN over a two-dimensional fitness landscape. The background contours represent fitness levels, with higher values indicating better solutions. Each node corresponds to a solution, and the black solid lines form the NBN, where each node is connected to its nearest better neighbor. Dashed lines indicate connections that would exist in the original NBN but are removed due to exceeding a predefined step size threshold r , i.e., when the distance $\|\mathbf{x}, \beta(\mathbf{x})\| > r$.

The solution with the highest transition probability from a given solution \mathbf{x} , denoted by $\beta(\mathbf{x})$, is defined as

$$\begin{aligned} \beta(\mathbf{x}) &= \arg \max_{\mathbf{y} \in \mathbf{X}_N} p(\mathbf{y} | \mathbf{x}) \\ &= \arg \min_{\mathbf{y} \in \{\mathbf{y} \in \mathbf{X}_N | f(\mathbf{y}) > f(\mathbf{x})\}} \|\mathbf{y} - \mathbf{x}\|. \end{aligned} \quad (17)$$

It can be seen that $\beta(\mathbf{x})$ corresponds to the nearest better solution as defined in Eq. (1).

It is important to note that NBN is not identical to the maximum transition network constructed with a fixed step size r . As illustrated in Fig. 2, when the distance $\|\mathbf{x}, \beta(\mathbf{x})\| > r$, the edge between the two solutions is removed.

Note that when the dimensionality D is sufficiently large and the mutation probability ρ is not too close to 0 or 1, the binomial distribution $\mathcal{B}(D, \rho)$ can be approximated by a normal distribution according to the Central Limit Theorem, i.e., $k \sim \mathcal{N}(D\rho, D\rho(1-\rho))$. This approximation parallels the derivation of the maximum transition probability in continuous optimization, indicating a fundamental consistency between continuous and discrete search spaces in terms of their probabilistic transition mechanisms.

C. Maximum Transition Network

From the theoretical results in both continuous and combinatorial optimization, it follows that the nearest better solution closely approximates the maximum transition solution. In other words, NBN fundamentally represents the maximum transition network. This concise theoretical insight elucidates the effectiveness of NBN: it simplifies the fitness landscape's structure while preserving its critical features. Note, if multiple nearest better solutions exist, one is selected at random, as they share the same transition probability.

Note that both the mutation probability density $p_m(\mathbf{a} | \mathbf{b})$ and the overall transition probability $p(\mathbf{a} | \mathbf{b})$ are directional; that is, they describe the likelihood of moving from solution \mathbf{b} to \mathbf{a} , and generally $p(\mathbf{a} | \mathbf{b}) \neq p(\mathbf{b} | \mathbf{a})$. Moreover, the theoretical argument holds even when the sample is small or non-uniformly distributed. While a large, uniformly distributed sample can provide an unbiased approximation of the original solution space, an NBN constructed from a smaller, biased

Algorithm 1: Calculation of nearest better solutions by traversal (CNBST)

Input: A set of sampled solutions \mathcal{S}

Output: The nearest better relationship β

```

1: for each solution  $\mathbf{x} \in \mathcal{S}$  do
2:    $d_{\text{NBD}}(\mathbf{x}) = \infty$ 
3:   for each solution  $\mathbf{y} \in \mathcal{S}$  and  $\mathbf{x} \neq \mathbf{y}$  do
4:     if  $f(\mathbf{y}) > f(\mathbf{x})$  and  $\|\mathbf{x} - \mathbf{y}\| < d_{\text{NBD}}(\mathbf{x})$  then
5:        $d_{\text{NBD}}(\mathbf{x}) = \|\mathbf{x} - \mathbf{y}\|$ 
6:        $\beta(\mathbf{x}) = \mathbf{y}$ 
7:     end if
8:   end for
9: end for
```

sample may not capture all landscape features. Nevertheless, it still represents the maximum transition probability network relative to the given sample set and effectively characterizes the search difficulty and some landscape features within that region.

IV. CALCULATION OF THE NEAREST-BETTER NETWORK

NBN was first introduced in [35], constructed via the traversal algorithm CNBST (Algorithm 1) from [39]. This method has a time complexity of $O(N^2D)$, where N is the number of sampled solutions and D is the problem dimensionality.

Here, we present a more efficient algorithm for computing the NBN for the assignment problem [40], which exemplifies a broad class of COPs, including the TSP and OneMax. In earlier analyses, problems were uniformly modeled using binary encodings. Although many combinatorial problems can be expressed in binary form, such representations may introduce redundancy in some cases, potentially reducing computational efficiency. To address this, we adopt the assignment problem as a more general modeling framework. Note that binary-encoded problems can be regarded as a special case of the assignment problem.

A. Distance metric

From the definition of NBN, the relationship between two solutions is defined based on the distance between two solutions, and the distance metric is different for different problems.

In One-Max problems with D digits, the Hamming distance is used as the distance metric. In the symmetric TSP with D cities, the Dice coefficient [41] is used as the distance metric, the distance between two solutions \mathbf{a} and \mathbf{b} defined by the Dice coefficient is:

$$\|\mathbf{a}, \mathbf{b}\| = 1 - \frac{2|M(\mathbf{a}) \cap M(\mathbf{b})|}{|M(\mathbf{a})| + |M(\mathbf{b})|} \quad (18)$$

where for a solution $\mathbf{a} = [a_1, a_2, \dots, a_D]$, $M(\mathbf{a}) = \{(a_i, a_{(i+1)\%D}), (a_{(i+1)\%D}, a_i) \mid i = 1, 2, \dots, D\}$ is the set of all edges that connect two cities for the solution.

B. Problem representation

We now consider the assignment problem [42], which widely exists in real-world applications. In the model, the search space is defined by a finite set of variables, $\mathbf{x} \in \mathbf{V}_1 \times \mathbf{V}_2 \times \dots \times \mathbf{V}_D$, and each variable of the solution x_i has an associated domain \mathbf{V}_i of values that can be assigned to it. A solution $\mathbf{x} = [x_1, x_2, \dots, x_D]$ is an assignment of a value $v \in \mathbf{V}_i$ to variable x_i and it is denoted by $x_i = v$. In a TSP with D cities, a variable x_i in a solution is a move from city i to the city x_i . This can be formalized by associating one variable to each city and each variable x_i has then $D - 1$ associated values, $x_i \in \mathbf{V}_i = \{a \mid a = 1, 2, \dots, D, a \neq i\}$. In a OneMax problem with D digits, a variable x_i indicates the digit at the i^{th} dimension, $x_i \in \mathbf{V}_i = \{0, 1\}$.

C. The proposed algorithm

Algorithm 2: Calculation of nearest better solutions by division (CNBSD)

Input: A set of sampled solutions \mathbf{S} ,
the set of all the unselected variable domain set \mathbf{D}_V ,
and the minimum number of the size of the calculated set N_m

Output: The nearest better relationship β , and the best solution \mathbf{b} of \mathbf{S} .

- 1: **if** $|\mathbf{S}| \leq N_m$ **then**
- 2: $\beta = \text{CNBST}(\mathbf{S})$
- 3: $\mathbf{b} = \arg \max_{\mathbf{a} \in \mathbf{S}} f(\mathbf{a})$
- 4: **else**
- 5: Record all the best solutions for each subset $\mathbf{P} = \emptyset$
- 6: Randomly select a domain set \mathbf{V}_k from \mathbf{D}_V
- 7: $\mathbf{D}_V = \mathbf{D}_V - \{\mathbf{V}_k\}$
- 8: Divide \mathbf{S} by the k^{th} domain set \mathbf{V}_k into different subset,
 $\mathbf{S} = \mathbf{S}^1 \cup \dots \cup \mathbf{S}^t$
- 9: **for** each subset \mathbf{S}_x^i **do**
- 10: $(\beta_i, \mathbf{b}_i) = \text{CNBSD}(\mathbf{S}_x^i, \mathbf{D}_V, N_m)$
- 11: $\beta = \min(\beta, \beta_i)$
- 12: $\mathbf{P} = \mathbf{P} \cup \{\mathbf{b}_i\}$
- 13: **end for**
- 14: $\beta_b = \text{CNBST}(\mathbf{P})$
- 15: $\beta = \min(\beta, \beta_b)$
- 16: $\mathbf{b} = \arg \max_{\mathbf{a} \in \mathbf{P}} f(\mathbf{a})$
- 17: **end if**

Inspired by the KD -tree [43], a classical space-partitioning data structure that efficiently supports nearest neighbor search in continuous spaces, we propose a similar strategy to improve the computational efficiency of our method.

In the proposed method, we divide the solution set into several smaller solution sets by one of the domains associated with one dimension, until the solution set is small enough to calculate their nearest better solutions, normally with $N_m = 20$ solutions. For one divided solution set, if it is divided into multiple subsets, it gathers the best solution from each subset to calculate its NBN by CNBST; otherwise, the NBN is calculated directly by Algorithm 2.

As stated in Line 8 of Algorithm 2, a solution set \mathbf{S} is randomly divided into multiple subsets. and CNBSD is used to compute the best solution and nearest better relationships (β) for each subset. For the nearest better relationships of the current solution set \mathbf{S} , the nearest better relationships β for the

Algorithm 3: Calculation of the nearest better solutions by random division (CNBSRD)

Input: A set of sampled solutions \mathbf{S} ,
the number of loops for calculation L ,
and the minimum size of the calculated set N_m

Output: The nearest better relationship β

- 1: **for** $l \leftarrow 1$ to L **do**
- 2: Initialize \mathbf{D}_V : $\mathbf{D}_V = \{\mathbf{V}_1, \mathbf{V}_2, \dots, \mathbf{V}_D\}$
- 3: $(\beta_i, \mathbf{b}_i) = \text{CNBSD}(\mathbf{S}, \mathbf{D}_V, N_m)$
- 4: Update β : $\beta_i = \min(\beta, \beta_i)$
- 5: **end for**

solutions within each subset have already been calculated, as shown in Line 10. For each subset's best solution, the nearest better solution (excluding the best solution of the current solution set) must belong to the current solution set, which must be the best solution of the other subsets. Therefore, we can use CNBSD to compute the nearest better relationships for the set of best solutions \mathbf{P} from these subsets, as shown in Line 14.

Finally, by merging the results of the nearest better relationships (i.e., select the nearest-better solution for each solution), we obtain the nearest better relationship β for the current solution set \mathbf{S} . With more random partitionings, the accuracy of the nearest better relationship calculation is more accurate. This is done in Algorithm 3. The code is released on the Open Framework of Evolutionary Computation (OFEC): <https://github.com/Changhe160/OFEC>.

Dividing the solution set into subsets is equivalent to projecting solutions onto specific dimensions. Calculating nearest neighbors from a single projection incurs errors. By the Johnson-Lindenstrauss lemma [44], to achieve error ϵ , the number of projections needed is $L > \frac{\ln(N)}{\epsilon^2}$. Each projection involves N nearest-better relationships computations, each with a time cost of $O(D)$. As a result, the overall time complexity is $O\left(\frac{N \ln N \cdot D}{\epsilon^2}\right)$. The dependency of the time complexity on the inverse square of the precision ϵ is typical in approximation methods such as random projection, and is generally considered an acceptable trade-off, especially in approximate computations.

As shown in Fig. 3, we compare the running time of CNBSI (the exact algorithm) and CNBSRD (the approximate algorithm) required to ensure that the average relative error of $d_{\text{NBD}}(\mathbf{x})$ is less than 0.05, i.e., $\frac{1}{N} \sum_{i=1}^N \frac{|d_{\text{NBD}}(\mathbf{x}_i) - \hat{d}_{\text{NBD}}(\mathbf{x}_i)|}{\hat{d}_{\text{NBD}}(\mathbf{x}_i)} \leq \epsilon = 0.05$ where $d_{\text{NBD}}(\mathbf{x}_i)$ denotes the approximated NBD calculated by CNBSRD, and $\hat{d}_{\text{NBD}}(\mathbf{x}_i)$ denotes the exact value obtained by CNBSI. From the figure, it can be observed that CNBSRD exhibits a clear advantage in terms of running time when $\epsilon = 0.05$ and $N > 10^5$. Compared to CNBSI with a time complexity of $O(N^2 D)$, CNBSRD significantly reduces the computational cost, showing a growth trend closer to $O(N \log N)$. Moreover, this algorithm is parallelizable. In Algorithm 3, we can parallelize the processing of each division, which is beneficial for optimizing the use of computing resources and for accelerating the NBN calculation.

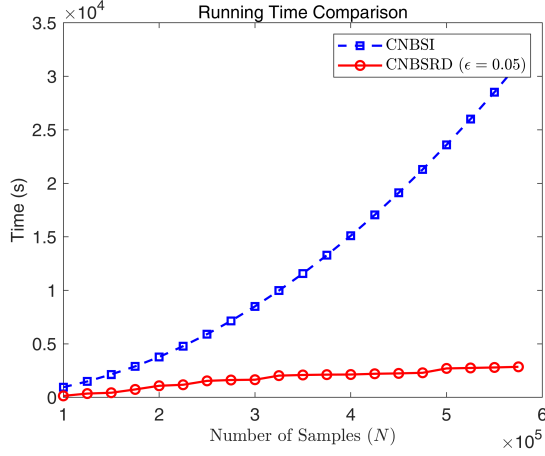


Fig. 3. Running time of the two algorithms, where both algorithms are implemented using multithreading in a system equipped with an Intel(R) Xeon(R) CPU E5-2699 v4 @ 2.20GHz, featuring 88 cores. Each algorithm was independently executed 30 times.

D. Local sampling

To observe the problem from different scales, we can perform a local sampling in a local area around a center solution \mathbf{x}_c within a distance of K . The local area around a center solution \mathbf{x}_c within a distance of K , $S(\mathbf{x}_c, K)$, is defined by:

$$S(\mathbf{x}_c, K) = \{\mathbf{a} \in \mathbf{X} \mid \|\mathbf{a}, \mathbf{x}_c\| \leq K\} \quad (19)$$

The proposed algorithm can directly calculate the NBN structure for local region sampled solutions. However, in our experiments, we found out that our algorithms perform well when the sampled solutions are evenly distributed. But when the distribution of sampled solutions is concentrated, some subsets may contain many solutions and it is time-consuming to divide such subsets. In a local sampled solutions set, many solutions share the same values with the center solution \mathbf{x}_c , and thus we remove the subset that contains \mathbf{x}_c and this is done after Line 8 of Algorithm 2.

V. NBN ASSISTED ANALYSIS OF COMBINATORIAL PROBLEMS AND ALGORITHMS

In this section, we demonstrate how the NBN can be applied to analyze combinatorial optimization problems and the behavior of optimization algorithms. We first introduce a set of NBN-based analysis metrics,

A. Analysis metric

In our previous work [9], we proposed several NBN-based metrics for landscape features such as modality, BoA, ruggedness, and neutrality. These metrics assumed uniform data sampling. However, when data comes from local or algorithmic sampling, this assumption no longer holds, making the original metrics unsuitable. In this subsection, we introduce new metrics that do not depend on uniform data distribution, enabling better analysis of problems and algorithms.

- Evolutionary path (\tilde{P})

NBN, $\mathbf{G} = (\mathbf{X}_N, \mathbf{E})$, is essentially a tree structure, where each solution is attracted by only one nearest better solution except for the global optima. For each solution \mathbf{x} , there exists a path connecting \mathbf{x} to the global optima \mathbf{o} . Considering that NBN is the maximum transition probability network, it can be proven that this path represents the evolution path with the highest probability for the solution \mathbf{x} to converge to the global optimum. The definition of this evolutionary path is as follows:

$\tilde{P}(\mathbf{x}, \mathbf{o}) = [\mathbf{p}_1, \mathbf{p}_2, \dots, \mathbf{p}_m]$ where $\mathbf{p}_1 = \mathbf{x}$, $\mathbf{p}_k = \mathbf{o}$, and m represents the number of nodes along the path.

- Distance of an evolutionary path ($d(\tilde{P})$)
For any given evolutionary path \tilde{P} , its complexity can be gauged by the maximum distance between its nodes. An increased maximum distance implies a reduced likelihood of transitioning to the subsequent node, thereby indicating a higher degree of difficulty of the evolutionary path. This measure is quantified as the distance of the evolutionary path in this paper, defined as $d(\tilde{P}) = \max_{i=1}^{k-1} \|\mathbf{p}_i, \mathbf{p}_{i+1}\|$.
- Distance of a solution set to the optima ($d(\mathbf{T}, \mathbf{o})$)

Evolutionary algorithms focus on solutions with either higher fitness values or greater evolutionary potential. As long as there is one solution in the solution set, \mathbf{T} , that possesses a shorter evolutionary path, the algorithm based on this set is more likely to converge to the global optimum.

Accordingly, this paper defines the distance of the shortest evolutionary path of a solution set as the distance between a solution set and the global optimum, denoted as:

$$d(\mathbf{T}, \mathbf{o}) = \min_{\mathbf{t} \in \mathbf{T}} d(\tilde{P}(\mathbf{t})), \quad (20)$$

Furthermore, the shortest evolutionary path from the solution set to the global optimum is given by:

$$\tilde{P}(\mathbf{T}, \mathbf{o}) = \tilde{P}(\mathbf{t}), \mathbf{t} = \arg \min_{\mathbf{t} \in \mathbf{T}} d(\tilde{P}(\mathbf{t})) \quad (21)$$

- Identification of optima in biased data-based NBN
The previously proposed metric for identifying local optima, referred to as FOM1 (Filter Optima Metric 1) [9], relies solely on the magnitude of the Nearest-Better Distance d_{NBD} . Specifically, a solution is considered an optimum if:

$$d_{\text{NBD}}(\mathbf{x}) \geq \vartheta \quad (22)$$

However, optimal solutions are inherently those with superior fitness. To improve accuracy, FOM2 (Filter Optima Metric 2) integrates both fitness and NBD within a biased-data-based NBN. A solution is identified as an optimum only if it satisfies both a fitness threshold and a distance threshold :

$$f(\mathbf{x}) \geq \theta \wedge d_{\text{NBD}}(\mathbf{x}) \geq \vartheta \quad (23)$$

where θ and ϑ are user-defined fitness and distance thresholds, respectively.

Fig. 4 illustrates a comparison between FOM1 and FOM2 using EAX data on the `rue500-1` instance. As shown in the figure, early in the evolutionary process, the algorithm explores broadly, resulting in large NBD values even

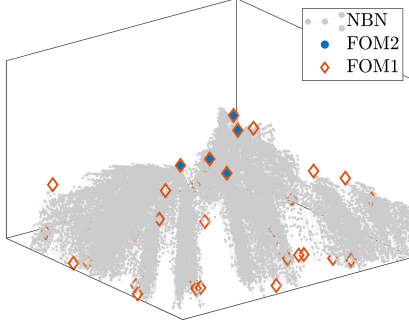


Fig. 4. Comparison of FOM1 and FOM2 on the rue500-1 TSP instance using EAX-generated data

in regions with poor fitness. Consequently, FOM1 may misclassify these low-quality solutions as local optima. FOM2 addresses this issue by incorporating fitness information, enabling more accurate identification of high-quality optima.

B. Feature analysis of the traveling salesman problem

This subsection aims to analyze the global and local structures of the fitness landscapes of three typical TSP instances and the search data of LKH and EAX. By combining this with the landscape features, this paper aims to identify the challenges for the algorithms in solving these problems. With the analysis, this paper aims to provide insights for the design and optimization of algorithms.

1) *TSP instances selection*: TSPLib [45] is a widely used benchmark dataset and the portgen generator [46] generates TSP instances (referred to as a rue instance) by randomly placing points on a two-dimensional plane.

We select three typical TSP instances: u574, rue500-1, and rue500-2. u574 is a commonly used TSP instance from TSPLib, containing 574 nodes. This instance was also used for observation and analysis in the paper of LON [47]. Researchers can compare the results of this paper with those of LON [47] to deepen the understanding of the fitness landscape of the TSP instances. Both rue500-1 and rue500-2 have 500 nodes generated by the portgen generator. TABLE I shows the success rates of EAX and LKH, i.e., the number of runs that found the global optimum versus the total number of runs, where both algorithms use the recommended parameters. As shown in TABLE I, the performances of LKH, NLKH, EAX on the two TSP instances are quite different. They both have 500 nodes, why do EAX and LKH behave so differently? The analysis of this subsection tries to answer this question.

TABLE I
SUCCESS RATE OF EAX, LKH AND NLKH ON DIFFERENT TSP INSTANCES

	u574	rue500-1	rue500-2
EAX	30/30	2/30	30/30
LKH	30/30	30/30	4/30
NLKH	30/39	24/30	21/30

To answer the question above, we first take a look at the working mechanism of the three algorithms. LKH mainly

consists of three techniques: local search operators, restart, crossover, and a GA framework. LKH will restart several times and in each restart, it generates a random solution, optimizes this solution using local search operators, and crosses over this solution with the iteration-best solution or a solution in the GA population to find a better solution. The framework of NeuroLKH constructs a local TSP network, infers edge and node weights using a trained sparse graph neural network, and subsequently performs further search using LKH guided by these learned weights. The key to EAX lies in its GA framework and an efficient crossover operator. It is a single-population algorithm that maintains a certain level of diversity during the evolutionary process.

2) *Global and local structure of TSP*: By comparing the results of the three TSP instances in Fig. 6, Fig. 5, and Fig. 7, we can find several features:

- **Ruggedness**

The TSP instance exhibits ruggedness from global to local regions with numerous straight hanging points. While in Fig. 6, Fig. 5, and Fig. 7, we can see that whether it's NBN with LON data or NBN with total data, the fitness landscape has been considerably smoothed out. This indicates that local search operators can smoothen the structure of the TSP fitness landscape, which also validates the importance of local search operators for TSP.

- **Modality**

From the results in Fig. 6, we can see that rue500-1 exhibits a single BoA globally, with multiple BoAs emerging in the local structure when K is smaller than 50. In the local structure of $K = 12$, there are two connected BoAs.

3) *Comparison between LON and NBN*: For the same dataset, the structures of LON and NBN exhibit similarities as shown in Fig 5 and Fig 7. Due to the high time complexity of the force-directed algorithm used in LON (N^2 , where N is the number of solutions), it can only visualize the best 0.01% of solutions. The similarities between LON and NBN suggest that although NBN only preserves the nearest-better relationship within the network, it still retains the features of BoAs.

LON relies on local search operators to explore relationships between solutions, which smoothens the fitness landscape, whereas NBN can visualize solutions from any source and retains important features such as ruggedness as illustrated in Fig. 6. The experiments below also show that NBN provides more information for a more profound analysis of the problem features and algorithm behaviors.

4) *Algorithm behavior analysis*: To compare the capabilities of LON, LDEE, and NBN in analyzing algorithm's behavior, we evaluate TSP instances using the data generated by each tool individually.

a) *Analysis based on LON*: LON can display the connections between local optima and further illustrate the BoAs. As illustrated in Fig. 5 and Fig. 7, all three TSP instances have two BoAs, and in rue500-1, most regions of the BoAs are connected. It seems that rue500-1 is easier than the other two instances. But is that correct? From the results in TABLE I, LKH has a low success rate on rue500-2. LON does not

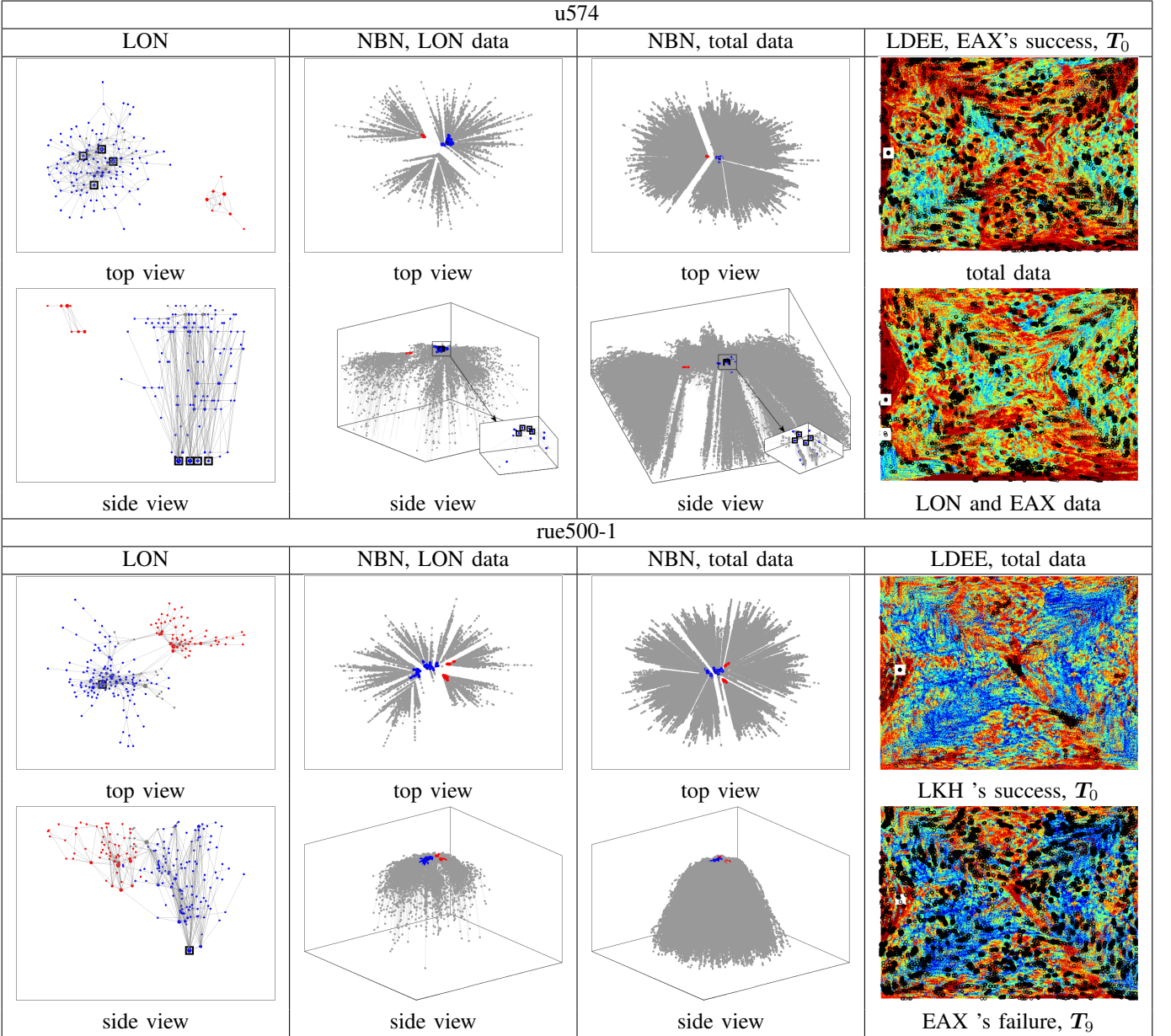


Fig. 5. Comparison of NBN, LON, and LDEE on u54 and rue500-1: Recommended parameters [47] are used in LON and figures of LDEE are generated by the tool provided by its authors [20]. There are three sources of data: data generated by LON, EAX, and LKH, that is LON data, EAX data, and LKH data. Total data is the union of all these data. Each point in LON represents a local optimum, and connections between points indicate transitions between the solutions by the 4-opt operator. The height of each point indicates its fitness value (lower values indicate better fitness). The color of points (from blue to red) represents the basin to which they belong, while gray points belong to multiple basins. Black rectangles are the global optima. This color-coding scheme is the same for NBN. LDEE is a two-dimensional grid image in which each grid represents a solution. The color gradient from blue to red indicates the fitness value of solutions (red is the best). White rectangles denote the locations of the optima, and black circles are the solutions of an algorithm's trajectory T . T_i indicates the i^{th} trajectory of an algorithm.

provide a clear answer as to whether the algorithm consistently gets stuck in the red local optima, as shown in Fig. 7.

On the other hand, both u574 and rue500-2 have two separate BoAs, yet EAX consistently finds the global optimum on the two instances. However, in rue500-1, which appears simpler with two connected BoAs, the algorithm's success rate is quite low. Based on LON visualization, it is difficult to identify the specific factors causing the poor performance of EAX in this instance.

b) Analysis based on LDEE: LDEE maps all solutions onto a two-dimensional plane by minimizing the total distance

between each pair of solutions. From the visualization, we can see that much information is lost, making it difficult to infer details about the landscape features. We can only observe the relative positions of the solutions and their fitness values.

LDEE focuses on minimizing the overall relative distances between all pairs of solutions, which may cause the loss of some critical information. The information about the distance between optima is of great importance to the researcher. As shown in Fig. 5, in NBN of u574, four optima are very close to each other. In LON of u574, the four optima are also mutually reachable. In the LDEE visualization with total data, the

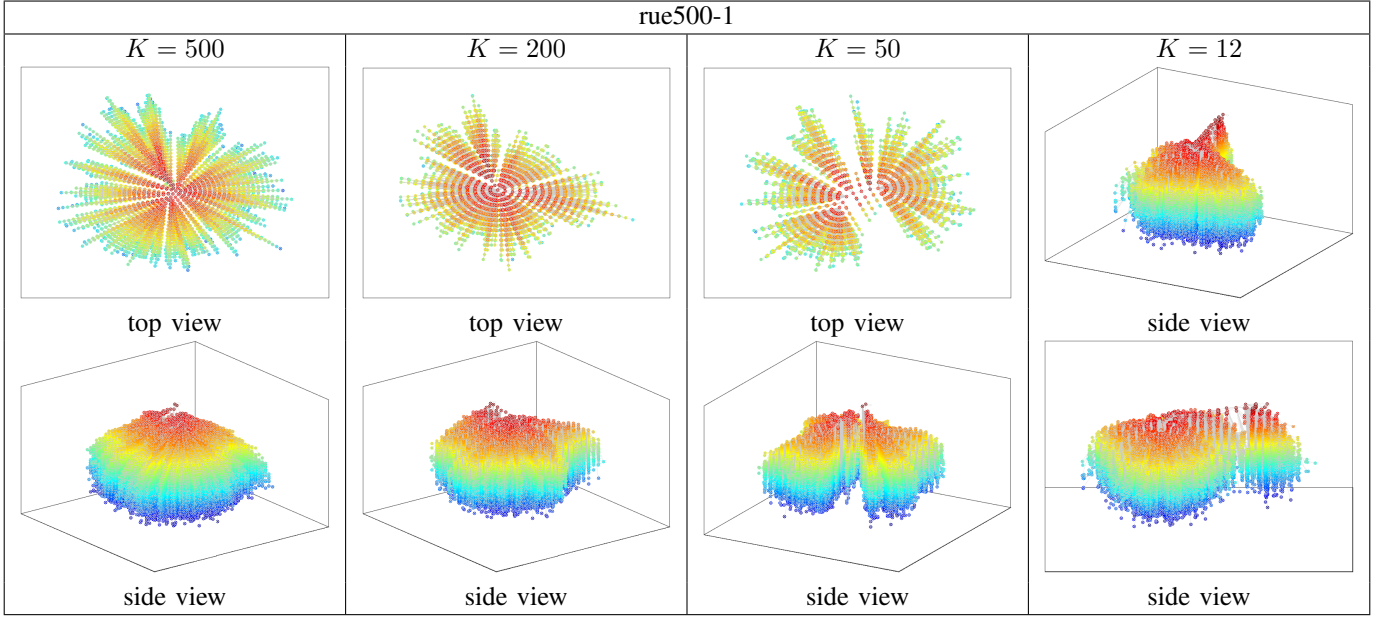


Fig. 6. Visualization of NBN of rue500-1 from different scales with $N = 10^6$ solutions generated by local random sampling, K is the radius of the local region.

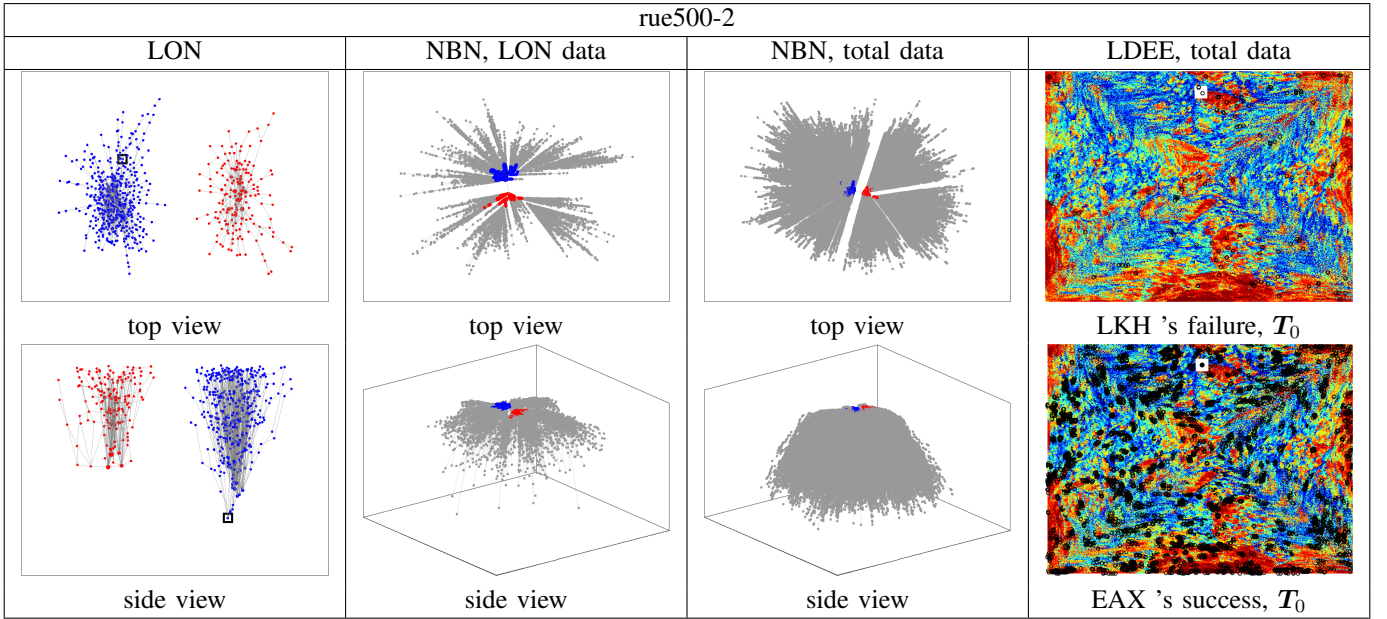


Fig. 7. Comparison of NBN, LON, and LDEE on rue500-2

positions of the four optima are close. However, in the LDEE visualization with LON and EAX data, we can see two sets of distant optima (white rectangles), which can be misleading, since one may believe that there are two groups of optimal solutions far apart in u574. The difference between LDEE with total data and with LON, EAX data also indicates that LDEE's mapping involves a significant degree of randomness.

Furthermore, it is hard to draw any effective conclusions about algorithm behavior from the LDEE visualization. In the case of EAX's failure on rue500-1, we can see that EAX finds some solutions very close to the global optimum. But despite this proximity, why does EAX fail to find the global optimum? LDEE does not provide answers to this question.

Similarly, in the case of LKH's failure in rue500-2, LDEE shows that the solutions generated by LKH are close to the global optimum as shown in Fig. 7. It seems that LKH with the nearest solution can converge to the global optimum using any local search operator. But is this the case? The following NBN-based analysis shows that in this trajectory T_0 , LKH gets stuck in a deceptive funnel.

c) *Analysis based on NBN*: To better analyze the behaviors of the three algorithms, different trajectories are shown in Fig. 8. The statistical information of $d(\hat{P}(T, \mathbf{o}))$ is shown in TABLE II, where "Failed TSP instance" indicates the instances where they have a low success rate. "min", "max", "mean", and "SD" represent the minimum value, maximum

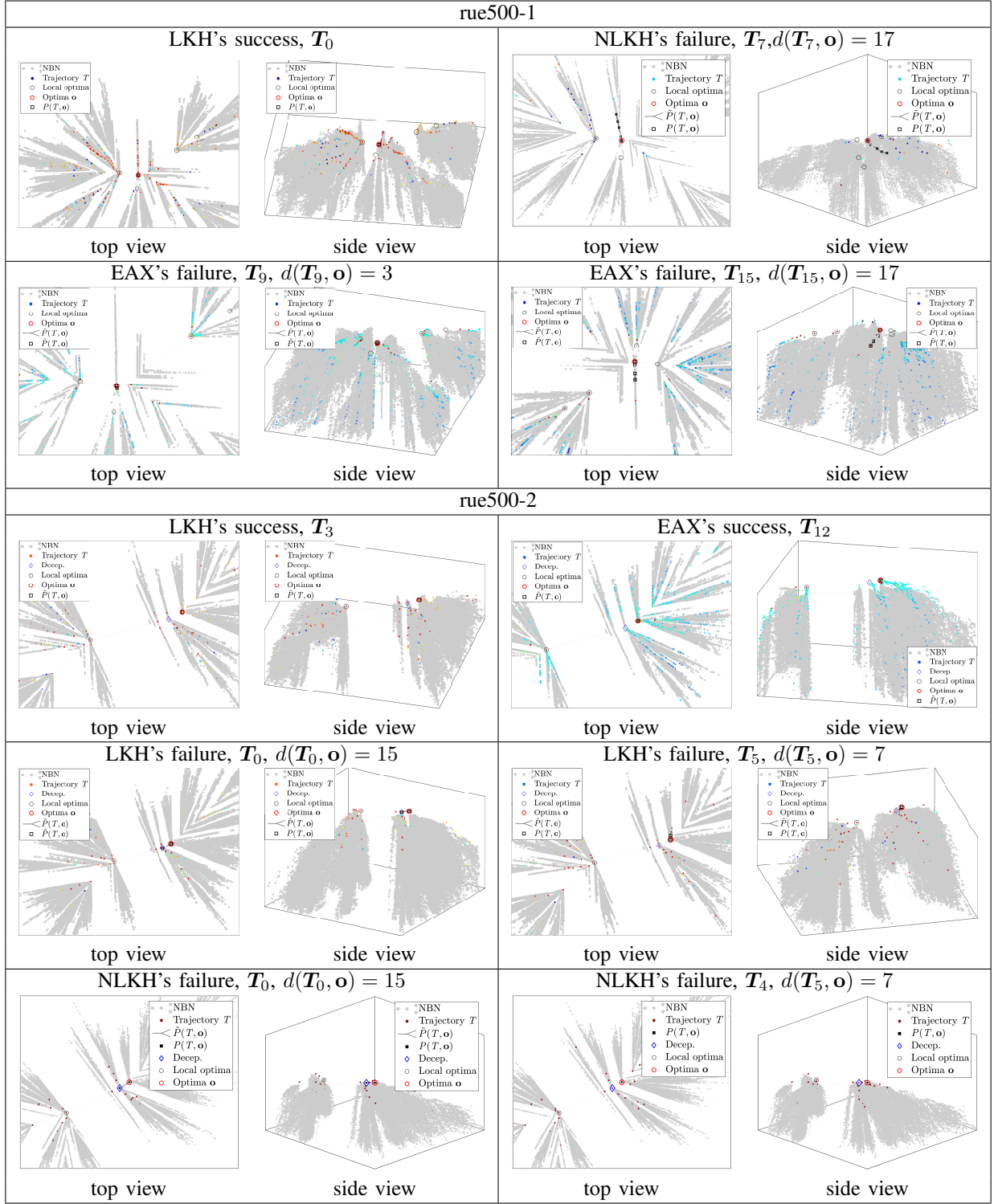


Fig. 8. NBN visualization with total data, where the gray network is NBN, colored stars are the solutions of an algorithm's trajectory T . T_i indicates the i^{th} trajectory of an algorithm. Black rectangles indicate the local optima and red circles are the global optima o . Diamond markers indicate deceptive solutions. Black rectangles are the solutions along the shortest evolutionary path from the trajectory T to the global optima $P(T, o)$. $d(T, o)$ is the distance of the evolutionary path. For the trajectories of both EAX and NLKH, the color represents the generated iteration of the solutions. For LKH's trajectories, the color represents the number of runs that the solutions are generated.

value, mean value, and standard deviation of $d(\tilde{P}(T, o))$, respectively. “Fails (Deceptive/Total)” indicates the number of failures when the algorithm gets stuck in deceptive solutions versus the total number of failures.

EAX's behaviors: EAX shows strong exploration and con-

sistently finds the global optimum's basin of attraction, even in failed trials. However, it has a low success rate on rue500-1, which has five optima, due to insufficient interaction across multiple basins.

From TABLE I, we see that EAX fails only on rue500-1.

TABLE II
STATISTICAL INFORMATION OF $d(\tilde{P}(\mathbf{T}, \mathbf{o}))$ OVER THE 30 INDEPENDENT TRAJECTORIES FOR THE THREE ALGORITHMS ON THEIR FAILED TSP INSTANCES.

Algorithm	Failed TSP instance	min	max	Avg \pm SD	Fails (Deceptive/Total)
EAX	rue500-1	3	17	9.25 ± 3.94267	-/28
LKH	rue500-2	7	15	14.6923 ± 1.53846	25/26
NLKH	rue500-1	12	17	14.8333 ± 2.26691	-/6
NLKH	rue500-2	14	15	14.8889 ± 0.31427	8/9

TABLE III
STATISTIC DATA OF THE FUNNELS AROUND OPTIMA, $K = 16$

	Id	Decep.	$\ \mathbf{n}, \mathbf{o}\ $	NBD	$N = 10,000$		$N = 100,000$		$N = 1,000,000$	
					Δf	$d(\mathbf{S}(\mathbf{n}, K), \mathbf{o})$	Δf	$d(\mathbf{S}(\mathbf{n}, K), \mathbf{o})$	Δf	$d(\mathbf{S}(\mathbf{n}, K), \mathbf{o})$
u574	1		15	11	6.86E-03	37	6.63E-03	34	7.26E-03	34
	2		17	12	9.09E-03	40	5.49E-03	41	5.19E-03	45
	3		17	13	-1.47E-02	38	-1.04E-02	41	-1.08E-02	32
rue500-1	1		12	12	2.57E-03	32	6.85E-04	37	1.88E-04	34
	2		17	10	2.56E-03	41	7.59E-04	38	1.28E-03	37
rue500-2	1		17	14	1.73E-02	44	1.69E-02	39	1.63E-02	36
	2		11	11	8.14E-04	36	2.98E-04	34	7.06E-04	33
	3		13	13	-1.83E-03	41	5.81E-04	36	7.82E-04	32
	4	✓	15	15	-1.43E-03	38	-4.43E-04	35	-6.69E-04	30
	5		16	12	3.10E-04	40	4.18E-04	36	3.71E-04	31

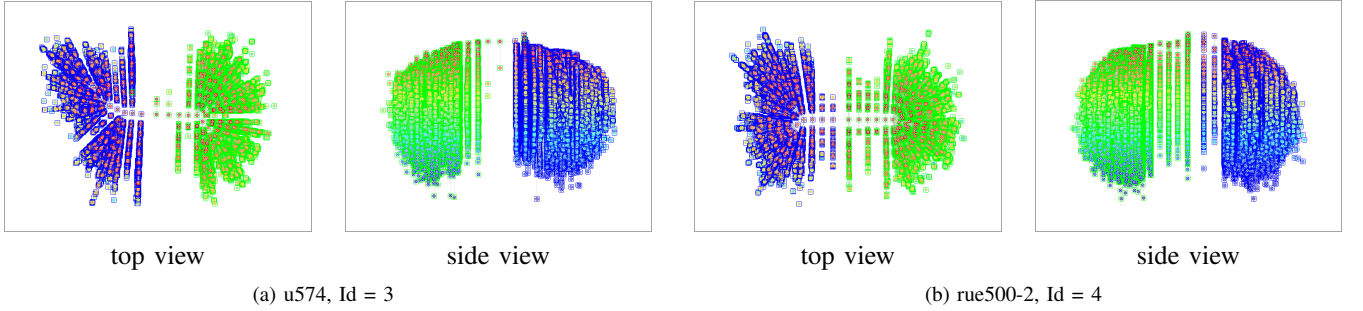


Fig. 9. NBN of the combined data of $N = 10^6$ solutions from local sampling with a radius of $K = 17$ around the possible deceptive solution and global optima, where the blue rectangles represent the solutions around the possible deceptive solution and the green rectangles represent the solutions around the optima.

In Fig. 8, we observe that solutions exist within the BoAs of the global optima in both success and failure cases. Even in trajectory T_{15} when the distance to the optima is relatively large, $d(\tilde{P}(T_{15}, \mathbf{o})) = 17$, there still exist several solutions in BoA of the global optima. EAX does not suffer from a lack of diversity in detecting the BoA of the optimum. On the contrary, it successfully locates the BoA of the global optima.

Moreover, TABLE II shows that $d(\tilde{P}(\mathbf{T}, \mathbf{o}))$ varies significantly across different trajectories, suggesting that the algorithm converges to different locations in these trajectories, which also validates the ability of EAX to maintain diversity.

Although EAX can detect the BoA of global optima in all these trajectories, it still has a low success rate on rue500-1 as shown in TABLE I. Then, we further analyze EAX's behavior on rue500-1. In the 9th trajectory T_9 , the distance between T_9 and the global optima \mathbf{o} is only 3. It seems that the local search operators applied to the nearest solution could easily find the global optimum. However, EAX relies on edge assembly crossover to improve the solutions. When multiple BoAs exist, EAX retains individuals within multiple basins simultaneously, reducing inter-basin interaction efficiency and

leading to algorithm's stagnation. The result of the optima screened according to Eq. (23) with $\theta = 0.99$ and $\vartheta = 30$ also supports this conclusion, where rue500-1 has 5 optima, while the other two instances have only 2 optima. All the optima are marked as circles in Fig. 8.

LKH's behaviors: LKH suffers from a unique failure mode: it frequently converges to deceptive local optima, particularly in rue500-2, where modality is not the main difficulty—deceptiveness is.

As shown in TABLE I, LKH struggles with rue500-2. Compared to the EAX's behaviors, we know that modality is not the challenge that LKH encounters.

In Fig. 8, we can see that in rue500-2 LKH tends to converge to the local optima and the blue diamond-shaped solution. Even in the LKH's successful trajectory, T_3 , there are many solutions around the blue diamond-shaped solution. Only one set of solutions (orange stars) generated in the same run are situated around the global optima. Besides, EAX also has many solutions around the blue diamond-shaped solution. It seems that EAX treats it like a local optimum.

The data in TABLE II also corroborates this phenomenon. Among the 26 failures, LKH is stuck in the blue diamond-

shaped solution 25 times. Then, is the solution deceptive? To verify this hypothesis, we need to answer two questions: (1) Why is that the other two instances do not have deceptive solutions? (2) Why is the blue diamond-shaped solution the only deceptive solution in rue500-2?

(1) Deceptive Solution Filtering: We know that a deceptive solution is a solution that is close to the global optimum with better BoAs, so algorithms are attracted by the deceptive solution and thus ignore the global optima. Based on this, we filter out all the potential deceptive solutions in all three instances as shown in TABLE III. The largest local search operator used by LKH is the 5-opt, which indicates that LKH can find the best solutions in a local area with a radius $K \leq 10$. For a solution with NBD smaller than 10, LKH can find its nearest better solution using the 5-opt local search operator.

Thus, any possible deceptive solution should have an NBD larger than 10 so that LKH can converge to it instead of the global optimum. Additionally, the deceptive solution should be closer to the global optimum, so that it can shadow the global optimum for LKH. Thus, we filter out all the possible potential solutions based on the following metric:

$$d_{\text{NBD}}(\mathbf{n}) \geq 10 \wedge \|\mathbf{n}, \mathbf{o}\| \leq 17 \quad (24)$$

Next, we analyze the local structure around the potential deceptive solution and the global optimum. We performed local sampling around both the deceptive solution and the global optimum, resulting in two solution sets, $\mathcal{S}(\mathbf{n}, K)$ and $\mathcal{S}(\mathbf{o}, K)$ with a sampling radius of $K = 17$. Then, we analyze the difference of the average fitness of the two solution sets, denoted as $\Delta \bar{f}$ in TABLE III, calculated using the following formula:

$$\Delta \bar{f} = \frac{\sum_{\mathbf{x}_i \in \mathcal{S}(\mathbf{o}, K)} f(\mathbf{x}_i)}{N} - \frac{\sum_{\mathbf{x}_i \in \mathcal{S}(\mathbf{n}, K)} f(\mathbf{x}_i)}{N} \quad (25)$$

From TABLE III, we observe that only two funnels have better local areas than the global optimum: Funnel 3 of u574 and Funnel 4 of rue500-2. Interestingly, Funnel 4 of rue500-2 is the deceptive solution that we predicted, i.e., the blue diamond-shaped solution in Fig. 8. This indicates that both funnels are potentially deceptive solutions.

(2) BoA Distance Analysis: We need to further analyze whether the BoAs of the two funnels are close to the BoA of the optima so that the solutions in the BoA of the global optimum are easily attracted by the funnel. By analyzing the NBN of the combined data of the two solution sets, as shown in TABLE III, we found out that the sampled solution set of funnel 4 of rue500-2 is closer to the global optimum compared to funnel 3 of u574, with a smaller $d(\mathcal{S}(\mathbf{n}, K), \mathbf{o})$. This distance is even shorter when compared to some other funnels, particularly in a larger sampled solution set ($N = 1e6$). Furthermore, Fig. 9 also shows that many solutions connect the two funnels in NBN of rue500-2, $\text{Id} = 4$ than in u574, $\text{Id} = 3$. This suggests that the solutions around the global optimum in rue500-2 are more easily attracted to funnel 4.

Note that the NBN created using the two local sampling datasets is biased, with few solutions located at the center of the funnel and the global optima. Therefore, the distance $d(\mathcal{S}(\mathbf{n}, K), \mathbf{o})$ may not be accurate, and the true value is likely

smaller. However, for different funnels, the distribution of the sampled solutions remains consistent, making the distances $d(\mathcal{S}(\mathbf{n}, K), \mathbf{o})$ of different funnels comparable.

NLKH's behaviors: NLKH improves over LKH by increasing the chance of reaching the global basin, but remains vulnerable to deceptive funnels due to biased learning from underrepresented complex instances.

As shown in Fig. 8 and supported by the data in TABLE II, the behavior of NLKH on rue500-2 closely resembles that of LKH, with the algorithm repeatedly falling into the deceptive funnel. Although NLKH employs a sparse graph neural network to infer edge and node weights, thereby identifying promising regions of the search space and increasing the likelihood of entering the global basin of attraction—ultimately achieving a higher success rate than LKH on rue500-2—it does not fully eliminate the influence of deceptive funnels through weight adjustment. As a result, NLKH remains susceptible to being trapped in the deceptive funnel.

Interestingly, on rue500-1, NLKH exhibits a lower success rate compared to LKH. In Fig. 8, NLKH demonstrates failure traits that are strikingly similar to those observed on rue500-2, with only a sparse presence of solutions in the global optimum basin. The data in TABLE II further confirms that, in each failure case, the obtained solutions are considerably distant from the global optimum, indicating that NLKH is attracted to other basins of local optima. Moreover, the algorithm tends to fall into different basins across runs, suggesting a bias in the learned node weights that favors basins of local optima.

A plausible explanation for this issue lies in the long-tail distribution of the training dataset. TSP instances with complex features—such as multiple basins of attraction or deceptive funnels—may be underrepresented during training, limiting the generalization ability of the sparse graph network to such problem types. Given that the training process of the network is not directly interpretable, further investigation is required to better understand its learning behavior and enhance its performance on structurally complex TSP instances.

5) Conclusions from the Analysis: Based on the NBN-assisted analysis, three primary challenges are identified for TSP: ruggedness, modality, and deception. Most TSP local search operators can overcome ruggedness challenges, as NBN structures generated from optimization data are remarkably smoother compared to randomly sampled NBN structures.

EAX struggles with modality, as its population often spreads across multiple basins, hindering convergence to the global optimum. LKH handles modality better via restarts but is prone to deception, often converging to nearby local optima. NLKH improves LKH by guiding search toward the global basin but still suffers from deceptive funnels due to biased training.

VI. CONCLUSIONS AND FUTURE WORK

In this paper, we offered a straightforward proof indicating that NBN fundamentally represents the maximum probability transition network. We also presented an efficient calculation method for NBN with logarithmic linear time complexity for assignment problems. Furthermore, we conducted an in-depth analysis in OneMax problems and TSP. For the first time, we

found that the fitness landscape of OneMax exhibits neutrality, ruggedness, and modality features. We also uncovered some limitations of the state-of-the-art TSP algorithms: LKH, which relies on its local search operators, fails when there are deceptive solutions near the global optima. While, EAX, based on a single population, efficiently maintains diversity. However, when multiple attraction basins exist, it retains individuals within multiple basins simultaneously, reducing inter-basin interaction efficiency and leading to algorithm's stagnation as well. NLKH outperforms LKH in reaching the global basin but still struggles with deceptive funnels due to biased training.

We believe that since NBN can reveal the underlying challenges of the problems, it can also solve them. To tackle the limitations of current TSP algorithms in dealing with modality and deception challenges, we aim to develop NBN-based algorithms capable of adaptively learning these landscape features.

REFERENCES

- [1] G. Laporte, "The vehicle routing problem: An overview of exact and approximate algorithms," *Eur. J. Oper. Res.*, vol. 59, no. 3, pp. 345–358, 1992.
- [2] M. Pop, "Genome assembly reborn: Recent computational challenges," *Brief. Bioinform.*, vol. 10, no. 4, pp. 354–366, 2009.
- [3] C. J. Alpert, D. P. Mehta, and S. S. Sapatnekar, Eds., *Handbook of Algorithms for Physical Design Automation*. CRC Press, 2008.
- [4] R. Tinós, K. Helsgaun, and D. Whitley, "Efficient recombination in the lin-kernighan-helsgaun traveling salesman heuristic," in *Proc. Parallel Problem Solving from Nature (PPSN XV)*, 2018, pp. 95–107.
- [5] Y. Nagata and S. Kobayashi, "A powerful genetic algorithm using edge assembly crossover for the traveling salesman problem," *INFORMS J. Comput.*, vol. 25, no. 2, pp. 346–363, 2013.
- [6] L. Xin, W. Song, Z. Cao, and J. Zhang, "NeuroLKH: Combining deep learning model with lin-kernighan-helsgaun heuristic for solving the traveling salesman problem," in *Adv. Neural Inf. Process. Syst. (NeurIPS)*, vol. 35, 2021, pp. 7472–7483.
- [7] J. Scholz, "Genetic algorithms and the traveling salesman problem: A historical review," *arXiv preprint arXiv:1901.05737*, 2019.
- [8] S. Liu, Y. Zhang, K. Tang, and X. Yao, "How good is neural combinatorial optimization? a systematic evaluation on the traveling salesman problem," *IEEE Comput. Intell. Mag.*, vol. 18, no. 3, pp. 14–28, 2023.
- [9] Y. Diao, C. Li, S. Zeng, S. Yang, and C. A. C. Coello, "Nearest-better network for fitness landscape analysis of continuous optimization problems," *IEEE Trans. Evol. Comput.*, 2024, early Access.
- [10] T. Weise and Z. Wu, "Difficult features of combinatorial optimization problems and the tunable W-model benchmark problem for simulating them," in *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO)*, 2018, pp. 1769–1776.
- [11] P. F. Stadler, "Fitness landscapes," in *Biological Evolution and Statistical Physics*, 2002, pp. 183–204.
- [12] F. Zou, D. Chen, H. Liu, S. Cao, X. Ji, and Y. Zhang, "A survey of fitness landscape analysis for optimization," *Neurocomputing*, vol. 503, pp. 129–139, 2022.
- [13] S. A. Kauffman and E. D. Weinberger, "The NK model of rugged fitness landscapes and its application to maturation of the immune response," *J. Theor. Biol.*, vol. 141, no. 2, pp. 211–245, 1989.
- [14] H. Aguirre and K. Tanaka, "Insights on properties of multiobjective MNK-landscapes," in *Proceedings of the IEEE Congress on Evolutionary Computation (CEC)*, vol. 1, 2004, pp. 196–203.
- [15] W. Beaudoin, S. Verel, P. Collard, and C. Escalut, "Deceptiveness and neutrality: The ND family of fitness landscapes," in *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO)*, 2006, pp. 507–514.
- [16] M. Mitchell, J. H. Holland, and S. Forrest, "The royal road for genetic algorithms: Fitness landscapes and GA performance," Santa Fe Institute, Tech. Rep., 1992.
- [17] D. F. Lochtefeld and F. W. Ciarallo, "Multiobjectivization via helper-objectives with the tunable objectives problem," *IEEE Trans. Evol. Comput.*, vol. 16, no. 3, pp. 373–390, 2012.
- [18] D. E. Goldberg, "Simple genetic algorithms and the minimal, deceptive problem," in *Genetic Algorithms and Simulated Annealing*. Pitman, 1987, pp. 74–88.
- [19] G. Ochoa, K. M. Malan, and C. Blum, "Search trajectory networks: A tool for analysing and visualising the behaviour of metaheuristics," *Appl. Soft Comput.*, vol. 109, p. 107492, 2021.
- [20] K. Michalak, "Low-dimensional Euclidean embedding for visualization of search spaces in combinatorial optimization," *IEEE Trans. Evol. Comput.*, vol. 23, no. 2, pp. 232–246, 2019.
- [21] T. Smith, P. Husbands, P. J. Layzell, and M. O'Shea, "Fitness landscapes and evolvability," *Evol. Comput.*, vol. 10, no. 1, pp. 1–34, 2002.
- [22] G. Ochoa, S. Verel, F. Daolio, and M. Tomassini, "Local optima networks: A new model of combinatorial fitness landscapes," in *Recent Advances in the Theory and Application of Fitness Landscapes*, 2014, pp. 233–262.
- [23] M. Lipsitch, "Adaptation on rugged landscapes generated by iterated local interactions of neighboring genes," in *Proc. Int. Conf. Genet. Algorithms (ICGA)*, 1991, pp. 128–135.
- [24] C. M. Reidys and P. F. Stadler, "Neutrality in fitness landscapes," *Appl. Math. Comput.*, vol. 117, no. 2–3, pp. 321–350, 2001.
- [25] R. Morgan and M. Gallagher, "Analysing and characterising optimization problems using length scale," *Soft Comput.*, vol. 21, no. 7, pp. 1735–1752, 2017.
- [26] A. S. Bosman, A. P. Engelbrecht, and M. Helbig, "Visualising basins of attraction for the cross-entropy and the squared error neural network loss functions," *Neurocomputing*, vol. 400, pp. 113–136, 2020.
- [27] K. M. Malan, "Characterising continuous optimisation problems for particle swarm optimisation performance prediction," Ph.D. dissertation, University of Pretoria, 2014.
- [28] C. Fonlupt, D. Robilliard, and P. Preux, "A bit-wise epistasis measure for binary search spaces," in *Proc. Int. Conf. Parallel Problem Solving from Nature (PPSN)*, 1998, pp. 47–56.
- [29] M. Lunacek and D. Whitley, "The dispersion metric and the CMA evolution strategy," in *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO)*, 2006, pp. 477–484.
- [30] W. Li, X. Meng, and Y. Huang, "Fitness distance correlation and mixed search strategy for differential evolution," *Neurocomputing*, vol. 458, pp. 514–525, 2021.
- [31] W. Luo, X. Lin, J. Zhang, and M. Preuss, "A survey of nearest-better clustering in swarm and evolutionary computation," in *Proceedings of the IEEE Congress on Evolutionary Computation (CEC)*, 2021, pp. 1961–1967.
- [32] M. Preuss, "Basin identification by means of nearest-better clustering," in *Multimodal Optimization by Means of Evolutionary Algorithms*, 2015, pp. 75–114.
- [33] —, "Nearest-better-based niching," in *Multimodal Optimization by Means of Evolutionary Algorithms*, 2015, pp. 139–170.
- [34] X. Lin, W. Luo, and P. Xu, "Differential evolution for multimodal optimization with species by nearest-better clustering," *IEEE Trans. Cybern.*, vol. 51, no. 2, pp. 970–983, 2021.
- [35] Y. Diao, C. Li, S. Zeng, and S. Yang, "Nearest better network for visualization of the fitness landscape," in *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO)*, 2023, pp. 815–818.
- [36] S. Droste, T. Jansen, and I. Wegener, "On the analysis of the (1+1) evolutionary algorithm," *Theor. Comput. Sci.*, vol. 276, no. 1, pp. 51–81, 2002.
- [37] C. P. Robert and G. Casella, *Monte Carlo Statistical Methods*. Springer, 2004.
- [38] N. Hansen and A. Ostermeier, "Completely derandomized self-adaptation in evolution strategies," *Evol. Comput.*, vol. 9, no. 2, pp. 159–195, 2001.
- [39] M. Preuss, "Niching the CMA-ES via nearest-better clustering," in *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO)*, 2010, pp. 1711–1718.
- [40] S. Raggl, A. Beham, V. A. Haider, S. Wagner, and M. Affenzeller, "Discrete real-world problems in a black-box optimization benchmark," in *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO)*, 2018, pp. 1745–1752.
- [41] L. R. Dice, "Measures of the amount of ecologic association between species," *Ecology*, vol. 26, no. 3, pp. 297–302, 1945.
- [42] M. Dorigo and T. Stützle, "Ant colony optimization: Overview and recent advances," in *Handbook of Metaheuristics*, 2019, pp. 311–351.
- [43] J. L. Bentley, "Multidimensional binary search trees used for associative searching," *Commun. ACM*, vol. 18, no. 9, pp. 509–517, 1975.

- [44] J. M. Kleinberg, "Two algorithms for nearest-neighbor search in high dimensions," in *Proc. ACM Symp. Theory Comput. (STOC)*, 1997, pp. 599–608.
- [45] G. Reinelt, "TSPLIB—a traveling salesman problem library," *ORSA J. Comput.*, vol. 3, no. 4, pp. 376–384, 1991.
- [46] G. Gutin and A. Punnen, *The Traveling Salesman Problem and Its Variations*. Springer, 2006.
- [47] G. Ochoa and N. Veerapen, "Mapping the global structure of TSP fitness landscapes," *J. Heuristics*, vol. 24, no. 3, pp. 265–294, 2018.



Yiya Diao received her Ph.D. degree in Control Science and Engineering from China University of Geosciences, Wuhan, China, in 2025. She is currently a postdoctoral researcher at the School of Computer Science and Technology, Harbin Institute of Technology, Shenzhen, Guangdong, China. Her research interests include fitness landscape analysis and combinatorial optimization.



Changhe Li (Member, IEEE) received the B.Sc. and M.Sc. degrees in computer science from the China University of Geosciences, Wuhan, China, in 2005 and 2008, respectively, and the Ph.D. degree in computer science from the University of Leicester, Leicester, U.K., in 2011.

He is currently a Professor with the School of Artificial Intelligence, Anhui University of Science & Technology. His research interests include intelligent optimization and machine learning.



Sanyou Zeng received the M.Sc. degree in mathematics from Hunan University, Changsha, China, in 1995, and the Ph.D. degree in computer science from Wuhan University, Wuhan, China, in 2002.

He is currently a Professor with the China University of Geosciences, Wuhan. His current research interests include evolutionary computation with machine learning for solving problems with constraints, multiobjective, dynamic environments, and expensive costs, especially antenna design problems.



XinYe Cai (M'14) received the BSc in information engineering from Huazhong University of Science and Technology, China in 2004, the MSc in electronic engineering from University of York, UK, in 2006 and the PhD in electrical engineering from Kansas State University, U.S, in 2009, respectively.

He was a full Professor with School of Control Sciences and Engineering, Dalian University of Technology, China. Currently he is a full Professor with State Key Laboratory of Digital Intelligent Technology for Unmanned Coal Mining, as well as

School of Artificial Intelligence, Anhui University of Science & Technology, China. His current research interests include intelligent optimization, machine learning, computer vision, industrial artificial intelligence, and their applications in smart mine. Prof. Cai is an Associate Editor of *Swarm and Evolutionary Computation*.



Wenjian Luo received the BS and PhD degrees from Department of Computer Science and Technology, University of Science and Technology of China, Hefei, China, in 1998 and 2003. He is presently a professor of School of Computer Science and Technology, Harbin Institute of Technology, Shenzhen, China. His research interest mainly focuses on the secure intelligent systems, including artificial intelligence security, privacy computing, big data analysis, as well as the fundamental algorithms and techniques of machine learning, immune computing,

swarm intelligence. He has published more than 100 international journal and conference papers. He is a distinguished member of CCF and a senior member of IEEE, ACM, and CAAI. He currently serves as an area editor of *Applied Soft Computing Journal*, and an associate editor (or editorial board member) for several journals including *Information Sciences Journal*, *Swarm and Evolutionary Computation Journal*, *Journal of Information Security and Applications*, *Complex & Intelligent Systems Journal*, and *Systems and Soft Computing Journal*. He served as the chair of the IEEE CIS ECTC Task Force on Artificial Immune Systems from 2018 to 2022. He has been a member of the organizational team of more than twenty academic conferences, in various functions, such as general chair, program chair, symposium chair, tutorial chair and publicity chair.



Shengxiang Yang (Senior Member, IEEE) received the Ph.D. degree from Northeastern University, Shenyang, China, in 1999. He is currently a Professor with the School of Computer Science and Informatics, De Montfort University, Leicester LE1 9BH, UK. He has more than 500 publications with an H-index of 80 in Google Scholar.

His current research interests include evolutionary computation, swarm intelligence, artificial neural networks, data mining and data stream mining, and relevant real-world applications. He serves as

an Associate Editor/Editorial Board Member of a number of international journals, such as the *IEEE Transactions on Evolutionary Computation*, *IEEE Transactions on Cybernetics*, *Information Sciences*, *Enterprise Information Systems*, and *CAAI Transactions on Intelligence Technology*.



Carlos A. Coello Coello (Fellow, IEEE) received a Ph.D. in computer science from Tulane University, New Orleans, LA, USA, in 1996.

He is a Professor (CINVESTAV-3F Researcher) with the Department of Computer Science of CINVESTAV-IPN, Mexico City, Mexico. He has authored and coauthored over 550 technical papers and book chapters. He has also coauthored the book *Evolutionary Algorithms for Solving Multi-Objective Problems* (Second Edition, Springer, 2007). His publications currently report over 75400 citations

in Google Scholar (his H-index is 104). His research interests include evolutionary multiobjective optimization and constraint-handling techniques for evolutionary algorithms.