

Article

A Comparative Study of Structure-Preserving Representations for Variable-Turbine-Number Wind Farm Layout Optimization

Haichuan Yang ¹, Yifei Yang ^{2,*}, Shanxian Lin ¹, Qiong Fu ³, Guodong Cui ⁴, Ancai Zhang ⁵
and Yuichi Nagata ^{1,*}

- ¹ Graduate School of Technology, Industrial and Social Sciences, Tokushima University, Tokushima 770-8506, Japan; you.kaisen@tokushima-u.ac.jp (H.Y.); c612535073@tokushima-u.ac.jp (S.L.)
² Faculty of Science and Technology, Hirosaki University, Hirosaki 036-8560, Japan
³ Faculty of Engineering, University of Toyama, Toyama 930-8555, Japan; d2378004@ems.u-toyama.ac.jp
⁴ Department of Engineering, Wesoft Company Ltd., Kawasaki 210-0022, Japan; kt.sai@wesoft.co.jp
⁵ School of Automation and Electrical Engineering, Linyi University, Linyi 276000, China; zhangancai@lyu.edu.cn
* Correspondence: yyf7236@hirosaki-u.ac.jp (Y.Y.); nagata@is.tokushima-u.ac.jp (Y.N.)

Abstract

Wind farm layout optimization is commonly studied under a fixed number of turbines, although turbine quantity is closely coupled with wake effects and power generation efficiency. This study formulates the problem as a variable-turbine-number optimization task and systematically compares three representative variable-length approaches: dimension activation/deactivation, representation mapping, and variable-length evolutionary representation. The study also investigates structure-preserving search strategies based on binary occupancy representations. The findings suggest that, for variable-turbine-number wind farm layout optimization, the effective preservation and exploitation of spatial layout structures are more critical than merely handling dimensional changes. From a fitness-landscape perspective, structure-preserving representations appear to create more coherent neighborhoods and more accessible search topologies. These properties facilitate transitions across heterogeneous turbine-number subspaces and improve search performance.

Keywords: wind farm layout optimization; structure-preserving representations; metaheuristics

MSC: 90C59; 68W50; 90C27; 90C26; 65K10



Academic Editors: Kaizhou Gao,
Naiqi Wu and Yaping Fu

Received: 18 June 2026

Revised: 23 July 2026

Accepted: 24 July 2026

Published: 29 July 2026

Copyright: © 2026 by the authors.
Licensee MDPI, Basel, Switzerland.
This article is an open access article
distributed under the terms and
conditions of the [Creative Commons
Attribution \(CC BY\)](https://creativecommons.org/licenses/by/4.0/) license.

1. Introduction

Wind Farm Layout Optimization (WFLO) aims to improve wind farm efficiency and power generation performance by optimizing turbine placement under wake interactions [1,2]. Figure 1 illustrates wind farm layouts before and after optimization. Since WFLO is typically formulated as a computationally expensive black-box optimization problem, metaheuristic algorithms (MHAs) are widely used to obtain high-quality approximate solutions without requiring gradient information [3–5]. Previous WFLO studies have commonly assumed a fixed number of turbines when evaluating layout optimization methods. However, in practical applications, the number of turbines is closely coupled with wake interactions and overall wind farm efficiency. Different algorithms may exhibit substantially different performance at different turbine scales [6], making it difficult to identify both the optimal turbine number and the corresponding layout through repeated fixed-scale optimization experiments [7,8].

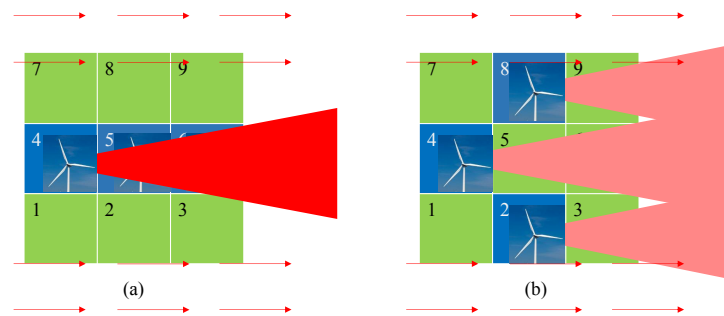


Figure 1. Illustration of wind farm layouts before optimization (a) and after optimization (b). The arrows indicate the wind direction, and the red shaded regions represent the wake-affected areas downstream of the wind turbines, where the wind speed is reduced due to wake effects.

Treating the turbine number as a decision variable provides a more flexible way to address the coupling between turbine quantity and wind farm efficiency. However, variable-turbine-number WFLO introduces additional optimization challenges, since changes in the turbine number alter the effective solution structure and search-space characteristics [9]. Such heterogeneous search spaces may exhibit increased discontinuity and complexity, making effective exploration more difficult [10–12]. This problem can therefore be regarded as a variable-length optimization problem [13].

To avoid explicit dimensionality changes, some studies have attempted to address this trade-off using multi-objective optimization [14]. Typical objectives include power generation, efficiency, cost, and environmental impacts [15–20]. More recent studies have explicitly considered the turbine number together with total power output [21]. In contrast, this study focuses on explicitly modeling the turbine number within the solution representation, allowing variable-turbine-number WFLO to be treated as a variable-structure optimization problem.

Existing variable-dimensional approaches can be broadly categorized into three types:

- (1) Dimension activation/deactivation methods [22,23], which embed the problem in a fixed-dimensional space and control the effective number of variables by activating or deactivating selected dimensions.
- (2) Representation mapping approaches [24,25], which maintain fixed-length representations and transform them into variable-length solutions through dedicated mapping or decoding schemes.
- (3) Variable-length evolutionary representations [13], typically realized by Genetic Algorithms (GA) [26] with flexible chromosome structures, where the number of encoded components is directly determined by the chromosome itself. In these methods, crossover and mutation operate on representations whose lengths are not fixed a priori, enabling variable-dimensional solutions to be handled naturally without explicit dimension activation or mapping.

Building on this classification, this study investigates three representative variable-structure approaches to WFLO and integrates them with three metaheuristic frameworks: GA, Differential Evolution (DE) [27], and Spherical Evolution (SE) [28]. These optimizers are selected as representative population-based search methods with different search characteristics and have been widely applied to continuous optimization and WFLO problems [29–31]. We evaluate these methods in terms of convergence, stability, and adaptability to varying turbine numbers. Experimental results suggest that optimization performance is related not only to how turbine-number variation is handled but also to how spatial layout structures are preserved during search. In particular, representations that maintain more explicit structural information tend to show greater adaptability under heterogeneous variable-turbine-number WFLO scenarios.

These observations motivate an additional investigation of structure-preserving search methods based on binary occupancy representations [13,32]. Unlike variable-length sequence representations, binary occupancy representations explicitly preserve grid-level spatial occupancy

information. Based on this representation, different structure-aware search mechanisms are introduced, including standard crossover-based evolution, region-based structural recombination, and local perturbation search using simulated annealing (SA) [33]. The results of these additional investigations further support the observation that preserving meaningful spatial layout structures during search may benefit optimization performance in variable-turbine-number WFLO.

The remainder of this paper is organized as follows. Section 2 reviews related work, and Section 3 introduces the metaheuristic frameworks. Section 4 presents the investigated variable-structure representations and search methods. Section 5 describes the experimental setup and comparative results, while Section 6 discusses their implications. Section 7 concludes the paper. Appendix A provides a detailed description of the fitness function used to evaluate the wind farm layouts.

2. Related Work

Recent WFLO studies from 2022 to 2026 have mainly improved model fidelity and optimization efficiency through realistic wind data and grid-resolution analysis [34], wake and complex-terrain modeling [35,36], surrogate-assisted optimization [37], multi-objective layout expansion [38], reinforcement-learning-assisted search [39], further surrogate-assisted optimization [40], and practical engineering constraints [41]. While these studies enhance the realism, efficiency, and applicability of WFLO, they generally adopt a predefined solution representation and focus on physical modeling, objective formulation, or search efficiency.

In contrast, this study addresses how variable-turbine-number layouts should be represented and searched. Existing variable-length methods developed for neural architecture search [42,43] and service composition [44] do not explicitly account for the wake-coupled spatial dependencies of WFLO. Therefore, this study compares alternative representation schemes and investigates whether preserving grid-level occupancy structures improves search across heterogeneous turbine-number subspaces.

The above distinction is further clarified in Table 1: recent studies mainly improve physical modeling, surrogate evaluation, objectives, or constraints, whereas this study focuses on the mathematical representation and structure-preserving search of variable-turbine-number layouts.

Table 1. Comparison with recent WFLO optimization studies.

Study	Main Focus	Variable Turbine Number	Representation-Level Distinction
Masoudi et al. [34]	Realistic wind data and grid resolution	No	Uses a prescribed layout representation.
Cao et al. [35]	Wake-induced turbulence modeling	No	Improves the physical model rather than the representation.
Hu et al. [36]	Complex-terrain modeling	No	Focuses on environmental fidelity.
Wang et al. [37]	CFD/Kriging surrogate optimization	No	Reduces evaluation cost under a given representation.
Boucekara et al. [38]	Multi-objective wind farm layout/expansion	Yes	Uses binary grid occupancy to vary turbine number, but focuses on power-efficiency trade-offs rather than representation-level comparison.
Dong et al. [39]	Reinforcement-learning-assisted GA	No	Improves search decisions within a predefined encoding.
Shao et al. [40]	PCE/Kriging surrogate optimization	No	Focuses on computational efficiency.
Malisani et al. [41]	Offshore alignment constraints	No	Focuses on alignment constraints rather than representation design.
This study	Structure-preserving variable-number WFLO	Yes	Provides a systematic representation-level comparison and introduces structure-preserving binary occupancy methods for variable-turbine-number WFLO.

3. Metaheuristic Algorithms

The MHAs examined in this study comprise four core operations: initialization, search, evaluation, and selection. The notation used in the variable-length approaches is summarized in Table 2.

Table 2. Core mathematical notation used in Section 3.

Symbol	Definition
n	Population size.
d	Dimension of the search space; in this study, it corresponds to the number of available grid cells.
$X_i^t = (x_{i,1}^t, x_{i,2}^t, \dots, x_{i,d}^t)$	The i -th individual at iteration t , where $x_{i,j}^t$ denotes its j -th component.
$I_{i,j}$	Discretized grid-cell index obtained from $x_{i,j}$ by rounding and boundary correction.
$O_i = (O_{i,1}, O_{i,2}, \dots, O_{i,d})$	The i -th offspring solution, where $O_{i,j}$ denotes its j -th component.
$r_1, r_2, r_3, \dots$	Indices of randomly selected parent individuals.
$S(\cdot)$	Generic population-based search operator.
c	Crossover point used in the one-point crossover operation of GA.
δ	Mutation perturbation used in GA.
F	Scaling factor used in DE.
CR	Crossover rate used in DE.
u_j	A random number sampled from $[0, 1]$ for the j -th dimension in DE.
SF	Scale factor used in SE to control the search radius.
η	Reduced dimensionality used in SE, where $\eta \leq d$.
θ_j	Random angular variable used in SE, sampled from $[0, 2\pi]$.
$\ \cdot\ _2$	Euclidean norm.
$f(\cdot)$	Fitness function used to evaluate a candidate layout.
b	Index of the parent individual considered for replacement during selection.

3.1. Initialization

During the initialization phase, the algorithm parameters are set. Then, a population of n individuals is randomly generated in a d -dimensional search space. Each individual is denoted by

$$X_i = (x_{i,1}, x_{i,2}, \dots, x_{i,d}) \in \mathbb{R}^d. \quad (1)$$

Let d denote the number of available discrete cells in the wind farm, i.e., the maximum number of candidate turbine locations after excluding infeasible regions. Each solution is represented as a d -dimensional vector X_i , where each coordinate $x_{i,j}$ is interpreted as a latent index of a grid cell. The real-valued vector is discretized by

$$I_{i,j} = \min\{d, \max\{1, \text{round}(x_{i,j})\}\}, \quad I_{i,j} \in \{1, 2, \dots, d\}. \quad (2)$$

Since the search operators may generate real-valued components outside the valid grid-index range, boundary correction is applied after rounding. Specifically, any rounded value smaller than 1 is clipped to 1, and any rounded value larger than d is clipped to d .

3.2. Search Operation

According to Tang's study [28], the search operation of many MHAs exhibits several recurring structural patterns or behavioral characteristics, formulated as follows:

$$O_{i,j} = X_{r_1,j}^t + S\left(X_{r_2,j}^t, X_{r_3,j}^t, \dots, X_{r_n,j}^t\right). \quad (3)$$

In this equation, t denotes the iteration index, and $O_{i,j}$ represents the j -th dimension of the i -th offspring individual generated during the transition from iteration t to $t + 1$. $X_{r_1,j}^t$ denotes the j -th dimension of the r_1 -th parent individual in the population at iteration

t , while $(X_{r_{2,j}}^t, X_{r_{3,j}}^t, \dots, X_{r_{n,j}}^t)$ are the corresponding dimensions of other selected parent individuals. The operator $S(\cdot)$ represents a population-based search operator whose specific form depends on the employed MHAs.

3.2.1. Search Operations of Genetic Algorithm

The search operation of the GA in this study can be represented as follows.

$$O_{i,j} = \begin{cases} X_{i,j}^t, & 1 \leq i \leq \lfloor 0.25 \cdot n \rfloor \text{ and } 1 \leq j \leq d, \quad \text{elitism,} \\ X_{r_1,j}^t, & \lfloor 0.25 \cdot n \rfloor < i \leq \lfloor 0.9 \cdot n \rfloor \text{ and } 1 \leq j < c, \quad \text{crossover,} \\ X_{r_2,j}^t, & \lfloor 0.25 \cdot n \rfloor < i \leq \lfloor 0.9 \cdot n \rfloor \text{ and } c \leq j \leq d, \quad \text{crossover,} \\ X_{i,j}^t + \delta, & \lfloor 0.9 \cdot n \rfloor < i \leq n \text{ and } 1 \leq j \leq d, \quad \text{mutation,} \end{cases} \quad (4)$$

where c is a randomly selected crossover point within the length d of the individuals, δ denotes a perturbation (e.g., a flip or random noise), and X_{r_1} and X_{r_2} are randomly selected individuals from the parent population. Here, all individuals are sorted from best to worst according to their fitness. It should be noted that the crossover operation used here is one-point crossover, which is consistently adopted in all other GA implementations throughout this paper.

3.2.2. Search Operations of Differential Evolution

Since its introduction in 1997, DE has undergone nearly three decades of development, resulting in various variants of search operations. The following highlights one of the most representative search operations:

$$O_{i,j} = \begin{cases} X_{r_1,j}^t + F \cdot (X_{r_2,j}^t - X_{r_3,j}^t), & \text{if } u_j < CR, \\ X_{i,j}^t, & \text{if otherwise,} \end{cases} \quad (5)$$

where F and CR are hyperparameters, u_j is a random value between 0 and 1, and X_{r_1} , X_{r_2} , and X_{r_3} are randomly selected individuals from the parent population. SHADE [45] is an improved DE algorithm because it updates the values of F and CR following the Cauchy and normal distributions, respectively, while keeping track of the successful parameter history. These adaptive parameters significantly balance the exploitation and exploration capabilities of the DE, allowing it to achieve excellent performance on the standard test set.

3.2.3. Search Operations of Spherical Evolution

When SE was proposed, it emulated DE's multi-search strategy. The following outlines some of the most representative search operations:

$$O_{i,j} = X_{r_1,j}^t + \begin{cases} SF \cdot \|X_{r_2,*}^t - X_{r_3,*}^t\|_2 \cdot \prod_{k=j}^{\eta-1} \sin(\theta_k), & \text{if } j = 1, \\ SF \cdot \|X_{r_2,*}^t - X_{r_3,*}^t\|_2 \cdot \cos(\theta_{j-1}) \cdot \prod_{k=j}^{\eta-1} \sin(\theta_k), & \text{if } 1 < j \leq \eta - 1, \\ SF \cdot \|X_{r_2,*}^t - X_{r_3,*}^t\|_2 \cdot \cos(\theta_{j-1}), & \text{if } j = \eta, \end{cases} \quad (6)$$

where X_{r_1} , X_{r_2} , and X_{r_3} are randomly selected individuals from the parent population. Index i denotes the individual and j the dimension. SF is a scale factor used to adjust the search radius, with its value being greater for worse individuals. $\|X_{r_2,*}^t - X_{r_3,*}^t\|_2$ gives the radius (L2 norm) of the hypersphere in the reduced space, where $\eta \leq d$ is the dimension after reduction and “*” denotes all the η dimensions. Each angle θ_j is drawn from a uniform distribution on $[0, 2\pi]$.

Figure 2 illustrates the workflows of GA, DE, and SE. On the left, each matrix represents a population, where rows and columns denote individuals and solution dimensions, respectively. X_1^t , X_2^t , and X_3^t are parent individuals, while X_1^{t+1} , X_2^{t+1} , and X_3^{t+1} denote their

offspring. Red squares indicate updated dimensions, with more red squares corresponding to larger solution changes. On the right, the 2D plots depict the search space, where dashed contour lines represent fitness iso-curves. The GA performs discrete recombination toward promising regions, DE generates adaptive directional perturbations based on population differences, and SE explores broader hyperspherical regions through stochastic scaling.

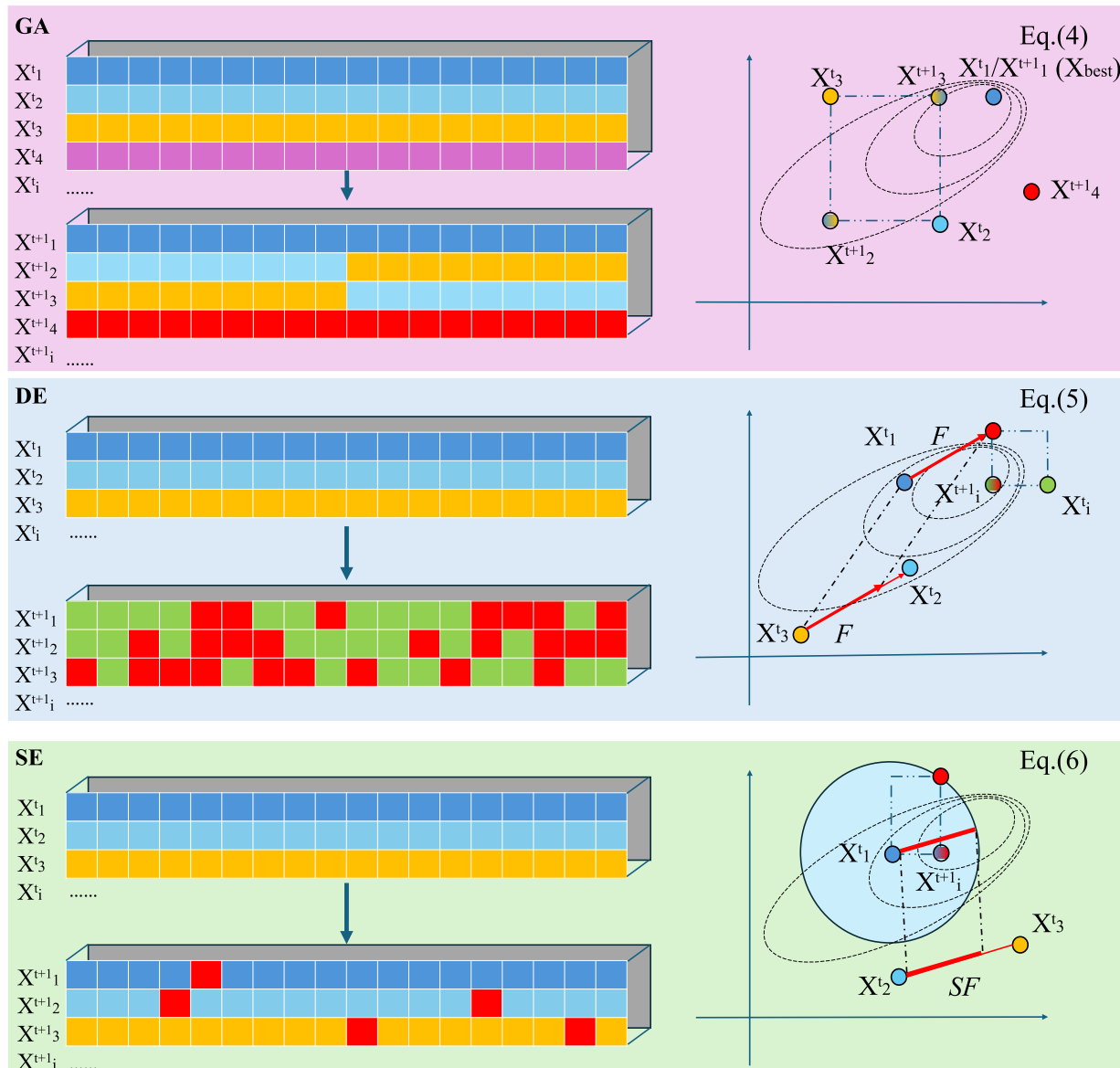


Figure 2. The workflow of GA, DE, and SE. In the population matrices on the left, rows represent individuals and columns represent solution dimensions, while red cells indicate updated dimensions. In the two-dimensional search-space illustrations on the right, the x- and y-axes represent two selected solution dimensions, and the dashed curves denote fitness contours. The arrows illustrate the search directions generated by the corresponding algorithms.

3.3. Evaluation and Selection

Following the search operation, the MHAs evaluate the fitness of all offspring individuals in O . Each offspring O_i is compared with a corresponding parent individual in the current population X^t , and the superior solution is retained for the next generation:

$$X^{t+1}_b = \begin{cases} O_i, & \text{if } f(O_i) > f(X^t_b), \\ X^t_b, & \text{otherwise.} \end{cases} \quad (7)$$

This formulation considers a maximization problem, where $f(\cdot)$ denotes the problem-specific fitness function used in this study to evaluate the quality of a candidate layout, whose detailed computation is provided in Appendix A. The index b denotes the parent individual subject to replacement according to the selection strategy of the employed MHA.

In one-to-one replacement schemes, such as DE and SE, each offspring competes directly with its corresponding parent, and thus $b = i$, where $i \in \{1, 2, \dots, n\}$. In population-level selection schemes, such as GA, individuals are first ranked according to fitness, and only the top 25% are retained for the next generation. In this case, $b = i$, where both indices are defined over the sorted index set $\{1, 2, \dots, \lfloor 0.25n \rfloor\}$, with $b = i = 1$ corresponding to the best individual and $b = i = \lfloor 0.25n \rfloor$ denoting the individual at the 25th percentile.

4. Variable-Length Approaches

Variable-length optimization problems can generally be addressed through three main types of approaches: dimension activation/deactivation methods, representation mapping approaches, and variable-length evolutionary representations. Among them, representation mapping approaches have already been applied to the WFLO problem in our previous work [25]. Therefore, this paper does not repeat that formulation in detail. In this study, we focus on the other two categories. Specifically, the dimension activation/deactivation method and the variable-length evolutionary representation are newly designed for the variable-length WFLO problem considered in this work.

In addition to these representative variable-length paradigms, this study further investigates a series of binary occupancy-based structural search methods. In these methods, a candidate layout is represented by a fixed-length binary array whose dimension is equal to the number of grid cells. Each element of the array corresponds to one grid cell, where a value of 1 indicates that a turbine is installed in the cell, and a value of 0 indicates that no turbine is installed. Although these methods employ fixed-length binary representations, the effective turbine number is adaptively determined by the number of activated grid cells. These methods support three levels of structural preservation: standard one-dimensional crossover, two-dimensional structure-aware crossover, and local structure-preserving perturbation.

The additional notation used in the variable-length approaches is summarized in Table 3. To avoid redundancy, this table only includes the core symbols that are repeatedly used across multiple equations or representation schemes in Section 4. Local variables, such as c_k , h , w , r_s , c_s , r_e , c_e , p_0 , and p_1 , are defined immediately before or after their corresponding equations and are therefore not included in the table.

Table 3. Additional notation used in Section 4.

Symbol	Definition
I_i	Integer index vector obtained by discretizing X_i .
$\mathcal{P}_i$	Set of occupied grid-cell indices after duplicate removal.
b_i	Binary occupancy vector, where $b_{i,j} = 1$ indicates that grid cell j is occupied.
n_i	Number of occupied grid cells in the activation/deactivation representation.
m_i	Effective turbine number, or chromosome length in the variable-length representation.
$\Phi(\cdot)$	Mapping from a grid-cell index to its physical coordinate.
L_i	Wind farm layout decoded from the i -th solution.
$\Pi(\cdot)$	Repair or constraint-handling operator for ensuring feasibility.
Ω_{NA}	Set of infeasible grid cells where turbines cannot be installed.
B_i	Two-dimensional binary layout matrix reshaped from X_i .
R, C	Numbers of rows and columns of the wind farm grid, respectively.
p_m	Bit-flip mutation probability in the binary occupancy representation.
T	Temperature parameter used in simulated annealing.

4.1. Dimension Activation/Deactivation Methods

In this study, activation is realized via real-to-binary encoding, where duplicate indices are removed, thereby implicitly adjusting the turbine number. Specifically, each real-valued vector X_i is first discretized into an integer vector $I_i = (I_{i,1}, \dots, I_{i,d})$ according to Equation (2). Based on this integer representation, turbine-count variability is induced by merging duplicate indices. We define the set of occupied grid indices as

$$\mathcal{P}_i = \{I_{i,1}, I_{i,2}, \dots, I_{i,n_i}\}, \quad (8)$$

where duplicates are automatically removed by the set operation. The resulting turbine count is $n_i = |\mathcal{P}_i| \leq d$.

A binary occupancy vector $b_i \in \{0, 1\}^d$ is then constructed via the indicator function

$$b_{i,j} = \mathbb{I}(j \in \mathcal{P}_i) = \begin{cases} 1, & \exists k \text{ s.t. } I_{i,k} = j, \\ 0, & \text{otherwise,} \end{cases} \quad j = 1, \dots, d. \quad (9)$$

Hence, modifying X_i can create/remove duplicates in I_i , which adaptively decreases/increases n_i while keeping the search dimension fixed at d .

Let $\Phi(j) = (x_j, y_j)$ denote the predefined mapping from grid index j to its physical coordinates. The final layout is the set of turbine coordinates corresponding to occupied cells:

$$L_i = \{\Phi(j) \mid b_{i,j} = 1, j = 1, \dots, n_i\}. \quad (10)$$

Overall, the proposed encoding defines a deterministic pipeline $X_i \xrightarrow{\text{round}} I_i \xrightarrow{\text{deduplicate}} \mathcal{P}_i \xrightarrow{\text{integer-to-binary}} b_i \xrightarrow{\Phi} L_i$, which supports variable turbine numbers without explicit changes in dimensionality. The overall workflow of the proposed dimension activation/deactivation approach, including the real-to-binary encoding process and the generation of the final layout, is illustrated in Figure 3.

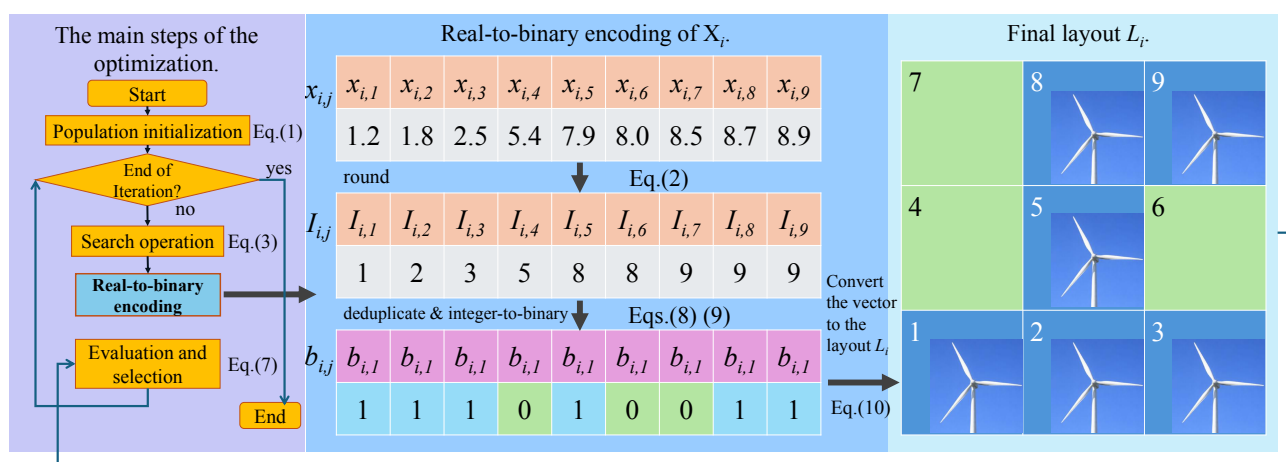


Figure 3. Dimension activation/deactivation methods. Here and in the subsequent figures, “Eq.” and “Eqs.” are used as abbreviations for “Equation” and “Equations”, respectively.

4.2. Variable-Length Evolutionary Representations

In contrast, variable-length evolutionary representations directly encode the turbine number in the solution structure. This formulation supports dynamic chromosomes and enables search across heterogeneous subspaces. During evolution, the vector length can change. Crossover and mutation are directly applied to variable-length representations, thereby explicitly controlling the turbine number. Specifically, X_i consists of m_i elements,

where $1 \leq m_i \leq d$, and m_i corresponds to the turbine number. During initialization, the length m_i of each individual is randomly generated.

It should be noted that this representation is only implemented within the GA framework in Equation (4). This is because GA operators are inherently sequence-based and can be naturally extended to individuals of varying lengths without modifying their fundamental structure. In contrast, algorithms such as DE and SE rely on dimension-wise arithmetic operations defined in a fixed-dimensional vector space. Their search operators require consistent dimensionality across individuals and cannot be directly applied to variable-length representations without redefining their core mechanisms.

During the search process, crossover and mutation operate directly on variable-length structures. The algorithm exhibits an implicit division of labor: mutation explores different turbine-number subspaces globally, while crossover refines layout structures locally within each subspace, leading to an effective exploration–exploitation balance. The offspring generation process under the variable-length representation can be formulated as

$$O_i = \begin{cases} X_i^t, & 1 \leq i \leq \lfloor 0.25n \rfloor, \text{ elitism,} \\ O_k^{(1)}, & i = \lfloor 0.25n \rfloor + k, \quad k = 1, \dots, \frac{n_c}{2}, \text{ crossover,} \\ O_k^{(2)}, & i = \lfloor 0.25n \rfloor + \frac{n_c}{2} + k, \quad k = 1, \dots, \frac{n_c}{2}, \text{ crossover,} \\ \mathcal{M}(), & \lfloor 0.9n \rfloor < i \leq n, \text{ mutation,} \end{cases} \quad (11)$$

where n denotes the population size, and $n_c = \lfloor 0.9n \rfloor - \lfloor 0.25n \rfloor$ denotes the number of offspring generated by crossover (assumed to be even). Here, X_i denotes the i -th individual in the population, as defined in Equation (1). The crossover phase produces pairs of offspring from selected parent individuals.

Let two parent individuals be randomly selected from the population, defined as $X_{r_1}^t = (x_{r_1,j}^t)_{j=1}^{m_{r_1}}$, $X_{r_2}^t = (x_{r_2,j}^t)_{j=1}^{m_{r_2}}$, where m_{r_1} and m_{r_2} denote the lengths of the corresponding individuals.

A crossover point is sampled within their common range: $c_k \sim U\{1, \dots, \min(m_{r_1}, m_{r_2})\}$. Then, the k -th pair of offspring is generated by variable-length one-point crossover:

$$O_k^{(1)} = (x_{r_1,j}^t)_{j=1}^{c_k} \parallel (x_{r_2,j}^t)_{j=c_k+1}^{m_{r_2}}, \quad (12)$$

$$O_k^{(2)} = (x_{r_2,j}^t)_{j=1}^{c_k} \parallel (x_{r_1,j}^t)_{j=c_k+1}^{m_{r_1}}, \quad (13)$$

where $\parallel$ denotes the concatenation operator.

Mutation is implemented by resampling the individual length and reconstructing a new solution. Specifically,

$$\mathcal{M}() = \Pi(X'), \quad X' \sim \mathcal{U}(\mathcal{X}_{m'_i}), \quad m'_i \sim U\{1, \dots, d\}, \quad (14)$$

where $\mathcal{X}_{m'_i}$ denotes the feasible solution space of dimension m'_i , and $\Pi(\cdot)$ represents a constraint-handling operator that ensures feasibility. The discretization follows Equation (2), with constraint handling applied to ensure feasibility. The final layout is also obtained via the mapping $\Phi(\cdot)$.

$$L_i = \{\Phi(I_{i,j}) \mid j = 1, \dots, m_i\}. \quad (15)$$

Here, $I_{i,j}$ is defined as in Equation (2). This operation preserves partial structural information while allowing offspring to inherit different lengths, enabling direct transitions across subspaces of different dimensionalities. Compared to dimension activation/deactivation methods, this representation eliminates the need for deduplication and binary encoding,

and it directly generates layouts from variable-length vectors. The general workflow of the proposed variable-length evolutionary representation approach is illustrated in Figure 4.

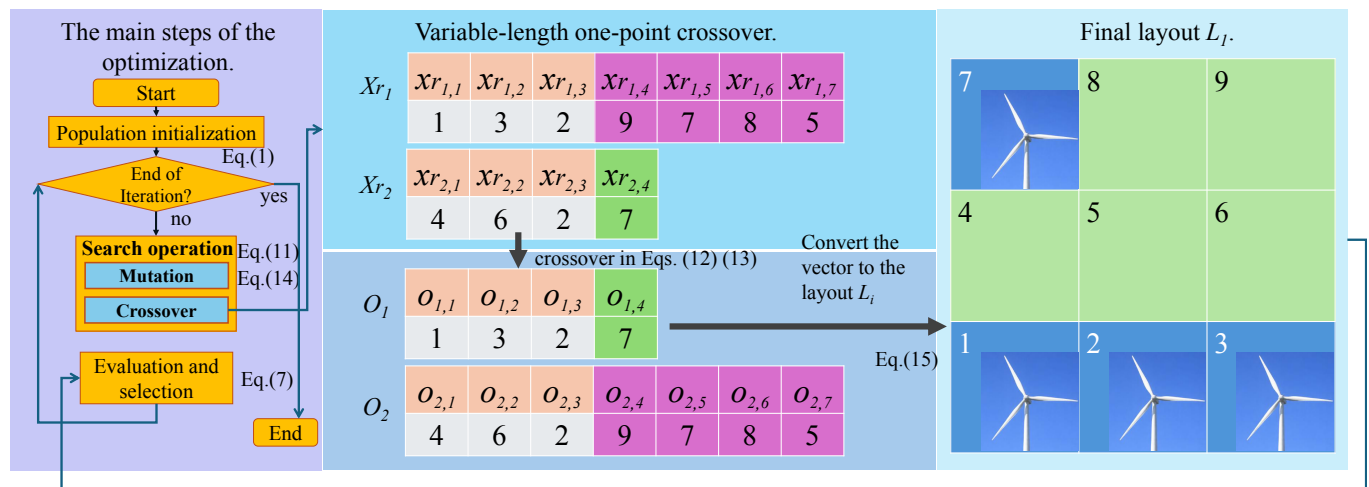


Figure 4. Variable-length evolutionary representations.

4.3. Binary Occupancy-Based Structural Search Methods

4.3.1. Standard Binary Occupancy GA

In this study, we further design a binary occupancy vector representation for the variable-length WFLO problem. Different from the real-valued activation/deactivation approach, turbine existence is directly represented by binary occupancy states. Specifically, each individual is encoded as

$$X_i = (x_{i,1}, x_{i,2}, \dots, x_{i,d}) \in \{0, 1\}^d, \quad (16)$$

where d denotes the total number of grid cells in the wind farm. The turbine number is determined by the number of activated components:

$$m_i = \sum_{j=1}^d x_{i,j}. \quad (17)$$

Thus, although the chromosome length remains fixed, the effective turbine number varies adaptively during evolution. During initialization, the turbine number m_i is randomly sampled from the feasible range, and the corresponding binary positions are randomly activated.

The search process follows the standard GA framework, including elitism, crossover, and mutation. The offspring generation process can be formulated as Equation (11). For crossover, two parent individuals X_{r1} and X_{r2} are selected, and a crossover point $c_k \sim U\{1, \dots, d-1\}$ is sampled. The offspring are generated by one-point crossover:

$$O_k^{(1)} = (x_{r1,j}^t)_{j=1}^{c_k} \parallel (x_{r2,j}^t)_{j=c_k+1}^d, \quad (18)$$

$$O_k^{(2)} = (x_{r2,j}^t)_{j=1}^{c_k} \parallel (x_{r1,j}^t)_{j=c_k+1}^d, \quad (19)$$

Mutation is implemented by bit flipping. For each dimension, the binary value is inverted with probability

$$p_m = \frac{1}{d}. \quad (20)$$

The choice of $p_m = 1/d$ serves as a simple adaptive mechanism, reducing the flipping probability of each bit as the dimensionality increases so that only a limited number of grid

cells are inverted even for larger wind farms, thereby effectively preserving the structural integrity of the current layout. The mutation operator is therefore defined as

$$M(X_i) = (\tilde{x}_{i,1}, \tilde{x}_{i,2}, \dots, \tilde{x}_{i,d}), \quad (21)$$

$$\tilde{x}_{i,j} = \begin{cases} 1 - x_{i,j}, & r_j < p_m, \\ x_{i,j}, & \text{otherwise,} \end{cases} \quad r_j \sim U(0,1). \quad (22)$$

After crossover and mutation, a repair operator $\Pi(\cdot)$ is applied to ensure feasibility. All infeasible cells are deactivated, and if no turbine remains, one feasible cell is randomly activated. The repaired offspring is represented as

$$\hat{O}_i = \Pi(O_i). \quad (23)$$

Let Ω_{NA} denote the infeasible grid-cell set and $\Phi(j)$ the mapping from grid index j to physical coordinates. The final layout is obtained by

$$L_i = \{\Phi(j) \mid \hat{x}_{i,j} = 1, j \notin \Omega_{NA}\}. \quad (24)$$

Compared with the dimension activation/deactivation method, the proposed representation directly manipulates binary occupancy states without real-valued discretization or duplicate removal. More importantly, compared with the variable-length evolutionary representation, the binary occupancy vector preserves the spatial existence state of every grid cell in the wind farm. In the variable-length evolutionary representation, a layout is encoded only as a sequence of selected turbine positions, and grid cells without turbines are not explicitly represented. As a result, crossover and mutation mainly operate on the selected-position sequence, which may weaken the continuity between the encoded structure and the complete spatial layout. In contrast, the binary occupancy representation maintains a one-to-one correspondence between chromosome components and grid cells, so both occupied and unoccupied locations are retained in the representation. This representation allows the search process to exploit grid-level layout information more directly. At the same time, the turbine number can still vary through the number of activated components. Therefore, the proposed representation provides a more structure-preserving fixed-dimensional formulation for variable-turbine-number WFLO. The overall workflow of the standard binary occupancy GA is illustrated in Figure 5.

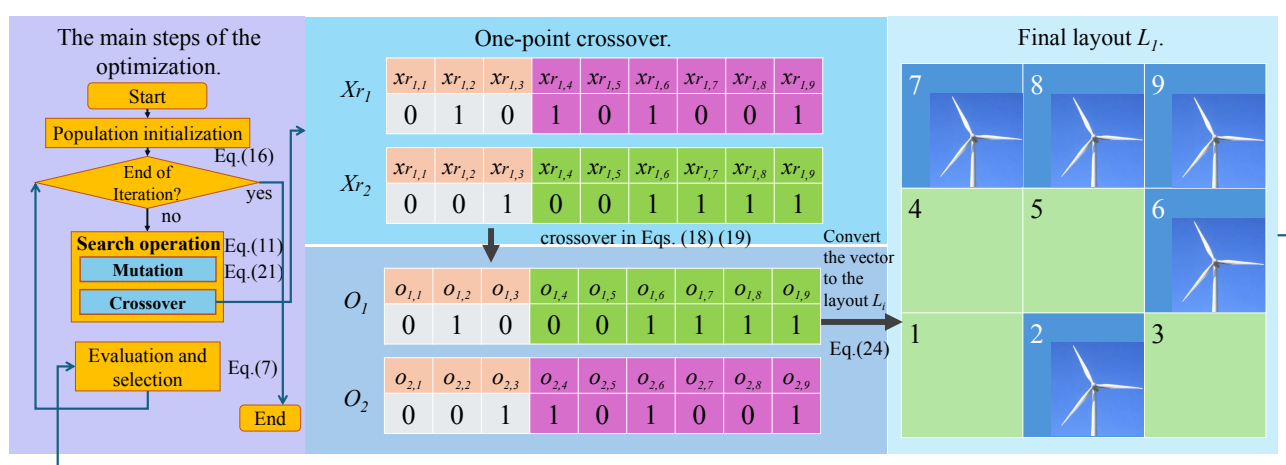


Figure 5. Standard binary occupancy GA.

4.3.2. Structure-Based Binary Occupancy GA

To further exploit the spatial structure of the wind farm grid, this study further designs a structure-based binary occupancy vector representation. Similar to the binary occupancy vector representation in Section 4.3.1, each individual is still encoded as a fixed-length binary vector in Equation (16). The turbine number is determined by the number of activated positions in Equation (17).

The main difference from the standard binary occupancy vector representation lies in the design of the crossover operator. In the standard approach, crossover is performed by one-point recombination on the one-dimensional chromosome. However, wind farm layouts are inherently two-dimensional spatial structures, where adjacent grid cells usually exhibit stronger spatial correlations due to wake interactions and local layout patterns. Therefore, direct crossover on the one-dimensional sequence may destroy meaningful local structural information.

To preserve local structural characteristics in the two-dimensional layout, the proposed structure crossover directly performs region-exchange operations on the two-dimensional grid structure. Let the wind farm grid consist of R rows and C columns, such that

$$d = R \times C. \quad (25)$$

Each binary vector is first reshaped into a two-dimensional layout matrix:

$$B_i = \text{reshape}(X_i, R, C), \quad B_i \in \{0, 1\}^{RC}. \quad (26)$$

For two parent individuals X_{r1} and X_{r2} , their corresponding layout matrices are denoted as B_{r1} and B_{r2} , respectively. A rectangular exchange region is then randomly generated. Specifically, the height h and width w of the rectangle are sampled as

$$h \sim U\{\lceil \alpha_r R \rceil, \dots, R\}, \quad w \sim U\{\lceil \alpha_c C \rceil, \dots, C\}, \quad (27)$$

where α_r and α_c are parameters controlling the minimum exchange-region size. In this study, $\alpha_r = \alpha_c$. The top-left corner of the rectangular region is then randomly sampled as

$$r_s \sim U\{1, \dots, R - h + 1\}, \quad c_s \sim U\{1, \dots, C - w + 1\}. \quad (28)$$

The corresponding bottom-right corner is given by

$$r_e = r_s + h - 1, \quad c_e = c_s + w - 1. \quad (29)$$

Based on this rectangular region, two offspring are generated by exchanging the corresponding layout structures between the two parents:

$$C_k^{(1)}(p, q) = \begin{cases} B_{r2}(p, q), & r_s \leq p \leq r_e, c_s \leq q \leq c_e, \\ B_{r1}(p, q), & \text{otherwise,} \end{cases} \quad (30)$$

$$C_k^{(2)}(p, q) = \begin{cases} B_{r1}(p, q), & r_s \leq p \leq r_e, c_s \leq q \leq c_e, \\ B_{r2}(p, q), & \text{otherwise.} \end{cases} \quad (31)$$

The two-dimensional offspring layouts are subsequently flattened back into one-dimensional binary vectors:

$$O_k^{(1)} = \text{reshape}(C_k^{(1)}, 1, d), \quad O_k^{(2)} = \text{reshape}(C_k^{(2)}, 1, d). \quad (32)$$

The overall offspring generation process still follows the standard GA framework in Equation (11). The mutation operator, repair operator $\Pi(\cdot)$, and the final wind farm layout are identical to those used in Section 4.3.1. Compared with the standard binary occupancy GA, the proposed structure-based representation preserves more two-dimensional spatial layout information during crossover. By exchanging contiguous rectangular regions rather than simple one-dimensional segments, the proposed method can inherit local turbine-layout structures from parent individuals more effectively. Meanwhile, because the number of activated cells within the exchanged region may differ between parents, the turbine number can still adapt dynamically during evolution. Therefore, the proposed method maintains compatibility with the fixed-length GA framework while introducing a more spatially aware recombination mechanism for variable-turbine-number WFLO problems. The overall workflow of the structure-based binary occupancy GA is illustrated in Figure 6.

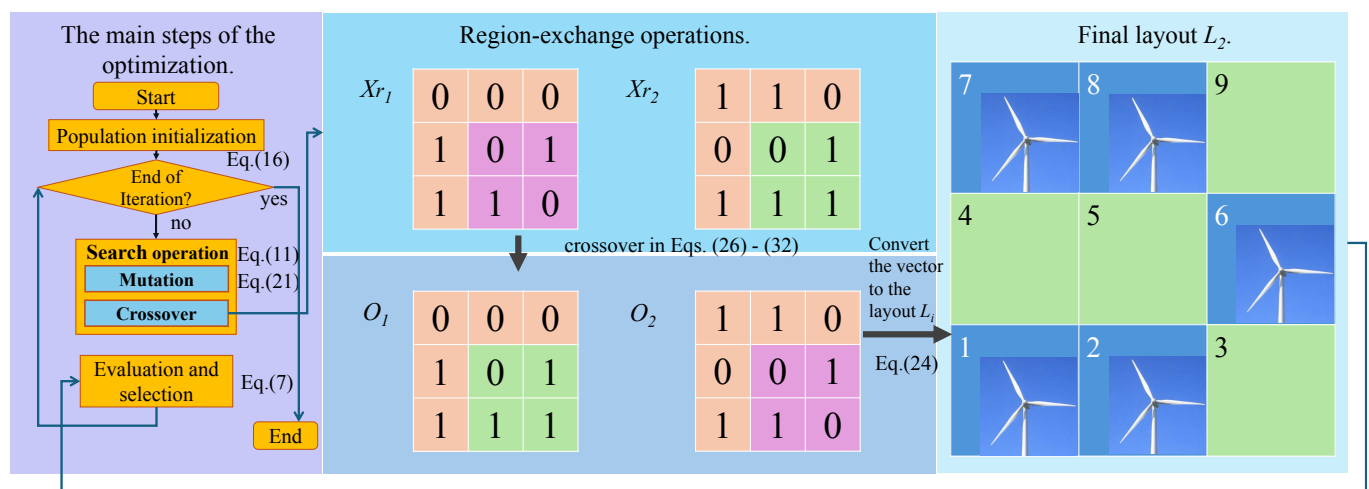


Figure 6. Structure-based binary occupancy GA.

4.4. Binary Occupancy Simulated Annealing

In addition to GA-based binary occupancy vector representations, this study further introduces a SA-based search strategy for the variable-turbine-number WFLO problem. Similar to Section 4.3.1, each solution is encoded as a fixed-length binary occupancy vector in Equation (16). Different from population-based evolutionary algorithms, SA maintains only one current solution during the search process. The algorithm starts from a randomly generated feasible binary layout. Specifically, an initial turbine number m is randomly sampled from the feasible range, and m feasible grid cells are randomly activated. A repair operator is then applied to ensure feasibility.

Let X^t denote the current solution at iteration t , where X is represented as defined in Equation (16). A neighboring solution Y^t is generated by one of the following three neighborhood operations:

$$Y^t = \begin{cases} A(X^t), & r < 0.33, \\ R(X^t), & 0.33 \leq r < 0.66, \\ M(X^t), & r \geq 0.66, \end{cases} \quad r \sim U(0,1), \quad (33)$$

where $A(\cdot)$ denotes the add operation, $R(\cdot)$ denotes the remove operation, and $M(\cdot)$ denotes the move operation. The add operation activates one randomly selected feasible empty grid cell:

$$A(X) = X', \quad x'_p = 1, \quad p \sim U(\{j \mid x_j = 0, j \notin \Omega_{NA}\}). \quad (34)$$

The remove operation deactivates one randomly selected existing turbine, while ensuring that at least one turbine remains in the layout:

$$R(X) = X', \quad x'_p = 0, \quad p \sim \mathcal{U}(\{j \mid x_j = 1\}). \quad (35)$$

The move operation removes one existing turbine and activates one feasible empty grid cell:

$$M(X) = X', \quad x'_{p_1} = 0, \quad x'_{p_0} = 1, \quad (36)$$

where

$$p_1 \sim \mathcal{U}(\{j \mid x_j = 1\}), \quad p_0 \sim \mathcal{U}(\{j \mid x_j = 0, j \notin \Omega_{NA}\}). \quad (37)$$

After each neighborhood operation, a repair operator $\Pi(\cdot)$ is applied:

$$\hat{Y}^t = \Pi(Y^t). \quad (38)$$

The repair operator deactivates all infeasible cells. If no turbine remains after repair, one feasible grid cell is randomly activated.

The candidate solution is accepted according to the Metropolis criterion [46]. Let $f(\cdot)$ denote the fitness function and

$$\Delta f = f(\hat{Y}^t) - f(X^t). \quad (39)$$

For a maximization problem, the transition rule is defined as

$$X^{t+1} = \begin{cases} \hat{Y}^t, & \Delta f > 0, \\ \hat{Y}^t, & \Delta f \leq 0 \text{ and } r < \exp(\Delta f / T^{(q)}), \quad r \sim U(0, 1), \\ X^t, & \text{otherwise,} \end{cases} \quad (40)$$

where $T^{(q)}$ denotes the temperature at the q -th temperature level. During each temperature level, $T^{(q)}$ is kept unchanged while L consecutive neighborhood transitions are performed. After that, the temperature is updated as

$$T^{(q+1)} = \sigma T^{(q)}, \quad (41)$$

where

$$\sigma = \left(\frac{T_{\text{end}}}{T_0} \right)^{1/T_{\text{max}}}. \quad (42)$$

Here, T_0 is the initial temperature, T_{end} is the final temperature, T_{max} denotes the number of temperature levels, and L denotes the number of neighborhood transitions at each temperature level. Since SA maintains only one current solution, L is not a population size. The total number of function evaluations is approximately $T_{\text{max}}L$. The final wind farm layout is obtained by Equation (24).

Compared with the GA-based binary occupancy vector representation, the SA-based method does not rely on crossover or population-level selection. Instead, it explores the variable-turbine-number search space through three local neighborhood operations: adding, removing, and moving turbines. The add and remove operations enable direct transitions between turbine-number subspaces, while the move operation refines the spatial layout within a given turbine-number region. Therefore, this method provides a simple but effective local-search-based mechanism for handling variable-turbine-number WFLO problems under the binary occupancy representation. The overall workflow of the binary occupancy SA is illustrated in Figure 7.

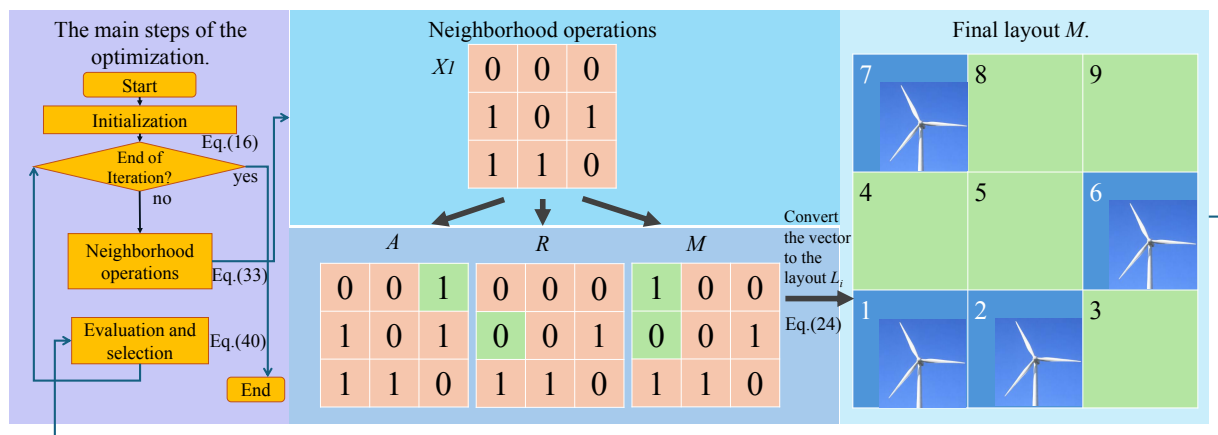


Figure 7. Binary occupancy SA.

5. Experiments and Results

To comprehensively evaluate the effectiveness of different variable-length encoding strategies in WFLO, a series of comparative experiments were conducted. The goal is to assess both optimization performance and structural adaptability when the turbine number is treated as an implicit or explicit decision variable.

In this study, three representative encoding paradigms are considered, corresponding to the main categories of variable-length optimization:

- T- (Variable-length evolutionary representations): This approach directly encodes the turbine number into the solution structure, allowing individuals to have variable lengths.
- C- (Representation mapping approaches): This strategy maintains a fixed-length encoding and maps it to variable-length solutions through decoding.
- D- (Dimension activation/deactivation methods): This approach embeds the problem in a fixed-dimensional space while dynamically adjusting the effective turbine number via activation or deactivation of dimensions.

These encoding strategies are integrated into different metaheuristic frameworks, resulting in the following algorithm variants: T-GA, C-GA, D-GA, C-SHADE, D-SHADE, C-SE, and D-SE.

In addition to the previously defined encoding strategies, three structure-preserving binary occupancy approaches are further considered in this study:

- B- (Standard Binary Occupancy representation): This approach directly represents turbine existence using a binary occupancy vector, where each grid cell corresponds to a binary state indicating turbine presence or absence.
- S- (Structure-based Binary Occupancy representation): This strategy extends the binary occupancy representation by incorporating structure-preserving region-exchange crossover, enabling the inheritance of local spatial layout patterns.
- A- (Binary Occupancy Simulated Annealing representation): This approach employs binary occupancy encoding combined with neighborhood-based local search using simulated annealing.

These strategies are integrated into different optimization frameworks, resulting in the following algorithm variants: B-GA, S-GA, and A-SA.

5.1. Experimental Setup

The parameter settings for all algorithms are summarized in Table 4. Although the population size n differs among algorithms, the total number of fitness evaluations is fixed

at 20,000 to ensure equal computational resources. For example, if SE uses a population size of $n = 20$, it performs $20,000/20 = 1000$ iterations, whereas SHADE, with $n = 100$, executes $20,000/100 = 200$ iterations. Each algorithm is independently executed 51 times to obtain statistically reliable results. It should be noted that for the wind scenario P5, due to the significantly higher computational cost, each algorithm is executed 20 times instead. All experiments were implemented in MATLAB 2023b and executed on a PC equipped with an Intel(R) Core(TM) i9-13900 CPU (2.00 GHz) and 32 GB RAM.

Table 4. Parameter settings of algorithms.

Algorithms	Parameters
GA [26]	$n = 100$
SHADE [45]	$n = 100$
SE [28]	$n = 20, SF \in (0, 1], \eta = (d < 10) \cdot 3 + (d \geq 10) \cdot \mathcal{U}\{6, 10\}$
SA [33]	$T_0 = 1.0, T_{\text{end}} = 10^{-4}, L = 100, T_{\text{max}} = 200$

5.2. Benchmark Configuration

The evaluation function (described in the Appendix A) measures layout quality by considering power output and wake effects. To evaluate robustness under different environmental conditions, five representative wind scenarios (Wind types P1–5) were considered. These scenarios differ in wind distribution characteristics and wake interaction complexity, providing a diverse testbed for assessing algorithm performance. For each scenario, multiple test instances (F1–F13) were constructed to capture variability in layout constraints and search landscapes.

5.3. Performance Evaluation Criteria

To assess algorithmic performance, both statistical tests and visualization tools were used:

1. Wilcoxon Rank-Sum Test [47] ($\alpha = 0.05$): This non-parametric test was used for pairwise comparisons between the baseline algorithm and its competitors. Unless otherwise specified, each algorithm was independently executed 51 times, while 20 runs were used for the more computationally expensive P5 scenario. The results are reported using the $W/T/L$ notation, where W , T , and L denote the numbers of cases in which the baseline algorithm performs significantly better, statistically similarly, and significantly worse, respectively. In the tables, “+”, “=”, and “−” indicate superiority, parity, and inferiority relative to the baseline.
2. Friedman Test [48] ($\alpha = 0.05$): This non-parametric test was used to compare multiple algorithms based on their ranks, where a lower rank indicates better overall performance. Since the Friedman tests indicate significant differences among the compared algorithms, the analysis focuses on the ranking results, and detailed statistics are omitted for brevity.
3. Wilcoxon Signed-Rank Test [49] ($\alpha = 0.05$): This paired non-parametric test was used when comparing related samples, without assuming normality of the performance differences.
4. Convergence Curves: These curves show the optimization process, with the number of function evaluations on the x -axis and the objective value on the y -axis.
5. Box-and-Whisker Plots: These plots summarize the distribution of repeated runs using the interquartile range, median, extrema, and outliers. A wider distribution generally indicates lower stability.

5.4. Analysis of Results

5.4.1. Analysis of Variable-Dimensional Representation Strategies

Table 5 compares the performance of three variable-dimensional representation strategies under different metaheuristic frameworks, while Figure 8 further illustrates their final fitness distributions, convergence behavior, and turbine-number evolution.

Table 5. Performance comparison of variable-dimensional representation strategies under different metaheuristic search frameworks.

T-GA			C-GA			D-GA			C-SHADE			D-SHADE			C-SE			D-SE		
P1	mean	std	mean	std		mean	std		mean	std		mean	std		mean	std		mean	std	
F1	725.99	14.00	714.17	26.68	+	712.10	7.38	+	745.25	21.26	—	684.44	3.12	+	725.65	24.74	=	704.08	5.71	+
F2	702.89	14.02	677.16	10.57	+	691.57	7.52	+	681.83	6.51	+	678.88	1.78	+	680.30	6.08	+	687.17	4.51	+
F3	729.87	11.02	715.47	25.38	+	718.26	6.79	+	750.40	12.94	—	685.04	4.51	+	738.36	23.60	—	701.72	4.81	+
F4	681.71	13.10	662.09	8.43	+	683.11	7.05	=	668.84	8.31	+	664.04	2.25	+	665.69	6.11	+	672.25	3.81	+
F5	685.16	9.80	666.55	15.79	+	681.58	7.59	+	670.27	9.78	+	664.30	1.78	+	666.13	8.95	+	672.04	3.30	+
F6	728.84	14.89	726.09	26.29	=	714.67	7.48	+	750.30	11.48	—	681.71	3.55	+	733.47	25.99	—	699.14	5.51	+
F7	687.29	12.00	671.14	7.25	+	695.96	6.54	—	679.20	1.13	+	677.95	3.38	+	679.28	1.22	+	689.80	3.35	=
F8	714.20	14.56	689.57	19.77	+	701.30	8.15	+	718.76	20.35	—	680.38	1.53	+	691.97	20.57	+	695.00	4.79	+
F9	729.36	12.08	713.89	26.70	+	715.94	8.65	+	748.37	17.43	—	684.51	3.50	+	729.01	24.02	=	703.75	5.64	+
F10	703.49	13.46	680.80	20.85	+	680.94	7.68	+	711.68	20.44	—	672.78	1.56	+	682.29	17.58	+	688.00	4.82	+
F11	705.83	15.17	687.30	24.36	+	697.22	6.39	+	705.06	21.13	=	672.91	1.56	+	683.54	17.09	+	687.58	4.56	+
F12	726.46	15.03	722.57	25.36	=	714.88	8.01	+	747.28	18.75	—	682.98	3.18	+	738.15	19.65	—	702.26	5.58	+
F13	696.53	13.84	681.94	12.44	+	702.85	8.57	—	690.81	11.21	+	680.24	1.48	+	682.83	8.61	+	695.78	4.29	=
W/T/L	-/-/-		11/2/0			10/1/2			5/1/7			13/0/0			8/2/3			11/2/0		
p-value	-		0.00			0.01			0.81			0.00			0.01			0.00		
Rank	2.08		5.46			3.00			2.31			6.77			4.08			4.31		
P2	mean	std	mean	std		mean	std		mean	std		mean	std		mean	std		mean	std	
F1	657.82	10.03	655.09	5.36	=	679.64	3.24	—	665.25	3.19	—	666.25	2.83	—	662.63	5.25	—	679.66	1.84	—
F2	648.88	6.61	648.98	3.48	=	667.29	2.57	—	655.80	3.12	—	659.26	3.49	—	651.67	3.49	—	669.12	1.55	—
F3	652.00	6.24	652.74	4.79	=	668.41	2.93	—	657.39	2.40	—	655.24	3.03	—	654.59	5.00	—	668.73	1.36	—
F4	648.75	6.48	650.17	3.91	—	667.18	2.94	—	656.94	2.77	—	655.97	3.66	—	654.30	3.31	—	666.81	2.14	—
F5	652.31	8.39	650.16	4.65	=	668.40	2.87	—	656.01	2.85	—	655.83	2.83	—	653.62	4.38	—	668.06	2.01	—
F6	653.53	8.39	653.05	5.12	=	674.90	3.61	—	660.47	2.60	—	660.12	3.93	—	657.60	4.47	—	674.20	2.91	—
F7	655.34	8.22	653.34	5.34	=	677.14	3.37	—	661.29	3.57	—	660.93	2.93	—	657.39	3.85	—	676.62	2.41	—
F8	653.38	7.61	652.12	5.28	=	673.51	3.25	—	659.05	2.92	—	661.75	2.90	—	655.34	4.64	—	674.60	1.66	—
F9	652.69	8.90	653.32	4.85	=	674.86	3.09	—	660.82	2.97	—	660.87	2.92	—	658.19	4.54	—	675.25	1.77	—
F10	653.74	7.93	653.38	5.98	=	673.14	3.43	—	660.59	3.34	—	661.16	2.92	—	658.32	3.76	—	674.10	2.15	—
F11	656.28	9.10	651.94	4.72	+	675.31	2.85	—	660.03	2.81	—	661.11	2.48	—	657.38	3.44	—	675.01	2.12	—
F12	653.40	8.22	653.72	5.32	=	677.60	3.40	—	662.52	3.45	—	664.14	3.93	—	660.54	4.71	—	678.30	2.28	—
F13	657.52	9.90	655.16	5.86	=	677.95	2.68	—	663.51	3.75	—	664.56	3.32	—	660.59	3.67	—	678.70	1.70	—
W/T/L	-/-/-		1/11/1			0/0/13			0/0/13			0/0/13			0/0/13			0/0/13		
p-value	-		0.05			1.00			1.00			1.00			1.00			1.00		
Rank	6.38		6.62			1.62			3.62			3.38			5.00			1.38		
P3	mean	std	mean	std		mean	std		mean	std		mean	std		mean	std		mean	std	
F1	800.27	7.65	789.81	12.15	+	757.30	8.40	+	803.39	2.63	—	723.80	4.88	+	801.96	2.39	=	745.24	5.57	+
F2	779.86	7.92	753.52	13.99	+	739.66	8.58	+	767.51	5.96	+	703.66	4.59	+	768.32	2.31	+	727.40	5.03	+
F3	798.54	6.63	781.06	15.55	+	759.76	7.76	+	798.87	4.35	=	723.61	5.04	+	797.55	3.52	=	742.17	5.26	+
F4	770.88	6.85	756.96	12.23	+	736.26	7.59	+	762.96	3.20	+	698.18	5.69	+	760.51	3.19	+	718.23	5.13	+
F5	772.06	5.30	757.90	9.58	+	743.39	6.81	+	763.56	4.05	+	707.29	4.07	+	762.08	3.34	+	728.86	3.70	+
F6	794.92	5.86	785.79	12.27	+	757.72	6.63	+	794.20	11.42	=	721.49	6.09	+	796.63	2.35	—	741.13	3.92	+
F7	767.19	6.17	744.78	11.24	+	736.09	6.75	+	744.42	4.79	+	697.74	3.99	+	745.01	3.83	+	721.23	4.25	+
F8	790.89	7.24	771.46	14.79	+	749.12	8.66	+	785.20	1.95	+	714.28	5.37	+	782.67	2.06	+	737.37	5.80	+
F9	799.46	7.44	782.45	15.36	+	759.39	7.17	+	801.75	2.83	=	722.19	4.50	+	800.49	2.78	=	743.13	4.66	+
F10	784.09	9.07	775.73	11.47	+	747.18	7.31	+	784.67	3.67	=	710.23	4.31	+	782.85	3.47	=	732.65	4.71	+
F11	784.51	6.64	771.78	12.75	+	752.76	8.19	+	782.40	3.15	+	714.49	5.84	+	781.80	3.37	+	736.46	5.05	+
F12	797.33	6.79	786.47	14.00	+	759.44	7.44	+	800.88	4.92	—	721.09	4.93	+	800.52	2.79	—	745.73	5.45	+
F13	764.84	5.91	752.58	12.63	+	743.14	6.20	+	769.20	0.66	—	707.33	4.40	+	766.06	3.29	=	729.51	4.36	+
W/T/L	-/-/-		13/0/0			13/0/0			6/4/3			13/0/0			6/5/2			13/0/0		
p-value	-		0.00			0.00			0.14			0.00			0.07			0.00		
Rank	1.85		3.92			5.00			1.85			7.00			2.38			6.00		

Table 5. Cont.

T-GA			C-GA			D-GA			C-SHADE			D-SHADE			C-SE			D-SE		
P4	mean	std	mean	std		mean	std		mean	std		mean	std		mean	std		mean	std	
F1	599.42	4.33	570.24	6.32	+	587.12	4.66	+	570.49	2.81	+	565.16	2.91	+	572.95	3.27	+	577.55	3.68	+
F2	580.72	3.77	553.30	4.15	+	569.63	4.53	+	549.96	3.60	+	548.87	3.15	+	551.87	3.24	+	560.92	2.81	+
F3	598.04	4.82	569.26	4.73	+	584.60	4.62	+	568.08	2.65	+	562.18	3.23	+	570.08	2.11	+	573.47	3.26	+
F4	582.59	4.64	556.92	5.56	+	568.99	5.21	+	551.97	3.02	+	545.70	3.37	+	554.67	3.07	+	558.40	3.81	+
F5	601.48	4.29	570.68	5.13	+	584.72	4.72	+	567.49	3.21	+	561.38	3.06	+	569.67	3.34	+	572.62	3.07	+
F6	601.02	4.52	575.02	4.17	+	588.41	5.39	+	570.40	2.55	+	564.82	3.40	+	573.52	2.70	+	575.69	2.79	+
F7	585.37	4.59	552.74	4.45	+	571.71	4.50	+	549.68	2.63	+	546.80	3.72	+	550.83	2.83	+	561.15	3.40	+
F8	591.56	3.73	561.84	4.10	+	578.84	5.26	+	561.11	2.48	+	558.18	3.16	+	563.04	2.69	+	569.89	3.50	+
F9	597.59	4.56	570.34	5.35	+	585.20	4.38	+	570.20	2.43	+	563.08	3.21	+	572.12	2.65	+	574.12	3.19	+
F10	591.64	3.91	564.42	5.58	+	578.55	4.49	+	562.82	3.18	+	556.16	3.34	+	564.12	2.56	+	567.40	3.03	+
F11	599.36	4.06	569.98	5.58	+	585.41	3.97	+	568.91	2.96	+	562.46	3.11	+	570.37	2.88	+	575.39	3.06	+
F12	598.23	4.01	572.83	4.07	+	587.47	5.00	+	570.87	2.62	+	565.41	3.65	+	573.18	2.39	+	576.40	2.79	+
F13	594.80	4.43	560.30	5.87	+	581.94	5.40	+	559.16	2.95	+	556.39	3.40	+	561.61	3.52	+	570.03	2.79	+
W/T/L	-/-/-		13/0/0			13/0/0			13/0/0			13/0/0			13/0/0			13/0/0		
p-value	-		0.00			0.00			0.00			0.00			0.00			0.00		
Rank	1.00		4.62			2.00			5.92			7.00			4.46			3.00		
P5	mean	std	mean	std		mean	std		mean	std		mean	std		mean	std		mean	std	
F1	235.65	1.36	231.13	3.36	+	226.72	1.64	+	233.87	0.86	+	219.05	0.97	+	233.56	0.76	+	224.04	1.13	+
F2	231.02	1.14	222.44	2.33	+	221.60	1.19	+	220.65	2.53	+	213.92	1.06	+	222.16	1.18	+	219.20	0.74	+
F3	235.51	1.30	231.02	3.66	+	228.05	1.52	+	230.98	1.27	+	218.82	1.19	+	232.59	1.00	+	223.36	0.90	+
F4	229.19	1.33	224.37	2.64	+	221.30	2.45	+	224.14	1.16	+	211.18	1.08	+	223.97	0.83	+	217.01	1.38	+
F5	232.51	1.20	226.85	2.64	+	225.49	1.81	+	226.40	1.69	+	217.34	0.94	+	226.59	0.86	+	222.23	0.97	+
F6	236.35	1.20	232.60	2.18	+	227.56	1.84	+	231.24	2.03	+	218.95	0.93	+	232.68	0.81	+	223.74	1.17	+
F7	229.25	1.00	220.17	4.15	+	220.61	1.65	+	219.89	2.49	+	211.09	0.96	+	222.19	1.72	+	217.33	1.16	+
F8	233.74	1.58	225.17	3.38	+	225.14	1.52	+	228.35	1.02	+	216.28	1.03	+	228.30	0.73	+	221.96	1.35	+
F9	235.60	1.36	229.85	4.29	+	227.75	1.59	+	232.25	2.66	+	218.68	1.44	+	232.73	0.95	+	223.94	1.06	+
F10	232.57	1.91	227.53	3.92	+	223.87	1.80	+	228.90	1.09	+	215.01	1.29	+	229.13	0.86	+	220.56	1.23	+
F11	233.70	1.06	227.82	3.65	+	226.57	1.80	+	229.89	1.73	+	218.08	1.20	+	230.20	1.04	+	223.79	1.09	+
F12	235.70	1.34	229.21	4.96	+	227.96	1.73	+	232.18	1.97	+	219.29	1.37	+	233.21	0.77	+	224.03	1.18	+
F13	229.92	1.23	222.39	2.81	+	221.59	1.64	+	222.91	1.53	+	214.19	0.93	+	223.99	1.63	+	219.01	0.98	+
W/T/L	-/-/-		13/0/0			13/0/0			13/0/0			13/0/0			13/0/0			13/0/0		
p-value	-		0.00			0.00			0.00			0.00			0.00			0.00		
Rank	1.00		3.38			4.77			3.38			7.00			2.46			6.00		

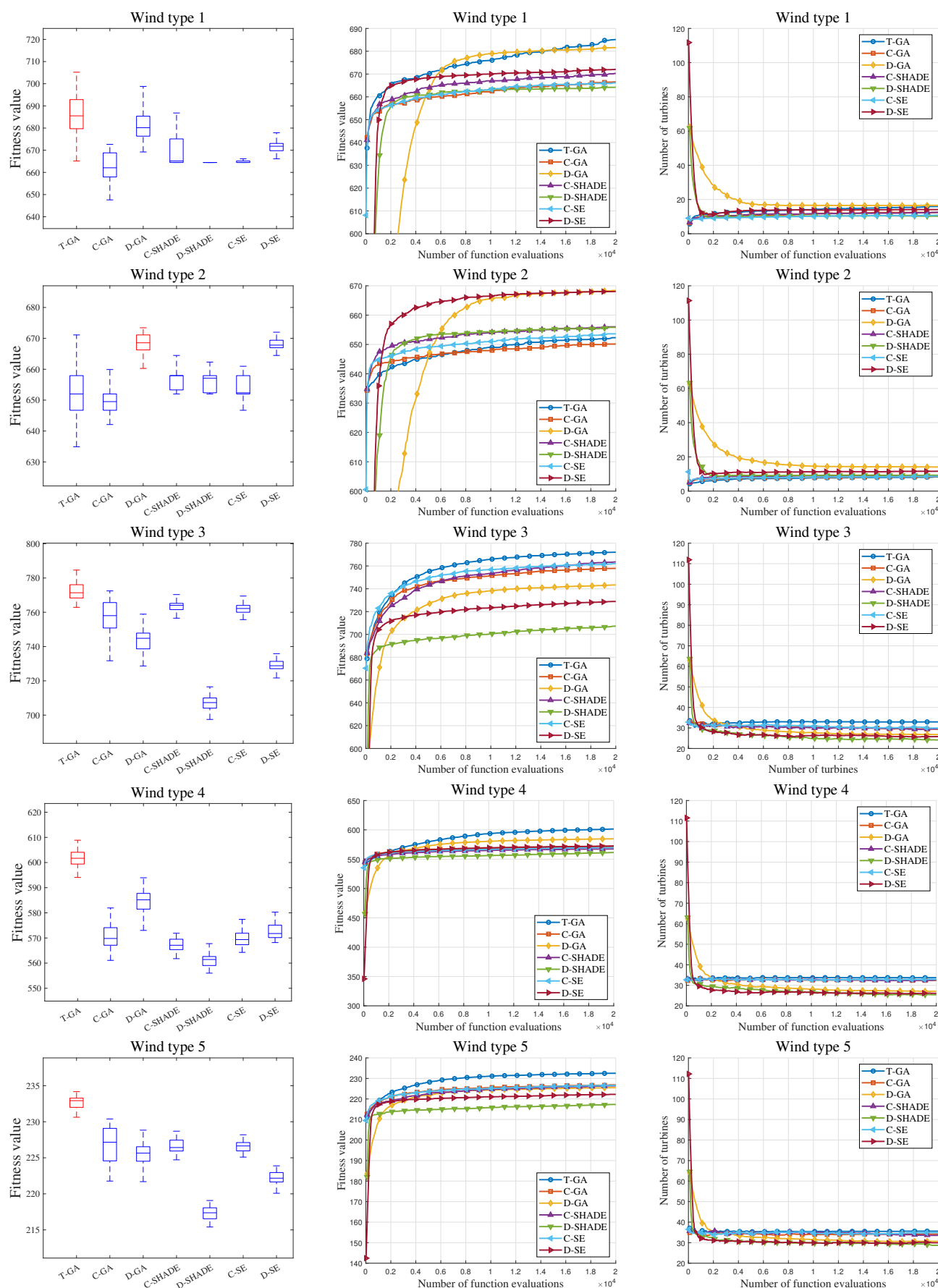


Figure 8. Search dynamics of variable-dimensional representation strategies under different wind scenarios.

In Table 5, T-GA achieves the best or near-best Friedman rankings in most scenarios and shows particularly clear advantages in P4 and P5. This trend is also visible in Figure 8, where T-GA obtains higher final fitness values and continues to improve during the middle and late stages of optimization in complex wind scenarios. Since the chromosome length directly represents the turbine number, T-GA can adjust turbine quantity while preserving partial layout components through crossover. This makes it more suitable for heterogeneous search spaces where turbine number, wake effects, and spatial arrangement are strongly coupled.

In contrast, D-based methods generally obtain weaker rankings in Table 5. Although dimension activation/deactivation enables variable turbine numbers within a fixed-dimensional space, the final layout is produced through discretization, duplicate removal, and binary activation. These indirect transformations may weaken the correspondence between the encoded vector and the decoded layout. As a result, small changes in the encoded space can lead to discontinuous layout changes, making it difficult to preserve meaningful spatial structures during search.

C-based methods are generally more competitive than D-based methods, suggesting that representation mapping maintains a more consistent relationship between fixed-length encoding and variable-length layouts. This indicates that even within fixed-dimensional search frameworks, structural consistency between representation and layout remains important.

Figure 8 also shows that the advantage of T-GA is scenario-dependent. In P1 and especially P2, several fixed-length methods converge rapidly and remain competitive, which is consistent with their better rankings in Table 5. This suggests that, when wind conditions are relatively regular, fixed-dimensional representations may already capture the dominant layout structure, and additional variable-length flexibility provides limited benefit.

The turbine-number curves in Figure 8 provide further evidence that turbine quantity is adaptively adjusted rather than fixed or monotonically changed. Different algorithms converge to different turbine-number ranges under the same land constraint, reflecting their different abilities to coordinate turbine-number adaptation with spatial layout optimization. In complex scenarios, this joint adaptation becomes more important, which explains the stronger performance of T-GA from P3 to P5.

To further quantify the practical significance of the observed differences, effect sizes and their 95% confidence intervals in Table 6 were analyzed using T-GA as the baseline. A positive effect size indicates an advantage of T-GA, whereas a negative value indicates that the compared algorithm performs better. In P1, T-GA shows strong positive effects over C-GA, D-GA, D-SHADE, C-SE, and D-SE, with effect sizes ranging from 0.692 to 1.000. The corresponding confidence intervals are mostly above zero, indicating that these advantages are statistically stable. The only exception is C-SHADE, for which the effect size is negative and the 95% confidence interval crosses zero, suggesting that no reliable superiority of T-GA can be established in this comparison. In P2, the results reveal a different pattern: T-GA does not show a robust advantage over C-GA, as the confidence interval includes zero, and it is consistently inferior to D-GA, C-SHADE, D-SHADE, C-SE, and D-SE, as indicated by effect sizes of -1.000 with confidence intervals entirely below zero. In P3, T-GA again demonstrates clear advantages over C-GA, D-GA, D-SHADE, and D-SE, while its advantages over C-SHADE and C-SE are not statistically robust because their confidence intervals overlap zero. In P4, T-GA achieves uniformly positive effect sizes of 1.000 against all compared methods, with confidence intervals of $[1.000, 1.000]$, indicating highly consistent dominance across all test instances. A similar pattern is observed in the final scenario, where T-GA also shows complete positive effects over all compared algorithms. Overall, these results suggest that the variable-length evolutionary representation used by T-GA can be highly effective in several wind scenarios, but its

advantage is not universal. In particular, its performance is scenario-dependent, and representation mapping or dimension activation/deactivation strategies may be more competitive under certain wind conditions such as P2.

Table 6. Effect sizes and 95% confidence intervals of pairwise comparisons using T-GA as the baseline

Scenario	Baseline	Compared Algorithm	Effect Size	95% CI_Low	95% CI_High
P1	T-GA	C-GA	1.000	1.000	1.000
		D-GA	0.758	0.319	1.000
		C-SHADE	−0.275	−0.802	0.385
		D-SHADE	1.000	1.000	1.000
		C-SE	0.692	0.176	1.000
		D-SE	0.956	0.780	1.000
P2	T-GA	C-GA	0.516	−0.099	0.912
		D-GA	−1.000	−1.000	−1.000
		C-SHADE	−1.000	−1.000	−1.000
		D-SHADE	−1.000	−1.000	−1.000
		C-SE	−1.000	−1.000	−1.000
		D-SE	−1.000	−1.000	−1.000
P3	T-GA	C-GA	1.000	1.000	1.000
		D-GA	1.000	1.000	1.000
		C-SHADE	0.363	−0.319	0.824
		D-SHADE	1.000	1.000	1.000
		C-SE	0.473	−0.154	0.890
		D-SE	1.000	1.000	1.000
P4	T-GA	C-GA	1.000	1.000	1.000
		D-GA	1.000	1.000	1.000
		C-SHADE	1.000	1.000	1.000
		D-SHADE	1.000	1.000	1.000
		C-SE	1.000	1.000	1.000
		D-SE	1.000	1.000	1.000
P5	T-GA	C-GA	1.000	1.000	1.000
		D-GA	1.000	1.000	1.000
		C-SHADE	1.000	1.000	1.000
		D-SHADE	1.000	1.000	1.000
		C-SE	1.000	1.000	1.000
		D-SE	1.000	1.000	1.000

5.4.2. Analysis of Occupancy-Based Structural Search Strategies

As shown in Table 7, the occupancy-based methods generally achieve better performance than T-GA, indicating that explicitly preserving grid-level occupancy information is beneficial for variable-turbine-number WFLO. Among these methods, B-GA serves as the standard binary occupancy GA, while S-GA1 and S-GA2 are two structure-based binary occupancy GA variants introduced to examine the influence of different region-exchange scales. Specifically, S-GA1 uses a larger rectangular exchange region with $\alpha = 0.7$, whereas S-GA2 uses a smaller exchange region with $\alpha = 0.1$. A-SA further adopts the same binary occupancy representation but replaces population-based crossover with simulated-annealing-based neighborhood search.

Compared with T-GA, B-GA generally obtains better results. This improvement does not come from a larger feasible solution space but from a more stable occupancy-based encoding. In T-GA, only selected turbine positions are stored as a variable-length sequence, whose order is not necessarily consistent with spatial adjacency. Therefore, one-dimensional crossover and resampling-based mutation may easily disturb useful layout components. In contrast, B-GA assigns each grid cell a fixed binary occupancy state, reducing encoding redundancy and allowing bit-flip mutation to introduce more localized changes.

Table 7. Performance comparison of occupancy-based structural search strategies under different optimization frameworks.

	A-SA		S-GA1			S-GA2			B-GA			T-GA		
	mean	std	mean	std		mean	std		mean	std		mean	std	
F1	754.31	0.67	746.96	6.63	+	737.02	6.49	+	744.22	7.02	+	725.99	14.00	+
F2	736.00	0.95	720.21	11.56	+	714.80	11.32	+	717.82	12.54	+	702.89	14.02	+
F3	754.75	0.51	749.04	4.55	+	737.01	6.78	+	741.97	7.28	+	729.87	11.02	+
F4	715.16	0.51	707.72	6.86	+	697.42	9.27	+	700.44	9.10	+	681.71	13.10	+
F5	715.19	0.49	703.64	8.87	+	690.18	12.24	+	699.54	8.63	+	685.16	9.80	+
F6	754.48	0.71	748.38	4.48	+	739.28	5.05	+	743.71	6.95	+	728.84	14.89	+
F7	723.27	0.61	716.80	7.66	+	709.03	7.03	+	712.69	5.01	+	687.29	12.00	+
F8	746.00	0.80	736.73	8.11	+	727.19	6.50	+	731.59	8.40	+	714.20	14.56	+
F9	754.38	0.87	746.59	7.40	+	737.12	7.08	+	741.35	7.16	+	729.36	12.08	+
F10	734.92	0.55	728.85	5.49	+	717.93	5.73	+	722.79	6.96	+	703.49	13.46	+
F11	734.74	0.76	726.77	5.58	+	718.21	6.06	+	719.43	7.06	+	705.83	15.17	+
F12	754.38	0.69	747.44	5.62	+	737.17	5.99	+	744.65	7.24	+	726.46	15.03	+
F13	743.46	1.08	730.56	5.63	+	723.53	5.64	+	726.77	6.15	+	696.53	13.84	+
W/T/L	-/-/-		13/0/0			13/0/0			13/0/0			13/0/0		
p-value	-		0.00			0.00			0.00			0.00		
Rank	1.00		2.00			4.00			3.00			5.00		
	mean	std	mean	std		mean	std		mean	std		mean	std	
F1	691.08	3.38	684.40	4.11	+	685.53	4.77	+	686.32	4.65	+	657.82	10.03	+
F2	672.21	0.45	669.59	1.92	+	669.98	1.78	+	669.40	1.84	+	648.88	6.61	+
F3	675.35	2.33	673.00	2.72	+	673.92	3.11	+	673.06	3.33	+	652.00	6.24	+
F4	672.05	0.13	669.52	2.20	+	670.07	2.10	+	670.09	1.91	+	648.75	6.48	+
F5	675.95	2.16	672.71	3.06	+	673.29	2.86	+	672.82	3.34	+	652.31	8.39	+
F6	681.74	2.07	679.00	2.35	+	678.98	3.01	+	677.63	2.81	+	653.53	8.39	+
F7	683.08	1.37	680.14	1.48	+	680.85	2.00	+	680.64	2.09	+	655.34	8.22	+
F8	678.83	2.42	676.13	3.27	+	676.74	2.94	+	676.25	2.08	+	653.38	7.61	+
F9	684.92	3.23	679.38	4.00	+	680.32	3.55	+	679.51	4.23	+	652.69	8.90	+
F10	678.88	2.80	676.20	2.99	+	676.71	2.30	+	676.27	2.68	+	653.74	7.93	+
F11	683.49	4.09	678.48	3.90	+	678.97	3.99	+	679.98	3.94	+	656.28	9.10	+
F12	686.18	2.93	681.61	3.14	+	682.17	3.11	+	681.98	3.06	+	653.40	8.22	+
F13	686.19	3.69	681.07	3.48	+	681.78	2.61	+	681.82	3.49	+	657.52	9.90	+
W/T/L	-/-/-		13/0/0			13/0/0			13/0/0			13/0/0		
p-value	-		0.00			0.00			0.00			0.00		
Rank	1.00		3.77			2.38			2.85			5.00		
	mean	std	mean	std		mean	std		mean	std		mean	std	
F1	813.88	6.66	813.50	5.86	=	805.77	9.23	+	811.29	7.63	+	800.27	7.65	+
F2	787.37	6.89	787.71	6.65	=	786.26	7.24	=	786.26	7.48	=	779.86	7.92	+
F3	809.29	6.31	810.39	4.19	=	803.43	8.13	+	806.58	8.85	=	798.54	6.63	+
F4	787.66	5.75	785.40	4.21	+	780.30	8.37	+	783.31	6.73	+	770.88	6.85	+
F5	782.22	3.12	782.16	2.69	=	777.08	5.79	+	780.07	4.67	+	772.06	5.30	+
F6	799.62	6.93	804.47	4.57	-	797.20	7.87	+	799.29	6.84	=	794.92	5.86	+
F7	779.39	3.69	776.57	3.77	+	773.25	5.81	+	773.51	4.23	+	767.19	6.17	+
F8	802.02	7.15	803.30	5.34	=	796.57	7.83	+	800.53	6.85	=	790.89	7.24	+
F9	812.54	6.59	813.71	4.36	=	806.31	9.41	+	808.30	9.57	+	799.46	7.44	+
F10	803.31	5.19	800.36	5.57	+	793.95	7.58	+	796.73	8.21	+	784.09	9.07	+
F11	795.38	5.19	797.01	4.05	-	792.06	6.77	+	794.36	4.87	=	784.51	6.64	+
F12	809.25	4.84	811.57	3.55	-	803.24	9.92	+	808.01	7.82	=	797.33	6.79	+
F13	779.40	3.46	775.99	2.92	+	772.60	4.93	+	774.22	4.33	+	764.84	5.91	+
W/T/L	-/-/-		4/6/3			12/1/0			7/6/0			13/0/0		
p-value	-		0.53			0.00			0.00			0.00		
Rank	1.54		1.46			4.00			3.00			5.00		
	mean	std	mean	std		mean	std		mean	std		mean	std	
F1	618.88	2.45	610.81	4.31	+	611.13	4.01	+	611.30	4.42	+	599.42	4.33	+
F2	598.82	1.86	590.84	5.32	+	591.35	4.60	+	592.04	4.26	+	580.72	3.77	+
F3	613.63	1.24	608.81	4.01	+	609.85	3.90	+	608.04	4.38	+	598.04	4.82	+
F4	598.93	2.00	588.42	5.74	+	592.55	4.41	+	593.31	3.43	+	582.59	4.64	+
F5	613.99	0.84	609.49	3.85	+	610.66	3.05	+	609.68	3.15	+	601.48	4.29	+
F6	616.63	1.57	609.46	4.39	+	611.72	3.05	+	610.84	3.60	+	601.02	4.52	+
F7	601.83	5.03	594.13	5.27	+	596.31	5.12	+	595.34	5.60	+	585.37	4.59	+
F8	608.95	1.92	601.36	5.50	+	603.32	4.80	+	603.85	3.62	+	591.56	3.73	+
F9	615.30	1.19	608.99	4.88	+	611.21	4.06	+	610.47	3.15	+	597.59	4.56	+
F10	610.14	2.12	601.31	4.58	+	603.95	3.97	+	602.77	4.20	+	591.64	3.91	+
F11	615.33	1.60	608.87	4.28	+	611.44	3.71	+	610.27	3.62	+	599.36	4.06	+
F12	617.82	2.05	610.44	4.26	+	611.26	3.81	+	611.23	3.68	+	598.23	4.01	+
F13	611.43	3.59	606.57	4.58	+	606.32	3.64	+	606.09	4.55	+	594.80	4.43	+
W/T/L	-/-/-		13/0/0			13/0/0			13/0/0			13/0/0		
p-value	-		0.00			0.00			0.00			0.00		
Rank	1.00		3.77			2.38			2.85			5		
	mean	std	mean	std		mean	std		mean	std		mean	std	
F1	238.52	0.96	238.46	0.90	=	237.49	1.32	+	237.52	1.51	+	235.65	1.36	+
F2	232.74	1.07	233.19	0.95	-	231.88	1.27	+	232.52	1.14	=	231.02	1.14	+
F3	237.76	0.95	237.70	0.69	=	236.13	1.26	+	236.67	0.97	+	235.51	1.30	+
F4	231.82	0.98	231.52	0.96	=	230.49	1.41	+	230.60	1.29	+	229.19	1.33	+
F5	234.73	0.47	233.97	0.67	+	233.42	0.97	+	233.50	0.96	+	232.51	1.20	+
F6	238.76	0.98	238.25	1.39	=	237.06	1.73	+	237.41	1.39	+	236.35	1.20	+
F7	232.26	0.60	231.59	0.57	+	230.87	0.83	+	230.87	0.94	+	229.25	1.00	+
F8	236.28	0.78	236.23	1.10	=	234.83	1.04	+	235.14	1.64	+	233.74	1.58	+
F9	238.02	1.14	237.82	1.26	=	236.68	1.27	+	236.95	1.38	+	235.60	1.36	+
F10	235.76	0.65	235.83	0.66	=	234.06	1.26	+	234.73	1.09	+	232.57	1.91	+
F11	236.38	0.68	236.58	0.63	=	235.04	1.13	+	235.73	0.78	+	233.70	1.06	+
F12	238.79	0.73	238.47	0.45	=	237.21	1.34	+	237.15	1.28	+	235.70	1.34	+
F13	233.10	0.54	233.00	0.41	=	232.03	1.13	+	231.82	0.90	+	229.92	1.23	+
W/T/L	-/-/-		2/10/1			13/0/0			12/1/0			13/0/0		
p-value	-		0.03			0.00			0.00			0.00		
Rank	1.23		1.77			3.77			3.23			5.00		

However, B-GA still performs crossover on a one-dimensional binary vector, so it mainly preserves index-contiguous segments rather than true two-dimensional spatial patterns. To address this limitation, S-GA1 and S-GA2 exchange rectangular regions directly on the wind farm grid, enabling offspring to inherit spatially meaningful local layout structures from parent solutions. The difference between these two variants reflects a trade-off between global structural transfer and local refinement: S-GA1 emphasizes large-scale structural inheritance through a larger exchange region, whereas S-GA2 favors more localized recombination through a smaller exchange region.

A-SA exhibits a different search behavior from the GA-based methods. As shown in the turbine-number evolution curves in Figure 9, its turbine number changes more smoothly and gradually. This behavior is mainly related to its single-solution local-search mechanism: instead of recombining multiple layouts through crossover, A-SA updates the current solution through add, remove, and move operations. Although this smooth evolution is partly a property of local search, it also means that consecutive solutions differ only slightly, so existing spatial structures are less frequently disrupted during turbine-number adaptation. This structural continuity may contribute to the stable convergence and robustness of A-SA in complex wind scenarios.

To further evaluate the practical significance of the observed differences, effect sizes and their 95% confidence intervals in Table 8 were analyzed using A-SA as the baseline. A positive effect size indicates an advantage of A-SA, whereas a negative value favors the compared algorithm. In P1, P2, and P4, A-SA consistently achieves the maximum effect size of 1.000 against all competing methods, with confidence intervals of [1.000, 1.000], indicating highly consistent superiority across all benchmark instances. In P3, A-SA maintains complete superiority over S-GA2, B-GA, and T-GA, while its comparison with S-GA1 yields an effect size close to zero (−0.011), and the corresponding 95% confidence interval spans zero, suggesting no statistically reliable performance difference between the two methods. In P5, A-SA continues to outperform all competing algorithms, although the advantage over S-GA1 is relatively weaker, with an effect size of 0.604 and a 95% confidence interval of [0.055, 1.000]. The effect-size analysis further confirms the robustness of the proposed occupancy-based simulated annealing strategy. Except for a few scenario-dependent cases involving S-GA1 and S-GA2, A-SA consistently demonstrates large positive effects over the other occupancy-based and variable-dimensional search strategies.

To further analyze the robustness of A-SA, we performed a post-analysis of the turbine-number search path under the P5 wind condition. Let m^t denote the turbine number of the best solution at iteration t . The average step-wise turbine-number change is defined as

$$\overline{\Delta m} = \frac{1}{T-1} \sum_{t=1}^{T-1} |m^{t+1} - m^t|. \quad (43)$$

A smaller value of $\overline{\Delta m}$ indicates a smoother turbine-number trajectory and thus higher search-path stability.

In addition, the net turbine-number change and the net fitness improvement are defined as

$$\Delta m_{\text{net}} = m^T - m^1, \quad \Delta f_{\text{net}} = f^T - f^1, \quad (44)$$

where m^1 and m^T denote the turbine numbers at the first and last iterations, respectively, and f^1 and f^T denote the corresponding best fitness values.

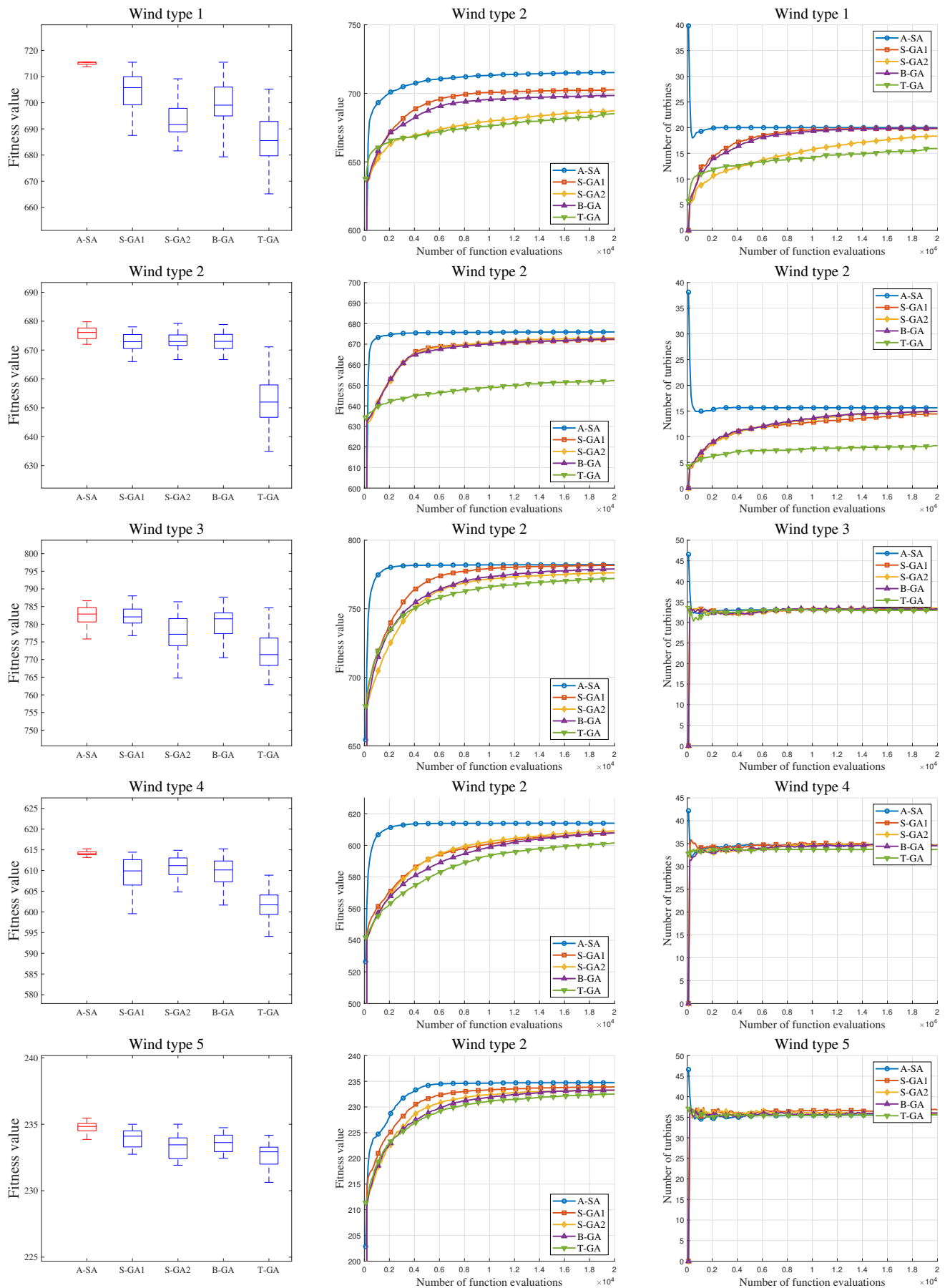


Figure 9. Search dynamics of occupancy-based structural search strategies under different wind scenarios.

Table 8. Effect sizes and 95% confidence intervals of pairwise comparisons using A-SA as the baseline.

Scenario	Baseline	Compared Algorithm	Effect Size	95%CI_Low	95%CI_High
P1	A-SA	S-GA2	1.000	1.000	1.000
		S-GA1	1.000	1.000	1.000
		B-GA	1.000	1.000	1.000
		T-GA	1.000	1.000	1.000
P2	A-SA	S-GA2	1.000	1.000	1.000
		S-GA1	1.000	1.000	1.000
		B-GA	1.000	1.000	1.000
		T-GA	1.000	1.000	1.000
P3	A-SA	S-GA2	1.000	1.000	1.000
		S-GA1	−0.011	−0.659	0.604
		B-GA	1.000	1.000	1.000
		T-GA	1.000	1.000	1.000
P4	A-SA	S-GA2	1.000	1.000	1.000
		S-GA1	1.000	1.000	1.000
		B-GA	1.000	1.000	1.000
		T-GA	1.000	1.000	1.000
P5	A-SA	S-GA2	1.000	1.000	1.000
		S-GA1	0.604	0.055	1.000
		B-GA	1.000	1.000	1.000
		T-GA	1.000	1.000	1.000

As shown in Table 9, the average value of $\overline{\Delta m}$ over the 13 P5 cases is only 0.145, indicating that the turbine number changes smoothly along the search trajectory. Even in F11, which shows the largest turbine-number adjustment, the average step-wise change remains below 0.20. Meanwhile, all representative cases show negative Δm_{net} , suggesting that A-SA tends to gradually remove redundant turbines while improving the objective value. This behavior is consistent with the design of the add, remove, and move neighborhood operations. The add and remove operations change the turbine number locally, while the move operation preserves the turbine number and only relocates one turbine. Therefore, A-SA can move across neighboring turbine-number subspaces without abruptly destroying the existing spatial layout structure.

Table 9. Representative search-path stability results of A-SA under the P5 wind condition.

Case	Role	$\overline{\Delta m}$	$\max \Delta m$	$\sigma_{\Delta m}$	Δm_{net}	Δf_{net}
Mean	Average over F1–F13	0.145	7.52	0.737	−16.40	43.20
F13	Smoothest turbine-number path	0.108	4.05	0.434	−6.20	37.17
F12	Largest fitness improvement	0.192	8.85	0.974	−27.20	51.26
F11	Largest turbine-number adjustment	0.199	12.50	1.141	−27.30	48.42
F5	Lowest fitness improvement	0.114	6.25	0.521	−11.00	31.87

The results also indicate that smoother search paths do not necessarily imply weaker optimization ability. For example, F12 achieves the largest fitness improvement, with $\Delta f_{\text{net}} = 51.26$, while still maintaining a relatively smooth turbine-number trajectory. These observations support the robustness of A-SA and help explain why a simple local-search mechanism can be effective for variable-turbine-number WFLO.

Figure 10 shows the average step-wise turbine-number change of A-SA under the P5 wind condition. For all 13 cases, $\overline{\Delta m}$ remains below 0.20, indicating that the turbine number changes smoothly during the search process. F11 and F12 show relatively larger values, suggesting more active turbine-number adjustment, whereas F5 and F13 exhibit the smoothest trajectories. Overall, the results support the search-path stability of A-SA across different benchmark cases.

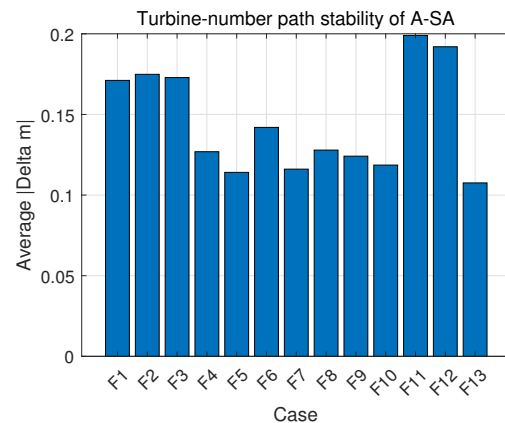


Figure 10. Turbine-number path stability of A-SA under the P5 wind condition. Smaller $\overline{\Delta m}$ indicates a smoother turbine-number trajectory.

Overall, the results indicate that the key issue in variable-turbine-number WFLO is not only how to represent different turbine-number subspaces, but also how to design search operators that are consistent with the spatial structure of wind farm layouts. The improvement from T-GA to B-GA mainly comes from a more canonical occupancy encoding and localized perturbation, while the further advantages of S-GA and A-SA arise from stronger preservation of two-dimensional layout structures and structural continuity during search. These observations reinforce the main conclusion that effective exploitation of spatial layout structures is more critical than merely enabling variable-length representation.

5.5. Parameter Sensitivity and Robustness Analysis

The sensitivity analysis in Table 10 indicates that the initial temperature T_0 affects the absolute optimization performance of A-SA, and that the preferred temperature setting is problem-dependent. A relatively small T_0 is more suitable for P1 and P2, whereas larger initial temperatures generally yield better performance on P3–P5. This observation suggests that stronger exploration during the early search stage is beneficial for the fitness landscapes of P3–P5, while a more conservative search is preferable for P1 and P2.

Nevertheless, variations in T_0 do not alter the overall advantage of A-SA over B-GA. With $T_0 = 0.1$, A-SA achieves an aggregated win/tie/loss record of 43/20/2 against B-GA with $n = 100$ and 62/3/0 against B-GA with $n = 20$ over the $5 \times 13 = 65$ test cases. In particular, A-SA achieves statistically significant advantages over both B-GA variants on most problem groups. Although its comparison with B-GA ($n = 100$) on P3 is predominantly statistically equivalent, with a win/tie/loss record of 2/10/1 and $p = 0.95$, no systematic disadvantage is observed.

Moreover, a descriptive comparison of the mean values shows that A-SA with either $T_0 = 1$ or $T_0 = 10$ still outperforms both B-GA variants across all 65 test cases. Therefore, the initial temperature mainly affects the magnitude of the performance improvement rather than the qualitative ranking between A-SA and B-GA. Since both A-SA and B-GA adopt the same binary occupancy representation, the advantage of A-SA cannot be explained by the representation alone. Instead, it is more closely associated with the local and incremental search mechanism of SA. The add, remove, and move neighborhood operations modify only a small portion of the layout at each iteration, allowing promising spatial structures to be retained while neighboring turbine-number and layout configurations are explored. By contrast, crossover-based search may introduce more extensive changes and disrupt beneficial spatial patterns. The consistent advantage of A-SA over different B-GA configurations across a reasonable range of T_0 therefore supports the effectiveness and robustness of the structure-preserving local search behavior of SA.

Table 10. Parameter sensitivity of A-SA and performance comparison with B-GA.

	A-SA $T_0 = 0.1$		A-SA $T_0 = 1$			A-SA $T_0 = 10$			B-GA $n = 100$			B-GA $n = 20$	
P1	mean	std	mean	std	=	mean	std	+	mean	std	+	mean	std
F1	754.18	0.82	754.31	0.67	=	753.53	1.27	+	744.22	7.02	+	733.46	6.93
F2	736.00	0.63	736.00	0.95	=	735.65	1.00	=	717.82	12.54	+	708.34	9.82
F3	754.53	0.74	754.75	0.51	=	754.36	0.80	=	741.97	7.28	+	728.94	8.47
F4	715.27	0.42	715.16	0.51	=	714.82	0.95	=	700.44	9.10	+	692.63	7.95
F5	715.13	0.72	715.19	0.49	=	714.67	0.89	+	699.54	8.63	+	691.39	11.63
F6	754.61	0.74	754.48	0.71	=	754.37	0.78	=	743.71	6.95	+	736.27	5.58
F7	723.75	0.93	723.27	0.61	=	723.15	0.63	=	712.69	5.01	+	706.93	5.06
F8	745.96	1.07	746.00	0.80	=	745.23	0.99	+	731.59	8.40	+	721.25	8.02
F9	754.39	0.63	754.38	0.87	=	753.99	1.21	=	741.35	7.16	+	732.60	7.04
F10	734.63	0.73	734.92	0.55	=	734.16	0.92	=	722.79	6.96	+	713.67	10.66
F11	735.02	0.61	734.74	0.76	+	734.68	0.69	+	719.43	7.06	+	715.18	7.32
F12	754.52	0.77	754.38	0.69	=	753.84	0.95	+	744.65	7.24	+	734.20	7.78
F13	743.45	0.98	743.46	1.08	=	742.83	1.26	+	726.77	6.15	+	718.99	9.08
W/T/L	-/-/-		1/12/0			6/7/0			13/0/0			13/0/0	
p-value	-		0.80			0.00			0.00			0.00	
P2	mean	std	mean	std	=	mean	std	+	mean	std	+	mean	std
F1	690.51	2.82	691.08	3.38	=	688.00	3.93	+	686.32	4.65	+	686.60	4.37
F2	671.88	1.23	672.21	0.45	-	672.12	0.23	=	669.40	1.84	+	667.84	3.86
F3	676.22	2.07	675.35	2.33	=	675.17	2.38	+	673.06	3.33	+	673.04	3.21
F4	671.97	1.33	672.05	0.13	=	672.17	0.35	=	670.09	1.91	+	668.10	2.97
F5	675.95	2.35	675.95	2.16	=	674.63	2.75	+	672.82	3.34	+	673.66	2.97
F6	682.22	1.58	681.74	2.07	=	681.13	1.65	+	677.63	2.81	+	677.35	3.58
F7	683.43	1.18	683.08	1.37	=	682.60	1.02	+	680.64	2.09	+	679.41	2.38
F8	679.78	2.52	678.83	2.42	+	679.12	3.19	=	676.25	2.08	+	676.03	3.40
F9	683.13	2.99	684.92	3.23	-	681.89	4.06	=	679.51	4.23	+	680.61	3.42
F10	680.18	2.85	678.88	2.80	+	678.32	2.41	+	676.27	2.68	+	675.58	4.23
F11	684.33	3.27	683.49	4.09	=	682.45	3.72	+	679.98	3.94	+	679.78	3.66
F12	687.07	2.48	686.18	2.93	=	684.73	3.25	+	681.98	3.06	+	682.12	3.39
F13	686.56	3.58	686.19	3.69	=	683.10	2.61	+	681.82	3.49	+	682.41	3.42
W/T/L	-/-/-		2/9/2			9/4/0			13/0/0			13/0/0	
p-value	-		0.19			0.00			0.00			0.00	
P3	mean	std	mean	std	=	mean	std	+	mean	std	+	mean	std
F1	811.41	6.81	813.88	6.66	-	817.03	6.02	-	811.29	7.63	=	801.73	10.30
F2	785.97	7.81	787.37	6.89	=	791.63	4.45	-	786.26	7.48	=	779.23	10.21
F3	805.23	8.46	809.29	6.31	-	813.78	3.50	-	806.58	8.85	=	802.17	8.94
F4	782.37	7.46	787.66	5.75	-	787.58	5.63	-	783.31	6.73	=	777.26	8.50
F5	781.17	3.22	782.22	3.12	=	783.22	2.98	-	780.07	4.67	=	776.75	4.85
F6	798.10	6.79	799.62	6.93	=	803.70	4.62	-	799.29	6.84	=	791.97	9.67
F7	775.42	6.62	779.39	3.69	-	780.00	3.70	-	773.51	4.23	+	767.74	7.42
F8	798.10	8.82	802.02	7.15	-	805.60	4.50	-	800.53	6.85	=	793.02	10.09
F9	810.36	7.13	812.54	6.59	=	814.33	6.67	-	808.30	9.57	=	801.30	11.53
F10	799.00	8.05	803.31	5.19	-	805.51	5.12	-	796.73	8.21	=	791.52	11.01
F11	793.48	6.70	795.38	5.19	=	798.45	4.93	-	794.36	4.87	=	789.55	7.28
F12	805.36	7.57	809.25	4.84	-	811.68	5.53	-	808.01	7.82	-	796.83	9.94
F13	776.57	4.79	779.40	3.46	-	780.11	3.51	-	774.22	4.33	+	771.44	5.16
W/T/L	-/-/-		0/5/8			0/0/13			2/10/1			12/1/0	
p-value	-		0.00			0.00			0.95			0.00	
P4	mean	std	mean	std	=	mean	std	+	mean	std	+	mean	std
F1	616.05	3.47	618.88	2.45	-	618.82	2.45	-	611.30	4.42	+	613.45	3.71
F2	596.98	2.48	598.82	1.86	-	598.65	1.82	-	592.04	4.26	+	593.16	3.83
F3	613.10	1.48	613.63	1.24	-	613.92	1.01	-	608.04	4.38	+	611.23	2.66
F4	597.37	3.18	598.93	2.00	-	598.69	2.16	=	593.31	3.43	+	594.25	3.68
F5	612.88	1.69	613.99	0.84	-	613.71	1.06	-	609.68	3.15	+	610.95	2.41
F6	614.90	3.08	616.63	1.57	-	616.73	2.35	-	610.84	3.60	+	611.64	2.34
F7	600.10	4.76	601.83	5.03	=	603.52	3.93	-	595.34	5.60	+	596.57	6.24
F8	607.14	2.52	608.95	1.92	-	609.33	1.81	-	603.85	3.62	+	604.66	3.63
F9	614.25	2.20	615.30	1.19	-	615.48	1.26	-	610.47	3.15	+	611.41	3.84
F10	606.72	2.71	610.14	2.12	-	609.99	1.95	-	602.77	4.20	+	604.40	3.71
F11	614.46	2.27	615.33	1.60	=	615.60	1.48	-	610.27	3.62	+	612.17	3.09
F12	615.62	2.95	617.82	2.05	-	617.47	2.37	-	611.23	3.68	+	611.96	3.85
F13	610.13	3.63	611.43	3.59	=	613.41	3.38	-	606.09	4.55	+	606.69	3.54
W/T/L	-/-/-		0/3/10			0/1/12			13/0/0			13/0/0	
p-value	-		0.00			0.00			0.00			0.00	
P5	mean	std	mean	std	=	mean	std	=	mean	std	=	mean	std
F1	238.15	0.81	238.52	0.96	=	238.53	0.96	=	237.52	1.51	=	236.81	1.22
F2	231.69	0.95	232.74	1.07	-	232.92	1.38	-	232.52	1.14	-	231.14	1.90
F3	236.99	1.14	237.76	0.95	-	237.78	0.90	-	236.67	0.97	=	235.31	1.49
F4	230.54	1.27	231.82	0.98	-	231.86	0.81	-	230.60	1.29	=	229.99	1.53
F5	234.20	0.72	234.73	0.47	-	234.82	0.46	-	233.50	0.96	+	232.92	1.20
F6	237.93	0.90	238.76	0.98	-	238.73	0.78	-	237.41	1.39	=	235.96	1.13
F7	231.56	1.01	232.26	0.60	-	232.02	0.73	=	230.87	0.94	+	230.49	1.23
F8	235.03	1.18	236.28	0.78	-	235.65	1.22	=	235.14	1.64	=	233.92	1.83
F9	237.26	1.05	238.02	1.14	-	237.70	1.04	=	236.95	1.38	=	235.87	1.74
F10	234.86	0.91	235.76	0.65	-	235.24	1.02	=	234.73	1.09	=	233.41	1.50
F11	236.03	0.91	236.38	0.68	=	236.49	0.73	=	235.73	0.78	=	234.38	0.90
F12	237.84	1.40	238.79	0.73	-	238.88	1.06	-	237.15	1.28	=	236.19	1.79
F13	232.15	1.17	233.10	0.54	-	233.01	0.51	-	231.82	0.90	=	230.83	1.61
W/T/L	-/-/-		0/2/11			0/6/7			2/10/1			11/2/0	
p-value	-		0.00			0.00			0.04			0.00	

5.6. Computational Complexity Analysis

Let N denote the population size, d the encoding dimension, m_i the encoded length of an individual i , and n_i the turbine number in the decoded layout. For all methods, the dominant computational cost comes from the wake-effect evaluation during fitness computation, which requires $O(n_i^2)$ per individual.

For D-type methods, the search operation has complexity $O(d)$. The decoding procedure includes rounding, duplicate removal, and binary construction. Although most steps are linear, duplicate handling introduces a worst-case complexity of $O(d \log d)$. Therefore, the per-iteration complexity is

$$O\left(N(d \log d + n_i^2)\right). \quad (45)$$

For C-type methods, the fixed-length latent representation is mapped to a variable-length solution through a sequential genotype–phenotype transcription process. The mapping cost scales linearly with both the latent dimension and the decoded solution length, i.e.,

$$O(d + n_i). \quad (46)$$

Thus, the per-iteration complexity is

$$O\left(N(d + n_i + n_i^2)\right). \quad (47)$$

For T-type methods, the representation directly operates on variable-length structures. Since the search and decoding operations scale linearly with the representation dimension, the per-iteration complexity is

$$O\left(N(d + n_i^2)\right). \quad (48)$$

For the proposed B-GA and S-GA methods, binary-vector manipulation, repair, reshaping, and region-based structural crossover require at most linear complexity with respect to the number of grid cells $O(d)$. Including wake-effect evaluation, their per-iteration complexity is

$$O\left(N(d + n_i^2)\right). \quad (49)$$

For A-SA, only one current solution is maintained during search. Each neighborhood transition, including structural manipulation, repair, decoding, and wake-effect evaluation, requires

$$O(d + n_i^2). \quad (50)$$

Given L neighborhood transitions per level, the overall complexity becomes

$$O\left(L(d + n_i^2)\right). \quad (51)$$

Therefore, all methods share the same dominant quadratic evaluation cost associated with wake-effect calculations. Their differences mainly arise from representation-specific operations: D-type has an additional $O(d \log d)$ duplicate-handling cost, C-type introduces a linear $O(d + n_i)$ transcription cost, while T-type, B-GA, S-GA, and A-SA introduce only linear structural overhead.

5.7. Memory Complexity Analysis

The memory complexity is analyzed in terms of the storage required for solution representations, offspring or candidate solutions, and fitness values during the search process. For T-GA, each individual is represented by a variable-length chromosome, where

the length of the i -th individual is m_i . Therefore, the population-level storage depends on the total chromosome length of all individuals, and the memory complexity is $O\left(\sum_{i=1}^N m_i\right)$. Since $m_i \leq d$, the worst-case memory complexity of T-GA becomes $O(Nd)$. The storage of offspring and fitness values does not change this asymptotic order.

For C-GA, D-GA, C-SHADE, D-SHADE, C-SE, and D-SE, the underlying search representation is fixed-dimensional, even though the decoded layouts may contain different numbers of turbines. These methods need to store the population, offspring or trial vectors, and the corresponding fitness values, all based on vectors of length d . Therefore, their dominant memory complexity is $O(Nd)$. The additional temporary variables introduced by mapping, activation/deactivation, or algorithm-specific search operations do not change the overall memory order.

For B-GA and S-GA, each solution is encoded as a fixed-length binary occupancy vector, where each grid cell corresponds to one binary variable. Hence, the population and offspring storage also require $O(Nd)$ memory. In S-GA, the temporary two-dimensional grid used for structure-based region exchange still contains d elements, and it therefore does not alter the asymptotic memory complexity.

For A-SA, no population is maintained. The algorithm only stores the current solution, a candidate neighboring solution, and the best solution found so far. Since each solution is a binary occupancy vector of length d , the memory complexity of A-SA is $O(d)$.

In summary, the fixed-length population-based methods have memory complexity $O(Nd)$. T-GA requires $O\left(\sum_{i=1}^N m_i\right)$ memory on average and $O(Nd)$ memory in the worst case. A-SA has the lowest memory requirement, with complexity $O(d)$, because it maintains only a small number of single-solution vectors.

5.8. Empirical Runtime Analysis

To supplement the asymptotic complexity analysis, we measured the average wall-clock runtime of all algorithms under identical hardware conditions, and the results are shown in Figure 11. The P1 wind condition with the L0 terrain constraint was used, corresponding to a 12×12 grid without infeasible cells. As shown in Figure 11, A-SA requires 0.958 s on average. Although it is slightly slower than C-SE, B-GA, and S-GA, it remains faster than the other six methods, whose runtimes range from 1.105 to 3.461 s. Thus, A-SA achieves strong optimization performance without introducing excessive computational overhead.

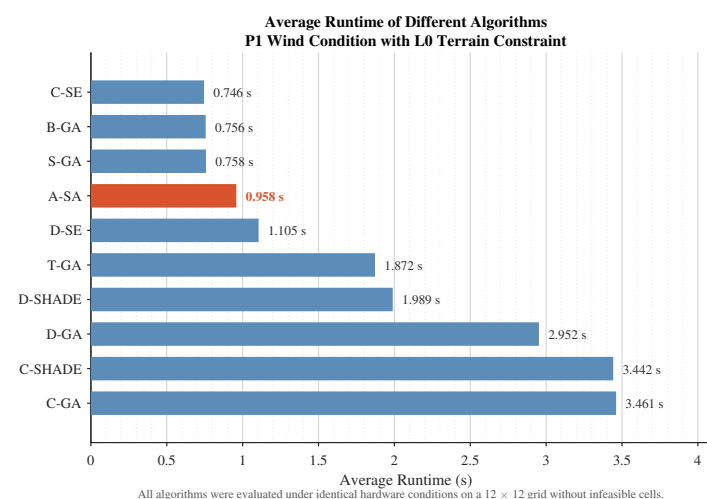


Figure 11. Average wall-clock runtime of the compared algorithms under the P1 wind condition and L0 terrain constraint on a 12×12 grid without infeasible cells. The red bar highlights A-SA, while the blue bars represent the other compared algorithms. All algorithms were evaluated under identical hardware conditions.

The P1–L0 case was selected because it is the simplest and most reproducible experimental condition. Under more demanding conditions, such as P5, a single run may require several hours, making wall-clock measurements more susceptible to background processes and system scheduling. Therefore, the reported results are intended to provide a controlled comparison of the relative computational costs of the algorithms rather than their absolute runtimes for all test conditions.

5.9. Scalability to Large-Scale Wind Farms

Based on the computational and memory complexity analyses in Sections 5.6 and 5.7, the proposed methods can be extended to larger candidate grids in principle. When the number of candidate grid cells d increases to more than 500 or 1000, the representation-level operations of most methods grow linearly with d , except for the D-type methods, which include an additional $O(d \log d)$ duplicate-handling cost. In contrast, the dominant computational burden in WFLO comes from wake-effect fitness evaluation, which scales mainly with the number of installed turbines n_i , approximately $O(n_i^2)$, rather than directly with the number of candidate cells. Therefore, increasing d mainly enlarges the search space and memory usage, while the evaluation cost depends more strongly on the turbine number selected in the decoded layout.

From a memory perspective, population-based methods require $O(Nd)$ storage, whereas A-SA requires only $O(d)$ memory because it maintains a single current solution, a candidate solution, and the best solution found so far. This makes A-SA relatively attractive for large-scale candidate grids. Meanwhile, B-GA and S-GA retain linear representation overhead and preserve grid-level spatial structures, but their population-level storage and recombination costs increase with N and d . For very large-scale WFLO, surrogate-assisted fitness evaluation and parallel computation are promising directions for further improving computational efficiency.

6. Discussion

6.1. Composite Search-Space Characteristics of Variable-Turbine-Number WFLO

Variable-turbine-number WFLO is not merely an extension of conventional fixed-dimensional optimization. In fixed-turbine-number WFLO, the optimization process is conducted within a single layout space of predetermined dimensionality. Once the turbine number becomes a decision variable, however, the overall solution space is no longer a single homogeneous landscape but rather a collection of heterogeneous subspaces with different dimensions. Let $\mathcal{X}_n$ denote the layout space containing n turbines. The complete search space of variable-dimensional WFLO can then be expressed as

$$\mathcal{X} = \bigcup_{n=1}^{N_{\max}} \mathcal{X}_n. \quad (52)$$

Under this formulation, the optimizer no longer searches within a single continuous landscape, but instead navigates a composite landscape formed by multiple interconnected turbine-number subspaces. Each $\mathcal{X}_n$ corresponds to a fixed-turbine-number layout optimization problem, while transitions between different $\mathcal{X}_n$ represent structural modifications caused by adding or removing turbines. Consequently, the difficulty of variable-dimensional WFLO arises not only from the enlarged search space, but also from the lack of intrinsic continuity between different subspaces.

In conventional fixed-dimensional landscapes, neighborhoods are usually defined through small positional perturbations, and local search therefore has a relatively clear geometric meaning [50,51]. Under variable-dimensional conditions, however, neighborhood relationships become substantially more complicated. The neighborhood of a solution

includes not only positional adjustments within the same turbine-number subspace, but also structural transitions into other subspaces through turbine addition, deletion, or replacement. In other words, the neighborhood structure of variable-dimensional WFLO simultaneously contains geometric neighborhoods and dimensional neighborhoods. The former describes continuous changes within a fixed subspace, whereas the latter characterizes transitions across different $\mathcal{X}_n$. Together, they determine the accessibility and searchability of the composite landscape.

From the perspective of fitness landscape theory [52], such a composite search landscape can be viewed as highly rugged and structurally discontinuous. Adding a turbine is not a small perturbation of the objective function, because the newly introduced turbine not only contributes additional power generation but also modifies the wake interactions among existing turbines. Similarly, removing a turbine does not simply reduce one power term; it may release downstream regions and fundamentally alter the wake coupling structure of the entire layout. As a result, transitions from $\mathcal{X}_n$ to $\mathcal{X}_{n+1}$ or $\mathcal{X}_{n-1}$ are often accompanied by abrupt changes in the objective structure. Different turbine-number subspaces therefore may not behave as a smoothly connected manifold-like search space, but they can instead be interpreted as discrete landscape fragments connected through a limited number of structural transitions.

6.2. Representation-Induced Neighborhood Structures

This perspective has direct implications for representation design. Different encoding methods effectively define different neighborhood structures within the composite landscape. Dimension activation/deactivation methods embed all solutions into a fixed-dimensional encoding space. Although this appears to create a unified search space, the decoding process involving discretization, duplicate removal, and activation may distort locality. Neighboring points in the encoding space do not necessarily correspond to neighboring layouts in the phenotype space. Consequently, these representations may introduce additional landscape distortion [53], causing the optimizer to perceive local structures that are inconsistent with actual layout modifications. Rather than eliminating the discontinuities of variable-dimensional landscapes, such methods effectively conceal them within the decoding procedure.

Mapping-based methods instead construct a continuous latent representation that generates layouts with varying turbine numbers. From a landscape perspective, this can be interpreted as building a parameterized projection over the composite search space. Their main advantage lies in preserving a certain degree of continuity, allowing directional information to accumulate in the latent space. However, this smoothness may also become restrictive. If high-quality layouts are distributed across separated turbine-number subspaces, excessively smooth latent mappings may struggle to traverse these disconnected regions effectively. Therefore, mapping-based approaches are likely to be more suitable for problems in which neighboring turbine-number subspaces exhibit relatively consistent structural characteristics.

Explicit variable-length representations expose the composite structure more directly. Since chromosome lengths are allowed to vary, the optimizer can explicitly transition between different $\mathcal{X}_n$. Their advantage therefore lies in enhanced cross-subspace exploration capability. Nevertheless, this also imposes stronger requirements on the search operators. If crossover or mutation changes chromosome length without preserving meaningful spatial structures, the optimizer may frequently jump into regions unrelated to the original layout, resulting in fragmented search trajectories despite improved dimensional mobility.

6.3. Structure Preservation in Occupancy-Based Representations

From a landscape perspective, occupancy-based representations provide a fundamentally different interpretation of variable-dimensional WFLO. Instead of treating the problem as one of changing dimensionality, they reformulate it as occupancy variation on a fixed spatial grid. Under this formulation, all $\mathcal{X}_n$ are embedded in a common binary hypercube, while turbine numbers correspond to different Hamming-weight layers [54]. Adding or removing turbines becomes movement between adjacent layers, whereas turbine relocation corresponds to local occupancy modification within the same or neighboring layers. The resulting representation induces a layered landscape topology:

$$\mathcal{B} = \{0, 1\}^d, \quad n = \sum_{j=1}^d x_j. \quad (53)$$

Within this topology, turbine-number subspaces are no longer completely disconnected. Instead, they are naturally linked through Hamming neighborhoods [54]. Addition, deletion, and movement operators correspond to local transitions either across layers or within nearby regions of the same layer. Occupancy-based representations can be viewed as inducing a more regular and interpretable topological structure on the variable-dimensional landscape.

This interpretation suggests that the key challenge of variable-dimensional optimization is not merely enlarging the search range, but rather defining meaningful neighborhood relationships. If two solutions are close in encoding space but correspond to fundamentally different physical layouts, local search loses practical significance. Conversely, if neighboring encodings correspond to interpretable structural modifications, such as adding a local turbine, removing a strongly interfering turbine, or adjusting a localized occupancy pattern, then the optimizer can progressively exploit structural information embedded within the objective function.

This viewpoint is also consistent with previous WFLO observations. We reported that a GA-enhanced DE hybrid algorithm, despite outperforming DE on the CEC2017 benchmark, consistently underperformed the original DE on WFLOP [55]. We attributed this to GA operations disrupting search directions established by DE. From the perspective of the present study, this can be interpreted as a loss of structural continuity: increased diversity does not necessarily induce meaningful transitions in the layout space, whereas preserving neighborhood relationships and spatial structures may be more important than simply enhancing exploration.

6.4. Implications for Convergence and Search-Space Exploration

The intrinsic difficulty of variable-dimensional WFLO arises from three factors. First, the global landscape comprises multiple heterogeneous turbine-number subspaces. Second, each subspace is highly multimodal because of nonlinear wake coupling. Third, transitions between subspaces introduce additional discontinuities through turbine insertion and deletion. Effective representations must simultaneously satisfy two conflicting requirements. On the one hand, they should allow sufficient accessibility across turbine-number layers. On the other hand, they should preserve local spatial continuity so that meaningful layout structures are not destroyed during dimensional transitions. In this sense, the central challenge of variable-dimensional WFLO lies in balancing cross-subspace mobility and structural continuity.

From this perspective, the primary implication of this study is not that a single representation is universally optimal, but rather that representation fundamentally determines the landscape perceived by the optimizer. Although the objective function itself remains unchanged, different representations may transform it into highly fragmented discrete

landscapes, smoother latent landscapes, or layered occupancy landscapes with regular neighborhood structures. Future research should therefore focus on quantitatively characterizing these induced landscape properties, including locality, fitness autocorrelation, cross-layer accessibility, local-optimum distributions, and fitness barriers between turbine-number layers. A deeper understanding of these landscape characteristics may ultimately provide a more principled foundation for designing search operators specifically suited to variable-dimensional WFLO.

7. Conclusions

From a methodological perspective, this study formulates wind farm layout optimization as a variable-turbine-number optimization problem and systematically compares several representative variable-structure approaches, including dimension activation/deactivation, representation mapping, and variable-length evolutionary representations. In addition, binary occupancy-based structure-oriented search strategies are further investigated to examine the role of spatial layout information during optimization. Experimental results show that the effectiveness of different representations and search mechanisms strongly depends on wind conditions. In more heterogeneous scenarios with complex wake interactions, methods that preserve and exploit spatial layout structures more explicitly generally demonstrate better convergence, robustness, and adaptability.

From this perspective, several design principles can be drawn for future variable-turbine-number WFLO algorithms. First, the representation should preserve explicit spatial occupancy information whenever possible, because both occupied and unoccupied grid cells contribute to the structure of the layout and the resulting wake interactions. Second, the search operators should be consistent with the representation: local operators are suitable when gradual turbine-number adjustment and layout refinement are desired, whereas disruptive recombination should be used carefully because it may destroy useful spatial structures. Third, effective algorithms should balance turbine-number exploration and layout-structure exploitation. In other words, a good variable-turbine-number WFLO method should be able to explore different turbine-number subspaces while maintaining stable and interpretable search paths within each subspace.

It should be noted that the main contribution of this work lies primarily in the empirical observation and comparative analysis of these behaviors. The results suggest that the preservation of meaningful spatial layout structures may play an important role in variable-turbine-number WFLO. However, the current study does not yet rigorously isolate the effects of representation and search dynamics, nor does it provide a theoretical analysis of the underlying search-space topology and structural continuity across turbine-number subspaces. Therefore, the conclusions of this paper should be regarded as exploratory observations and reasonable hypotheses rather than strict mechanistic proofs. Future work will focus on structure-aware search mechanisms and the theoretical characterization of variable-structure WFLO problems. In particular, further studies will investigate the relationship between spatial layout structures, search operators, and search-space topology, with the aim of better understanding how structural information influences optimization behavior in heterogeneous WFLO scenarios.

From the perspective of real-world implementation, several practical constraints should be considered when applying variable-turbine-number WFLO methods. In this study, land-use restrictions are incorporated through the wind farm land model described in Appendix A.2. The wind farm area is discretized into feasible and infeasible grid cells, and turbines can only be installed in feasible cells. This setting allows the optimization process to account for restricted regions caused by environmental, safety, regulatory, or landowner-related constraints. In addition, the objective function in Appendix A.5 includes

the aggregate cost term $T(n)$, which penalizes layouts with different numbers of turbines and approximately reflects installation-related costs.

However, the present study also has several limitations. First, cable routing and maintenance costs are not modeled as independent engineering subproblems. Instead, they are only approximately represented through the aggregate cost formulation. In practical wind farm design, cable-network topology, substation location, terrain-dependent construction costs, transportation access, and maintenance accessibility may all influence the final layout. Second, the current experiments are conducted on simplified gridded wind farm scenarios, whereas real-world offshore wind farms often involve additional constraints, such as seabed conditions, vessel accessibility, offshore installation requirements, and grid-connection limitations. Third, the present evaluation model assumes predefined wind conditions. Future research should further investigate dynamic wind conditions, including time-varying wind direction, wind speed distributions, seasonal variations, and uncertainty in wind-resource estimation. Finally, although the fitness function considers power generation and an aggregate cost term, the problem is still formulated as a single-objective optimization task. Extending the proposed structure-preserving representations to multi-objective WFLO, where energy production, cost, land-use impact, cable routing, reliability, and maintenance accessibility are optimized simultaneously, is an important direction for future work.

Author Contributions: Conceptualization, H.Y. and G.C.; methodology, H.Y., Y.Y. and Y.N.; software, H.Y. and S.L.; validation, H.Y., Q.F., A.Z. and Y.N.; formal analysis, H.Y. and Y.N.; investigation, H.Y. and A.Z.; resources, H.Y.; data curation, H.Y.; writing—original draft preparation, H.Y.; writing—review and editing, Y.Y. and Y.N.; visualization, H.Y. and Q.F.; supervision, Y.Y., G.C., and Y.N.; project administration, Y.N.; funding acquisition, H.Y. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by the Japan Society for the Promotion of Science (JSPS) KAKENHI under grant numbers 26K21336 and 25K00139.

Data Availability Statement: The source code, dataset preprocessing scripts, and experimental configuration files used to support the findings of this study are available at <https://github.com/Hai-chuan-Yang/A-Comparative-Study-of-Variable-Length-Encoding-Methods-for-Wind-Farm-Layout-Optimization> (accessed on 18 July 2026).

Conflicts of Interest: Author Guodong Cui was employed by the company Wesoftware Company Ltd. The remaining authors declare that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.

Abbreviations

The following abbreviations are used in this manuscript:

WFLO	Wind Farm Layout Optimization
MHAs	Metaheuristic Algorithms
GA	Genetic Algorithm
DE	Differential Evolution
SE	Spherical Evolution
SA	Simulated Annealing

Appendix A. The Wind Farm Layout Problem Model

Within wind farm layout problem models, there are typically five sub-models: the wind condition model, the wake effect model, the wind turbine model, the wind farm land model, and the wind farm cost model. In this study, we employ a wind farm layout problem model that considers participation among landowners, as proposed in [56]. This model

adequately accounts for land-use constraints, making it suitable for optimizing wind farms in different regions.

Appendix A.1. The Wind Condition Model

Wind conditions are represented by a discrete joint probability distribution

$$P = \{(\theta_i, v_j, f_{ij}) \mid i = 1, \dots, N_\theta, j = 1, \dots, N_v\}, \quad \sum_{i=1}^{N_\theta} \sum_{j=1}^{N_v} f_{ij} = 1, \quad (\text{A1})$$

where θ_i denotes a discretized wind direction, v_j a wind speed bin, and f_{ij} the occurrence probability of wind state (θ_i, v_j) . This formulation provides a unified representation for all wind scenarios considered in this study. The wind profiles used for evaluation can be grouped into two categories.

(1) Direction-only uncertainty (constant wind speed). Profiles P1–P3 assume a fixed ambient wind speed $v = 13$ m/s, while differing in the probability distributions over wind directions. These profiles represent increasing levels of directional variability, ranging from a single prevailing direction to multiple directions with non-uniform probabilities. In this case, the joint distribution degenerates to a directional probability distribution $p(\theta_i)$.

(2) Direction–speed joint uncertainty. Profiles P4–P5 consider both wind direction and wind speed uncertainties. P4 is a synthetic multi-direction, multi-speed distribution with identical probability structures across directions. The P5 scenario (Figure A1) is derived from a real offshore wind dataset NeoWins [57] provided by the New Energy and Industrial Technology Development Organization (NEDO), Japan. The site is located at $33^\circ 53' 50.49''$ N and $134^\circ 44' 13.51''$ E, with a water depth of approximately 35 m and a distance of 1.9 km from the coast. At a hub height of 80 m, the long-term mean wind speed is 7.53 m/s with a standard deviation of 0.288 m/s. Wind speed follows a Weibull distribution (shape $k = 2.00$, scale $c = 8.50$ m/s), while the direction–speed joint uncertainty is defined by the wind rose with non-uniform directional frequencies and dominant sectors, leading to asymmetric wake effects. The vertical profile is modeled by a power law with exponent 0.085. Compared with synthetic scenarios, P5 incorporates coupled direction–speed variability and stochastic fluctuations, providing a realistic and challenging benchmark for WFLO.

Detailed parameter settings of all wind profiles are summarized in Table A1. For notational convenience in subsequent formulations, we denote $p(\theta_i, v_j) = f_{ij}$, i.e., $p(\theta, v)$ represents the probability mass function associated with the discrete distribution P .

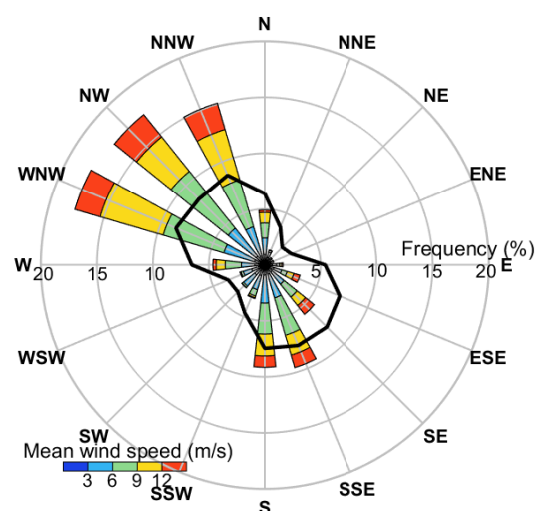


Figure A1. Wind rose of the P5 offshore wind scenario derived from NEDO data.

Table A1. Summary of wind profiles used in the study.

Profile	Wind Directions θ	Wind Speeds v (m/s)	Probability Structure
P1	Single direction (0)	13	$f_{11} = 1$
P2	$\{0, \frac{\pi}{2}, \pi, \frac{3\pi}{2}\}$	13	$f_{i1} = \frac{1}{4}, \quad i = 1, \dots, 4$
P3	$\{0, \frac{\pi}{3}, \frac{2\pi}{3}, \pi, \frac{4\pi}{3}, \frac{5\pi}{3}\}$	13	$(f_{11}, f_{21}, f_{31}, f_{41}, f_{51}, f_{61}) = (0.2, 0.3, 0.2, 0.1, 0.1, 0.1)$
P4	$\{0, \frac{\pi}{6}, \frac{2\pi}{6}, \frac{3\pi}{6}, \frac{4\pi}{6}, \frac{5\pi}{6}, \frac{6\pi}{6}, \frac{7\pi}{6}, \frac{8\pi}{6}, \frac{9\pi}{6}, \frac{10\pi}{6}, \frac{11\pi}{6}\}$	$\{13, 10, 7\}$	$(f_{i1}, f_{i2}, f_{i3}) = (0.058333, 0.016667, 0.008333), \quad i = 1, \dots, 12$
P5	N_θ sectors (measured)	N_v bins (measured)	Derived from wind rose data

Appendix A.2. The Wind Farm Land Model

The wind farm area is discretized into a two-dimensional regular grid, where each grid cell may host at most one wind turbine. Let $i \in \{1, 2, \dots, G\}$ denote the index of a grid cell. Following the original setting, the land area consists of $G = 144$ square cells arranged in $S_r \times S_c$ rows and columns, with $S_r = S_c = 12$.

The spatial location of grid point i in the physical coordinate system is defined through mapping

$$(x_i, y_i) = \Phi(i), \quad (\text{A2})$$

where $\Phi(\cdot)$ transforms the discrete grid index into the corresponding physical coordinates. The mapping $\Phi(\cdot)$ implicitly consists of two stages. First, the grid index i is assigned to a standard two-dimensional index coordinate (λ_i, φ_i) in an $S_r \times S_c$ grid, where

$$\lambda_i = i - S_c \left\lfloor \frac{i-1}{S_c} \right\rfloor - 1, \quad \varphi_i = \left\lfloor \frac{i-1}{S_c} \right\rfloor. \quad (\text{A3})$$

Second, the standard coordinates are converted into physical coordinates (in meters) via

$$x_i = \left(\lambda_i + \frac{1}{2}\right)C, \quad y_i = \left(\varphi_i + \frac{1}{2}\right)C, \quad (\text{A4})$$

where C denotes the cell width. The coordinates (x_i, y_i) therefore represent the center location of cell i and are used as the geometric inputs to the subsequent wake and power generation models. This mapping incorporates the grid geometry, including the cell width C and the grid-center offset, such that (x_i, y_i) represents the center location of cell i in the wind farm.

To account for different wind directions, the physical coordinates are further expressed in a wind-aligned coordinate system. Given a wind direction angle θ , the coordinates of turbine i in the wind-direction reference frame are obtained via a planar rotation,

$$\begin{bmatrix} x_i^\theta \\ y_i^\theta \end{bmatrix} = R(\theta) \begin{bmatrix} x_i \\ y_i \end{bmatrix}, \quad R(\theta) = \begin{bmatrix} \cos \theta & -\sin \theta \\ \sin \theta & \cos \theta \end{bmatrix}. \quad (\text{A5})$$

For clarity, the land-use constraints are illustrated in Figure A2. The wind farm area is discretized into a grid of candidate cells, where different colors represent feasibility conditions imposed by the landowner. Specifically, the green cells denote feasible locations where wind turbines can be installed, while the red shaded cells indicate restricted or prohibited regions (e.g., due to environmental, safety, or regulatory constraints). These infeasible areas are excluded from the candidate set during the optimization process, ensuring that all generated layouts satisfy practical land-use requirements.

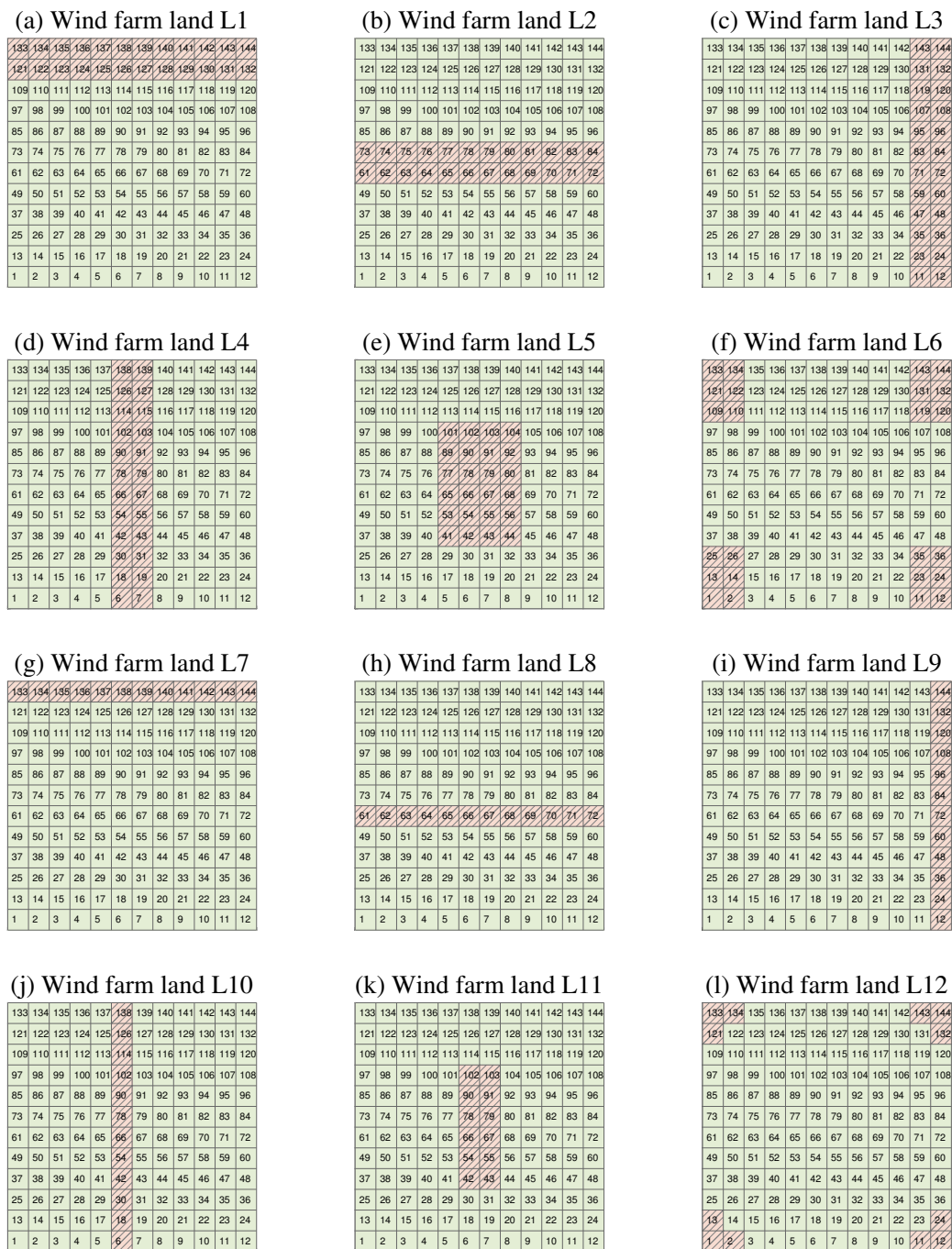


Figure A2. A diagram of a wind farm showing restrictions placed by the landowner. Reproduced from Ref. [30].

Appendix A.3. The Wake Effect Model

In the wind farm model proposed by [56], the classical wake effect model is utilized, which assumes the conservation of momentum [58]. According to the law of conservation of momentum, the wake radius behind a turbine increases linearly with respect to downwind distance [59]. Figure A3 illustrates the wake effect behind a wind turbine with an ambient wind speed of v_0 . The wake wind speed v_1 just behind the turbine can be approximated as $\frac{1}{3}v_0$ [60]. The symbols r_0 , r_1 , and v_2 represent the rotor radius, wake effect radius, and wake velocity, respectively, at a specific downwind distance d that varies with wind direction θ . The decay constant α determines the rate at which the wake expands with

distance. Equation (A6) is the expression for the law of conservation of momentum for wind turbines.

$$\pi r_0^2 v_1 + \pi(r_1^2 - r_0^2)v_0 = \pi r_1^2 v_2 \quad (\text{A6})$$

v_2 can be derived as follows:

$$v_2 = v_0 \left[1 - \frac{2}{3} \left(\frac{r_0}{r_1} \right)^2 \right] \quad (\text{A7})$$

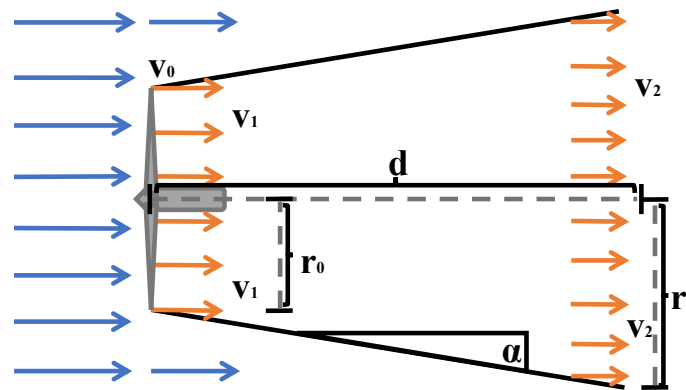


Figure A3. The wake effect model of a single wind turbine. Reproduced from Ref. [30].

The wake effect radius can be derived as $r_1 = r_0 + \alpha d$, and the dimensionless scalar α is given as follows:

$$\alpha = \frac{0.5}{\ln\left(\frac{h}{s}\right)} \quad (\text{A8})$$

where h is the wind turbine's hub height and, s represents the surface roughness of the wind farm area.

If wind turbine i is affected by the wake of multiple upwind turbines, we can use the index set A to represent all the wind turbines in the upwind direction of turbine i . The following equation can be derived:

$$\left(1 - \frac{v_i}{v_0} \right)^2 = \sum_{j=1}^A \left(1 - \frac{v_{i,j}}{v_0} \right)^2 \quad (\text{A9})$$

where, under a variety of wake effects, v_i represents the actual wind speed at wind turbine i , and $v_{i,j}$ represents the wind speed of turbine i only considering the wind turbine j 's wake effect using Equation (A7). By solving Equation (A9), the actual wind speed v_i is as follows:

$$v_i = v_0 \left[1 - \sqrt{\sum_{j=1}^A \left(1 - \frac{v_{i,j}}{v_0} \right)^2} \right] \quad (\text{A10})$$

For turbines i and j , the downstream distance and lateral offset under wind direction θ are defined as

$$d_{ij}^\theta = x_i^\theta - x_j^\theta, \quad \Delta_{ij}^\theta = y_i^\theta - y_j^\theta. \quad (\text{A11})$$

Following the wake expansion model, turbine i is affected by the wake of turbine j if it is located downstream and lies within the lateral extent of the wake at distance d_{ij}^θ . This condition is encoded by the indicator function

$$\mathbb{I}_{ij}^\theta = 1 \left(d_{ij}^\theta > 0, \left| \Delta_{ij}^\theta \right| \leq r_0 + \alpha d_{ij}^\theta \right), \quad (\text{A12})$$

where $1(\cdot)$ equals 1 if the condition is satisfied and 0 otherwise. The indicator $\mathbb{I}_{ij}^\theta$ provides an explicit realization of the upstream turbine set A .

Using this notation, Equation (A10) can be equivalently expressed as

$$v_i(\theta, v_0, X) = v_0 \left[1 - \sqrt{\sum_{j \neq i} \mathbb{I}_{ij}^\theta \left(\frac{2}{3} \left(\frac{r_0}{r_0 + \alpha d_{ij}^\theta} \right)^2 \right)^2} \right], \quad (\text{A13})$$

which is mathematically equivalent to Equation (A10) but makes the geometric screening of upstream wake interactions is explicit, and X denotes the wind farm layout represented by the solution vector, which determines the spatial coordinates of turbines as defined in Equations (A2)–(A5). In the metaheuristic optimization process, X corresponds to an individual in the population representing a candidate layout.

Appendix A.4. Power Generation Model

Given the effective wind speed at each turbine obtained from the wake-induced wind speed model, the power output of turbine i is determined by its power curve. Let $v_i(\theta, v, X)$ denote the effective wind speed at turbine i under wind direction θ , ambient wind speed v , and layout X . The corresponding power output is given by

$$o_i(\theta, v, X) = W(v_i(\theta, v, X)), \quad (\text{A14})$$

where $W(\cdot)$ represents the turbine power generation function.

Following the standard piecewise definition, the power curve $W(\cdot)$ is expressed as

$$W(v_i) = \begin{cases} 0, & \text{if } v_i < 2 \\ 0.3v_i^3, & \text{if } 2 \leq v_i < 12.8 \\ 629.1, & \text{if } 12.8 \leq v_i < 18 \\ 0, & \text{if } 18 \leq v_i \end{cases} \quad (\text{A15})$$

The wind farm model utilized in this study uses the same type (*GE1.5sle*) of wind turbines as in [56]. The turbines have a diameter of 77 m ($r_0 = 38.5$ m) and a hub height of 88 m. The power output $o_i(\theta, v, X)$ therefore provides the final mapping from the wind field and turbine layout to the electrical output of each turbine.

Appendix A.5. Objective Function of the Wind Farm Layout Optimization Problem

To maximize power production in a wind farm with a given wind distribution profile, it is essential to determine the optimal location for each wind turbine. A key consideration is minimizing the wake effect in order to provide wind turbines downwind with a consistent and high input wind speed. The objective function of Mosetti et al. [61] is used as follows:

$$\max_{n, X} \frac{E(n, X)}{T(n)} \quad (\text{A16})$$

where X represents a feasible layout for a wind farm with n wind turbines. In metaheuristics, X is the individual of the algorithm. The term $E(n, X)$ represents the total power generated by the n wind turbines arranged according to the layout defined by X . Mean-

while, $T(n)$ refers to the total cost of installing n wind turbines on the wind farm. $E(n, X)$ and $T(n)$ can be calculated as follows:

$$E(n, X) = \sum_{i=1}^n \sum_{\theta, v} p(\theta, v) o_i(\theta, v, X), \quad T(n) = n \left(\frac{2}{3} + \frac{1}{3} e^{-0.00174n^2} \right) \quad (\text{A17})$$

where θ is the wind direction with the wind speed v . $p(\theta, v)$ is the probability distribution of θ and v . $o_i(\theta, v, X)$ is the power output of wind turbine i under layout X with θ and v .

References

1. Chehouri, A.; Younes, R.; Ilinca, A.; Perron, J. Review of performance optimization techniques applied to wind turbines. *Appl. Energy* **2015**, *142*, 361–388. [\[CrossRef\]](#)
2. Samorani, M. *The Wind Farm Layout Optimization Problem*; Springer: Berlin/Heidelberg, Germany, 2013.
3. Yang, X.S. Optimization and metaheuristic algorithms in engineering. *Optim. Metaheuristic Algorithms Eng.* **2013**, 1–23. [\[CrossRef\]](#)
4. Abdel-Basset, M.; Abdel-Fatah, L.; Sangaiah, A.K. Metaheuristic algorithms: A comprehensive review. *Comput. Intell. Multimed. Big Data Cloud Eng. Appl.* **2018**, 185–231. [\[CrossRef\]](#)
5. Bajaj, I.; Arora, A.; Hasan, M.F. Black-box optimization: Methods and applications. In *Black Box Optimization, Machine Learning, and No-Free Lunch Theorems*; Springer: Berlin/Heidelberg, Germany, 2021; pp. 35–65.
6. Yang, Y.; Tao, S.; Li, H.; Yang, H.; Tang, Z. A Multi-Local Search-Based SHADE for Wind Farm Layout Optimization. *Electronics* **2024**, *13*, 3196. [\[CrossRef\]](#)
7. Yeghikian, M.; Ahmadi, A.; Dashti, R.; Esmaeilion, F.; Mahmoudan, A.; Hoseinzadeh, S.; Garcia, D.A. Wind farm layout optimization with different hub heights in manjil wind farm using particle swarm optimization. *Appl. Sci.* **2021**, *11*, 9746. [\[CrossRef\]](#)
8. Yang, K.; Kwak, G.; Cho, K.; Huh, J. Wind farm layout optimization for wake effect uniformity. *Energy* **2019**, *183*, 983–995. [\[CrossRef\]](#)
9. Nguyen, T.T.; Yang, S.; Branke, J. Evolutionary dynamic optimization: A survey of the state of the art. *Swarm Evol. Comput.* **2012**, *6*, 1–24. [\[CrossRef\]](#)
10. Zebulum, R.S.; Vellasco, M.; Pacheco, M.A. Variable length representation in evolutionary electronics. *Evol. Comput.* **2000**, *8*, 93–120. [\[CrossRef\]](#) [\[PubMed\]](#)
11. Lissovoi, A.; Oliveto, P.S. Computational complexity analysis of genetic programming. In *Theory of Evolutionary Computation: Recent Developments in Discrete Optimization*; Springer: Cham, Switzerland, 2020; pp. 475–518.
12. Liu, Y.; Wang, H. Surrogate-assisted hybrid evolutionary algorithm with local estimation of distribution for expensive mixed-variable optimization problems. *Appl. Soft Comput.* **2023**, *133*, 109957. [\[CrossRef\]](#)
13. Ryerkerk, M.L.; Averill, R.C.; Deb, K.; Goodman, E.D. Solving metameric variable-length optimization problems using genetic algorithms. *Genet. Program. Evolvable Mach.* **2017**, *18*, 247–277.
14. Li, M.; Jiang, X.; Carroll, J.; Negenborn, R.R. A multi-objective maintenance strategy optimization framework for offshore wind farms considering uncertainty. *Appl. Energy* **2022**, *321*, 119284. [\[CrossRef\]](#)
15. Hou, P.; Zhu, J.; Ma, K.; Yang, G.; Hu, W.; Chen, Z. A review of offshore wind farm layout optimization and electrical system design methods. *J. Mod. Power Syst. Clean Energy* **2019**, *7*, 975–986. [\[CrossRef\]](#)
16. Sorkhabi, S.Y.D.; Romero, D.A.; Yan, G.K.; Gu, M.D.; Moran, J.; Morgenroth, M.; Amon, C.H. The impact of land use constraints in multi-objective energy-noise wind farm layout optimization. *Renew. Energy* **2016**, *85*, 359–370. [\[CrossRef\]](#)
17. Yin Kwong, W.; Yun Zhang, P.; Romero, D.; Moran, J.; Morgenroth, M.; Amon, C. Multi-objective wind farm layout optimization considering energy generation and noise propagation with NSGA-II. *J. Mech. Des.* **2014**, *136*, 091010. [\[CrossRef\]](#)
18. Chen, Y.; Li, H.; He, B.; Wang, P.; Jin, K. Multi-objective genetic algorithm based innovative wind farm layout optimization method. *Energy Convers. Manag.* **2015**, *105*, 1318–1327. [\[CrossRef\]](#)
19. Moreno, S.R.; Pierezan, J.; dos Santos Coelho, L.; Mariani, V.C. Multi-objective lightning search algorithm applied to wind farm layout optimization. *Energy* **2021**, *216*, 119214. [\[CrossRef\]](#)
20. Montoya, F.G.; Manzano-Agugliaro, F.; López-Márquez, S.; Hernández-Escobedo, Q.; Gil, C. Wind turbine selection for wind farm layout using multi-objective evolutionary algorithms. *Expert Syst. Appl.* **2014**, *41*, 6585–6595. [\[CrossRef\]](#)
21. Zhang, C.; Li, H.; Li, X.; Cheng, J.; Lei, Z.; Gao, S. Probabilistic bootstrap-based evolutionary algorithm for three-objective wind farm turbine position optimization. *Swarm Evol. Comput.* **2025**, *96*, 101972. [\[CrossRef\]](#)
22. Xue, B.; Zhang, M.; Browne, W.N. Particle Swarm Optimization for Feature Selection in Classification: A Multi-Objective Approach. *IEEE Trans. Cybern.* **2013**, *43*, 1656–1671. [\[CrossRef\]](#) [\[PubMed\]](#)
23. Ryerkerk, M.; Averill, R.; Deb, K.; Goodman, E. A survey of evolutionary algorithms using metameric representations. *Genet. Program. Evolvable Mach.* **2019**, *20*, 441–478. [\[CrossRef\]](#)

24. Yang, H.; Yang, Y.; Zhang, Y.; Tang, C.; Hashimoto, K.; Nagata, Y. Chaotic Map-Coded Evolutionary Algorithms for Dendritic Neuron Model Optimization. In Proceedings of the 2024 IEEE Congress on Evolutionary Computation (CEC), Yokohama, Japan, 30 June–5 July 2024; pp. 1–8.
25. Yang, H.; Li, H.; Yang, Y.; Chiba, N.; Kagami, S.; Hashimoto, K.; Nagata, Y. Chaotic map-coded metaheuristics for metameric variable-length problems. *Genet. Program. Evolvable Mach.* **2025**, *26*, 19. [\[CrossRef\]](#)
26. Holland, J.H. *Adaptation in Natural and Artificial Systems: An Introductory Analysis with Applications to Biology, Control, and Artificial Intelligence*; The MIT Press: Cambridge, MA, USA, 1992. [\[CrossRef\]](#)
27. Storn, R.; Price, K. Differential evolution—a simple and efficient heuristic for global optimization over continuous spaces. *J. Glob. Optim.* **1997**, *11*, 341–359. [\[CrossRef\]](#)
28. Tang, D. Spherical evolution for solving continuous optimization problems. *Appl. Soft Comput.* **2019**, *81*, 105499. [\[CrossRef\]](#)
29. Ahmad, M.F.; Isa, N.A.M.; Lim, W.H.; Ang, K.M. Differential evolution: A recent review based on state-of-the-art works. *Alex. Eng. J.* **2022**, *61*, 3831–3872. [\[CrossRef\]](#)
30. Yang, H.; Gao, S.; Lei, Z.; Li, J.; Yu, Y.; Wang, Y. An improved spherical evolution with enhanced exploration capabilities to address wind farm layout optimization problem. *Eng. Appl. Artif. Intell.* **2023**, *123*, 106198. [\[CrossRef\]](#)
31. Yang, Y.; Li, H.; Lei, Z.; Yang, H.; Wang, J. A nonlinear dimensionality reduction search improved differential evolution for large-scale optimization. *Swarm Evol. Comput.* **2025**, *92*, 101832. [\[CrossRef\]](#)
32. Emami, A.; Noghreh, P. New approach on optimization in placement of wind turbines within wind farm by genetic algorithms. *Renew. Energy* **2010**, *35*, 1559–1564. [\[CrossRef\]](#)
33. Kirkpatrick, S.; Gelatt, C.D., Jr.; Vecchi, M.P. Optimization by simulated annealing. *Science* **1983**, *220*, 671–680. [\[CrossRef\]](#) [\[PubMed\]](#)
34. Masoudi, S.M.; Baneshi, M. Layout optimization of a wind farm considering grids of various resolutions, wake effect, and realistic wind speed and wind direction data: A techno-economic assessment. *Energy* **2022**, *244*, 123188. [\[CrossRef\]](#)
35. Cao, L.; Ge, M.; Gao, X.; Du, B.; Li, B.; Huang, Z.; Liu, Y. Wind farm layout optimization to minimize the wake induced turbulence effect on wind turbines. *Appl. Energy* **2022**, *323*, 119599. [\[CrossRef\]](#)
36. Hu, W.; Yang, Q.; Yuan, Z.; Yang, F. Wind farm layout optimization in complex terrain based on CFD and IGA-PSO. *Energy* **2024**, *288*, 129745. [\[CrossRef\]](#)
37. Wang, Z.; Tu, Y.; Zhang, K.; Han, Z.; Cao, Y.; Zhou, D. An optimization framework for wind farm layout design using CFD-based Kriging model. *Ocean. Eng.* **2024**, *293*, 116644. [\[CrossRef\]](#)
38. Boucekara, H.R.; Ramli, M.A.; Javaid, M.S. Wind farm layout optimization/expansion of real wind turbines with a parallel collaborative multi-objective optimization algorithm. *Energies* **2024**, *17*, 5632. [\[CrossRef\]](#)
39. Dong, G.; Qin, J.; Wu, C.; Xu, C.; Yang, X. Reinforcement learning-enhanced genetic algorithm for wind farm layout optimization. *Renew. Energy* **2025**, 125093.
40. Shao, Y.X.; Wang, Z.F.; Naung, S.W.; Zhang, K.; Yao, Y.; Zhou, D. Towards high-fidelity wind farm layout optimization using polynomial chaos expansion and Kriging model. *Energy* **2025**, *338*, 138820. [\[CrossRef\]](#)
41. Malisani, P.; Bartement, T.; Bozonnet, P. Offshore wind farm layout optimization with alignment constraints. *Wind. Energy Sci.* **2025**, *10*, 1611–1623. [\[CrossRef\]](#)
42. Gong, Y.; Sun, Y.; Peng, D.; Chen, X. Bridge the gap between fixed-length and variable-length evolutionary neural architecture search algorithms. *Electron. Res. Arch.* **2024**, *32*, 263–292.
43. Wen, L.; Gao, L.; Li, X.; Li, H. A new genetic algorithm based evolutionary neural architecture search for image classification. *Swarm Evol. Comput.* **2022**, *75*, 101191. [\[CrossRef\]](#)
44. Jiang, Y.; Tang, L.; Liu, H.; Zeng, A. A variable-length encoding genetic algorithm for incremental service composition in uncertain environments for cloud manufacturing. *Appl. Soft Comput.* **2022**, *123*, 108902. [\[CrossRef\]](#)
45. Tanabe, R.; Fukunaga, A. Success-History Based Parameter Adaptation for Differential evolution. In Proceedings of the 2013 IEEE Congress on Evolutionary Computation, Cancun, Mexico, 20–23 June 2013; pp. 71–78.
46. Metropolis, N.; Rosenbluth, A.W.; Rosenbluth, M.N.; Teller, A.H.; Teller, E. Equation of State Calculations by Fast Computing Machines. *J. Chem. Phys.* **1953**, *21*, 1087–1092. [\[CrossRef\]](#)
47. García, S.; Molina, D.; Lozano, M.; Herrera, F. A study on the use of non-parametric tests for analyzing the evolutionary algorithms' behaviour: A case study on the CEC'2005 special session on real parameter optimization. *J. Heuristics* **2009**, *15*, 617–644.
48. Zimmerman, D.W.; Zumbo, B.D. Relative power of the Wilcoxon test, the Friedman test, and repeated-measures ANOVA on ranks. *J. Exp. Educ.* **1993**, *62*, 75–86. [\[CrossRef\]](#)
49. Wilcoxon, F. Individual comparisons by ranking methods. In *Breakthroughs in Statistics: Methodology and Distribution*; Springer: New York, NY, USA, 1992; pp. 196–202.
50. Reidys, C.M.; Stadler, P.F. Combinatorial landscapes. *SIAM Rev.* **2002**, *44*, 3–54. [\[CrossRef\]](#)
51. Zou, F.; Chen, D.; Liu, H.; Cao, S.; Ji, X.; Zhang, Y. A survey of fitness landscape analysis for optimization. *Neurocomputing* **2022**, *503*, 129–139. [\[CrossRef\]](#)

52. Stadler, P.F. Fitness landscapes. In *Biological Evolution and Statistical Physics*; Springer: Berlin/Heidelberg, Germany, 2002; pp. 183–204.
53. Rothlauf, F. Representations for Genetic and Evolutionary Algorithms. In *Representations for Genetic and Evolutionary Algorithms*; Springer: Berlin/Heidelberg, Germany, 2006; pp. 9–32.
54. Harary, F.; Hayes, J.P.; Wu, H.J. A survey of the theory of hypercube graphs. *Comput. Math. Appl.* **1988**, *15*, 277–289. [[CrossRef](#)]
55. Ye, Z.; Yang, H.; Chiba, N.; Hashimoto, K. Comparative Study of Hybridization and Parameter Tuning Improvement Methods for EAs in WFLOP. In *CIIS 2023: 2023 the 6th International Conference on Computational Intelligence and Intelligent Systems, Tokyo Japan, 25–27 November 2023*; Association for Computing Machinery: New York, NY, USA, 2024; pp. 59–65.
56. Ju, X.; Liu, F.; Wang, L.; Lee, W.J. Wind farm layout optimization based on support vector regression guided genetic algorithm with consideration of participation among landowners. *Energy Convers. Manag.* **2019**, *196*, 1267–1281. [[CrossRef](#)]
57. New Energy and Industrial Technology Development Organization (NEDO). NeoWins Offshore Wind Condition Map. Available online: https://offshorewind.infop.nedo.go.jp/Nedo_Webgis/top.html (accessed on 17 January 2026).
58. Wu, Y.K.; Su, P.E.; Su, Y.S.; Wu, T.Y.; Tan, W.S. Economics- and Reliability-Based Design for an Offshore Wind Farm. *IEEE Trans. Ind. Appl.* **2017**, *53*, 5139–5149. [[CrossRef](#)]
59. Schepers, J. *ENDOW: Validation and Improvement of ECN's Wake Model*; Energy Research Centre of the Netherlands ECN Petten: Petten, The Netherlands, 2003.
60. Du Pont, B.L.; Cagan, J. An Extended Pattern Search Approach to Wind Farm Layout Optimization. *J. Mech. Des.* **2012**, *134*, 081002. [[CrossRef](#)]
61. Mosetti, G.; Poloni, C.; Diviacco, B. Optimization of wind turbine positioning in large windfarms by means of a genetic algorithm. *J. Wind. Eng. Ind. Aerodyn.* **1994**, *51*, 105–116. [[CrossRef](#)]

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.