

Article

Enhanced Hyper-Cube Framework Ant Colony Optimization for Combinatorial Optimization Problems

Ali Ahmid ^{1,*}, Thien-My Dao ² and Ngan Van Le ²¹ Independent Researcher, Montréal, QC H3C 1K3, Canada² Mechanical Engineering Department, École de Technologie Supérieure ÉTS, Montréal, QC H3C 1K3, Canada; thien-my.dao@etsmtl.ca (T.-M.D.); vanngan.le@etsmtl.ca (N.V.L.)

* Correspondence: ali-elmbrok-salem.ahmid.1@ens.etsmtl.ca

Abstract: Solving of combinatorial optimization problems is a common practice in real-life engineering applications. Trusses, cranes, and composite laminated structures are some good examples that fall under this category of optimization problems. Those examples have a common feature of discrete design domain that turn them into a set of NP-hard optimization problems. Determining the right optimization algorithm for such problems is a precious point that tends to impact the overall cost of the design process. Furthermore, reinforcing the performance of a prospective optimization algorithm reduces the design cost. In the current study, a comprehensive assessment criterion has been developed to assess the performance of meta-heuristic (MH) solutions in the domain of structural design. Thereafter, the proposed criterion was employed to compare five different variants of Ant Colony Optimization (ACO). It was done by using a well-known structural optimization problem of laminate Stacking Sequence Design (SSD). The initial results of the comparison study reveal that the Hyper-Cube Framework (HCF) ACO variant outperforms the others. Consequently, an investigation of further improvement led to introducing an enhanced version of HCFACO (or EHCFCACO). Eventually, the performance assessment of the EHCFCACO variant showed that the average practical reliability became more than twice that of the standard ACO, and the normalized price decreased more to hold at 28.92 instead of 51.17.

Keywords: combinatorial optimization; Ant Colony Optimization (ACO); buckling load factor; composite laminate



Citation: Ahmid, A.; Dao, T.-M.; Le, N.V. Enhanced Hyper-Cube Framework Ant Colony Optimization for Combinatorial Optimization Problems. *Algorithms* **2021**, *14*, 286. <https://doi.org/10.3390/a14100286>

Academic Editors: Hsiang-Ling Chen, Yun-Chia Liang, Mehmet Fatih Tasgetiren and Quan-Ke Pan

Received: 30 August 2021

Accepted: 28 September 2021

Published: 29 September 2021

Publisher's Note: MDPI stays neutral with regard to jurisdictional claims in published maps and institutional affiliations.



Copyright: © 2021 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) license (<https://creativecommons.org/licenses/by/4.0/>).

1. Introduction

Combinatorial optimization is devoted to the mathematical process of searching for the optimal solution (maxima or minima) of an objective function with a discrete domain of decision variables. The possible number of solutions for a combinatorial optimization problem is equal to $[D]^n$, where D is the discrete design domain vector and n represents the number of design variables [1]. Therefore, the optimization problem becomes more computationally difficult to be solved when the number of design variables increases. Accordingly, many combinatorial optimization problems are hard to solve within deterministic polynomial time (or NP-hard). A Travelling Salesman (or TSP) is a typical example of this type of optimization problem where the number of cities to be visited is given and the shortest path needs to be determined [2]. As the number of cities increases, the number of possible solutions increases too and this leads to the computational complexity of the problem, where it is not possible to enumerate all these solution possibilities with limited computation resources, such as memory size or processor speed. Furthermore, many structural design problems fall under this category of optimization problems because of the discrete nature of their design domain. It makes the objective function a multimodal (non-convex) function. As a consequence, the design space will have more than one optimal solution which then turns the structure design into an NP-hard design optimization problem. Trusses, beams, and composite structures are good examples of

real-life combinatorial optimization problems [3]. In recent years, several studies have been published regarding the optimization of composite laminated structures [4]. This interest of the research community has been driven by the shared property of high-specific strength (or strength/weight ratio), thermal stability, corrosion resistance, and the high impact strength of the composite materials [5]. Hence, to solve such problems, many optimization techniques have been developed.

The Ant Colony Optimization (ACO) algorithm demonstrated a significant performance improvement in solving NP-hard combinatorial optimization problems. The Traveling Salesman Problem (TSP) is a good example of such problems and it is solved using an early version of ACO [6]. The improvements in subsequent ACO algorithms focused on enhancing the algorithm variants to yield better searching and computational performance. As a result of the improved algorithm performance, many new applications of ACO appeared, such as the probabilistic TSP using estimation based ACO in Weiler and Biesinger's [7] work. Gambardella and Dorigo [8] used a hybrid ACO with a novel local search tool to solve the Sequential Ordering Problem (SOP). Zheng and Zecchin [9] introduced a novel approach of ACO to optimize a water distribution system design. Furthermore, the optimization of the space truss design, using improved ACO, was demonstrated by Kaveh and Talatahari [10].

In the field of structural design problems, a well-known optimization problem of Stacking Sequence Design (SSD) of composite laminate has been solved. It is done by using ACO to determine the optimal stacking sequence design for maximum critical buckling load factor or natural frequency [11]. Aymerich and Serra [12] examined the ACO performance in solving the Stacking Sequence Design problem and he compared a modified standard ACO performance with Genetic Algorithm (GA) and Tabu Search (TS). He found that ACO performed much better than GA and TS in terms of solution cost and quality. Rama Mohan Rao [13] presented a hybrid ACO–TS algorithm to optimize the stacking sequence of a composite laminate subjected to bidirectional compression loading. He concluded that ACO is an effective optimization technique if combined with an appropriate local searching tool. Bloomfield and Herencia [14] conducted a comparison study of three meta-heuristics of GA, ACO, and Particle Swarm Optimization (PSO) to determine the optimal stacking sequence composite laminate. Based on the results of this comparison study, ACO was found to outperform GA and PSO algorithms in the field of Stacking Sequence Design (SSD). This remarkable performance of ACO in solving such NP-hard combinatorial optimization is expected where it was designed to solve discrete optimization problems [2].

The literature of SDD optimization is full of valuable contributions that advanced our knowledge in this specific area. This subject of research still evaluates debates on various issues, not only how we could achieve the desired maximum buckling load, but also how we could make this happen efficiently at the lowest possible computational cost. In this regard, selecting the optimization algorithm to solve a SDD problem was not addressed clearly, i.e., most published studies do not explain why a certain algorithm has been selected. Therefore, the selection of an efficient optimization algorithm needs to develop a substantial compromise criterion to determine which MH becomes a cost-effective solution for a SDD optimization problem. Well-defined selection criteria will help to make the appropriate selection in this regard. Additionally, investigating possible improvements of candidate MH exploration and exploitation features could lead to finding a new cost-effective solution that defeats the solution obtained by the original MH.

In the current study, a comprehensive assessment criterion has been developed to assess the performance of the meta-heuristic (MH) solutions in the domain of structural design. Thereafter, the proposed criterion was employed to compare five different variants of Ant Colony Optimization (ACO). This was done based on a well-known structural optimization problem of laminate Stacking Sequence Design (SSD) [1].

The Enhanced HCF-ACO (EHCFACO) variant presented here is as novel optimizer for structural design optimization problems. The standard HCF-ACO variant was never ex-

amined before as a solver of any structural design optimization problems, to the best of our knowledge, even though it has been used successfully to solve other NP-hard combinatorial optimization problems. Furthermore, a comprehensive performance assessment criterion for MHs was used to solve structural design problems. The new criterion introduced new performance measures such as a fitness–distance correlation factor, performance rate, and solution quality. Both the proposed MH and the comprehensive assessment criterion presented here will help engineers optimize their designs more efficiently.

This paper is structured as follows: firstly, it presents a review of standard ACO followed by a brief description of other considered ACO variants; secondly, the new optimization approach is explained in detail; then, the performance assessment criterion is introduced, followed by the numerical experiments section; next, the results of the performance survey are discussed; finally, the paper concludes with a summary of the study's research contributions, limitations, and prospective directions for future research.

2. Ant Colony Optimization Algorithms (ACOs)

Dorigo [6] developed the basic Ant Colony Optimization (ACO) algorithm, or Ant System (AS), which is a metaheuristic approach inspired by the collaborative work of ants in finding the source of food. The ants cooperate to find the best possible path from their colony to the food source, Figure 1. Ants search cooperatively using pheromone trails. During their searching tour, the ants communicate by depositing a certain amount of a substance, called the pheromone, on their way to the food site. The following group of ants tends to follow the paths with higher pheromone concentration. Over time, the less selected paths gradually lose information due to the pheromone evaporation process.

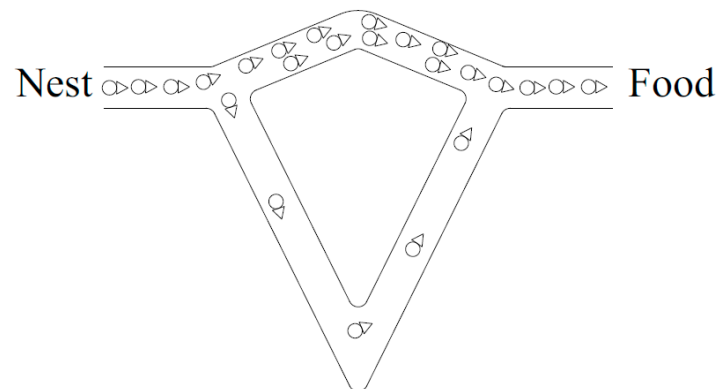


Figure 1. Cooperative search of ants using pheromone trails.

The virtual ants traveling and selecting paths can be interpreted as a probabilistic selection of certain nodes, of which they are part of the solution, in the path based on the pheromone value. The ACO general procedure is illustrated in Algorithm 1.

Algorithm 1 Ant Colony Optimization procedure.

- 1: **Initialization**
 - 2: **While** (termination criteria not satisfied) **Do**
 - 3: Construct Solutions Table by Ants
 - 4: Local Search (optional)
 - 5: Global Pheromones Updating
 - 6: **End** ACO algorithm
-

To understand the mathematical interpretation of ACO, there is a need to go through each step of the ACO procedure shown in Algorithm 1. ACO starts with initial values of the pheromone trail τ_0 , set to a small value for all ant trails as this gives all nodes j of the

design variable i , an equal probability of selection. Next, each ant starts to construct its own solution by applying the rule of selection, which has the following general form:

$$p_{ij}^{(k)} = \frac{\tau_{ij}^\alpha \cdot \eta_{ij}^\beta}{\sum_{j \in N_i^k} \tau_{ij}^\alpha \cdot \eta_{ij}^\beta}, \forall j \in N_i^k \tag{1}$$

$p_{ij}^{(k)}$ represents the probability of selecting the path $i j$ for the k^{th} ant, τ_{ij} is the updated pheromone trail, η_{ij} denotes the value of heuristic information for each feasible solution s , $N_i^{(k)}$ indicates the neighborhood nodes of the k^{th} ant, when it is located at node i and α, β are the amplification parameters for pheromone trails and the influence of heuristic information on the algorithm behavior, respectively [15]. At the end of each tour, all the pheromone trails are updated through two steps of pheromone evaporation and depositing, according to the following formula:

$$\tau_{ij}^{k(t+1)} = (1 - \rho) \cdot \tau_{ij}^{k(t)} + \sum_{k=1}^n \Delta \tau_{ij}^{k(t)} \tag{2}$$

where ρ is the evaporation rate, $\rho \in (0, 1]$, and $\Delta \tau_{ij}^{k(t)}$ is the amount of deposited pheromone by ant $k(t)$ that could be determined as:

$$\Delta \tau_{ij}^{k(t)} = \frac{Q}{L_{k(t)}} \tag{3}$$

where Q is a constant and $L_{k(t)}$ represents the distance traveled by ant $k(t)$. Equation (3) is the basic form of the pheromone trail updating which is used to solve the TSP optimization problem and it could be implemented in more general form:

$$\Delta \tau_{ij}^{k(t)} = \begin{cases} \xi \cdot \frac{f_{worst}}{f_{best}}, & \text{if } i, j \in \text{global best solution} \\ 0, & \text{otherwise} \end{cases} \tag{4}$$

where f_{worst}, f_{best} are the worst and the best values of the objective function f obtained by N ants in tour t and ξ is the global pheromone scaling factor [16]. Eventually, the ACO loop continues until one of the termination conditions is met.

In a SDD optimization problem, the thickness of each ply (equivalent to the distance between the cities in TSP) is assumed to be constant, so the heuristic information value, h , will be constant all over the ant tours t which simplifies the probability of selection, in Equation (1), into:

$$p_{ij}^{(k)} = \frac{\tau_{ij}^\alpha}{\sum_{j \in N_i^k} \tau_{ij}^\alpha}, \forall j \in N_i^k \tag{5}$$

Local search

The procedure of the ACO algorithm includes the option of improving the intensification feature of the ACO algorithm by adding some local search algorithms or movements that could improve the search of the solution neighborhood [2].

2.1. Elitist Ant Colony (EACO)

Gambardella and Dorigo [8] introduced an improved version of the ACO algorithm that uses the elitism strategy. The idea behind this strategy is a reinforcement of the best solution path found once the algorithm is initialized. The rule of pheromone updating for EACO is written as follows:

$$\tau_{ij}^{k(t+1)} = (1 - \rho) \cdot \tau_{ij}^{k(t)} + \sum_{k=1}^n \Delta \tau_{ij}^{k(t)} + e \cdot \Delta \tau_{ij}^{k(t_{best})} \tag{6}$$

$$\Delta\tau_{ij}^{k(t_{best})} = \frac{f_{best}}{\sum_{k=1}^n f_i} \quad (7)$$

where the reinforcement of selection probability of the best path (t_{best}) occurs by adding the value of $e \cdot \Delta\tau_{ij}^{k(t_{best})}$ where e is the weighting parameter and it represents the number of elitist ants [13].

2.2. The Rank-Based Ant Colony Optimization (RBACO)

Bullnheimer and Hartl [17] proposed a new extension of the ACO that enhances the performance of the original EACO by ranking the ants based on their path length. The deposited value of pheromone decreases according to its rank index, m . Moreover, only the best ants, s , will be updated, which prevents the over-concentration of pheromones on local optima paths chosen by other ants. Hence, the pheromone updating rule of RBACO is:

$$\tau_{ij}^{k(t+1)} = \rho \cdot \tau_{ij}^{k(t)} + \sum_{\mu=1}^{\sigma-1} \Delta\tau_{ij}^{\mu} + \sigma \cdot \Delta\tau_{ij}^{k(t_{best})} \quad (8)$$

2.3. Max-Min Ant Colony (MMACO)

Previous ACO algorithms used the strategy of reinforcing only the best-found paths. This strategy could cause the excessive increase of pheromone values on optimal local paths causing all other ants to follow this path. To overcome this drawback, Stützle and Hoos [18] proposed a modified version of ACO that limits the pheromone values to a specific interval, $[\tau_{min}; \tau_{max}]$. In addition, the initialization of pheromone value is set to the upper limit of the pheromone interval, with a small evaporation rate to increase the algorithm search diversification. The pheromone rule is:

$$\tau_{ij}^{k(t+1)} = \rho \cdot \tau_{ij}^{k(t)} + \Delta\tau_{ij}^{k(t_{best})} \quad (9)$$

and $\tau_{ij}^{k(t)} \in [\tau_{min}; \tau_{max}]$.

where $[\tau_{min}; \tau_{max}]$ values are determined by the following formulas:

$$\tau_{max} = \frac{1}{(1-\rho)} \cdot \frac{f_{worst}}{f_{best}}$$

$$\tau_{min} = \frac{\tau_{max} \cdot (1 - \sqrt[n]{P_{worst}})}{(\frac{n}{2} - 1) \cdot \sqrt[n]{P_{worst}}}$$

where p_{best} denotes the probability of the best solution, it has a value greater than 0, while n represents the number of ants.

2.4. Best-Worst Ant Colony (BWACO)

Zhang and Wang [19] presented BWACO as an extension of MMACO, where the algorithm exploitation capability benefits from both best and worst solutions. During the search tour, the pheromone trail update uses the positive return of the best solution and the negative one generated by the worst solution. The pheromone updating rule can be written as:

$$\tau_{ij}^{k(t+1)} = \left[\rho \cdot \tau_{ij}^{k(t)} + \Delta\tau_{ij}^{k(t_{best})} \right]_{\tau_{min}}^{\tau_{max}} - \lambda \cdot \Delta\tau_{ij}^{k(t_{worst})} \quad (10)$$

where λ is a coefficient that has value within $[0, 1]$ interval and it could be noticed that BWACO became MMACO if $\lambda = 0$.

2.5. Hyper-Cube Framework ACO (HCFACO)

The different algorithms of ACO build a limited hyperspace of the pheromone values. The Hyper-Cube Framework of ACO algorithms, proposed by Blum in 2001, generates a binary convex hull hyperspace from pheromone values for the feasible solutions. In

other words, the values of the pheromone vector, $\tau = [\tau_1, \tau_2, \tau_3, \dots, \tau_n]$, are limited to the interval $[0, 1]$, and this is carried out by changing the pheromone update rule. The following formula expresses the rule of pheromone updating in HCFACO:

$$\tau_{ij}^{k(t+1)} = (1 - \rho) \cdot \tau_{ij}^{k(t)} + \rho \cdot \sum_{k=1}^n \Delta \tau_{ij}^{k(t)} \quad (11)$$

where:

$$\Delta \tau_{ij}^{k(t_{best})} = \frac{f_{best}}{\sum_{k=1}^n (f_i)}, \text{ and } n \text{ is the number of ants follow the same best path.}$$

HCFACO algorithms overcome the undesirable problem of different behavior of standard ACO algorithms when the same objective function is scaled, which affects the algorithm robustness. Moreover, it reduces the search effort and improves the algorithm search diversification [2]. Lastly, it is worth mentioning that the HCF update rule is not limited to standard ACO algorithm (or Ant System (AS)) as it can also be used with MMACO, where the maximum and minimum limits of MMACO pheromone trail are set to be 0 and 1, respectively [20].

3. Enhanced Hyper-Cube ACO Algorithms

Dorigo experimentally observed that using local search techniques can improve the overall performance of the ACO [2,21]. Local search can be carried out by hybridizing the ACO with local search algorithms such as Tabu search or using permutation operators to explore the solution neighborhood [12,22]. The commonly used operators in SSD optimization problem are two-points permutation and swap. Two-points permutation means selecting two bits in the solution string and reversing the order of the bits in between [1]. The swap operator is used to switch the position of two randomly selected bits of the solution string [23].

The HCFACO algorithms presented here adopted two other permutation operators to perform the algorithm enhancement. The first operator is called a bit flip (also known as single point mutation), which is used successfully with Permutation Genetic Algorithm [1]. The second operator is inspired by using one of the Tabu Search movements named the insertion [2]. The proposed Enhanced HCFACO procedure is listed in Algorithm 2.

The Enhanced HCFACO Algorithms starts by defining the standard ACO parameters such as the maximum number of iterations ($Iter_{max}$), number of ants (n_{Ants}), number of design variables (N_V), the initial pheromone trail (τ_0), and evaporation rate (r). In addition, the solution convergence rate counter is imposed [$Iconv_{max}$] and its value determines whether the convergence rate is slow or fast. When the ACO loop starts, all solution edges have the same deposited pheromone trail τ_0 , which gives all nodes the same probability of selection to be a part of the feasible solution. The artificial ants, $k = 1 : n_{Ants}$, start building the solution table, $S_i(n_{Ants}, N_V)$, by randomly choosing a node d_i on their way to build the solution vector $S_i(k, N)$. Next, the evaluation of the solutions table is carried out by calling the objective function, and the obtained values are stored in $f(ige, S_i(1 : n_{Ants}, N_V))$ matrix. The best solution of the stacking sequence design has the maximum value of the objective function listed in $f(ige, S_i)$ matrix of the current iteration. The best solution of each iteration ige is stored in the best solution matrix $S^*(ige)$. Thereafter, the global pheromone trail update is performed as described in the Hyper-Cube Framework of ACO in Equation (11).

The local search actions are enforced as soon as the best solution of the current tour is determined. Following this, a comparison of the generated solutions with the best solution obtained so far is made. The best solution matrix is then updated if any improvement is detected. Finally, the HCFACO loop continues until the termination criteria are met. The global optimal solution is determined as the best solution matrix member with the maximum value of the objective function.

Algorithm 2 Enhanced HCFACO procedure

(f) objective function, (N_V) number of the design variables, (n_{Ants}) number of ants, (N_n) length of the design domain vector, ($Iter_{max}$) maximum number of iterations, (τ_0) initial pheromone trail, (ρ) evaporation rate, ($Iconv_{max}$) maximum convergence rate.

Input: (f^*) the optimal solution value, (S^*) optimal solution vector, (ige^*) number of optimal solution iterations.

Output:

Initialization:

1: $ige = 0$, initialize the EHCFAO main loop counter.
 2: $S_0(1 : n_{Ants}, N_V, \tau_0)$, generate initial population.
 3: $f_0(1 : n_{Ants}, S_0)$, evaluate the initial population.
 4: $\tau(N_V, N_n)$, update global pheromone trail.
 5: **While** (termination criterion not satisfied) **Do:**
 6: $ige = ige + 1$, update loop counter.
 7: $S_{ige}(1 : n_{Ants}, N_V, \tau) = RouletteWheelSelection(\frac{\tau_i}{\sum \tau_i})$, construct the new solution table.
 8: $f_{ige}(1 : n_{Ants}, S_{ige})$, evaluate the objective function of the new solution.
 9: $\left. \begin{aligned} [f_{ige}^* i_s] &= \max(f_{ige}) \\ S_{ige}^* &= S_{ige}(i_s, :) \end{aligned} \right\}$, determine the current optimal solution.

- **Local search:**

10: $\left. \begin{aligned} S_{ige}^{ins}(S_{ige}^*, N_V) &= insertion(S_{ige}^*, randi(N_V), randi(N_V)) \\ f^{ins} &= f(1 : S^{ins}) \end{aligned} \right\}$, insertion movement.
 11: $\left. \begin{aligned} S_{ige}^{b_flip}(S_{ige}^*, N_V) &= bitflip(S_{ige}^*, randi(N_V)) \\ f^{b_flip} &= f(1 : S^{b_flip}) \end{aligned} \right\}$, bit flip movement.
 12: $\left. \begin{aligned} [f_{ige}^* i_{ls}] &= \max[f_{ige}^* f^{b_flip} f^{ins}] \\ S_{ige}^* &= S_{ige}(i_{ls}, :) \end{aligned} \right\}$, update the current optimal solution.

- **Update global optimal solution:**

13: $f^* = \max(f_{1:ige}^*)$

- **Global pheromone updating:**

14: $\tau^{(ige+1)} = (1 - \rho) \cdot \tau^{ige} + \rho \cdot \sum_{k=1}^n \Delta \tau_{ij}^{S_{ige}^*}, \Delta \tau_{ij}^{S_{ige}^*} = \frac{f^*}{\sum_{k=1}^n (f_i)}$

- **Convergence check:**

15: $I_{conv} = sum(find(f_{1:ige}^* = f^*))$, number of global optimal solution occurrences.
 16: If $I_{conv} > Iconv_{max}$, end, imposed convergence rate has been achieved.
 17: If $\left| \max(f_{1:ige}^*) - \min(f_{1:ige}^*) \right| \approx 0$, end, all artificial ants followed the same path.
 18: **End while**
 19: **Print** f^*, S^*, ige^* .
 19: **End**

4. Performance Evaluation

The time required by an algorithm to find the global optima is widely used to evaluate its performance [24]. However, a single performance measure cannot reflect the effectiveness of the algorithm in exploring the design space or determining solution quality. In the current study, three different groups of performance measures have been applied to ensure a fair evaluation of the proposed algorithm.

4.1. Computational Effort

In addition to the elapsed time, literature has shown that other measures can be used to measure computational effort. The first is the price PS , which is defined as the number of objective function evaluations within a search run and reflects the computational cost of

the search process. The second measure is practical reliability (PR) and it is defined as the percentage of runs that achieve practical optima (PO), at a specific run. Practical optima is defined as the solution with 0.1% error in the best possible solution [1]. The last is the normalized price nPS , which is defined as the ratio of price and practical reliability [1,25,26]. Finally, the performance rate measure, $Prate$, is also considered to link the computation effort with the number of function evaluations [24].

$$P_{rate} = \frac{\text{number of successful runs}}{\left(\begin{matrix} \text{number of} \\ \text{function} \\ \text{evaluations} \end{matrix} \right) \cdot \left(\begin{matrix} \text{total number} \\ \text{of runs} \end{matrix} \right)} \quad (12)$$

4.2. Solution Quality

The solution quality of an algorithm can be measured by determining the absolute error between the current solution and the best-known global solution [1,25,26].

$$Q = \left| \frac{S^* - S_{opt}}{S_{opt}} \right| \cdot 100 \quad (13)$$

4.3. Fitness Landscape Analysis

The design space of a combinatorial optimization problem can significantly affect the search performance of an algorithm. The notion of fitness landscape appeared in literature as an answer to the question of “what does the design space look like?”. The fitness landscape is defined by the feasible solutions set, the objective function (fitness), and the structure of the solution neighborhood. To find the connection between the fitness landscape and the problem hardness, Jones and Forrest [27] introduced a Fitness Landscape–Distance Correlation (FDC) to determine the hardness of optimization problems to be solved using Genetic Algorithm (GA). The distance mentioned here is defined as the number of movements that should be imposed on a Solution S_i to eliminate dissimilarity with the optimal solution S_{opt} . The proposed correlation by Jones is computed using the correlation factor, r :

$$r(F, D) = \frac{CFD}{\sigma_F \cdot \sigma_D} \quad (14)$$

where CFD indicates the $Cov(F, D)$ and σ_F, σ_D are the standard deviation of F and D , respectively. The values of the correlation coefficient r are limited to interval $[-1, 1]$ where negative values are desirable for maximization and indicate better searching performance. Finally, using the scattering of fitness versus the distance to the global optima can reveal valuable information about FDC of the optimization problem solved by an algorithm [18,26].

5. Numerical Experiments

To demonstrate the performance of the new approach, we selected a well-known NP-hard combinatorial optimization problem in the field of composite laminated structures. The optimization objective is maximizing the critical buckling load of composite laminated plate exposed to bidirectional compression loading. The decision variables are the fiber orientation of each composite layer (lamina) which form the optimal stacking sequence of the laminate (a group of layers). To employ ACO as an optimization algorithm for a SSD optimization problem, there is a need to understand specific problem characteristics such as solution representation, constraints, and objective function formulation. In meta-heuristic algorithms, the solution (stacking sequence) takes the form of a bit string that consists of a combination of plies with the available angle fiber orientations (e.g., 0° , $\pm 45^\circ$ and 90°). The different solutions have integer coding with 1, 2, and 3 numbers, which represent the three possible fiber orientations, respectively. For instance, the laminate with $[2132231]_s$ stacking sequence describes the laminate of $[\pm 45, 0_2, 90_2, 45, \pm 45, 90_2, 0_2]_s$ fiber orientations.

The simplicity of using an integer representation along with significant performance gains made it the most widely used method in meta-heuristic optimization algorithms for composite laminated design. The buckling load factor λ_b for simply supported rectangular laminated plate subjected to bi-axial loading is determined as follows:

$$\lambda_b(p, q) = \pi^2 \frac{D_{11}(\frac{p}{a})^4 + 2(D_{12} + 2D_{66})(\frac{p}{a})^2 + D_{22}(\frac{q}{b})^4}{(\frac{p}{a})^2 N_x + (\frac{q}{b})^2 N_y} \tag{15}$$

where D_{ij} denotes the bending stiffness, N_x is the axial loading in x-direction, N_y is the axial loading in y-direction, p and q are the number of half waves in x, y directions. The critical buckling load factor λ_{cb} is defined as the minimum obtained value of $\lambda_b(p, q)$. The critical values of p and q are linked to different factors such as laminate material, the number of plies, loading conditions, and the plate aspect ratio. In uniaxial loading and a simply supported plate, the critical buckling load occurs when $p = 1$ whereas in biaxial the critical buckling loads, it needs to be determined as the minimum value of $\lambda_b(p, q)$ [1,13]. Finally, the constraints in stacking sequence optimization with constant laminate thickness t could be imposed as follow:

- Symmetry constraint is enforced by optimizing half of the laminate.
- Balancing constraint is enforced by selecting θ_2 for the standard fiber orientation set of $0, \pm 45$ and 90 .
- Only $N/4$ ply orientations are needed to describe laminate as a result of balancing constraints.
- Contiguity constraint is handled by using the penalty parameter (β).
- The critical buckling load factor objective function f_{obj} could be formulated as:

$$f_{obj} = (1 - \beta) \cdot \max(\lambda_{cb}(p, q)) \tag{16}$$

To compare the performance of the proposed algorithm alongside the other ACO algorithms, we implemented all the algorithms presented here using MATLAB R2019b software. The benchmarking problem from the literature of Stacking Sequence Design optimization is accredited to Le Riche and has been used by previous studies [1,23]. The original problem describes a simply supported plate subjected to an in-plane biaxial loading as shown in Figure 2.

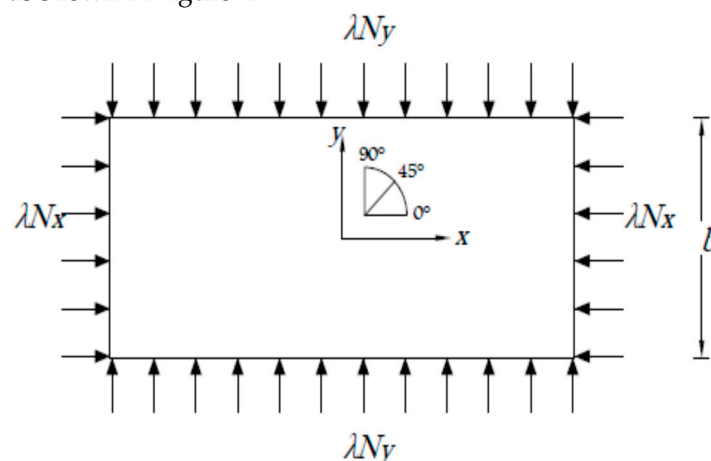


Figure 2. Simply supported plate subjected to biaxial loading.

The thickness of each ply t_i is assumed constant, and the ply orientations are limited to $0, \pm 45$ and 90 sets of angles. The number of plies N_L is constant. The required properties, dimensions, and loading conditions are listed in Tables 1 and 2. The objective function is set to maximize the critical buckling load. The constraints are integrated into the solution (e.g., balanced laminate, symmetrical, etc.). The implemented ACO algorithms were executed on the same computer for the same number of experiments; $N_{exp} = 200$. This number was

used to overcome the stochastic behaviour of meta-heuristic algorithms [1]. Furthermore, this number of experiments was conducted over ten different random generating seeds of 301, 2, 50, 75, 111, 200, 167, 225, 11, and 25. Then, the average of the performance measures values was used in the comparison of different ACO algorithms.

Table 1. Graphite-epoxy lamina's properties.

$E_1(GPa)$	$E_2(GPa)$	$G_{12}(GPa)$	ν_{12}
127.59	13.03	6.41	0.3

Table 2. Graphite-epoxy lamina's geometrical and loading data.

N_L	$t(mm)$	$a(mm)$	$b(mm)$	$N_x(N/m)$	N_x/N_y
64	0.127	508	254	175	1

Lastly, all ACO algorithms were examined at two different levels of convergence rate, slow and fast. The slow rate forces the algorithm searching loop to stop after 56 iteration without improvement, while the fast rate needs just 10 iterations to be terminated [1].

5.1. ACO Parameters Setting

To ensure a fair assessment of the ACO algorithms performance, the following standard ACO parameters were assumed for all implemented algorithms: number of Ants $n_{Ants} = 25$, the maximum number of iterations $Iter_{max} = 1000$, evaporation rate $r = 0.1$, the parameter of the pheromone trail relative importance $\alpha = 1$, and initial one trail $\tau_0 = 0.004$ (except for MMACO and BWACO algorithms were $\tau_0 = 1$). Best solution probability was $P_{best} = 0.05$ for MMACO and BWACO algorithms and lastly the coefficient of worst solution pheromone trail was $\lambda = 0.6$ for BWACO algorithm only.

5.2. Termination Criteria

All algorithms stop as soon as one of the following conditions are satisfied:

- If there is no improvement in the solution after 10 or 56 iterations.
- If the number of iterations exceeds 150 and the best solution is equal to the worst solution (means all artificial ants following the same path).
- If a maximum number of iterations has been generated.

6. Results and Discussion

The case study described in the previous section was optimized using nine different algorithms: standard ACOA, EACO, RBACO, MMACO, BWACO, HCF/EHCF for both ACO and MMACO algorithms. Analysis of the algorithm's performance will be divided into two parts. First, the performance of ACO algorithms with Hyper-Cube Framework will be assessed. The second part is dedicated to the comparison of EHCFAACO algorithm with the rest of ACO algorithms.

6.1. Hyper-Cube Framework ACO Algorithms Results Analysis

Referring to Section 2.5, the Hyper-Cube Framework (HCF) can be applied for both versions of the standard ACOA and MMACO. Hence, this part of the analysis is devoted to determining which version of both ACO algorithms, with HCF and EHCF, could exhibit a better performance. The performance measures for the original ACOA, MMACO, HCFACO, HCFMMACO, EHCFAACO, and EHCFAACO are listed in Table 3. The performance values listed in Table 3 reveal that applying HCF to the ACOA positively affected the overall performance of ACO. The average practical reliability increased by 22–56% and the normalized price declined from 51.17 to 36.91 for the fast convergence rate and from 181.12 to 94.24 for the slow one. The performance rate doubled at the slow rate while

remaining the same for the fast. The FDC correlation coefficient r decreased slightly for both levels of convergence.

Table 3. The performance measures of Hyper-Cube Framework ACO algorithms.

Performance Measure	Convergence Rate	ACOA	HCFACO	EHCFAO	MMACO	HCFMMACO	EHCMMACO
Elapsed time, t_s , min	Slow	0.87	0.01	2.16	1.12	0.96	1.34
	Fast	3	4.15	6.86	4.62	6.68	4.39
Reliability %	Slow	35.71	77.1	89.6	16	13.37	54.36
	Fast	36.45	93.15	98.95	87.7	91.05	98.25
Normalized Price, nP_s	Slow	51.17	36.91	28.92	152.23	169.26	52.2
	Fast	181.12	94.24	81.49	118.53	117.3	98.25
Performance Rate, P_{rate}	Slow	0.0196	0.0272	0.0347	0.0068	0.0063	0.0206
	Fast	0.0056	0.0106	0.0123	0.0088	0.0090	0.0106
Quality %	Slow	99.69	99.93	99.96	98.28	98.57	99.81
	Fast	99.69	99.97	99.98	99.96	98.57	99.98
Fitness-Distance Correlation	Slow	−0.80	−0.67	−0.79	−1	−1	−0.82
	Fast	−0.84	−0.71	−0.73	−0.81	−0.77	−0.75

Further improvement of HCFACO performance was acquired when the proposed local search movements were imposed. The average practical reliability became more than twice of the standard ACO, and the normalized price decreased more to hold at 28.92 instead of 51.17 and 81.49 instead of 181.2 for both convergence rates. The performance rate was slightly increased, and the FDC correlation coefficient r was partially improved. On the contrary, HCFMMACO performed poorly compared to the original MMACO. However, the performance of MMACO improved when the Enhanced HCF was applied, but the computational effort became more costly. Based on these results, we conclude that EHCFAO delivers an inexpensive solution with significant performance. Eventually, the solution convergence of the algorithms mentioned above was plotted, Figure 3: solution convergence of ACO-MMACO and their Hyper-Cube Framework variants, for the same selected experiment (seeds = 75).

6.2. Other ACO Algorithms Results Analysis

All ACO algorithms successfully found the best-known value of the maximum critical buckling load factor, λ_{cb} . A different sample of optimal SSD obtained by different ACO algorithms is listed in Table 4. The solution convergence of EACO, RBACO, and BWACO for a selected experiment (seeds = 75) is graphically illustrated in Figure 4.

It is observed from Table 4 that the optimal stacking sequence design followed the same pattern of switching between two groups of 90_2 and ± 45 fiber orientations which confirm the results of previous studies [1,23].

On the other hand, the solution convergence plot in Figure 4 illustrates that both RBACO and BWACO algorithms develop gradual search trends on their way to the optima whereas EACO and EHCFAO algorithms smoothly converge to the global optima. Further, the numerical experiments confirm the fluctuation of ACO algorithms in finding the global optimal solution due to their stochastic nature, as illustrated in Figure 5.

According to the introduced performance assessment criteria in Section 5, the average values of different performance measures of reliability, performance rate, solution quality, normalized price, and searching effort coefficient were determined for EACO, RBACO, BWACO at fast and slow convergence rates. These results, alongside the EHCFAO results, are plotted in Figures 6–12 to provide a sensible comparison of the performance evaluation of the proposed algorithm with other ACO algorithms. The average practical reliability of the algorithms is introduced in Figure 6. Both EACO and RBACO algorithms show low practical reliability values, with 10% and 8%, respectively, while BWACO presented

better value at the slow rate of convergence, but it poorly performed at the fast rate. The EHCFACO algorithm exhibited significant reliability values of 89.6–98.95%. Furthermore, it demonstrated the highest performance rate measure (0.012–0.035), see Figure 9. The solution quality results of the algorithms are summarized and depicted in Figure 10, which reveals that all ACO algorithms produce an excellent solution quality for this particular case study.

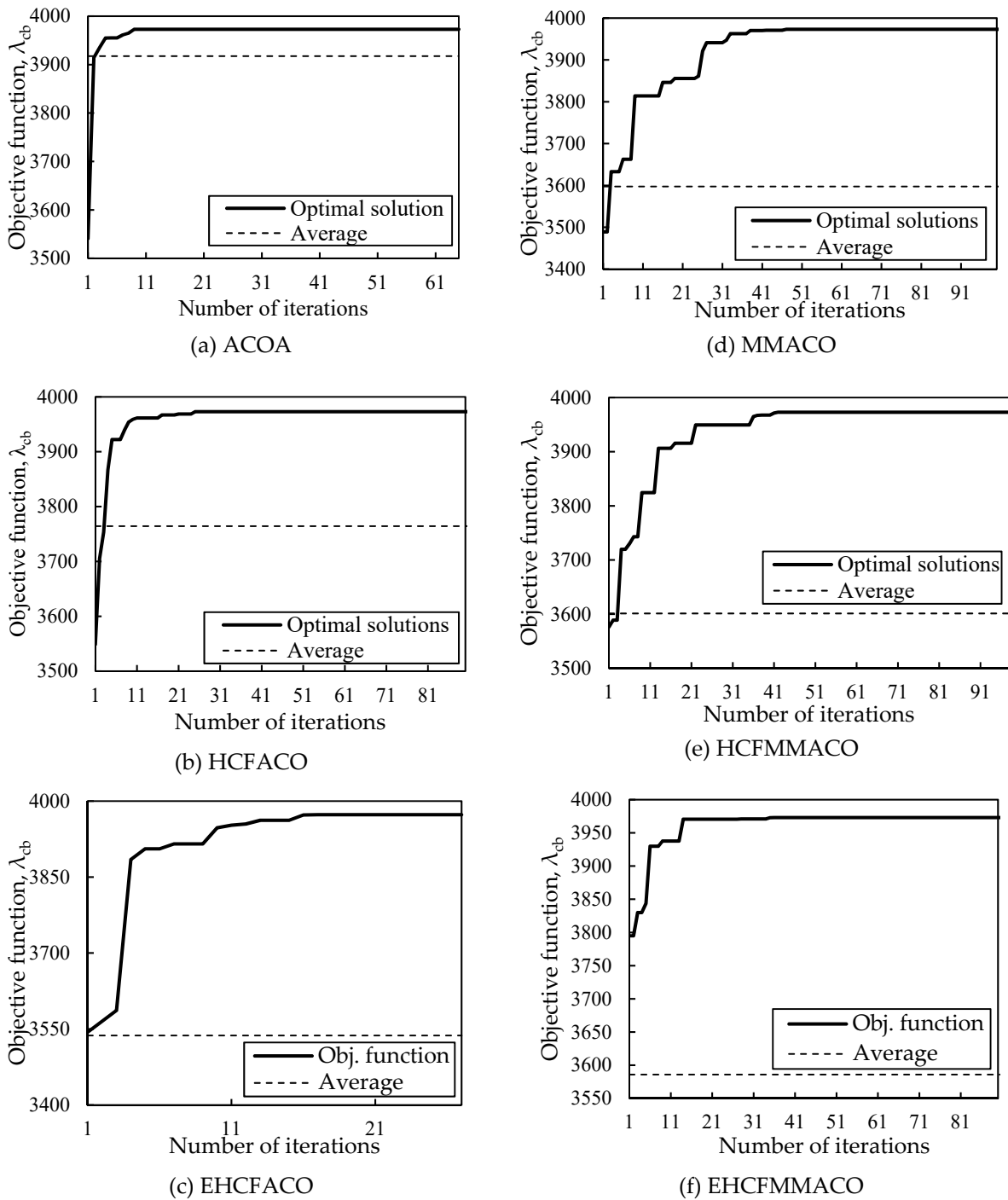


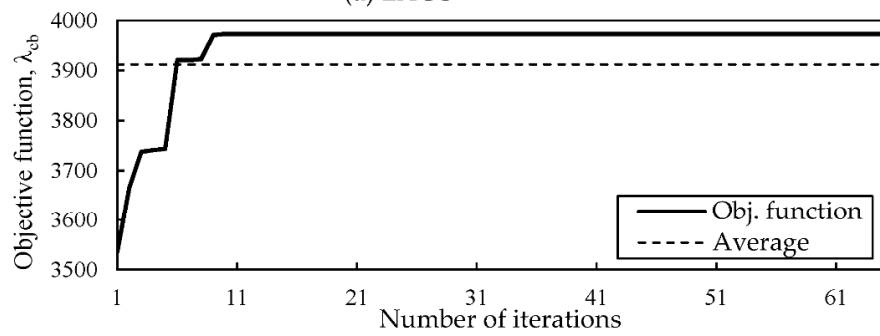
Figure 3. Solution convergence of ACO-MMACO and their Hyper-Cube Framework variants.

Table 4. The optimal stacking sequence for 64 ply laminates subjected to biaxial loading without contiguity constraint ($N_y = N_x = 1$ and $a/b = 2$).

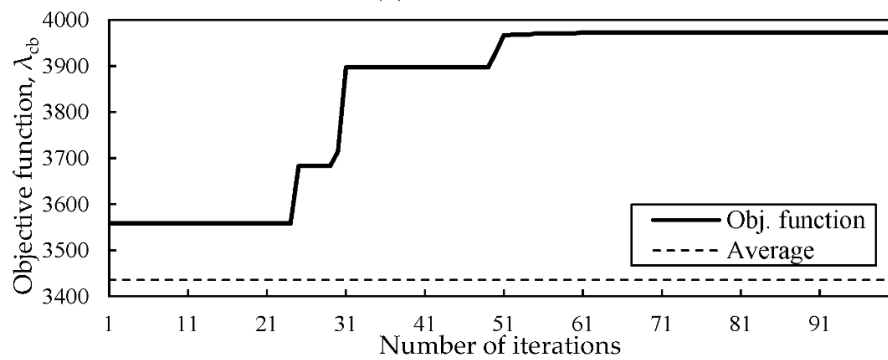
Algorithm	Optimal Stacking Sequence Design	Critical Buckling Load Factor, λ_{cb}
EACO	$[2333332333323333]_s$ $[\pm 45^\circ / 90^\circ_{10} / \pm 45^\circ / 90^\circ_8 / \pm 45^\circ / 90^\circ_8]_s$	3973.01
RBACO	$[233332333333332]_s$ $[\pm 45^\circ / 90^\circ_8 / \pm 45^\circ / 90^\circ_{18} / 45^\circ]_s$	
BWCACO	$[33332323222222]_s$ $[90^\circ_8 / \pm 45^\circ / 90^\circ / \pm 45^\circ / 90^\circ / \pm 45^\circ / 90^\circ \pm 45_2]_s$	
EHCFAO	$[33332223222222]_s$ $[90^\circ_{10} / \pm 45^\circ_2 / 90^\circ / \pm 45^\circ_3 / 90^\circ / \pm 45^\circ_8]_s$	



(a) EACO



(b) RBACO



(c) BWACO

Figure 4. Solution convergence of EACO, RBACO, and BWACO.

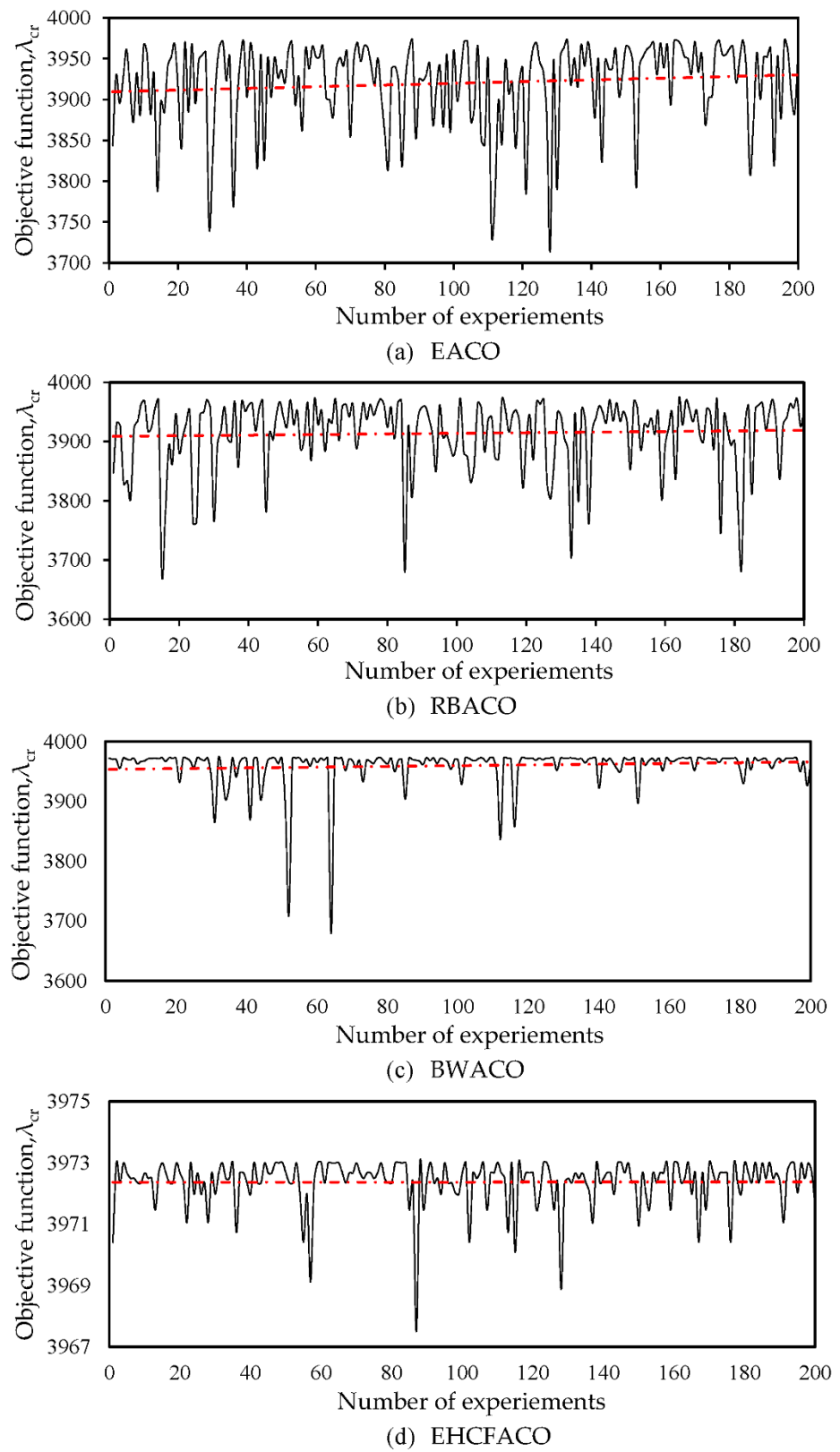


Figure 5. Critical buckling load factor vs. number of experiments.

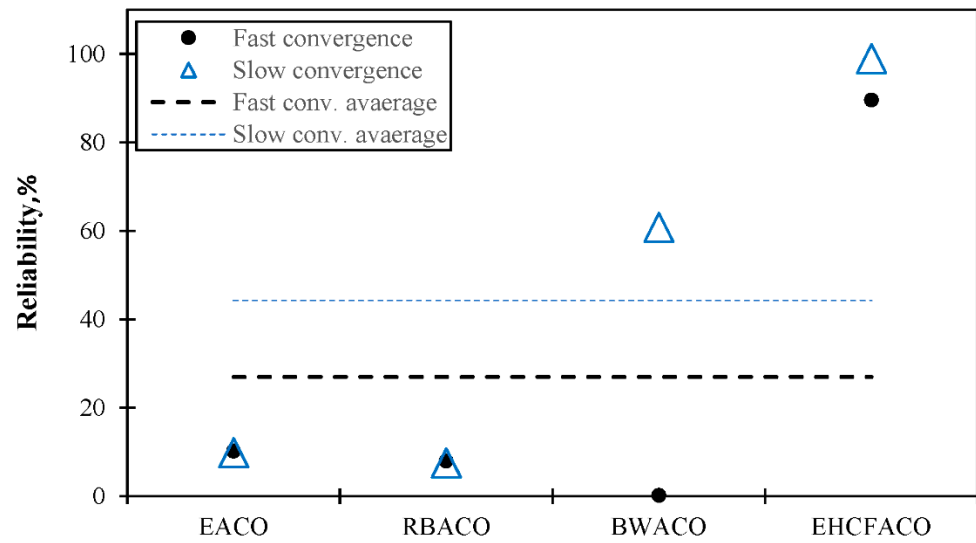


Figure 6. Reliability of EACO, RBACO, BWACO, and EHCFAO algorithms.

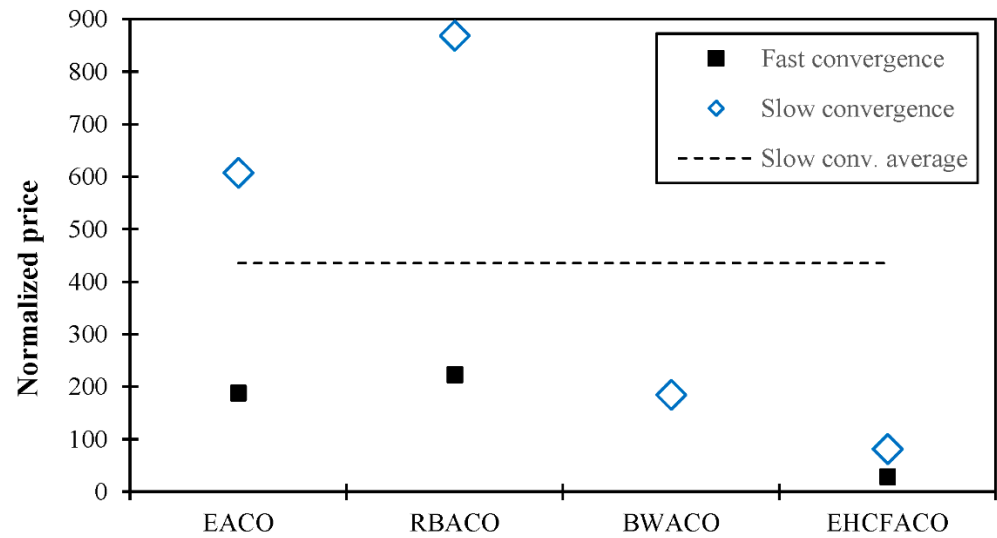


Figure 7. Normalized price of EACO, RBACO, BWACO, and EHCFAO algorithms solutions.

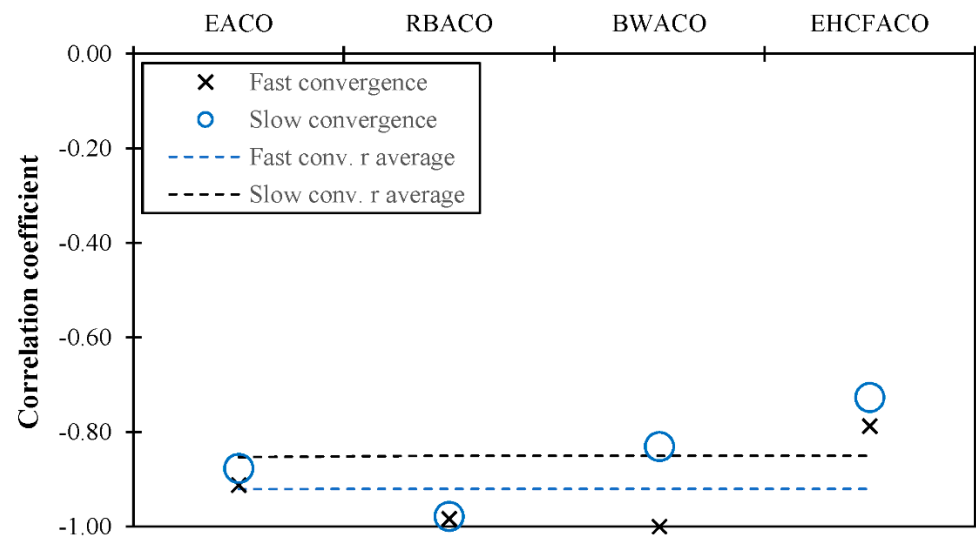


Figure 8. Correlation coefficient of EACO, RBACO, BWACO, and EHCFAO algorithms.

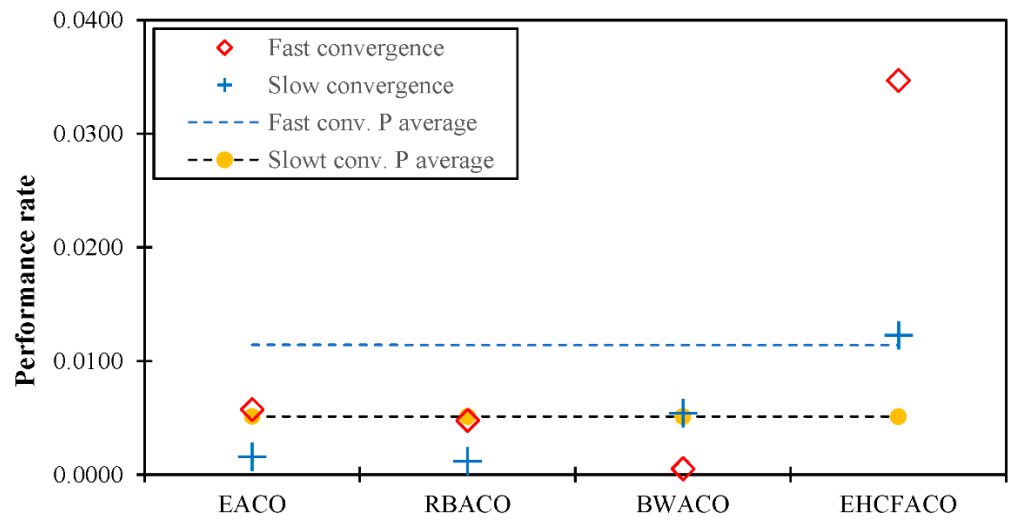


Figure 9. Performance rate of EACO, RBACO, BWACO, and EHCFCACO algorithms.

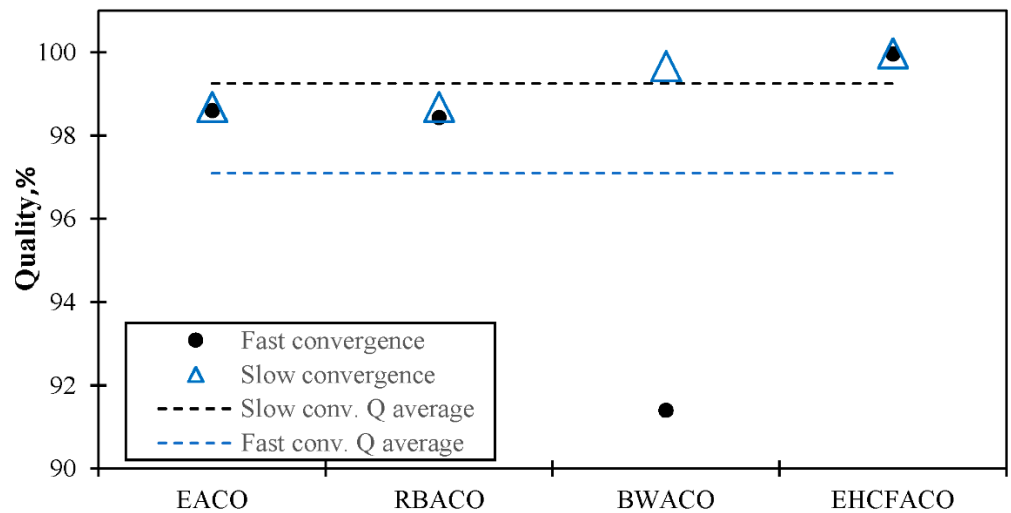


Figure 10. Solution quality of EACO, RBACO, BWACO, and EHCFCACO algorithms.

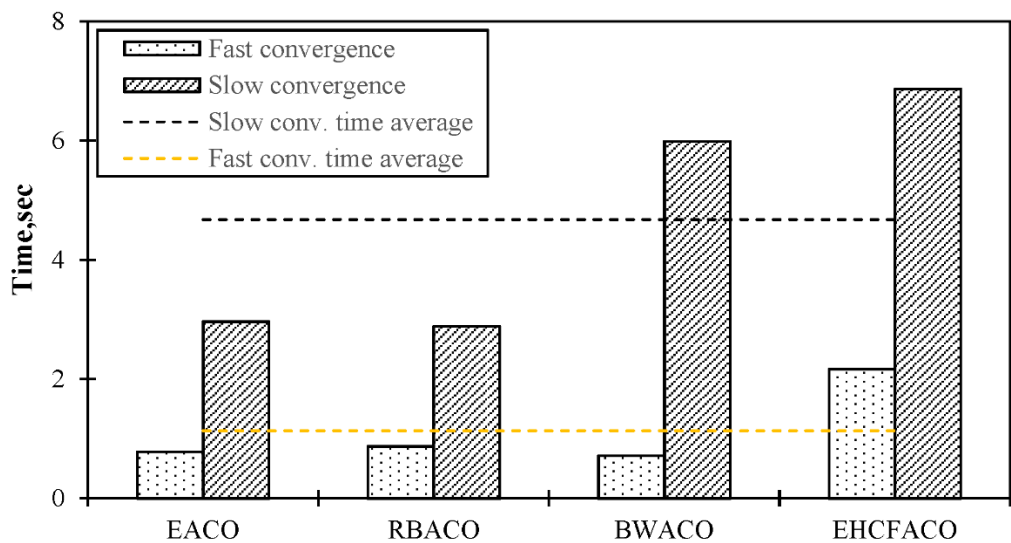


Figure 11. Elapsed time of EACO, RBACO, BWACO, and EHCFCACO algorithms to find the optimal solution.

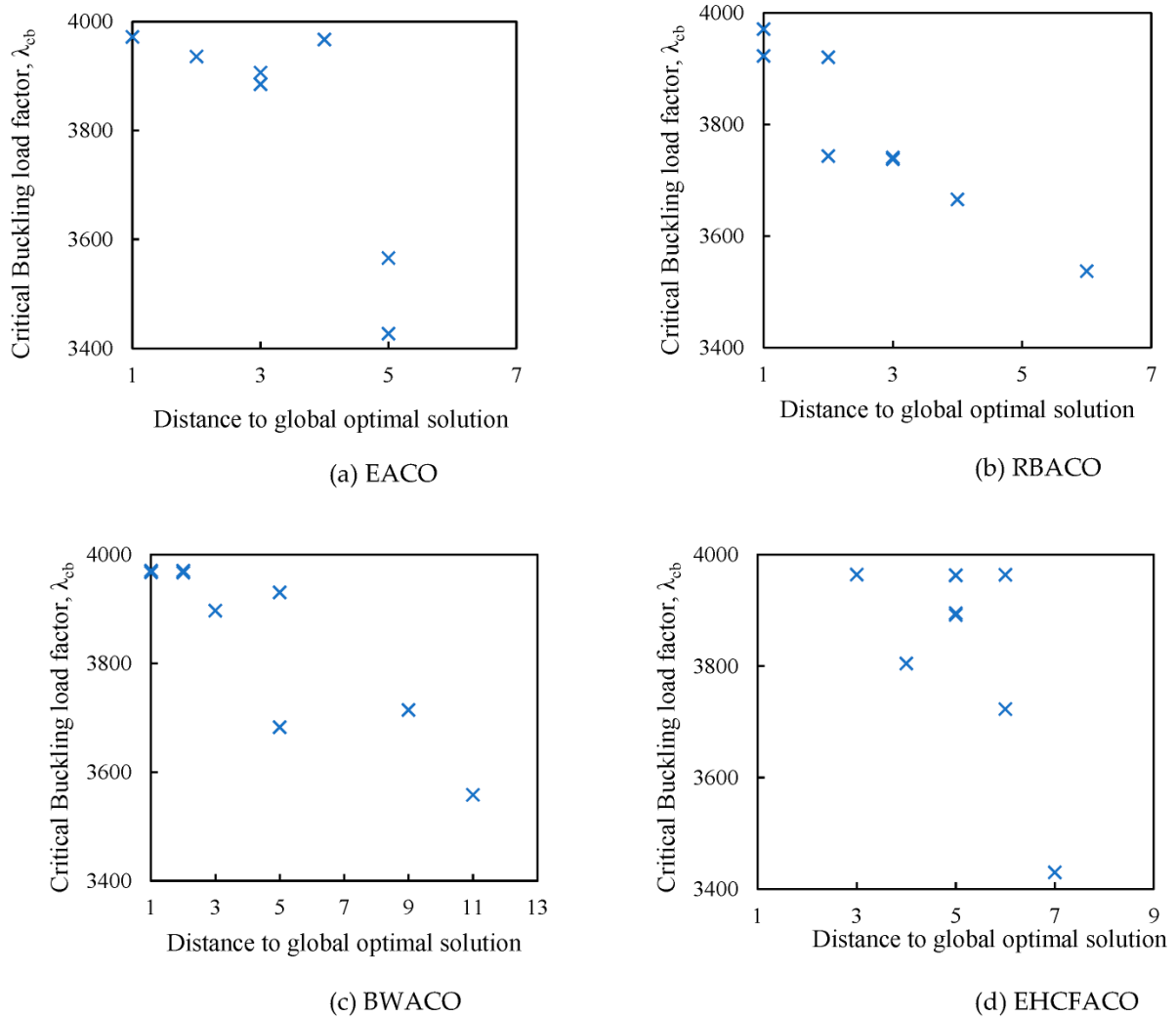


Figure 12. Fitness vs. distance to global optimal solution of ACO algorithms.

In terms of solution cost, the normalized price results plotted on the scatter chart are illustrated in Figure 7. As mentioned before, the normalized price measure reflects the balance between the solution cost and the reliability. Thus, it is quite clear that EHCFAO outperformed other ACO algorithms. BWACO comes second at a slow convergence rate, whereas RBACO and EACO deliver a costly solution.

The numerical experiments results show that ACO variants with high reliability need more computational time to find the global optimal solution while the number of iterations required to find it is low. The Fitness-Distance Correlation values demonstrated the complexity of such a NP-hard optimization to be solved where ACOs with high reliabilities have low r values. Eventually, applying the Hyper-Cube Framework (HCF) to standard ACO has a significant influence on the overall performance of ACO. In addition, imposing local search movements, as an enhancement of exploitation effort, helped HCFACO to deliver a cost-effective solution.

7. Conclusions

A comprehensive MHs performance assessment criterion was developed in the current study. In addition to the computation time and statistical matrices, several new measures such as the practical reliability, price (computational cost), normalized price, performance rate, solution quality, and fitness landscape analysis were included. Additionally, two different convergence rates were imposed on examining the MHs at both the slow and the fast rates. The reproducibility of the numerical experiments' results was also considered

within the procedure of MH assessment. Thereafter, the proposed criterion was employed to compare five different variants of Ant Colony Optimization (ACO). The proposed measures demonstrated a comprehensive assessment of the compared ACOs performance. The initial results of the comparison study revealed that the Hyper Cube Framework (HCF) ACO variant outperforms the others. Consequently, an investigation of further improvement led to introduce an enhanced version of HCFACO (or EHCFAACO). The new variant has advanced intensification features that use both the insertion and bit-flip movements to enhance the local search effort. Eventually, the EHCFAACO variant was compared with other ACO variants and it exhibited a significant performance.

Based on the findings of the proposed assessment scheme presented through this research work, it was noticed that MHs with high-reliability solutions need more time to find the global optima but need a relatively small number of computation iterations. This observation led to another important remark, i.e., the MHs that show good performance are hardly exploring the design space. It is the case where their Fitness-Distance Correlation, r , and figures become lower than the other MHs. The difference in r values was noticed, but it does not negatively impact the overall performance of the designated MHs, except in increasing the computational time. Furthermore, it was observed that applying the Hyper-Cube Framework (HCF) to standard ACO has a significant influence on the overall performance of ACO. Moreover, imposing the local search movements, as an enhancement of exploitation effort, helped HCFACO to deliver a cost-effective solution. These improvements, in ACO performance, are in line with the suggestions made by the previous studies.

The MHs literature is full of studies that approve the noticeable impact of the parameters' fine-tuning on the MHs performance; for all examined MHs, the parameters have been assumed constant. This perceived ignorance of such an aspect refers to the associated computation cost of this repetitive investigation, and such an exhaustive practice would diminish the scope of this paper significantly.

The current study allows engineers the access to an efficient variant of ACO MH to optimize their structure designs at low computational cost. Furthermore, the comprehensive assessment criterion makes it easier for the engineer to decide between any other obtainable MHs. Eventually, the obtained data of the numerical experiments conducted in this paper are available to other researchers to assess and reproduce the results obtained in Supplementary Materials.

Supplementary Materials: The following are available online at <https://www.mdpi.com/article/10.3390/a14100286/s1>, Data sheets: Numerical experimental data.

Author Contributions: Conceptualization, A.A. and T.-M.D.; methodology, A.A.; software, A.A.; validation, A.A., T.-M.D., and N.V.L.; writing—original draft preparation, A.A.; writing—review and editing, T.-M.D.; supervision, N.V.L.; project administration, T.-M.D. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Not applicable.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Le Riche, R.; Haftka, R.T. Optimization of laminate stacking sequence for buckling load maximization by genetic algorithm. *AIAA J.* **1993**, *31*, 951–956. [[CrossRef](#)]
2. França, P.; Sosa, N.M.; Pureza, V. An adaptive tabu search algorithm for the capacitated clustering problem. *Int. Trans. Oper. Res.* **1999**, *6*, 665–678. [[CrossRef](#)]
3. Gholizadeh, S.; Milany, A. An improved fireworks algorithm for discrete sizing optimization of steel skeletal structures. *Eng. Optim.* **2018**, *50*, 1829–1849. [[CrossRef](#)]

4. Nguyen, P.D.; Vu, Q.-V.; Papazafeiropoulos, G.; Thiem, H.T.; Vuong, P.M.; Duc, N.D. Optimization of Laminated Composite Plates for Maximum Biaxial Buckling Load. *VNU J. Sci. Math. Phys.* **2020**, *36*. [[CrossRef](#)]
5. Bouvet, C. *Mechanics of Aeronautical Composite Materials*; John Wiley & Sons: New York, NY, USA, 2017.
6. Dorigo, M. *Ant Colony OPTIMIZATION—New Optimization Techniques in Engineering*; Onwubolu, G.C., Babu, B.V., Eds.; Springer: Berlin/Heidelberg, Germany, 1991; pp. 101–117.
7. Weiler, C.; Biesinger, B.; Hu, B.; Raidl, G.R. Heuristic Approaches for the Probabilistic Traveling Salesman Problem. In *International Conference on Computer Aided Systems Theory*; Springer: Cham, Switzerland, 2015.
8. Gambardella, L.M.; Dorigo, M. An Ant Colony System Hybridized with a New Local Search for the Sequential Ordering Problem. *INFORMS J. Comput.* **2000**, *12*, 237–255. [[CrossRef](#)]
9. Zheng, F.; Zecchin, A.; Newman, J.P.; Maier, H.; Dandy, G.C. An Adaptive Convergence-Trajectory Controlled Ant Colony Optimization Algorithm with Application to Water Distribution System Design Problems. *IEEE Trans. Evol. Comput.* **2017**, *21*, 773–791. [[CrossRef](#)]
10. Kaveh, A.; Talatahari, S. An improved ant colony optimization for constrained engineering design problems. *Eng. Comput.* **2010**, *27*, 155–182. [[CrossRef](#)]
11. Koide, R.M.; Luersen, M.A. Maximization of Fundamental Frequency of Laminated Composite Cylindrical Shells by Ant Colony Algorithm. *J. Aerosp. Technol. Manag.* **2013**, *5*, 75–82. [[CrossRef](#)]
12. Aymerich, F.; Serra, M. Optimization of laminate stacking sequence for maximum buckling load using the ant colony optimization (ACO) metaheuristic. *Compos. Part A Appl. Sci. Manuf.* **2008**, *39*, 262–272. [[CrossRef](#)]
13. Rao, A.R.M. Lay-up sequence design of laminate composite plates and a cylindrical skirt using ant colony optimization. *Proc. Inst. Mech. Eng. Part G J. Aerosp. Eng.* **2008**, *223*, 1–18.
14. Bloomfield, M.W.; Herencia, J.E.; Weaver, P. Analysis and benchmarking of meta-heuristic techniques for lay-up optimization. *Comput. Struct.* **2010**, *88*, 272–282. [[CrossRef](#)]
15. Dorigo, M.; Birattari, M.; Stutzle, T. Ant colony optimization. *IEEE Comput. Intell. Mag.* **2006**, *1*, 28–39. [[CrossRef](#)]
16. Rao, S.S. *Engineering Optimization Theory and Practice*; John Wiley & Sons: New York, NY, USA, 2019.
17. Bullnheimer, B.; Hartl, R.F.; Strauss, C. A New Rank Based Version of the Ant System. A Computational Study. 1997. Available online: <https://epub.wu.ac.at/616/1/document.pdf> (accessed on 15 August 2021).
18. Stützle, T.; Hoos, H.H. MAX-MIN ant system. *Future Gener. Comput. Syst.* **2000**, *16*, 889–914. [[CrossRef](#)]
19. Zhang, Y.; Liu, L.; Wangmeng, Z.; David, Z.; Dongyu, Z. Best-worst ant system. In Proceedings of the 2011 3rd International Conference on Advanced Computer Control, Harbin, China, 18–20 January 2011.
20. Blum, C.; Dorigo, M. The Hyper-Cube Framework for Ant Colony Optimization. *IEEE Trans. Syst. Man Cybern. Part B Cybern.* **2004**, *34*, 1161–1172. [[CrossRef](#)] [[PubMed](#)]
21. Dorigo, M. Luca Maria Gambardella: Ant colony system: A cooperative learning. *IEEE Trans. Evol. Comput.* **1997**, *1*, 53–66. [[CrossRef](#)]
22. Katagiri, H.; Hayashida, T.; Nishizaki, I.; Guo, Q. A hybrid algorithm based on tabu search and ant colony optimization for k-minimum spanning tree problems. *Expert Syst. Appl.* **2012**, *39*, 5681–5686. [[CrossRef](#)]
23. Jing, Z.; Fan, X.; Sun, Q. Stacking sequence optimization of composite laminates for maximum buckling load using permutation search algorithm. *Compos. Struct.* **2015**, *121*, 225–236. [[CrossRef](#)]
24. Talbi, E.-G. *Metaheuristics: From Design to Implementation*; John Wiley & Sons: New York, NY, USA, 2009; Volume 74.
25. Kogiso, N.; Watson, L.T.; Gürdal, Z.; Haftka, R.T. Genetic algorithms with local improvement for composite laminate design. *Struct. Multidiscip. Optim.* **1994**, *7*, 207–218. [[CrossRef](#)]
26. Malan, K.M.; Engelbrecht, A.P. Fitness landscape analysis for metaheuristic performance prediction. In *Recent Advances in the Theory and Application of Fitness Landscapes*; Springer: Berlin/Heidelberg, Germany, 2014; pp. 103–132.
27. Jones, T.; Forrest, S. Fitness Distance Correlation as a Measure of Problem Difficulty for Genetic Algorithms. *ICGA* **1995**, *95*, 184–192.