Journal of
Marine Science and Engineering

MDPI

Article

Improving the Computational Efficiency for Optimization of Offshore Wind Turbine Jacket Substructure by Hybrid Algorithms

Ding-Peng Liu [1], Tsung-Yueh Lin [2] and Hsin-Haou Huang [1,*]

[1] Department of Engineering Science and Ocean Engineering, National Taiwan University, Taipei 10617, Taiwan; k40407123@gmail.com
[2] R/D Section, Research Department, CR Classification Society, Taipei 10487, Taiwan; tylin@crclass.org
* Correspondence: hsinhaouhuang@ntu.edu.tw; Tel.: +886-2-3366-5753

Received: 8 July 2020; Accepted: 18 July 2020; Published: 22 July 2020

Abstract: When solving real-world problems with complex simulations, utilizing stochastic algorithms integrated with a simulation model appears inefficient. In this study, we compare several hybrid algorithms for optimizing an offshore jacket substructure (JSS). Moreover, we propose a novel hybrid algorithm called the divisional model genetic algorithm (DMGA) to improve efficiency. By adding different methods, namely particle swarm optimization (PSO), pattern search (PS) and targeted mutation (TM) in three subpopulations to become "divisions," each division has unique functionalities. With the collaboration of these three divisions, this method is considerably more efficient in solving multiple benchmark problems compared with other hybrid algorithms. These results reveal the superiority of DMGA in solving structural optimization problems.

Keywords: hybrid algorithm; genetic algorithm; particle swarm optimization; pattern search; jacket substructure; structural optimization

1. Introduction

Offshore wind energy has become a pertinent source of renewable energy worldwide. The material cost of offshore wind substructures (OWSs) accounts for approximately 17% of the total cost of installation [1]. Therefore, reducing the mass of the structure is a primary objective of optimization for reducing cost. Safety constraints are also a fundamental concern to make optimizing OWSs a multi-objective task. Because analyzing offshore wind turbines (OWTs) must consider multiple design loads, namely nonlinear wind, wave, and gravity loads, optimizing OWSs has been recognized as a highly nonlinear problem with complex computation [2]. Furthermore, calculating objective function is based on finite element analysis (FEA) without an analytical form, also seen as "black-box optimization."

Knowledge of algorithms is of great importance for OWS optimization. Deterministic algorithms such as the gradient-based method have been widely used in OWS optimization [3,4]; however, this method may cause solutions trapping in a local optimum. In contrast to the gradient-based method, stochastic algorithms evaluate all the particles in a search space using a specific search process. Because of the randomness of the algorithms, the stochastic algorithms can be more likely to find the optimum of the real-world problem. The most common stochastic algorithm, standard genetic algorithm (SGA), was introduced to achieve a global optimum [5–8]. SGA includes crossover and mutation as evolutionary operators, originating from the concept of competition and survival of biological evolution. Such processes help SGA find a global optimum due to its randomness. Colliding body optimization [9] has recently been applied to OWS optimization. The colliding bodies are many particles simulating the behavior of colliding, which is also a searching process.

However, stochastic algorithms usually require more function evaluations. Numerous studies have proposed a surrogate model when applying SGA in complex real-world problems, which is too inefficient to be integrated with simulation analyses [10]. Conversely, the gradient method is efficient enough to be integrated with simulation analyses in optimization processes, but it fails to obtain a global optimum. Designing a more efficient stochastic algorithm has become a pertinent topic for solving real-world problems using simulation models with acceptable computational costs.

Hybridization of algorithms is an effective strategy for improving the performance of original algorithms. Hybrid algorithms can preserve the desired features of the original algorithms. For highly nonlinear complex problems, conventional deterministic methods tend to find a local optimum. Some global modifications, e.g., Hofmeister [11] did overcome this issue, but stochastic algorithms can mostly ensure the reliability of solutions. Pattern search (PS) is a common local search method that is hybridized with stochastic algorithms. Because stochastic algorithms are inaccurate and converge slowly during optimization, some researchers have combined local search with an evolutionary algorithm to improve its accuracy and convergence. In 2017, Li [12] selected the best particle among SGA to execute the PS method to obtain a relatively accurate result faster, namely pattern search genetic algorithm (PSGA). Similarly, particle swarm optimization (PSO) can also be hybridized with PS. For the optimized fuzzy proportional integral controller of the automatic generation controller, Sahu [13] combined PS with PSO by executing the two algorithms in turn. While PSO contributes to global exploration, PS is used for local exploitation.

Many researchers have attempted to combine SGA with other stochastic algorithms. For example, simulated annealing (SA) was inspired by the annealing process, whereas the probability for particles jumping to a neighborhood was based on the annealing function. Researchers implemented this concept into SGA to determine whether a particle can be replaced by a new particle from crossover operators [14]. This arrangement helped SGA to evolve toward a correct direction, and hence improved its efficiency. Pourvaziri [15] used SA for local search in an island model genetic algorithm (IMGA) to help each population search for feasible space.

Hybridizing SGA and PSO is also a feasible approach to finding a global optimum. Krink [16] proposed a simple hybrid scheme looping SGA, PSO, and hill-climber algorithms in sequence for local search, namely the lifecycle model genetic algorithm (LMGA). If the number of generations without improvement exceeds a given threshold, the executing algorithm changes.

In this study, we proposed a novel hybrid algorithm called the divisional model genetic algorithm (DMGA). We combined PS with SGA and PSO to accelerate the convergence of the optimization process while maintaining the diversity of the particles. The algorithm contains three divisions, with each division having a unique functionality, namely a divisional model. Detailed description of these algorithms is contained in Section 2. Then we conducted performance analyses of the proposed algorithm comparing with various hybrid algorithms in Section 3. Finally, in Section 4 we solved a jacket substructure (JSS) optimization problem with several hybrid algorithms. The main goal of the present study is to demonstrate the superiority of the proposed hybrid algorithms in solving JSS optimization problem. We chose a four-legged jacket substructure with X-type interleaving braces planted in the Taiwan Strait as the optimization model. Through updating the radii and thicknesses of structural members, we targeted a lighter design under safety constraints in ultimate environmental loads.

2. Materials and Methods

2.1. Original Algorithms

The introduction and development of the algorithms referred to in this paper are described as follows. In this section, we present the original algorithms: SGA, PSO and PS. Any modifications or improvement of DMGA are based on these algorithms and will be discussed in Section 2.2.

2.1.1. Standard Genetic Algorithm (SGA)

In 1950, Turing [17] stated that computing intelligence could be achieved through the evolutionary process of "the survival of the fittest." In 1975, Holland [18] proposed a random search algorithm, SGA, inspired by Darwin's theory of evolution. In SGA, characterized evolutionary operators, namely selection, crossover, and mutation, are dedicated to generating new chromosomes (data points) to replace old chromosomes. Because operators can be easily hybridized with other algorithms, many modified or hybrid algorithms have originated from SGA.

Because of the randomness of SGA, obtaining a global optimum can be guaranteed if the evaluation time is infinite [19]. Although the probability of finding a global optimum using SGA is higher than in other algorithms, practically SGA might obtain a local optimum under a limited computational time or slowly converge in nonlinear or multi-optimum problems, which can be improved by hybridizing with other algorithms.

2.1.2. Particle Swarm Optimization (PSO)

PSO is an evolutionary algorithm developed by Kennedy in 1995 [20]. This algorithm was inspired by social interaction of animals. For example, when a flood occurs, birds search for food, with each bird simultaneously flying in a direction instructed by other birds aligned with self-experience. Shi modified PSO in 1998 [21] by adding inertial weight to balance the global and local search by exciting the "flying" of PSO. Shi stated that the inertia weight should be between 0.8 and 1.2 to obtain a global optimum in a moderate number of iterations. This modified form of PSO is as follows:

$$v_i^{t+1} = \omega v_i^t + \varphi_1 \beta_1 \left(p_i - x_i^t\right) + \varphi_2 \beta_2 \left(p_g - x_i^t\right), \tag{1}$$

$$x_i^{t+1} = x_i^t + v_i^{t+1}, \tag{2}$$

where ω is inertia weight; φ_1 and φ_2 are two positive constants defined according to the problem; β_1 and β_2 are two random numbers between 0 and 1; x_i^t and v_i^t denote the position and velocity of i^{th} particle in t^{th} generation, respectively. Next, the velocity of i^{th} particle in $t + 1^{th}$ generation v_i^{t+1} is updated, through solving Equation (1), to refer to p_i and p_g, which represent the personal best position and global best position, respectively. Afterwards, particles are updated to a new position x_i^{t+1} by invoking Equation (2).

2.1.3. Pattern Search (PS)

Developed in 1961 by Hooke [22], PS is a type of local search that does not require the gradient of a function. PS varies in a single dimension at one time by a step size to improve the solution. The step size is halved when the result has variations in any direction. Finally, when the step size is small enough, the optimization is judged to be convergent.

2.2. Divisional Model Genetic Algorithm (DMGA)

In this section, the structure and the procedure of the divisional model are described in detail. Next, the SGA operators and the three divisions collaborating in DMGA are introduced.

The divisional model divides the whole population into three subpopulations to execute in the respective divisions. At the beginning of the optimization, all the subpopulations are initialized and the fitness of every chromosome is calculated. Afterwards, the best chromosomes in each division gather in the pattern search genetic algorithm division, PS-GA, in preparation for local search, named "Elite migration." Conversely, the least favorable particles migrate to the targeted mutation division, which is called "Abandon" to increase the diversity of the division, as presented in Figure 1.

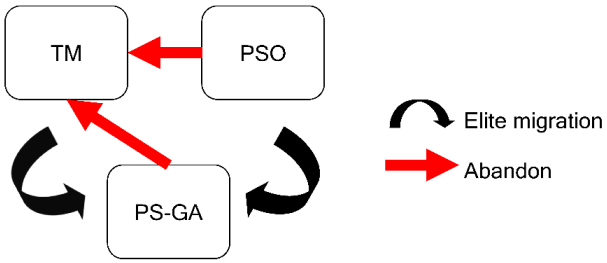

Figure 1. Structure of Divisional Model Genetic Algorithm (DMGA) algorithm, in which the population is divided into Targeted Mutation (TM), Pattern Search-Genetic Algorithm (PS-GA) and Particle Swarm Optimization (PSO) divisions.

Following the migration process, the chromosomes evolve in different divisions as shown in the depicted flowchart of DMGA in Figure 2. After an elite migration process, the optimal chromosome in the PS-GA division is selected as the initial point to conduct PS with an adaptive strategy, as described in Section 2.2.3. This chromosome is regarded as the best solution in this generation. Then the chromosomes in the PSO division are updated by using the best solution as the global best position for PSO. The targeted mutation (TM) division then determines the targeted mutation range according to the performance of all the chromosomes abandoned inside. The main TM range of each dimension is located in the part of the particles with better performance. Finally, SGA with the acquired TM range operates in the PS-GA and TM divisions to generate new chromosomes. Two chromosomes from the divisions are selected and complete a single-point crossover and mutation separately. The optimization process stops until the termination criteria is satisfied. To understand the procedure more clearly, the pseudo-code of DMGA is referred to Appendix A.

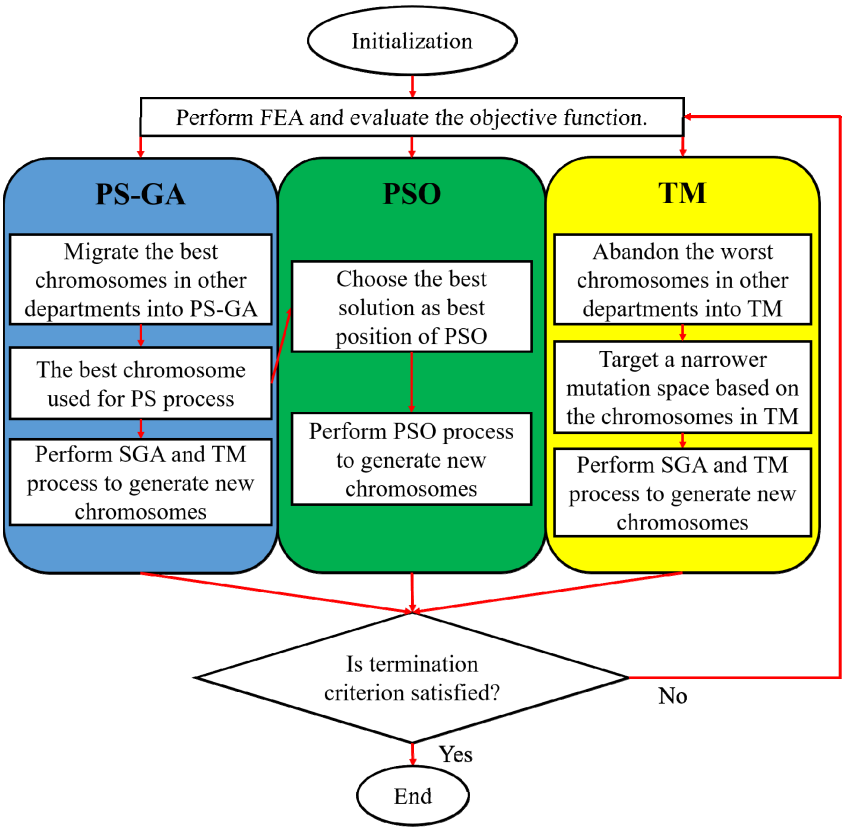

Figure 2. Flowchart of the divisional model genetic algorithm (DMGA).

2.2.1. Initialization

The full population N was divided into three parts for the three divisions. Hence, the population of one division n is equal to N/3. Next, the initial populations of all the divisions were randomly generated in a feasible space. We then randomized the initial populations according to the lower bound and search range of the problem, as per the following formula:

$$x_i^j = lowerbound^j + range^j * rand[0,1], \tag{3}$$

where x_i^j represents the i^{th} particle/chromosome in the j^{th} dimension; $lowerbound^j$ and $range^j$ represent the lower bound and range defined in the j^{th} dimension.

When we considered the parameters of each algorithm, the initial velocity of PSO is defined as zero for all the particles. Next, we initialized the parameters for SGA: mutation probability p_m, and the size of the dominant population N_e is equal to $n/2$. Lastly, we defined the parameters of PS, the step size and tolerance, according to the scale of the target problem.

2.2.2. SGA Operators

Three types of SGA operators are used in DMGA: selection, crossover, and mutation. The selection operator is responsible for selecting two particles from a dominant population as parents for the crossover operator. After a fitness evaluation, the better half of the chromosomes is chosen as the dominant population. Next, two different chromosomes as parents are selected, referring to the selection probability p_s of each chromosome, which is the ratio of its ranking to the summation of rankings, as per the following formula:

$$p_s^i = 1 - \frac{ns + 1 - ranking^i}{\sum_{i=1}^{ns} ranking^i}, \tag{4}$$

where p_s^i denotes the selection probability of i^{th} chromosome in the division; $ranking^i$ is the fitness ranking of i^{th} chromosome. The total number of the dominant population ns is half that of the division population.

Single-point crossover and mutation were adopted in this study. Two selected chromosomes coded in binary were parents, and we randomly generated a position in the string as the cutting point. The single-point crossover operator swaps the genes of the parents behind the cutting point, generating two children chromosomes. The mutation operator is performed on a single gene of a chromosome. If a randomly generated number from 0 to 1 is under p_m, a gene is randomly chosen and replaced by a random value in the mutation range. The mutation range is contracted as the TM range when the random number is under $p_m/2$, as revealed in Figure 3.

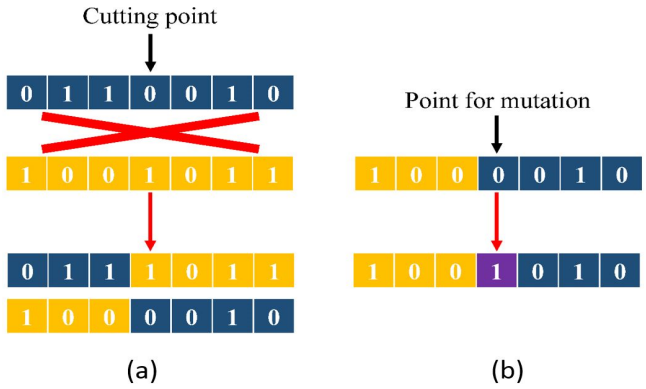

Figure 3. Operators of standard genetic algorithm (SGA): (**a**) crossover and (**b**) mutation operators. The chromosomes are coded in binary.

2.2.3. PS-GA Division

The PS-GA division modifies the PSGA. After each generation, the optimal chromosome of the entire population is selected to complete PS as a local search around the present position. Therefore, after the stochastic algorithm in any division generates a favorable solution, the deterministic algorithm directly improves the solution in this division. The flowchart of PS is presented in Figure 4.

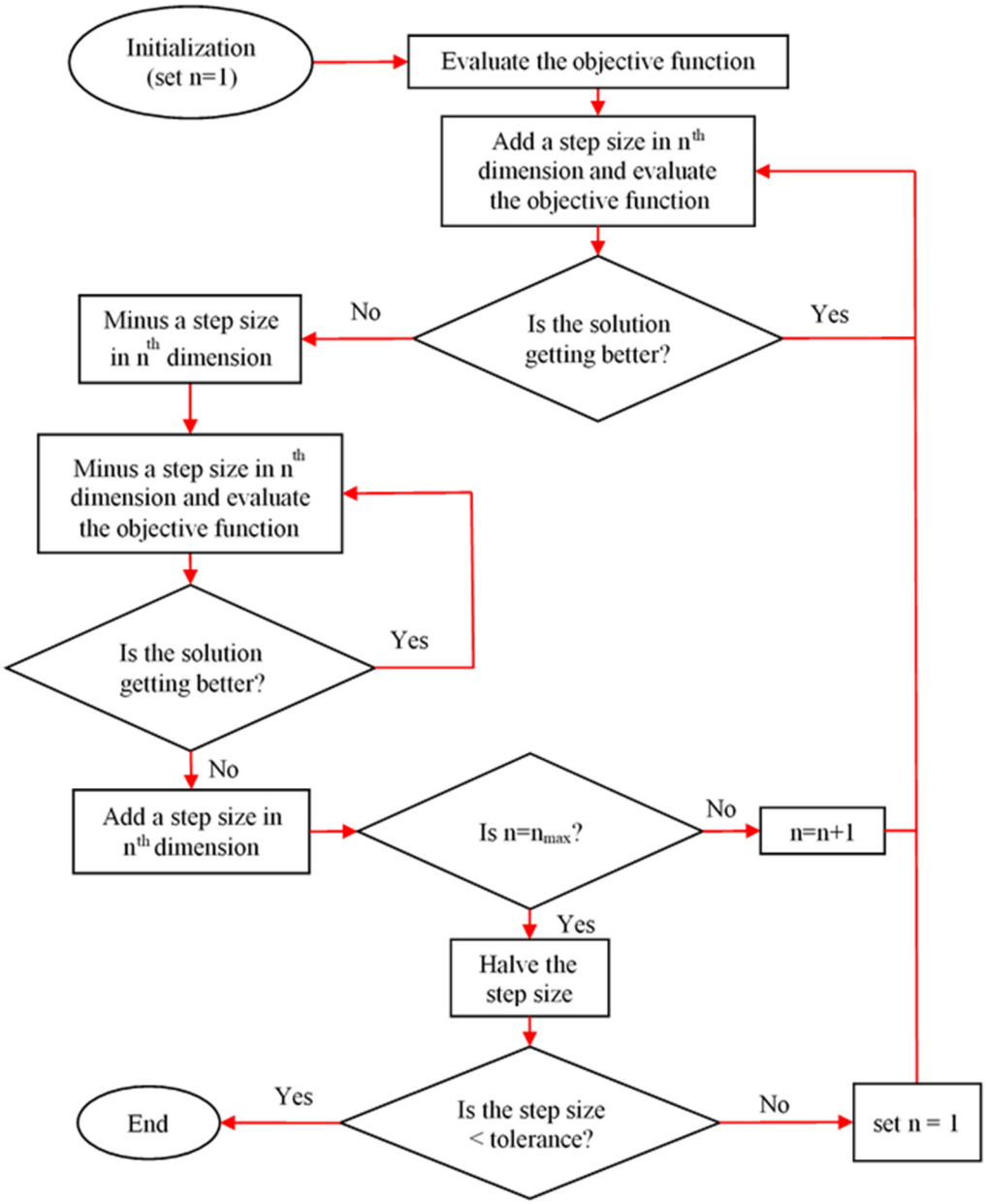

Figure 4. Flowchart of the pattern search (PS).

To improve the efficiency of PS, we added an adaptive strategy to adjust the step size. As the generation passes, the initial PS step size and PS tolerance decreased linearly with the number of the generation to prevent too many evaluations being required in the forepart of optimization. When the step size becomes relatively small, the optimization becomes more concise, and thus the PS process is improved. The step size and tolerance of the PS are initialized according to Equations (5) and (6).

$$step\_size = initial\_size * (Max\_generation - generation) / Max\_generation \quad (5)$$

$$tolerance = initial\_tolerance * (Max\_generation - generation) / Max\_generation \quad (6)$$

Finally, the selection, crossover, and mutation operators are performed in this division. Therefore, the beneficial chromosomes are delivered to the next generation and are considered to have the most potential to achieve a better solution than other divisions due to the gathering of superior particles.

2.2.4. PSO Division

PSO is another stochastic algorithm used in DMGA. However, PSO searches for a global solution in a different manner from SGA. PSO updates every particles by selecting both the global best and self-experience particles, whereas SGA updates chromosomes by selecting parents to deliver their beneficial genes to the next generation. This strategy reduces the risk of trapping in a local optimum. PSO independently updates every particle that belongs to a division in accordance with Equations (1) and (2).

Regarding the global best position in the population of the PS-GA division, PSO in this division converges faster than the traditional PSO. As particles approach to the global best position, a better solution may be discovered in the next generation. Thus, DMGA increases the diversity of searching directions to offer a higher probability of achieving a global optimum.

2.2.5. TM Division

In contrast to the PS-GA division, the least favorable particles of other divisions are gathered in the TM division. The TM division targets a narrower mutation range with a higher possibility of containing a global solution. As shown in Figure 5, all the particles in the division are firstly divided into two sections according to the median of variables in each dimension. Then the total fitness of the particles in both sections are calculated—for instance, high fitness means lighter structural mass under constrained safety criteria in this study—to determine the superior section. The superior section in each dimension contributes to form the TM range for the mutation operator in the PS-GA and TM divisions. Finally, particles in the TM division perform the SGA operators to generate a new generation.

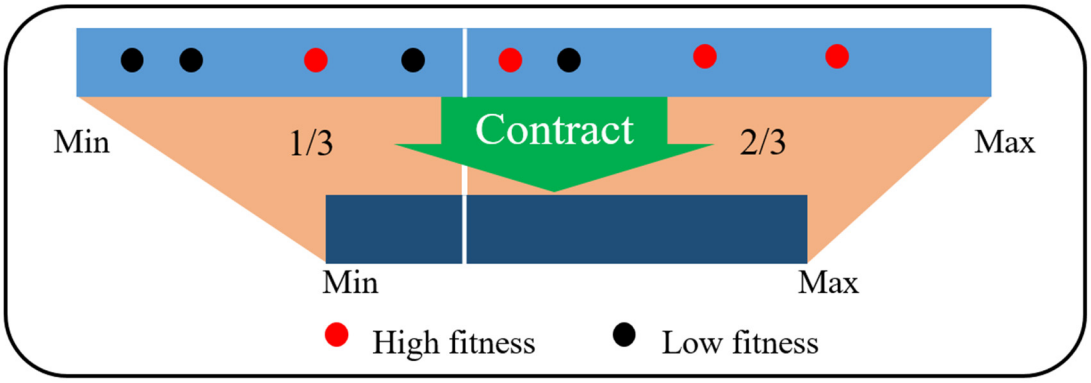

Figure 5. Contraction of the search range through TM. Every point and the white line indicate every chromosome and median in one dimension. In this case, the right side is the superior part of this dimension, applied with a 2/3 contraction.

The TM range is arranged according to the superior part of each dimension, as per Equations (7)–(9), by contracting the section with the lower fitness by one-third and the other section by two-thirds. If the mean fitness values of the two sections are equal, however, both sections shrink by half.

$$\text{If the lower part is superior : } \begin{cases} lowerbound^j_{TM} = lowerbound^j * \frac{2}{3} + median^j * \frac{1}{3} \\ range^j_{TM} = \frac{1}{3} * \left(range^j + median^j - lowerbound^j \right) \end{cases}, \quad (7)$$

$$\text{If the upper part is superior}: \begin{cases} lowerbound^j_{TM} = lowerbound^j * \frac{1}{3} + median^j * \frac{2}{3} \\ range^j_{TM} = \frac{2}{3} * (range^j - \frac{median^j}{2} + \frac{lowerbound^j}{2}) \end{cases}, \quad (8)$$

$$\text{If the both parts are equal}: \begin{cases} lowerbound^j_{TM} = lowerbound^j * \frac{1}{2} + median^j * \frac{1}{2} \\ range^j_{TM} = \frac{1}{2} * range^j \end{cases}, \quad (9)$$

where $lowerbound^j_{TM}$ and $range^j_{TM}$ represent the TM lower bound and range in j^{th} dimension, whereas $lowerbound^j$ and $range^j$ denote the original ones. According to median value $median^j$ of chromosomes in j^{th} dimension, we obtain the lower bound and range for TM.

3. Benchmark Study

The performance of DMGA is discussed in this section and compared with other algorithms. All algorithms were coded using MATLAB software. To avoid attributing performance to a randomly chosen initial population, we conducted 100 runs of each test function and recorded whether we reached the optimum and the number of function evaluations. Every run started from a different initial population, with the population size set at 60. In order to prove the potential of DMGA in diverse applications, we chose diverse test functions with different difficulties. Because some test functions are more complex than others, to make them comparable, we set the limitation of function evaluations very high, say 600,000, so that convergence (if exists) can be achieved without being bounded by the limit. For real optimization applications, the number of evaluations should be set according to limitation of computation time, for example, the jacket substructure problem in Section 4 was limited to 1500 because of time-consuming finite element analysis of each evaluation. Finally, the effectiveness and efficiency of the algorithms were calculated by the success rate and the number of function evaluations, respectively.

3.1. Test Functions and Algorithms Setup

We selected 13 common test functions for the optimization with high diversity for analyzing the performance of algorithms. The test functions came from the CEC2013 benchmark and other resources [23–26]. All the test functions and references are listed in Table A2 in Appendix B. The search range and tolerance for each test function is given according to the property of the function.

We defined the mutation probability p_m as 0.5 and p_{mi} as 0.1. Other parameters for PS and PSO were set according to the property of each test function, as listed in Table A3. The results for all the test functions calculated by DMGA are presented and compared with SGA, PSO, and their hybrids. In this study, we inferred four hybrid algorithms (as described in the Section 1), which are based on SGA, PSO and PS. We listed the name and reference of the algorithms below:

- PSGA by Li et al. [12]
- Hybrid pattern search PSO (PSPSO) by Sahu et al. [13]
- Lifecycle model GA (LMGA) by Krink and Løvbjerg [16]
- Hybrid GA and PSO (HGAPSO) by Juang [27]

It is worth mentioning that we replaced the hill-climber in LMGA by PS in this study to make a fair comparison with other algorithms.

3.2. Performance Analysis

The overall computational results listed in Table 1 present the efficiency and effectiveness of selected algorithms when solving different test functions. The results of DMGA obtained 100% effectiveness in 9 of the 14 test functions, and the number of function evaluations generally stayed lower than other algorithms when reaching similar effectiveness. However, several test functions were relatively difficult for DMGA to solve, but it could still obtain over 60% effectiveness.

Table 1. Computational results of the SGA and PSO and their hybrids.

Function	Algorithm / Effectiveness [1] / Efficiency [2]	DMGA	SGA	PSO	PSGA	PSPSO	HGAPSO	LMGA
Ackley		100	0	56	68	100	91	99
		5380	0	1451	252,278	1891	219,310	13,040
Schwefel		100	100	6	100	2	100	100
		1722	175,399	1330	34,476	195	48,414	22,643
Rastrigin		100	0	0	71	100	0	0
		3197	0	0	203,757	4389	0	0
De Joung		100	100	100	100	100	100	100
		183	3434	395	192	194	686	419
Rosenbrock		99	1	67	2	79	100	98
		35,237	320,580	8424	235,324	10,072	64,466	154,925
Goldstein-Price		100	100	100	100	100	100	100
		446	3365	482	643	527	1259	524
Easom		100	100	99	100	95	100	100
		962	11,912	7244	3284	1608	6288	1909
Zakharov		100	0	13	3	100	95	99
		10,370	0	1209	507,315	2251	265,632	57,163
Hartman ($H_{6,4}$)		80	96	6	98	54	40	56
		7502	161,007	136,880	65,964	836	32,072	84,422
Eggholder		64	86	0	72	8	49	97
		90,029	164,835	0	164,080	25,302	66,124	106,781
Schaffer		84	93	0	95	27	31	24
		118,778	188,019	0	159,428	56,886	74,706	286,874
Styblinski-Tang		100	42	16	99	12	100	100
		738	387,744	1084	35,968	242	26,356	13,725
Beale		100	100	89	100	94	100	100
		729	3775	537	2484	624	6559	749

[1] Times of success in 100 runs of each function. [2] Average number of function evaluations in 100 runs for each function.

Although SGA and PSO are both stochastic algorithms, they have their own specialties in different test functions. As a result, their hybrids with PS, PSGA and PSPSO showed the same tendency to solve some functions whereas the other algorithms did not. It is worth noting that these two algorithms perform well in the Rastrigin function, which shows the hybridization of PS improving the convergence toward local minimum. However, PS in LMGA became less effective in the Rastrigin function because of the limited number of function evaluations when three algorithms took turns to solve the problem.

HGAPSO is the combination of SGA and PSO conducted in each half of the population separately, showing superiorities from both original algorithms. Nevertheless, it generally cost more function evaluations than other algorithms composed of PS. From the previous discussion, it can be seen that DMGA successfully combined the three original algorithms in an effective way, thus improving effectiveness and efficiency.

In addition, we conducted a ranking analysis based on the effectiveness and efficiency of each algorithm. First, we compared the effectiveness of the algorithms. If there were algorithms with the same level of effectiveness, we then compared the efficiencies as a ranking rule. Table 2 lists the ranking

of every algorithm in solving each function. The average of the rankings of functions was calculated to compare the overall performance among the algorithms.

Table 2. Rankings of the performance analyses.

Algorithm / Function / Rank	DMGA	SGA	PSO	PSGA	PSPSO	HGAPSO	LMGA
Ackley	2	7	6	5	1	4	3
Schwefel	1	5	6	3	7	4	2
Rastrigin	1	4	4	3	2	4	4
De Joung	1	7	4	2	3	6	5
Rosenbrock	2	7	5	6	4	1	3
Goldstein-Price	1	7	2	5	4	6	3
Easom	1	5	6	3	7	4	2
Zakharov	2	7	5	6	1	4	3
Hartman ($H_{6,4}$)	3	2	7	1	5	6	4
Eggholder	4	2	7	3	6	5	1
Schaffer	3	2	7	1	5	4	6
Styblinski-Tang	1	5	6	4	7	3	2
Beale	1	4	6	3	6	5	2
Avg. Rank	1.77	4.92	5.46	3.46	4.46	4.31	3.08

Although there is rarely an algorithm outperforming other algorithms for all functions since the performance is always problem dependent, DMGA was ranked first and performed the best in 7 out of 13 test functions. With regard to the Eggholder function and Schaffer's function, DMGA was ineffective because of the ill-conditioning. DMGA failed to detect the correct TM range and might trap in the local minimum during the PS process. Despite this, in other functions, DMGA was superior in both effectiveness and efficiency. Similar to DMGA, LMGA is also a hybridization of PS, SGA, and PSO, but each operator in LMGA is executed in sequence. Therefore, LMGA would not change the operator until the optimization gets stuck in the same solution for several generations resulting in an unused original algorithm.

DMGA was superior for two main reasons. First, the collaboration of the divisional model improved the performance. Through the migration between the divisions, the best solution in each generation proceeded through different operators. Thus, the local optimum became active to keep the optimization process searching for the global optimum. Figure 6 reveals that in the forepart of the DMGA optimization, each division reached the best solution one by one for a rapid convergence. In the initial population, the best solution was located in the TM division. Afterward, PS-GA division improved the solution during 7th to 9th iterations. After trapping in the local optimum for several iterations through repeated collaboration of PSO and PS-GA, the global optimum was obtained.

The second reason for the superiority of the DMGA was the mutation strategy. The original mutation and TM operators were used simultaneously to mutate more effectively. As shown in Figure 7, Easom's function is a highly nonlinear function without apparent tendency. The minimum of this function is -1 when x_1 and $x_2 = \pi$, whereas the value stays zero at other locations. Therefore, we might search the solution mainly through mutation. DMGA had the best mutation strategy, resulting in the fastest convergence.

We then checked the TM range shown in Figure 8 and revealed that the solution was contained in the TM range throughout the optimization process. Hence, this strategy enhanced the efficiency of the mutation. DMGA not only converged faster than other algorithms but also searched for a global optimum with an efficient mutation operator. DMGA is competitive at completing multiple optimization problems.

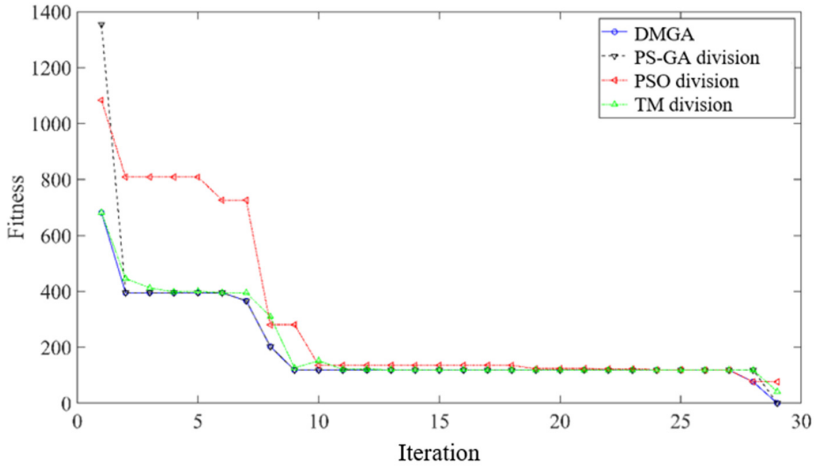

Figure 6. Fitness plot of each division in the DMGA versus iterations in solving Schwefel's function.

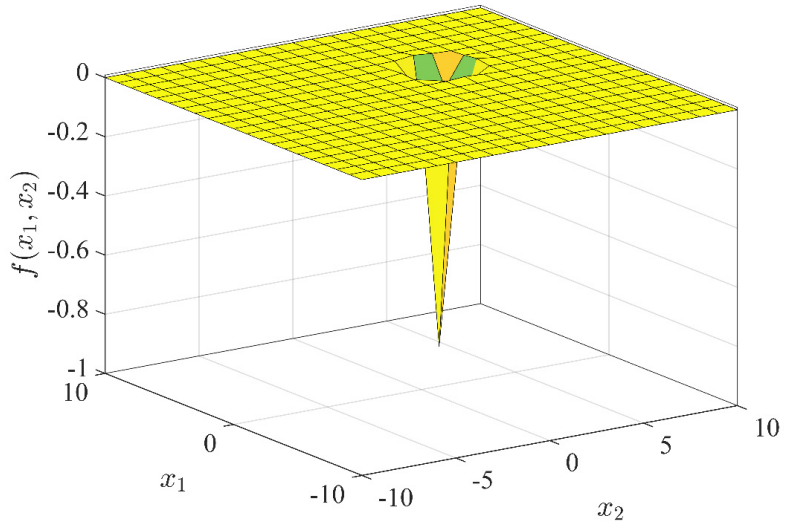

Figure 7. Surface plot of Easom's function when x_1 and x_2 = −10 to 10.

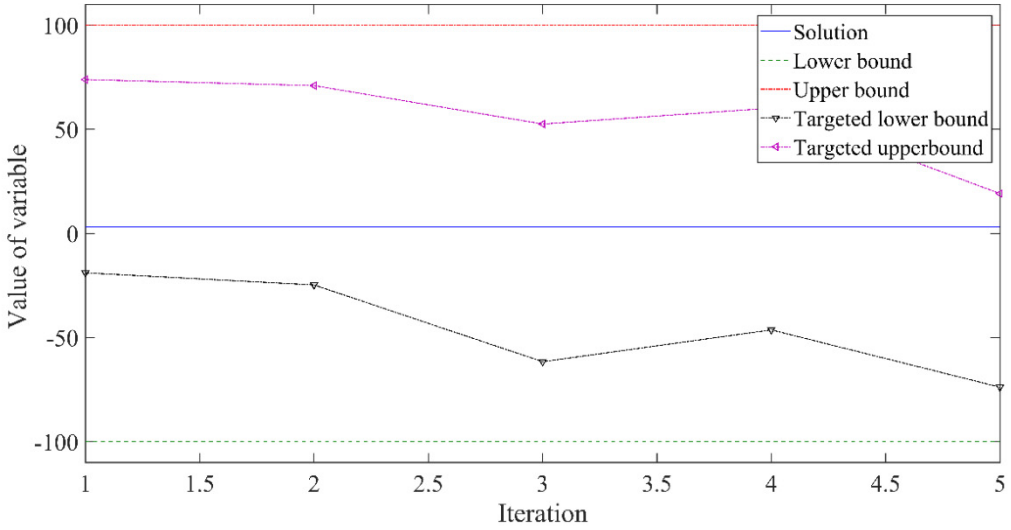

Figure 8. Search range of the DMGA by targeted mutation versus iterations in solving Easom's function.

4. Jacket Substructure Optimization

4.1. Finite Element Modelling

For a real-world problem, an offshore wind turbine substructure was used to minimize its weight as the objective function. We built a finite element analysis model of the JSS by using Abaqus commercial software package, as depicted as Figure 9. The reference design was obtained from the preliminary drawing in a Taiwan offshore pilot project, provided by a private wind energy developer. The height and foot print of this JSS are 38 m and 11 m by 11 m square, respectively. The topology of the tubular members are four stories, of which they are categorized in three zones. The four legs are connected by X-type braces on each face and in each story. Next, we defined sectional properties, thickness and radius of the members as design variables, numbered from brace to leg, and the atmospheric zone to the immersion zone. The design variables included the thickness and radius, numbered from brace to leg, and the atmospheric zone to the immersion zone. $t_1 \sim t_8$ and $r_1 \sim r_8$ represent the thickness and radius of the brace and leg in the atmospheric, splash, and immersion zones, respectively. The topologies of the piles and members of the footprint remained as is. The whole structure was modelled by beam elements, where the wind load was imposed at the top of the tower and the wave loads were computed on the nodes of elements. The beam elements in the support structure were categorized into primary (legs) and secondary (braces) elements. The joints at legs and braces were modelled by truncating the secondary beams with the profiles of the primary beams, so that the overlapping problem was eliminated, i.e., the braces did not extrude into legs (shown in detail in Figure 9). Then a rigid connection represented the joint bridges and the truncated breaks from the leg member to the braces. After the length of each beam and its corresponding profile were determined, the mass calculation was performed element by element, multiplying the sectional mass and its length. The whole model contained 720 nodes in the substructure. The model was conducted as elastic static analysis, with a concern of computational efficiency for thousands of design evaluations in the optimization exploration. Although some studies prefer dynamic analysis to capture more realistic structural responses, those combined with stochastic approaches suffered from low computational efficiency [28]. It could be a rational simplification of the analysis in the design iteration, and the proposed DMGA attempts an alternation algorithm, which is also compatible with a dynamic solver in the future. The masses of the equipment and secondary structures (e.g., Nacelle, blades and access platform) were added to the finite element model, i.e., 240 tons of RNA, 60 tons of the flange. The tower was also modelled as tapered tubular members at a height of 90 m and 515 tons of weight, according to the specification of this wind turbine, and it was connected to the top of the jacket through rigid elements.

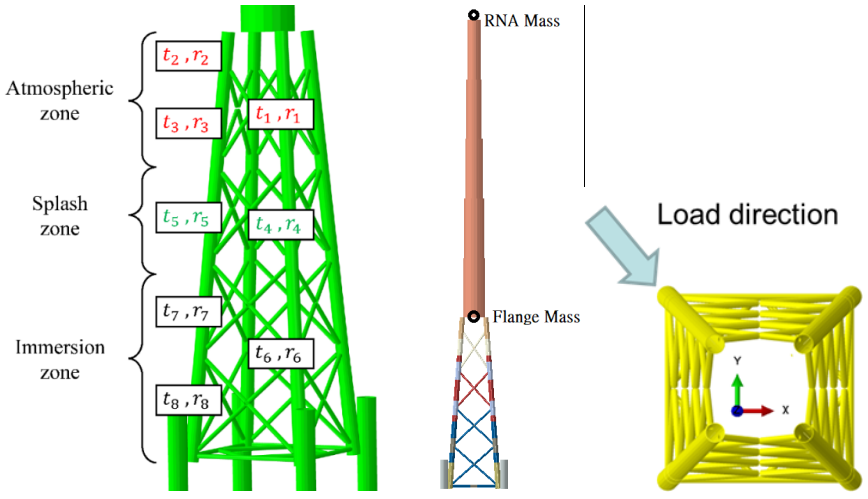

Figure 9. Finite element analysis (FEA) model of the jacket support structure.

The loads of wind, wave, and current were applied to the finite element models through line loads (i.e., force per unit length) on the beam elements and a concentrated force for the blades wind loads at the nacelle. Despite fatigue usually being the design-driving load case for jacket structures, this study focuses on the performance of optimization algorithms. Thus, only survival conditions were considered to reduce the computational time of function evaluation. According to the ultimate limit state (ULS) specified in the design load case in the local requirement [29] for this reference offshore wind turbine, the turbine has to comply with class IA specifications as per the IEC61400 [30], and the substructure has to survive in an ultimate environmental condition of 50-year return period. A load factor of 1.35 was applied to the environmental loads for ultimate strength assessment [31]. A parked wind turbine with a blade pitch control fault condition under the extreme gust sustained for 3 s was considered. Wind speed of class IA was applied on this turbine, 50 m/s. The gust factor was defined as 1.4 according to the rule [30]. Therefore, the gust speed reached 70 m/s in the analysis, with a wind profile of ground shear exponent 0.13. A 4 MW wind turbine, derived from the NREL 5 MW reference model [32], was with a rotor diameter of 120 m. The wind load on the rotor and nacelle assembly (RNA) was generated by utilizing steady blade element momentum method (BEM), which was 2030 kN. The line load on the tower was then computed via empirical formula, which models the pressure drag as a drag coefficient times diameter and velocity square at that elevation, resulting in a total 683 kN. Wind loads on the jacket and flange were regarded as small compared to the wind turbine, and hence were neglected. Due to lack of information about the structural profiles of blades of this assumed turbine and damping of RNA, the aero-elasticity and induced dynamic load by gust were neglected. A constant wind load was applied, 2030 kN horizontally, resulting in a 260 MN·m overturning moment on the legs of substructure. The acting direction aligned with the diagonal of the footprint, so that compression stress was taken by one leg.

On the other hand, according to the site investigation of the pretend location in the Taiwan Strait [33], the local requirements of the wave condition for the Taiwan offshore demonstrating wind turbines pose a significant wave height at 8 m in a 50-year return period. Therefore, the maximum wave height in this extreme sea state was expected to be 14.8 m, i.e., 1.85 times the significant wave height [33]. According to the categories of regular wave theory, a minimum wave period (highest wave dynamics) of 12.7 s was determined on the breaking limit (Figure 10), which corresponds to 5th-order Stokes wave regime. Table 3 lists the properties of the extreme wind and wave conditions. According to the aforementioned local requirements, steady current velocity with a seabed shear profile on the surface, 1.0 m/s, was superposed on the velocity field of wave. In contrast to the steady wind load, the transient wave loads are more complex and nonlinear. As the diameters of JSS tubular members are much smaller than the wave length, the wave loads are can be computed by the semi-empirical Morison equation [34], as per Equation (10), where ρ is the water density; C_m and C_d are the coefficients of the inertia force and the drag force respectively, which differ for the abovementioned zones; u and $\dot{u}$ are the flow velocity and acceleration respectively; and D is the member diameter.

$$F = \rho C_m \frac{\pi}{4} D^2 \dot{u} + \frac{1}{2} \rho C_d D u|u| \tag{10}$$

Table 3. Parameters for wind and wave loads.

Mean Wind Speed	Gust Factor	Gust Wind	Wave Height	Wave Period	Wave Length
50 m/s	1.4	70 m/s	14.8 m	12.7 s	197 m

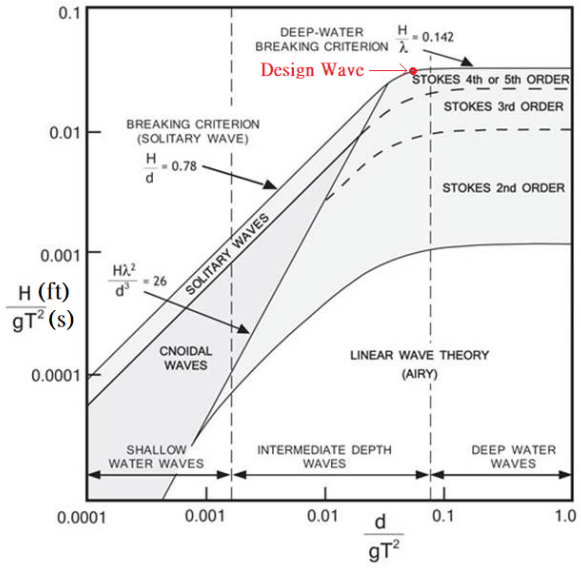

Figure 10. Regular wave theory and the design wave [35].

Observing the Equation (10), the total wave load is proportional not only to D but also to D^2. The nonlinear wave pertains a sharper crest and higher particle acceleration. Figure 11 demonstrates the wave profile and the two components of wave load. As the radius changes, as shown in Figure 12 for example, owing to change of volume and area applied by the wave loads, the nonlinear force and its corresponding time of extremity change during the wave propagation. Due to the topology of the jacket not changing, u and $\dot{u}$ are the same when changing radius. The velocities and accelerations are computed on the nodes of structure in advance, and then the forces are computed based on the Morison equation and provide the radius of each member in the design iteration. In order to apply the maximum wave load on the structure, the wave direction is assumed to align with wind and the load in a whole wave period is computed, and then its maximum value is extracted for static structural analysis. For the reference design, the maximum overturning moment on the substructure is about 100 MN·m. Since the dynamics of wind load is not considered, the relatively lower wave load is also treated as static.

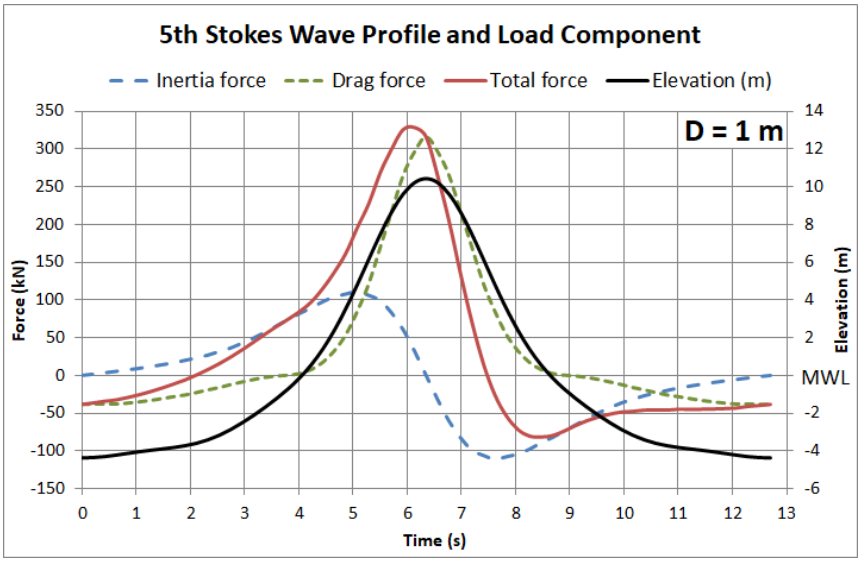

Figure 11. Wave profile and load component.

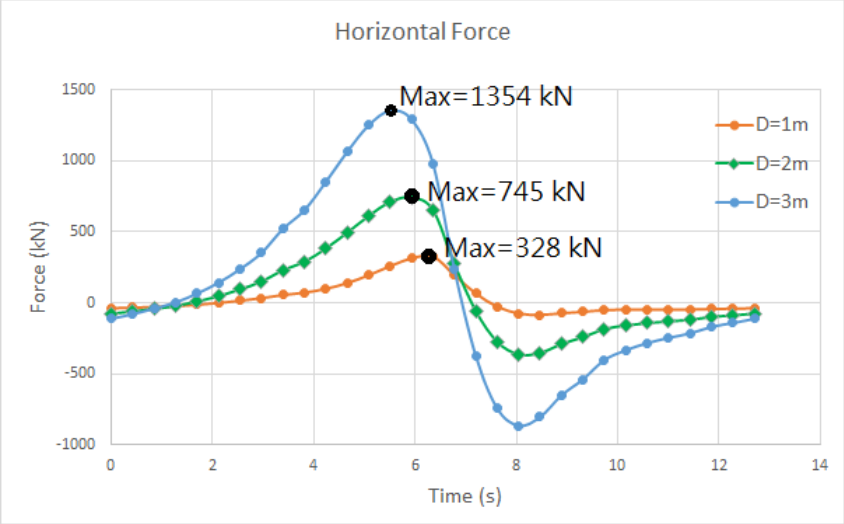

Figure 12. Wave load on a monopile tubular member with different diameters.

4.2. Algorithm Setup

The optimization problem of JSS is defined as a constrained nonlinear programming problem, which can be illustrated as follows. The design variables are the thickness $\vec{t}$ and radius $\vec{r}$ of members of the substructure, constrained between the lower bound t_{Lj}, r_{Lj} and upper bound t_{Uj}, r_{Uj}, as per Equations (11) and (12); the objective of the optimization is the mass of the substructure, as per Equation (13), which is subject to four constraints, two geometric constraints g_1, g_2 (Equations (14) and (15)), and two safety constraints g_3, g_4 as per NORSOK N-004 (Equations (16) and (17)).

$$\vec{t} = [t_1, t_2, t_3, \ldots, t_n] \in R^n,\ t_{Lj} \le t_j \le t_{Uj}, \qquad j = 1, \ldots, n \tag{11}$$

$$\vec{r} = [r_1, r_2, r_3, \ldots, r_n] \in R^n, r_{Lj} \le r_j \le r_{Uj}, \qquad j = 1, \ldots, n \tag{12}$$

$$\min f(\vec{t}, \vec{r}) \tag{13}$$

$$g_1(\vec{t}, \vec{r}) = r_j/t_j \le 60, \qquad j = 1, \ldots, n \tag{14}$$

$$g_2(\vec{t}, \vec{r}) = t_j - r_j \leqq 0, \qquad j = 1, \ldots, n \tag{15}$$

$$\text{s.t. } g_3(\vec{t}, \vec{r}) \leqq 355 \tag{16}$$

$$g_4(\vec{t}, \vec{r}) \leqq 1 \tag{17}$$

Regarding the rule, the lower bound of the structural thickness t_{Lj} is 6 mm. The ratio of radius to thickness must be less than 60, and the radius must be larger than the thickness. Accordingly, we set two geometry constraints g_1, g_2. Since the dimensions of the structure legs are generally greater than those of braces, we set the minimum thickness of the legs to be 26 mm for accelerating the optimization process.

Random initialization over the design domain was used to avoid potential local minimum biased by the reference design. In the first generation, most chromosomes violated the safety constraints; hence, we added multiples of mass to the objective function to manifest the violation of safety constraints, i.e., soft constraints [36]. Soft constraints can be violated during the process, but they introduce high penalties on the objective function. g_3 is the ultimate stress criteria, indicating the yield strength σ_y^{S355} (=355 MPa) for S355 steel. g_4 is the local buckling criteria, calculated through equations given by NORSOK N-004 [37], as described in Appendix C. If the computational result violates the constraints,

the objective function will increase by the multiple of the structure mass. Because the local buckling criteria are stricter than the ultimate stress criteria, the multiplier for ultimate stress criteria is higher. To filter out appropriate designs in the early generations, we divided the violation of the local buckling criteria into three stages, thus improving the efficiency of the searching process. The overview of the optimization procedure is depicted in Figure 13.

$$f(\vec{t}) = Mass + US + LC, \tag{18}$$

$$US = \begin{cases} Mass * 10, & g_3 \geq \sigma_y^{S355} \\ 0, & g_3 < \sigma_y^{S355} \end{cases} \tag{19}$$

$$LC = \begin{cases} Mass * 5, & g_4 \geq 1.5 \\ Mass * 3, & 1.5 > g_4 \geq 1.0 \\ 0, & g_4 < 1.0 \end{cases} \tag{20}$$

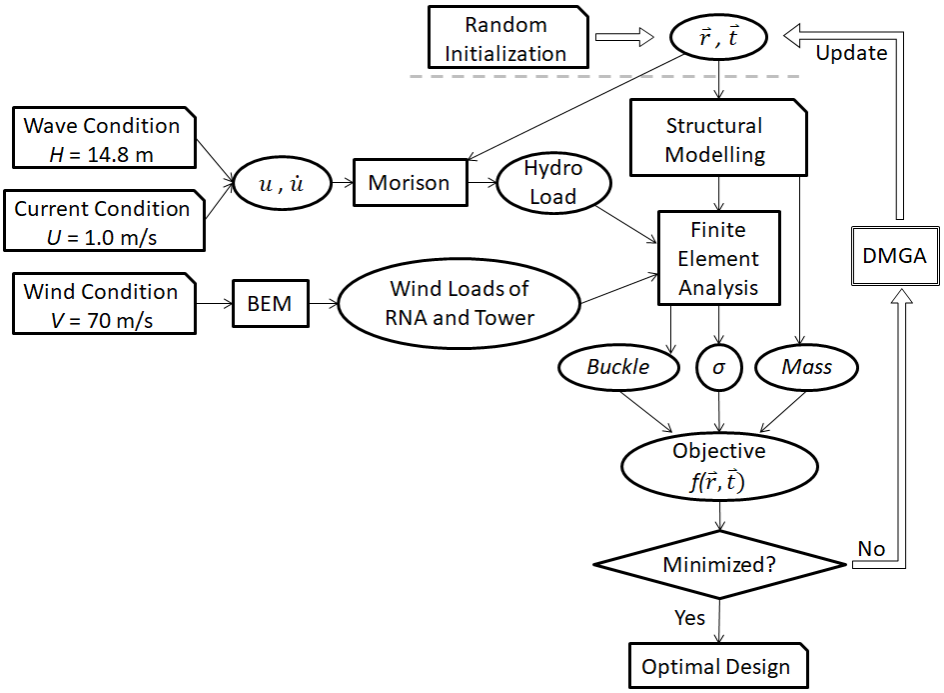

Figure 13. Optimization procedure.

In this section, we applied DMGA and other algorithms to optimize JSS, and compared the performance of design exploration among them to determine which algorithm is superior in the jacket optimization problem. We considered the ultimate stress and local buckling criteria of offshore structures as highly nonlinear constraints. The single objective function, the penalized mass of JSS, is also a nonlinear function in terms of sectional properties. To solve this complicated problem, the optimization was based on the evaluations of FEA model. By coding the algorithms in Python, we could repeatedly conduct analysis for different cases automatically. Finally, the convergence rates and resulting designs obtained by each algorithm were compared.

We selected the algorithms for comparison according to the results of the performance analyses. SGA and PSO are original algorithms of the DMGA. By ranking the algorithms, PS-GA and LMGA are considered better hybrid algorithms. All the algorithms with the same initial population set to 60 are coded in Python to run an optimization process in Abaqus software. Therefore, we set all the needed parameters for the JSS optimization, as listed in Table 4.

Table 4. Parameters for JSS optimization.

PS Step Size	PS Tolerance	ω for PSO	φ_1 for PSO	φ_2 for PSO
80/6 [1]	20/1.5 [1]	1	1	0.5

[1] These parameters are for the radius and thickness, respectively.

4.3. Results and Discussion

Under 1500 function evaluations, the optimization processes of all the algorithms are shown in Figure 14. The results indicate that although the DMGA's mass descended slower than that of the PS-GA in early generations, it plunged earlier than other algorithms in approximately 300 function evaluations. The mass decreased considerably until 400 function evaluations in DMGA, reaching the lightest mass among all the algorithms (Figure 14). Moreover, DMGA only consumed half of the function evaluations SGA spent. This indicated that DMGA was the most efficient among these hybrid algorithms. The two conventional algorithms, SGA and PSO, converged slower than hybrid algorithms with a higher mass. SGA outperformed PSO with a more favorable result and convergence rate, while PSGA outperformed LMGA. From the previous discussion in performance analysis, it can be seen that simply taking the original algorithms in turn cannot improve but slowed down the efficiency by PSO in this case. In terms of objective function, we found that DMGA reached the lightest mass. Even though the results of other three algorithms are only slightly heavier than DMGA's below 10 tons, DMGA consumed only one-third computational time. By comparing to the conventional SGA and PSO, DMGA saved 26 and 67 tons relatively. This indicates that with DMGA it is possible to save the computational cost significantly and acquire a competitive result. All the facts prove that DMGA is suitable to deal with an OWS optimization problem.

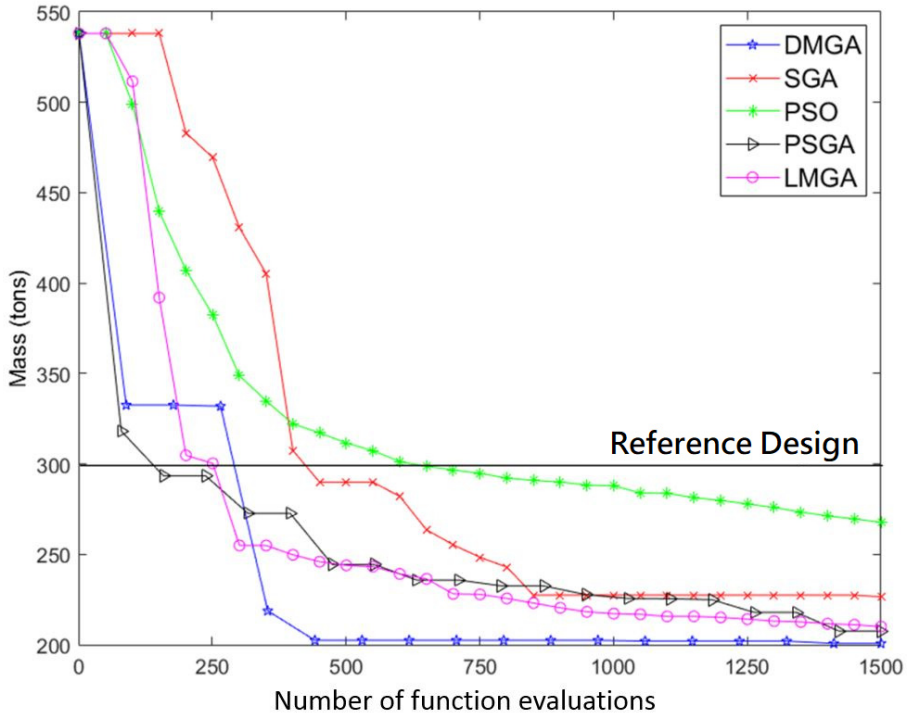

Figure 14. Mass versus the number of function evaluations in solving the jacket substructure optimization problem by hybrid algorithms.

Table 5 lists the results of the JSS optimization, including the thickness, radius, and total structural mass obtained by each algorithm.

Table 5. The results of the JSS optimization.

Algorithm / Member	Thickness (mm) / Radius (mm)	DMGA	SGA	PSO	PSGA	LMGA	Reference
Atmospheric brace		7.2	14	7.4	12	14.5	18.9
		100	100	165	100	101	203
Atmospheric upper leg		43.4	49	33.9	45	41.7	42.5
		463	447	673	463	513	600
Atmospheric lower leg		49	49	48.9	56	40	42.5
		390	492	557	363	450	600
Splash brace		6	12	18.6	6	16.6	12.8
		106	93	277	100	143	226
Splash leg		27.2	30.9	34.1	28	32.2	34
		569	527	521	566	510	600
Immersion brace		10.6	9	10	7	6	18.9
		119	115	387	124	173	203
Immersion upper leg		36.7	48	28.1	42	34.9	42.5
		565	461	674	507	580	600
Immersion lower leg		33	62	58.5	33	41.8	67.5
		837	546	611	837	644	650
Mass (ton)		200	226	267	207	209	300
Converged iterations		400	850	over 1500	1420	1400	-
Maximum stress (MPa)		229	228	228	229	235	203
Member of maximum stress		Immersion lower leg	Atmospheric upper leg	Immersion lower leg	Immersion lower leg	Immersion lower leg	Splash leg

Generally, the braces of the structure had a lower thickness and radius than the legs of the structure. Hybrid algorithms enlightened the mass more than the original ones due to the local search by PS, which regulated values concisely to a local optimum. DMGA obtained the lightest mass (200 tons), 12% lighter than that obtained by SGA, the most commonly used algorithm in JSS optimization. The dimensions of the DMGA's result suggested that the atmospheric members were thicker than those of the immersed ones, whereas the others stated the opposite suggestion. Manufacturing the large difference of diameter in a continuous leg might be impracticable and expensive, and the abnormal distribution of mass along the vertical direction may also cause problems due to its dynamic behavior. This is because the design variables are free in the design space and practically, more design constraints, such as the cost model and the natural frequency [36], should be included. Nevertheless, this indicates that DMGA obtained a different local optimum from the other algorithms, with this result potentially being the global optimum in the present tests. DMGA obtained a better result not merely dependent on the local search, but the wide search to find the global optimum. As more load cases are considered, the local search is expected to be restrained, contracting to the outstanding DMGA.

Figure 15 shows the stress contours of the optimized OWSs. Stress concentrated along the structural leg, mostly in the atmospheric zone and immersion zone with maximum value below 250 MPa, reacting to the direction of external forces. It is worth noting that DMGA strengthened the lower leg in the atmospheric zone and splash zone. The radius of structural members distributed unevenly, which was unusual among all the structures. Moreover, the upper braces were thicker than the lower ones. These evidences support that DMGA's result was closer to the global optimum than others. Apart from this, all nominal stresses in the contours were below the allowable stress. We can

deduce that the LC constraint governed this case rather than the US constraints. The same conclusion can be inferred that the maximum stress in the reference jacket was lower than other optimized ones. That means the buckling constraint had not been reached in the reference design, so there was an allowance to reduce weight from it.

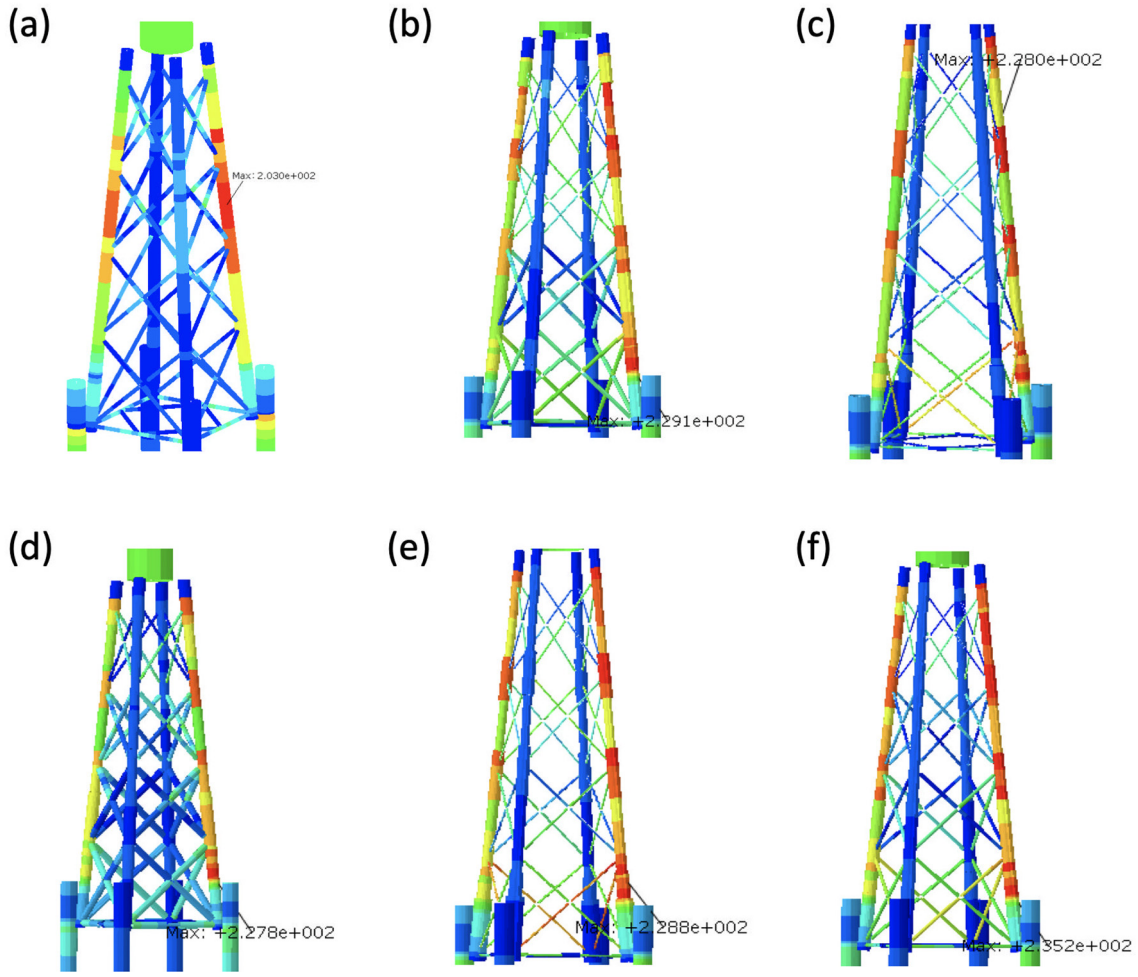

Figure 15. Nominal stress contour of each design, (**a**) Reference, (**b**) DMGA, (**c**) SGA, (**d**) PSO, (**e**) PSGA, and (**f**) Lifecycle model GA (LMGA).

5. Conclusions

This study introduced a novel hybrid algorithm called DMGA that combines GA, PSO, and PS in a divisional model. Each division in DMGA represents one algorithm as the searching scheme and shares information with the other divisions through migration. In performance analyses, the proposed algorithm was tested with 13 test functions to determine the rate of problem-solving as the effectiveness and the computational cost as efficiency. The results demonstrated that DMGA outperformed the other four hybrid algorithms. Moreover, DMGA was the most effective algorithm in seven test functions, obtaining high efficiency.

Because the purpose of this study was comparing hybrid algorithms for a JSS optimization problem, we applied these hybrid algorithms to reducing the mass of the JSS along with structural safety constraints. With a complex and nonlinear loading condition, the simulation analysis of the JSS requires considerable computational cost. Thus, the efficiency and ability to find a global minimum are critical. Finally, DMGA obtained the lightest structural mass, which was 12% (26 tons) lighter than that of SGA, and DMGA required only half number of evaluations to reach a superior result compared

with the other algorithms. These results indicate that DMGA is an effective and efficient algorithm, which can be applied to optimize the JSS for offshore wind turbines.

Author Contributions: Conceptualization, D.-P.L. and H.-H.H.; methodology, D.-P.L. and T.-Y.L.; software, D.-P.L. and T.-Y.L.; validation, D.-P.L. and T.-Y.L.; formal analysis, D.-P.L.; investigation, D.-P.L. and T.-Y.L.; data curation, D.-P.L. and T.-Y.L.; writing—original draft preparation, D.-P.L; writing—review and editing, T.-Y.L. and H.-H.H.; visualization, D.-P.L. and T.-Y.L.; supervision, H.-H.H.; funding acquisition, H.-H.H. All authors have read and agreed to the published version of the manuscript.

Funding: The research reported in this paper is supported by the Ministry of Science and Technology (MOST), Taiwan under Grant No. 108-2628-E-002-007-MY3, and by National Taiwan University, Taipei, Taiwan under Grant Nos. NTU-CDP-108L7743 and NTU-CDP-109L7725.

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

Appendix A

Table A1. Pseudo-code of the DMGA algorithm.

Procedure of DMGA
Setting parameters
Initialize random chromosomes/particles in three departments: *PS-GA*, *PSO* and *TM*
repeat
Evaluate the fitness of all chromosomes/particles
Replace two worst $x_{i\in\overrightarrow{PS-GA}}$ with the best of $x_{i\in\overrightarrow{PSO}}$ and $x_{i\in\overrightarrow{TM}}$ respectively
Replace two worst $x_{i\in\overrightarrow{TM}}$ with the worst of $x_{i\in\overrightarrow{PS-GA}}$ and $x_{i\in\overrightarrow{PSO}}$ respectively
p_g = ***Pattern search*** (the best of $x_{i\in\overrightarrow{PS-GA}}$)
Replace the worst of $x_{i\in\overrightarrow{PSO}}$ with p_g
Update $x_{i\in\overrightarrow{PSO}}$ according to Equations (2) and (3)
for each dimension j **do**
LOW:=$\{x^j_{i\in TM} \mid x^j_{i\in TM} <$ ***median***$(x^j)\}$
$HIGH$:=$\{x^j_{i\in TM} \mid x^j_{i\in TM} >$ ***median***$(x^j)\}$
low^j:=$(\Sigma$***fitness***$(x^j_{i\in TM}\in LOW))/n(LOW)$
$high^j$:=$(\Sigma$***fitness***$(x^j_{i\in TM}\in HIGH))/n(HIGH)$
if $(low^j > high^j)$ **then**
update $targeted\_range^j$ by Equation (7)
else-if $(low^j < high^j)$ **then**
update $targeted\_range^j$ by Equation (8)
else
update $targeted\_range^j$ by Equation (9)
end-if
end-for
$x_{i\in\overrightarrow{PS-GA}}$:=***SGA***$(PS\text{-}GA)$
$x_{i\in\overrightarrow{TM}}$:=***SGA***(TM)
until termination conditions are met

Appendix B

Table A2. List of test functions.

Function Name	Equation	Search Range	Tolerance %	Ref.
Ackley $D = 5$	$F_1(x) = -20\exp\left(-0.2\sqrt{\frac{1}{D}\sum_{i=1}^{D} x_i^2}\right) - \exp\left(\frac{1}{D}\sum_{i=1}^{D}\cos(2\pi x_i)\right) + e + 20$	$[-32,32]^D$	10^{-2}	[24]
Schwefel $D = 5$	$F_2(x) = 418.9829D + \sum_{i=1}^{D} -x_i \sin\sqrt{\lvert x_i\rvert}$	$[-500,500]^D$	10^{-2}	[24]
Rastrigin $D = 10$	$F_3(x) = \sum_{i=1}^{D}\left[x_i^2 - 10\cos(2\pi x_i) + 10\right]$	$[-10,10]^D$	10^{-2}	[24]
De Joung $D = 3$	$F_4(x) = \sum_{i=1}^{D} x_i^2$	$[-5,5]^D$	10^{-2}	[22]
Rosenbrock $D = 4$	$F_5(x) = \sum_{i=1}^{D}\left[100 * \left(x_i^2 - x_{i+1}\right)^2 + (x_i - 1)^2\right]$	$[-5,10]^D$	10^{-3}	[22]
Goldstein-Price $D = 2$	$F_6(x) = \left[1 + (x_1 + x_2 + 1)^2 * \left(19 - 14x_1 + 13{x_1}^2 - 14x_2 + 6x_1x_2 + 3{x_2}^2\right)\right] * \left[30 + (2x_1 - 3x_2)^2 * \left(18 - 32x_1 + 12{x_1}^2 - 48x_2 - 36x_1x_2 + 27{x_2}^2\right)\right]$	$[-2,2]^D$	10^{-2}	[22]
Easom $D = 2$	$F_7(x) = -\cos(x_1)\cos(x_2)\exp\left\{-\left[(x_1 - \pi)^2 + (x_2 - \pi)^2\right]\right\}$	$[-100,100]^D$	10^{-2}	[22]
Zakharov $D = 5$	$F_8(x) = \sum_{i=1}^{D} x_i^2 + \left(\sum_{i=1}^{D} 0.5ix_i\right)^2 + \left(\sum_{i=1}^{D} 0.5ix_i\right)^4$	$[-5,10]^D$	10^{-3}	[22]
Hartmann ($H_{6,4}$) $D = 6$	$F_9(x) = -\sum_{i=1}^{4} c_i \exp\left(\sum_{j=1}^{6} a_{ij}\left(x_j - p_{ij}\right)^2\right)$, $a_{ij} = \begin{bmatrix} 10.0 & 3.00 & 17.0 & 3.50 & 1.70 & 8.00 \\ 0.05 & 10.0 & 17.0 & 0.10 & 8.00 & 14.0 \\ 3.00 & 3.50 & 1.70 & 10.0 & 17.0 & 8.00 \\ 17.0 & 8.00 & 0.05 & 10.0 & 0.10 & 14.0 \end{bmatrix}_{ij}$; $c_i = \begin{bmatrix} 1.0 \\ 1.2 \\ 3.0 \\ 3.2 \end{bmatrix}_i$ $p_{ij} = \begin{bmatrix} 0.1312 & 0.1696 & 0.5569 & 0.0124 & 0.8283 & 0.5886 \\ 0.2329 & 0.4135 & 0.8307 & 0.3736 & 0.1004 & 0.9991 \\ 0.2348 & 0.1451 & 0.3522 & 0.2883 & 0.3047 & 0.6650 \\ 0.4047 & 0.8828 & 0.8732 & 0.5743 & 0.1091 & 0.0381 \end{bmatrix}_{ij}$	$[0,1]^D$	10^{-2}	[22]
Eggholder $D = 2$	$F_{10}(x) = \sum_{i=1}^{D}\left\{-(x_{i+1} + 47)\sin\left(\sqrt{\left\lvert x_{i+1} + \frac{x_i}{2} + 47\right\rvert}\right)\right\} - \sin\left(\sqrt{\left\lvert x_i + \frac{x_{i+1}}{2} + 47\right\rvert}\right)(-x_i)$	$[-512,512]^D$	10^{-1}	[23]
Schaffer $D = 2$	$F_{11}(x) = 0.5 + \frac{\sin^2\left(\sqrt{x_1^2 + x_2^2}\right) - 0.5}{\left[1 + 0.001\left(x_1^2 + x_2^2\right)\right]^2}$	$[-100,100]^D$	10^{-3}	[23]
Styblinski-Tang $D = 5$	$F_{12}(x) = 1/2 * \sum_{i=1}^{D}\left(x_i^4 - 16x_i^2 + 5x_i\right)$	$[-5,5]^D$	10^{-3}	[23]
Beale $D = 2$	$F_{13}(x) = (1.5 - x_1 + x_1x_2)^2 + (2.25 - x_1 + x_1{x_2}^2)^2 + (2.625 - x_1 + x_1{x_2}^3)^2$	$[-4.5,4.5]^D$	10^{-3}	[25]

Parameters for PS were defined as the power of 10 according to the search range to complete local searches with high efficiency. Next, by the number of local minima of each function, we set inertial weight ω as a multiple of 0.4 to control the activity of PSO. Similar to inertial weight, φ_1 and φ_2 are positive constants for the global experience term and self-experience term, respectively, which control the convergence of the algorithm. In this study, we set the values of these constants as a multiple of 0.5.

Table A3. Parameter settings for the algorithms in the performance analyses.

Test Function	PS Step Size	PS Tolerance	ω for PSO	φ_1 for PSO	φ_2 for PSO
Ackley	1	0.01	0.4	1	1
Schwefel	10	0.1	0.4	1	1
Rastrigin	1	0.001	0.8	1	1
De Joung	0.1	0.001	0.4	0.5	0.5
Rosenbrock	0.1	0.001	0.8	1	1
Goldstein-Price	0.1	0.001	0.4	0.5	0.5
Easom	10	0.01	0.4	1	1
Zakharov	1	0.01	0.4	1	1
Hartman($H_{6,4}$)	0.1	0.001	0.8	2	2
Eggholder	10	0.1	0.8	2	2
Schaffer	10	0.001	0.8	2	2
Styblinski-Tang	1	0.001	0.4	1	1
Beale	0.1	0.001	0.4	1	1

Appendix C

One of the safety constraints g_4 is calculated by the local buckling criteria of axial compression and bending in NORSOK N004 [35], shown as following equations, whichever is greater:

$$\frac{N_{Sd}}{N_{cl,Rd}} + \frac{\sqrt{M_{y,Sd}^2 + M_{z,Sd}^2}}{M_{Rd}} \leq 1.0, \tag{A1}$$

or

$$\frac{N_{Sd}}{N_{c,Rd}} + \frac{1}{M_{Rd}} \left\{ \left[\frac{C_{my} M_{y,Sd}}{1 - \frac{N_{Sd}}{N_{Ey}}} \right]^2 + \left[\frac{C_{mz} M_{z,Sd}}{1 - \frac{N_{Sd}}{N_{Ez}}} \right]^2 \right\} \leq 1.0, \tag{A2}$$

where N_{Sd} denotes design axial compression force, and $M_{y,Sd}$, $M_{z,Sd}$ represent in-plane and out-of-plane design bending moment. The above variables are obtained from the FEA results. N_{Ey} and N_{Ez} are Euler buckling strengths corresponding to the member y and z axes, respectively; $N_{c,Rd}$, $N_{cl,Rd}$ denote design axial compressive resistance and axial local buckling resistance, respectively; M_{Rd} is design bending moment resistance; C_{my} and C_{mz} represent reduction factors corresponding to the member y and z axes, respectively. These variables are calculated by further defined equations in the standard.

References

1. Delay, T.; Jennings, T. *Offshore Wind Power: Big Challenge, Big Opportunity*; Carbon Trust CTC743: London, UK, 2008.
2. Muskulus, M.; Schafhirt, S. Design optimization of wind turbine support structures-a review. *J. Ocean Wind Energy* **2014**, *1*, 12–22.
3. Chew, K.H.; Tai, K.; Ng, E.; Muskulus, M. Optimization of offshore wind turbine support structures using an analytical gradient-based method. *Energy Procedia* **2015**, *80*, 100–107. [CrossRef]
4. Chew, K.H.; Tai, K.; Ng, E.; Muskulus, M. Analytical gradient-based optimization of offshore wind turbine substructures under fatigue and extreme loads. *Mar. Struct.* **2016**, *47*, 23–41. [CrossRef]
5. Pasamontes, L.B.; Torres, F.G.; Zwick, D.; Schafhirt, S.; Muskulus, M. Support structure optimization for offshore wind turbines with a genetic algorithm. In Proceedings of the ASME 2014 33rd International Conference on Ocean, Offshore and Arctic Engineering, San Francisco, California, USA, 8–13 June 2014; p. V09BTA033-V09BT09A.
6. Schafhirt, S.; Zwick, D.; Muskulus, M. Reanalysis of jacket support structure for computer-aided optimization of offshore wind turbines with a genetic algorithm. In Proceedings of the The Twenty-Fourth International Ocean and Polar Engineering Conference: International Society of Offshore and Polar Engineers, Busan, Korea, 15–20 June 2014.

7. AlHamaydeh, M.; Barakat, S.; Nasif, O. Optimization of Support Structures for Offshore Wind Turbines Using Genetic Algorithm with Domain-Trimming. *Math. Probl. Eng.* **2017**, *2017*, 5978375. [CrossRef]
8. Gentils, T.; Wang, L.; Kolios, A. Integrated structural optimisation of offshore wind turbine support structures based on finite element analysis and genetic algorithm. *Appl. Energy* **2017**, *199*, 187–204. [CrossRef]
9. Kaveh, A.; Sabeti, S. Structural optimization of jacket supporting structures for offshore wind turbines using colliding bodies optimization algorithm. *Struct. Design Tall Spec. Build.* **2018**, *27*, e1494. [CrossRef]
10. Chehouri, A.; Younes, R.; Ilinca, A.; Perron, J. Review of performance optimization techniques applied to wind turbines. *Appl. Energy* **2015**, *142*, 361–388. [CrossRef]
11. Hofmeister, B.; Bruns, M.; Rolfes, R. Finite element model updating using deterministic optimisation: A global pattern search approach. *Eng. Struct.* **2019**, *195*, 373–381. [CrossRef]
12. Li, F.; Lam, K.Y.; Wang, L. Power allocation in cognitive radio networks over Rayleigh-fading channels with hybrid intelligent algorithms. *Wirel. Netw.* **2018**, *24*, 2397–2407. [CrossRef]
13. Sahu, R.K.; Panda, S.; Sekhar, G.C. A novel hybrid PSO-PS optimized fuzzy PI controller for AGC in multi area interconnected power systems. *Int. J. Electr. Power Energy Syst.* **2015**, *64*, 880–893. [CrossRef]
14. Gandomkar, M.; Vakilian, M.; Ehsan, M.A. Combination of genetic algorithm and simulated annealing for optimal DG allocation in distribution networks. In Proceedings of the Canadian Conference on Electrical and Computer Engineering, Saskatoon, SA, Canada, 1–4 May 2005; pp. 645–648.
15. Pourvaziri, H.; Naderi, B. A hybrid multi-population genetic algorithm for the dynamic facility layout problem. *Appl. Soft Comput.* **2014**, *24*, 457–469. [CrossRef]
16. Krink, T.; Løvbjerg, M. The lifecycle model: Combining particle swarm optimisation, genetic algorithms and hillclimbers. In *International Conference on Parallel Problem Solving from Nature—PPSN VII. PPSN 2002. Lecture Notes in Computer Science*; Springer: Berlin/Heidelberg, Germany, 2002; Volume 2439, pp. 621–630.
17. Turing, A.M. Computing machinery and intelligence. *Mind* **1950**, *59*, 433–460. [CrossRef]
18. Holland, J.H. Adaptation in natural and artificial systems. In *An Introductory Analysis with Application to Biology, Control, and Artificial Intelligence*; University of Michigan Press: Ann Arbor, MI, USA, 1975.
19. Xing, L.N.; Chen, Y.W.; Cai, H.P. An intelligent genetic algorithm designed for global optimization of multi-minima functions. *Appl. Math. Comput.* **2006**, *178*, 355–371. [CrossRef]
20. Kennedy, J. Particle swarm optimization. In *Encyclopedia of Machine Learning*; Springer: Berlin/Heidelberg, Germany, 1995; pp. 760–766.
21. Shi, Y.; Eberhart, R. A modified particle swarm optimizer. In Proceedings of the Evolutionary Computation Proceedings, 1998 IEEE World Congress on Computational Intelligence, Anchorage, AK, USA, 4–9 May 1998; pp. 69–73.
22. Hooke, R.; Jeeves, T.A. "Direct Search" solution of numerical and statistical problems. *J. ACM (JACM)* **1961**, *8*, 212–229. [CrossRef]
23. Chelouah, R.; Siarry, P. Tabu search applied to global optimization. *Eur. J. Oper. Res.* **2000**, *123*, 256–270. [CrossRef]
24. Mishra, S.K. Some new test functions for global optimization and performance of repulsive particle swarm method. *SSRN Electron. J.* **2006**, 1–24. [CrossRef]
25. Liang, J.J.; Qu, B.Y.; Suganthan, P.N.; Hernández-Díaz, A.G. *Problem Definitions and Evaluation Criteria for the CEC 2013 Special Session on Real-Parameter Optimization*; Technical Report; Computational Intelligence Laboratory, Zhengzhou University: Zhengzhou, China; Nanyang Technological University: Singapore, 2013; Volume 201212, pp. 3–18.
26. Beale, E. *On an Iterative Method for Finding a Local Minimum of a Function of More than One Variable*; Technical Report 25; Princeton University, Statistical Techniques Research Group: Princeton, NJ, USA, 1958.
27. Juang, C.F. A hybrid of genetic algorithm and particle swarm optimization for recurrent network design. *IEEE Trans. Syst. Man Cybern. Part B (Cybern.)* **2004**, *34*, 997–1006. [CrossRef] [PubMed]
28. Häfele, J. A Numerically Efficient and Holistic Approach to Design Optimization of Offshore wind Turbine Jacket Substructures. Ph.D. Thesis, Gottfried Wilhelm Leibniz Universität, Hanover, Germany, 2019.
29. Development Plan of Wind Power in Taiwan (English Translation of Chinese Title). Available online: https://energywhitepaper.tw/upload/201712/151376189216283.pdf (accessed on 29 May 2020).
30. International Electrotechnical Commission (IEC). *IEC 61400-3 Part 3: Design Requirements for Offshore Wind Turbines*; IEC: Geneva, Switzerland, 2009.

31. Det Norske Veritas and Germanischer Lloyd (DNVGL). *Loads and Site Conditions for Wind Turbines*; DNVGL-ST-0437; DNVGL: Oslo, Norway, 2016.
32. Jonkman, J.; Butterfield, S.; Musial, W.; Scott, G. *Definition of a 5-MW Reference Wind Turbine for Offshore System Development*; NREL/TP-500-38060; National Renewable Energy Lab (NREL): Golden, CO, USA, 2009.
33. Offshore Wind Power Demonstration Incentive Program–Annex 3: Requirements of Specification (English Translation of Chinese Title). Available online: https://law.moj.gov.tw/LawClass/LawAll.aspx?pcode=J0130063 (accessed on 29 May 2020).
34. Morison, J.; Johnson, J.; Schaaf, S. The force exerted by surface waves on piles. *J. Pet. Technol.* **1950**, *2*, 149–154. [CrossRef]
35. Chakrabarti, S.K. *Hydrodynamics of Offshore Structures*; WIT Press: Southampton, UK, 1987.
36. Constrained Optimization—Wikipedia. Available online: https://en.wikipedia.org/wiki/Constrained_optimization (accessed on 29 May 2020).
37. Standards Norway. *NORSOK Standard N-004 Design of Steel Structures*; Standards Norway: Lysaker, Norway, 2004.