buildings

MDPI

Article

Evaluation of Metaheuristic-Based Methods for Optimization of Truss Structures via Various Algorithms and Lèvy Flight Modification

Gebrail Bekdaş, Melda Yucel and Sinan Melih Nigdeli *

Department of Civil Engineering, Istanbul University–Cerrahpaşa, 34320 Istanbul, Turkey; bekdas@istanbul.edu.tr (G.B.); melda.yucel@yahoo.com.tr (M.Y.)
* Correspondence: melihnig@istanbul.edu.tr

Abstract: Truss structures are one of the major civil engineering members studied in the optimization research area. In this area, various optimization applications such as topology, size, cost, weight, material usage, etc., can be conducted for different truss structure types. In this scope with the present study, various optimization processes were carried out concerning two different large-scale space trusses to minimize the structural weight. According to this state, three structural models provided via two different truss structures, including 25 bar and 72 bar truss models, were handled for evaluation of six different metaheuristics together with the modification of Lèvy flight for three of the algorithms using swarm intelligence by considering both constant and variable populations, and different ranges for iterations, too. Additionally, the effects of the Lèvy flight function and whether it is successful or not in terms of the target of optimization were also investigated by comparing with some documented studies. In this regard, some statistical calculations were also realized to evaluate the optimization method performance and detection of optimum values for any data stably and successfully. According to the results, the Jaya algorithm can handle the optimization process successfully, including the case, without grouping truss members. The positive effect of Lèvy flight on swarm-based algorithms can be seen especially for the gray wolf algorithm.

Keywords: truss structures; optimization; metaheuristics; Lèvy flight; swarm intelligence; Jaya algorithm

Citation: Bekdaş, G.; Yucel, M.; Nigdeli, S.M. Evaluation of Metaheuristic-Based Methods for Optimization of Truss Structures via Various Algorithms and Lèvy Flight Modification. *Buildings* **2021**, *11*, 49. https://doi.org/10.3390/buildings11020049

Academic Editor: Oldrich Sucharda
Received: 29 December 2020
Accepted: 29 January 2021
Published: 31 January 2021

1. Introduction

In civil engineering, the most significant issue is to provide the required safety for any engineering structure. Of course, this state must be actualized by designers or engineers without putting people's lives in danger; besides, the design must be cost-effective and also sustainable for eco-friendly structures and their members. However, to actualize this is not so fast and easy because of the desired and expected conditions that may not be able to be provided at the same time and with only a single step. On the other respect, the results obtained may not suitable or enough economic for the required conditions. For this reason, iterative processes, which benefit from both determining an optimistic design in terms of cost, effort, etc., and prevent spending much time, gain importance. From the past to nowadays, metaheuristic algorithms, which came as one of the optimization techniques and have been improved in the direction of some special features sourced on nature or science, are one of the best methods that can be selected for the mentioned state.

In this regard, several applications were performed via a wide range of metaheuristics in civil engineering, especially for structural engineering. As an example, in 2013, a study was carried out that is concerned with the generation of the best-tuned mass damper (TMD) parameters for structures. In the mentioned study, analyses were operated under different historical earthquake conditions by using harmony search (HS) algorithm to make real of this [1]. Parcianello et al. (2017) improved the viscous damper model for the frames depending on an optimization process conducted with genetic algorithms (GA) to

increase of seismic performance of these structures [2]. On the other hand, Talatahari et al. (2015) developed a hybrid optimization tool by using eagle strategy (ES) and differential evolution (DE) algorithms to minimize the weight for different steel frame structures [3]. In addition, they evaluated the optimization success of this algorithm compared with DE. Gholizadeh and Ebadijalal (2018) benefited from a recently developed metaheuristic algorithm, which is named as center of mass optimization (CMO), to optimally arrange bracings on steel frames under seismic loading [4]. With the other respect, weight minimization of eccentrically braced frames was performed in the direction of seismic performance-based analysis by Fathali et al. (2020). Four algorithms, including accelerated water evaporation optimization (AWEO), PSO, with classical (CBO) and enhanced (ECBO) colliding bodies optimization, were utilized to observe the performance of this method [5]. Moreover, Kayabekir et al. (2016) presented a book where optimum designs were generated for reinforced concrete structures containing slender columns, shear walls, and cylindrical walls formed as post-tensioned axially symmetric, etc., through the usage of metaheuristic algorithms [6]. Vaez and Qomi (2018) carried out a study, which is related to providing minimum weight for reinforced concrete shear walls by investigating the optimum placement and diameter of steel bars together with wall properties by using PSO, FA, whale optimization algorithm (WOA), and crow search algorithm (CSA) [7]. In the study performed by Sheikholeslami et al. (2016), two metaheuristics, including firefly algorithm and harmony search, were hybridized and applied to two different reinforced concrete retaining walls to provide the best cost [8]. Optimum design for a cantilever retaining wall was also carried out with the aim of cost minimization by Aydogdu (2017), where peak ground acceleration was considered towards this object by combining of biogeography-based optimization method with Lèvy flight [9]. Additionally, the design of retaining wall with the optimum cost was realized depend on seismically performance based by handling an enhanced kind of genetic algorithm (non-dominated sorting).

Similarly, various optimization studies were also performed for truss structures too. One of these is a study, which was carried out to provide the minimum weight for space truss structures by Camp (2007) through using the big bang–big crunch (BB–BC) optimization algorithm [10]. While executing this process, deflection and stress limitations besides material conditions were considered. In the year 2011, another study was performed for trusses in the direction of minimization of structure mass by providing of optimum size and shape of them [11]. To make real of this, an optimization technique, which is one of the oldest and called particle swarm optimization (PSO), was utilized, and it was applied for four different structure models. Moreover, Miguel and Miguel (2012) carried out a work which is related to optimization of shape and size of trusses by using nodal coordinates and section areas of bars, respectively, to reach minimum weight [12]. For this aim, they used two optimization tools, including the firefly algorithm (FA) and harmony search (HS), for analyzing four different truss models. In 2015, Bekdaş et al. (2015) applied a metaheuristic known as flower pollination algorithm (FPA) to generate the best truss size by minimizing structure weight, and they used three truss models containing planar and space form to actualize this process [13]. On the other hand, Kaveh and Ghazaan (2017) used a metaheuristic algorithm to optimize truss structures to improve the dynamic performance of them under frequency constraints [14]. For this reason, they benefited from a method, which is known as vibrating particles systems (VPS), and was developed by inspiring from dynamic behavior of structures. Tejani et al. (2018) also solved the problem related to multiobjective optimization of five different truss structure models from the literature [15]. In this respect, they applied the symbiotic organisms search (SOS) algorithm by combining the multiobjective adaptive control technique.

The widest application area of metaheuristic-based optimization is truss structures in structural engineering. Furthermore, in Table 1, a comparison is carried out by generating a summarization of optimization properties about some literature studies, which are benefited from the validation of the present results. The current study and compared

documented methods are compared to each other in terms of the variety of the number of the used algorithms, the number of design variables, etc.

Table 1. Comparing some literature research and the present study.

Researchers	Number of Used Classical Algorithms	Number of Hybridized/ Modified Algorithms	Algorithm	Number of Investi-gated Truss Models	Number of Maximum Design Variables	Biggest Number of Compared Other Methods	Citation
Camp (2007)	1	-	Big bang–big crunch (BB–BC) optimization	3	16	4	[10]
Dede et al. (2011)	1	-	Genetic algorithm (GA)	4	96	13	[16]
Gandomi et al. (2013)	1	-	Krill herd (KH)	1	8	10	[17]
Degertekin and Hayalioglu (2013)	1	-	Teaching-learning based optimization (TLBO)	4	29	8	[18]
Kaveh et al. (2014)	-	1	Hybrid particle swarm and swallow swarm optimization (HPSSO)	6	29	6	[19]
Kaveh et al. (2014b)	-	1	Chaotic swarming of particles (CSP)	4	59	5	[20]
Camp and Farschin (2014)	-	1	Modified teaching-learning based optimization (TLBO)	3	26	7	[21]
Bureerat and Pholde (2016)	-	1	Adaptive Differential evolution algorithm (ADEA)	4	29	6	[22]
Degertekin et al. (2017)	1	-	Heat transfer search (HTS)	3	29	8	[23]
Bekdaş et al. (2017)	1	-	Flower pollination algorithm (FPA)	2	72	0	[24]
Present Study	6	3	Defined in Section 2	3	25	7	-

In the current study, two different space truss structure models as 25 bar and 72 bar that are generally used as the benchmark problems, were handled to ensure optimum design parameters, including section areas and also objective functions such as minimum cost or weight, etc. In this regard, three separate cases were emphasized by considering the numbers of increment of design parameters. According to this, the first and second one is related to combining/grouping of structure bars for truss models, besides that, the last is the case, where grouping for the 25 bar model is not realized. Six different metaheuristic algorithms and three improved versions of them using Lèvy flight as a novel application were applied to the mentioned cases by generating two sub-cases of different maximum iteration and population numbers. As given in Table 1, the present study includes the application of 9 algorithms with three novel modifications. In addition to that, a comparative investigation using nine applied methods and seven documented methods was presented by choosing the same optimum design benchmark problems.

2. Materials and Methods

When the general of nature is considered, it is understood that livings have features developed for various aims such as surviving, feeding, continuity of species. To see these features, many examples such as that fox benefits from a magnetic area of the world while it is hunting, chameleon changes color intending to hide from danger; cuckoo birds use other bird's nests for continuity of self-species and hedgehog throws the quills by stretching itself under danger, are possible. If when all these processes are analyzed, it is seen that livings change their defense or attack mechanisms as conveniently to the conditions and uses a kind of species-specific heuristic optimization, which ensures that using of

limited opportunities exist in themselves under the fittest time and form to maintain of vital activities.

These heuristic optimization processes belonging to livings in nature have engaged the attention of researchers working on basic sciences, and they generated various algorithms, which express these processes mathematically as the most frequently and commonly used in the literature from these algorithms called metaheuristics are explained in headings taken below and employed in the study.

The employed algorithms are flower pollination algorithm (FPA), artificial bee colony (ABC) algorithm, bat algorithm (BA), Jaya algorithm (JA), gray wolf optimization (GWO) and harmony search (HS). These algorithms have unique features and imitations from a process. These are detailly explained in the subsections of Section 2, but major generations and differentiation are as follows:

- FPA, ABC, BA and GWO are nature-inspired algorithms. HS is a music-inspired one, while JA does not use a direct imitation. Jaya word means victory in Sanskrit. Due to that, a bond can be only generated by assuming the reaching of an optimum result as a victory;
- FPA uses Lèvy distribution in its classical form. Since wolves, bees and bats can also act as random flying or moving members, modified versions of these algorithms with Lèvy distribution that characterize the random flight are investigated;
- Generally, metaheuristic algorithms have two stages of optimization using a probability to select one of these stages (phases) in an iteration. ABC is a three-stage (phase) algorithm, while JA has only a single phase. The others are classical two-phase algorithms;
- JA has no user-defined specific parameter in the formulation, while the others need parameters in formulations and selection of a stage;
- BA uses a three-step formulation (frequency, velocity and new solution) to update a solution.
- GWO uses three unique, different solutions with different calculations. These solutions are named with three types of wolves.

In Appendix A, the commonly used type-specific parameters belonging to each metaheuristic method and some common expressions concerned with candidate solutions for each design variable can be seen. In addition, the used functions in equations of algorithm optimizations are indicated in Appendix A.

2.1. Flower Pollination Algorithm (FPA)

Plants, especially flowering plants, can gain attract to self of some insect species like bees, flies, etc., because of that they have stimuli such as special color, smell and various aromatic secretories. In addition to these features, insects also contribute to the pollination process, which is required for ensuring the continuity of species by the run to flowers with the help of nature-sourced effects like wind, water, etc.

The flower pollination algorithm (FPA), which was developed by inspiration from this process by Yang [25], is one of the metaheuristic algorithms frequently used nowadays. In FPA, four different rules, which are related to the property of pollination process, the behavior of pollination, and flower constancy, and formalized by inspired from flowery plants, are kept insight [13,26,27]:

- Cross-pollination is realized via the transfer of pollen between flowers of different plants from the same species. Pollen carriers (pollinators) suit to rules Lèvy distribution (Equation (1)) by jump with far steps or fly. This process is called global pollination;
- Self-pollination occurs due to the pollen transferring within the self of a flower or between different flowers of the same plant. This pollination kind is local pollination;
- The case of flower constancy is the cooperation among pollen carriers with flower types. This is a development within the process of flower pollination;
- Local and global pollination is controlled with a probability value, which is named switch probability and has a value between 0 and 1.

In the optimization process, which is performed via applying all these rules, two different ways are followed to obtain the optimum values. To make real this, the type of search must be determined by controlling search change/switch probability (sp.), which is one of the FPA parameters. This search type is named as:

- The global search that solutions are determined by search from more extent area, if sp is bigger than a randomly generated number;
- The local search process that solutions are searched from a smaller area if this value sp is smaller than the generated random number between 0 and 1 (rand).

The value of the ith design variable of the jth value of population including *nf* number of flowers ($X_{i,j}$) is updated as the new solution ($X_{i,\ new}$) as given below. $X_{i,g_{best}}$ is the best current solution of the ith design variable.

$$Levy = \left(\frac{1}{\sqrt{2\pi}}\right)(\text{rand})^{-1.5}\ e^{\left(-\frac{1}{2\ \text{rand}}\ \right)} \tag{1}$$

$$X_{i,\ new} = \begin{cases} \text{sp} > \text{rand}, & X_{i,j} + Levy\left(X_{i,g_{best}} - X_{i,j}\right) \\ \text{sp} < \text{rand}, & X_{i,j} + \text{rand}\ (X_{i,m} - X_{i,n}) \end{cases} \tag{2}$$

$$\text{n} = \text{ceil}\ (\text{rand} \times nf) \tag{3}$$

$$\text{m} = \text{ceil}\ (\text{rand} \times nf) \tag{4}$$

2.2. Artificial Bee Colony (ABC) Algorithm

In 2005, the artificial bee colony (ABC) algorithm introduced by Karaboğa was developed through simulated the food source searching behaviors of bee colonies [28].

In bee colonies, honey bees are divided into three different categories as a worker, onlooker, and scout. Moreover, in the algorithm, these categories represent the negative feedback, positive feedback, and random motions, respectively. Initial food sources are produced randomly in the search space of the problem. Half of the honeybees that have the aim to provide the increase of the substantiality of nectar in the hive are worker bees. The other half contains onlooker bees, and flocking behavior around the food source starts with worker bees. Worker bees record each food source to the memories, and information is shared with onlooker bees waiting in the hive. According to shared information, onlooker bees collect the food sources within the near-environment of the hive; scout bees collect the ones in long-distance from the hive. Onlooker and scout bees share the food source information with worker bees by a return to the hive. If the information on the new food source is better than the information on the initial food source's position, this is updated [29–31].

In this way, the ABC algorithm can deliver a solution to various optimization problems thanks to the simulation with the natural process, which is the maximization of nectar amount by determining the position of food sources optimally by honey bees.

Some assumptions are applied in this algorithm, too [32]. These are given below:

- It is accepted that number of worker and onlooker bees are equal to each other in the total bee population;
- The food sources express candidate solutions and are assumed that each bee completely consumes this source by going to a single food source. Hence, the food source number is half of the total bee number;
- Later, worker bees transform into scout bees to search for the new ones substituted for finished foods.

There are four separate stages for expressing this process via the ABC algorithm. These are determining of initial food sources, the worker bee stage that found of the new sources by worker bees, onlooker bee stage that evaluated nectar qualities of new sources, and finally, scout bee stages that found the new ones, in the case that exist finished food.

About improving initial food sources, first, the worker bee stage is performed. For this; one food source is selected as randomly (n), and the position of the old source is updated with a probability value ($\phi_{i,j}$) between the range of $[-1, 1]$ for a design variable/parameter determined randomly (p) for a problem with *vn* design variables/parameters, and source nectars are calculated again. From sources that their positions were updated, all of the food sources that are better than the initial ones are updated by changed with the old food source. This process for modification of the design variable of the jth population ($X_{p,j}$) is carried out via equations expressed below:

$$X_{p,new} = X_{p,j} + \phi_{i,j} \left(X_{p,j} - X_{p,n}\right) \tag{5}$$

$$n = \text{ceil} \left(\text{rand} \times eb\right) \tag{6}$$

$$p = \text{ceil} \left(1 + (vn - 1)\ \text{rand}\right) \tag{7}$$

The second stage is the onlooker bee stage. In this stage, onlooker bees are kept informed by worker bees with regards to food sources' nectar amount. The food quality/rate/possibility according to the nectar amount of each source is calculated and evaluated by onlooker bees. This operation is related to the selection of ones, which have a high ratio of nectar from renewed food sources, and in this way, the sources, which are rich in nectar, are ensured the improvement of them as continuously by determined. In addition, this process is performed via Equation (8) in case that food possibility (Equation (9)) is bigger than a number randomly generating. $nectar_j$ (Equation (10)) is quality value for jth food sources (candidate solution) calculated depending to the objective function and considering of problem type; F_j is objective function value belonging jth solution and $food\ possibility_j$ is nectar rate existing in each source (rate of nectar quality):

$$\text{if rand} < food\ possibility_j,\ X_{p,\ new} = X_{p,j} + \phi_{i,j} \left(X_{p,j} - X_{p,n}\right) \tag{8}$$

$$food\ possibility_j = \frac{nectar_j}{\sum_{j=1}^{fsn} nectar_j} \tag{9}$$

$$nectar_j = \frac{1}{1 + F_j} \tag{10}$$

The process of searching for new sources by abandoning sources that cannot be optimized (namely nectar of them) is named as scout bee stage. Improving parameter (ip) belonging to each source is controlled according to a constant value of source improvement limit (SIL), which is defined at the start of the optimization process to use in this stage. By scout bees, determination of new ones for each source that has ip values exceeding the value of SIL is realized utilizing Equation (11) as below for the defined maximum ($X_{i,max}$) and minimum ($X_{i,min}$) values. For modification with Lèvy distribution, the rand function is replaced with Equation (1).

$$\text{if } ip_J > \text{SIL},\ X_{i,new} = X_{i,min} + \text{rand}\ \left(X_{i,max} - X_{i,min}\right) \tag{11}$$

2.3. Bat Algorithm (BA)

Small bats known as micro bat perform echolocation behavior acting as a kind of radar to locate their prey, protect from obstacles, and can detect the place of hollows/clefts where they lived at night. These bats listen to echoes returning from objects by transmitting a very noisy sound. In addition, the frequency of transmitted sound shows alteration according to features of bats. Bat algorithm (BA), which is a metaheuristic method, is developed by Yang through be idealized of echolocation behavior and these features belonging to bats [33]. Furthermore, bats fly with variable frequency values, loudness, and velocity, which can be used in designing update equations of algorithms [27,34].

On the other hand, some assumptions are required to be able to use the BA algorithm in the optimization process, as in the other algorithm types [33,35]. These assumptions are as below:

- All bats use audio echo to sense distance;
- Bats move randomly at any X_i location via V_i velocity by using a sound, which has a constant frequency, variable λ wavelength, and loudness with A_0 value, to search for the prey. They can adjust the wavelength of emitted pulses or frequency automatically, and the pulse emission rate (r) depending on closeness to its target ($r \in [0, 1]$);
- Although loudness value can change in different ways, this value changes between an extremely high (positive) initial value (A_0), and a constant minimum value (A_{min}). A_0 is 1 due to that bat search its prey with a very loud sound in the beginning; also, when it is considered that bat just found the prey, and abandons to giving out a sound as temporarily, A_{min} can be taken as 0.

In the direction of these assumptions, it is required that each bat have different loudness and pulse emission value; besides, some different properties must observe for determining new values (locations) belonging to candidate solutions in the optimization process performing via the BA algorithm. These are frequency (f_j for the jth member with the minimum; f_{min} and the maximum; f_{max}) and velocity vectors, and the process is carried out with equations as below, respectively:

$$f_j = f_{min} + (f_{max} - f_{min})\text{rand} \tag{12}$$

$$V_{i,new} = V_{i,j} + \left(X_{i,j} - X_{i,g_{best}}\right) f_j \tag{13}$$

$$X_{i,new} = X_{i,j} + V_{i,new} \tag{14}$$

An iterative replacement (update) is in question for new locations designated for bats (solution value of design variables). This replacement is a process called local search, and calculation is made with the help of Equation (15):

$$\text{if rand} > r_j, \; X_{i,new} = X_{i,g_{best}} + (-1 + 2\text{rand})\, A_{mean} \tag{15}$$

On the other hand, the values of loudness and pulse emission rate should be updated as long as iterations progressed. The reason for it is that distances of bats to foods or their prey change per the updated locations of bats. About this subject, generally, pulse emission rate increases according to a decrease of loudness when bat found the prey, according to the expression of Yang [33]. In this direction, parameters are updated along with iterations via Equations (16) and (17):

$$A_{j,new} = \alpha\,(A_{min} + (A_0 - A_{min})\text{rand}) \tag{16}$$

$$r_{j,new} = r_j^0\,(1 - e^{-\gamma t}) \tag{17}$$

In Appendix A, the calculated and utilized expressions for the optimization process were given except the commonly used expressions. In the Lèvy improved BA, the A_{mean} value in Equation (15) is replaced with Equation (1).

2.4. Jaya Algorithm (JA)

Jaya algorithm, which is recently developed by Rao (2016), and has a working principle similar to the teaching-learning based optimization (TLBO), is one of the metaheuristic methods [36]. This algorithm always tries to be closer to the approach of being optimal. In this regard, the main targets of the algorithm are both reaching the best solution and moving away from the worst solution ($X_{i,g_{worst}}$). On the other hand, the algorithm takes the name from Jaya, which is a Sanskrit word, meaning victory. It is harmonious with this operation, and it is aimed to achieve optimization of the solution.

When the Jaya algorithm is compared with the other algorithms, generally, it evaluates fewer functions to obtain the best same result. For this reason, it is required fewer operations than others in the process of convergence to the ideal solution. However, the application of the algorithm is simple and also not including special parameters, which are considered superiorities [36]. However, the possibility that the algorithm locks to local optimum increases due to the algorithm searches the best and worst results around a smaller area compared to the other methods, and this case may cause to be not able to evaluate better results.

In each iteration, a single-stage is enough for all variables to obtain a new optimum solution. Equation (18) utilizing for this is as below:

$$X_{i,\,new} = X_{i,j} + \text{rand}\left(X_{i,g_{best}} - \left|X_{i,j}\right|\right) - \text{rand}\left(X_{i,g_{worst}} - \left|X_{i,j}\right|\right) \quad (18)$$

2.5. Gray Wolf Optimization (GWO)

Mirjalili et al. (2014) developed an algorithm, which is called gray wolf optimization (GWO), and it is a kind of metaheuristic optimization method by considering the hierarchy of leadership of gray wolves, which usually live in groups, and have hunting mechanism in real nature [37].

In nature, gray wolves, which underlie this algorithm, create groups containing 5–12 members on average. Wolves categorize into four different as alpha, beta, delta, and omega wolves due to the existence of hierarchy between themselves. Group leader named as the alpha wolf, has responsibilities such as hunting, determining of sleep place with waking time by managing the other wolves in the group, too. In this respect, it is required that the alpha wolf, which is in the leading position, is the best member in terms of managing the pack, not the strongest member. In addition, this shows that the case of organized and discipline keeping in the pack is more significant than power. Wolf at the second level of the hierarchy, is the beta wolf and participates in the group as the alpha wolf's assistant in many respects. Moreover, when the case that alpha wolf dies or is at a very advanced age, the beta wolf is considered as the most possible member as a candidate, which will be able to replace. In the group, the main task of this wolf is both providing to keep informed of delta (δ) and omega (ω) wolves by transferring the instructions transmitted to the leader and giving their instructions to these wolves situated in lower level than itself by applying the directions taken from alpha. In the third and fourth steps of hierarchy, delta and omega, which are less strong wolves, respectively, and the delta wolf can gain an advantage over only omega wolves. For this reason, it can be said that the weakest member in the group is the omega wolf in terms of leadership/dominance. The representative demonstration of social hierarchy among gray wolves takes place in Figure 1 [38,39].

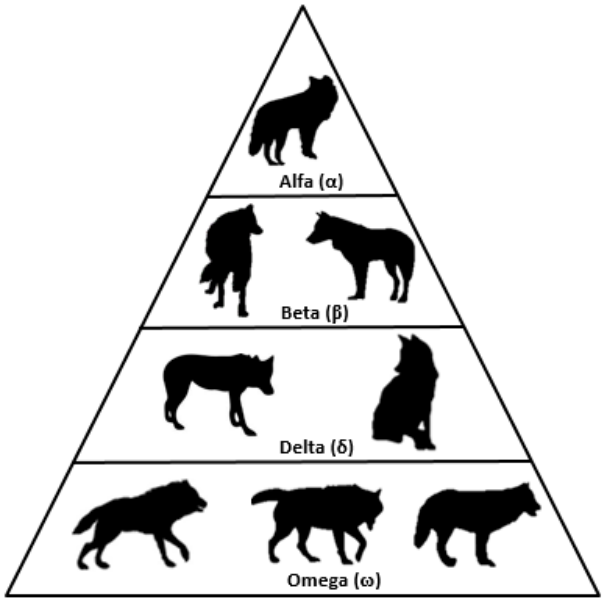

Figure 1. Social hierarchy pyramid among gray wolves [39].

On the other hand, group hunting is a social behavior that gray wolves performed in real nature, and also, this natural behavior was benefited for algorithm design-oriented at optimization process. In this respect, first, the strongest three wolves need to be determined among wolves in the group. Here, the best three solutions set is determined from among each member in the initial wolf ensemble; in other words, candidate solutions, and alpha (α) as the leader-member (the best convenient member in terms of the objective function), the second strong member beta (β) and third one delta (δ) solutions are defined. All solutions except three candidate solutions are represented by the other wolves that remained in the pack, and these members are considered omega (ω) solutions. Hence, hunting behavior reflecting the optimization process is directed by α, β, and δ wolves. In addition, there is a certain order followed by wolves during hunting, and according to this order, first, the prey is followed, then encircled by wolves. In the meantime, $\vec{D}$, which expresses the distance between prey and any wolf surrounding it, is calculated with Equation (19). Moreover, wolves can update their locations around the prey randomly, depending on prey (Equation (20)) [37,40]. In addition, for Lèvy modification of GWO, rand function existing in $\vec{A}$ vector formulation (Equation (22)) is changed via Equation (1):

$$\vec{D} = \left| \vec{C}\, X_{i,pr} - X_{i,j} \right| \tag{19}$$

$$X_{i,new} = X_{i,pr} - \vec{A}\vec{D} \tag{20}$$

$$\vec{C} = 2\ \text{rand} \tag{21}$$

$$\vec{A} = 2\vec{a}\ \text{rand} - \vec{a} \tag{22}$$

$$\vec{a} = 2 - 2\frac{t}{\text{stopping criteria}} \tag{23}$$

As mentioned before, the hunting process is directed by α, β, and δ wolves. Figure 2, which shows this case more clearly, takes in below. In this stage, hunting is performed by wolves attacking the surrounded prey. However, as seen in the figure, first, it is required that hunting be directed healthfully and determining of locations (according to the prey) of

α, β and δ wolves that are assumed to have information well about the potential position of the prey. For this, the following equations are applied (Equations (24)–(29)), respectively:

$$\vec{D}_{\alpha} = \left| \vec{C}_1 X_{i,\alpha} - X_{i,j} \right| \tag{24}$$

$$\vec{D}_{\beta} = \left| \vec{C}_2 X_{i,\beta} - X_{i,j} \right| \tag{25}$$

$$\vec{D}_{\delta} = \left| \vec{C}_3 X_{i,\delta} - X_{i,j} \right| \tag{26}$$

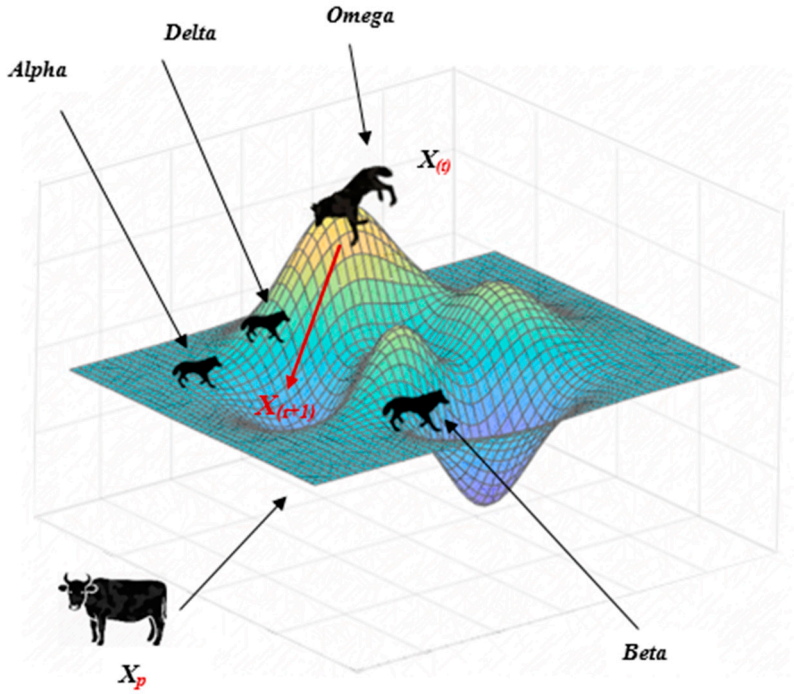

Figure 2. Determining the new position of any omega wolf concerning prey according to wolves α, β, and δ [40].

In these equations, $\vec{D}_{\alpha}$, $\vec{D}_{\beta}$ and $\vec{D}_{\delta}$ are the distance between any gray wolf and alpha, beta and delta wolves, respectively:

$$X_{i,\alpha_{new}} = X_{i,\alpha} - \vec{A}_1 \vec{D}_{\alpha} \tag{27}$$

$$X_{i,\beta_{new}} = X_{i,\beta} - \vec{A}_2 \vec{D}_{\beta} \tag{28}$$

$$X_{i,\delta_{new}} = X_{i,\delta} - \vec{A}_3 \vec{D}_{\delta} \tag{29}$$

In the above equations, respectively, $\vec{A}$ and $\vec{C}$ vectors are expressed individually for alpha, beta and delta wolves, and the approximate distance between any omega (ω) wolf, namely the current solution, and α, β and δ wolves can be determined via Equations (27)–(29). The latest updated position of the current solution is calculated as in Equation (30):

$$X_{i,new} = \frac{X_{i,\alpha_{new}} + X_{i,\beta_{new}} + X_{i,\delta_{new}}}{3} \tag{30}$$

The wolf, which updates its position, is ready to attack prey. However, also, there are some rules about mathematically expressing the case that wolf can realize the attack.

Accordingly, in this algorithm, the $\vec{a}$ value was reduced for the wolf that can approach (decreasing of the distance between them) to prey. In addition, $\vec{A}$ vector, which determines the case of attacking to prey by a wolf, utilizing this decreasing and take values between $-\vec{a}$ and $\vec{a}$. Hence, the environment, which is taken place in the hunting process, can be expressed as in Figure 3. On the other hand, a new position of hunter wolf can take values between the position of prey and own current position. As a summary, the case that wolf can attack prey actualizes to the condition in Equation (31) [14,41]:

$$\text{if } |\vec{A}| < 1,\ X_{i,\text{ new}} = X_{i,pr} \tag{31}$$

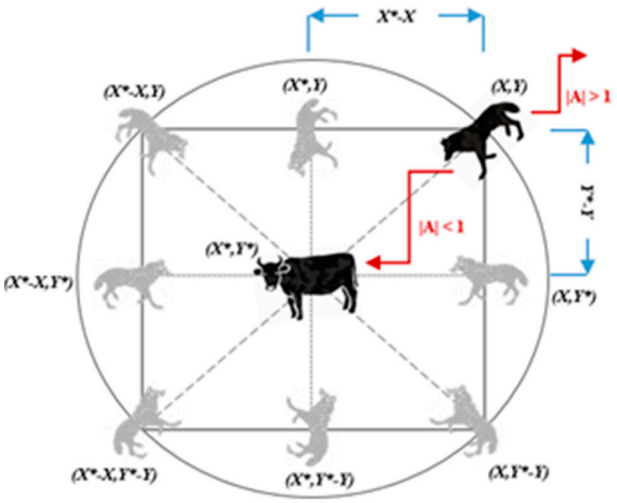

Figure 3. Condition of attack to prey by omega (ω) wolf, and the possible cases of its new position determined according to prey [41].

2.6. Harmony Search (HS)

One of the aims of musicians while offering a musical performance to listeners is to transfer work euphoniously. For this reason, the process of creation of an effective and good music work through a combination of notes, which are the best reflecting the works and most harmonious each other, continues to gain the appreciation of listeners.

Geem et al. (2001) [42], who was inspired by this process, developed an algorithm in 2001, which is known as the harmony search (HS) method, and based on natural (impromptu) music performance. The aim is to search for the liked harmony [43]. The functioning of this developed algorithm is not only offering of a music work by melodizing via the best notes, at the same time, but it also takes shape according to natural performance of a musician and idea that to optimally implement while performing of this, too.

When analyzed this process, three possible activities, which can be performed during the natural performance of a musician, and equivalents of these in terms of actions realized by harmony search is [44]:

- To play any popular musical piece completely from own memory: *usage of harmony memory*;
- To play something like to a known work: *pitch (tone) adjusting*;
- To compose/melodize new or random notes: *randomization*

Usage of memory is an important notion due to resembles case that of obtaining the best harmonies, which will transfer to new harmony memory, like selecting the most convenient persons in the genetic algorithms (GA). Here, a case that harmony memory consideration rate, which has a value between 0 and 1, is very high, causes to not finding out well of almost whole harmonies in old harmony memory [45].

Furthermore, on optimization of a problem, natural music performance can be considered as a process expressing that determining of optimum values for design variables. The most compatible notes/harmonies express the optimum values of design variables; the best, in other words, "most euphonic" music work, which occurs with the combination of these notes, expresses the case that performing of the objective function belonging to design problem.

The followed route by algorithm on optimization of design values changes according to the different cases. In each iteration, obtaining the optimized new values of variables is depending on the usage of harmony search memory. This case is determined via harmony memory consideration rate (HMCR) according to Equation (32).

- If the HMCR value is bigger than a randomly generated number, memory usage is not possible. In this case, randomly new notes should be generated;
- In the other case, notes recorded in harmony memory can be played from a specific pitch by remembering:

$$X_{i,\,new} = \begin{cases} HMCR > rand, & X_{i,\,min} + rand\,(X_{i,max} - X_{i,min}) \\ HMCR < rand, & X_{i,n} + rand\left(\frac{-1}{2}, \frac{1}{2}\right) PAR\,(X_{i,max} - X_{i,min}) \end{cases} \quad (32)$$

2.7. The Benchmark Truss Structure Problems

In this study, two large-scale truss models were handled for optimum sizing of bar sections in the direction of weight minimization. These are represented in Figures 4 and 5 as the geometry of 25 and 72 bar truss structures together with node and bar numbers in the space coordinate system, respectively.

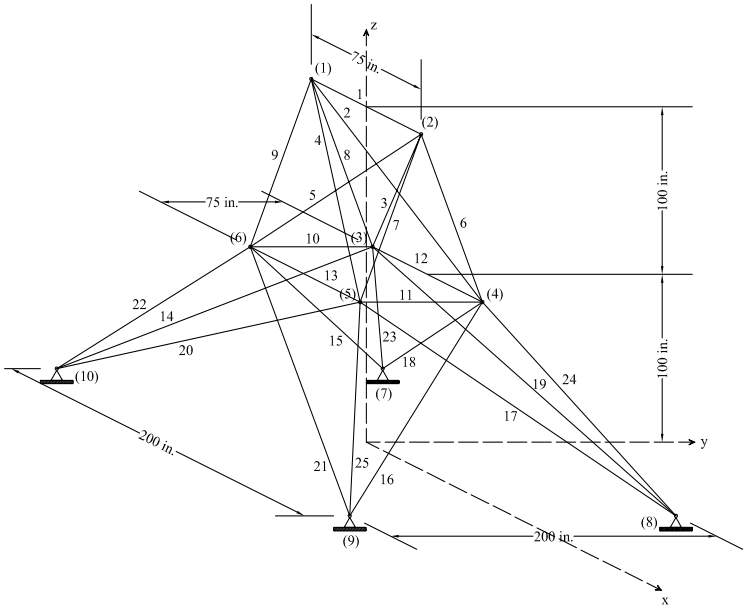

Figure 4. Design model and variables of a 25 bar truss structure [13].

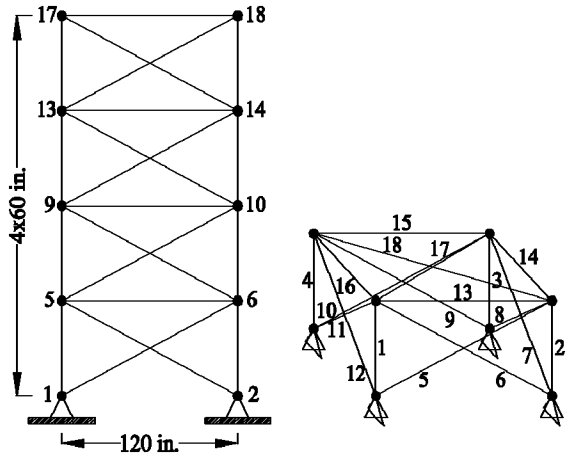

Figure 5. Design model and variables of a 72 bar truss structure [13].

In the optimization process, three cases were conducted to provide the minimum weight for these structures by the grouping of bars both 25 and 72 bar truss, and without grouping of 25 bar truss intended for evaluation of designing based on increment on the number of design variables. In this regard, design variables and constraints, which were determined by collecting in the same section area according to axis similarity and symmetry of bars for grouping cases, were given in Table 2. The material of the structures is aluminum. The same methodology can be applied for different materials, and design regulation rules can also be integrated as design constraints. Especially, the slenderness limits of the members under compressive forces are needed to be checked. In addition, the multiple loading conditions are in Tables 3 and 4; design limitations as displacement and stress constraints with bar groups are in Tables 5 and 6, for 25 and 72 bar trusses, respectively. The optimum results for all cases were ensured without permitted the violation of any constraint through penalizing the solutions, which exceed the limitations.

Table 2. Information for optimization process corresponding for 25 and 72 bar truss structures.

	Definition	Symbol	Limit/Value	Unit	Truss Model
Design Variables	Cross-section of truss bars	A_{bar}	0.01–3.4 0.1–3.0	$inch^2$	25-Bar 72-Bar
Design Constants	Elasticity modulus	E_s	10^7	psi	Both
	Weight per unit of volume of bars	ρ_s	0.1	$lb/inch^3$	
	Bar number	-	25 72	-	25-Bar 72-Bar
	Node number	-	10 20	-	25-Bar 72-Bar
	Bar group number	-	8 16	-	25-Bar 72-Bar

Table 3. Loading conditions on nodes for a 25 bar truss model.

Case	Node Number	Load			Unit
		P_x	P_y	P_z	
1	1	1000	10,000	−5000	lb/inch^2
	2	0	10,000	−5000	
	3	500	0	0	
	6	500	0	0	
2	1	0	20,000	−5000	
	2	0	−20,000	−5000	

Table 4. Loading conditions on nodes for a 72 bar truss model.

Case	Node Number	Load			Unit
		P_x	P_y	P_z	
1	17	5000	5000	−5000	lb/inch^2
2	17	0	0	−5000	
	18	0	0	−5000	
	19	0	0	−5000	
	20	0	0	−5000	

Table 5. The design constraints for a 25 bar truss structure.

Structural Member		Description	Constraints		Unit
Nodes			Displacement		
All		Limitation of Displacements Occurred on Nodes	$\delta < \mid \mp 0.35 \mid$		inch
Group Number	Design Variables		Compression Stress	Tension Stress	
1	A_1	Limitation required for stresses occurred on bars	$\sigma_c > -35,092$	$\sigma_t < +40,000$	psi
2	$A_{2–5}$		$\sigma_c > -11,590$		
3	$A_{6–9}$		$\sigma_c > -17,305$		
4	$A_{10–11}$		$\sigma_c > -35,092$		
5	$A_{12–13}$		$\sigma_c > -35,092$		
6	$A_{14–17}$		$\sigma_c > -6759$		
7	$A_{18–21}$		$\sigma_c > -6959$		
8	$A_{22–25}$		$\sigma_c > -11,080$		

Table 6. The design constraints for a 72 bar truss structure.

Structural Member		Description	Constraints		Unit
Nodes			Displacement		
All		Limitation of Displacements Occurred on Nodes	$\delta <\mid \mp 0.25 \mid$		inch
Group Number	Design Variables		Compression Stress	Tension Stress	
1	$A_{1–4}$	Limitation required for stresses occurred on bars	$\sigma_c > -25,000$	$\sigma_t < +25,000$	psi
2	$A_{5–12}$				
3	$A_{13–16}$				
4	$A_{17–18}$				
5	$A_{19–22}$				
6	$A_{23–30}$				
7	$A_{31–34}$				
8	$A_{35–36}$				
9	$A_{37–40}$				
10	$A_{41–48}$				
11	$A_{49–52}$				
12	$A_{53–54}$				
13	$A_{55–58}$				
14	$A_{59–66}$				
15	$A_{67–70}$				
16	$A_{71–72}$				

On the other hand, when performing optimization, six different metaheuristics containing HS, ABC, BA, FPA, GWO, and JA, besides ABCL, BAL and GWOL, were applied by modifying the structure of three algorithms with Lévy distribution. Additionally, optimization results, including the best weight together optimal section areas, were determined to a specific population number as 30 and different iteration numbers. Following, for grouping cases (Sections 3.1 and 3.2), the values of optimization parameters were handled in the range of 10–30 with 5 intervals and 1000–20,000 by increasing 500; and in not grouping case for 25 bar truss (Section 3.3), the same ranges, and 5000–120,000 by increasing 5000 for population and iteration numbers, respectively, to detect the most effective parameters in the sense of saving time and effort during that minimum weight was determined.

3. Numerical Examples

3.1. 25-Bar Truss Optimization with Bar Grouping

The first case is an application performed for 25 bar truss sizing optimization by the grouping of whole bars. To realize this, bars were assigned to specific section areas by collecting similar ones symmetrically according to axes in the spaceplane. While performing the optimization process, the aforementioned stages were applied as the usage of a constant population number as 30 together with different iteration numbers that can be seen in Figure 6. Moreover, several populations and iteration numbers were interworked. The minimum weight is obtained as 545.0413 lb through JA in 20,000 iterations.

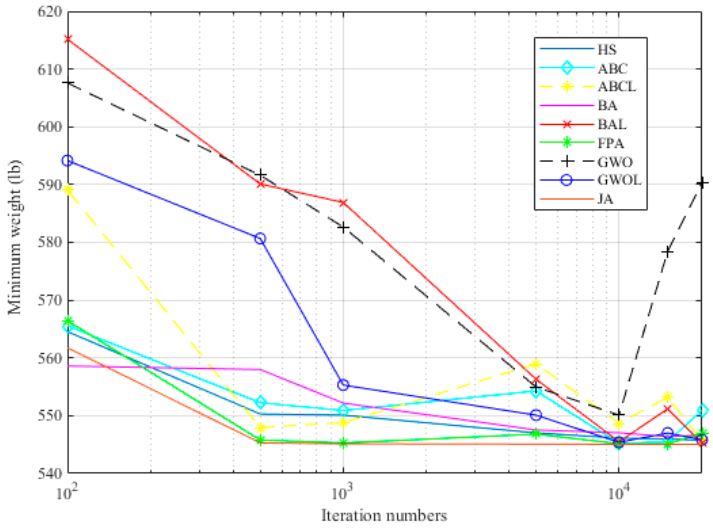

Figure 6. Variation of the values for minimum weight corresponding to different iteration numbers with a constant (30) population.

In addition to this, optimization results of some literature studies were given in Table 7 to evaluate and compare the results found by used algorithms currently. In addition, in Table 8, the results, including minimum weight and statistical calculations, and optimum design variables, can be seen for each metaheuristic. As it is understood from these tables, the most successful algorithm is JA due to be able to determine the minimum weight with a very small standard deviation. Moreover, JA and FPA with the usage of Lèvy distribution by nature outperform the compared literature results in the best optimum value. It is seen that the Lèvy distribution has a positive effect on the best and mean optimum values.

Table 7. Optimum results ensured from previous studies for 25 bar truss with grouping.

Group Number	Design Variables	Previous Studies						
		KH [17]	**TLBO [18]**	**HPSSO [19]**	**CSP [20]**	**TLBO [21]**	**ADEA [22]**	**HTS [23]**
1	A_1	0.01025	0.0100	0.0100	0.010	0.0100	0.0100	0.010000
2	$A_{2–5}$	2.02437	2.0712	1.9907	1.910	1.9878	5.6406	2.070200
3	$A_{6–9}$	3.04154	2.9570	2.9881	2.798	2.9914	8.5941	2.970031
4	$A_{10–11}$	0.01029	0.0100	0.0100	0.010	0.0102	0.0100	0.010000
5	$A_{12–13}$	0.01081	0.0100	0.0100	0.010	0.0100	0.0100	0.010000
6	$A_{14–17}$	0.68950	0.6891	0.6824	0.708	0.6828	1.9368	0.670790
7	$A_{18–21}$	1.62002	1.6209	1.6764	1.836	1.6775	4.7857	1.617120
8	$A_{22–25}$	2.65501	2.6768	2.6656	2.645	2.6640	7.5921	2.698100
Best weight		545.175	545.09	545.164	545.09	545.175	545.1657	545.13
Mean weight		545.483	545.41	545.556	545.20	545.483	545.2200	545.47
Standard deviation		0.306	0.42	0.432	0.487	0.306	0.0730	0.476
Population number		-	30	-	50	-	-	-
Iteration number		-	-	-	350	-	-	-
Total analysis number		12,199	15,318	13,326	17,500	12,199	10,000	7653

HTS: heat transfer search, ADEA: adaptive differential evolution algorithm, HPSSO: hybrid particle swallow swarm optimization, CSP: chaotic swarming of particles, TLBO: teaching-learning based optimization, KH: krill herd.

Table 8. Optimization results and the best parameters for 25 bar truss with bar grouping.

Group Number	Design Variables	Current Study								
		HS	**ABC**	**ABCL**	**BA**	**BAL**	**FPA**	**GWO**	**GWOL**	**JA**
1	A_1	0.0100	0.0115	0.0100	0.0100	0.0100	0.0100	0.0229	0.0108	0.0100
2	A_{2-5}	2.1375	1.9193	2.1071	1.9694	1.9774	2.0491	1.9229	2.0044	2.0420
3	A_{6-9}	2.8910	3.1549	2.9346	3.1356	3.0507	3.0341	3.0712	3.0422	3.0045
4	A_{10-11}	0.0100	0.0100	0.0100	0.0100	0.0100	0.0101	0.0100	0.0104	0.0100
5	A_{12-13}	0.0100	0.0102	0.0103	0.0100	0.0102	0.0100	0.2285	0.0142	0.0100
6	A_{14-17}	0.6887	0.6749	0.6415	0.6808	0.6923	0.6770	0.6142	0.6817	0.6816
7	A_{18-21}	1.5994	1.6627	1.6051	1.6258	1.6566	1.6035	1.7070	1.6375	1.6229
8	A_{22-25}	2.6991	2.6352	2.7508	2.6432	2.6412	2.6764	2.7241	2.6607	2.6737
Best weight		545.3419	545.4145	545.3736	545.3712	545.1191	545.0738	548.9530	545.1282	545.0378
Mean weight		548.3861	545.5653	545.4441	549.9517	546.4605	545.0738	548.9533	545.1287	545.0440
Standard deviation		1.160	0.104	0.108	1.940	5.560	2.10×10^{-13}	6.36×10^{-5}	1.88×10^{-4}	3.03×10^{-3}
Best population number		30	25	30	25	20	25	20	20	20
Best iteration number		18,000	13,000	5000	12,500	10,000	2500	11,000	11,500	15,000

3.2. 72-Bar Truss Optimization with Bar Grouping

The second case is the grouping of bars belonging 72 bar truss to provide optimal design variables and the best weight solution. In Figure 7, optimization results as minimum weight (379.6172 lb via JA in 15,000 iterations) are presented that they were obtained according to the constant population and various iteration numbers.

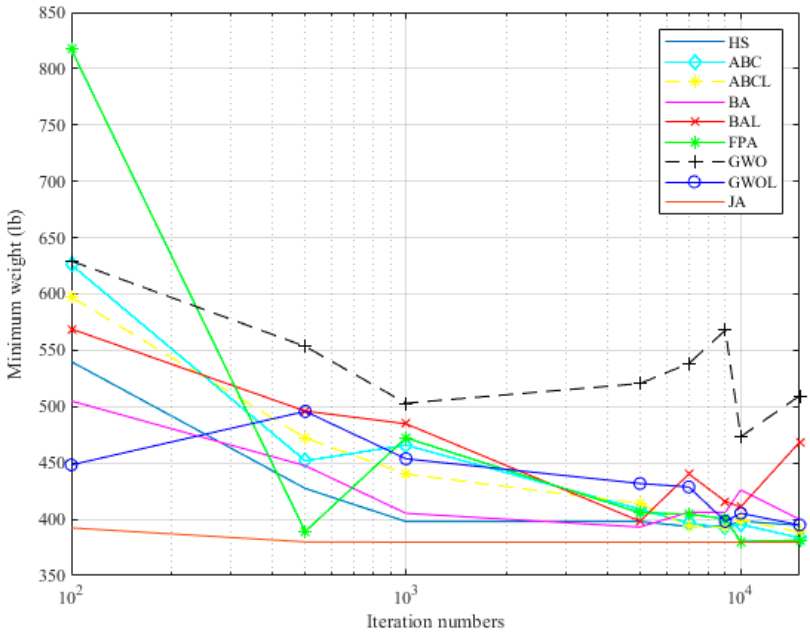

Figure 7. Minimum weight changing ensured according to multiple iteration numbers and a constant (30) population.

In Table 9, all current optimization results were represented with some parameter evaluations. Literature studies are also given in Table 10. For this case, the best meta-heuristic is again JA owing to reach the minimum weight, which is a far smaller value than the other algorithms. Furthermore, according to this algorithm, the standard deviation for the objective function is very little and so that it can be recognized that JA has the

best performance in every respect when the other current methods and the used previous algorithms are compared.

3.3. 25-Bar Truss Optimization without Bar Grouping

The third case is concerned with the determination of optimal section areas for each bar individually. In this regard, the best weight as a minimum (545.1083 lb) was achieved with JA by considering the only constant population in 100,000 iterations, as seen in Figure 8. In addition, it was understood that the objective function might be provided by usage of the best parameter combinations as 15 and 115,000 for population and iteration numbers with JA, respectively, according to the results of Table 11. For this case, the only documented optimization was proposed by Bekdaş et al. [24].

Table 9. Current optimum results with the best parameters for 72 bar truss (grouping).

Group Number	Design Variables	Current Study								
		HS	**ABC**	**ABCL**	**BA**	**BAL**	**FPA**	**GWO**	**GWOL**	**JA**
1	$A_{1–4}$	2.2301	2.0781	1.7910	1.7682	2.0679	1.9057	1.7845	2.0458	1.8899
2	$A_{5–12}$	0.4946	0.5187	0.5195	0.5720	0.4854	0.4916	0.5249	0.4864	0.5119
3	$A_{13–16}$	0.1000	0.1026	0.1000	0.1000	0.1000	0.1000	0.1000	0.1137	0.1000
4	$A_{17–18}$	0.1212	0.1000	0.1000	0.1384	0.1000	0.1003	0.1000	0.1003	0.1000
5	$A_{19–22}$	1.1985	1.2594	1.3484	1.3535	1.3457	1.3372	1.0110	1.1140	1.2702
6	$A_{23–30}$	0.5009	0.4559	0.4860	0.5347	0.5319	0.4907	0.6469	0.5504	0.5120
7	$A_{31–34}$	0.1000	0.1001	0.1000	0.1000	0.1181	0.1000	0.1000	0.1161	0.1000
8	$A_{35–36}$	0.1000	0.1000	0.1023	0.1000	0.1003	0.1000	0.1000	0.1063	0.1000
9	$A_{37–40}$	0.5477	0.6239	0.5061	0.6166	0.4839	0.5590	0.6409	0.5198	0.5234
10	$A_{41–48}$	0.5334	0.4710	0.5002	0.4921	0.5194	0.5088	0.5318	0.4672	0.5165
11	$A_{49–52}$	0.1032	0.1000	0.1000	0.1000	0.1005	0.1000	0.1000	0.1683	0.1000
12	$A_{53–54}$	0.1000	0.1000	0.1000	0.1567	0.1420	0.1000	0.3552	0.1000	0.1000
13	$A_{55–58}$	0.1626	0.1570	0.1558	0.1644	0.1562	0.1560	0.1502	0.1516	0.1565
14	$A_{59–66}$	0.5040	0.6202	0.5634	0.5031	0.5214	0.5373	0.4666	0.5651	0.5457
15	$A_{67–70}$	0.3831	0.3898	0.5024	0.3699	0.3972	0.4890	0.2846	0.3832	0.4104
16	$A_{71–72}$	0.7333	0.5725	0.5738	0.6129	0.5235	0.5802	0.9665	0.7247	0.5678
Best weight		386.2662	383.4078	381.5569	385.6475	381.9249	380.4598	398.7166	386.5409	379.6156
Mean weight		405.3741	402.1776	395.4222	417.7893	382.0331	380.4598	398.7167	386.5411	379.6172
Standard deviation		5.820	16.100	8.710	48.700	0.220	2.79×10^{-13}	5.77×10^{-5}	3.65×10^{-5}	7.49×10^{-4}
Best Population number		25	30	30	15	10	25	15	30	30
Best iteration number		13,000	14,500	19,000	20,000	17,500	18,000	8500	19,500	17,500

Table 10. Some literature studies concerning 72 bar truss (grouping).

Group Number	Design Variables	Previous Studies						
		BB-BC [10]	GA [16]	TLBO [18]	CSP [20]	TLBO [21]	ADEA [22]	HTS [23]
1	$A_{1–4}$	1.8577	1.702	1.90640	1.94459	1.8807	1.8861	1.9001
2	$A_{5–12}$	0.5059	0.496	0.50612	0.50260	0.5142	0.5231	0.5131
3	$A_{13–16}$	0.1000	0.100	0.10000	0.10000	0.1000	0.1000	0.1000
4	$A_{17–18}$	0.1000	0.100	0.10000	0.10000	0.1000	0.1000	0.1000
5	$A_{19–22}$	1.2476	1.288	1.26170	1.26757	1.2711	1.2576	1.2456
6	$A_{23–30}$	0.5269	0.469	0.51110	0.50990	0.5151	0.5043	0.5080
7	$A_{31–34}$	0.1000	0.100	0.10000	0.10000	0.1000	0.1000	0.1000
8	$A_{35–36}$	0.1012	0.100	0.10000	0.10000	0.1000	0.1000	0.1000
9	$A_{37–40}$	0.5209	0.505	0.53170	0.50674	0.5317	0.5200	0.5550
10	$A_{41–48}$	0.5172	0.550	0.51591	0.51651	0.5134	0.5235	0.5227
11	$A_{49–52}$	0.1004	0.109	0.10000	0.10752	0.1000	0.1000	0.1000
12	$A_{53–54}$	0.1005	0.118	0.10000	0.10000	0.1000	0.1000	0.1000
13	$A_{55–58}$	0.1565	0.154	0.15620	0.15618	0.1565	0.1568	0.1566
14	$A_{59–66}$	0.5507	0.604	0.54927	0.54022	0.5429	0.5394	0.5407
15	$A_{67–70}$	0.3922	0.442	0.40966	0.42229	0.4081	0.4083	0.4084
16	$A_{71–72}$	0.5922	0.604	0.56976	0.57941	0.5733	0.5734	0.5669
Best weight		379.85	379.63	379.63	379.97	379.632	379.6943	379.73
Mean weight		382.08	-	-	381.56	379.759	379.8961	382.26
Standard deviation		1.912	-	-	1.803	0.149	0.0791	1.940
Population number		-	-	-	-	-	-	-
Iteration number		-	-	-	-	-	-	-
Total analysis number		6942	19,709	19,709	10,500	21,542	15,600	13,166

HTS: heat transfer search, ADEA: adaptive differential evolution algorithm, CSP: chaotic swarming of particles, GA: genetic algorithm, TLBO: teaching-learning based optimization, BB-BC: big-bang big crunch.

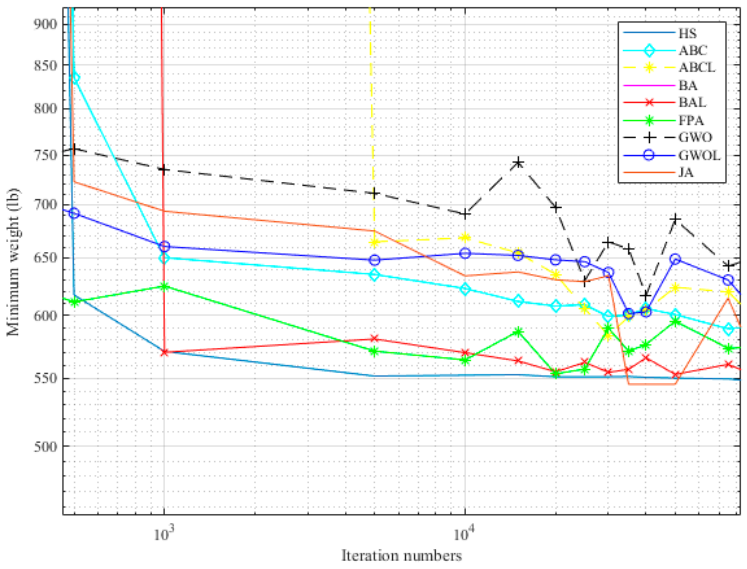

Figure 8. Changing for minimum weight values concerning different iteration numbers via a constant (30) population.

Table 11. Optimum results provided with the best parameters for 25 bar truss without grouping.

Design Variables	Documented Study [24]	Current Study								
	FPA	HS	ABC	ABCL	BA	BAL	FPA	GWO	GWOL	JA
A_1	0.0100	0.4458	0.1298	0.0104	0.1184	0.0100	0.0965	0.7796	0.0103	0.0100
A_2	2.3903	3.1494	2.2184	3.4000	2.0174	2.6515	1.4802	1.2254	1.9548	1.9019
A_3	1.8524	2.0411	1.9449	1.4015	2.2780	1.6965	2.9122	2.2786	1.2900	2.4622
A_4	2.0935	2.1586	2.0351	3.3351	1.8349	2.5179	1.2167	2.8893	2.5339	1.6803
A_5	1.9749	2.1175	1.9137	0.7374	2.4292	1.5376	2.9552	1.6550	2.5293	2.4353
A_6	2.9549	2.7672	2.4382	3.4000	2.8319	3.3445	2.3174	2.5574	2.8853	2.7733
A_7	2.9379	2.7920	2.7738	2.9510	3.0574	3.2573	2.6665	2.7372	2.6204	2.7611
A_8	3.0085	2.8634	3.0920	2.4397	3.1506	2.8895	3.2496	3.0388	2.7839	3.4000
A_9	2.4974	2.8615	1.9781	2.3134	2.9135	2.1074	3.2680	2.9789	2.0761	2.8591
A_{10}	0.0100	0.1332	0.1210	0.2810	0.1278	0.1041	0.0100	0.6568	0.0159	0.0100
A_{11}	0.0104	1.6132	0.3037	0.0100	0.0860	0.0294	0.0100	0.4757	0.0756	0.0100
A_{12}	0.0100	0.4209	0.0157	0.0196	0.0100	0.0107	0.1100	0.8414	0.0115	0.0100
A_{13}	0.0100	0.3124	0.3610	0.0100	0.0100	0.0132	0.0100	0.3416	0.0154	0.0100
A_{14}	0.7058	1.2763	1.3285	0.5596	0.7848	0.5993	0.8670	0.9344	0.9142	0.8295
A_{15}	0.5950	0.4679	0.3222	0.6389	0.7687	0.6265	0.7245	0.8662	0.4807	0.6832
A_{16}	0.8043	1.5477	1.1122	0.8279	0.6498	0.7242	0.6477	1.6382	1.3634	0.6205
A_{17}	0.6149	0.1954	0.6027	0.5991	0.6166	0.7340	0.4533	0.4981	0.5772	0.5581
A_{18}	1.7011	1.7648	1.8217	2.6633	1.6228	1.7418	1.2780	2.1802	2.4427	1.4748
A_{19}	1.7259	1.0280	2.0910	1.2955	1.8506	1.7004	1.8226	2.2122	2.2463	1.8439
A_{20}	1.8375	2.1691	2.1350	2.4677	1.4679	1.7927	1.4716	2.0611	2.0330	1.5456
A_{21}	1.3793	1.5454	1.7626	0.8297	1.4096	1.3135	2.0145	2.0826	2.2709	1.4880
A_{22}	2.3446	2.0615	3.0436	2.2567	2.3569	2.1048	3.1522	3.0129	1.7381	2.9342
A_{23}	2.5744	2.0553	3.0927	2.1876	3.0532	2.6214	3.2085	3.0249	2.6066	3.0578
A_{24}	3.1464	3.3649	3.0028	3.4000	3.0771	3.2652	2.5699	1.7545	3.0829	2.6271
A_{25}	2.5920	2.9254	2.0149	3.4000	2.1005	2.7427	1.8336	1.9041	2.7111	2.0332
Best weight	543.20	585.9394	573.9692	565.9572	549.4372	545.2959	548.1256	604.4136	578.7288	542.9822
Mean weight	-	603.0324	599.9786	566.0596	551.1452	552.0253	548.1256	604.4137	578.7288	543.0017
Standard deviation	-	16.400	17.400	0.165	1.180	34.400	1.18×10^{-13}	9.04×10^{-6}	1.79×10^{-6}	8.22×10^{-3}
Best population number	20	15	30	15	25	30	25	10	30	15
Best iteration number	100,000	95,000	115,000	35,000	30,000	110,000	55,000	120,000	90,000	115,000

4. Results

4.1. 25-Bar Truss Optimization with Bar Grouping

As it was mentioned previously, first, an optimization operation was applied by utilizing a constant population and different iterations (Figure 6). Generally, it can be recognized from this figure, all of the ones can be considered as successful except GWO, ABCL, and BA in terms of nearing the minimum weight. However, in this meaning, the best algorithm is JA, thanks to the finding of weight as 545.041 lb at minimum. Nevertheless, Lèvy flight is effective for both BA and especially GWO.

On the other respect, according to the results in Tables 6 and 7, from the used algorithms in the current applications, JA is the method that can find the minimum weight as 545.0378 lb, which is a smaller value than the best one from literature studies, and it was provided with an extremely small deviation. Additionally, except for JA, it appeared that FPA could also be accepted as accomplish for minimization of the weight according to the given literature results. Thus, this algorithm employing Lèvy distribution in its initial form can be minded as effective and useful due to that it can achieve this objective by making little standard error. If we evaluate the other methods, it can be said that GWOL and BAL

almost approximate to the minimum weight for grouping case of the 25 bar truss. Here, the significant point is that Lèvy flight is noteworthily effective in terms of minimizing of weight by all of the swarm intelligence-based algorithms, especially for GWO, besides that the required iteration numbers were decreased with this function for BA and especially ABC, too.

4.2. 72-Bar Truss Optimization with Bar Grouping

According to Figure 7, with the usage of a constant population with various iterations, it can be said that the most effective methods are FPA and JA cause that the other ones could not make convergence exactly to the minimum structural weight of the 72 bar truss structure. However, the best one is JA by reaching the minimum weight of 379.617 lb. In addition, in this case, Lèvy flight is notably efficient for only GWO.

On the other respect, via Tables 9 and 10, it is seen that the best algorithm is JA in finding the minimum weight as 379.6156 lb by deviating from this value with a so small rate. Although the provided result by JA is close to the best ones among literature studies in general scope, this can be considered as a more effective method thanks to the small deviation.

Moreover, the other algorithms are not successful in an exact way in terms of reaching the minimum weight value besides that FPA has a comparatively efficient performance. However, positive effects (such as decreasing of standard deviation, etc.) and performance of Lèvy flight on the benefited algorithms are drawn attention in the direction of the realization of the optimization target. In this case, the Lèvy flight function improved, especially GWO, by decreasing weight by nearly 13 lb amount.

4.3. 25-Bar Truss Optimization without Bar Grouping

It can be understood from Figure 8; the best method is only JA through the achievement of the minimized truss weight as 545.108 lb for 25 bar without grouping of bars. HS, FPA, and BAL also can be accepted as usable relatively to the other ones. Furthermore, Lèvy flight affected all handled algorithms positively in the sense of convergence to more the minimized weight.

According to Table 11, JA is the top-ranking between the expressed all techniques (literature and current studies) by far thanks to minimizing of truss weight as 542.9822 lb; even this result was got via the pretty minor rate for deviation as 8.22×10^{-3}. On the other side, Lèvy flight influenced all population-based algorithms utilizing reduction of minimum weight with comparison to the classical structure of the mentioned methods. Additionally, this function provides to decrease of the best necessary iteration numbers for ABC with GWO, too.

5. Conclusions

As a result, it can be said that the JA algorithm is the best option to be benefitted from the minimization of the structural weight regarded the handled truss models. This algorithm could succeed in this operation by providing to occur of very small standard deviations, too. On the other side, when the Lèvy flight was evaluated, the constant population taken as 30 is always not good to prefer in terms of improving minimization performance for swarm intelligence-based algorithms according to both cases compared to each other. Because it can be understood from the results, algorithms combined with Lèvy flight are more powerful and efficient about finding the optimum value of the weight when the population number shows a change by adjusting the required iteration numbers.

Except for these, Lèvy flight is chiefly useful for developing of GWO algorithm's minimization performance compared with the other ones. For this respect, in the other optimization studies, hybridization of GWO with this function can be benefited for many objectives.

Author Contributions: Conceptualization, G.B., M.Y. and S.M.N.; methodology, G.B., M.Y. and S.M.N.; software, G.B. and M.Y.; validation, M.Y. and S.M.N.; formal analysis, G.B., M.Y. and S.M.N.; investigation, M.Y.; resources, M.Y.; data curation, M.Y.; writing—original draft preparation, M.Y.;

writing—review and editing, S.M.N.; visualization, M.Y.; supervision, G.B and S.M.N.; project administration, G.B. All authors have read and agreed to the published version of the manuscript.

Funding: This study was funded by the Scientific Research Projects Coordination Unit of Istanbul University-Cerrahpaşa. Project number: FYO-2019-32735.

Data Availability Statement: The corresponding data of the paper can be obtained by requesting via e-mail to the corresponding author.

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

Appendix A

Table A1. Some expressions used as special by BA.

Abbreviation	Explanation
$\mathbf{V_{i,new}}$	New velocity value for ith design variable
$\mathbf{V_{i,j}}$	Current velocity value of ith design variable within jth candidate solution (bat)
f_j	Frequency of jth candidate solution
A_{mean}	Mean of sound loudness values belonging all bats
$A_{j,new}$	New value, which will be assigned for loudness for ith bat
$r_{j,new}$	New value, which will be determined for sound vibration emission rate of bats

Table A2. Indication of basic and special algorithm parameters.

	Notation	Property	Algorithm
Population Number	HMS	Total harmony number or harmony memory size	HS
	eb/*ob*/*fsn*	Number of employee bee/onlooker bee/food source number	ABC
	fn	Number of total fireflies	FA
	bn	Number of bats	BA
	nf	Total flower number	FPA
	wn	Gray wolf number in the pack	GWO
	pn	Population number	JA

Table A2. *Cont.*

	Notation	Property	Algorithm
Characteristic Parameters	PAR	Parameter providing of generating random number depended on music tone, between limits of variable/pitch adjusting rate	HS
	HMCR	Harmony memory consideration rate	
	ip	Parameter, which takes value according to the case that food sources can be optimized	ABC
	SIL	Limit condition, which is considered for which sources must be renewed by scout bees and is assigned at the beginning of the optimization	
	β_0	Minimum ($r_{jk} = 0$) attractiveness value ($\beta_0 \in [0,1]$)	FA
	γ	Light absorption coefficient ($\gamma \in [0,1]$)	
	α_t	Randomization parameter	
	f_{min}	Minimum value of frequency	BA
	f_{max}	Maximum value of frequency	
	A_0	Loudness of bats during the initial state	
	A_{min}	Minimum value of sound loudness	
	r_j^0	Sound vibration emission rate of bats in the initial state	
	β	Random number, which is determined in the range [−1,1]	
	α	Constant coefficient that effective for transmission of loudness	
	γ	Constant coefficient that determines of sound vibration emission rate	
	sp	Switch probability/search change	FPA
	$\vec{C}$	Coefficient vector	GWO
	$\vec{A}$	Vector determining the case that wolf attacks to prey	
	$\vec{a}$	Vector affecting the distance between prey with wolf	

Table A3. Definition of expressions used in algorithms as general and commonly.

	Expression	Explanation	Algorithm
Some Solution Values for Design Variables	$X_{i,new}$	New value of ith design variable	All
	$X_{i,min}$	Low limit of ith design variable (minimum value)	HS ABC
	$X_{i,max}$	Upper limit of ith design variable (maximum value)	
	$X_{i,j}$	Initial matrix value of jth candidate solution belonging to ith design variable	All out of HS
	$X_{i,g_{best}}$	ith design variable value belongs to the solution with the best value in terms of the objective function	All out of HS, ABC, and FA
	$X_{i,g_{worst}}$	ith design variable value belongs to the solution with the worst value in terms of the objective function	JA
	$X_{i,n}$	Value of nth solution for ith design variable	FPA HS

Table A3. *Cont.*

	Expression	Explanation	Algorithm
	$X_{i,m}$	Value of mth solution for ith design variable	FPA
	$X_{i,k}$	Value of a specific kth firefly for ith design variable	FA
	$X_{i,pr}$	Initial matrix value of ith design variable corresponding to pth solution (prey)	GWO
	$X_{p,\ new}$	New value of pth design variable	ABC
	$X_{p,j}$	Value (in initial solution matrix) of jth solution for pth design variable	
	$X_{p,n}$	nth solution value of pth design variable	
Optimization Elements	n	nth candidate vector selected randomly from the initial matrix	All out of BA, FA, and JA
	M	mth candidate vector selected randomly from the initial matrix	FPA
	P	Randomly selected parameter as from whole design variables	ABC
	vn	Number of total design variable, which will be optimized	FA, ABC
	T	Stage of the current iteration	FA, BA, GWO
	stopping criteria	Total iteration number	GWO

Table A4. Functions utilized in algorithms.

	Name	Task	Algorithm
Functions	rand	Generation random number between 0 and 1	All
	ceil ( )	Round the number in parentheses to equal/bigger natural number than it	FPA, ABC, HS
	min ( )	Determine the smallest one among value in a certain amount	All out of HS and ABC
	max ( )	Determine the biggest one among value in a certain amount	JA
	mean ( )	Calculate the average of element values in an array	BA
	abs ( )	Absolute of number within the parentheses	JA, GWO
	sort ( )	Queue the members in any array from small to big	GWO
	exp ( )	Give the power of *e* to the degree of number within parentheses	FPA
	sqrt ( )	Take the square root of a number	FA
	sum ( )	Give the total of element values in any number array	ABC

References

1. Bekdaş, G.; Nigdeli, S.M. Mass ratio factor for optimum tuned mass damper strategies. *Int. J. Mech. Sci.* **2013**, *71*, 68–84. [CrossRef]
2. Parcianello, E.; Chisari, C.; Amadio, C. Optimal design of nonlinear viscous dampers for frame structures. *Soil Dyn. Earthq. Eng.* **2017**, *100*, 257–260. [CrossRef]
3. Talatahari, S.; Gandomi, A.H.; Yang, X.S.; Deb, S. Optimum design of frame structures using the eagle strategy with differential evolution. *Eng. Struct.* **2015**, *91*, 16–25. [CrossRef]
4. Gholizadeh, S.; Ebadijalal, M. Performance based discrete topology optimization of steel braced frames by a new metaheuristic. *Adv. Eng. Softw.* **2018**, *123*, 77–92. [CrossRef]
5. Fathali, M.A.; Rohollah, S.; Vaez, H. Optimum performance-based design of eccentrically braced frames. *Eng. Struct.* **2020**, *202*, 109857. [CrossRef]
6. Kayabekir, A.E.; Bekdaş, G.; Nigdeli, S.M. *Metaheuristic Approaches for Optimum Design of Reinforced Concrete Structures: Emerging Research and Opportunities*; IGI Global: Hershey, PA, USA, 2020.
7. Vaez, S.R.H.; Qomi, H.S. Bar layout and weight optimization of special RC shear wall. *Structures* **2018**, *14*, 153–163. [CrossRef]

8. Sheikholeslami, R.; Khalili, B.G.; Sadollah, A.; Kim, J. Optimization of reinforced concrete retaining walls via hybrid firefly algorithm with upper bound strategy. *KSCE J. Civ. Eng.* **2016**, *20*, 2428–2438. [CrossRef]
9. Aydogdu, I. Cost optimization of reinforced concrete cantilever retaining walls under seismic loading using a biogeography-based optimization algorithm with Lèvy flights. *Eng. Optim.* **2017**, *49*, 381–400. [CrossRef]
10. Camp, C.V. Design of space trusses using Big Bang–Big Crunch optimization. *J. Struct. Eng.* **2007**, *133*, 999–1008. [CrossRef]
11. Gomes, H.M. Truss optimization with dynamic constraints using a particle swarm Algorithm. *Expert Syst. Appl.* **2011**, *38*, 957–968. [CrossRef]
12. Miguel, L.F.F.; Miguel, L.F.F. Shape and size optimization of truss structures considering dynamic constraints through modern metaheuristic algorithms. *Expert Syst. Appl.* **2012**, *39*, 9458–9467. [CrossRef]
13. Bekdaş, G.; Nigdeli, S.M.; Yang, X.S. Sizing optimization of truss structures using flower pollination algorithm. *Appl. Soft Comput.* **2015**, *37*, 322–331. [CrossRef]
14. Kaveh, A.; Ghazaan, M.I. Vibrating particles system algorithm for truss optimization with multiple natural frequency constraints. *Acta Mech.* **2017**, *228*, 307–322. [CrossRef]
15. Tejani, G.G.; Pholdee, N.; Bureerat, S.; Prayogo, D. Multiobjective adaptive symbiotic organisms search for truss optimization problems. *Knowl. Based Syst.* **2018**, *161*, 398–414. [CrossRef]
16. Dede, T.; Bekiroğlu, S.; Ayvaz, Y. Weight minimization of trusses with genetic algorithm. *Appl. Soft Comput.* **2011**, *11*, 2565–2575. [CrossRef]
17. Gandomi, A.H.; Alavi, A.H.; Talatahari, S. Structural Optimization Using Krill Herd Algorithm. In *Swarm Intelligence and Bio-Inspired Computation*; Elsevier: Amsterdam, The Netherlands, 2013; pp. 335–349.
18. Degertekin, S.O.; Hayalioglu, M.S. Sizing truss structures using teaching-learning-based optimization. *Comput. Struct.* **2013**, *119*, 177–188. [CrossRef]
19. Kaveh, A.; Bakhshpoori, T.; Afshari, E. An efficient hybrid particle swarm and swallow swarm optimization algorithm. *Comput. Struct.* **2014**, *143*, 40–59. [CrossRef]
20. Kaveh, A.; Sheikholeslami, R.; Talatahari, S.; Keshvari-Ilkhichi, M. Chaotic swarming of particles: A new method for size optimization of truss structures. *Adv. Eng. Softw.* **2014**, *67*, 136–147. [CrossRef]
21. Camp, C.V.; Farshchin, M. Design of space trusses using modified teaching–learning based optimization. *Eng. Struct.* **2014**, *62*, 87–97. [CrossRef]
22. Bureerat, S.; Pholdee, N. Optimal truss sizing using an adaptive differential evolution algorithm. *J. Comput. Civ. Eng.* **2016**, *30*, 04015019. [CrossRef]
23. Degertekin, S.O.; Lamberti, L.; Hayalioglu, M.S. Heat transfer search algorithm for sizing optimization of truss structures. *Lat. Am. J. Solids Struct.* **2017**, *14*, 373–397. [CrossRef]
24. Bekdaş, G.; Nigdeli, S.M.; Yang, X.S. Size optimization of truss structures employing flower pollination algorithm without grouping structural members. *Int. J. Theor. Appl. Mech.* **2017**, *1*, 269–273.
25. Yang, X.S. Flower pollination algorithm for global optimization. In *International Conference on Unconventional Computing and Natural Computation*; Springer: Berlin/Heidelberg, Germany, 2012; pp. 240–249.
26. Yang, X.S.; Koziel, S.; Leifsson, L. Computational optimization, modelling and simulation: Recent trends and challenges. *Procedia Comput. Sci.* **2013**, *18*, 855–860. [CrossRef]
27. Yang, X.S.; Bekdaş, G.; Nigdeli, S.M. (Eds.) *Metaheuristics and Optimization in Civil Engineering*; Springer: Cham, Switzerland, 2016; ISBN 9783319262451.
28. Karaboga, D. *An Idea Based on Honey Bee Swarm for Numerical Optimization*; Technical Report-tr06; Computer Engineering Department, Engineering Faculty, Erciyes University: Talas, Turkey, 2005; Volume 200, pp. 1–10.
29. Erdoğmuş, P. Doğadan esinlenen optimizasyon algoritmaları ve optimizasyon algoritmalarının optimizasyonu. *Düzce Üniversitesi Bilim Ve Teknol. Derg.* **2016**, *4*, 293–304.
30. Tapao, A.; Cheerarot, R. Optimal parameters and performance of artificial bee colony algorithm for minimum cost design of reinforced concrete frames. *Eng. Struct.* **2017**, *151*, 802–820. [CrossRef]
31. Yahya, M.; Saka, M.P. Construction site layout planning using multi-objective artificial bee colony algorithm with Lèvy flights. *Autom. Constr.* **2014**, *38*, 14–29. [CrossRef]
32. Karaboga, D.; Basturk, B. A powerful and efficient algorithm for numerical function optimization: Artificial bee colony (ABC) algorithm. *J. Glob. Optim.* **2007**, *39*, 459–471. [CrossRef]
33. Yang, X.S. *A New Metaheuristic Bat-Inspired Algorithm*; Nature Inspired Cooperative Strategies for Optimization (NICSO 2010); González, J.R., Pelta, D.A., Cruz, C., Terrazas, G., Krasnogor, N., Eds.; Springer: Berlin/Heidelberg, Germany, 2010; ISBN 9783642125386.
34. Gandomi, A.H.; Yang, X.S.; Talatahari, S.; Alavi, A.H. (Eds.) Metaheuristic Algorithms in Modeling and Optimization. In *Metaheuristic Applications in Structures and Infrastructures*; Elsevier: Amsterdam, The Netherlands, 2013; pp. 1–24.
35. Kaveh, A.; Zakian, P. Enhanced bat algorithm for optimal design of skeletal structures. *Asian J. Civ. Eng.* **2014**, *15*, 179–212.
36. Rao, R. Jaya: A simple and new optimization algorithm for solving constrained and unconstrained optimization problems. *Int. J. Ind. Eng. Comput.* **2016**, *7*, 19–34.
37. Mirjalili, S.; Mirjalili, S.M.; Lewis, A. Grey wolf optimizer. *Adv. Eng. Softw.* **2014**, *69*, 46–61. [CrossRef]

38. Doğan, C. Balina Optimizasyon Algoritması ve gri Kurt Optimizasyonu Algoritmaları Kullanılarak yeni Hibrit Optimizasyon Algoritmalarının Geliştirilmesi. Master's Thesis, T.C. Erciyes University, Kayseri, Turkey, 2019.
39. Doğan, L. Robot yol Planlaması için gri kurt Optimizasyon Algoritması. Master's Thesis, Bilecik Şeyh Edebali University, Bilecik, Turkey, 2018.
40. Faris, H.; Aljarah, I.; Al-Betar, M.A.; Mirjalili, S. Grey wolf optimizer: A review of recent variants and applications. *Neural Comput. Appl.* **2018**, *30*, 413–435. [CrossRef]
41. Şahin, İ.; Dörterler, M.; Gökçe, H. Optimum design of compression spring according to minimum volume using grey wolf optimization method. *Gazi J. Eng. Sci.* **2017**, *3*, 21–27.
42. Geem, Z.W.; Kim, J.H.; Loganathan, G.V. A new heuristic optimization algorithm: Harmony search. *Simulation* **2001**, *76*, 60–68. [CrossRef]
43. Bekdaş, G.; Nigdeli, S.M. Değişik periyotlu yapılar için optimum pasif kütle sönümleyici özelliklerinin belirlenmesi. In *1. Türkiye Deprem Mühendisliği ve Sismoloji Konferansı*; 11–14 Ekim; ODTÜ: Ankara, Turkey, 2011; pp. 1–8.
44. Koziel, S.; Yang, X.S. (Eds.) *Computational Optimization, Methods and Algorithms*; Springer: Berlin/Heidelberg, Germany, 2011; Volume 356, ISBN 978-3-642-20858-4.
45. Rao, S.S. *Engineering Optimization Theory and Practice*, 4th ed.; John Wiley & Sons: Hoboken, NJ, USA, 2009; ISBN 978-0-470-18352-6.