information

MDPI

Article

Harmony Search Method with Global Sharing Factor Based on Natural Number Coding for Vehicle Routing Problem

Liqun Liu [1,*], Jiuyuan Huo [2], Fei Xue [1] and Yongqiang Dai [1]

[1] College of Information Science and Technology, Gansu Agricultural University, Lanzhou 730070, China; xuef@st.gsau.edu.cn (F.X.); dyq@gsau.edu.cn (Y.D.)
[2] School of Electronic and Information Engineering, Lanzhou Jiaotong University, Lanzhou 730070, China; huojy@mail.lzjtu.cn
* Correspondence: liulq@gsau.edu.cn; Tel.: +86-13909482426

Received: 24 December 2019; Accepted: 3 February 2020; Published: 5 February 2020

Abstract: This paper proposes an improved Harmony Search algorithm, and gives the definition of the Global Sharing Factor of the Harmony Search (HS) algorithm. In the definition, the number of creations of the HS algorithm is applied to the sharing factor and calculated. In this algorithm, the natural harmony encoding method is used to encode the initial harmony, and the total path length of all vehicles is taken as the optimization objective function. A new harmony generation strategy is proposed as follows: each tone component in an evolution is calculated separately using the new learning strategy and update strategy. In the calculation process, the tone component is judged by whether it needs to be adjusted according to the adjustment strategy. In this way, the problems of singularity and randomness of the new harmony generation strategy of basic HS are solved to improve the diversity of algorithm solutions. Then, a new Harmony Search method with Global Sharing Factor based on natural number coding and decoding for the Vehicle Routing Problem (GSF-HS-VRP) is proposed. The improved Global Sharing Factor-Harmony Search-Vehicle Routing Problem (GSF-HS-VRP) algorithm is applied to capacity-limited vehicle path optimization problems compared with the HS, Improved Harmony Search (IHS), Global-best Harmony Search (GHS), and Self-adaptive Global Best Harmony Search (SGHS) algorithms. The small-scale data and Solomon examples were adopted as the experimental data. Compared with the other four algorithms, the GSF-HS-VRP algorithm has the shortest running time, more rapid convergence speed, and higher efficiency. In the multi-vehicle test, with the increase of the number of vehicles, the optimized path of the vehicle is more satisfied in relation to the actual needs of customers. The results showed that this method could effectively improve the optimization performance of the capacity-limited vehicle routing problem.

Keywords: Harmony Search algorithm; natural number coding; learning strategy; update strategy; global sharing factor; capacity-limited vehicle routing optimization problem

1. Introduction

The vehicle routing problem (VRP) is an important part of logistics mode and logistics management [1]. Proper selection of the vehicle's driving route is extremely important to improve the economic efficiency of the logistics model [1]. The vehicle routing problem is a typical Non-deterministic Polynomial-Complete (NPC) problem [2–5], which are usually solved with heuristic algorithms such as Genetic Algorithm (GA), Particle Swarm Optimization (PSO), etc. However, these algorithms all have the disadvantages that it is easy for them to fall into the local minimum and generate a lot of useless loops, which leads to the long running time of GA and premature convergence of PSO [6]. Swarm intelligence optimization algorithms mainly include the Artificial Bee Colony (ABC) algorithm [7], Locust Swarm (LS) Algorithm [8], Moth-Flame Optimization (MFO) [9] Algorithm, Shuffled Frog Leaping Algorithm (SFLA) [10], Fruit Fly Optimization Algorithm (FOA) [11], Harmony Search (HS) algorithm [12], etc. Various swarm intelligence optimization algorithms have their own characteristics and advantages. The ABC Algorithm [7] simulates the characteristics of swarm bees. Through the cooperation of three kinds of bees, the random and targeted evolution of swarms gradually converges to the optimum. Locust Swarms (LS) are a new multi-optima search technique explicitly designed for non-globally convex search spaces [8]. Ldquosmartrdquo is used by LS to start points to scout for promising new areas of the search space before using particle swarms and a greedy local search technique to find a local optimum [8,13]. The Moth-Flame optimization (MFO) algorithm is an optimization algorithm generated by simulating the biological behavior of a moth using the transverse orientation to find a light source [9]. However, the MFO algorithm itself does not have the ability to jump out of the local optimum, and it has the defects of being easily trapped in the local optimum and insufficient convergence speed in the later stage [14]. The Shuffled Frog Leaping Algorithm (SFLA) [10] is a swarm intelligence algorithm that simulates the characteristics of information sharing and communication during frog foraging. The evolutionary idea of the Fruit Fly Optimization Algorithm (FOA) [11] originated from the simulation of the foraging behavior of the fruit fly, which is a global optimization evolutionary algorithm.

The Harmony Search (HS) algorithm [12] is a swarm intelligence algorithm that simulates the process of music players' repeated tuning of the pitch of each musical instrument through memory to achieve a wonderful harmony process. It has been successfully applied to many engineering problems. The HS algorithm has the advantages of simple concept, fast convergence, easy implementation, a novel generation of solutions, and a few parameters to adjust. It shows better performance than the Genetic Algorithm, Simulated Annealing algorithm, and Tabu Search algorithm on related issues [15,16]. The HS algorithm has been successfully applied in different fields. A new block-matching (BM) algorithm that combines HS with a fitness approximation model is proposed in the literature [17], and the set of motion vectors is evolved through HS operators until the best possible motion vector is identified. In the literature [18], a new dynamic clustering algorithm based on the hybridization of Harmony Search (HS) and fuzzy c-means to automatically segment MRI brain images in an intelligent manner was presented. Improvements to Harmony Search (HS) in circle detection were proposed that enable the algorithm to robustly find multiple circles in larger data sets and still work on realistic images that are heavily corrupted by noisy edges [19]. A new approach to estimate the vanishing point using a harmony search (HS) algorithm was proposed in [20]. Ref. [21] presents a simple mathematical analysis of the explorative search behavior of the Harmony Search (HS) algorithm.

Aiming at the singularity and randomness of the new harmony generation strategy, from the perspective of improving the diversity of algorithm solutions, the global sharing factor is introduced to improve the new harmony generation strategy, and an improved HS algorithm is proposed. The new algorithm is applied to solve the capacity-limited vehicle routing optimization problem. Then, a new Harmony Search method with Global Sharing Factor based on natural number coding and decoding for the Vehicle Routing Problem (GSF-HS-VRP) is proposed to improve the optimization performance.

2. Theory Basis

2.1. Capacity-Limited Vehicle Routing Problem

There are many variants of the VRP problems, of which the Capacity-limited Vehicle Routing Problem (CVRP) is more common [1]. In a CVRP, all customers have the same service type of delivery or receipt. All vehicles carry the same weight of load, each customer's demand is not greater than the vehicle's load, and the vehicle starts off and returns to the distribution center [1].

2.2. Mathematical Description of Optimization of VRPs

The definition of the Capacity-limited Vehicle Routing Problem (CVRP) is shown in Equation (1) [1].

$$\min D(i,j)=\sum_{i=0}^{N}\sum_{j=0}^{N}\sum_{s=1}^{K}c_{ij}x_{ijs}$$

$$s.t.\begin{cases}\sum_{i=0}^{N}x_{ijs}=y_{js}, j=1,2,\cdots,N; s=1,2,\cdots,K\\ \sum_{j=0}^{N}x_{ijs}=y_{is}, i=1,2,\cdots,N; s=1,2,\cdots,K\\ \sum_{i=0}^{N}q_i y_{is}\le q_s, s=1,2,\cdots,K\\ \sum_{s=1}^{K}y_{is}=\begin{cases}1, i=1,2,\cdots,N\\ K, i=0\end{cases}\end{cases} \tag{1}$$

Here, $D(i,j)$ represents the total path length of all vehicles, and c_{ij} represents the transportation distance between customer *i* and customer *j*. q_i represents the transportation demand of customer *i*, and q_s represents the load capacity of vehicle s. N is the total amount of customers, in which $i,j=1,2,\cdots,N$ represents the customer number, and *i* = 0 and *j* = 0 represent the number of the distribution center. K is the total amount of vehicles, in which $s=1,2,\cdots,K$ represents the vehicle number, and $\{x_{ijs}=1\}$ represents a directly connected path of the vehicle s between the customer *i* and the customer *j*; otherwise, $\{x_{ijs}=0\}$. y_{is} and y_{js} respectively represent that the needs of customer *i* and customer *j* are satisfied by the vehicles.

2.3. Basic HS Algorithm

The basic concept of the basic HS algorithm can be described as follows. It randomly generates harmonies such as $x^1,x^2\cdots,x^{HMS}$ to form harmony memory (HM), whose size is HMS. Each harmony is composed of *W* different tone components, i.e., if the harmony number is $j, j=1,2,\cdots,HMS$, the harmony is represented as $x^j=\left(x_1^j,x_2^j,\cdots,x_W^j\right)$. The initial form of HM is as expressed in Equation (2), where HMCR is the probability that takes values from the harmony memory, PAR is the probability of pitch fine-tuning, and BW is the bandwidth of pitch fine-tuning, $T_{\max}$ is the maximum number of algorithm evolutions. r_1, r_2, and *r* are random numbers generated by the rand () function when running in the basic harmony search algorithm, and $r_1,r_2,r\in[0,1]$.

$$HM=\left[\begin{array}{c|c}x^1 & f\left(x^1\right)\\ x^2 & f\left(x^2\right)\\ \vdots & \vdots\\ x^{HMS} & f\left(x^{HMS}\right)\end{array}\right]=\left[\begin{array}{cccc|c}x_1^1 & x_2^1 & \cdots & x_{\mathrm{W}}^1 & f\left(x^1\right)\\ x_1^2 & x_2^2 & \cdots & x_{\mathrm{W}}^2 & f\left(x^2\right)\\ \vdots & \vdots & \cdots & \vdots & \vdots\\ x_1^{HMS} & x_2^{HMS} & \cdots & x_{\mathrm{W}}^{HMS} & f\left(x^{HMS}\right)\end{array}\right] \tag{2}$$

According to Equation (3), set the new harmony to be $x^{new} = \left(x_1^{new}, x_2^{new}, \cdots, x_W^{new}\right)$; then, each tone component $x_i^{new}\left(i = 1, 2, \cdots, W\right)$ of the new harmony is generated by three rules: learning harmony memory, fine-tuning tone components, and the random selection of tone components. $x_i^{\mathrm{R}andom}$ refers to randomly generating a new harmonic tone component.

$$x_i^{new} = \begin{cases} x_i^{new} \in \left\{x_i^j \,\middle|\, j = 1, 2, \cdots, HMS\right\}, & 0 < r_1 \le HCMR \\ x_i^{\mathrm{R}andom} \quad , & HCMR < r_1 < 1 \end{cases} \tag{3}$$

Then, the tone component x_i^{new} produced by Equation (3) is judged by the probability PAR whether fine-toning adjustment is required, which is described as Equation (4).

$$x_i^{new^{new}} = \begin{cases} x_i^{new} \pm r \times BW, & 0 < r_2 \le PAR \\ x_i^{new}, & PAR < r_2 \le HCMR \end{cases} \tag{4}$$

Then, the harmony memory is updated according to the following update strategy described in Equation (5), where $f(x^i)$ represents the value of the objective function or fitness.

$$\begin{gathered} \text{If } \; f(x^{new}) < f(x^{worst}) = \max_{j=1,2,\cdots,HMS} f\left(x^j\right), \\ \text{then } \; x^{worst} = x^{new} \end{gathered} \tag{5}$$

The above process is repeated until the number of evolutions $T_{\max}$ is reached.

3. GSF-HS-VRP Algorithm

Similar to other swarm intelligence algorithms, the HS algorithm also has problems such as premature convergence and stagnant convergence, which are mainly caused by its own search mechanism. Firstly, the HS algorithm is a single-body evolutionary algorithm [16] in which a new harmony is randomly generated through learning harmony memory (HM), and then adjusted by the tone fine-tuning mechanism and bandwidth. This generated method has obvious defects such as certain singularity, poor search ability, and requiring a long number of evolutions to converge. Once the generated harmony is near the optimal harmony, it is easy to fall into local optimum and cause premature convergence. Secondly, the HS algorithm provides a mechanism to introduce a new harmony, i.e., to generate a new harmony by randomly selecting the tone just based on the value of the harmony memory value probability (HCMR) parameter. Since most of the parameters including BW and HCMR are chosen from the fixed experience value, the accuracy of the HS algorithm is not particularly high [22].

The paper addresses the singularity and randomness of the new harmony generation strategy; from the perspective of improving the diversity of algorithm solutions, the global sharing factor of the harmony search method is introduced to improve the new harmony generation strategy, and a new Harmony Search method with Global Sharing Factor based on natural number coding and decoding for the Vehicle Routing Problem (GSF-HS-VRP) is proposed and applied to solve the capacity-limited vehicle routing optimization problem.

3.1. Concept Definition

3.1.1. Global Sharing Factor of the Harmony Search Algorithm

The concept of global sharing factor was proposed in the literature [23,24]. It is a sharing factor that changes nonlinearly only along with the number of trials, and it increases rapidly from a small initial value to a steady-state value. Thus, it can be used to suppress the randomness of tone fine-tuning.

In this paper, the global sharing factor is adopted to the HS algorithm to improve the efficiency of the tone generation and fine-tuning mechanism.

The global share factor is shown in Equation (6).

$$\alpha_G = \left(1-\delta_G\right)\cdot\alpha_{final} \tag{6}$$

$$\delta_G = 1-\left(\alpha_{init}/\alpha_{final}\right)^{1/t}, t\in\left[1,T_{\max}\right] \tag{7}$$

Here, α_{G} is the global sharing factor, and δ_G is a parameter for calculating the global sharing factor, which is denoted in Equation (7). α_{init} and α_{final} are the initial value and final value of the parameters, respectively, and their values are set by experience as α_{init} = 0.1 and α_{final} = 1.2 [22,23]. T is the number of trials, and in Equation (7), t is the number of trials $t\in\left[1,T_{\max}\right]$.

3.1.2. Coding Method

In our algorithm, the initial harmony is encoded using a natural number encoding method. Assuming that the dimension of the tone component x_i^j is *W*, then a tone component x_i^j represents each customer number in the routing optimization natural number encoding, and each customer number should be unique. For example, using an eight-digit integer coding (*W* = 8), i.e., the total number of customers is *N* = 8, then the number from 1 to 8 represents each unique customer number, and 0 represents the number of the distribution center. Therefore, in the encoding, each tone component x_i^j must be different and unique. In the improved routing optimization method in this paper, $f(x^j), j=1,2,\cdots,HMS$ represents the total path length D of all vehicles, which is the final optimization objective function.

3.1.3. Learning Strategy

In the algorithm, the harmony memory whose size is HMS is randomly generated to find out the global optimal harmony $x^{g,now}=(x_1^g,x_2^g,\cdots,x_{\mathrm{W}}^g)$ in one harmony evolution. Each tone component $x_i^j, i=1,2,\cdots,W$ of each harmony $x^j, j=1,2,\cdots,HMS$ in the harmony memory adaptively evolves through the learning strategy, which is shown in Equation (8).

$$x_i^{j,new} = x_i^{j,old} + (\mathrm{int})(\alpha_G\times r\times(x^{g,now}-x_i^{j,old})) \tag{8}$$

3.1.4. Updating Strategy

In a harmony evolution, if the objective function value $f(x^{j,new})$ obtained by learning strategy evolution is slightly lower than $f(x^{j,old})$, i.e., $f(x^{j,new})\geq f(x^{j,old})$ exists. Then, every tone component $x_i^j, i=1,2,\cdots,W$ of each harmony $x^j, j=1,2,\cdots,HMS$ in the harmony memory adaptively evolves through the update strategy, which is shown in Equation (9).

$$x_i^{j,new} = x_i^{j,old} + (\mathrm{int})((1\text{-}\alpha_G)\times r\times(x^{g,now}-x_i^{j,old})) \tag{9}$$

3.1.5. Randomly Generating Strategy

In a harmony evolution, if the objective function value $f(x^{j,new})$ obtained by updating the strategy evolution is slightly lower than $f(x^{j,old})$, i.e., $f(x^{j,new}) \geq f(x^{j,old})$ exists, use Equation (10) to randomly generate the non-repeating harmony tone.

$$x_i^{j,new} = r\%W + 1 \tag{10}$$

The learning strategy, updating strategy, and randomly generating strategy are all calculated for each tone component of the harmony. In Equations (8)–(10), $x_i^{j,old}$ represents each tone component in the last evolution, and $x_i^{j,new}$ represents the new value of the corresponding tone component obtained by learning strategy, updating strategy, and randomly generation strategy after evolution.

3.1.6. Adjusting Strategy

When using the global sharing factor α_G to calculate the learning strategy and updating strategy for the tone component, there are two situations that need to be considered for adjustment.

The first case is whether the evolved $x_i^{j,new}$ still belongs to the correct coding range. $x_i^{j,new}$ represents a newly generated customer number whose value should belong to the range of natural number coding. If it exceeds the range, a random value calculated by the formula (int)((rand()%W)) to adjust its value to the right encoding range.

The second case is whether the evolved $x_i^{j,new}$ is the same as the value of other newly generated $x_i^{j,new'}$. The newly generated customer number $x_i^{j,new}$ should be unique; thus, if the situation of duplicate values occurs, they need to be compared one by one and modified with the method of randomly generating harmony tones.

3.1.7. Decoding Method

The arrangement of the vehicles is mainly based on the order of the customer number and the maximum load of the vehicle, i.e., the vehicle is arranged in the first vehicle path according to the customer's coding order until the maximum load limit of the vehicle is exceeded, and then the second vehicle path is arranged in turn, and so on until the customer is completely arranged. In this paper, the decoding method of [1] was adopted, in which the encoding order of the customer is decoded into the vehicle path sequence of each vehicle in accordance with the maximum load limit per vehicle.

3.1.8. Objective Function

The objective function, i.e., the fitness, refers to the sum of the path lengths according to the total number of vehicles. There are several vehicles with several paths, and the sum length of the paths is taken as the objective function. The calculation method for the objective function is shown in Equation (11). In this paper, each harmony is used as the input of the objective function. Thus, the value of objective function is the total length of the vehicle path generated by the *W* integer-encoded harmony tones, each of the harmony is composed of *N* customers, where *W* = *N*.

$$\begin{aligned} & if\left(\sum_{i=0}^{W} q[x_i^j][x_{i+1}^j]\right) \leq q_s, s = 1, 2, \cdots K \\ & then \qquad \min D(i,j) = f(x^j) = \sum_{i=0}^{W}\sum_{s=1}^{K} c_s[x_i^j][x_{i+1}^j], j = 1, 2, \cdots HMS \end{aligned} \tag{11}$$

Among them, $q[x_i^j][x_{i+1}^j]$ represents the transportation demand of customer *i*, $i = 1, 2, \cdots, W$ represents each unique customer number, and $i = 0$ represents the number of the distribution center. The path corresponding to the *s*-th ($s = 1, 2, \cdots, K$) vehicle always starts from the distribution center ($i=0$) and then returns to it ($i=0$). q_s represents the load capacity of the *s*-th vehicle. $c_s[x_i^j][x_{i+1}^j]$ represents the transportation distance between customer *i* and customer *i*+1 when the *s*-th vehicle is satisfied, and therefore, $\sum_{i=0}^{W}\sum_{s=1}^{K} c_s[x_i^j][x_{i+1}^j]$ represents the sum length of the total paths of K vehicles.

3.2. Theory of Algorithm

The theory of algorithm was described as follows. Set the maximum number of music creations to be $T_{\max}$. After initializing the harmony memory, the harmony fitness $f(x^j), j=1,2,\cdots,HMS$ is sorted in descending order to find the optimal harmony $x^{g,now}=(x_1^g,x_2^g,\cdots,x_W^g)$ in one harmony creation evolution. The global sharing factor α_G is calculated, and each tone component $x_i^j, i=1,2,\cdots,W$ of each harmony $x^j, j=1,2,\cdots,HMS$ in the harmony memory adaptively evolves through the learning strategy according to Equation (8). If $f(x^{j,new})<f(x^{j,old})$, replace $x_i^{j,old}$ with $x_i^{j,new}$ respectively. Otherwise, each tone component evolves through the updating strategy according to Equation (9). If still $f(x^{j,new})\geq f(x^{j,old})$, use Equation (10) to randomly generate harmony tone.

The above-mentioned harmony tone generation method is repeated until the number of evolution times reaches $T_{\max}$; then, the optimal harmony and optimized path sequence are output.

3.3. Algorithm Process

Algorithm: GSF-HS-VRP

Input: the maximum number of evolutions $T_{\max}$, the size of the harmony memory HMS, and the number of tone component *W*.

Output: global optimal harmony x^g and optimized path sequence.

Step 1: Randomly generate initial harmony individuals whose number is HMS, which is marked as. $X_{HMS}=\left\{x^j \middle| x^j=\left(x_1^j,x_2^j,\cdots,x_W^j\right), j=1,2,\cdots,HMS\right\}$. The harmony memory is initialized according to Equation (1). Calculate the fitness $f(x^j)$ of the *j*-th ($j=1,2,\cdots,HMS$) harmony x^j according to Equation (11).

Step 2: Each tone component $x_i^j, i=1,2,\cdots,W$ of each harmony $x^j, j=1,2,\cdots,HMS$ is updated.

```
Begin
Sort each harmony by fitness $f(x^j)$ in descending order to determine $x^{g,now}$.
Calculate $\alpha_G$ according to Equations (6) and (7).
For ($x_i^j, i=1,2,\cdots,W$)
    Update $x_i^{j,old}$ through the learning strategy according to Equation (8).
End
if ($f(x^{j,new})<f(x^{j,old})$)
    For ($x_i^j, i=1,2,\cdots,W$)
        $x_i^{j,old}=x_i^{j,new}$
    End
else
    For ($x_i^j, i=1,2,\cdots,W$)
```

```
            Update $x_i^{j,old}$ through the update strategy according to Equation (9).
        End
    if ( $f(x^{j,new}) < f(x^{j,old})$ )
        For ( $x_i^j, i = 1, 2, \cdots, W$ )
            $x_i^{j,old} = x_i^{j,new}$
        End
    else
        For ( $x_i^j, i = 1, 2, \cdots, W$ )
            Randomly generate harmony tone $x_i^{j,new}$ according to Equation (10).
        End
    End
```

Step 3: Determine whether the number of evolutions has reached $T_{\max}$. If not, go to Step 2. Otherwise, the algorithm stops and outputs x^g and the optimized path sequence.

3.4. Algorithm Flowchart

The algorithm flowchart of GSF-HS-VRP is shown in Figure 1.

3.5 Algorithm Efficiency Analysis

From the efficiency analysis of the HS algorithm in the literature, the time complexity of the HS algorithm is $O(T_{\max} \times W)$, and the space complexity is $O(HMS \times W)$. According to the previous descriptions of the improvements of the new algorithm, it can be concluded that the time complexity of the GSF-HS-VRP algorithm is $O(T_{\max} \times W \times HMS \times K)$, and the space complexity is $O(HMS \times W \times W + K)$. The time complexity and space complexity of GSF-HS-VRP varies with the values of W and K, which are dependent on the size of the small-scale data and Solomon example dataset.

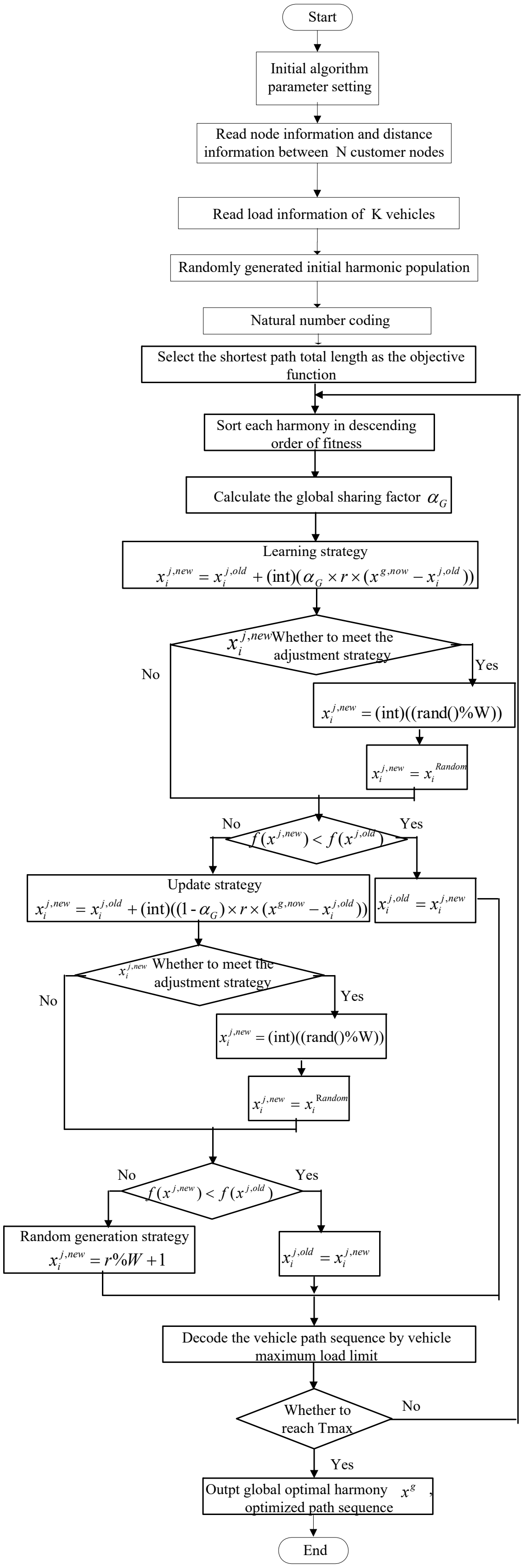
Start
Initial algorithm parameter setting
Read node information and distance information between N customer nodes
Read load information of K vehicles
Randomly generated initial harmonic population
Natural number coding
Select the shortest path total length as the objective function
Sort each harmony in descending order of fitness
Calculate the global sharing factor α_G
Learning strategy
$x_i^{j,new} = x_i^{j,old} + (\text{int})(\alpha_G \times r \times (x^{g,now} - x_i^{j,old}))$
$x_i^{j,new}$ Whether to meet the adjustment strategy
No
Yes
$x_i^{j,new} = (\text{int})((\text{rand}()\%\text{W}))$
$x_i^{j,new} = x_i^{Random}$
No
Yes
$f(x^{j,new}) < f(x^{j,old})$
Update strategy
$x_i^{j,new} = x_i^{j,old} + (\text{int})((1 - \alpha_G) \times r \times (x^{g,now} - x_i^{j,old}))$
$x_i^{j,old} = x_i^{j,new}$
$x_i^{j,new}$ Whether to meet the adjustment strategy
No
Yes
$x_i^{j,new} = (\text{int})((\text{rand}()\%\text{W}))$
$x_i^{j,new} = x_i^{\text{Random}}$
No
Yes
$f(x^{j,new}) < f(x^{j,old})$
Random generation strategy
$x_i^{j,new} = r\%W + 1$
$x_i^{j,old} = x_i^{j,new}$
Decode the vehicle path sequence by vehicle maximum load limit
Whether to reach Tmax
No
Yes
Outpt global optimal harmony x^g , optimized path sequence
End

Figure 1. Flowchart of Harmony Search method with Global Sharing Factor based on natural number coding and decoding for Vehicle Routing Problem (GSF-HS-VRP) algorithm.

4. Experiment Design and Results Analysis

4.1. Experiment Design

Experiments were carried out on the HS algorithm Improved Harmony Search (IHS) algorithm [25], Global-best Harmony Search (GHS) algorithm [26], Self-adaptive Global Best Harmony Search (SGHS) Algorithm [27], and GSF-HS-VRP algorithm to optimize the vehicle routing problem. The computer processor of the personal computer used in the experiment was Intel Core i5 2.6 GHz and it had a memory of 4.0 GB. The experiment's programs were programmed by Visual C++ 2010. The final test results were averaged after 30 runs independently.

The experimental parameters were set as follows: HMS = 20, HMCR = 0.9, PAR = 0.3, BW = 0.01, $T_{\max}$ = 20. In the IHS algorithm, the minimum value of pitch tune probability PARmin = 0.01, the maximum value of pitch tune probability PARmax = 0.99, the minimum value of pitch tune bandwidth BWmin = 0.0001, and the maximum value of pitch tune bandwidth BWmax = $1/\left(20\times\left(UB-LB\right)\right)$ [25]. In the GHS algorithm, PARmin = 0.01, PARmax = 0.99 [22]. In the SGHS algorithm, the initial probability value of the harmony memory is HMCRm = 0.98, the standard deviation of the value of the harmony memory HMCRs = 0.01, and the probability of the harmony memory is normally distributed within [0.9,1.0], the average initial value of bandwidth PARm = 0.9, the standard deviation of pitch tune bandwidth PARs = 0.05, the pitch tune bandwidth is normally distributed within [0.0,1.0], the specific evolution numbers Learning Period (LP) = 10, minimum value of the pitch tune bandwidth BWmin = 0.0001, the maximum value of the pitch tune bandwidth BWmax = $1/\left(20\times\left(UB-LB\right)\right)$ [27]. UB and LB are the maximum value and minimum value of the search range of the algorithm, respectively. In the GSF-HS-VRP algorithm of this paper, according to sharing factor theory and experimental analyses in the literature [22,23], α_{init} is set to 0.1, and α_{final} is set to 1.2.

According to the natural number coding method, adjusting strategy, and encoding strategies, the HS, IHS, GHS, and SGHS algorithms in the experiments have been modified to solve the vehicle routing problem, which is to ensure the consistency of the parameters and optimization results of all the five algorithms in the experimental tests.

4.2. Results Analysis

4.2.1. Experiment Results and Analysis of Small-Scale Data

Firstly, the experiment was tested using small-scale data [1]. The distance $c_s[x_i^j][x_{i+1}^j]$ between customer *i* and customer *i*+1 and the demand $q[x_i^j][x_{i+1}^j]$ of customer *i* are shown in Table 1. The experimental parameters were set as follows: the number of customer nodes is *N* = 8, i.e., the number of harmony tone components is *W* = 8, where 1–8 represents each unique customer number, and 0 represents the number of the distribution center. The number of vehicles is *K* = 2, and the capacity of each car is q_s = 8. The comparison of the results of fixed iterations and average runtime are shown in Table 2, and the average optimum of each algorithm is shown in Figure 2. The comparison results of vehicle routing optimization for each algorithm are shown in Table 3.

Table 1. Table of customer distance and demand.

$c_s[x_i^j][x_{i+1}^j]$	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**
0	0	4	6	7.5	9	20	10	16	8
1	4	0	6.5	4	10	5	7.5	11	10
2	6	6.5	0	7.5	10	10	7.5	7.5	7.5

3	7.5	4	7.5	0	10	5	9	9	15
4	9	10	10	10	0	10	7.5	7.5	10
5	20	5	10	5	10	0	7	9	7.5
6	10	7.5	7.5	9	7.5	7	0	7	10
7	16	11	7.5	9	7.5	9	7	0	10
8	8	10	7.5	15	10	7.5	10	10	0
Demand $q[x_i^j][x_{i+1}^j]$ **/Ton**	0	1	2	1	2	1	4	2	2

Table 2. Comparison of the results of fixed iterations. HS: Harmony Search, IHS: Improved Harmony Search, GHS: Global-best Harmony Search, SGHS: Self-adaptive Global Best Harmony Search.

Algorithm	Average Optimum	Standard Deviation	Average Runtime
HS-VRP	7.15E+01	3.28E+00	7.27E+00
IHS-VRP	7.14E+01	5.09E+00	6.62E+00
GHS-VRP	7.06E+01	4.97E+00	7.69E+00
SGHS-VRP	7.04E+01	4.91E+00	1.30E+01
GSF-HS-VRP	6.96E+01	4.45E+00	7.01E+00

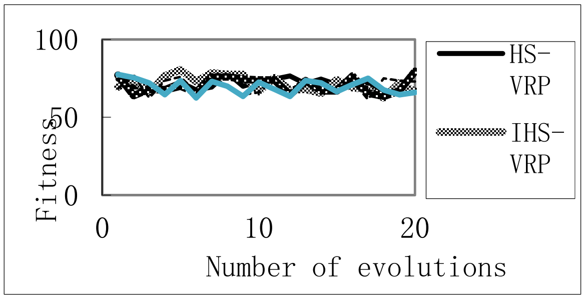

Figure 2. Average optimum of each algorithm.

Table 3. Comparison of vehicle routing optimization for each algorithm.

Algorithm	Shortest Path Value	Optimization Path of Vehicle 1	Optimization Path of Vehicle 2
HS-VRP	67.50	3-7-8	6-5-2
IHS-VRP	66.50	1-2-5-3	6-7-4
GHS-VRP	79.50	6-2-3	4-7-5-8-1
SGHS-VRP	73.00	1-2-7	6-8-4
GSF-HS-VRP	66.00	4-5-1-3	6-7-2

4.2.2. Experiment Results and Analysis of Standard Test Data

Then, the standard test data in the famous Solomon example [28] is used for the experiment, and the C101 dataset is selected as the test set. Experimental parameters were set as follows: the number of customer nodes is $N = 100$, i.e., the number of harmony tone components is $W = 100$, where 1 to 100 represents each unique customer number and 0 represents the number of the distribution center. The tests are conducted according to the number of vehicles $K = 2$, 3, and 10, respectively, and the vehicle capacity of each vehicle is 200.

Results and Comparison of Two Vehicles

In the case of two vehicles, i.e., $K = 2$, the comparison results of fixed iteration times and average runtime are shown in Table 4. The average optimum iteration curves of four algorithms were drawn in Figure 3. The comparison results of vehicle routing optimization of each algorithm are shown in Table 5.

Table 4 Comparison results at fixed iteration times.

Algorithm	Average Optimum	Standard Deviation	Average Runtime
HS-VRP	6.40E+02	4.75E+01	1.56E+02
IHS-VRP	6.50E+02	4.59E+01	6.15E+01
GHS-VRP	6.33E+02	5.56E+01	9.69E+01
SGHS-VRP	6.61E+02	6.35E+01	3.97E+02
GSF-HS-VRP	6.39E+02	4.20E+01	3.12E+01

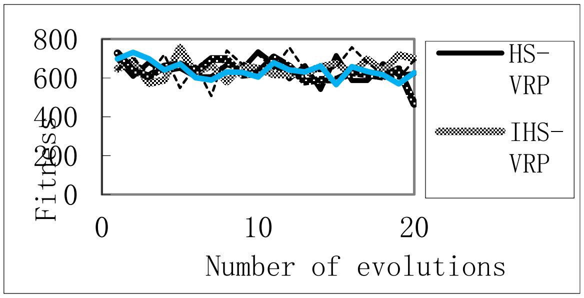

Figure 3. Average optimum iteration curve of each algorithm.

Table 5. Comparison results of vehicle routing optimization of each algorithm.

Algorithm	Shortest Path Value	Optimization Path of Vehicle 1	Optimization Path of Vehicle 2
HS-VRP	622.39	69-35-38-95-53-58-56-66	93-90-60-59-20-24-7-74-39-3
IHS-VRP	699.87	93-2-6-22-43-35-54-69-28	68-34-50-30-17-58-80-26-63-45-39
GHS-VRP	645.82	7-70-58-38-74-33	10-78-81-54-49-24-20-5-17-27-30-69-72
SGHS-VRP	692.53	27-33-1-95-46-58-34-84-49	69-68-10-63-23-9-61-21-89-5-15
GSF-HS-VRP	569.83	54-42-13-89-2-100-41-22	88-26-85-25-74-50-20-34

Results and Comparison of Three Vehicles

In the case of three vehicles, i.e., $K = 3$, the comparison results of fixed iteration times and average runtime are shown in Table 6. The average optimum iteration curves of five algorithms are drawn in Figure 4. The comparison results of vehicle routing optimization of each algorithm are shown in Table 7.

Table 6. Comparison results of fixed iteration times.

Algorithm	Average Optimum	Standard Deviation	Average Runtime
HS-VRP	1.02E+03	7.21E+01	7.11E+01
IHS-VRP	1.03E+03	8.10E+01	7.13E+01
GHS-VRP	1.05E+03	7.23E+01	1.24E+02
SGHS-VRP	1.02E+03	7.60E+01	5.93E+02
GSF-HS-VRP	9.77E+02	7.63E+01	3.66E+01

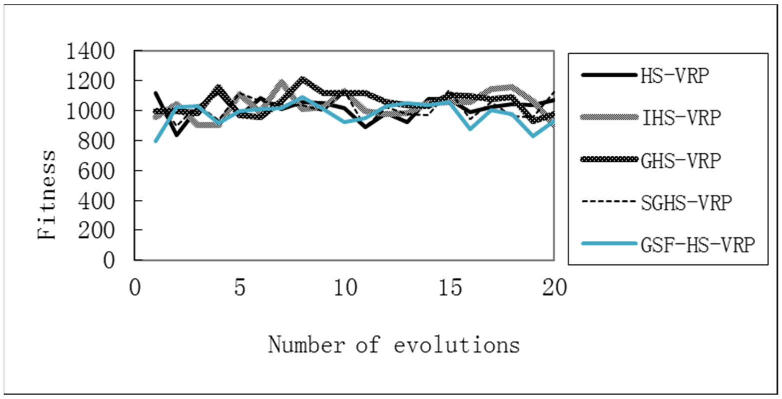

Figure 4. Average optimum iteration curve of each algorithm.

Table 7. Comparison results of vehicle routing optimization of each algorithm.

Algorithm	**Shortest Path Value**	**Optimization Path of Vehicle 1**	**Optimization Path of Vehicle 2**	**Optimization Path of Vehicle 3**
HS-VRP	1068.12	23-4-50-40-81-86-78-85-11-18-8-37	43-76-88-21-92-15-6-25-59	28-74-98-94-49-34-22-52-95-1
IHS-VRP	1062.02	33-16-19-64-73-66-86-44-21-38-68	95-11-2-63-98-61-57-40	100-29-48-6-89-74-20-80-84-69-72
GHS-VRP	972.97	42-74-8-7-6-2-26-48	56-69-95-52-53-18-68-50-94-43-27	16-14-45-10-63-81-54
SGHS-VRP	1125.37	22-66-63-6-83-12-50-13	54-58-87-88-52-60-17-89-55	95-56-10-93-42-35-64-18-8
GSF-HS-VRP	928.07	55-44-82-1-52-88-74-18-90	54-53-85-93-100-37-26-3	33-47-63-60-81-66-32

Results and Comparison of Ten Vehicles

In the case of 10 vehicles, i.e., $K = 10$, the comparison results of fixed iteration times and average runtime are shown in Table 8. The average optimum iteration curves of five algorithms are shown in Figure 5. The comparison results of the vehicle routing optimization of each algorithm are shown in Table 9.

Table 8. Comparison results at fixed iteration times.

Algorithm	Average Optimum	Standard Deviation	Average Runtime
HS-VRP	3.86×10^3	9.63×10^1	3.05×10^2
IHS-VRP	3.84×10^3	9.39×10^1	3.08×10^2
GHS-VRP	3.86×10^3	9.45×10^1	3.12×10^2
SGHS-VRP	3.83×10^3	7.25×10^1	1.99×10^3
GSF-HS-VRP	3.82×10^3	6.08×10^1	7.45×10^1

Table 9. Comparison optimization path results of the vehicle routing optimization of each algorithm.

Algorithm	Shortest Path Value	Vehicle 1	Vehicle 2	Vehicle 3	Vehicle 4	Vehicle 5	Vehicle 6	Vehicle 7	Vehicle 8	Vehicle 9	Vehicle 10
HS-VRP	3964.08	60-48-52-33-11-23-28-58-15	71-90-34-27-65-3-36-87-1-6-31-84-68	7-62-96-39-57-40-64-56-8	93-86-24-20-99-82-80-76-73-49-4-18	74-32-12-21-25-53-89	38-59-29-26-50-72-37-61-95-77-81-67-91	51-16-85-75-70-46-97	13-2-100-94-42-83-17-63-79	45-69-10-92-5-41-54-78-19-47-66-55-14-43-44	35-88-30-98-22-9
IHS-VRP	3939.66	63-100-75-59-43-68-16-56-89	69-87-98-1-54-57-62-8-51-40	24-65-10-70-66-53-14-18-35-49-76-85-78	94-12-81-77-9-73-61-5-11-7-79-80-31-29-23	97-42-72-17-3-93-27-38-46	41-13-26-30-21-34-33-96-39-19-45	92-2-64-55-67-52-48-20-15-84-50-82	74-32-47-91-95-44-90-37-86-36-99	71-58-28-22-4-60-88-6	12-36-69-0-0-25-83
GHS-VRP	3995.30	25-82-9-49-16-64-6-99-26-89-20	92-85-24-60-14-38-75-4-48-34-52	8-91-12-54-93-95-51-31	100-80-18-87-76-81-2-23-35-33	19-22-42-47-46-84-65-29-78-79-71-30-59	43-73-72-41-67-44-88-90-86-10-7-56-28	98-77-27-37-68-36-83-32-61-15-55-50	63-3-1-53-45-40-96-13-94-17-66	97-5-57-39-58-11-69-62-21	80-72-4-19-17-0-0-74-70
SGHS-VRP	3950.63	76-15-58-49-73-86-93-7	74-62-95-33-18-10-69-42	96-6-64-65-78-21-48-85-54-79	81-9-46-39-68-72-83-61-27-12-22-34	31-51-82-14-43-57-29-41-60-56-94-91	13-55-19-99-3-23-1-8-92-90-84-17-44-20	100-75-77-26-47-4-87-66-67-53-38-40-28	25-50-70-80-98-45-88-59-37-5	71-89-24-2-52-32-11-97-16-30	36-63-35
GSF-HS-VRP	3855.13	35-84-100-68-34-5-53-37-4-67-65-46	6-92-2-96-10-60-94-17-45-54-59	32-78-98-51-18-13-30-21-70-86	79-75-8-9-19-91-25-99-27-1-48-39	16-88-95-97-7-74	14-12-69-61-93-77-50-55-41-62-44-24-43	15-26-38-23-31-40-71-72-83-3-81	73-64-56-28-76-42-36-47-85-22-80-89-20	63-82-57-66-87-90-58	33-49-29-52-11

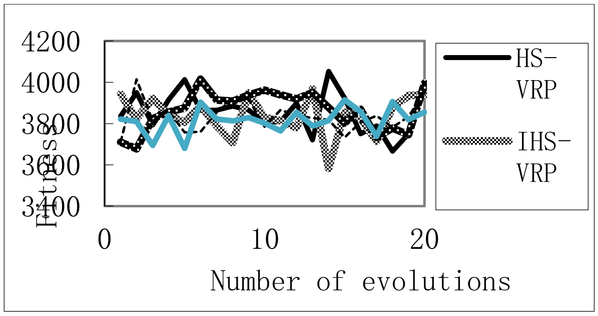

Figure 5. Average optimum iteration curve of each algorithm.

4.2.3. Wilcoxon's Test

Wilcoxon's test can be used to perform individual comparisons between two algorithms (pairwise comparisons), and the *p*-value in a pairwise comparison is independent from another one [13,29]. We conducted the behaviors of the five algorithms by means of Wilcoxon's rank sum test, which known as a non-parametric statistical significance proof for independent samples. The *p*-values produced by Wilcoxon's test in the experiments are shown in Table 10. Through the comparison of the computed *p*-values, we can draw the conclusion that the results of the GSF-HS-VRP algorithm are better than the ones acquired by the other four algorithms, especially for the problem of the Solomon example ($K = 3$).

Table 10. *P*-Values produced by Wilcoxon's test for the different datasets.

GSF-HS-VRP vs.	HS-VRP	IHS-VRP	GHS-VRP	SGHS-VRP
Small-scale data [1]	0.0255	0.0244	0.0440	0.0481
Solomon example $K = 2$ [28]	0.0882	0.0543	0.0860	0.0126
Solomon example $K = 3$ [28]	0.0068	0.0108	0.0010	0.0133
Solomon example $K = 10$ [28]	0.0208	0.0199	0.0468	0.0561

4.2.4. Analysis of Results

It can be seen from the above test results that the GSF-HS-VRP algorithm shows a better average optimum for both small-scale data and standard test data. Compared with the other four algorithms, the GSF-HS-VRP algorithm has the shortest runtimes, achieves the effect of rapid convergence, and improves the efficiency of the algorithm. The algorithm's average optimum iteration curve also shows that under the condition of fixed evolution times, the GSF-HS-VRP algorithm has a relatively stable evolutionary curve, making it less prone to premature convergence. In the multi-vehicle test, with the increase of the number of vehicles, the optimized path of the vehicle can satisfy the actual needs of more customers. The above results demonstrated that the GSF-HS-VRP algorithm has higher efficiency and reliability than the other four algorithms, which can effectively improve the optimization performance of the capacity limited-vehicle routing problem.

5. Conclusions

In this paper, a new Harmony Search method with Global Sharing Factor based on natural number coding for the Vehicle Routing Problem is proposed. From the perspective of improving the diversity of algorithm solutions, the number of creations of the harmony search algorithm is applied to the sharing factor, and the global sharing factor of the HS algorithm is proposed to improve the new harmony generation strategy. This method adopts natural number coding with high efficiency. The test results show that the optimal path results of this method are significantly improved compared with other HS, IHS, GHS, and SGHS algorithms in solving the problem of capacity-limited vehicle path optimization.

Author Contributions: L.L. conceived and designed the Harmony Search Method and the experiments, and wrote the paper; J.H., F.X. and Y.D. performed the experiments and analyzed the experiments. All authors have read and agreed to the published version of the manuscript.

Funding: This work is supported by the Gansu Agricultural University Discipline Construction Fund Project (Grant No. GAU-XKJS-2018-253, GAU-XKJS-2018-255, GAU-XKJS-2018-251), Gansu Province Higher Education Research Project (Grant No. 2019B-086, 2019A-056), Development Foundation Project of College of Information Science and Technology of Gansu Agricultural University (Grant No. GAU-XKFZJJ-2020-11), Young Advisor Fund of Gansu Agricultural University (Grant No. GAU-QDFC-2019-02), National Nature Science Foundation of China (Grant No. 61741201), 2019 General Planning Project of the 13th Five-Year Plan for Education Science in Gansu Province (Grant No. GS[2019]GHB2152), National Higher Education for Basic Computer Education Research Project (Grant No. 2018-AFCEC-214，2018-AFCEC-213).

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

References

1. Yan, T. Research on Improved Harmony Search Algorithms for VRP. Master's Thesis, Zhejiang Normal University, Hangzhou, China, 25 March 2015.
2. Hu, L.; Wang, Z.; Zhao, F. Gaussian harmony search algorithm for driver satisfaction based logistics distribution path optimization. *Appl. Res. Comput.*, **2015**, *32*, 3622–3625, 3676.
3. Lau, H.; Chan, T.; Tsui, W.; Pang, W. Application of genetic algorithms to solve the multidepot vehicle routing problem. *IEEE Trans. Autom. Sci. Eng.* **2010**, *7*, 383–392.
4. Murray, C.; Park, W. Incorporating human factor considerations in unmanned aerial vehicle routing. *IEEE Trans. Syst. Man Cybern. Syst*. **2013**, *43*, 860–869.
5. Pavone, M.; Frazzoli, E.; Bullo, F. Adaptive and distributed algorithms for vehicle routing in a stochastic and dynamic environment. *IEEE Trans. Autom. Control* **2011**, *56*, 1259–1269.
6. Zhao, Z.; Wan, F. The solving of Vehicle Routing Problem based on hybrid harmony search algorithm. In Proceedings of the 3rd International Conference on Computational Intelligence and Industrial Application (PACIIA), Wuhan, China, 6–7 November 2010; pp. 379–382.
7. Karaboga, D.; Basturk, B. On the Performance of Artificial Bee Colony Algorithm. *Appl. Soft Comput.* **2008**, *8*, 687–697.
8. Chen, S.; Locust, S. A new multi-optima search technique. In Proceedings of the Eleventh Conference on Congress on Evolutionary Computation, Trondheim, Norway, 18-21 May 2009.
9. Mirjalili, S. Moth-flame optimization algorithm: A novel nature-inspired heuristic paradigm. *Knowl.-Based Syst.* **2015**, *89*, 228–249.
10. Eusuff, M.; Lansey, K. Optimization of Water Distribution Network Design Using the Shuffled Frog Leaping Algorithm. *J. Water Sources Planning Manag*. **2003**, *129*, 210–225.
11. Pan, W. A new fruit fly optimization algorithm: Taking the financial distress model as an example. *Knowl.-Based Syst.* **2012**, *29*, 69–74.
12. Geem, Z.; Kim J. Loganathan G. A new heuristic optimization algorithm: Harmony search. *Simulation* **2001**, *76*, 60–68.
13. Cuevas, E.; González, A.; Zaldívar, D.; Pérez-Cisneros, M. An optimisation algorithm based on the behaviour of locust swarms. *Int. J. Bio-Inspir. Comput*. **2015**, *7*, 402–407.
14. Wang, G.; Jin, J. Moth-flame optimization algorithm fused on refraction principle and opposite-based learning. *Comput. Eng. Appl.* **2019**, *55*, 46-51.
15. Han, H.; Pan, Q.; Liang, J. Application of improved harmony search algorithm in function optimization. *Comput. Eng.* **2010**, *36*, 245–247.
16. Zhao, P.; Liu, S. Novel intelligent optimization algorithm and its improvement. *J. Chin. Comput. Syst.* **2010**, *31*, 955–958.
17. Cuevas, E.; Block-matching algorithm based on harmony search optimization for motion estimation. *Appl. Intell*. **2013**, *39*, 165–183.
18. Alia, O.; Mandava, R.; Aziz, M. A hybrid harmony search algorithm for MRI brain segmentation. *Evol. Intell.* **2011**, *4*, 31–49.
19. Fourie, J. Robust circle detection using Harmony Search. *J. Optim*. **2017**, *2017*, doi:10.1155/2017/9710719.

20. Moon, Y.; Zong, W.; Han, G. Vanishing point detection for self-driving car using harmony search algorithm. *Swarm Evol. Comput.* **2018**, *41*, 111–119.
21. Das, S.; Mukhopadhyay, A.; Roy, A; Abraham, A.; Panigrahi, B. Exploratory power of the Harmony Search Algorithm: Analysis and improvements for global numerical optimization. *IEEE Trans. Syst. Man Cybern.* **2011**, *41*, 89–106.
22. Liu, L.; Huo J.; Wang, L.; Han, J. Harmony search algorithm based on shuffled frog leaping and bacterial foraging and its application in image. *J. Front. Comput. Sci. Technol.* 2**015**, *9*, 119–128.
23. Liu, L.; Huo J.; Wang, L. Harmony search algorithm with global sharing factor. *J. Chongqing Univ. Technol. (Nat. Sci.)* **2014**, *28*, 82–86.
24. Liu, L.; Wang, L.; Han, J. Shuffled frog leaping algorithm based on global sharing factor. *Comput. Eng*. **2013**, *39*, 162–166, 171.
25. Mahdavi, M.; Fesanghary, M.; Damangir, E. An improved harmony search algorithm for solving optimization problems. *Appl. Math. Comput.* **2007**, *188*, 1567–1579.
26. Omran, M.; Mahdavi, M. Global-best harmony search. *Appl. Math. Comput.* **2008**, *198*, 634–656.
27. Pan, Q.K.; Suganthan, P.N.; Tasgetiren, M.F.; Liang, J.J. A self-adaptive global best harmony search algorithm for continuous optimization problems. *Appl. Math. Comput.* **2010**, *216*, 830–848.
28. Available online: http://web.cba.neu.edu/~msolomon/problems.htm (accessed on 4 February 2020).
29. García, S.; Molina, D.; Lozano, M.; Herrera, F. A study on the use of non-parametric tests for analyzing the evolutionary algorithms' behaviour: A case study on the CEC'2005 Special Session on Real Parameter Optimization. *J. Heuristics* **2009**, *15*, 617–644.