Harmony Search for Generalized Orienteering Problem: Best Touring in China

Zong Woo Geem[1], Chung-Li Tseng[2], and Yongjin Park[3]

[1] Johns Hopkins University, Environmental Planning and Management Program,
729 Fallsgrove Drive #6133, Rockville, Maryland 20850, USA
geem@jhu.edu
[2] University of Missouri,
Department of Engineering Management,
215 Engineering Management, Rolla,
Missouri 65409, USA
chungli@umr.edu
[3] Keimyung University,
Department of Transportation Engineering,
1000 Sindang, Dalseo, Daegu, 704-701, South Korea
ypark@kmu.ac.kr

Abstract. In order to overcome the drawbacks of mathematical optimization techniques, soft computing algorithms have been vigorously introduced during the past decade. However, there are still some possibilities of devising new algorithms based on analogies with natural phenomena. A nature-inspired algorithm, mimicking the improvisation process of music players, has been recently developed and named Harmony Search (HS). The algorithm has been successfully applied to various engineering optimization problems. In this paper, the HS was applied to a TSP-like NP-hard Generalized Orienteering Problem (GOP) which is to find the utmost route under the total distance limit while satisfying multiple goals. Example area of the GOP is eastern part of China. The results of HS showed that the algorithm could find good solutions when compared to those of artificial neural network.

1 Introduction

Over the several decades, optimization techniques such as linear programming (LP), non-linear programming (NLP), and dynamic programming (DP) have gathered attention among engineers. However, the mathematical techniques can excellently perform mostly in simple and ideal models.

In order to overcome the shortcomings of mathematical techniques, nature-inspired soft computing algorithms have been introduced. Many evolutionary or meta-heuristic algorithms have been developed that combine rules and randomness mimicking natural phenomenon [1-8].

The purpose of this paper is to introduce a recently-developed nature-inspired algorithm, Harmony Search, and to apply the algorithm to a TSP-like NP-hard Generalized Orienteering Problem (GOP), proposed by Wang et al. [9].

L. Wang, K. Chen, and Y.S. Ong (Eds.): ICNC 2005, LNCS 3612, pp. 741 – 750, 2005.

2 Harmony Search Algorithm

Harmony Search (HS) algorithm was recently developed in an analogy with music improvisation process where music players improvise the pitches of their instruments to obtain better harmony [10].

The HS algorithm has been successfully applied to various benchmarking and real-world problems including traveling salesperson problem [10], parameter optimization of river flood model [11], design of pipeline network [12], and design of truss structures [13]. Consequently, the HS algorithm provides a possibility of success in a TSP-like NP-hard problem.

As existing soft computing algorithms are found in the paradigm of natural processes, a new algorithm can be conceptualized from a musical performance process (for example, a jazz trio) involving searching for a better harmony. Musical performance seeks a best state (fantastic harmony) determined by aesthetic estimation, as the optimization process seeks a best state (global optimum: minimum cost; minimum error; maximum benefit; or maximum efficiency) determined by objective function evaluation. Aesthetic estimation is done by the set of the pitches sounded by joined instruments, as objective function evaluation is done by the set of the values produced by composed variables; the aesthetic sounds can be improved practice after practice, as the objective function values can be improved iteration by iteration.

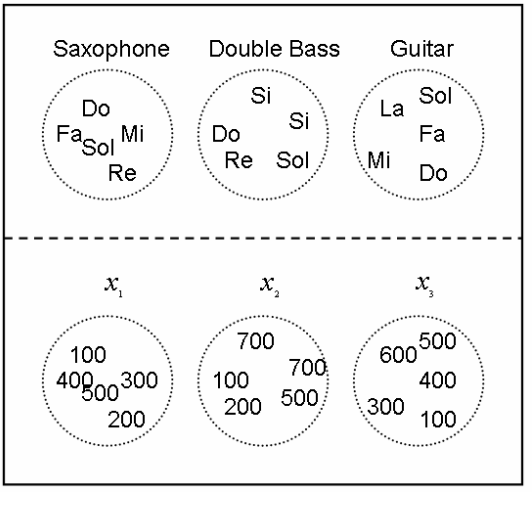

Fig. 1. Structure of Harmony Memory

Figure 1 shows the structure of the Harmony Memory (HM) that is the core part of the HS algorithm. Consider a jazz trio composed of saxophone, double bass, and guitar. There exist certain amount of preferable pitches in each musician's memory: saxophonist, {Do, Fa, Mi, Sol, Re}; double bassist, {Si, Do, Si, Re, Sol}; and guitarist, {La, Sol, Fa, Mi, Do}. If saxophonist randomly plays Sol out of its memory {Do, Fa, Mi, Sol, Re}, double bassist Si out of {Si, Do, Si, Re, Sol}, and guitarist Do out of

{La, Sol, Fa, Mi, Do}, the new harmony (Sol, Si, Do) becomes another harmony (musically C-7 chord). And if this new harmony is better than existing worst harmony in the HM, the new harmony is included in the HM and the worst harmony is excluded from the HM. This procedure is repeated until fantastic harmony is found.

In real optimization, each musician can be replaced with each decision variable, and its preferred sound pitches can be replaced with each variable's preferred values. Let us set that each decision variable represents pipe diameter between two nodes and the music pitches {Do, Re, Mi, Fa, Sol, La, Si} correspond to pipe diameters {100mm, 200mm, 300mm, 400mm, 500mm, 600mm, 700mm}, respectively. And if first variable chooses 500mm out of {100mm, 400mm, 300mm, 500mm, 200mm}, second one {700mm} out of {700mm, 100mm, 700mm, 200mm, 500mm}, and third one {100mm} out of {600mm, 500mm, 400mm, 300mm, 100mm}, those values (500mm, 700mm, 100mm) make another solution vector. And if this new vector is better than existing worst vector in the HM, the new vector is included in the HM and the worst vector is excluded from the HM. This procedure is repeated until certain stopping criterion is satisfied.

According to the above algorithm concept, the steps of HS for the generalized orienteering problem are as follows:

Step 1. Initialize the Parameters for Problem and Algorithm.

Step 2. Initialize the Harmony Memory (HM).

Step 3. Improvise a New Harmony.

Step 4. Update the Harmony Memory.

Step 5. Check the stopping criterion.

2.1 Initialize Parameters

In Step 1, the optimization problem is specified as follows:

$$\text{Minimize } f(\mathbf{x}). \tag{1}$$

$$\text{Subject to } x_i \in \mathbf{X}_i, i = 1,2,...,N. \tag{2}$$

where $f(\mathbf{x})$ is an objective function; $\mathbf{x}$ is the set of each decision variable x_i; $\mathbf{X}_i$ is the set of possible range of values for each decision variable, that is, $\mathbf{X}_i = \{x_i(1), x_i(2),..., x_i(K)\}$ for discrete decision variables $(x_i(1) < x_i(2) < ... < x_i(K))$; N is the number of decision variables (number of music instruments); and K is the number of possible values for the discrete variables (pitch range of each instrument).

For the GOP, the objective function becomes the total score of individual goals, as shown in Equation 7; and each decision variable represents each city, having the value of next city number to move.

The HS algorithm parameters are also specified in this step: Harmony Memory Size (HMS) (= number of solution vectors), Harmony Memory Considering Rate (HMCR), Pitch Adjusting Rate (PAR), and Stopping Criteria (= number of improvisation). Here, HMCR and PAR are the parameters of HS algorithm explained in Step 3.

2.2 Initialize Harmony Memory

In Step 2, the Harmony Memory (HM) matrix, as shown in Equation 3, is filled with as many randomly generated solution vectors as the size of the HM (HMS).

$$\begin{bmatrix} x_1^1 & x_2^1 & \cdots & x_{N-1}^1 & x_N^1 \\ x_1^2 & x_2^2 & \cdots & x_{N-1}^2 & x_N^2 \\ \vdots & \cdots & \cdots & \cdots & \cdots \\ x_1^{HMS-1} & x_2^{HMS-1} & \cdots & x_{N-1}^{HMS-1} & x_N^{HMS-1} \\ x_1^{HMS} & x_2^{HMS} & \cdots & x_{N-1}^{HMS} & x_N^{HMS} \end{bmatrix} \begin{matrix} \Rightarrow f(\boldsymbol{x}^1) \\ \Rightarrow f(\boldsymbol{x}^2) \\ \Rightarrow \vdots \\ \Rightarrow f(\boldsymbol{x}^{HMS-1}) \\ \Rightarrow f(\boldsymbol{x}^{HMS}) \end{matrix} \tag{3}$$

2.3 Improvise New Harmony

A new harmony vector, $\boldsymbol{x}' = (x_1', x_2', ..., x_N')$ is generated by following three rules: HM consideration; Pitch adjustment; or totally random generation. For instance, the value of the first decision variable (x_1') for the new vector can be chosen from values stored in HM ($x_1^1 \sim x_1^{HMS}$). Value of other variables (x_i') can be chosen in the same style. There is also a possibility that totally random value can be chosen. HMCR parameter, which varies between 0 and 1, sets the rate whether a value stored in HM is chosen or a random value is chosen, as follows:

$$x_i' \leftarrow \begin{cases} x_i' \in \{x_i^1, x_i^2, ..., x_i^{HMS}\} & w.p. \quad HMCR \\ x_i' \in \boldsymbol{X}_i & w.p. \quad (1\text{-}HMCR) \end{cases} \tag{4}$$

The HMCR is the rate of choosing one value from historical values stored in HM while (1-HMCR) is the rate of randomly choosing one value from the possible value range.

After choosing the new harmony vector $\boldsymbol{x}' = (x_1', x_2', ..., x_N')$, pitch-adjusting decision is examined for each component of the new vector. This procedure uses the PAR parameter to set the rate of pitch adjustment as follows:

$$x_i' \leftarrow \begin{cases} \text{Adjusting Pitch} & \text{w.p.} \quad PAR \\ \text{Doing Nothing} & \text{w.p.} \quad (1-PAR) \end{cases} \tag{5}$$

In the pitch adjusting process, a value moves to its neighboring value with probability of PAR, or just stays in its original value with probability (1-PAR). If the pitch adjustment for x_i' is determined, its position in the value range $\mathbf{X}_i$ is identified in the form of $x_i(k)$ (the kth element in $\mathbf{X}_i$), and the pitch-adjusted value for $x_i(k)$ becomes

$$x_i' \leftarrow x_i(k+m). \tag{6}$$

where $m \in \{..., -2, -1, 1, 2, ...\}$ is a neighboring index used for discrete-type decision variables.

The HMCR and PAR parameters in Harmony Search help the algorithm find globally and locally improved solution, respectively.

2.4 Update Harmony Memory

If the new harmony vector, $\boldsymbol{x}' = (x_1', x_2', ..., x_N')$ is better than the worst harmony in the HM, judged in terms of the objective function value, the new harmony is included in the HM and the existing worst harmony is excluded from the HM.

2.5 Check Stopping Criterion

If the stopping criterion (maximum number of improvisations) is satisfied, computation is terminated. Otherwise, Steps 3 and 4 are repeated.

3 Generalized Orienteering Problem

In this study, HS is applied to generalized orienteering problem (GOP). The objective of GOP is to find the optimal tour under the constraint of total distance limit while satisfying multiple goals.

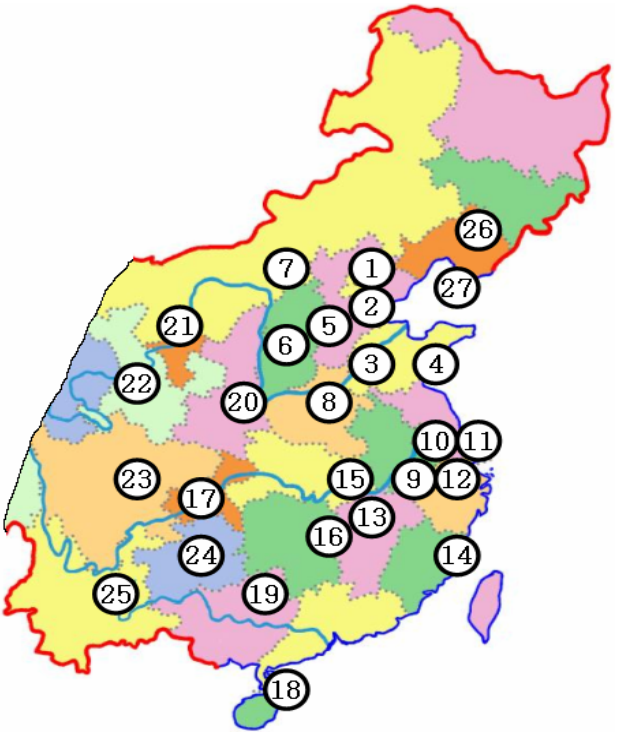

Fig. 2. Map of 27 cities in eastern part of China

If a traveler visits eastern part of China, as shown in Figure 2, and he/she wants to travel as many cities as possible with the purpose of best fulfilling multiple factors such as 1) natural beauty, 2) historical interest, 3) cultural event, and 4) business opportunities under the limited total moving distance, his/her travel can become generalized orienteering problem where each city has certain quantified scores for all factors and the estimation of a tour is performed based on the summation of those scores in the tour.

The GOP is a generalization of the orienteering problem (OP) and the main difference between the two is that each city in GOP has multiple scores while each city in OP has only one score [14-16].

Table 1. Physical location and score vector for each city

No.	Name	Longitude	Latitude	S_1	S_2	S_3	S_4
1	Beijing	116.40	39.91	8	10	10	7
2	Tianjin	117.18	39.16	6	5	8	8
3	Jinan	117.00	36.67	7	7	5	6
4	Qingdao	120.33	36.06	7	4	5	7
5	Shijiazhuang	114.50	38.05	5	4	5	5
6	Taiyuan	112.58	37.87	5	6	5	5
7	Huhehaote	111.70	40.87	6	6	5	5
8	Zhengzhou	113.60	34.75	5	6	5	5
9	Huangshan	118.29	29.73	9	3	2	2
10	Nanjing	118.75	32.04	7	8	8	6
11	Shanghai	121.45	31.22	5	4	9	9
12	Hangzhou	120.15	30.25	9	8	7	6
13	Nanchang	115.88	28.35	7	6	5	5
14	Fuzhou	119.30	26.10	6	5	5	7
15	Wuhan	114.30	30.55	6	6	8	6
16	Changsha	113.00	28.20	6	6	6	5
17	Guangzhou	113.15	23.15	6	6	5	10
18	Haikou	110.35	20.02	7	3	4	8
19	Guilin	110.29	25.28	10	4	4	4
20	Xi'an	108.92	34.28	5	9	8	6
21	Yinchuan	106.27	38.48	5	7	5	5
22	Lanzhou	103.80	36.03	7	6	5	6
23	Chengdo	104.07	30.66	6	7	6	5
24	Guiyang	106.00	26.59	8	5	4	5
25	Kunming	102.80	25.05	9	7	7	6
26	Shenyang	123.40	41.80	5	8	5	6
27	Dalian	121.60	38.92	7	5	6	7

Let V be the set of N points and E the set of edges between points in V. G = {V, E} is a complete graph. Each edge in E has a symmetric, non-negative cost d(i, j) which becomes the distance or travel time between point i and j. Assume the starting point is point 1 and the end point is point N. Each point i in V has a non-negative score vector $S(i) = (S_1(i), S_2(i), \ldots, S_m(i))^T$, where m is the number of individual goals, and $S_g(i)$ is the score of point i with respect to goal g.

A differentiable objective function that defines total score of a path P, which starts at point 1 and ends at point N can be formulated as follows:

$$Z = \sum_{g=1}^{m} W_g \left[\left\{ \sum_{i \in P} \left[S_g(i) \right]^k \right\}^{1/k} \right]. \quad (7)$$

where W_g is the weight of goal g, and the exponent k is set to 5 in this problem.

Table 1 presents city data such as city number, longitude, latitude, and score vector. S_1, S_2, S_3, and S_4 are the scores approximately scaled from 1 to 10 in the aspects of natural beauty, historical interest, cultural event, and business opportunities, proposed by Wang et al. [9].

4 Computation and Results

For applying HS to GOP, the values of algorithm parameters such as number of music instruments (= number of decision variables), number of improvisations (= number of function evaluations), HMCR, PAR, and HMS are specified.

In GOP, number of music instruments (= 27) is substituted with the number of decision variables that represent every city, and the value of each decision variable represents its next assigned city; Number of improvisations (= 50,000) stands for the number of maximum iterations or objective function evaluations.

HMCR is the rate of choosing any one value from the HM, and thus (1 - HMCR) is the rate of choosing any value from all the possible range of each decision variable. For the computation, various HMCR's are used.

PAR is originally the rate of moving to a neighboring value from one value in HM, but this parameter is modified for GOP. In this computation, PAR becomes the rate of moving to the nearest city from one city. There are total three PAR's (PAR1 = 0.35, PAR2 = 0.105, and PAR3 = 0.045) that are the rates of moving to nearest, second nearest, and third nearest cities, respectively.

HMS is the number of harmony vectors simultaneously stored in HM. For this computation, various HMS's are used.

In this computation, a tour starts from city 1, and next city is assigned based on the following three rules:

Rule 1. Choose any city stored in HM as a next city with probability HMCR × (1-PAR), where PAR = PAR1 + PAR2 + PAR3.

Rule 2. Choose the nearest city as a next city with probability HMCR × PAR1; Or choose the second nearest city with probability HMCR × PAR2; Or choose the third nearest city with probability HMCR × PAR3.

Rule 3. Choose next city randomly with probability (1-HMCR).

Whenever the HS visits new city, the scores of four goals in the city are added using Equation 7 and the distance up to the city is also added using trigonometric formulas on spherical surface (average earth radius, r = 6,371 km) using Equation 8.

$$d(x, y) = r \cdot \arcsin\left(\sin(c_1) \sin\left(\frac{\pi(90 - b_1)}{180} \right) \Big/ \sin(e) \right) \tag{8}$$

where

$$e = \arctan(d_1) + \arctan(d_2)$$

$$d_1 = \cos\frac{c_2}{2} \Big/ \cos\frac{c_3}{2} \tan\frac{c_1}{2}, \quad d_2 = \sin\frac{c_2}{2} \Big/ \sin\frac{c_3}{2} \tan\frac{c_1}{2}$$

$$c_1 = (a_1 - a_2)\frac{\pi}{180}, \quad c_2 = (b_2 - b_1)\frac{\pi}{180}, \quad c_3 = \pi - (b_1 + b_2)\frac{\pi}{180}.$$

where $d(\cdot)$ is a function calculating the distance (in kilometer) between two cities (x and y); a_1 is longitude of city x; b_1 is latitude of city x; a_2 is longitude of city y; and b_2 is latitude of city y.

If the total distance of a tour is over the distance limit (5,000 km in the problem), penalty (the absolute difference between computed distance and limit distance) is also taxed to the original summarized score.

There are five different weight vectors including W_0 = (0.25, 0.25, 0.25, 0.25), W_1 = (1, 0, 0, 0), W_2 = (0, 1, 0, 0), W_3 = (0, 0, 1, 0), and W_4 = (0, 0, 0, 1). The first weight gives equal weight to each of the four goals. The four other weight vectors stress one goal and ignore the other three. And, each weight case runs 45 times with different HMS's and HMCR's.

Table 2. Comparison of GOP results from HS and ANN

Weight	Method	Score	Distance	Tour
W_0	HS	12.38	4993.4	1-2-3-10-11-12-9-13-17-19-16-20-6-5-1
	ANN	12.38	4993.4	1-2-3-10-11-12-9-13-17-19-16-20-6-5-1
W_1	HS	13.08	4985.4	1-2-3-15-24-19-13-9-12-10-4-27-1
	ANN	13.05	4987.7	1-2-3-4-10-11-12-9-13-16-19-24-20-6-5-1
W_2	HS	12.56	4910.6	1-26-27-4-10-12-9-13-16-15-20-8-3-2-1
	ANN	12.51	4875.1	1-2-26-27-3-10-11-12-9-13-15-20-6-5-1
W_3	HS	12.78	4987.5	1-2-3-5-6-20-8-15-16-13-9-12-11-10-4-27-1
	ANN	12.78	4987.5	1-2-3-5-6-20-8-15-16-13-9-12-11-10-4-27-1
W_4	HS	12.40	4845.2	1-2-27-4-10-11-12-14-17-16-15-3-1
	ANN	12.36	4989.8	1-2-3-10-9-13-16-17-14-12-11-4-27-1

Table 2 represents the best tours in five different weight vector cases and compares them with the tours obtained using artificial neural network (ANN) approach [9]. Compared to the results of ANN, HS could find better score solutions in cases of W_1, W_2, and W_4 while find same score solutions in cases of W_0 and W_3: With the weight vector W_0, HS found 12.38 as the best score with the score range between 11.95 ~ 12.38; with the weight vector W_1, HS found 13.08 as the best score with the score range between 12.58 ~ 13.08; with the weight vector W_2, HS found 12.56 as the best score with the score range between 12.34 ~ 12.56; with the weight vector W_3, HS found 12.78 as the best score with the score range between 12.50 ~ 12.78; with the weight vector W_4, HS found 12.40 as the best score with the score range between 12.14 ~ 12.40.

5 Conclusions

In this study, a recently-developed nature-inspired algorithm, HS, has been introduced and applied to an NP-hard GOP whose objective is to find the best tour in eastern part of China. The algorithm, HS, mimics three major behaviors of music players: 1) memory consideration; 2) pitch adjustment; and 3) random choice. These behaviors can be successfully translated in GOP: 'memory consideration' becomes that HS chooses any one city from the cities stored in HM; 'pitch adjustment' is that HS chooses the nearest city as next city; and 'random choice' is that HS chooses any one city from all the possible cities.

After applied to GOP, HS could find equal or better solutions when compared with those of ANN. In order for HS to obtain better results in GOP in the future, some additional operators especially for GOP might be implemented along with existing memory consideration and pitch adjustment operators. Also, it is expected that HS, as a nature-inspired algorithm, can be applied to other optimization problems in various fields.

References

1. Fogel, L. J., Owens, A. J. and Walsh. M. J.: Artificial Intelligence Though Simulated Evolution. John Wiley, Chichester, UK (1966)
2. De Jong, K.: Analysis of the Behavior of a Class of Genetic Adaptive Systems. Ph.D. Thesis, University of Michigan, Ann Arbor, MI, USA (1975)
3. Koza, J. R.: Genetic Programming: A Paradigm for Genetically Breeding Populations of Computer Programs to Solve Problems. Report No. STA-CS-90-1314, Stanford University, Stanford, CA, USA (1990)
4. Holland, J. H.: Adaptation in Natural and Artificial Systems. University of Michigan Press, Ann Arbor, MI, USA, (1975)
5. Goldberg, D. E.: Genetic Algorithms in Search Optimization and Machine Learning. Addison Wesley, MA, USA (1989)

6. Glover, F.: Heuristic for Integer Programming using Surrogate Constraints. Decision Sciences. 8(1) (1977) 156-166
7. Dorigo, M., Maniezzo, V., and Colorni, A.: The Ant System: Optimization by a Colony of Cooperating Agents. IEEE Transactions on Systems, Man, and Cybernetics-Part B. 26(1) (1996) 29-41
8. Kirkpatrick, S., Gelatt, C., and Vecchi, M.: Optimization by Simulated Annealing. Science. 220(4598) (1983) 671-680
9. Wang, Q., Sun, C., and Golden, B. L.: Using Artificial Neural Networks to Solve Generalized Orienteering Problems. Proceedings of Artificial Neural Networks in Engineering Conference (ANNIE '96). (1996)
10. Geem, Z. W., Kim, J. H., and Loganathan, G. V.: A New Heuristic Optimization Algorithm: Harmony Search. Simulation. 76(2) (2001) 60-68
11. Kim, J. H., Geem, Z. W., and Kim, E. S.: Parameter Estimation of the Nonlinear Muskingum Model using Harmony Search. Journal of the American Water Resources Association. 37(5) (2001) 1131-1138
12. Geem, Z. W., Kim, J. H., and Loganathan. G. V.: Harmony Search Optimization: Application to Pipe Network Design. International Journal of Modelling and Simulation. 22(2) (2002) 125-133
13. Kang, S. L., and Geem, Z. W.: A New Structural Optimization Method Based on the Harmony Search Algorithm. Computers and Structures. 82(9-10) (2004) 781-798
14. Chao, I. -M., Golden, B. L., and Wasil, E. A.: The Team Orienteering Problem. European Journal of Operational Research. 88 (1996) 464-474
15. Chao, I. -M., Golden, B. L., and Wasil, E. A.: A Fast and Effective Heuristic for the Orienteering Problem. European Journal of Operational Research. 88 (1996) 475-489
16. Tasgetiren, M. F., and Smith, A. E.: A Genetic Algorithm for the Orienteering Problem. Proceedings of Congress on Evolutionary Computation 2000 (CEC 2000). (2000), 1190-1195.