An Adaptive Harmony Search Algorithm with Zipf Distribution

Shih-Pang Tseng and Jaw-Shyang Wu

Department of Computer Science and Entertainment Technology,
Tajen University, Pingtung, Taiwan
{tsp,jswu}@tajen.edu.tw

Abstract. Harmony search (HS) can be applied to various optimization problems and easy to implement. In this paper, we try to improve HS by change the reference probability distribution of harmony memory. Zipf distribution is used to balance the intensification and diversification. In addition, we propose the adaptive mechanism to avoid setting the new parameter. Experimental results show that the improvement is effective on the high dimensional numerical function optimization problem.

1 Introduction

Optimization trys to find the best solution within an allowed solutions set to achieve some objectives for many theoretical and practical problems. These optimization problems can be categorized into two kinds: continuous and discrete. In the continuous optimization problem, the solutions can be encoded with real-valued variables. Otherwise, the solutions can be encoded with the discrete variables in the discrete optimization problems. Whether the continuous or discrete ones, The allowed solutions set, usually denoted by *search space* contains a great number of possible solutions, it is not practical to explore the whole search space for many problems. For these optimization, to find the optimal solution in the search space is impractical by an affordable computing cost. Metaheuristics [1] are developed for these kinds of optimization problems. A metaheuristic is an approximate algorithm to explore the search space as efficiently and effectively as possible. A metaheuristic try to find the sub-optimal, or near-optimal solution by an affordable computing cost. In the past 30 years, there were many metaheuristics proposed, such as genetic algorithms [2], ant system[3], and so on. In the first ten years of 21 century, a new metaheuristic, Harmony search (HS) [4], is proposed and successfully applied in theoretical and physical domains, such as solving Sudoku [5], cell-phone network [6], Ground Water Modeling [7] clustering web documents [8], and so on. Harmony search is inspired by the process of music to search for a perfect state of harmony.

In HS, the solution of this problem is call by the *harmony*. The harmony memory(HM) is used to store a set of harmonies. HS constructs new harmony by referring these harmonies in harmony memory. Only one new harmony (solution) in one iteration of HS. If the new harmony is better than the worst one in the HM, the new one would be inserted into the HM to replace the worst one.

J.-S. Pan et al. (eds.), *Intelligent Data Analysis and Its Applications, Volume 2,*
Advances in Intelligent Systems and Computing 298,
DOI: 10.1007/978-3-319-07773-4_16,

A harmony is coded to a vector of attributes in HS. In constructing new harmony, for each attribute of harmony, there are three possible choices to decide its value, as following:

1. **Usage of harmony memory**: pick the attribute value from the harmony selected randomly in HM.
2. **Pitch adjusting**: like the above, but add a little change to the attribute value.
3. **Randomization**: randomly assign a value to the attribute.

The usage of harmony memory ensures that good harmonies are considered as elements of new harmony. In order to use HM effectively, a parameter $r_{accept} \in [0, 1]$, called harmony memory considering (or accepting) rate, is used to define the probability of usage of harmony memory in constructing a new harmony. The pitch adjusting makes a little change to the attribute's value from harmony memory. The parameter $r_{pa} \in [0, 1]$ is called pitch rate which defines the probability of pitch adjusting after the usage of harmony memory. This try to tune the value to find a better solution. The randomization increases the diversification of the solutions to avoid premature optimization.

1.1 Motivation

The balance between intensification and diversification [1] is the most important to all metaheuristics. In general, the diversification can prevent the premature optimization and get better final solution. Moreover, the diversification usually reduces the converging speed and needs more computation time. The randomization of HS can be used to increase the diversification. However, the probability of randomization, $1 - r_{accept}$, is a sensible parameter of HS. It should be small enough to ensure the convergence. For this reason, it is a better way to increase the size of harmony memory, and another way should be involved to enhance the intensification.

So we can increase the size of harmony memory to handle the diversity of HS. And we do not consider the intensification factor to the size of harmony memory. It makes the size of harmony memory not a sensitive parameter. Thus we can use a larger size of harmony memory to keep more solutions in the process of harmony search. In original HS, all harmonies in harmony memory are with the same reference probability in constructing a new harmony. The distribution of reference probabilities is an uniform distribution. In this paper, we redesign the structure of harmony memory. And the distribution of reference probabilities is changed to Zipf distribution. Consequently, the better harmony is with more probability, and the worse harmony is with less probability. In addition, the difference of reference probabilities between better and worse harmonies can be clearly control by a single parameter z.

It is not easy to decide the algorithm parameters values of the metaheuristics.[9] For the balance between intensification and diversification, the Zipf distribution and one new parameter z are involved in HS. In our previous work, ZHS [10], we

applied various z on different numeric optimization functions. It shows that ZHS is beter than HS on the balance between intensification and diversification. However, it is still a shortcoming to find a suitable value of z for different functions in ZHS. It is the best way to set the value of z adaptively. It means that the z would be encoded as one part of the harmony, and the value of z should be adjusted for different functions.

In this paper, a Adaptive Zipf harmony search (AZHS) is proposed to more effectively and efficiently solve the numerical function optimization problem. The remainder of the paper is organized as follows. Section 2 briefly describe the Zipf distribution. Section 3 provides a detailed description of AZHS we proposed. Performance evaluation of the AZHS is presented in Section 4. Conclusion is given in Section 5.

2 Zipf Distribution

Zipf distribution is derived by the Zipf's law which originally states the skewness of natural language [11][12]. George Kingsley Zipf proposed the fact that the frequency of one word is proportional to the power inverse of its rank. The similar relationship is discovered in some economic domains, such as the income distribution. In recent years, the Zipf distribution is used to describe the assess behavior of Internet, such as the visits to web sites[13]. In this work, the Zipf distribution is used to control the reference behavior to harmony memory, and to balance between intensification and diversification.

Zipf distribution is a discrete probability distribution. N denotes the number of elements. k_i is the rank of element i, k_i is an integer and $1 \leq k_i \leq N$. The important parameter of Zipf distribution is z which can adjust the inequality. z is a real number and $z \geq 0$. The probability of the element i, $P(i)$, is proportional to the z power inverse of its rank, k_i.

$$P(i) \propto \frac{1}{{k_i}^z} \tag{1}$$

$$P(i) = \frac{{k_i}^{-z}}{\sum_{j=1}^{N} {k_j}^{-z}} \tag{2}$$

Figure 1 shows the probabilities of Zipf distribution with different z values. When $z = 0$, the Zipf distribution is equivalent to the uniform distribution in this special case. When z is larger, the difference between the first one and the last one is larger.

3 Proposed Method

In traditional HS, harmony memory is only a set of harmonies. All harmonies in harmony memory are with the same reference probability in constructing a new

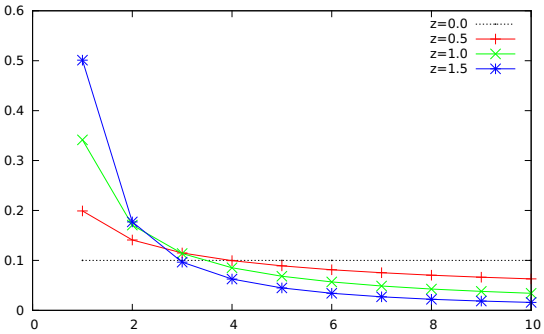

Fig. 1. Zipf distribution with different z values

harmony. On the other hand, the harmony memory is organized to a sorted list in AZHS. The best harmony is in the head of sorted list, and the worst one is in the tail. The main difference between ZHS and traditional HS is the reference probability distribution to the harmonies in harmony memory. The reference probabilities of harmonies in harmony memory is related to their position, rank, in this sorted list. The reference probability of each harmony is proportional to the z power inverse of its rank. When a new harmony has been constructed, it would be inserted into the appropriate position in the sorted list if it is better than the worst one. And then the worst harmony should be removed from the harmony memory. HS can be considered as a special case of AZHS, when z is always zero. Figure 2 shows the structure of Zipf harmony memory which size is 10.

h_1 $p_1 = 0.34$	h_2 $p_2 = 0.17$	h_3 $p_3 = 0.11$	h_4 $p_4 = 0.085$	h_5 $p_5 = 0.068$	h_6 $p_6 = 0.056$	h_7 $p_7 = 0.048$	h_8 $p_8 = 0.043$	h_9 $p_9 = 0.037$	h_{10} $p_{10} = 0.034$

Fig. 2. An example of Zipf Harmony memory, $size = 10$

Algorithm 1 shows the detail of AZHS. In the beginning to construct a new harmony, a new z would be assigned for the following steps. The assignment of z is similar to the assignment of attribute values in HS. There are three ways to assign z value.

1. **Usage of harmony memory**: pick the z value from the harmony selected randomly in HM.
2. **Pitch adjusting**: like the above, but add a little change to the z value.
3. **Randomization**: randomly assign a value to z.

In this step, the reference probability distribution of HM is uniform, due to the z is not assigned. After thr z is assigned, the attributes of new harmony would be assigned in sequence according the z value. If the z value is larger, the constructing process would prefer the better harmonies in HM. If the z value is smaller, the reference probability distribution of HM is more uniform in the constructing process.

Algorithm 1. Adaptive Zipf Harmony Search

```
Initiate Harmony Memory
for i = 1 to max number of iterations do
  if rand() < r_accept then
    Choose a z value from Harmony Memory
    if rand() < r_pa then
      Adjust the z value
    end if
  else
    Assign a random z value
  end if
  for j = 1 to number of dimensions do
    if rand() < r_accept then
      Choose a value from Harmony Memory for this attribute
      if rand() < r_pa then
        Adjust the value
      end if
    else
      Assign a random value to the attribute
    end if
  end for
  Insert new harmony into Harmony Memory
end for
Return the best harmony
```

4 Experiment Result

In this section, we evaluate the performance of the proposed algorithm, AZHS and compare with original HS. The empirical analysis was conducted on an virtual machine in CHT Hicloud(http://hicloud.hinet.net/) with 4 CPUs and 8GB of memory using CentOS 6.4 running Linux kernel 2.6.32. Moreover, all the programs are written in C++ and compiled using g++ (GNU C++ compiler).

In this research, we choose five benchmarks of numerical function optimization, F1 $\sim$ F5. These benchmark functions are well-known and used in many researches. The function F1 is the De Jongs function 1, sphere function. Next, the function F2 is the axis parallel hyper-ellipsoid function. The function F3 is the rotated hyper-ellipsoid function The function F4 is the De Jongs function 2, Rosenbrocks valley. The Rastrigins function is the function F5. The F1, F2

and F3 are unimodal; and the F4 and F5 are multimodal. The formulae and ranges of benchmark functions are shown in Table 1. All these benchmarks are the minimization problems.

In our experiments, all the simulations are carried out for 30 runs. The r_{accept} is always set to 0.9 . The probability of pitch adjusting, r_{pa}, is 0.3 . The Zipf distribution parameter z is varied from 0 to 10.0. We choose two different harmony memory sizes, 50 and 100.

Table 1. Benchmarks for Numerical Function Optimization

	Functions	Range
F1	$f_1(\mathbf{x}) = \sum_{i=1}^{n} {x_i}^2$	$-100 \leq x_i \leq 100$
F2	$f_2(\mathbf{x}) = \sum_{i=1}^{n} i{x_i}^2$	$-5.12 \leq x_i \leq 5.12$
F3	$f_3(\mathbf{x}) = \sum_{i=1}^{n} \sum_{j=1}^{i} {x_j}^2$	$-65.536 \leq x_i \leq 65.536$
F4	$f_4(\mathbf{x}) = \sum_{i=1}^{n-1} 100({x_{i+1}} - {x_i}^2)^2 + (x_i - 1)^2$	$-2.048 \leq x_i \leq 2.048$
F5	$f_5(\mathbf{x}) = 10n + \sum_{i=1}^{n} {x_i}^2 - 10\cos 2\pi x_i$	$-100 \leq x_i \leq 100$

To simplify the discussion of the simulation results, we use the following conventions. Let v denote either the value of final solution, Δ_v the enhancement of v_ϕ with respect to v_ψ

in percentage. In other words, Δ_v is defined as follows:

$$\Delta_v = \frac{v_\phi - v_\psi}{v_\psi} \times 100\% \tag{3}$$

where v is either the value of final solution for our experiments on numerical function optimization problems. The v_ϕ and v_ψ are the result of two different algorithms. For example, the HS and the AZHS

Table 2 shows the comparison of HS and AZHS when the dimensions of benchmarks is set to 50, and the evaluations is set to 100000. In general, the AZHS is apparently better than the HS, except one. Because all benchmark are minimization problem, the negative Δ_v means the AZHS is better. The z^* denotes the average z value of the best harmony in different tries. Table 3 shows the experimental result when the dimensions of benchmarks is set to 50, and the evaluations is set to 100000. In general, the AZHS is significantly better than the HS in all cases.

Table 2. Experimental results of 30 dimensions with 100000 evaluations

	HM	HS	AZHS	Δ_v	z^*
F1	50	0.001039	0.000599	-42	5.91
F2		0.0164	0.0104	-37	5.92
F3		2.53	1.66	-35	5.97
F4		56.0	35.1	-37	6.90
F5		0.277	0.208	-25	6.03
F1	100	0.00417	0.00057	-86	6.28
F2		0.0562	0.0098	-82	5.92
F3		9.51	1.59	-83	5.19
F4		41.5	46.9	13	5.64
F5		0.562	0.135	-76	6.12

Table 3. Experimental results of 60 dimensions with 500000 evaluations

	HM	HS	AZHS	Δ_v	z^*
F1	50	0.549	0.016	-97	6.17
F2		11.3	0.576	-94	6.12
F3		1956.2	91.6	-95	6.70
F4		152.6	134.9	-11	6.04
F5		20.5	14.0	-31	5.81
F1	100	0.925	0.017	-98	5.85
F2		19.080	0.547	-97	5.53
F3		3068.4	85.0	-97	5.66
F4		136.5	134.8	-1	5.42
F5		21.8	15.0	-31	6.24

5 Conclusion

Harmony search (HS) is a relatively new metaheuristic in the past 10 years. We proposed the Adaptive Zipf harmony search (AZHS) which used the Zipf distribution to be the reference probability distribution of harmony memory. In addition, we involve the adaptive mechanism to set the z parameter automatically. It makes the better balance between intensification and diversification. We applied the AZHS on the numerical function optimization problem. In elementary, the experimental result shows that AZHS is a significantly enhancement of HS.

Acknowledgment. The authors would also like to thank the Chunghwa Telecom and Networked Communications Program, Taiwan for the support to this paper.

References

1. Blum, C., Roli, A.: Metaheuristics in combinatorial optimization: Overview and conceptual comparison. ACM Comput. Surv. 35(3), 268–308 (2003)
2. Goldberg, D.E.: Genetic Algorithms in Search, Optimization and Machine Learning. Addison-Wesley Longman Publishing Co., Inc. (1989)
3. Dorigo, M., Maniezzo, V., Colorni, A.: The ant system: Optimization by a colony of cooperating agents. IEEE Transactions on Systems, Man, and Cybernetics-Part B 26, 29–41 (1996)
4. Geem, Z.W., Kim, J.H., Loganathan, G.V.: A new heuristic optimization algorithm: Harmony search. Simulation 76(2), 60–68 (2001)
5. Geem, Z.W.: Harmony search algorithm for solving sudoku. In: Apolloni, B., Howlett, R.J., Jain, L. (eds.) KES 2007, Part I. LNCS (LNAI), vol. 4692, pp. 371–378. Springer, Heidelberg (2007)
6. Zhang, R., Hanzo, L.: Iterative multiuser detection and channel decoding for ds-cdma using harmony search. Signal Processing Letters 16, 917–920 (2009)
7. Ayvaz, M.T.: Simultaneous determination of aquifer parameters and zone structures with fuzzy c-means clustering and meta-heuristic harmony search algorithm. Advances in Water Resources 30, 2326–2338 (2007)
8. Mahdavi, M., Chehreghani, M.H., Abolhassani, H., Forsati, R.: Novel meta-heuristic algorithms for clustering web documents. Applied Mathematics and Computation 201(1-2), 441–451 (2008)
9. Geem, Z.W., Sim, K.B.: Parameter-setting-free harmony search algorithm. Applied Mathematics and Computation 217(8), 3881–3889 (2010)
10. Tseng, S.P., Lin, W.W.: Improving harmony search by zipf distribution. In: 2012 Sixth International Conference on Genetic and Evolutionary Computing (ICGEC), pp. 115–118 (August 2012)
11. Zipf, G.K.: Selected Studies of the Principle of Relative Frequency in Language. Harvard University Press (1932)
12. Zipf, G.K.: Human Behavior and the Principle of Least Effort. Addison-Wesley, Reading (1949)
13. Adamic, L.A., Huberman, B.A.: Zipf's law and the Internet. Glottometrics 3, 143–150 (2002)