State-of-the-Art in the Structure of Harmony Search Algorithm

Zong Woo Geem[1]

Abstract The harmony search algorithm has been so far applied to various optimization problems. Also, the algorithm structure has been customized on a case-by-case basis by tweaking the basic structure. The objective of this chapter is to introduce the state-of-the-art structure of the basic harmony search algorithm.

1 Introduction

For optimization, people have traditionally used calculus-based algorithms that give gradient information in order to find the right direction to the optimal solution. However, if variables are discrete instead of continuous, they cannot have derivatives. To overcome this situation, the harmony search (HS) algorithm has used a novel stochastic derivative [1] which utilizes the experiences of musicians in Jazz improvisation and can be applicable to discrete variables. Instead of the inclination information of an objective function, the stochastic derivative of HS gives a probability to be selected for each value of a decision variable. For example, if the decision variable x_1 has three candidate values {1, 2, 3}, the partial stochastic derivative of the objective function with respect to x_1 at each discrete value gives the selection probability for each value like 20% for 1; 30% for 2; and 50% for 3. While cumulative probability becomes unity (100%), the probability for each value is updated iteration by iteration. Desirably the value, which is included in the optimal solution vector, has higher chance to be chosen as the iterations progress.

With this stochastic derivative information, the HS algorithm has been applied to various science and engineering optimization problems that include [2, 3]:

Real-world applications
- Music composition
- Sudoku puzzle

[1] Environmental Planning and Management Program, Johns Hopkins University, Baltimore, Maryland, USA, Email: geem@jhu.edu

- Timetabling
- Tour planning
- Logistics

Computer science problems

- Web page clustering
- Text summarization
- Internet routing
- Visual tracking
- Robotics

Electrical engineering problems

- Energy system dispatch
- Photo-electronic detection
- Power system design
- Multi-level inverter optimization
- Cell phone network

Civil engineering problems

- Structural design
- Water network design
- Dam scheduling
- Flood model calibration
- Groundwater management
- Soil stability analysis
- Ecological conservation
- Vehicle routing

Mechanical engineering problems

- Heat exchanger design
- Satellite heat pipe design
- Offshore structure mooring

Bio & medical applications

- RNA structure prediction
- Hearing aids
- Medical physics

In addition to the above-mentioned various applications, the HS algorithm also has various algorithm structures that can be applicable to so many different problems. Thus, this chapter arranges the basic structure of the HS algorithm so that users can easily customize the algorithm for their own optimization problems.

2 Basic Structure of Harmony Search Algorithm

The HS algorithm was originally inspired by the improvisation process of Jazz musicians. Figure 1 shows the analogy between improvisation and optimization: Each musician corresponds to each decision variable; musical instrument's pitch range corresponds to decision variable's value range; musical harmony at certain time corresponds to solution vector at certain iteration; and audience's aesthetics corresponds to objective function. Just like musical harmony is improved time after time, solution vector is improved iteration by iteration.

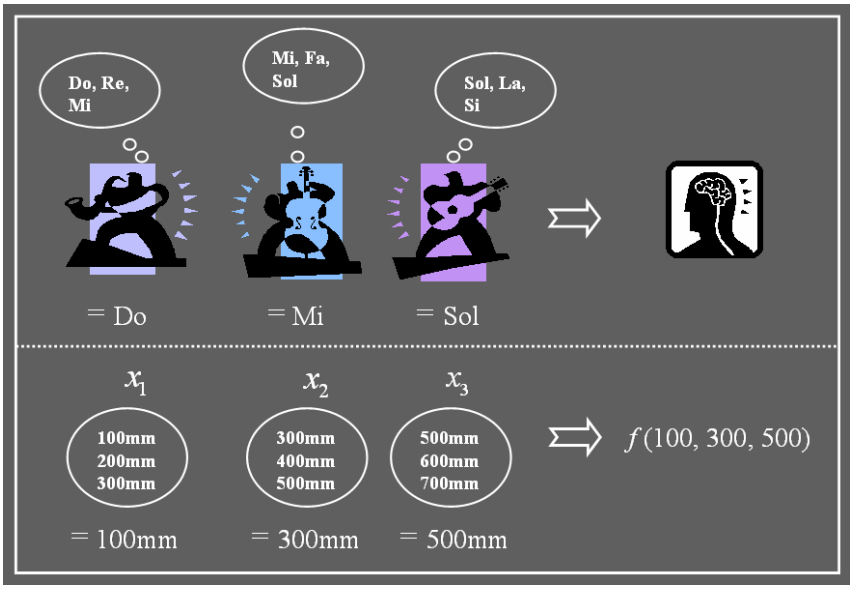

Fig. 1 Analogy between Improvisation and Optimization

This section introduces each step of the HS algorithm in detail, including 1) problem formulation, 2) algorithm parameter setting, 3) random tuning for memory initialization, 4) harmony improvisation (random selection, memory consideration, and pitch adjustment), 5) memory update, 6) performing termination, and 7) cadenza.

2.1 Problem Formulation

The HS algorithm was devised for solving optimization problems. Thus, in order to apply HS, problems should be formulated in the optimization environment, having objective function and constraints:

$$\text{Optimize (minimize or maximize)} \quad f(\mathbf{x}) \tag{1}$$

Subject to

$$h_i(\mathbf{x}) = 0; \quad i = 1, \ldots, p; \tag{2}$$

$$g_i(\mathbf{x}) \geq 0; \quad i = 1, \ldots, q. \tag{3}$$

$$x_i \in \mathbf{X}_i = \{x_i(1), \ldots, x_i(k), \ldots, x_i(K_i)\} \text{ or } x_i^L \leq x_i \leq x_i^U \tag{4}$$

The HS algorithm searches entire solution area in order to find the optimal solution vector $\mathbf{x} = (x_1, \ldots, x_n)$, which optimizes (minimizes or maximizes) the objective function as in Equation 1. If the problem has equality and/or inequality conditions, these can be considered as constraints in Equations 2 and 3. If the decision variable has discrete values, the set of candidate values for the variable becomes $x_i \in \mathbf{X}_i = \{x_i(1), \ldots, x_i(k), \ldots, x_i(K_i)\}$; and if the decision variable has continuous values, the set of candidate values for the variable becomes $x_i^L \leq x_i \leq x_i^U$.

The HS algorithm basically considers the objective function only. However, if a solution vector generated violates any of the constraints, 1) the algorithm abandons the vector or 2) considers it by adding certain amount of penalty to the objective function value. Also, HS can be applied to multi-objective problems by conjugating with Pareto set.

2.2 Algorithm Parameter Setting

Once the problem formulation is ready, algorithm parameters should be set with certain values. HS contains algorithm parameters including harmony memory size (HMS), harmony memory considering rate (HMCR), pitch adjusting rate (PAR), maximum improvisation (MI), and fret width (FW).

HMS is the number of solution vectors simultaneously handled in the algorithm; HMCR is the rate ($0 \leq \text{HMCR} \leq 1$) where HS picks one value randomly from musician's memory. Thus, (1-HMCR) is the rate where HS picks one value randomly from total value range; PAR ($0 \leq \text{RAR} \leq 1$) is the rate where HS tweaks the value which was originally picked from memory. Thus, (1-PAR) is the rate where HS keeps the original value obtained from memory; MI is the number of iterations. HS improvises one harmony (= vector) each iteration; and FW is arbitrary length only for continuous variable, which was formerly called as bandwidth (BW). For more information of the term, a fret is the metallic ridge on the neck of

a string instrument (such as guitar), which divides the neck into fixed segments (see Figure 2), and each fret represents one semitone. In the context of the HS algorithm, frets mean arbitrary points which divide the total value range into fixed segments, and fret width (FW) is the length between two neighboring frets. Uniform FW is normally used in HS.

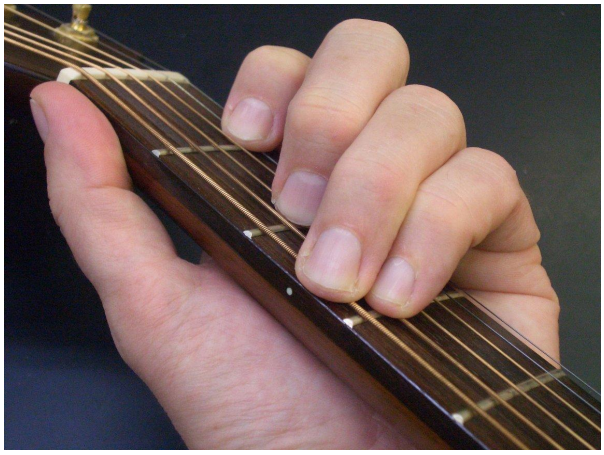

Fig. 2 Frets on the Neck of a Guitar

Originally fixed parameter values were used. However, some researchers have proposed changeable parameter values. Mahdavi et al. [4] suggested that PAR increase linearly and FW decrease exponentially with iterations:

$$PAR(I) = PAR_{min} + \left(PAR_{max} - PAR_{min}\right) \times \frac{I}{MI} \tag{5}$$

$$FW(I) = FW_{max} \exp\left[\ln\left(\frac{FW_{min}}{FW_{max}}\right) \frac{I}{MI} \right] \tag{6}$$

Mukhopadhyay et al. [5] suggested that FW be the standard deviation of the current population when HMCR is close to 1.

$$FW(I) = \sigma(\mathbf{x}_i) = \sqrt{\text{var}(\mathbf{x}_i)} \tag{7}$$

Geem [6] tabulated fixed parameter values, such as number of variables, HMS, HMCR, PAR, and MI, after surveying various literatures. FW normally ranges from 1% to 10% of total value range.

Furthermore, some researchers have proposed adaptive parameter theories that enable HS to automatically have best parameter values at each iteration [3, 7].

2.3 Random Tuning for Memory Initialization

After problem is formulated and the parameter values were set properly, random tuning process is performed.

In an orchestra concert, after oboe plays the note A (usually A440), other instruments randomly play any pitches out of playable ranges. Likewise, the HS algorithm initially improvises many random harmonies. The number of random harmonies should be at least HMS. However, the number can be more than HMS, such as twice or three times as many as HMS [8]. Then, top-HMS harmonies are selected as starting vectors.

Musician's harmony memory (HM) can be considered as a matrix:

$$\mathbf{HM} = \left[\begin{array}{cccc|c} x_1^1 & x_2^1 & \cdots & x_n^1 & f(\mathbf{x}^1) \\ x_1^2 & x_2^2 & \cdots & x_n^2 & f(\mathbf{x}^2) \\ \vdots & \cdots & \cdots & \cdots & \vdots \\ x_1^{HMS} & x_2^{HMS} & \cdots & x_n^{HMS} & f(\mathbf{x}^{HMS}) \end{array}\right] \tag{8}$$

Previously, the objective function values were sorted ($f(\mathbf{x}^1) \leq f(\mathbf{x}^2) \leq \ldots \leq f(\mathbf{x}^{HMS})$) in HM, but current structure does not require it any more.

2.4 Harmony Improvization

In Jazz improvisation, a musician plays a note by randomly selecting it from total playable range (see Figure 3), from musician's memory (see Figure 4), or by tweaking the note obtained from musician's memory (see Figure 5). Likewise, the HS algorithm improvises a value by choosing it from total value range or from HM, or tweaking the value which was originally chosen from HM.

Fig. 3 Total Playable Range of a Music Instrument

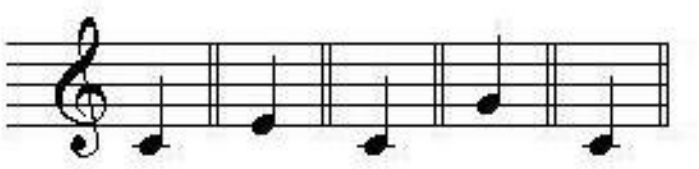

Fig. 4 Set of Good Notes in Musician's Memory

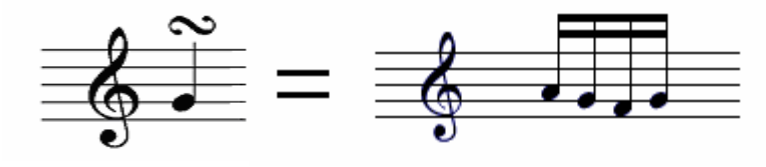

Fig. 5 Tweaking the Note Chosen from Musician's Memory

Random Selection: When HS determines the value x_i^{New} for the new harmony $\mathbf{x}^{New} = (x_1^{New}, \ldots, x_n^{New})$, it randomly picks any value from total value range ($\{x_i(1), \ldots, x_i(K_i)\}$ or $x_i^L \le x_i \le x_i^U$) with probability of (1-HMCR). Random selection is also used for previous memory initialization.

Memory Consideration: When HS determines the value x_i^{New}, it randomly picks any value x_i^j from HM = $\{x_i^1, \ldots, x_i^{HMS}\}$ with probability of HMCR. The index j can be calculated using uniform distribution $U(0,1)$:

$$j \leftarrow \text{int}(U(0,1) \cdot HMS) + 1 \tag{9}$$

However, we may use different distributions. For example, if we use $[U(0,1)]^2$, HS chooses lower j more. If the objective function values are sorted by j, HS will behave similar to particle swarm algorithm.

Pitch Adjustment: After the value x_i^{New} is randomly picked from HM in the above memory consideration process, it can be further adjusted into neighbouring values by adding certain amount to the value, with probability of PAR. For discrete variable, if $x_i(k) = x_i^{New}$, the pitch-adjusted value becomes $x_i(k+m)$ where $m \in \{-1, 1\}$ normally; and for continuous variable, the pitch-adjusted value becomes $x_i^{New} + \Delta$ where $\Delta = U(0,1) \cdot FW(i)$ normally.

The above-mentioned three basic operations (random selection, memory consideration and pitch adjustment) can be expressed as follows:

$$x_i^{New} \leftarrow \begin{cases} \begin{cases} x_i \in \{x_i(1),...,x_i(k),...,x_i(K_i)\} \\ x_i \in [x_i^{Lower}, x_i^{Upper}] \end{cases} & \text{w.p.} \quad (1-HMCR) \\ x_i \in \mathbf{HM} = \{x_i^1, x_i^2, ..., x_i^{HMS}\} & \text{w.p.} \quad HMCR \cdot (1-PAR) \\ \begin{cases} x_i(k+m) \text{ if } x_i(k) \in \mathbf{HM} \\ x_i + \Delta \text{ if } x_i \in \mathbf{HM} \end{cases} & \text{w.p.} \quad HMCR \cdot PAR \end{cases} \tag{10}$$

Especially for discrete variables, the HS algorithm has the following stochastic partial derivative which consists of three terms such as random selection, memory consideration and pitch adjustment [1]:

$$\frac{\partial f}{\partial x_i} = \frac{1}{K_i}(1-HMCR) + \frac{n(x_i(k))}{HMS} HMCR(1-PAR) + \frac{n(x_i(k-m))}{HMS} HMCR\,PAR \tag{11}$$

Also, the HS algorithm can consider the relationship among decision variables using ensemble consideration just as there exists stronger relationship among specific musicians (see Figure 6). The value x_i^{New} can be determined based on x_j^{New} if the two has the strongest relationship [9]:

$$x_i^{New} \leftarrow fn(x_j^{New}) \quad \text{where} \quad \max_{i \neq j} \left\{ \left| Corr(\mathbf{x}_i, \mathbf{x}_j) \right|^2 \right\} \tag{12}$$

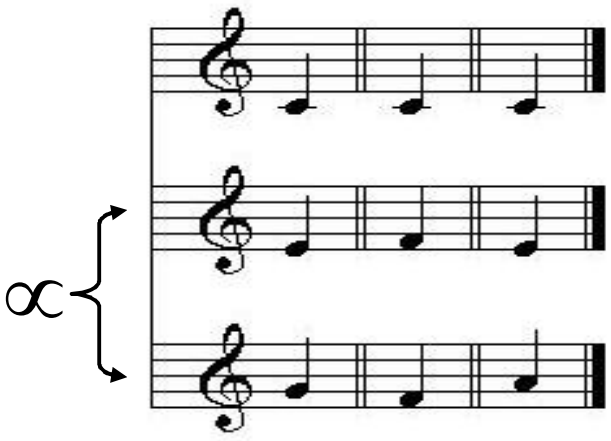

Fig. 6 Relationship between Specific Musicians

If the newly improvised harmony $\mathbf{x}^{New}$ violates any constraint, HS abandons it or still keeps it by adding penalty to the objective function value just like musicians sometimes still accept rule-violated harmony (see Figure 7).

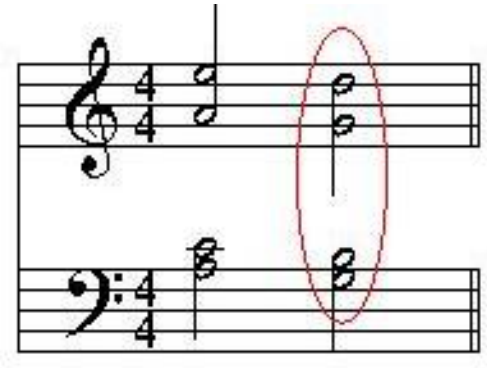

Fig. 7 Rule-Violated Harmony (Parallel Fifth)

2.5 Memory Update

If the new harmony $\mathbf{x}^{New}$ is better, in terms of objective function value, than the worst harmony in HM, the new harmony is included in HM and the worst harmony is excluded from HM:

$$\mathbf{x}^{New} \in \mathbf{HM} \quad \wedge \quad \mathbf{x}^{Worst} \notin \mathbf{HM} \tag{13}$$

However, for the diversity of harmonies in HM, other harmonies (in terms of least-similarity) can be considered. Also, maximum number of identical harmonies in HM can be considered in order to prevent premature HM.

If the new harmony $\mathbf{x}^{New}$ is the best one when compared with every harmony in HM, the new harmony can consider an additional process named *accidentaling*. In music, an accidental is a note whose pitch is not a member of a scale and the accidental sign raises (#) or lowers (b) the following note from its normal pitch as shown in Figure 8. Likewise, HS can further pitch-adjust every note of the new harmony if it is the ever-best harmony, which may find an even better solution:

$$x_i^{New} \leftarrow \begin{cases} x_i(k \pm m) \text{ for discrete var.} \\ x_i \pm \Delta \text{ for continuous var.} \end{cases}, i = 1, \ldots, n \tag{14}$$

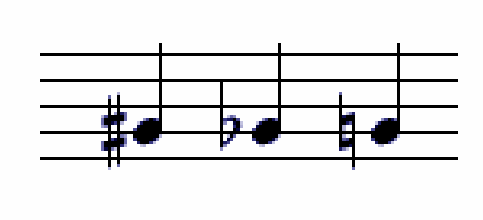

Fig. 8 Accidental for the Note Sol

2.6 Performing Termination

If HS satisfies termination criteria (for example, reaching MI), the computation is terminated. Otherwise, HS improvises another new harmony again.

2.7 Cadenza

Cadenza is a musical passage occurring at the end of a movement. In the context of the HS algorithm, cadenza can be referred to a process occurring at the end of the HS computing. In this process, HS returns the best harmony ever found and stored in HM.

6 Conclusions

This chapter arranged the up-to-date structure of the HS algorithm. Those, who are interested in applying the algorithm to their own optimization problems, may customize the structure into their problems.

The HS algorithm is still growing. The author hopes other researchers to suggest new ideas to make better shape of the algorithm structure.

References

1. Geem ZW (2008) Novel derivative of harmony search algorithm for discrete design variables. Applied Mathematics and Computation 199:223–230
2. Geem ZW (2009) Music-inspired harmony search algorithm: theory and applications. Springer, Berlin

3. Geem ZW (2009) Harmony search algorithms for structural design optimization. Springer, Berlin
4. Mahdavi M, Fesanghary M, Damangir E (2007) An improved harmony search algorithm for solving optimization problems. Applied Mathematics and Computation 188:1567–1579
5. Mukhopadhyay A, Roy A, Das S, Das S, Abraham A (2008) Population-variance and explorative power of harmony search: an analysis. In Proceedings of 3rd IEEE International Conference on Digital Information Management (ICDIM 2008), 13-16
6. Geem ZW (2006) Optimal cost design of water distribution networks using harmony search. Engineering Optimization 38:259-280
7. Wang CM, Huang YF (2009) Self-adaptive harmony search algorithm for optimization. Expert Systems with Applications, Online: doi:10.1016/j.eswa.2009.09.008
8. Degertekin S (2008) Optimum design of steel frames using harmony search algorithm. Structural and Multidisciplinary Optimization 36:393–401
9. Geem ZW (2006) Improved harmony search from ensemble of music players. Lecture Notes in Artificial Intelligence 4251:86–93