Parameter Controlled Harmony Search Algorithm for Solving the Four-Color Mapping Problem

Bnar Faisal A. Daham
Software Engineering Dept.
College of Engineering
Salahaddin University
Erbil - Iraq
Email: bnar_faisal {at} yahoo.com

Mohammed Nasseh Mohammed
Surveying Engineering Dept.
College of Engineering
Salahaddin University
Erbil - Iraq

Kanar Shukr Mohammed
Software Engineering Dept.
College of Engineering
Salahaddin University
Erbil - Iraq

Abstract- **The harmony search algorithm is a recently developed meta heuristic capable of solving discrete and continuous valued optimization problems. However, the nature of pre-defined constant parameters limits the exploitation of the algorithm. In this paper a number of deterministic parameter control rules are used in harmony search algorithm to fine-tune these parameters individually and dynamically, making harmony search a more dynamic algorithm which is able to achieve better results in solving the four color map problem. In this problem, it is said one can color the regions of a map using a maximum of four colors only with the constraint that no neighboring regions should share the same color. Neighboring regions mean two regions having a common boundary, not just a common point.**
Experimental results reveal that the new approach can find better solutions when compared to the original harmony search and several other heuristics, making harmony search a strong mechanism to perform optimization tasks.

Keywords: Optimization, Harmony Search Algorithm, Four-Color Mapping Problem, Map Coloring, Parameter Control.

I. INTRODUCTION

In mathematics and computer science, optimisation refers to a process of selecting or finding the best element from a set of available alternatives. Every process has the potential to be optimized, and indeed, many challenging problems in science, engineering, economics, and business can be formulated as optimization problems. The objective of a formulated optimization problem can be the minimization of time, cost, and risk or the maximisation of quality and efficiency [1].

The harmony search (HS) algorithm is one of the recently developed optimization algorithms works by generating a new vector that encodes a candidate solution, after considering all of the existing vectors. This forms a sharp contrast with conventional evolutionary approaches such as genetic algorithms that consider only two (parent) vectors in order to produce a new vector. HS algorithm increases the robustness and flexibility of the underlying search mechanism hence it ensures better solutions [2].

The nature of pre-defined constant parameters limits the exploitation of the algorithm, while deterministic parameter control approach uses predefined parameter alteration strategies to modify parameters deterministically [2][3]. In this paper deterministic parameter control has been used to dynamically modify the pitch adjusting rate (PAR) and harmony memory consideration rate (HMCR) parameters of HS algorithm by using a number of deterministic rules to adjust HS parameters making HS a more dynamic and efficient algorithm that is capable of finding better solutions.

Four Color Mapping is one of the combinatorial optimization problems that was first conjectured in 1852 by Francis Guthrie [4][5]. The Four-Color Mapping can be said that the regions of any simple planar map can be colored with only four colors, in such a way that any two adjacent regions have different colors [6][7][8]. Two regions are said to be adjacent if and only if a common boundary is shared, not just a common point [7][9]. Figure 1 illustrates the adjacency of regions in the given sample. Regions A and B have a common boundary, same goes with regions C and D. But regions A and D only share a common point, so do regions B and C. Therefore, by definition, regions A and B are adjacent, as well as regions C and D, but not regions A and D, and regions B and C [9]. Figure 2 illustrates 29 regions have been colored with four colors and no adjacent regions have the same color.

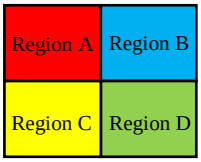


Figure 1: Adjacency of regions

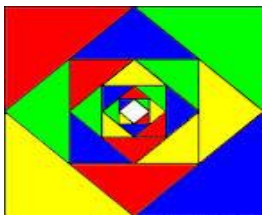


Figure 2: Graph colored using only four colors

In the rest of the paper; HS algorithm is overviewed in Section 2. Section 3 describes PCHS algorithm, and solving the Four-Color Mapping Problem using PCHS algorithm is described in section 4. In section 5, tests and results are presented. Finally, some concluding remarks are presented in Section 6.

II. HARMONY SEARCH ALGORITHM

Computer scientists have found a significant relationship between music and the process of looking for an optimal solution. This interesting connection led to the creation of the HS algorithm. It is a new kind of meta-heuristic algorithm mimicking musicians' approach to finding harmony while playing music [10]. When musicians try to create music, they

may use one or a combination of the three possible methods for musical improvisation which are as follows: (1) playing the original piece, (2) playing in a way similar to the original piece, and (3) creating a piece through random notes [9][11].

Zong Woo Geem et al. [10][12][13], noticed the similarity of this behavior in achieving the optimal solution to a problem. In 2001, he proposed three methods corresponding to the three techniques namely the use of (1) random selection, (2) memory consideration, (3) and pitch adjustment. These became the elements of the newly developed meta-heuristic optimization algorithm called the HS algorithm [9][10]. Figure 3 illustrates the basic Harmony Search Algorithm [9][10][11].

Begin
Define Objective function f(x), x=(x1,x2, …,xd)^T
Define harmony memory accepting rate (r_{accept})
Define pitch adjusting rate (r_{pa}), pitch limits and bandwidth
Generate Harmony Memory with random harmonies
While (t < max number of iterations (cycles))
  While (t < max number of variables)
    If (rand < r_{accept}), choose a value of var i
      If (rand < r_{pa}), adjust the value
      End if
    Else choose a random value
    End if
  End while
  Accept a new harmonics (solutions) if better
End while
Choose the best solution
End

Figure 3: Basic harmony search algorithm

Mathematically, the design of a basic HS algorithm is generally based on the following steps [1][13][14]:
Step 1. Initialize the problem and algorithm parameters.
Step 2. Initialize the Harmony Memory (HM).
Step 3. Improvise a New Harmony memory.
Step 4. Update the Harmony memory.
Step 5. Check the stopping criterion.

The capability of harmony search algorithm in solving problems has been proven effective in various studies [9]. In this study, the researchers aim to determine the feasibility of the said algorithm and its parameter controlled in solving Four-color mapping problem.

III. A PARAMETER CONTROLLED HARMONY SEARCH ALGORITHM

The parameters HMCR and PAR introduced in Step 3 above, help the algorithm to find globally and locally improved solutions [2][12]. Traditional HS uses fixed, pre-defined values for each of HMCR and PAR throughout the entire search process, making it hard to determine a good setting without a good amount of trial runs. The results of the search give no hint on which parameter HS should be adjusted in order to gain better performance. A number of deterministic parameter control rules are used to fine-tune HS parameters dynamically, in order to address the issues raised due to fixed parameters [2]. The aim is not only to produce a better searching result at the end, but also to maximize the utilization of every iteration. In other words, to improve the performance of the HS algorithm and eliminate the drawbacks lies with fixed values of HMCR and PAR. Parameter controlled harmony search PCHS algorithm uses variable values of HMCR and PAR in improvisation step (Step 3) changing dynamically in each iteration, ranging from 0 to 1 for each. The mentioned parameters are expressed as shown in equation (1) & (2) [2][15][16], and the design of general steps of the PCHS algorithm is shown in figure 4.

HMCR(iter)=HMCRmin+(((HMCRmax-HMCRmin)/maxIter)*iter) ...(1)
PAR(iter)=PARmin+(((PARmax-PARmin)/maxIter)*iter) ..... (2)
where,
HMCR(iter) = Harmony Memory Consideration Rate for each generation
HMCRmin = Minimum Harmony Memory Consideration Rate
HMCRmax = Maximum Harmony Memory Consideration Rate
PAR(iter) = Pitch Adjustment Rate for each generation
PARmin = Minimum Pitch Adjustment Rate
PARmax = Maximum Pitch Adjustment Rate
iter = Current iteration
Maxiter = Maximum number of iterations

PCHS Algorithm Procedure
Step1: Initialize PCHS Algorithm Parameters:
f(x): Objective Function
x: Decision Variable
N: Number of Decision Variables
HMS: Harmony Memory Size
$HMCR_{min}$: Minimum Harmony Memory Consideration Rate
$HMCR_{max}$: Maximum Harmony Memory Consideration Rate
PAR_{min}: Minimum Pitch Adjustment Rate
PAR_{max}: Maximum Pitch Adjustment Rate
Maxiter = Maximum Number of Iterations
Step2: Initialize Parameter Controlled Harmony Memory (PCHM)
Find best and worst harmony vectors for the PCHM
Step3: Improvise PCHM
While (not Maxiter)
PAR = PARmin + (((PARmax - PARmin) / maxIter) * iter)
HMCR = HMCRmin + (((HMCRmax - HMCRmin) / maxIter) * iter)
  For i=1 to number of decision variables N do
    If rand() < HMCR
    MemoryConsideration(i)
      If rand < PAR
      PitchAdjustment(i)
      End if
    Else
    RandomSelection(i) /*Choose a random value of variable*/
    End if
  End for
Step4: Update PCHM
Calculate the Objective Function of new harmony vector
  If (new solution < worst solution)
  Accept the new harmony and replace the worst in PCHM
  End if
Find best and worst harmony vectors for the PCHM
End while
Best = best current solution
End

Figure 4: Steps of parameter controlled harmony search algorithm

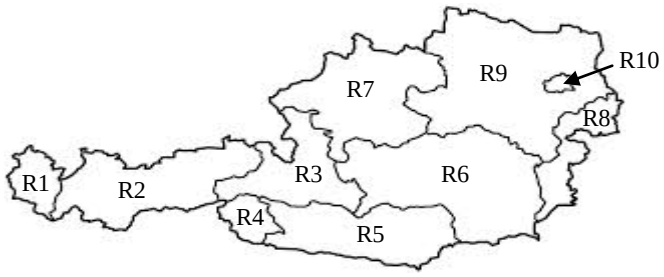
IV. SOLVING THE FOUR-COLOR MAPPING PROBLEM USING PCHS ALGORITHM

In this section, we will discuss the new approach of HS in achieving a four-colored map solution. Each candidate solution is generated as a vector whose variables correspond to the regions of the map. Vectors are arrays of integer variables

that serve as the solution to the problem. Regions to be colored take values 1, 2, 3, or 4 representing the four colors where each number stands for a color in the solution such as 1=red, 2=green, 3=blue and 4=yellow. The size of the solution is equivalent to the number of regions in the map. Figure 5 shows the representation of the solution. The general formula of each candidate solution is as follows:

R = <R1, R2, R3 ..., Rn> ..... (3)

where **R** represents the regions having values 1, 2, 3, or 4 and **n** is the number of regions in the map. The optimization process starts by generating random solutions in the harmony memory. Equation (3) illustrates the characteristics of each candidate solution in the memory. After initializing the harmony memory, it is followed by the improvisation process. Each possible solution is evaluated for its respective objective value. Then the optimization process ends if the process generates an optimal solution or when the maximum number of improvisations is reached. This is where a new harmony or a possible solution is generated through random selection, harmony memory consideration or pitch adjustment.



R=<R1,R2,R3,R4,R5,R6,R7,R8,R9,R10>

Figure 5: Vector representation for Austria map

The design of a basic PCHS algorithm generally includes the following steps:

1. Initialize PCHS algorithm Parameters: In order for the Harmony Search Algorithm to start, the following parameters must be set:

- Number of decision variables: Each harmony is composed of several decision variables.
- Number of cycles (iterations): One of the termination conditions of the optimization process.
- Harmony Memory Size: Refers to the number of solutions that will be stored in the harmony memory.
- Dynamic Harmony Memory Consideration Rate: HMCR is the core parameter of HS. It determines whether the new vector should be chosen from the harmony memory, or randomly selected from the range of all possible values. HS with a low HMCR takes less consideration of the historical values but focuses more on the entire value range when improvising new harmonies. HS with a high HMCR tries to produce a new harmony out of existing values stored in the harmony memory. The dynamic HMCR value that increases as the search is progressing is given by equation (1).
- Dynamic Pitch Adjustment Rate: PAR is the important parameter which mostly affect the rate of finding and converging to the optimal solution. It shows the greatest impact when solving continuous valued optimization problems. Traditional HS uses pre-defined and fixed PAR throughout the search. This either results in slow initial exploration in the solution space, or inefficiency in searching for the optimal solution. Dynamic PAR adjustment method are given by equation (2).

2. Initialize PCHM: In this step harmony memory is filled with randomly generated solution vectors.

3. Random Selection: Random selection is where a variable in the harmony takes a random value; in this problem it will select one random color from the available four color options.

4. Memory Consideration: Values or colors in the harmony memory have a chance to be selected when performing memory consideration.

5. Pitch Adjustment: Selected value from the memory can be altered to obtain a variation through pitch adjustment. Figure 6 illustrates how pitch adjustment is done.

```
Begin:
Input:
Public double: BestHarmony[] {get; set;}
Double: rand% random number generator with (1,2,3,4)
Integer: varIndex% Index of array
Integer: Temp% temporary variable
Integer: NCHV[varIndex]% Pattern
Integer: NVAR% Number of Variable
Process:
If r_p < par, then
//pick one of the neighbors of the region in question at random and
exchange colors between the region and the chosen neighbor, Specifically//
        BestHarmony=new double[NVAR+1]
        Random number generator
        Temp = NCHV[varIndex]
        NCHV[varIndex] = BestHarmony[varIndex]
End if
Output: NCHV
End
```

Figure 6: Pitch adjustment process

6. Objective (Fitness) Function:

The evaluation function will return a value that will represent the fitness of a harmony vector. The function aims to get a minimum result, wherein, the smaller the fitness value, the better the vector is. The value also represents the number of errors present in the solution. If the fitness value reaches zero, then that is the optimal solution and the optimization process will stop. Figure 7 illustrates how the objective function works.

```
Begin:
Input:
Integer: N% Number of regions
Integer: X[N]% Array of colored regions
Integer: Matrix[N][N]% Array of adjacency Matrix N by N depend on
number of region
Process:
    For i=1 to N do
        For j=1 to N do
            If i ≠ j then
                If Matrix[i][j]=1 then
                    If X[i]=X[j] then
                    Fit=Fit+1
```

```
                End If
            End If
        End for
    End for
    Fit=Fit / 2
Output: Fit
End
```

Figure 7: Objective function process

7. Illustrating the PCHS Algorithm Process to Map Coloring:

Using the map in Figure 5, random harmony vectors are initialized to represent the solution for the colored map using equation 3. In this example, we set our harmony memory to two so that HMS=2, R^1=[2,2,1,2,4,2,3,4,1,1] and R^2=[3,1,2,1,3,2,3,2,2,4]. Then improvise a new harmony vector through Random Selection, Memory Consideration, and Pitch Adjustment. After that we evaluate the harmony using the objective function process in Figure 7, only half of the matrix is evaluated since the matrix is symmetric. The smaller resulting value is the better result. R^1 is evaluated as shown bellow. Using the same function, the result of R^2=4.

$$
\begin{aligned}
R^1 = {} & 0\times0 + 1\times1 + 0\times0 + 0\times0 + 0\times0 + 0\times0 + 0\times0 + 0\times0 + 0\times0 + 0\times0 \\
& 0\times0 + 1\times0 + 0\times0 + 0\times0 + 0\times0 + 0\times0 + 0\times0 + 0\times0 + 0\times0 \\
& 0\times0 + 1\times0 + 1\times0 + 1\times0 + 1\times0 + 0\times0 + 0\times0 + 0\times0 \\
& 0\times0 + 1\times0 + 0\times0 + 0\times0 + 0\times0 + 0\times0 + 0\times0 \\
& 0\times0 + 1\times0 + 0\times0 + 0\times0 + 0\times0 + 0\times0 \\
& 0\times0 + 1\times0 + 1\times0 + 1\times0 + 0\times0 \quad = 2 \\
& 0\times0 + 0\times0 + 1\times0 + 0\times0 \\
& 0\times0 + 1\times0 + 0\times0 \\
& 0\times0 + 1\times1 \\
& 0\times0
\end{aligned}
$$

Improvisation of a new harmony vector and evaluation of a new harmony are both repeated until a series of finite number of improvisations is achieved. The best harmony in the harmony memory is selected to become the solution. The map will be colored using the corresponding colors of the values in the solution.

V. TESTS AND RESULTS

HS and PCHS algorithms were implemented in Microsoft Visual C# 2010 and run on a computer whose processor is Intel(R) Core™ 2.50 GHz, with 6 GB main memory. The algorithms were applied on 7 regions Australia map, 10 regions Austria map, 18 regions Iraqi map, 29 regions Kurdistan-Iraq map, 32 regions China map, 48 regions USA map, and finally 50 regions Africa continent map.

The comparison between the original and parameter controlled HS algorithms has been accomplished based on the time and number of cycles required to find the optimal solutions for each vector of regions. Ten to fifteen runs for each instance have been achieved and the standard deviation of time and number of cycles (required to reach the optimal solution) have been reported in Table 1.

TABLE 1: STANDARD DEVIATION OF ELAPSED TIME AND NUMBER OF CYCLES FOR HS AND PCHS ALGORITHMS FOR SOLVING FOUR-COLOR MAPPING PROBLEM

Regions	Time (Seconds)		Cycles	
	HS	*PCHS*	*HS*	*PCHS*
7	0.007656975	0.004690345	10.07223908	6.050619803
10	0.006988317	0.006958062	99.88413287	17.78057255
18	0.173869271	0.068198587	1939.580357	870.0691984
29	0.889333565	0.470241322	7042.100332	3603.780623
32	1.073332541	0.747933408	7576.397402	5990.872027
48	6.400353566	3.609366029	21703.68652	16648.60471
50	7.961991411	5.038387111	26454.82923	19906.64556

Table 1 shows how the PCHS algorithm has recorded higher performance than the HS algorithm. Figures 8 and 9 are illustrating the difference between the performance of both PCHS & HS algorithms in reaching the optimal solution in terms of time and number of cycles respectively.

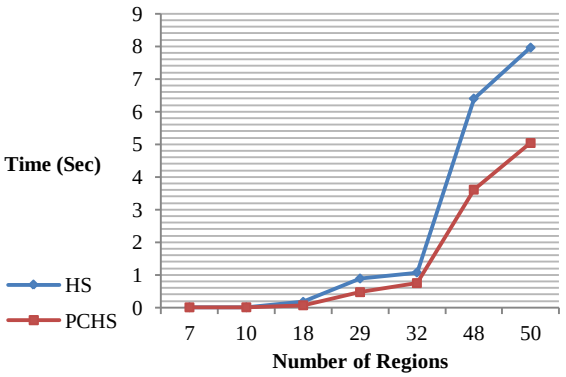


Figure 8: Difference in elapsed time between HS and PCHS algorithms

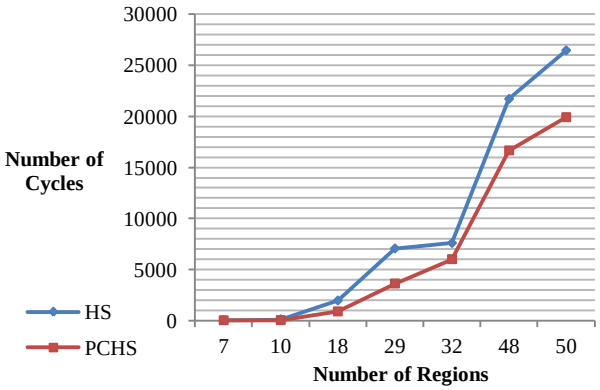


Figure 9: Difference in number of cycles between HS and PCHS algorithms

VI. CONCLUSIONS

This paper has discussed the impact of constant parameters in harmony search, and parameter controlled strategies for tuning them individually and dynamically. The improvements include better exploration of the solution space, maximized utilization of all iterations, and fine-tuning towards optimal solution. Dynamically parameter-tuned HS is good at locating and producing optimal solution.

The computational results illustrate that the PCHS algorithm presents better performance than the original HS algorithm where the PCHS reached the optimal solution faster than the original HS applying tests on maps with different number of regions. The results also indicate that the time and number of cycles increase as the size of the map increases.

This is because it will be more difficult to find an optimal solution as the size of the map gets bigger.

For future work a hybridization can be applied between HS and another optimization algorithm. One of the approaches is to use Ant or Bee colony algorithms in generating the harmonies (candidate solutions) instead of producing them randomly.

VII. REFERENCES

[1] Parikshit Yadav, Rajesh Kumar, S.K. Panda, and C.S. Chang, "An Intelligent Tuned Harmony Search algorithm for optimisation", Department of Electrical and Computer Engineering, National University of Singapore, Information Sciences 196, pp. 47-72, 2012.

[2] Ren Diao and Qiang Shen, "Deterministic Parameter Control in Harmony Search", Department of Computer Science, Aberystwyth University, UK, 2013.

[3] A.E. Eiben, R. Hinterding, and Z. Michalewicz, "Paramter Control in Evolutionary Algorithms", IEEE Transactions on Evolutionary Computation, vol. 3, no. 2, pp. 124–141, 1999.

[4] Robin Wilson, "Four Colours Suffice", Allen Penguin Press books, 2002.

[5] Mohammed S. Ibrahim, Ahmed T. Sadiq, and Ali M. Sagheer, "Hybrid Scatter Search Algorithm for 4-Color Mapping Problem", The 13th International Arab Conference on Information Technology (ACIT), 10-13 December 2012, ISSN 1812-0857.

[6] Georges Gonthier, "Formal Proof—The Four-Color Theorem", Notices of the AMS, Volume 55, Number 11, pp. 1382-1393, December 2008.

[7] Georges Gonthier, "A computer-checked proof of the Four Colour Theorem", Microsoft Research Cambridge, 2005.

[8] Andreea S. Calude, "The Journey of the Four Colour Theorem Through Time", Department of Mathematics, The University of Auckland, New Zealand, December 2000.

[9] Romie B. Horca and John Paul T. Yusiong, "Using Harmony Search Algorithm to Solve the N-Region Four Color Map Problem", Journal of Applied Computer Science & Mathematics, Computer Science Section, no. 12 (36), Suceava, 2012.

[10] Victor II M. Romero, Leonel L. Tomes, and John Paul T. Yusiong, "Tetris Agent Optimization Using Harmony Search Algorithm", International Journal of Computer Science Issues (IJCSI), Vol. 8, Issue 1, pp. 22-31, January 2011.

[11] Xin-She Yang, "Harmony Search as a Metaheuristic Algorithm", in: Music-Inspired Harmony Search Algorithm: Theory and Applications (Editor Z. W. Geem), Studies in Computational Intelligence, Springer Berlin, vol. 191, pp. 1-14, 2009.

[12] Kang Seok Lee and Zong Woo Geem, "A new meta-heuristic algorithm for continuous engineering optimization: harmony search theory and practice", Computer Methods in Applied Mechanics and Engineering 194, pp. 3902-3933, 2004.

[13] Sachin A. Patil and D. A. Patel, "An Overview: Improved Harmony Search Algorithm and Its Applications in Mechanical Engineering", International Journal of Engineering Science and Innovative Technology (IJESIT), Volume 2, Issue 1, pp. 433-444, January 2013.

[14] Xiaobo Liu, Zhihua Cai, and Chao Yu, "A Hybrid Harmony Search Approach Based on Differential Evolution", Journal of Information & Computational Science, pp. 1889-1900, Available at http://www.joics.com, Binary Information Press, October 2011.

[15] M. Mahdavi, M. Fesanghary, E. Damangir, "An improved harmony search algorithm for solving optimization problems", Applied Mathematics and Computation 188, pp. 1567–1579, 2007.

[16] Wan-li Xiang, Mei-qing An, Yin-zhen Li, Rui-chun He, Jing-fang Zhang, "An improved global-best harmony search algorithm for faster optimization", Expert Systems with Applications 41, pp. 5788–5803, 2014.