Solving Cell Placement Problem Using Harmony Search Algorithms

Rose M. Al Qasem
Department of Computer Engineering
Faculty of Engineering Technology
Al Balqa Applied University
Amman, Jordan
rose.alqasem@fet.edu.jo

Samah M. Massadeh
Department of Computer Engineering
Faculty of Engineering Technology
Al Balqa Applied University
Amman, Jordan
samah.massadeh@fet.edu.jo

Abstract—**Cell placement is a phase in the chip design process, in which cells are assigned to physical locations. A placement algorithm is a way that satisfies the objectives and minimizes the total area while keeping enough space for routing. Cell placement is an NP-complete problem of very large size. In order to solve this problem, diversified heuristic algorithms are used. In this work, a new algorithm is proposed based on the harmony search algorithm. The harmony search algorithm mimics music improvisation process to find the optimal solution. Cell placement problem has many constraints, so in this work, the harmony search algorithm is modified to adapt to these constraints. Experiment results show that this algorithm is efficient for solving cell placement and is characterized by good performance, solution quality and likelihood of optimality.**

Keywords-optimization; harmony search ; cell placement

I. INTRODUCTION

Chip design is a process of consecutive phases, and one of the important phases in creating a VLSI circuit is physical design. The physical design process has four consecutive steps, namely partitioning, cell placement, routing and compaction. In the cell placement stage, the description of the physical layout of the cells is produced by assigning geometric coordinates to the cells. The smaller feature size is the objective of this placement. The small size has many advantages like higher performance and lower power consumption. The objective of the placement algorithm is to find a layout that optimally or nearly optimally minimizes a cost function. The cost function realizes that the major part is the area, but quite often involves the aspect ratio, to make the chip as close to square as possible. Cell placement is one of the NP-complete problems which are generally combinatorial in nature and have a noisy solution space. Several heuristic optimization techniques have been produced to solve the placement problem by using a set of diversified algorithms like genetic algorithms [1] and simulated annealing [2]. A comprehensive summary of those techniques is introduced in [3].

Harmony search (HS) is a phenomenon-mimicking meta-heuristic algorithm which has been proposed in [4]. HS is a novel approach inspired from the musical process of searching a perfect state of harmony as in music improvisation where the pitch of each musical instrument is improved to accomplish this better state of harmony. The pitch determines the quality of music, just as the fitness function value determines the quality of solutions. Furthermore, the musician improvisations are analogous to local and global search schemes in optimization strategies. HS has been used to solve various types of optimization problems, such as traveling salesperson problem [4], function optimization [5], mechanical structure design [6], tour routing [7], water network design [8], pipe network optimization [9], vehicle routing [10] and optimization of data classification systems [11]. It is showing significant improvements over other heuristics. It can be produced for both continuous and discrete variables. A special discrete variation of the HS is produced on the basis of introducing the stochastic derivatives for the discrete variables involved [12]. For the class of binary optimization problems, the HS algorithm is equivalent to a certain evolutionary algorithm as shown in [13]. A lot of modified HS algorithms have been studied in order to enhance the performance of the original version [14, 15].

In this work, we present a solution to the cell placement problem using the HS methodology, and in future work we will compare its performance with well-known evolutionary algorithms. The results show that the proposed algorithm can improve the quality of the solution. The rest of the paper is organized as follows: Section II describes briefly the cell placement problem. Section III gives a formal description of the HS theory, and Section IV demonstrates the proposed HS algorithm for cell placement, followed by presentation and discussion of the experimental results in Section V. Finally, Section VI wraps up our work.

II. CELL PLACEMENT PROBLEM: REPRESENTATION AND FITNESS

In this work, each solution is a sequence represented by the normalized polish notation (RPN) [16]. The sequence includes module names and two relational operators: for n blocks, a string with n blocks (cells) and n-1 operators of the + or * type, corresponding to a vertical or horizontal cut respectively. As an example, the expression (12+34*+5+67+*) is an encoding for the arrangement representing a possible solution as illustrated in Figure 1.

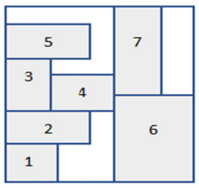

Fig. 1. Slicing structure.

Commonly, the new solutions are reproduced from the existing ones by using certain methods which are presented in [16, 17]. The quality of the new solution is evaluated by a fitness value. The proposed measure of the fitness reflects how far the solution is from the optimal one. The fitness measure is calculated by the following equation:

$$F=\alpha\frac{T}{WL}+(1-\alpha)\frac{W}{L} \qquad (1)$$

where, T is the total area of all cells regardless of the placement, L and W are the length and width of the rectangle enclosing all the cells and the product WL represents the area associated with the solution. The factor α used to determine the relative significance of the aspect ratio to the actual area, the value of the factor α is within the range of [0–1]; small values of α are used to favor square arrangements. If α=1 then aspect ratio is not optimized.

III. HARMONY SEARCH ALGORITHM

HS algorithm has been proposed in [4]. In order to summarize HS, it is required to idealize the process of improvisation done for the harmony. When the harmony is composed, the musicians try various possible combinations of the music pitches stored in the memory. This search for the better state of harmony is analogous to the procedure of finding the optimal solutions to the problem. In music improvisation stage, the musician can select one of three options: perform any pitch from the memory, perform a pitch adjacent to any other in the memory or perform a random pitch from the range of all possible pitches. Similarly, in HS algorithm each decision variable picks a value from three options: take any value from the memory, take a value adjacent to any value in the memory or take a random value from the domain of all possible values. HS algorithm formalized these three options for making a new heuristic and the three corresponding process are memory use or consideration, pitch (solution) adjustment and randomness. HS algorithm controls the three options by means of three main parameters: harmony memory (HM), harmony memory consideration rate (HMCR), and pitch adjustment rate (PAR).

A. *The GSA Behavior*

The HS algorithm tries to solve optimization problems, where a single objective is considered, by finding a vector which optimizes (minimizes or maximizes) a certain objective function. The details of the harmony algorithm are illustrated by the following steps.

1) Step 1

Initialize the harmony search memory (HM), the initial HM matrix contains a certain number of randomly generated solutions to the problem under consideration. For an n-dimension problem, a HM with the size of HMS (harmony memory size) is represented by the following matrix:

$$\text{HM}=\begin{bmatrix} x_1^1, x_2^1, \dots, x_n^1 \\ x_1^2, x_2^2, \dots, x_n^2 \\ \cdot \\ \cdot \\ \cdot \\ x_1^{HMS}, x_2^{HMS}, \dots, x_n^{HMS} \end{bmatrix} \qquad (2)$$

where, $(x_1^i, x_2^i, \dots, x_n^i\ )$ is a candidate solution and n is the dimension of the problem.

2) Step 2

Improvise a new solution from the HM. Each solution is produced based on harmony memory considering rules. Each component of new solution x_j is obtained as follows:

- Generate a random number R1 within the range [0-1]
- Compare the random number R1 with a HMCR. The HMCR is defined as the probability of selecting a component from the HM members.
- If R1<HMCR, the x_j of the new solution comes from the HM, it is chosen from the jth dimension of a random HM vector
- The obtained solution xj is further mutated according to PAR, where the PAR determines the probability of a candidate solution selected from the HM to be mutated.
- Generate a new random number R2 within the range [0-1]
- If R2<PAR, then adjust the obtained solution as follows:

$$x_j^{new} = x_j \pm \text{rand}(0,1).\text{BW} \qquad (3)$$

where BW is bandwidth factor, which is used to control the local search

- If the condition R1<HMCR fails, the xj variable of the new solution is generated randomly as follows:

$$x_j^{new} = l_j + (u_j - l_j).\text{rand}(0,1) \qquad (4)$$

where l, u is the lower and upper bound for the given problem. The generation of new solutions in the HS strategies makes full use of all the HM vectors.

3) Step 3

Update HM: After the new solution is generated, it is evaluated. If its objective function value is better than that of the worst member in the HM, the worst member will be replaced by the new solution. Otherwise, the generated solution is deleted.

4) Step 4

Repeat Step 2 to Step 3 until a termination criterion (e.g. maximum iterations) is satisfied.

The parameters of HMCR and PAR usually have critical effects in the optimization performance of the HS strategy. One of the challenging topics for the HS algorithm is choosing the appropriate values for these parameters. Authors in [18] produced an adaptive method for adjusting PAR and BW parameters in the improved HS (IHS) algorithm. In this method, in order to improve the performance of the HS, the values of PAR and BW increase and decrease dynamically and respectively with the growth of HS iterations. The updates of PAR and BW are often case dependent and are therefore difficult to determine.

IV. HS ALGORITHM BASED CELL PLACEMENT PROBLEM

As mentioned above, cell placement can be represented as a two-dimensional bin packing problem, where its main goal is to order the cells in a way that minimizes the area and produces near square shape while providing enough space for routing. In this work, we propose a new algorithm for cell placement problem based on HS algorithm, in which each pitch will be an agent looking for an optimal solution in the search space. It is difficult to solve the cell placement problem by the traditional HS algorithm, because this problem needs meet simultaneously several constraints. For this reason, the update of pitch (solution) depends on some rules, so as to solve the cell placement problem by using harmony search algorithm, there are some modifications that must be done as will be shown in the following procedure. The HS algorithm based cell placement problem is outlined as follows:

```
1. Harmony memory initialization
    1.1 Randomly generate initial population harmony memory of vectors (solutions) ($X^1, X^2, \ldots, X^{HMS}$) as many as the HMS where $X^i = x_j^i$, i=1…HMS and j=1…n, n is the dimension of the problem.
    1. 2 Store the vectors in HM matrix as shown in (2).
2. Construction and evaluation of new solution candidate.
    2. 1 Generate a new vector $X^{new}$:
    For each component (decision variable) $x_j^{new}$ of the new vector $X^{new}$
    do {
    Generate random number R1 within the range [0-1]
    If (R1<HMCR)
    Then
    {
    Let the decision variable $x_j$ in $X^{new}$ be the jth dimension of a randomly selected HM member ($x_j^i$). This selection must meet the constraints of the cell placement problem, otherwise another HM member should be selected.
    Generate random number R2 within the range [0-1]
    If (R2<PAR)
    Then
    Apply pitch adjustment distance BW to mutate $x_j$ as follows:
    If selected HM member $x_j^i$ is cell
    Then
    $x_j^{new} = x_j^i \pm \mathrm{BW}$        (5)
    Otherwise
    $x_j^{new} = x_j^i$        (6)
    }
    Else
    Let $x_j$ in $X^{new}$ be a random value.
    Where $x_j \in \{1, 2\ldots m\}$, m is the number of cells.
    }
    2.2 Evaluate the fitness of Xnew by objective fitness function f as in (1).
3. Update harmony memory
    If the fitness of new solution $X^{new}$ is better than the fitness of the worst HM member
    Then
    Replace the worst HM member with new solution $X^{new}$.
    Else
    Disregard $X^{new}$.
4. If Stopping Criteria Not Met Then Go to step 2 Else Stop.
```

The effects of parameters are analyzed. The best performance of HS algorithm for solving placement problem is detected, when the values of the parameters are set as follows: HMS is the size of the HM. It generally varies from 50 to 100 and the used value is 60. HMCR is the rate of choosing a value from the HM. It generally varies from 0.7 to 0.99 and the used value is 0.8. PAR is the rate of choosing a neighboring value. It generally varies from 0.1 to 0.5 and the used value is 0.22. BW is the amount of maximum change in solution adjustment and the used value is 1. It is possible to vary the parameters values as the search progresses.

V. RESULTS

The quality measures used in this study include the following: average fitness value for a set of runs, number of iterations employed (elapsed time), and number of successes in a set of runs (success rate). In this work, the performance of the HS algorithm for solving the cell placement problem was evaluated by using two methods. In the first method, instances from MCNC benchmark are chosen to assess the quality of the solutions. In the second method, the algorithm is performed on three cases of cell placement problem generated randomly and have known optimal solutions to measure the possibility of reaching the optimal solution.

In the first test, three cases from MCNC benchmarks, Xerox with 10 cells, Ami33 with 33 cells, and Ami49 with 49 cells, were chosen and tested. Table I shows a summary of the obtained results. The measure used in this test is the quality of solution in terms of wasted area. The number of replicates which carried out for each problem is 10 runs, each with different random starts. The algorithm is brought to stop when 6000 iterations have passed since the last improvement was happened or the number of iterations reaches 20,000, 40,000 and 60,000 for Xerox, Ami33 and Ami49 respectively. The results for the three instances of MCNC benchmark are summarized in Table I. The percentage of wasted area for the best, worst and mean case is noted in the last three columns of the table. The obtained results present that the proposed algorithm achieves good area utilization in the best and mean wasted area. Figure 2 illustrates the search improvement of the algorithm through time, waste area versus iterations number for Ami49 and Ami33. The figure clearly indicates that through the passage of time the proposed algorithm can achieve better results.

TABLE I. BENCHMARK RESULTS

			Wasted Area (%)		
	# of cells	# of Iterations	Best	Worst	Mean
Xerox	10	20,000	3.3	6.2	4.8
Ami33	33	40,000	5.4	8.8	6.5
Ami49	49	60,000	7.5	9.5	8.5

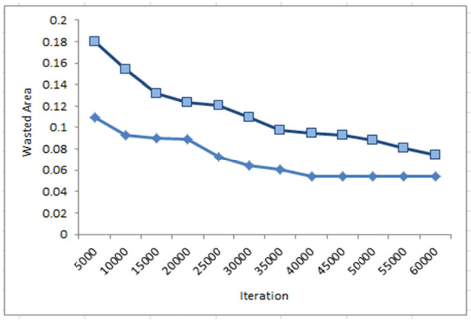

Fig. 2. Search progress. Waste area vs iterations (Ami49, Ami33).

In order to perform the second test, three optimal cases for 10, 20 and 30 cells were generated. As illustrated in Figure 3, in the first case, an instance is produced by cutting a square with total area of 25 square units into 10 various size blocks. In the second case, an instance is produced by cutting a square with total area of 100 square units into 20 various size blocks. In the third case, an instance is produced by cutting a square with total area of 225 square units into 30 various size blocks. The number of trials for each case is 10 times with different initial population. The algorithm is halted if the optimal solution is achieved or the number of iterations reaches 10,000, 25,000 and 40,000 for case1, case2 and case3 respectively. The metric used to evaluate the proposed algorithm in this test is the likelihood of optimality or success rate. Table II shows the result for the success rate for each case. As shown in the third column of Table II, in the first case, the success rate for the harmony search based cell placement is 90% over 10 different random trials. The algorithm obtained the optimal solution in most runs. In the second case, the success rate for this algorithm is 70%, whereas the success rate dropped to 50% in the last case.

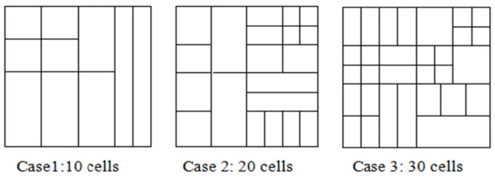

Fig. 3. Artificial problems for known optimal solution

TABLE II. ARTIFICIAL PROBLEM OF KNOWN OPTIMAL SOLUTION

# of cells	# of iterations	Success Rate(out of 10)
10	10,000	9
20	25,000	7
30	40,000	5

It is clear that this algorithm is robust for small size problems, but on the other side it needs enhancement to get better performance for the problems with medium and large size.

VI. CONCLUSION

In this work a new algorithm was presented to solve the cell placement problem which is based on HS algorithm. The proposed algorithm has achieved good results regarding the solution quality and the likelihood of optimality. The experiments show that the performance of the proposed algorithm appears in terms of increasing success rate for finding the optimal solution, and the quality of solution. It realizes promising area utilization in terms of minimum and average of wasted area for a set of trials. This study evidences that the HS algorithm is one efficient algorithm for solving the cell placement problem. Hence, in future work we will make a comparison between this algorithm and other optimization algorithms.

REFERENCES

[1] T. W. Manikas, M. H. Mickle, "A Genetic Algorithm for Mixed Macro and Standard Cell Placement", 45th Midwest Symposium on Circuits and Systems, Vol. 2, pp 115-118, 2002
[2] G. Nan, M. Li, D. Lin, J. Kou, "Adaptive Simulated Annealing for Standard Cell Placement", Advances in Natural Computation, Lecture Notes in Computer Science, Vol. 3612, pp. 943-947, Springer, 2005
[3] K. Shahookar, P. Mazumder, "VLSI Cell Placement Techniques", ACM Computer Survey, Vol. 23, No. 2, pp. 143–220, 1991
[4] Z. W. Geem, J. H. Kim, G. V. Loganathan, "A new heuristic optimization algorithm: harmony search," Simulation, Vol. 76, No. 2, pp. 60–68, 2001
[5] K. S. Lee, Z. W. Geem, "A new meta-heuristic algorithm for continuous engineering optimization: harmony search theory and practice", Computer Methods in Applied Mechanics and Engineering, Vol. 194, No. 36–38, pp. 3902–3933, 2005
[6] K. S. Lee, Z. W. Geem, "A new structural optimization method based on the harmony search algorithm", Computers and Structures, Vol. 82, No. 9-10, pp. 781–798, 2004
[7] Z. W. Geem, C. L. Tseng, Y. Park, "Harmony search for generalized orienteering problem: best touring in China", Lecture Notes in Computer Science, Vol. 3612, pp. 741–750, Springer, 2005
[8] Z. W. Geem, "Optimal cost design of water distribution networks using harmony search", Engineering Optimization, Vol. 38, No. 3, pp. 259-277, 2010
[9] Z. W. Geem, J. H. Kim, G. V. Loganathan, "Harmony search optimization: application to pipe network design", International Journal of Modelling and Simulation, Vol. 22, No. 2, pp. 125–133, 2002
[10] Z. W. Geem, K. S. Lee, Y. Park, "Application of Harmony Search to Vehicle Routing", American Journal of Applied Sciences, Vol. 2, No. 12, pp. 1552–1557, 2005
[11] X. Wang, X. Z. Gao, S. J. Ovaska, "Fusion of clonal selection algorithm and harmony search method in optimisation of fuzzy classification systems", International Journal of Bio-Inspired Computation, Vol. 1, No. 1-2, pp. 80–88, 2009
[12] Z. W. Geem, "Novel derivative of harmony search algorithm for discrete design variables", Applied Mathematics and Computation, Vol. 199, No. 1, pp. 223–230, 2008
[13] M. Padberg, "Harmony Search Algorithms for binary optimization problems", in: Operations Research Proceedings, pp. 343–348, Springer, 2012
[14] M. G. H. Omran, M. Mahdavi, "Global-best harmony search", Applied Mathematics and Computation, Vol. 198, No. 2, pp. 643–656, 2008

[15] Q. K. Pan, P. N. Suganthan, J. J. Liang, M. F. Tasgetiren, "A local-best harmony search algorithm with dynamic subpopulations", Engineering Optimization, Vol. 42, No. 2, pp. 101–117, 2010

[16] D. F. Wong, C. L. Liu, "A New Algorithm for Floorplan Design", 23rd ACM/IEEE Design Automation Conference, Las Vegas, Nevada, USA, pp. 101–107, February 6, 1986

[17] J. P. Cohoon, S. U. Hedge, W. N. Martin, D. S. Richards, "Distributed genetic algorithms for the floorplan design problem", IEEE Transactions on Computer-Aided Design, Vol. 10, No. 4, pp. 483–492, 1991

[18] M. Mahdavi, M. Fesanghary, E. Damangir, "An improved harmony search algorithm for solving optimization problems", Applied Mathematics and Computation, Vol. 188, No. 2, pp. 1567–1579, 2007