Performance Enhancement of Standard Cell Placement Techniques using Memetic Algorithm

Aaquil Bunglowala
Department of ECE
Sanghvi Inst.of Mangt. & Sc.
Indore, MP, India

Dr. B. M. Singhi
Department of ECE
Medicaps Inst.of Tech. & Mangt.
Indore, India

Dr. Ajay Verma
Department of EI
IET, DAVV
Indore, MP, India

ABSTRACT
The growing complexity in the electronic hardware now necessitates in improving the performance of searching algorithms. Genetic algorithms do not guarantee global optimum solution to NP-Hard problems but are generally good at finding acceptable solution to problems. In complex combinatorial spaces, hybridization with other optimization techniques can greatly improve the efficiency of search. Memetic algorithm (MA) is an improvisation over genetic algorithms (GA) that combines global and local search by using evolutionary algorithms to perform exploration while the local search methods are used for exploitation. Here, exploitation is the process of visiting entirely new regions of a search space where the gain can also be high.

This paper discusses the (MAs) as a solution to Standard Cell Placement (SCP) problem and procedures are laid down to strike a balance between genetic search and local search in MAs. A comparison of MA with the already established results for SCP using conventional and Hybrid techniques by the author depicts improvement in the performance of SCP algorithm in terms of solution quality and computing speed. About 15% improvement in overall wire-length was observed along side it being 25% faster over the Tabu Search (TS) algorithm discussed in previous works of the author.

General Terms: Algorithm, exploration, exploitation

Keywords: SCP, Memetic Algorithm (MA), NP-hard,.

1. INTRODUCTION
The design of circuit for VLSI is accomplished in several iterative stages including system specifications, functional design, logic design, circuit design and physical design. The physical design process permits layout of chip with overall optimized performance keeping the fabrication cost low. Such design process is complex and time consuming. This process is split into partitioning, floor-planning, placement, routing and compaction. Placement and routing are particularly important because a mistake made in these steps cannot be undone later. These two processes are interdependent and should be performed simultaneously.

Three types of cell placement techniques are common. They include macro cell placement, SCP and gate-array placement. In this paper the SCP is dealt with. It occurs when the standard cell layout design style is chosen for determining the layout of a chip. This gives a semi-regular layout which can be accomplished in a relatively short time. For Application Specific Integrated Circuit design, this approach is popular.

Standard cells are of same height but of varying widths. The cells are to be placed in rows and interconnections run through horizontal channels between the rows. In order to minimize the total area of the chip, one tries to minimize the interconnecting wire length. This is a NP-hard optimization problem.

1.1 SCP Problem
The task of SCP placement is to arrange cells on rows such that the cells do not overlap, interconnecting wires-lengths and the area of the chip are minimized and the placement is routable. Often the length and the height of the chip are given, or sometimes only the aspect ratio, or the aspect ratio and one side of the chip are specified. The number of feed through cells may also be specified by the designer.

The SCP can be stated as: Given an electrical circuit consisting of fixed rectangular shaped cells and a netlist stating interconnections among terminals on the periphery of the cells and on the periphery of the circuit itself, construct a layout indicating the position of each cell such that all the nets can be routed and to minimize the total area. The objective for high performance systems is to minimize the total delay of the system by minimizing the length of the critical paths. The quality of placement is based on several factors like layout area, completion of routing and circuit performance.

Let $c_1, c_2, \ldots c_k$ be the cells to be placed on the chip. Each c_n, $1 \leq n \leq k$, has associated with it a height H_n and width W_n. Let N= { $n_1, n_2, \ldots n_j$} be the set of nets representing the interconnections between different blocks. Let S = {$s_1, s_2, \ldots S_i$} represent the empty space allocated for routing between cells. Let L_n denote the estimated length of net. The variable k stands for number of cells, j represents number of nets and i represents the number of channels on the layout. The placement problem now is to determine a rectangle for each of these blocks in the row denoted by R= { $r_1, r_2, \ldots r_k$} such that:

1. Each cell can be placed in the corresponding rectangle of width W_n and height H_n.
2. No two rectangles overlap i.e. $r_p \cap r_q = \varnothing$, $1 \leq p, q \leq k$
3. The placement is routable, i.e. s_r, $1 \leq r \leq i$, is sufficient to route all the nets.
4. The area of the rectangle bonding R and S is minimized, minimizing the net area of chip.

5. The total wire length i.e. $\sum L_n$ is minimal

SCP problem is computationally very hard. It is a NP-hard problem. These problems can not be solved in polynomial time. Trying to evaluate every possible placement to determine the best would take time proportional to the factorial of number of cells. Therefore, to search through a large number of candidate placement configurations efficiently, a heuristic algorithm must be used.

2. SCP TECHNIQUES

Previous works of author [3,4] developed the algorithms for SCP using SA, HNN, GA and TS and Threshold-Acceptance (TA) are briefly presented in this section for the sake of completion.

2.1 SCP by SA

Initially, the starting temperature has to be computed, which here is problem dependent. The initial solution is generated randomly and the x and y coordinates are determined. To generate new solutions, the neighborhood operation is used. To find whether the selection criteria for selecting new solution is good, the metropolis acceptance rule [17]. The simulated annealing algorithm is used to calculate the number of rows and an upper limit for the row length. The upper limit for the row length is taken as 110% of the estimated row-length. The values of row length and the number of rows are determined by the help of sum of the widths of all cells and the aspect ratio. Hence the simulated annealing starts with a fixed number of rows and sufficient row length. During the placement process the algorithm has to place all cells in this given area. The algorithm tries to minimize the row length according to the penalty term. Several test runs were performed to test the performance of the algorithm. Several cooling schedules were tried for it. The detailed results were presented in [2].

2.2 SCP by Hopfield neural network

The task is to place the given number of cells on the given layout with shortest total wire length. The placement should be valid placement, i.e., (i) on a position only 1 cell should be placed (E_1); (ii) a cell should not be placed on more than 1 positions (E_2)and (iii) every cell should be exactly placed on the layout(E_3). The Sum of all these energy terms with E_4 being the objective function give the total energy in the Lyapunov function form. In order to develop the distance matrix D_{xy} the information given for chip and cell dimensions is used. According to the cell dimensions the position on the rows are assumed. So, D_{xy} stores all the positions on the layout after calculations. All the diagonal elements of the matrix are zero. Subsequently the network is initialized by application of some input values to each neuron [19, 3].

2.3 SCP by GA

Here [3], we presented a GA for SCP. In this algorithm, a solution string is represented by a set of alleles where the number of alleles is equal to the number of cells. Each allele indicates the index, the x- coordinates and row number of the cell. The fitness function used is the reciprocal of the total half perimeter wire length for all the nets.

2.4 TS Algorithm for SCP

TS is a relatively new method. It is based on local search techniques incorporating some concepts of artificial intelligence and tries to avoid getting stuck in a local minimum. It generates and evaluates several solutions from the current solution and then chooses the best one. The TS accepts the new solution even if it is worse than the best new solution. For the implementation of TS for SCP cell placement, i.e. ordered triplets is chosen. The objective function and neighborhood operator are also the same [3]. The neighborhood operator in SA chooses two cells randomly from the sequence of cells and then interchanges their positions in the sequence. In tabu list therefore the pair of cells which was exchanged is stored as a record and for the next moves, the same two cells cannot be considered again. The algorithm stops when, for a given number of iterations, it has not found any improvement. This given number will be referred to as 'wait for change'. The number of neighbors to be tested is called loop-iteration and the length of tabu list is called Tabu-Maximum.

2.5 TA for SCP

It is based on local search algorithms and is basically a simplification of SA. It starts with an initial feasible solution and uses a neighborhood operator to obtain a new one. If new solution is better, it is accepted; if it is worse than the old solution, it is accepted with a probability which depends on the temperature value. The selection of new solution is done in a different way. Similar to temperature parameter in SA, a parameter called threshold is considered here. The value of threshold is reduced after each iteration similar to SA. In TA algorithm, the difference between the quality of new solution and the old solution i.e. values of the objective function is computed. If this difference is greater than a threshold value, the new solution is accepted. The advantage of this selection process is that one does not have to compute the exponentially as in SA and also there is no need to generate a random number for the selection as needed in SA.

We have implemented SCP using TA algorithm [3]. The solution representation chosen was same as that for SA and the other methods. Also, the objective function and neighborhood operators were the same. The threshold value, number of iterations for inner loop and maximum number of iterations were given as parameters of the algorithm.

3. MEMETIC ALGORITHM

MA combines global and local search to perform exploration while the local search methods explore high gain area of the new search [4]. Whereas exploitation concentrates on previously visited points to maximize the gain that helps determine the area for next visit. A purely random search is good at exploration whereas a purely hill-climbing method is good at exploitation. Combination of these two strategies can be quite effective, but it is difficult to know where the best balance is set. So, one of the main objectives in implementing any MA is the means of achieving both techniques during the search. It is important to understand that injecting constructive initial solutions within a population is a form of local search. Also, the concept of clustering used to smooth the landscape being searched can be considered a different form of iterative improvement embedded within the MA.

The local search algorithms use greedy rather than steepest policy and work on principle of searching a neighborhood as a means of identifying a better solution. They continue until a local optimum is found. This may take a long time. Many of the local search procedures embedded within the MAs are not standard, i.e. they usually perform a shorter truncated local search.

3.1 MA for SCP

As shown in fig. 1 the proposed MA for SCP is based on the GA discussed earlier. A tile based local search is applied on each generation [2,4]. Heuristic is performed on part of the population to improve their fitness.

For the cell placement problem, the pure GA is combined with Tile-based local search in three different ways, referred to as performing local search on part of the population: (i) before the crossover (ii) after the crossover (iii) before and after the crossover.

Space Encoding for Standard Cell Placement
(a) set size-of-popu, total_gen, generation=0;
(b) set cross-over-rate, mutation_rate;
Random generation of initial population
Evaluation of initial population
While (generation ≤ total_gen)
Apply Genetic Algorithm
Apply Tile-based algorithm to Population;
End While
Return best solution in current population.

Figure 1: The Memetic Algorithm for SCP

4. RESULTS & DISCUSSIONS

Here the results from test runs of MA are presented. Also, the results are compared with SA, GA and parallel recombinative SA, TS, TA.

The parameters of the methods were set as under:

SA: cooling factor 0.89
GA: population size: 16, cross-over rate: 0.8, mutation: 0.4, inversion: 0.01
PRSA: population size: 32, selection strategy: complete against random old individual
TA: threshold-value: 2000, loop-iteration: 50, wait for change: 1000
TS: tabu-max: 15, max-set step 50 and wait for change: 10000
MA: population size: 16, cross-over rate: 0.85 mutation: 0.4

All these methods were run for the eight different configurations given in the table 1. The table gives the results of the wire length obtained all the five techniques on the given eight different configurations.

MA, TA and TS were run severally to find the best parameter combination. The parameters mentioned above gave the best results. The wire lengths obtained are given in the table. The best results are boldfaced and worst results are underlined. It is evident from the table that in all the cases MA gave better results than TA, TS and simulated annealing.

It is evident from table 2 that SA is the fastest method followed by TA. MA is faster as compared to TS, PRSA and GA.

Table 1: Wire-lengths for different configurations by all the six techniques

Configuration	**SA**	**PRSA**	**GA**	**TA**	**TS**	**MA**
1 (20 cells)	3780	3810	3840	3750	3750	**3125**
2 (20 cells)	3000	2940	2970	2940	2940	**2295**
3 (20 cells)	3570	3600	3570	3450	3450	**3025**
4 (40 cells)	2160	2066	2102	1973	1922	**1629**
5 (40 cells)	3158	3009	3279	2961	2859	**2485**
6 (70 cells)	6232	5959	6289	5773	5641	**5120**
7 (70 cells)	6972	6799	7057	6705	6478	**5865**
8 (100 cells)	3598	3020	2509	2404	2164	**1662**

Table 2: Time taken (in seconds) by all the six techniques

Configuration	**SA**	**PRSA**	**GA**	**TA**	**TS**	**MA**
1 (20 cells)	01	929	115	68	178	132
2 (20 cells)	01	666	111	55	138	105
3 (20 cells)	01	1807	112	63	165	124
4 (40 cells)	05	876	361	93	299	228
5 (40 cells)	07	4233	364	110	291	220
6 (70 cells)	78	30011	990	361	1109	828
7 (70 cells)	68	31216	1010	449	1228	912
8 (100 cells)	104	35914	1236	308	752	570

5. CONCLUSIONS

This research paper investigated MA for solving the SCP problem at physical design stage of VLSI design cycle. MA based on GA and Tile based local search on each generation makes it faster and more efficient. Also heuristic techniques were applied on a part of population to improve its fitness. MA is further compared with TS and TA. Both were based upon local search. In addition to individual studies of the methods, we compared all the current and previous methods in terms of solution quality and computing speed in connection with the SCP problem. MA gave improved results for wire-length (15% reduction) and also computation time (25% time faster). Future scope of this work includes comparison of MA based SCP Algorithm with other Hybrid techniques and furthermore multi-objective optimization of SCP using MA and its variants.

6. REFERENCES

[1] Beumont, O., Legrand, A. and Robert, Y., "Optimal algorithms for scheduling divisible workloads on heterogeneous systems", Proceedings of The International Parallel and Distributed Processing Symposium, 2003

[2] Areibi, S., Bao, X., Grewal, G., Banerji, D., Du, P., "A Comparison of Heuristics for FPGA Placement", ACTA International Journal of Computers and Applications

[3] Bunglowala, A., Singhi, B.M. et. al, "Performance Evaluation and Comparison and Improvement of Standard Cell Placement in VLSI Design", International Conference on Emerging Trends in Engineering and Technology, July 2008 [also published on CSDL, IEEE]

[4] Bunglowala, A., Singhi, B.M., "A Solution to combinatorial Optimization Problem using Memetic Algorithms", International Journal of Computer System Applications [IJCSA], ICAC, February 2008.

[5] Casanova, H., Legrand, A., Zagorodnov, D. and Berman, F., "Heuristics for scheduling parameter sweep applications in Grid environments", Heterogeneous Computing Workshop 2000, IEEE Computer Society Press, pp. 349-363.

[6] Cohoon, J.P. and Paris, W.D., "Genetic Placement", Proceedings of IEEE International Conference on Computer Aided Design, Santa Clara, 1986, p.p. 422-425

[7] Donath, W.E., "Complexity Theory and Design Automation", Proceedings of 17th Design Automation Conference, 1980

[8] Dutt, N.D. and Gajski, D.D., "Design Synthesis and Silicon Compilation", IEEE Design and Test of Computers, 1990, p.p. 8-23

[9] "First workshop on memetic algorithms (WOMA I)," Proc. 2000 Genetic and Evolutionary Computation Conference Workshop Program, pp. 95–130, July 8, 2000

[10] Gajski, D.D., "New VLSI Tools" Guest Editor's Introduction, IEEE Computers, 1983, p.p. 11-14

[11] Garey, M.R. and Johnson, D.S., "Computers and Intractability", A Guide to the Theory of NP-Completeness, San Francisco, 1979

[12] Goldberg, D. E. and Voessner, S., "Optimizing global-local search hybrids," in Proc. 1999 Genetic and Evolutionary Computation Conf., Orlando, FL, July 13–17, 1999, pp. 220–228.

[13] Land, M.W.S., "Evolutionary algorithms with local search for combinatorial optimization," Ph.D. dissertation, Univ. of California, San Diego, 1998.

[14] Leighton, F.T., "Complexity Issues in VLSI", Cambridge, 1983

[15] Lengauer, T. "Combinatorial Algorithm for Integrated Circuit Layout", Chichester, New York, 1990

[16] Sahni, S. and Bhatt, A., "The Complexity of Design Automation Problems", Proceedings of 17th Design Automation Conference, 1980, p.p. 402-411

[17] Shahookar, K. and Mazumder, P. "VLSI Placement Techniques", ACM Computing Serveys 1991, p.p. 143-220

[18] Shewani, N.A., "Algorithms for VLSI Physical Design Automation", Boston, 1993

[19] Tagliarini, G.A., Christ, J.F. and Page, E.W., "Optimization using Neural Networks", IEEE Transaction on Computers, 1991, p.p. 1347-1358