

Performance Analysis of VLSI Floor planning using Evolutionary Algorithm

P.Sivaranjani
Assistant Professor
Department of ECE
Kongu Engineering College

K.K.Kawya
M.E VLSI DESIGN
Department of ECE
Kongu Engineering College

ABSTRACT

Floorplanning is an important physical design step for hierarchical, building-block design methodology. When the circuit size get increases the complexity of the circuit also increases. To deal with the increasing design complexity the intellectual property (IP) modules are mostly used in floorplanning. This paper presents a Hybrid particle swarm optimization algorithm for floorplanning optimization. Here B*tree is used at the initial stage in order to avoid overlapping of modules and later, PSO algorithm along with the concept of crossover and mutation from Genetic algorithm is used to get optimal placement solution. The main objective of floorplanning is to minimize the chip area and interconnection wire length. The Experimental results on Microelectronic Center of North Carolina (MCNC) benchmark circuits shows that our algorithm performs better convergence than the other methods.

General Terms

Floorplanning, Very Large Scale Integrated Circuits(VLSI)

Keywords

Hybrid particle swarm optimization (HPSO), Genetic algorithm (GA), Crossover, Mutation, Microelectronic Center of North Carolina (MCNC), Very Large Scale Integrated Circuits(VLSI)

1. INTRODUCTION

As the demand for latest technology surge in the market, more research has been carried out in the field of IC design, which in turn makes VLSI design developed with more composite, reliable, compact and better performance. As the number of transistors in a single chip is countless, the IC design has become much more complex. A more sophisticated approach to the above problem is Floorplanning, which is an important physical design step for hierarchical, building-block design methodology. When the circuit size get increases, the solution space also increases simultaneously which make it impossible to find the global solution space. The HPSO algorithm can be adopted to solve floorplanning problem.

Recently, many researchers resort to stochastic optimization algorithms based on slicing [8] and non-slicing floorplan [10,9,4]. In [10][9], bounded slicing grid and sequence pair approaches were used to represent non-slicing floorplan with smaller solution space. Both these approaches use constraint graph to manipulate the transformation between the representation and their placement. But it is more complicated. To overcome the above problem pei-Ning Guo et al.,[6] proposed an O-tree representation based on non-slicing floorplan. Experiment results shows that this representation achieves minimum wire length and very less dead space. Yun-Chih Chang et al., [3] present an efficient,

flexible and effective data structure, B*tree based Simulated annealing schemes for representing non-slicing floorplans. This representation runs about 4.5 times faster than the O-tree, and consumes about 60% less memory and also achieves near optimum area utilization even for rectilinear modules. Finally, like O-tree, the size of the search space of the floorplanning problem by the B*-tree representation is only $O(n!2^{2n-2}/n^{1.5})$ [3], while it is $O((n!)^2)$ by the sequence pair representation [14]. Hence, the B*-tree representation is adopted in this paper.

Since VLSI floorplanning is NP-hard, different heuristic methods have been proposed. They are classified into two methods: the constructive method and the iterative method. The constructive method use heuristic strategy to construct a floorplan. The bottom-left (BL) and BL fill methods [2] are the most common constructive approaches. Huang et al. [7] presented a very effective heuristic algorithm, in which two important concepts, namely, the corner-occupying action and caving degree, were introduced to guide the packing process. The iterative method use metaheuristic approach such as simulated annealing, tabu search, genetic algorithm and particle swarm optimization algorithm to produce final best solution. Jianli Chen et al. [4] proposed a new heuristic method that uses a hybrid simulated annealing to represent non-slicing floorplan. In [13], a genetic algorithm was applied to tackle the VLSI floorplanning problem. In [1] S. Anand et al., proposed a new algorithm named as Simulated Spheroidizing Annealing Algorithm (SSAA) which is more suitable for problems of larger size floorplans. In [8] W-L Hung proposed genetic algorithm based thermal-aware floorplanning. This is the first paper considered thermal-aware floorplanning but it consider only one constraint area and it obtains the final fittest solution.

The particle swarm optimization (PSO), was developed by James Kennedy and Russell Eberhart in 1995. It is an stochastic optimization technique based on the movement and swarm intelligence. PSO is initialized with the population of random solution and it applies the concept of social interaction to problem solving. Unlike most of other population-based evolutionary algorithms, PSO is motivated by the simulation of social behaviour instead of the survival of the fitness. The advantages of PSO are simplicity in implementation and its ability of convergence is very quick. (Sun et al. 2006) originally introduces PSO into the floorplanning problem. The paper adopts the B*-tree floorplanning structure (Chang et al.2000) to generate an initial stage with overlap free for placement and utilizes PSO to find out the potential optimal placement solution. However, no implementation detail of the algorithm is mentioned, and only the area optimization is considered. It is unable to solve problems that optimize the area and wire length simultaneously, thus it is not an effective PSO algorithm for

general floorplanning problems. To overcome the above problem hybrid particle swarm optimization was introduced in which both area and wire length is considered and optimized results were obtained.

The rest of this paper is organized as follows. Section 2 describes the floorplanning problem. Section 3 depicts the mechanism of B*tree representation. Section 4 describes the details of HPSO and simulation results are shown in Section 5. A conclusion is drawn in Section 6

2. PROBLEM STATEMENT

2.1 Data Structure

Let the set of blocks be $A = \{a_1, a_2, \dots, a_N\}$ where N is a number of modules. Each block a_i , where $1 \leq i \leq N$, is rectangular in shape with width W_i and height H_i . The modules are placed in such a way that no two modules overlap with each other. The aspect ratio for each module is calculated using W_i/H_i . There are two types of modules in floorplanning.

Hard module: The module's whose shape (i.e. width and height) is fixed, and is denoted as (W, H) where W is the width and H is the height of the module.

Soft module: The module's whose width (W) and height (H) can be varied as long as the aspect ratio is within the given range but the area (A) is fixed. In this paper we used hard module approach to optimize floorplan.

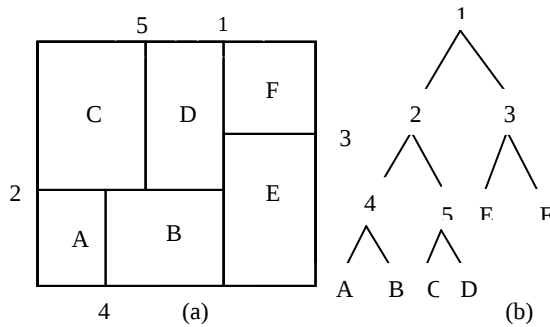


Fig 1: Slicing floorplan and its slicing tree (a) Slicing floorplan (b) Slicing tree

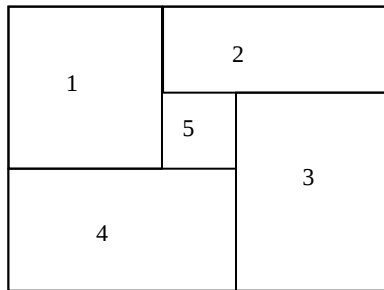


Fig 2: Non-slicing floorplan

2.2 Floorplan Structure

Floorplan can be represented in two ways, namely, slicing and non-slicing floorplan. The slicing structure can be bisected horizontally or vertically on recursive basis till each part contains only one module. Non-slicing floorplan cannot be bisected in this manner. In fig 1: (a) letter denotes modules and number denotes horizontal and vertical cut division. Figure 2 shows a non-slicing floorplan and the number in the Figure 2 denotes the number of modules in the floorplan. This paper considered non-slicing based floorplan representation.

3. B*TREE REPRESENTATION

This paper presented B* tree based representation at the initial stage to avoid overlapping of modules. The main advantage of using B*tree is it handles non-slicing structure which is flexible to deal with hard modules in $O(n)$ times. The procedure to construct B* tree is similar to the depth first search (DFS). while constructing B* tree, the first priority is given to construct the left sub tree followed by right sub tree in repetitive fashion, with its root representing the bottom left corner in the floorplan and thus the coordinate of the module is $(x_{root}, y_{root}) = (0,0)$. Let n_i be the root node, and n_j be either left or right node. If node n_j is the left child of root node n_i , module j is placed on the right hand side which is adjacent to the module i , i.e., $n_j = n_i + w_i$, where w_i is the width of the module i . similarly if node n_j is the right child of root node n_i , module j is placed above the corresponding root node with the constraint $x_i = x_j$

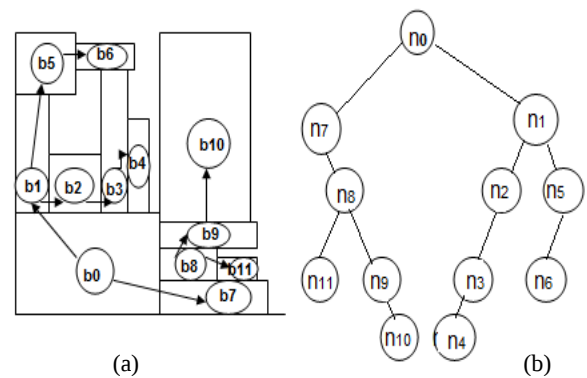


Fig 3: (a) Admissible placement (b) B*-tree representing the placement

Since $n0$ is the root node in figure 3 (b), the module $b0$ is placed on the bottom left corner. Then based on the DFS procedure $n7$ is the left child of $n0$, so the module $b7$ is placed on the right hand side of $b0$. Since there is no left sub-tree exist after the node $n7$, it starts to construct right sub-tree of node $n7$ (i.e. $n8$). This procedure continues until all the nodes in the tree are constructed in the recursive manner. The above construction takes only linear time.

4. HYBRID PARTICLE SWARM OPTIMIZATION

4.1 Particle Swarm Optimization

Particle swarm optimization (PSO) is an optimization method that optimizes a problem by iteratively trying to improve a candidate solution with regard to a given measure of quality. PSO optimizes a problem by having a population of candidate solutions (from b*tree). The individuals in the population are called as particles. By using PSO algorithm the particles are moved around in the search-space according to simple mathematical formulae over the particle's position and velocity. Each particle's movement is influenced by its local best known position and is also guided toward the best known positions in the search-space, which are updated as better positions are found by other particles. This is expected to move the swarm towards the best solutions. The velocity and position of each particle is updated according to the following equation (1) and (2).

$$V_{id}^t = W \times V_{id}^{t-1} + c_1 r_1 (P_{id} - X_{id}^{t-1}) + c_2 r_2 (P_{gd} - X_{id}^{t-1}) \quad (1)$$

$$X_{id}^t = X_{id}^{t-1} + V_{id}^t \quad (2)$$

where t is the iteration index, d is the number of dimensions (variables), w is the inertia weight, c_1 and c_2 are two positive constants, called acceleration constants, r_1 and r_2 are two random numbers within the range $[0,1]$. Equation (1) calculates a new velocity for each particle (potential solution) based on its previous velocity (V_{id}^t) and the personal best location P_{id} ; or $pBest$) which the particle has achieved so far and the global best location P_{gd} ; or $gBest$) of the population. Equation (2) updates i^{th} particle's position in solution hyperspace. Acceleration coefficient of each particle is calculated using equation (3) and (4).

$$C_1 = cIter \times (c_{1e} - c_{1s}) / MAXITER + c_{1s} \quad (3)$$

$$C_2 = cIter \times (c_{2e} - c_{2s}) / MAXITER + c_{2s} \quad (4)$$

where $cIter$ is the current iteration number and $MAXITER$ is the maximum number of allowable iteration, c_{1e} , c_{2e} represent the final value of the c_1 and c_2 and c_{1s} , c_{2s} represent the initial values of the c_1 and c_2 .

4.2 Genetic Algorithm

Genetic algorithms are computational procedure that mimics the natural process of evolution. The objective of the GA is to find an optimal solution to a problem. Inspired by the GA, a PSO algorithm with crossover and mutation operators for the floorplan problem is proposed.

4.2.1 Crossover and Mutation

Two point crossover operator [11] is used in this paper. In figure 4 two crossover points are selected randomly from particle 1. An element from starting of the particle 1 to the first crossover point is copied to the new particle and then the elements are selected randomly from particle 2 for the crossover point and copied in new particle and then the rest is copied from particle 1. For mutation operator, a element is selected randomly and its direction is changed.

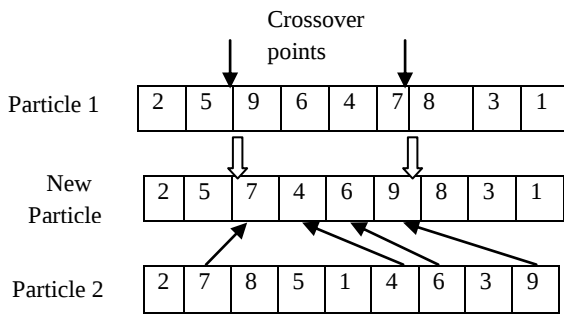


Fig 4: The crossover operator

4.3 Fitness Evaluation

In PSO each particle corresponds to a potential solution, therefore, a layout can be obtained after decoding the particle. Usually, the objective of the optima is to minimize the chip area or wire length of floorplanning. Considering the minimization of area, the fitness is equal to the value of area. Thus the fitness function for area can be represented as follows:

$$f(x) = W(x) \times H(x) \quad (5)$$

where $W(x)$ is the width of the corresponding particle x , and $H(x)$ is the height of the particle x . For multi-objective optimization the fitness function can be represented as

$$g(x) = \lambda_1 \times f(x) + \lambda_2 \times t(x) \quad (6)$$

where λ_1 and λ_2 are constant weights, $f(x)$ is the area of particle x , $t(x)$ is the wire length of particle x which is calculated by using HPWL (Half-perimeter wire length). It is defined as half-perimeter length of the smallest bounding box that encloses all pins. The HPWL of the p^{th} can be calculated according to the equation (7)

$$L_p = X_{max} - X_{min} + Y_{max} - Y_{min} \quad (7)$$

where X_{max} and X_{min} are the maximum and minimum x -coordinate of the HPWL bounding box of the net. Y_{max} and Y_{min} are the maximum and minimum y -coordinate of the HPWL bounding box of the net. The total wire length can be calculated using equation (8)

$$t = \sum_{p=1}^m L_p \quad (8)$$

4.4 Algorithm Description

STEP 1: Load modules data and initialize the parameter of the PSO algorithm

STEP 2: Generate an initial population, initialize the position and velocity of each particle, and calculate the P_{best} of each particle and the G_{best} population

STEP 3: calculate the fitness value of each particle by equation (5) and (6)

STEP 4: Check each particle, if its fitness value is better than P_{best} , update its P_{best} with the fitness value

STEP 5: Check each particle, if its fitness value is better than population's G_{best} , update its G_{best} with the fitness value

STEP 6: Adjust the position and velocity of each particle according to equations (1) and (2)

STEP 7: If termination condition is satisfied, the algorithm stops; otherwise, go to step 3.

5. EXPERIMENTAL RESULTS

5.1 Parameter Settings for Algorithm

All the cells are set as hard IP modules. The parameters of the PSO algorithm are set as follows: $W=0.95$, C_1 decreases linearly from 0.82 to 0.5 and c_2 increases from 0.4 to 0.83. The probability of crossover set as 0.85 and mutation as 0.005. The population size is set as 10 and the maximum number of generation is 10,000.

Table 1. Characteristic of MCNC benchmark circuits

CIRCUIT	MODULES	NETS	I/O PAD	PINS
APTE	9	97	73	287
XEROX	10	203	2	698
AMI33	33	123	42	522
AMI49	49	408	22	953

5.2 Performance of Algorithm

Consider the standard MCNC benchmark circuits APTE, XEROX, HP, AMI33 and AMI49. The simulation programs are implemented in MATLAB. Each benchmark circuits have standard number of modules, nets, i/o pad and pins. The characteristics of MCNC benchmark circuits [5] are shown in table 1.

The figure 5, 6, 7, 8 shows that the area and wire length are optimized simultaneously. The table 2 shows the comparison between HSA and HPSO.

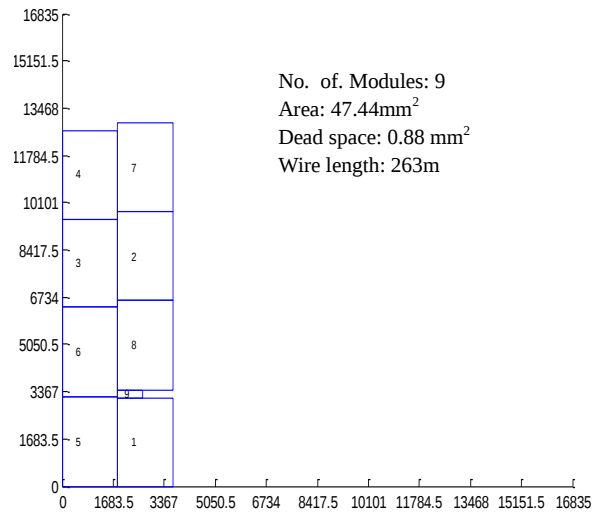


Fig 5: Simulation result on MCNC apte

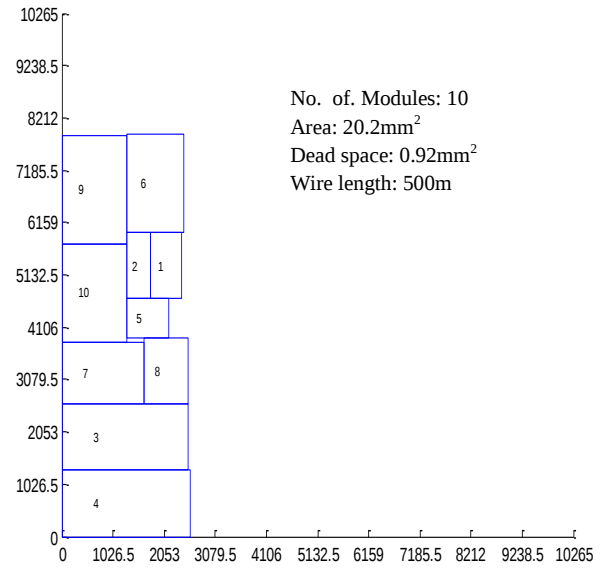


Fig 7: Simulation result on MCNC xerox

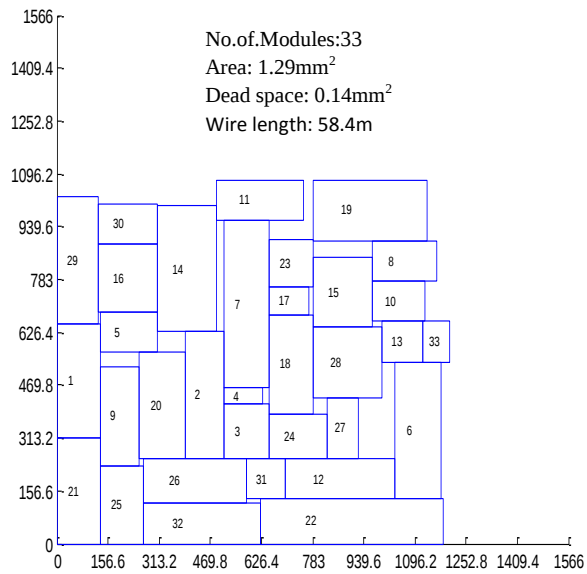


Fig 6: Simulation result on MCNC ami33

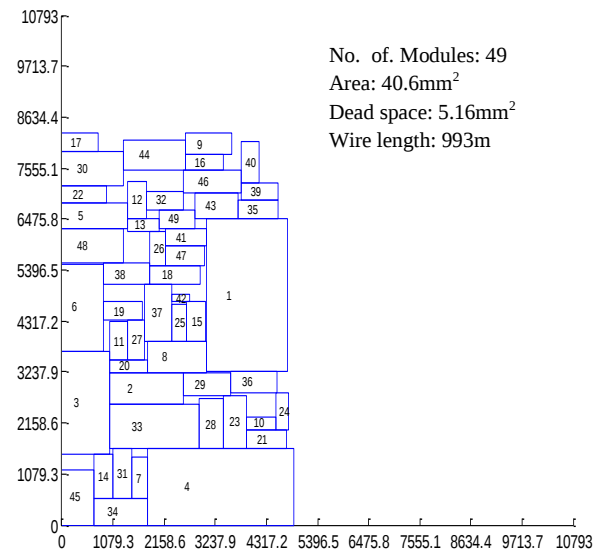


Fig 8: Simulation result on MCNC ami49

Table 2. Comparison between HSA and HPSO

Algorithm	Apte		Ami33		Ami49		xerox	
	Area (mm ²)	Wire (m)	Area (mm ²)	Wire (m)	Area (mm ²)	Wire (m)	Area (mm ²)	Wire (m)
HPSO	47.44	263	1.29	58.4	40.6	993	20.2	500
HSA	47.12	480	1.21	61.2	37.80	1020	20.89	513

From table 2, it can be seen that ,in terms of area utilization and wirelength, the proposed algorithm performs better result than HSA.In HPSO the wirelength was reduced considerably though there is small increase in area for the benchmark circuits apte,ami33 and ami49. Hence both wirelength and area has been optimized.

6. CONCLUSION AND FUTURE WORK

Very large-scale integrated floorplanning is NP-hard in combinatorial optimization problem. To solve it in an efficient way, a HYBRID PSO algorithm was proposed. Traditional PSO algorithms often shows weakness in solving discrete problems, the proposed algorithm can overcome this weakness in some degree. The experimental results for MCNC benchmark circuit demonstrated that the proposed algorithm can achieve the optimal and reasonable solution for the hard IP modules placement. The future work will focus on the parameter related to thermal-aware floorplanning.

7. REFERENCES

- [1] Anand, S. Saravanasankar, S. Subbaraj, P. 2012. Customized simulated annealing based decision algorithms for combinatorial optimization in VLSI floorplanning problem. *Comput Optim Appl* pp.667–689.
- [2] Bernard, C. 1983 The bottom-left bin packing heuristic: An efficient implementation. *IEEE Trans. Comput.*, vol. 32, no. 8, pp. 697–707.
- [3] Chang, Y-C., Chang, Y.W., Chang G.-M. Wu, and Wu, S.-W., 2000. B*-trees: a new representation for non-slicing floorplans. In *Proceedings of the ACM/IEEE Design Automation Conf.*, pp. 458–463.
- [4] Chen Jianli, Zhu Wenxing and Ali M.M 2011.A hybrid simulated annealing algorithm for non-slicing VLSI floorplanning. *IEEE Trans VLSI Syst Man Cybern C* 41(4), pp.544-553.
- [5] Guolong Chen, Wenzhong Guo, Yuzhong Chen. 2010.A PSO based intelligent decision algorithm for VLSI floorplanning. *Springer, Soft Comput* 14, pp.1329-1337.
- [6] Guo, P.N. Cheng, C.K. Yoshimura, T. 1999. An O-tree representation of non-slicing floorplan and its applications. In: *Proceedings of the 36th ACM/IEEE conference on design automation conference*, New Orleans, Louisiana, USA, pp 268–273.
- [7] Huang, W. Chen, D. and Xu, R. 2007.A new heuristic algorithm for rectangle packing. *Computers Oper. Res.*, vol. 34, no. 11, pp. 3270–3280.
- [8] Hung, W-L. Xie, Y.Vijaykrishnan, N. Addo-Quaye, C. Theodorides,T. and Irwin, M. J.2005. Thermal-Aware Floorplanning Using Genetic Algorithms. *IEEE In International Symposium on Quality Electronic Design (ISQED)*, pp.7695-7701.
- [9] Murata, H. Fujiyoshi, K.. Nakatake, S. and Kajatani, Y.1995.Rectangular-Packing-Based Module Placement *ICCAD*, pp. 472-479.
- [10] Nakatake, S. Fujiyoshi, K. Murata, H. and Kajitani, Y. 1996.Module Placement on BSG-Structure and IC Layout Applications. *ICCAD*, pp. 484-491.
- [11] Sivanandan, S.N. and. Deepa, S.N.2008. Book on Introduction to genetic algorithms. *Springer-Verlag Berlin Heidelberg*.
- [12] Sun, T.Y. Hsieh, S.T. Wang, H.M. Lin, C.W.2006. Floorplanning based on particle swarm optimization. *IEEE computer society annual symposium on VLSI*, Karlsruhe, Germany. *IEEE*,pp 7–11.
- [13] Valenzuela, C. and Wang, P.2002. VLSI placement and area optimization using a genetic algorithm to breed normalized postfix expressions. *IEEE Trans. Evol. Comput.*, vol. 6, no. 4, pp. 390–401.
- [14] Wong, D. F. and Tang, X. 2001. FAST-SP: A fast algorithm for block placement based on sequence pair.In *Proc. Asia and South Pacific Design Automation Conf.*, pp. 521–526.
- [15] The MCNC Benchmark Problems for VLSI Floorplanning. [Online].Available: <http://www.mcnc.org>.