VLSI Floorplan using Binary Particle Swarm Optimization

Rajendra Bahadur Singh

Gautam Buddha University, Greater Noida, India

Anurag Singh Baghel

Dept. of Computer Science, Gautam Buddha University, Greater Noida, India

ABSTRACT

Floorplanning is a crucial design step in the integrated circuit physical design cycle. It effects wire length, delay, area etc on IC. It determines the size, shape, and locations of modules in a chip and as such it estimates the total chip area, interconnects, and, delays. In this paper, Binary Particle Swarm Optimization algorithm has been proposed to optimize the area and wire length. To optimize the IC floorplan, two physical quantities have been considered such as area, and wire length for soft IP modules. Firstly, floorplan is designed by sequence pair representation without overlapping of the modules and then the BPSO algorithm explores the packing of all modules in floorplan to find better optimal performances i.e. area and wire length. The simulation results on Microelectronic Center of North Carolina benchmark show that BPSO algorithm gives better optimal area and wire length. The results obtained are compared with other algorithms; the area is improved maximum up to 7.5%.

Keywords

Floorplanning, non-slicing, Binary Particle Swarm Optimization, Sequence Pair, MCNC Benchmarks etc.

I. INTRODUCTION

The floor planning is the initial step of the physical configuration in the VLSI outline. It is a key configuration design to evaluate the chip range by considering the ideal arrangement of digital blocks with their interconnections. Every block comprises few hundred or a large number of cells that could be perform a particular operation. The blocks are in rectangular shape with various perspective proportions. The block can be ordered into two sorts in view of their shape adaptability one is the hard blocks and second is the soft blocks. Hard block has altered width and height while soft block width and height could be differed the length of its viewpoint proportion is inside the given range and its area is steady. The viewpoint proportion of a block is characterized as the proportion between the height and the width of a block. To improve the area of the chip, hard blocks are turned then width and height of soft blocks are changed, for example, without influencing area of the block. The traditional floor planning techniques ordinarily handles just block packing to minimize aggregate chip area; however present day floor planning strategies can be concocted as an altered framework floor planning.

Traditional floorplanning [1] [3] techniques manage the minimization of chip area by upgrading the module positions and their interconnections. In the present day physical designing of VLSI [2] [4], it is vital to outline the chip with every small size which expends less power. Keeping in mind, reduce routing difficulty of power nets, modules with indistinguishable voltage source have better to be set together.

In 2005 Tung et al [6] discuss two types of modern floorplanning problems: 1) fixed-outline floorplanning and 2) bus-driven floorplanning. The floorplanner uses B-tree representation with fast simulated annealing. For fixed-outline floorplanning, the authors present an adaptive Fast-SA that can dynamically change the weights in the cost function to optimize the wire-length under the outline constraint. Zhu et al [7] presented an efficient simulated annealing based VLSI floorplanning algorithm for slicing structure. The primary work is to find out the reason which causes dead space and then it is analyzed. After that an intuitive and fast approach is proposed to determine the reasonable orientation of each module. To generate neighborhood solution a perturbing operator of normalized polish expression is modified, and SA algorithm is used to search for the optimal floorplan solution. Jianli et al [8] proposed a hybrid simulated annealing algorithm for Non-slicing VLSI Floorplanning. To construct an initial B-tree the HSA uses a new greedy method and a new operation on the B-tree to explore the search space. In this paper the experimental results are taken on MCNC benchmarks and show that the HSA can produce optimal or nearly optimal solutions for all the tested problems very quickly.

In 2017, Zou D. et al [9] presented a memory based simulated annealing algorithm (MSA) for IC floorplanning. A new function has been formulated for the fixed outline floorplanning. MSA watches the number of successive failures, which is helpful for avoiding a plenty of blind searches. In 2017, Senthil Kumar et al. [10] proposed hybrid particle swarm optimization firefly algorithm to optimize the floorplan area and interconnect. Soft modules for MCNC and GSRC benchmarks have been simulated.

De-Xuan Zou et al [11] proposed an Improved Simulated Annealing (ISA) algorithm and area model for the fixed outline floor planning with hard modules. In case ISA meets premature convergence, it casually creates a new floorplan which is independent of the earlier one. This simple operation is very resourceful to help ISA to get rid of premature convergence. B-tree is employed in this paper to perturb a solution in each iteration. Nakaya et al [12] proposed an adaptive genetic algorithm to solve the floorplanning problem in VLSI layout design. In this paper the sequence-pair representation is accepted as the coding scheme of each chromosome. New genetic operators for the problem are presented to explore the search space efficiently.

Tang et al [13] proposed a memetic algorithm (MA) for a nonslicing and hard-module VLSI floorplanning problem. This MA is a hybrid GA that uses an effective genetic search method to explore the search space and an efficient local search method to exploit information in the search space. The exploration and exploitation are balanced by a novel bias search strategy. The MA has been implemented and tested on popular benchmark problems. Chen et al. [14] proposed a novel floorplanning algorithm based on Discrete PSO (DPSO) algorithm, in which integer coding based on module number was adopted. The principles of mutation and crossover operator in the GA are also incorporated into the proposed PSO algorithm to achieve better diversity and break away from local optima. The proposed algorithm can avoid the solution from falling into local minimum and have good convergence performance.

Abdullah et al. [15] proposed a Clonal Selection Algorithm using o-tree representation for slicing and hard-module VLSI floorplanning problem. This algorithm has been implemented and tested on common MCNC and GSRC benchmarks. Syzdykov et al. [16] proposed Ant Colony Optimization to Non-Slicing Floorplanning. There is a set of modules which are placed in non-overlapping manner on a two-dimensional rectangular plane. The author has used ant system algorithm simulation as a metaheuristic to produce possible layouts in order to reduce the total vacant area. The experimental results shown in this paper compare previous methods using ant colony algorithm in VLSI design.

In this paper, we focus on the IC floorplanning problem for soft IP modules. We proposed a binary particle swarm optimization (BPSO) algorithm with sequence pair representation of the IC floorplan to optimize the parameters i.e. area and the wire length which eventually affects the silicon cost in designing of IC. In this paper, floorplan of the ICs are initially represented by the sequence pair and finally employ the BPSO algorithm to find out the better result.

II. PROBLEM STATEMENT

VLSI floorplanning is to arrange the set of modules. Let P = {p_1, p_2, p_3 ……..p_m} be a set of rectangular modules where each block p_i ($1 \le i \le m$) has a specified widths w_i and heights h_i and N= {Net_i, $1 \le i \le n$} be a set of nets, where each net is a subset of modules, describing the connection between the modules. The issue is to pack every one of the modules into fixed outline floorplan region; with the end goal that they meet the accompanying conditions such as minimal area, minimal wire length [17], and no modules should overlap each other etc.

When modules are going to place in given fixed outline region then there are two constraints have to focus:

1. All modules must be packed inside given fixed outline
2. Two modules never overlap each other.

When area and wire length both are minimized simultaneously then fitness function can be calculated as

$$Cost(F) = \alpha\left(\frac{Area(F)}{Area*}\right) + (1-\alpha)\left(\frac{Wirelength(F)}{Wirelength*}\right) \quad (1)$$

Where, Area(F) is area of smallest rectangle enclosing all modules. Wirelength(F) is interconnection cost between modules. Area*, and Wirelength* is estimated minimum area, and minimum wire length respectively. Where, α is a constant weight, $0 \le \alpha \le 1$.

III. FLOORPLAN REPRESENTATION AND ALGORITHM DESCRIPTION

In this section, firstly representation of floorplan using sequence pair has been discussed. Then BPSO algorithm has been presented for soft modules to optimize the two parameters here i.e. area and the wire length. Concept of perturbation has been also discussed to find better optimal results.

(A) Floorplan Representation using Sequence Pair:

A primary research problem in the floorplanning is its representation. Various authors proposed different- different representation scheme that is corner block list, bounded slicing grid, o-tree, b-tree, sequence pair etc. These representation schemes are used to determine the lower left coordinates of each module before finally placing them in fixed outline region. We used sequence pair representation scheme.

The concept of sequence pair scheme is presented by Murata et al. [5] to encode VLSI floorplanning. In this representation scheme, pair of sequences of m elements is used, where m denotes the number of modules. Assume (π+, π-) denotes a sequence pair where π+ represents a positive sequence pair and π- represents a negative sequence pair. Let two modules p and q are present in both π+, π-. Modules p and q are placed in fixed outline region on the order of its sequence in both π+, π-.

Lemma1:

a)Module p will be placed left to module q if module p is before module q in both sequence pair π+, π-.
b) Module p will be placed right to module q if module p is after module q in both sequence pair π+, π-.
c)Module p will be placed above module q if module p is before module q in π+ and after in π-.
d) Module p will be placed below module q if module p is after module q in π+ and before in π-.

Two constrained graphs namely, Horizontal Constraint Graph (HCG) and Vertical Constraint Graph (VCG) are then constructed. HCG designed based on lemma 1(a) and 1(b) and From HCG we compute the x_i coordinate of p_i modules. Similarly, we construct the VCG based on lemma 1(c) and 1(d). From VCG we compute the y_i coordinate of p_i modules. Merging each x_i, and y_i coordinates of the p_i modules, it represents the lower left coordinate of each module (p_i, $1 \leq i \leq m$) in floorplan region.

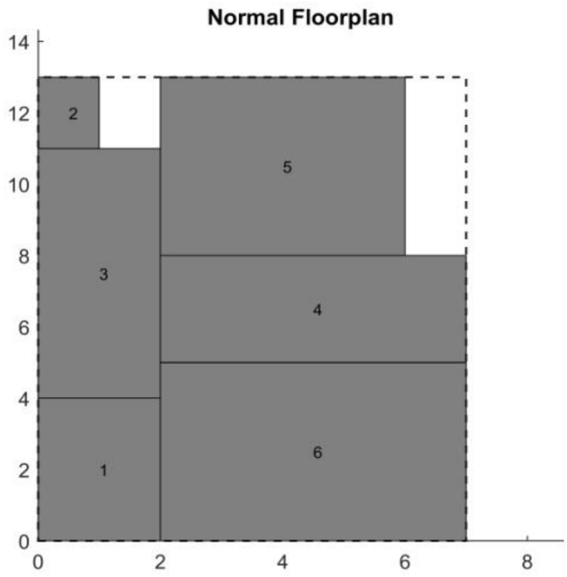

Fig.1. Floorplan for sequence pair (Γ+, Γ-) = {(2 3 1 5 4 6),(1 3 2 6 4 5)}

The width and height of the IC floorplan are determined by the longest path length in HCG and VCG, respectively. For sequence pair (Γ+, Γ-) = {(2 3 1 5 4 6), (1 3 2 6 4 5)}, final placement of all modules is shown in figure 1. White space in figure 1 is representing the unused area that is not covered by any modules.

(B) Particle Swarm Optimization:

Particle Swarm Optimization (PSO) is a population-based meta-heuristic algorithm, originally proposed by Kennedy and Elberhart. It is a population-based optimization technique animated by human social behavior, birds flocking and fish schooling. Large variants of PSO have been successfully applied to many optimization problems. This algorithm is induced by nature-inspired behavior such as birds flocking and got inspired by its collective, collaborative and self-learning behavior. Here each bird termed as a particle which represents a potential solution for optimization problem and group of birds are termed as swarm or population.

Every particle represents a potential solution which is expressed by two elements of a particle in the search space. These two elements are velocity and position. In every iteration, all of particle position and velocity are updated. Updating is not only depended on particles own personal experience, but also on its neighboring experience. The best position of each particle is represented by pbest and best position among all of the particles is represented by gbest. On this two pbest and gbest, PSO search for the better optimal solution in the search space by updating the position and velocity of each particle according to the following equations:

$$v_{id}^{t+1} = w * v_{id}^{t} + c_1 * r_1 * (p_{id}^{t} - x_{id}^{t}) + c_2 * r_2 * \left(p_{gd}^{t} - x_{id}^{t}\right) \qquad (2)$$

$$x_{id}^{t+1} = x_{id}^{t} + v_{id}^{t+1} \qquad (3)$$

$$w = (w_{min} - w_{max}) * \frac{maxiter-iter}{maxiter} + w_{max} \qquad (4)$$

Where t is the number of iteration, i is the number of the particle, d is the dimension of the particle, w is inertia weight, c_1 and c_2 are acceleration coefficients, r_1 and r_2 are two random numbers within the range [0, 1]. Further, p_{id}^{t} is the personal best location of the ith particle at the tth iteration and p_{gd}^{t} is the global best location in the population after the tth iteration in the dth dimension, v_{id}^{t+1} and v_{id}^{t} are updated velocity and previous velocity of ith particle at the tth iteration. Similarly, x_{id}^{t+1} and x_{id}^{t} are updated position and previous position of ith particle at the tth iteration. Maxiter and iter represent the maximum number of iteration and current iteration respectively. This algorithm stops if it is reached to given number of iteration or desired fitness value is achieved. The update of each particle is maintained according to equation 2 and 3.

(C) Binary Particle Swarm Optimization (BPSO):

The steps of algorithm can be described as follows:

Step1**:** load MCNC benchmark data and initialize the parameters of BPSO algorithm.

Step2: Generate randomly initial population and initialize the position and velocity of each particle.

Step3: Determine the fitness value of each particle by equation 1.

Step4: Compare the fitness value of each particle from its previous pbest, if it better then update the pbest.

Step5: Compare the fitness value of each particle from global best gbest, if it better then update the gbest to find the optimal result.

Step6: Update the velocity and position using equation 2 and 3 to find the better optimal result.

Step7: If termination condition is satisfied then stop otherwise go to step 3.

Conventional PSO is not so efficient if the problem is discrete nature. In the optimization of IC floorplanning, individual modules represent discrete nature data, so the BPSO algorithm is proposed to overcome the limitation of conventional PSO. Kennedy and Eberhart have implemented the BPSO algorithm which allows the algorithm to search in binary discrete problem spaces. It uses the concept of the velocity probability component to take between 0 and 1. In the BPSO, velocity is updated which remains unchanged, but for updating the position is re-defined by the rule.

$$x(t+1) = \begin{cases} 0 & if\ rand() \geq S(v(t+1)) \\ 1 & if\ rand() < S(v(t+1)) \end{cases} \qquad (5)$$

Where $S(\cdot)$ is the sigmoid function for transforming the velocity to the probability as the following expression:

$$S\left(v(t+1)\right) = \frac{1}{1+e^{-v(t+1)}} \qquad (6)$$

Where $\mathrm{rand}()$ is the pseudo-random number which selected from a uniform distribution over [0, 1]. In this paper, BPSO optimization is used to minimize the objective. The search space or dimension of BPSO is taken as 2 which decide the perturbation of randomly selected blocks. Four different perturbations are performed on the basis of the outcome of BPSO in these two dimensions. These are:

1. Rotation of block at right angle when bits in both the dimensions are 0, 0.

2. Randomly swapping of two blocks in π- when bits in both the dimension are 0, 1.

3. Randomly swapping of two blocks in π+ when bits in both the dimension are 1, 0.

4. Swapping both π+ and π- simultaneously when bits in both the dimension are 1, 1.

IV. RESULTS AND ANALYSIS

The optimization of IC floorplanning has been implemented for MCNC benchmarks using BPSO algorithm with SP representation in MATLAB 17A programming language, and the experiment was executed on an Intel (2.4GHz, 4GB RAM) machine running 62-bit window 2007. The proposed algorithm is tested on Microelectronic Centre of North Carolina (MCNC) benchmark circuits. All the modules are taken as soft IP modules.

The parameters selections [19] of BPSO algorithm are set as follows: population size 50; maximum iteration 100; acceleration coefficients c_1 & c_2 are 2.0; minimum and maximum velocity -6 and 6; and w_{min}, w_{max} are 0.95, 0.99 respectively.

In the experimental result, total wire length and the area are optimized simultaneously by equation 4. The weight constant α set to 0.5 because we given equal importance to both parameters area and wire length. Tested every benchmarks 10 times and the best results are compared with other algorithms results as shown in Table1.

Table1: Comparison of result (α = 0.5)

Algorithm	Apte		Xerox		Hp		Ami33		Ami49	
	Area	Wire	Area	Wire	Area	Wire	Area	Wire	Area	Wire
GA [18]	46.9	191	20.2	500	9.85	68.3	1.29	46.2	39.5	912
PSO [20]	47.31	263	20.2	477	9.5	136	1.28	69	38.8	880
Our Algo	46.98	677	19.8	606	9.11	230	1.27	108	38.55	1480

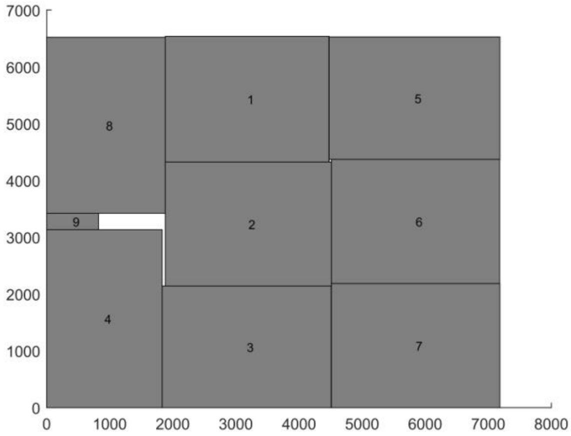

Fig. 2. Simulation result on MCNC Apte

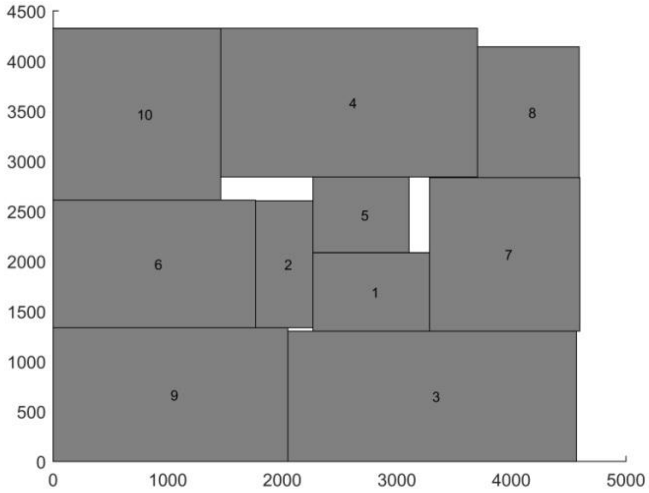

Fig. 3. Simulation result on MCNC Xerox

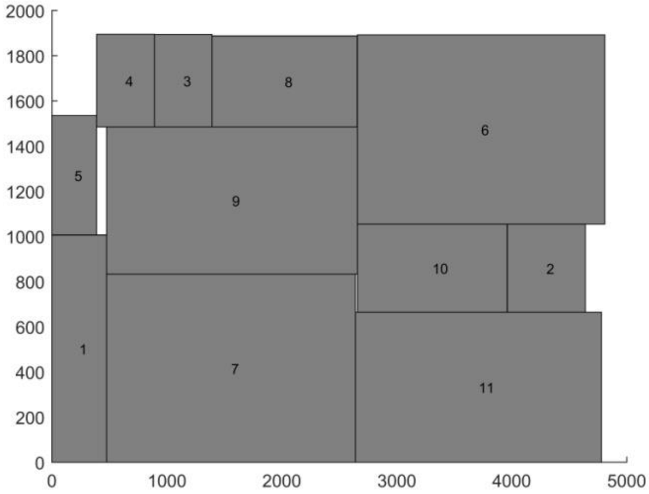

Fig. 4. Simulation result on MCNC HP

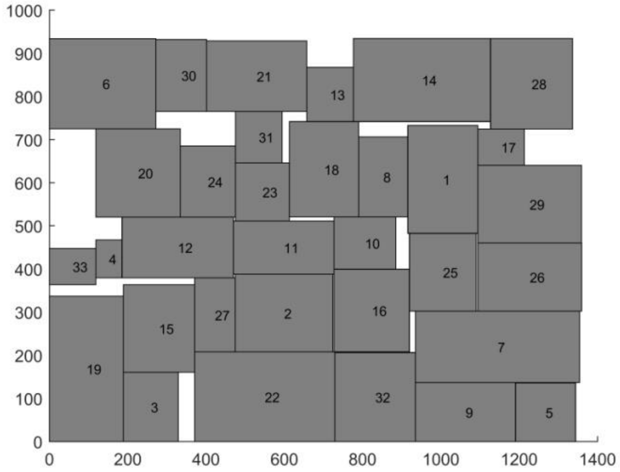

Fig.5. Simulation result on MCNC Ami33

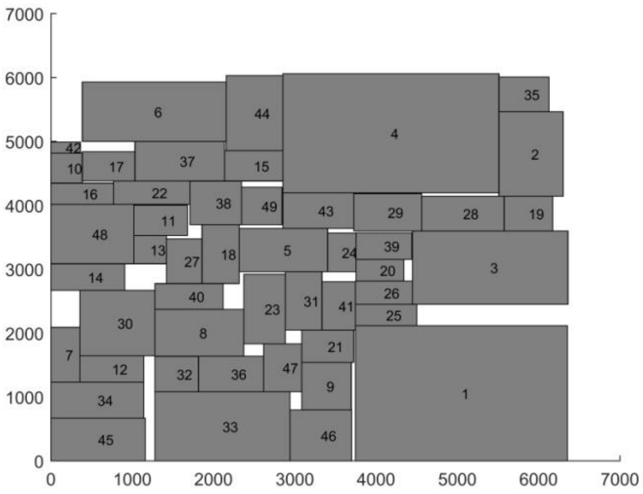

Fig. 6. Simulation result on MCNC Ami49

From Table1, it can be seen that the proposed BPSO algorithm gives better results than other algorithms in terms of area especially. After comparison as shown in Table1, we observed area is improved maximum up to 7.5 %. From figure 2 to figure 6 show the simulation result of MCNC benchmarks.

V. CONCLUSION AND FUTURE SCOPE

In this paper, we proposed BPSO algorithm with sequence pair representation to optimize the VLSI floorplanning problem for soft IP modules in efficient manner. The experimental results for MCNC benchmark circuits demonstrated that the BPSO algorithm can able to achieve the optimal result for soft modules placement. In future, we can work on 3D IC floorplan, thermal aware of Floorplan and other new meta-heuristics algorithm to find better result.

REFERENCES

[1] Alpert Charles J., Dinesh P. Mehta, and Sachin S. Sapatnekar, "Handbook of algorithms for physical design automation," Auerbach Publications, 2008.

[2] Vasant Pandian, ed. Handbook of research on modern optimization algorithms and applications in engineering and economics, IGI Global, 2016.

[3] S. K. Lim, "Practical problems in VLSI physical design automation," Springer Science & Business Media, 2008. .

[4] Singh Rajendra Bahadur, Anurag Singh Baghel, and Ayush Agarwal, "A review on VLSI floorplanning optimization using metaheuristic algorithms," In Electrical, Electronics, and Optimization Techniques (ICEEOT), International Conference on, pp. 4198-4202, IEEE, 2016.

[5] Murata, Hiroshi, et al. "VLSI module placement based on rectangle-packing by the sequence-pair." Computer-Aided Design of Integrated Circuits and Systems, IEEE Transactions on 15.12 (1996): 1518-1524.

[6] Chen, Tung-Chieh, and Yao-Wen Chang. "Modern floorplanning based on fast simulated annealing." Proceedings of the 2005 international symposium on Physical design. ACM, 2005.

[7] Lichen, Zhu, et al. "An efficient simulated annealing based VLSI floorplanning algorithm for slicing structure." Computer Science & Service System (CSSS), 2012 International Conference on. IEEE, 2012.

[8] Chen, Jianli, Wenxing Zhu, and M. M. Ali. "A hybrid simulated annealing algorithm for nonslicing VLSI floorplanning." Systems, Man, and Cybernetics, Part C: Applications and Reviews, IEEE Transactions on 41.4 (2011): 544-553.

[9] Zou, Dexuan, Gai-Ge Wang, Arun K. Sangaiah, and Xiangyong Kong, "A memory-based simulated annealing algorithm and a new auxiliary function for the fixed-outline floorplanning with soft blocks," Journal of Ambient Intelligence and Humanized Computing, pp. 1-12, 2017.

[10] Sivaranjani P., & A. Senthil Kumar, "Hybrid Particle Swarm Optimization-Firefly algorithm (HPSOFF) for combinatorial optimization of non-slicing VLSI floorplanning," Journal of Intelligent & Fuzzy Systems, 32(1), pp.661-669, 2017.

[11] Zou, De-Xuan, et al. "An Improved Simulated Annealing Algorithm and Area Model for the Fixed-Outline Floorplanning with Hard Modules." Computational and Business Intelligence (ISCBI), 2015 3rd International Symposium on. IEEE, 2015.

[12] Nakaya, Shingo, Tetsushi Koide, and Shin'ichi Wakabayashi. "An adaptive genetic algorithm for VLSI floorplanning based on sequence-pair." Circuits and Systems, 2000. Proceedings. ISCAS 2000 Geneva. The 2000 IEEE International Symposium on. Vol. 3. IEEE, 2000.

[13] Alupoaei, Stelian, and Srinivas Katkoori. "Ant colony optimization techniques for macrocell overlap removal." VLSI Design, 2004. Proceedings. 17th International Conference on. IEEE, 2004.

[14] Chen, Guolong, et al. "VLSI floorplanning based on particle swarm optimization." Intelligent System and Knowledge Engineering, 2008. ISKE 2008. 3rd International Conference on. Vol. 1. IEEE, 2008.

[15] Abdullah, Deen Md, et al. "VLSI floorplanning design using clonal selection algorithm." Informatics, Electronics & Vision (ICIEV), 2013 International Conference on. IEEE, 2013.

[16] SYZDYKOV, Mirzakhmet, and Madi UZBEKOV. "Ant Colony Optimisation Applied to Non-Slicing Floorplanning." Olympiads in Informatics 9 (2015).

[17] Sassone P. G., & Lim S. K., “Traffic: a novel geometric algorithm for fast wire-optimized floorplanning,” IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, 25(6), pp.1075-1086, 2006.

[18] Tang M., & Sebastian A., “A genetic algorithm for VLSI floorplanning using O-tree representation,” In Workshops on Applications of Evolutionary Computation (pp. 215-224) Springer, Berlin, Heidelberg, 2005.

[19] Shi Y., & Eberhart R. C., “Parameter selection in particle swarm optimization,” In International conference on evolutionary programming Springer, Berlin, Heidelberg, pp. 591-600, 1998.

[20] Chen G., Guo W., & Chen Y., “A PSO-based intelligent decision algorithm for VLSI floorplanning,” Soft Computing, 14(12), 1329-1337, 2010.