International Journal of Technical Innovation in Modern Engineering & Science (IJTIMES)
Impact Factor: 5.22 (SJIF-2017), e-ISSN: 2455-2585
Volume 5, Issue 08, August-2019

Analysis on different algorithms used for optimizing the dead space

Ganugaphati Venkata Sai Mohan[1], Venkat Rao Ganjanaboyina[2]

[1] *Electronics and Communication Engineering, LakiReddy Bali Reddy College of Engineering, Mylavaram, INDIA.*
[2] *Assistant Professor, Electronics and Communication Engineering , LakiReddy Bali Reddy College of Engineering, Mylavaram, INDIA.*

Abstract— Floor plan is the main step in physical design for build an Integrated circuit. The goal of floor plan is to optimize the area, interconnection between the modules and minimize the dead space. In latest technologies of vlsi, one of the major problem is more dead space in the design. There are various different algorithms used to minimize the dead space in top level as well as block level to manufacture an integrated circuit chip. In this paper, we analysis on different algorithms used to minimize the dead space for a good floor plan.

Keywords— vlsi, floor plan, dead space, physical design.

I. INTRODUCTION

After getting all input files from synthesis, we start an physical design stage. Physical design is to implement layout of the design. In physical design, we perform various stages as floor plan, placement, clock tree synthesis and routing. Floor plan is the first step for physical design. If we do floor plan in a good manner, it is very easy to further steps. If we do floor plan in a bad manner, it causes problems such as timing, congestion and routing problems. In floor plan, we specify core area, die area and core to IO space in the design. Macro placement is the main step while doing in the floor plan. In macro placement stage, we place macros inside of the core. Floor plan will do in both top level and block level also. In top level, tool will follow every module as in rectangular shape and it will meet conditions as no module overlap with each other and parallel to the coordinate axes. After specifying all dimensions of die and core we will do macro placement. Tool will follow default algorithms for macro placement. There are various different algorithms used for placing macros in the design. In top level, foundry people check among all floor plan algorithms which has to meet all constraints to get result in good floor plan. We will follow the algorithm which has minimize area, less number of iterations and minimize the dead space.

II. FLOOR PLAN CONSTRAINTS

We have to mention the constraints in floor plan stage such as core height, width, utilization, aspect ratio, die area, core to IO spacing, dead space and floor planning cost.
Aspect Ratio : The horizontal routing resources to the vertical routing resources or simply height to the width in the design. It decides the shape of the core.
Utilization : The sum of standard cell area and macros area within the core area. It is in percentage.
Area : product of width and height.
Dead space: The area which has unused i.e. not occupied any instances in that area is known as dead space. It is in percentage.

$$\text{Dead space} = \frac{floorplanarea - sumofallmacrosarea}{floorplanarea}$$

Floor planning Cost : The floor plan which has minimal area and minimal interconnections then floor planning cost will be minimum. We have to arrange all the modules with a minimum area for optimizing this floor plan cost.

III. FLOOR PLAN NOTATIONS

There are mainly two floor plan layout structures in the design. They are Slicing and Non-slicing floor plan. Slicing floor plan is nothing but a rectangular dissection that can be obtained by repeatedly splitting rectangles by horizontal or vertical lines into smaller rectangles. Once we dissect the rectangle, the floor plan will divide into two smaller rectangles. Slicing structure actually tells how to slice an floor plan repeatedly until we get the basic blocks. At every stage, we have one slice cut either horizontal or vertical slice. Every node represents either horizontal or vertical lines. Each leaf is a basic block. A third kind of node called wheel which appears for non-sliceable floor plan. A non-sliceable floor plan has obtained by repeatedly subdividing alone as wheel.

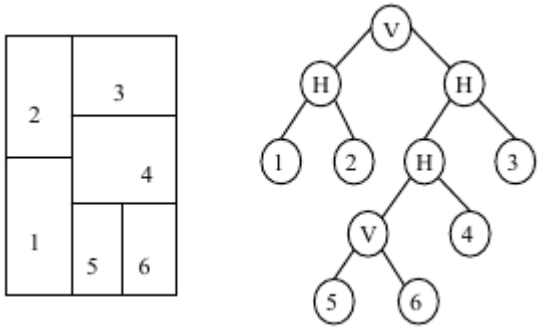


Fig.1.Slicing Floor plan& its corresponding tree structure

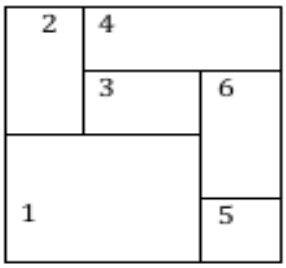
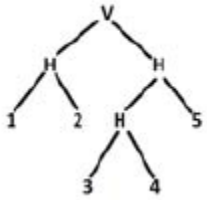


Fig.2.Non-Slicing Floor plan

A slicing tree can be represented in a compact by a polish expression or a postfix expression. In polish notation, V is the vertical cut and H is the horizontal cut.



12H34H5HV

Fig.3.Tree structure with polish notations

IV. FLOOR PLAN ALGORITHMS

In this paper, we analyze the various algorithms used for a good floor plan in the design. The algorithms such as genetic algorithm, particle swarm optimization (PSO) algorithm and simulated annealing algorithm. We have to find out among all algorithms which has the best approach for an good floor plan.

Genetic algorithm : One of the best approach for a good floor plan is genetic algorithm. It is also take from general scenario i.e. we have to initialize the population i.e. here population is nothing but initial solutions which are randomly generated. These solutions are evaluated by fitness function. After calculate fitness values we performed crossover and mutuation. If criterion reached that solution as best optimal solution. Otherwise again we select solution and performed crossover and mutuation and evaluate fitness values. This procedure will continued for number of iterations for find out an best optimal solution. The initial population i.e. solution contains random permutations of records which has rectangle. We have to calculate length and corresponding probabilities from poisson distribution. In this algorithm, there are mainly two stages i.e. mutuation and crossover. Before these, we have to select the solution i.e. selection. Selection is used to define the solution which has preserve for further reproductivity. Example for this selection is wheel selection. Crossover as previously we have existed solution so crossover creates an new solution from these existed ones. Example for this crossover is n-point crossover. Mutuation has three steps i.e. first one is we have to swap the positions of an rectangles, second one is switch the optype flag i.e. + to * or * to + and at last by increment ot decrement we mutuate the length with equal probabilities. In this mutuation, it introduces an novelity in the solution by during these steps in the algorithm. Example for this mutuation is binary mutuation. So we have to follow these steps for finding an optimal solution space in the design. If criterion reached in any iteration that solution is the best optimal solution. It gives optimal solution but takes more number of iterations for giving an optimum solution.

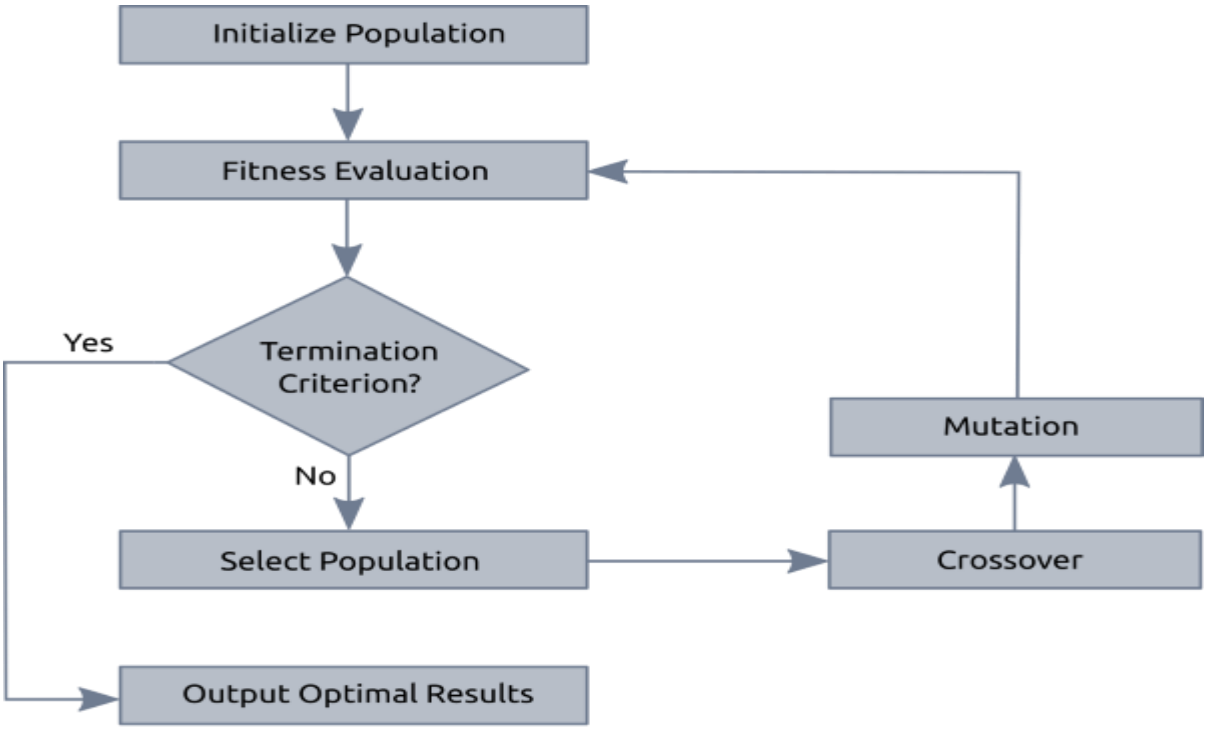


Fig.4.Genetic algorithm

Particle Swarm Optimization algorithm : This algorithm follows general scenario i.e. a group of birds are randomly searching food in a particular area. Only one bird food available in that area so all birds searched for that bird. All the birds don't know where the food is located in a particular area. But they know how the search for food is in each iteration. So, they have to find the best strategy to find the food as shown in Fig.5.

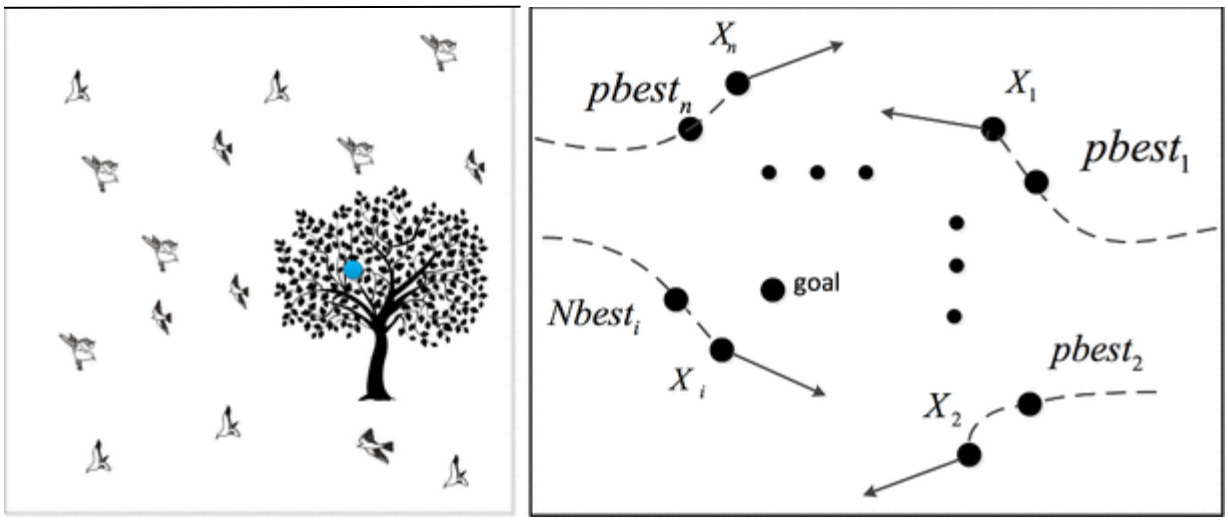


Fig.5.Birds scenario and its corresponding best solution

Particle swarm optimization algorithm follows this scenario to solve optimization problems in the design. Here, each solution is a bird in the search space we call it as particle. We have to initialize all the particles and corresponding random positions and their velocities. Here p is the particle position, pbest is the best particle, gbest is the global best particle and nbest is the next best particle. After initialize, we have to check particle fitness function is better than fitness function of best particle is it true then update present particle as the best particle otherwise all particles has been evaluated. If present particle has best particle we have to check fitness better than global best if it as yes, then update global best is the present particle. If particles not evaluated or present particle fitness function is not better we have to again set particle positions i.e. initialization. If all particles evaluated then update all particles velocities and positions. After that we have to check update all particles or not. If complete, then we have to check the best solution is optimized

or not. If not, again we will do same process until get optimized. This procedure was applied for a good optimal solution in the design. The parameter involved in this algorithm as velocity. This algorithm is the one of the best approach for finding an optimal solution space in the design. The procedure of particle swarm optimization algorithm as shown in the Fig.6.

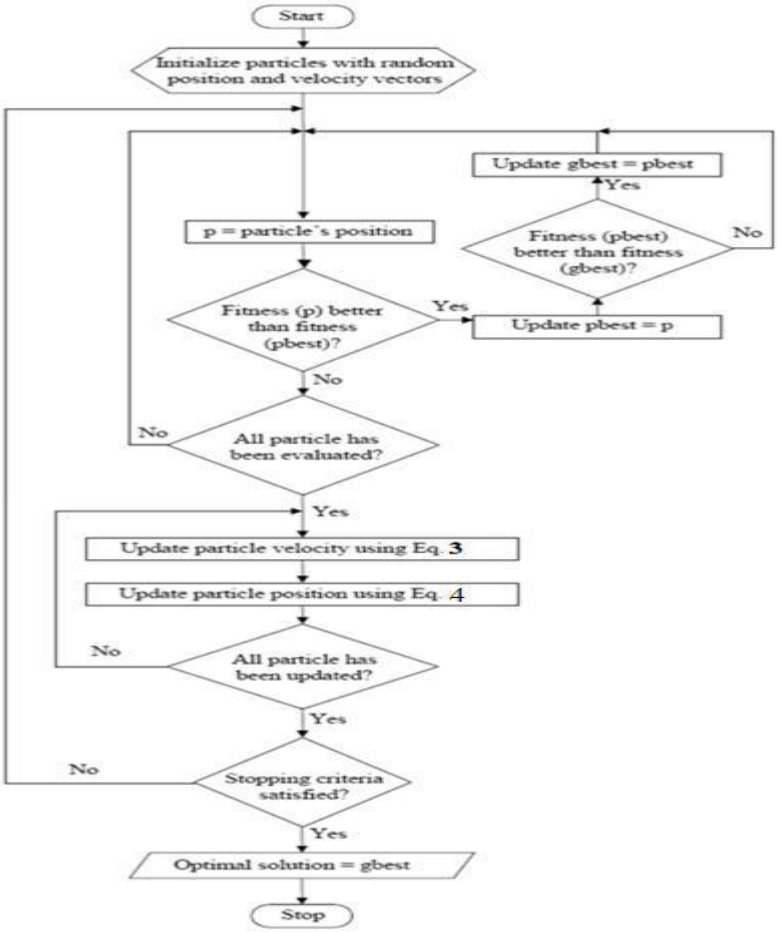


Fig.6.Particle Swarm Optimization algorithm

Simulated Annealing algorithm : The another algorithm is Simulated annealing algorithm. It is the best approach for finding an optimum solution in the design. It is inspired from the process of annealing in metal work. This metal work used for obtaining low energy states of a solid metal. The parameter used in this algorithm is temperature. The temperature of a solid metal is increases until it melts in a heat bath. The metal is then cooled until the particles are rearranged in the ground state of the solid by decreasing the temperature. The physical properties of the metal will change because its internal structure changes. This will happen if the maximum temperature is high enough and its decreases slowly. This idea will be implemented in this algorithm. We will have a temperature variable to simulate the heating process. We will assign it a high initial value and then slowly decreases it i.e. cooling as the algorithm runs. The solutions in the problem as an states in a physical system. The cost of solution is an energy of a state. In the algorithm, we will first set the initial temperature and create a random initial solution. After we get initial solution, we have to initialize the present solution as best solution and update the temperature. Then we have to check the system has cooled or not. We will select a neighbour by making a small change to our current solution. Then we decide that whether to move to that neighbour solution by particular distance. Finally, we decrease the temperature and continue looping.. We will accept solutions which are worst than our current solution as long as this temperature variable is still high. By doing this, we allow the algorithm to jump out of any local optimum it goes to early while it is executing. The chance of accepting worse solutions will reduce as the temperature decreases. We focus on the area of the search space in which a close to optimum solution can be found. This is an overview of an algorithm which is used to find out a best solution space in the design. By using this algorithm, we will reduced number of iterations and minimize the dead space area. The advantage of this algorithm is to avoiding the problem of getting stuck or caught in a local optimum and also finding a good solution for a good floor plan in the design. The simulated annealing approach is as shown in the Fig.7.

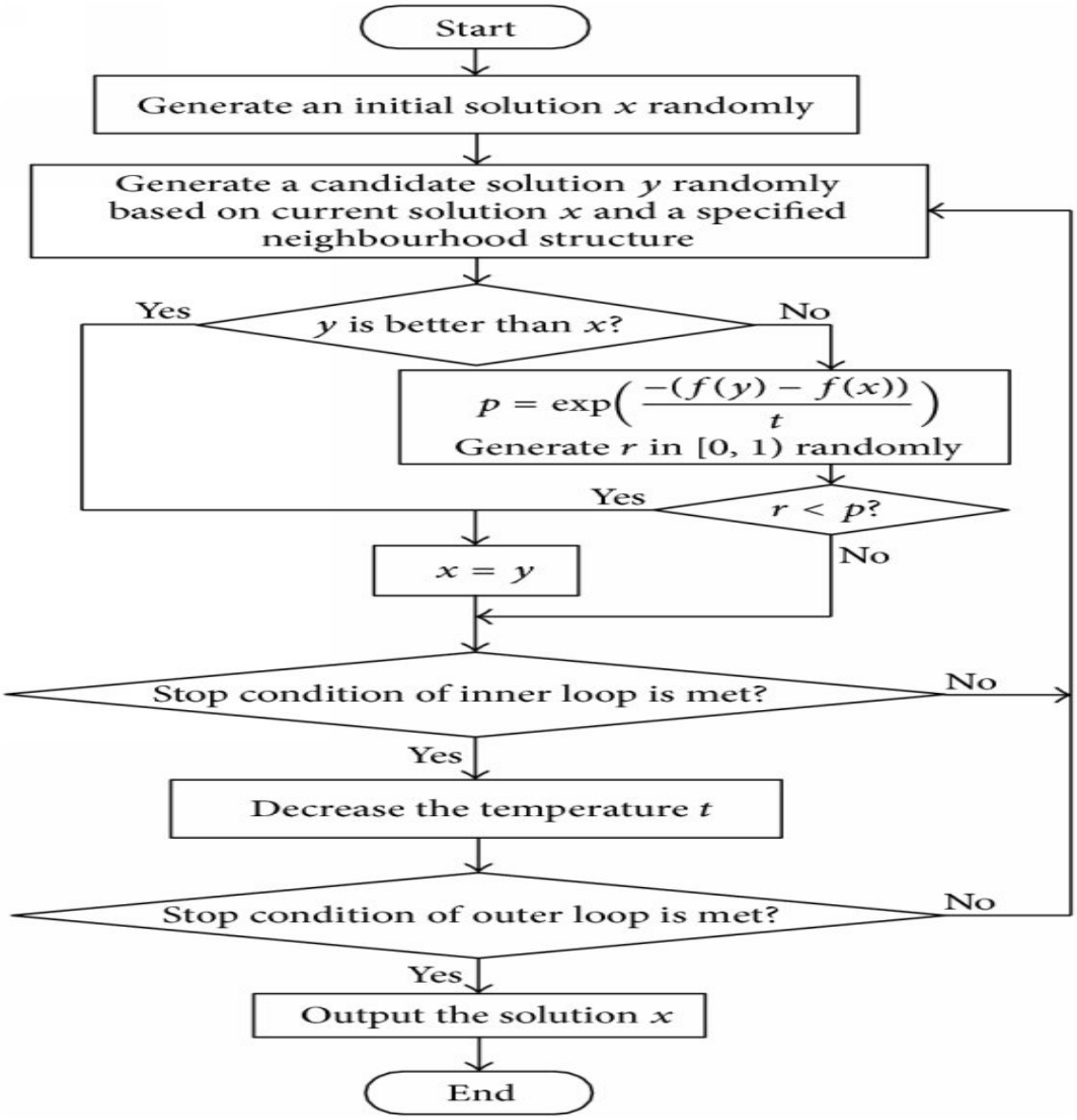


Fig.7.Simulated Annealing algorithm

V. CONCLUSION

There are different algorithms are used in top level for minimizing dead space and a good floor plan stage. Among all algorithms, we analyze these three algorithms and also find out which algorithm as best approach. These algorithms will implement on various benchmarks in top level such as apte, xerox, ami33 etc. Comparing to particle swarm optimization algorithm and genetic algorithm, simulated annealing algorithm gives less dead space for a good area optimization and also minimize interconnections between the modules. It is the best approach for a good floorplan in the design.

REFERENCES

[1] Rajendra Bahadur singh,Anurag singh Baghel, "*Dead Space reduction of floorplan using simulated annealing algorithm*", 2017 International Conference on energy, communication, data analytics and soft computing. **DOI:**10.1109/ICECDS.2017.8389822

[2] Singha, T.H.S.Dutta and M.De "Optimation of floorplanning using Genetic algorithm", Procedia technology4(2012)

[3] C.H.Chang, Y.W.Chang, G.M.Wu and S.W.Wu "*B* trees: A new representation for non-slicing floorplans",* Design automation conference.pp458-463 2000.

[4] Leena jain and amarbhir singh," *Non-slicing floorplan representations in vlsi floorplanning: A summary"* International journal of computer applications vol-71, no-15,june 2013.