Free Area Estimator for Simulated Annealing of VLSI Floor Plans

Ashwini Baligatti, Ashwini Desai ,Uday Wali

***Abstract*—VLSI floor planning problem encounter large dynamically changing decision trees and is known to be a NP Hard problem. Near optimal results can be obtained by non-deterministic methods such as simulated annealing. Annealing can be simulated as a stochastic process with several known and un-known variables in the process. Estimation of area for movement of individual modules at a given temperature depends on many physical parameters assigned to individual modules. In such a scenario, estimating free area within which a module can move at a given temperature is the prime factor that decides behavior of simulated annealing. We have implemented a new '*free area estimator*' that helps improve performance of simulated annealing algorithms. Some of the simulation results have been reported.**

***Index Terms* — Floorplan, Simulated Annealing, Temperature, Free area estimation**

I. INTRODUCTION

The physical design is a process wherein the circuit components with their respective geometries are mapped into layout of the circuit. The physical design of VLSI circuits is initiated with an important stage known as floor plan design which deals with the placement of set of rectangular areas allocated to specific circuit modules which are to be included on the chip. Each module consists of several thousand cells, each cell performing arithmetic and logical operations. The goal of floor plan is to minimize the total area of the chip. The total floorplan area includes combination of area of component and area for interconnect.

Simulated annealing is one of the most popular methods for floorplan optimization. This process is iterative in nature and generates a new floorplan area for each iteration. The floorplan with minimal area while retaining the functionality of each module is finally accepted. For each iteration a new floorplan is obtained. If the newly generated floorplan improves the configuration i.e decreases the area then the move is accepted. Otherwise, the new configuration is accepted conditionally. The probability of acceptance, in such case, depends largely on the simulated temperature and module parameters.

The process is implemented in C# on Microsoft .NET framework.

Manuscript received July 23, 2014.

Ashwini Baligatti, Electronics and Communication, KLE Dr. MSSCET, Belgaum, India, 9986882865, ashwini.baligatti@gmail.com.

Ashwini Desai, Electronics and Communication, KLE Dr. MSSCET, Belgaum, India, 9448230615, ashwini_pri@yahoo.co.in.

Dr. Uday Wali, Electronics and Communication, KLE Dr. MSSCET, Belgaum, India, 9972638499, udaywali@gmail.com.

II. OVERVIEW OF SIMULATED ANNEALING PROCESS

Simulated annealing is an iterative method which iterates continuously until the floorplan cannot be improved further.

Simulated annealing algorithm takes an existing solution, and then a series of random moves are applied to generate new configuration. Each move is accepted or rejected based on energy function, which is calculated for each new trial configuration. The minimum of the energy function correspond to the possible solutions. The best solution is the global minimum. One advantage of simulated annealing process as compared to other interchange strategies is it accepts both the moves that either increase or decrease the floorplan area, but the probability of accepting the moves that increase the area of floorplan depends on temperature.

The annealing process starts at higher temperature and temperature value id decreased gradually by some fixed ratio. The algorithm operates in two nested loops. In the outer loop, the value of T is lowered from an initial higher value, where it allows both uphill and downhill moves to be accepted. As the temperature decreases to a lower value then no uphill moves are accepted. In the inner loop, operations like move and rotate are applied at each value of temperature. For each move the change in the cost function is used to calculate the probability of acceptance P, P is compared to a random Number 'r' generated between 0 and 1. If P > r then move is accepted otherwise it is rejected and reversed.

By applying the move and rotate operations the area and configuration of modules are changed. These operations should be applied such that no two modules overlap and functionality of the module is not altered while minimizing the total area.

III. IMPLEMENTATION

The basic shapes are drawn using the geometry structure. This basic geometry structure consists of array of points

required to draw different geometries. The Basic tab is shown Figure 1 which includes various buttons for drawing different shapes.

Using Geometry structure the different shapes that can be drawn are triangle, rectangle, polygon, curve etc. Additional functions for Zooming the viewport area, inverting the picturebox area and rotating the geometries are included.

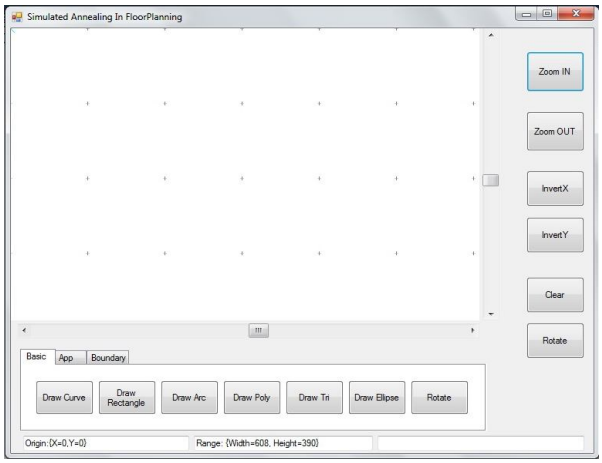

Figure 1: Basic Tab

A bounding Rectangle of certain height H and width W and area A is considered which is the top most module and consists of several other modules. The bounding box must be large enough to contain the modules assigned to it. The main aim of current application is to minimize the number of layers, number of boards or chips thus minimizing the area. This application is implemented using four basic classes namely, Geometry, Temperature profile, simulated annealing and Data_process class. Geometry class contains all the geometric and graphic related information . The temperature profile class provides linear and logarithmic temperature curves for annealing process. A temperature profile curve is shown in Figure 2.

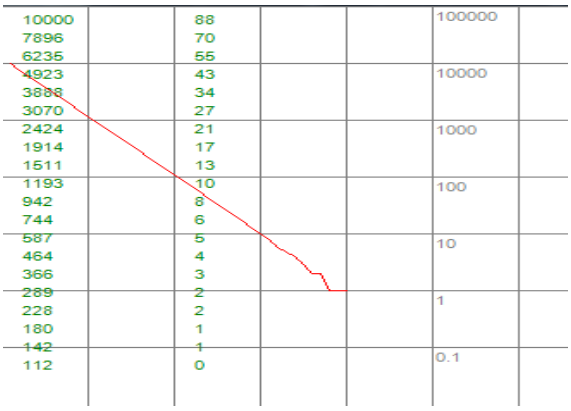

Figure 2: Logarithmic Temperature Profile

IV. RESULTS

Initially during the annealing process relaxation of bounding rectangle occurs, where all the modules occupy different positions. Molecules have high energy at higher temperatures thus movement is fast, similarly modules at high temperature have enough energy to move against gravity and hence module tries to move away. As temperature decreases the movement becomes slow and modules move to a low energy point, typically, a vertex in the crystal structure. Simulated annealing also behaves in a similar way, moving towards a fixed 'crystalline' structure identified by a boundary. So, the free area is a function of temperature. Certain stochastic variation in free area is also expected. As the temperature is gradually reduced, the modules move towards left top corner and the overall size of the layout reduces. Initial bounding Rectangle with some modules is shown in Figure 3.

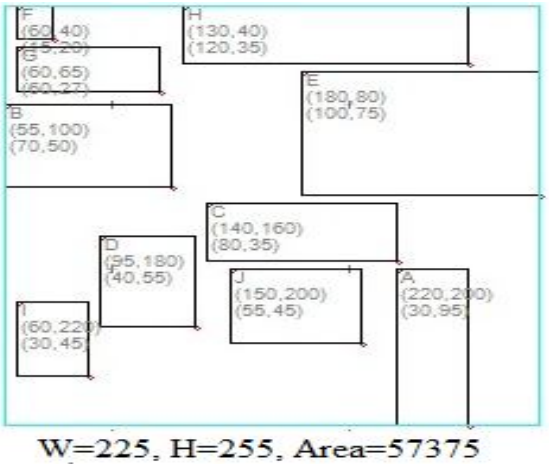

Figure 3: Initial Floorplan

At high temperatures, the modules are expected to move away from each other, trying to free away from the solid 'crystal.' This process, generally called Relaxation, is shown in Figure 4. During relaxation process the bounding rectangle increases in both width and height where the modules find a much large area to relax. The components are shifted to different location using move and translate methods. The component dimensions remain unaltered. Overlaps are not allowed. If there is an overlap, one of the component is moved till it occupies a free space. Motion of such a module is controlled critically as it affects the overall performance of the algorithm.

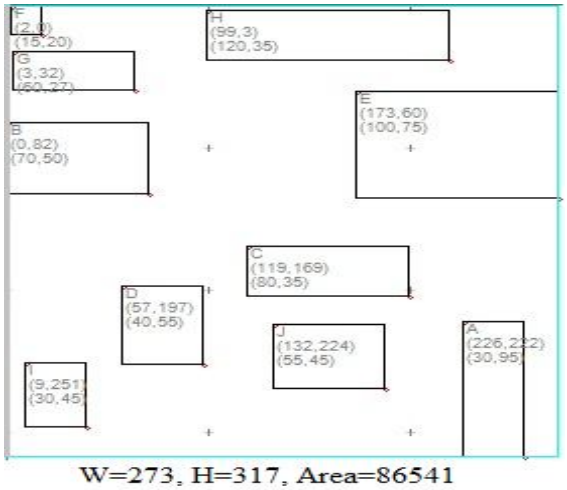

Figure 4: Relaxation of Bounding Rectangle

As the temperature reduces, free area around every module reduces and the movement of the module gets restricted. Thus the algorithm simulates the physical annealing process.

The component can be moved within the free area with varying degrees of freedom depending on temperature and proximity of adjacent components. Free area estimation for block 'D' is shown in Figure 5. Note that the boundary is limited by components in east-west direction but extent-limited in north-south directions.

To derive the area estimation, we start with a rectangle of dimensions equal to the component. It is increased on all the four sides by some small value, forming a much bigger rectangle with more area. This process is repeated for several cycles till it stabilizes, corresponding to the temperature. If the Free area overlaps any existing component, estimator stops expanding in that direction, but continues in other directions. Once the free area estimation is completed during an iteration, the block (or module) is moved towards the top left corner of the bounding rectangle. This movement is in anticipation that overall area will reduce with every iteration. Note that overall area will reduce if the edges of the bounding box move towards the left – top corner.

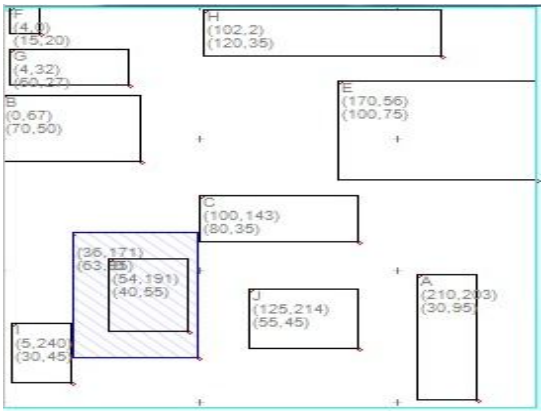

Figure 5: Free area estimation for Block D

Movement of the block within the free area is shown in Figure 6. This process is continued until all the blocks are moved towards the origin of bounding rectangle such that area is recomputed. This movement within the free space is continued for all the modules. Finally results obtained will minimize the floorplan area. Minimized area of floorplan is shown in Figure 7.

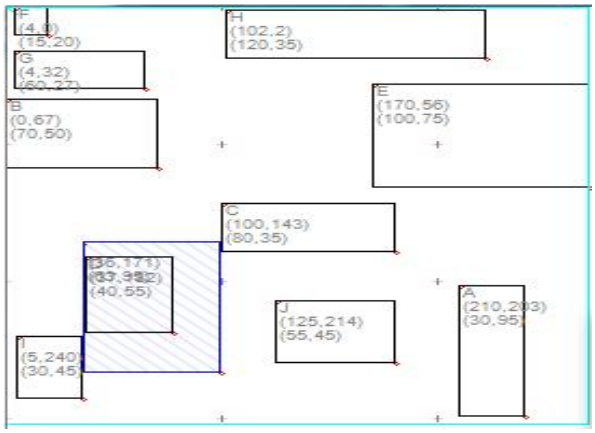

Figure 6: Movement of Block within Free Area

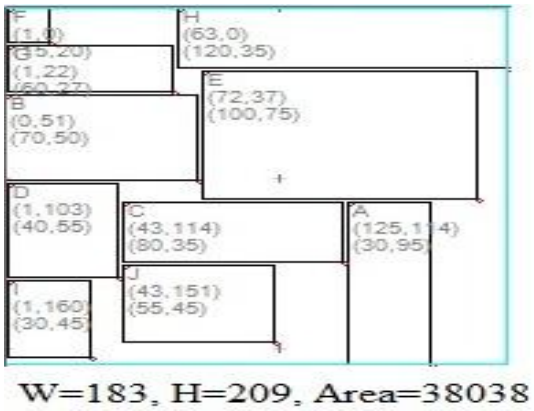

Figure 7: Near optimal Floorplan

V. CONCLUSION

Simulated annealing provides a stochastic method to find near optimal solution to many technology problems. Estimating how the simulated annealing takes place is difficult to compute. Hence, it is mostly left to the implementer to choose how annealing occurs (in simulation). We have suggested a specific mechanism, which we call the *free area estimator*, to implement these estimators.

Simulated annealing is one of the popular methods for floorplan optimization. We have used free area estimator as a means of estimating how the modules can move at various temperatures. Thus movement of component is constrained within the free area estimated. As we cool the simulation, we observe that the area used by the floor plan reduces, thus minimizing the area of floorplan.

The modules are bound within this rectangle and hence bound at all times. As the modules shift, new areas are created to be occupied by other modules. Thus estimation of free area offers a new way to implement simulated algorithms easily. This concept has been used here for floor planning of a VLSI chip. The method is versatile and extended to other areas of research as well.

It is noting worthy that there are a very large number of possibilities about how the modules move at various temperatures. This is similar to the case in nature where many types of crystal structures are seen. These will be studied, analyzed and reported in future.

ACKNOWLEDGMENT

The authors would like to thank C-Quad for their continuous involvement in the project and support. The authors would also like to thank KLE Dr MSSCET for providing necessary infrastructure.

REFERENCES

[1] Naveed Sherwani, - "Algorithms for VLSI Physical Design Automation", Kluwer Academic Publisher, 3rd Edition.

[2] E Balaguruswamy, -"Programming in C# ", A Primer, Second edition.

[3] Ashwini Baligatti, "Simulated Annealing in Floorplanning", M.Tech Project Thesis submitted to VTU June 2014.

Ms. Ashwini Baligatti completed her Bachelor of Engineering with major in Telecommunication Engineering at KLE Dr MSSCET, Belgaum. Presently she is Pursuing Master of Technology in VLSI Design and Embedded systems at KLE Dr MSSCET Belgaum.

Prof. Ashwini Desai completed her BE (E & CE) from GIT, Belgaum affiliated to Karnataka University, Dharwad and was awarded M.Tech. by VTU, Belgaum in the year 2005. She is currently pursuing her PhD, while working as Asst. Professor at KLEDRMSSCET Belgaum. She has presented her work in National and International conferences.

Dr. U. V. Wali is a Professor at KLEDRMSS CET Belgaum and Director at C-Quad computers, Belgaum. He was awarded Ph D from IIT Kharagpur in the year 1986 for his work. He has published papers in 8 International conferences and has also published 6 Journals. His research interests include VLSI physical design, testing and verification, software design and architecture, among others. He is a fellow of Institution of Engineers, India.