Received: Added at production | Revised: Added at production | Accepted: Added at production

DOI: xxx/xxxx

ORIGINAL ARTICLE

Fast Power Density Aware 3D-IC Floorplanning for Hard Macro-Blocks Using Best Operator Combination Genetic Algorithm

Naorem Yaipharenba Meitei*[1] | Krishna Lal Baishnab[2,3] | Gaurav Trivedi[3]

[1]Department of Electronics and Communication Engineering, National Institute of Technology Silchar, Assam, India

[2]Department of Electronics and Communication Engineering, National Institute of Technology Silchar, Assam, India

[3]Department of Electrical and Electronics Engineering, Indian Institute of Technology Guwahati, Assam, India

Correspondence
*Naorem Yaipharenba Meitei, Department of Electronics and Communication Engineering, NIT Silchar, Assam, India-788010. Email: naorem_rs@ece.nits.ac.in

Abstract

In this article, we propose a fast 3D-IC floorplanning method for hard macro-blocks that includes a thermal management scheme. It applies a genetic algorithm constituted by an optimal combination of crossover and mutation operations to identify the optimal solution for design variables, namely, total wire length, number of through-silicon vias (TSVs), and maximum average layer power density. The proposed method additionally makes use of a unique TSV placement scheme that arranges TSVs next to their respective functional blocks. To enable efficient heat transmission to the ambient environment, layers with higher power densities are placed closer to the heat sink. The proposed 3D-IC floorplanning approach provides the fewest TSVs, the lowest peak temperature, and promising values of wire length within the least amount of computation time. Compared to the recent fast thermal analysis for fixed-outline 3D-floorplanning, it generates 13.14% shorter wire length, 39.27% lower peak temperature, and 34.35% lesser number of TSVs on average with significant improvement in computation time, while analyzing GSRC thermal benchmark circuits.

KEYWORDS:
3D-IC; Through-silicon via; VLSI physical design process; thermal-aware floorplanning; genetic algorithm.

1 | INTRODUCTION

Three dimensional integrated circuit (3D-IC) allows integration of more than one layer on a single chip, where each layer represents different types of circuit with independent circuit density. Therefore, 3D-IC is considered as one of the best solutions to address the issues related with scaling down of semiconductor devices. Moreover, 3D-IC provides manifold benefits like reduced wirelength, improved chip reliability and performance as compared to its 2D counterpart. However, one of the most critical challenges associated with 3D-IC is heat dissipation of layers, which are not adjacent to the heat sink[1]. Thus, it is important to take thermal effects into consideration in every abstraction level of 3D-IC physical design process. On the other hand, interconnection between different layers of 3D-IC is realized with through-silicon vias (TSV). It is to mention that the size a TSV is larger than normal vias[2]. Therefore, minimizing the number of TSVs has become another primary objective in 3D-IC design process.

Floorplanning is considered as one of the most important steps of VLSI physical design process. In recent decades, researchers have been contributing, in profusion, in the field of floorplanning. In general, outline-free floorplanning is prevailed since last

many years. However, in the state-of-the-art fixed-outline floorplanning[3], the authors contributed a flow of hierarchical floorplanning with fixed-outline constraints, which is better suited for modern ASICs and SoCs design. In the pursuit of large number of device integration per chip area, the trend of integrated circuit design has been shifted from 2D-integration to 3D-integration. However, the thermal issue is escalated in 3D-integration due to vertical stack of layers resulting in increased power density[4] and less thermal conductivity of dielectric layers between active layers. A thermal-driven floorplanning algorithm for 3D-ICs[2] known as thermal driven combine bucket and 2D-array (CBA-T) not only reduces total wirelength but also optimizes the maximum on-chip temperature. CBA-T presents a resistive thermal model for 3D-IC for on-chip temperature estimation. On the other hand, efficient thermal via planning approach[5] allocates thermal vias in white spaces to reduce on-chip temperature. With a similar objective of reducing temperature, 3D-STAF[6] employs a force-directed technique to optimize leakage power. For addressing the thermal issue along with traditional goals of 3D-floorplanning, a mixed-integer linear programming based 3D-floorplanning is presented in a novel thermal optimization flow using incremental change[7].

Another critical challenge in 3D-floorplannning is the increased complexity due to enlarged solution space due to the presence of multiple-device layers. In fact, CBA-T[2] needs more than 12 Hrs of computation time to estimate an optimum solution on GSRC *n*300 benchmark circuit, which has 300 blocks and 1893 nets. In order to address the issue of design complexity in 3D-floorplanning, a hierarchical 3D-floorplanning algorithm[8] for wirelength optimization is developed. In this method, the 3D-floorplanning problem is dealt in two phases - partitioning of blocks into different layers and later 2D-floorplanning of several layers simultaneously, which is proved to be efficient. In a similar approach, multi-layer floorplanning for stacked ICs[9] exhibits that the number of possible configurations in 3D-floorplans represented by partitioned sequence pair[10] is less than the 2D-floorplans represented by a sequence pair. This scheme employs the block position enumeration technique[11] for efficient multi-layer floorplanning. Unlike the above-mentioned techniques of 3D-floorplanning, in fast placement-aware 3D-floorplanning[12] a single module is allowed to split into cells, which can be placed into different layers. Thus, there is the possibility that a single module is placed in different layers of 3D-IC, thereby, making the method to be placement-aware. On the other hand, the method in ref. [13], the area and positions of signal TSVs are considered while estimating the total wirelength, which was ignored in previous research works on 3D-floorplanning. From the above discussion, it is found that 3D-floorplanning consists of various conflicting design parameters which necessitates optimization.

Several optimization algorithms have been found engaging to solve 3D-IC physical design problems. Some of the literature is discussed in the following parts of the section. The use of genetic algorithm for solving the problem of 3D-IC floorplanning is reported in ref. [14]. In this method, peak temperatures as well as thermal gradients are considered in the objective function. In ref. [15], a study is presented to exhibit usage of 2D intellectual property blocks in the design of 3D-IC. Using skew binary trees, a fixed-outline driven floorplan representation is presented in ref. [16]. This method employs simulated annealing for wirelength optimization. A thermal aware 3D-floorplanner using multi-objective evolutionary algorithm is reported in[17]. This study includes three major parameters - mean, gradient and peak temperatures to achieve a better floorplan. In the fast fixed-outline 3D-floorplanning scheme[18], floorplanning of functional modules and placement of TSVs are performed simultaneously under fixed-outline constraint. This method results in significant improvement in total wirelength with a significantly faster computational speed. A hybrid optimization algorithm based on particle swarm optimization and genetic algorithm is employed in thermal-aware non-slicing floorplanning[19]. This method proved to be efficient in reducing average and peak temperatures. In a similar approach, a combination of ant colony optimization and simulated annealing method is employed for both 2D and 3D-floorplanning in ref. [20]. This study considers three design variables namely wirelength, area and number of TSVs. A routability-driven TSV-aware floorplanning of 3D-ICs is presented in[21]. This method focuses in reducing routing congestion with small wire length overhead. However, these two methods lack thermal management strategy, which is crucial in 3D-floorplanning. In the fast thermal analysis of fixed-outline floorplanning[22], the thermal distribution of each block placed in different layers is estimated before floorplanning. Based on the simulated thermal profile, bi-linear interpolation is employed to calculate temperature during floorplanning. The method proved to be efficient in reducing peak temperature during 3D-floorplanning within a short time. A fast thermal-aware fixed-outline floorplanning[23] presents the use of a differential nonlinear model, which can approximate temperature and minimize wire length simultaneously during floorplanning. A simulated annealing-based temperature aware 3D-floorplanning algorithm[24] is also presented. In this study, for thermal management, hot blocks are placed at the bottom layer where heat sink is assumed to be attached. Further, a reasonable thermal gradient is maintained between different layers as well as between blocks in the same layer. Temperature estimation is performed by incorporating, the Hotspot tool[25].

From the above literature survey, it is noticed that the hierarchical two-phase approach significantly reduces the complexity of 3D-IC floorplanning. It is also important to note that, in addition to other floorplanning design elements, thermal management

is crucial in 3D-floorplanning. On the other hand, the use of evolutionary algorithms proved to be efficient in addressing multi-objective optimization problems of 3D-IC floorplanning.

1.1 | Research Gaps

Based on the literature discussed above, the following scientific gaps are identified. In previous studies, optimization of the peak temperature as well as temperature gradient has been the primary strategy for thermal management. However, such methods require estimation of temperature of functional blocks for each solution in every iteration during the optimization process. Therefore, such thermal-aware methods require additional computation time to estimate temperature. For instance, the method CBA-T[2] needs more than 12 Hrs of computation time to estimate an optimum solution on GSRC *n*300 benchmark circuit, which has only 300 blocks and 1893 nets. Nevertheless, none of the studies has been reported which involves power density optimization to mitigate thermal issue. Another concern is that, in most of the TSV planning methods, TSVs are placed either in white space area or they are planned randomly along with the functional blocks during floorplanning. However, these methods do not assure placement of TSVs adjacent to their corresponding functional blocks thereby limiting the finding of the optimal wirelength. Therefore, it is essential to study methods which place TSVs adjacent to their respective functional blocks. Further, incorporation of genetic algorithm has been reported to be efficient in 3D-IC floorplanning having multiple conflicting parameters with high design complexity[14]. However, an optimal combination of operators can enhance the solution. Nevertheless, to the best of our knowledge, none of the literature is reported having a method to determine the best combination of operators with their respective optimal percentages. Therefore, a thorough study of multiple operators that constitute genetic algorithm is required.

Based on these research gaps, a fast 3D-IC floorplanning for hard-macro blocks which employs genetic algorithm with the best combination of operators, is proposed. Our main contributions are listed below:

a. An innovative 3D-IC floorplanning approach that optimizes for the highest average layer power density to reduce peak temperature.

b. A novel TSV planning scheme which places TSVs adjacent to their corresponding functional blocks.

c. Genetic algorithm with an innovative approach for determination of the best operator combination as well as respective percentages for the operators.

The rest of the paper is organized as follows. The problem formulation of 3D-IC floorplanning is presented in section 2. The proposed 3D-IC floorplanning method is discussed in Section 3. In section 4, a detailed discussion of the proposed optimization technique is depicted. Results of the proposed method with the detailed analysis and comparison with existing methods are stated in section 5. Finally, the article is concluded in section 6.

2 | PROBLEM FORMULATION

Given a set of rectangular functional units with predefined heights, widths and power densities, the problem of 3D-IC floorplanning is to allocate geometric locations for each of the units to different layers of the 3D-IC design such that total wirelength, number of TSVs, and peak temperature are minimized subject to all units are placed inside a user defined rectangular fixed-outline and no unit overlap with any other unit. Mathematically, the problem can be described as follows. Let us consider a set of N rectangular functional units $V = \{v_1, v_2, v_3, \cdots, v_N\}$ in which each unit v_i is defined by a triplet (h_i, w_i, p_i) where h_i, w_i and p_i represent height, width and power density of unit v_i. Also, let $E = \{e_1, e_2, e_3, \cdots, e_M\}$ be set of M hyperedges containing connection information of all functional units. Then, the problem of 3D-IC floorplanning is to allocate geometric locations for each unit v_i defined by a triplet (x_i, y_i, l_i) optimizing the design parameters total wirelength, number of TSVs, and peak temperature subject to non-overlapping and boundary constraints[26], described in subsections 2.1 and 2.2 respectively. Here, x_i, y_i, and l_i represent x-coordinate,y-coordinate and number of layer in which v_i is allotted. If K is the total number of layers in the 3D-IC design, then $1 \leq l_i \leq K$.

2.1 | Non-overlapping Constraint

a. For v_i to the left of v_j :

$$x_i + w_i \leq x_j \tag{1}$$

b. For v_i below v_j :

$$y_i + h_i \leq y_j \tag{2}$$

c. For v_i to the right of v_j :

$$x_i - w_i \geq x_j \tag{3}$$

d. For v_i above v_j :

$$y_i - h_i \geq y_j \tag{4}$$

where $1 \leq i, j \leq N$ in all cases.

2.2 | Boundary Constraint

If H and W denote the height and width of the rectangular fixed-outline inside which all units must be placed, the following conditions must be satisfied during floorplanning.

$$y_i + h_i \leq H \tag{5}$$

$$x_i + w_i \leq W \tag{6}$$

where $1 \leq i \leq N$ in all cases.

3 | THE PROPOSED 3D IC FLOORPLANNING

The proposed 3D-IC floorplanning involves two phases. The first phase is the partitioning of N input rectangular functional blocks into K different layer. It is followed by the second phase which is 2D-floorplanning of each layer. Design variables considered in the proposed method are number of TSVs, largest average layer power density of all layers, and total wirelength in terms of half-perimeter wirelength. Both the phases are discussed in details in the following subsections.

3.1 | Partitioning

The input information fed to the partitioning process are N rectangular blocks with height, width, power density associated to each block and netlist information of all blocks. The partitioning process begins with generating a random sequence of natural numbers whose length is equal to the number of input blocks, N, where each element in the sequence is unique and represents one of the input functional units. It is followed by calculating the area of each layer in the 3D-IC. If a_i denote the area of the unit v_i, where $1 \leq i \leq N$, and K is the number of layers in 3D-IC, then the layer area, A_L is expressed as in equation (7).

$$A_L = H \times W \tag{7}$$

Here, H and W are the height and width of the rectangular layer area and are expressed in equations (8) and (9) respectively [26].

$$H = \sqrt{(1+\gamma)\frac{\sum a_i}{K}\alpha};\ \forall\, i \in (1, 2, \ldots, N) \tag{8}$$

$$W = \sqrt{(1+\gamma)\frac{\sum a_i}{K\alpha}};\ \forall\, i \in (1, 2, \ldots, N) \tag{9}$$

Where α is aspect ratio of the rectangular area given by equation (10) and γ is the fraction of dead-space allowed in the design.

$$\alpha = \frac{H}{W} \tag{10}$$

The functional units are, now, distributed to each layer one by one, starting from the first element of the sequence generated until the total area of blocks reaches the layer area, A_L. The process continues upto the $K-1$ layers whereas the K^{th} layer accommodates the remaining blocks. Once all blocks are grouped in different layers, average layer power density for each layer is calculated with equation (11).

$$PD_{avg,k} = \frac{\sum p_{avg,i}}{A_L} \tag{11}$$

The heat sink is assumed to be attached to the K^{th} layer Further, number of TSVs is calculated using the connectivity information given in the netlist. More precisely, if a unit in one layer is connected to another unit in different layer, then a TSV is counted. Thus, all such connections of blocks lying in different layer makes the total number of TSVs and it can be expressed as in equation (12).

$$N_{tsv} = \sum_{i=1,j=1}^{N,N_n} C(i,j) \tag{12}$$

Here, blocks i and j lie in different layers of 3D-IC. Further, N_{tsv} denote the total number of TSVs, N is the total number of blocks and $C(i,j)$ represents connectivity of blocks i and j. $C(i,j)$ is set as 1 if blocks i and j share at least one common net in the given netlist. In this case

3.2 | Floorplanning

Once the partitioning process is complete, the second phase, that is floorplanning of each layer, is performed. In the proposed method of floorplanning, it is checked for each unit that if its height is larger than its width. If it is so, the block is rotated so that the resulting block's width is larger than its height. Later, blocks are sequenced in descending order of their widths and packing is performed. It is during the packing, geometric locations of all blocks are assigned in the form of x and y-coordinates. The process of packing in the proposed floorplanning is presented in **Algorithm 1**. Packing is performed subject to boundary

Algorithm 1 The proposed Floorplanning process

```
Sort blocks according to descending order of block widths
Initialize layer number k = 1;
for k = 1 to K do
    wtemp = 0;
    htemp = 0;
    n = 1;
    while n < totalLayerBlocks do
        xcor = wtemp;
        while htemp < H do
            Layer[k].block[n].x-cor=xcor;
            Layer[k].block[n].y-cor=htemp;
            htemp = htemp + height[n];
            wtemp = max(wtemp, xcor + width[n]);
            if htemp > H then
                htemp = htemp − height[n];
                break;
            end if
            n = n + 1;
            if n > totalLayerBlocks then
                break;
            end if
        end while
    end while
end for
```

constraint and non-overlapping constraints mentioned in subsections 2.1 and 2.2 respectively. More precisely, all blocks must be placed inside the layer area, A_L bounded by the fixed height H and fixed width W without overlapping one another. While packing, the first block in the sequence is placed at the bottom left corner of the A_L and other units from the sequence are placed on top of its predecessor one by one until cumulative height of the packed blocks becomes larger than or equal to H. The cumulative height is continuously checked whether it is larger than H or not. If it is larger than H, the topmost block is

removed and it is placed at the most left available space on the bottom. The cumulative height of packing is, then, updated and set it as the height of the block just placed. The process is repeated until all blocks in the layer, are placed. This completes the floorplan packing of a layer. The same process is repeated for all layers.

3.3 | TSV Planning

Planning of through silicon vias, is one of the important goals in the proposed 3D-IC floorplanning scheme. The number of TSVs required for a particular design of 3D-IC is determined during the partitioning phase 3.1. In the proposed TSV planning scheme, three steps are involved which are follows.

i. Identification of number of TSVs connected to each functional unit in all layers.

ii. Creating space for TSV placement adjacent to each block.

iii. Assigning geometric x and y coordinates for each TSV.

The number of TSVs associated with a particular functional block can be identified by exploiting the netlist information. **Algorithm** 2 illustrates how number of TSVs associated with each functional block are calculated. In this approach, for each connection of a block to another block that lie in different layer, one TSV for the block is counted. In the next step, dedicated

Algorithm 2 Finding number of TSVs associated with each block

```
Input: 3D Partitioned blocks
Initialize number of blocks = N
for i = 1 to N do
    TSVs_of_block [i]=0;
end for
for m = 1 to N do
    set variable n = 1;
    for n = 1 to N do
        if m ≠ n then
            if connect[m][n] = 1 then
                if layer_of[m] ≠ layer_of[n] then
                    nTSVs[i] ← nTSVs[i] + 1;
                end if
            end if
        end if
    end for
end for
```

space for TSV planning is created right next to each block so that TSVs can be placed with direct connection to their corresponding functional units. The process of creating TSV space adjacent to functional units, is demonstrated in **Algorithm** 3. As stated in section 3.2, the blocks are sequentially packed starting from the bottom till the cumulative height of the placed blocks reaches the height, H of the fixed-outline. This leads to column wise fashion of block arrangement in the output of the proposed floorplanning method. Therefore, for creating dedicated space for TSVs each column of blocks is shifted 2 μm right except for the first column. Following the space creation, TSVs are planned with the process represented by **Algorithm 4**. This process assigns all TSVs associated with each block with x-coordinates that equal to $x_i + w_i$, where x_i and w_i are the x-coordinate and width of the i^{th} block. This makes TSVs attached on the right of their corresponding blocks as shown in the Fig. 2. On the other hand, y-coordinates are assigned in such a way that each TSV is just on top of another except for the first one. Typical floorplan layouts for all layers created by the proposed 3D-IC floorplanning are shown in Fig. 1. Further, a part of the floorplan showing placed TSVs is presented in Fig. 2. .

Algorithm 3 Creating dedicated space for TSVs for each column

Input: 3D Floorplan Information
set column number $m = 1$;
while $m < number\_of\_columns$ **do**
    set column block number $n = 1$;
    while $n < number\_of\_blocks$ in m^{th} column **do**
        $x_n \leftarrow x_n + 2(m-1)$;
        $n \leftarrow n + 1$;
    end while
    $m \leftarrow m + 1$;
end while
Get floorplan with TSV space;

Algorithm 4 TSV Planning

Input: 3D Floorplan with TSV space
Initialize total number of blocks = N;
for $i = 1$ **to** N **do**
    set block TSV number $n = 1$;
    set $temp = 0$;
    while $n <$ no._of_TSVs connected to i^{th} block **do**
        $x.TSV[i][n] \leftarrow x_i + w_i$;
        $y.TSV[i][n] \leftarrow y_i + temp$;
        $temp \leftarrow temp + 1.5$;
        $n \leftarrow n + 1$;
    end while
end for
Get 3D floorplan with placed TSV;

3.4 | Thermal Management

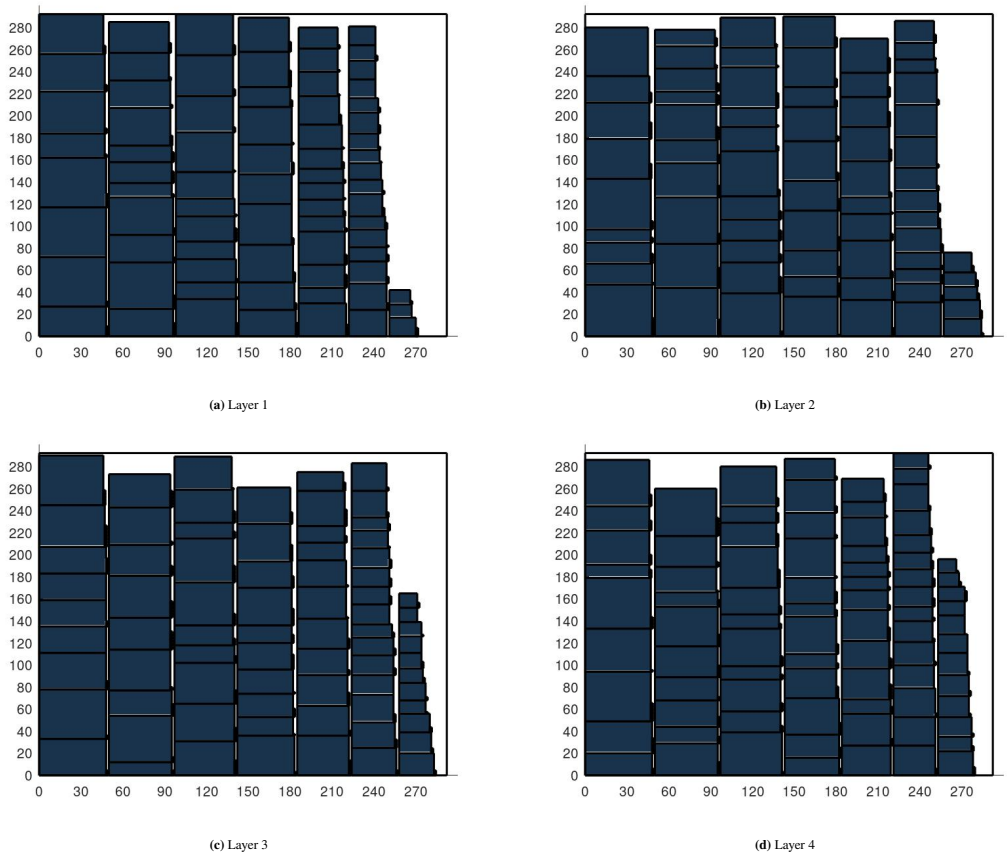
Incorporating thermal management strategy is one of the main objectives in the proposed method of 3D-IC floorplanning. In the proposed method each layer of 3D-IC is divided into rectangular grids of functional units. The temperature of different grids of the chip is directly proportional to its power density and it can be evaluated by posing as steady-state thermal diffusion problem (13)[26].

$$GT = P \tag{13}$$

where G, T and P are conductance, temperature and power density matrices respectively. Thus, optimizing power density can indirectly lead to peak temperature optimization. In addition to this, for efficient heat dissipation to the outer ambient environment, layers with higher power densities are relatively placed closer to the heat sink.

3.5 | Temperature Estimation

As stated earlier, thermal management is one of the important objectives in 3D-IC design. In the proposed 3D-IC floorplanning method, power density is considered as one of the design variables of optimization. The optimization of maximum average layer power density will indirectly lead to peak temperature optimization. To validate the above statement, steady state temperature of each and every functional unit is estimated using Hotspot 6.0 tool available at[27]. Values of all required parameters for simulation is taken as the default values available in Hotspot 6.0 tool. The results are presented and discussed in results and discussion section.



(a) Layer 1 **(b)** Layer 2

(c) Layer 3 **(d)** Layer 4

Figure 1 Floorplan layouts for GSRC *n*300 benchmark circuit

3.6 | Wirelength Estimation

Once, all blocks in all layers are placed, total wirelength (WL_{total}) is estimated as the sum of half-perimeter wirelength of all nets ($HPWL$) and wirelength due to TSVs (WL_{tsv}). Mathematically, WL_{total} can be expressed in equation (14).

$$WL_{total} = HPWL + WL_{tsv} \tag{14}$$

While $HPWL$ and WL_{tsv} can be expressed as in equations (15) and (16) respectively.

$$HPWL = \sum_{i=1}^{N_{net}} HPWL_i \tag{15}$$

$$WL_{tsv} = \sum_{i=1}^{N_{tsv}} t_{si} \times |l_{i,1} - l_{i,2}| \tag{16}$$

Where $HPWL_i$ denotes half-perimeter wirelength of the i^{th} net, t_{si} denotes thickness of silicon layer, $l_{i,1}$ and $l_{i,2}$ denote layer on which first and second ends of the i^{th} TSV lies. Further, $HPWL_i$ can be expressed as in equation (17)[26].

$$HPWL_i = |x_{max,i} - x_{min,i}| + |y_{max,i} - y_{min,i}| \tag{17}$$

Here, $x_{max,i}$ and $x_{min,i}$ represent maximum and minimum x-coordinates for the i^{th} net respectively. Similarly, $y_{max,i}$ and $y_{min,i}$ represent maximum and minimum y-coordinates for the i^{th} net respectively. An example for estimation of $HPWL$ is shown in Fig. 3 in which a particular net consists of three units. Here, the $HPWL$ is indicated by the dark-dashed line.

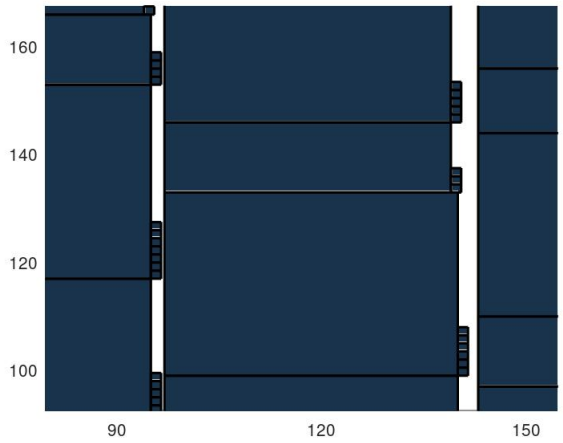


Figure 2 Figure showing TSVs placed adjacent to functional blocks

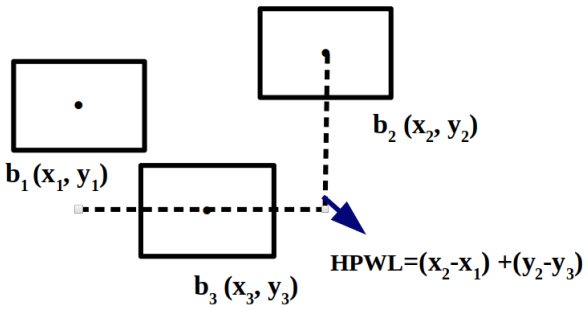


Figure 3 Figure showing half-perimeter wirelength estimation

4 | OPTIMIZATION WITH BEST OPERATOR COMBINATION GENETIC ALGORITHM

The proposed 3D-IC floorplanning method focuses in optimizing the design parameters total wirelength, total number of TSVs and largest average power density of layers in the 3D-IC. A genetic algorithm (GA) engaging elitism, crossover and mutation is employed for optimizing the design parameters. In order to find out the best combination of operators, several crossover and mutation operators are implemented. One-point, two-point, order, order-2, best-order, position crossover[28], and even crossover[29] operations are incorporated. At the same time, twors, insertion, inversion[30] and complement mutation[29] operations are implemented. Further, performance and results of each combination is analyzed. The best set of combination is chosen for the proposed 3D-IC floorplanning scheme. For encoding solutions, permutation encoding scheme[31] is applied. Therefore, each chromosome is represented by an integer sequence of length equal to the number of functional units in the input circuit. Further, each element in the sequence is unique and represents a functional block. As stated earlier in 3.1, in the process of partitioning, functional units in the sequence are distributed to layers by taking one at a time starting from the first element to create groups for different layers of the 3D-IC. Thus, a unique sequence results in a unique partitioning solution and hence a unique 3D-IC floorplan for the input circuit.

4.1 | Objective Function

The objective function for the optimization process is a function of design parameters namely - total wirelength, total number of TSVs and maximum of average power densities of all layers in the 3D-IC. If F denote the objective function, mathematically

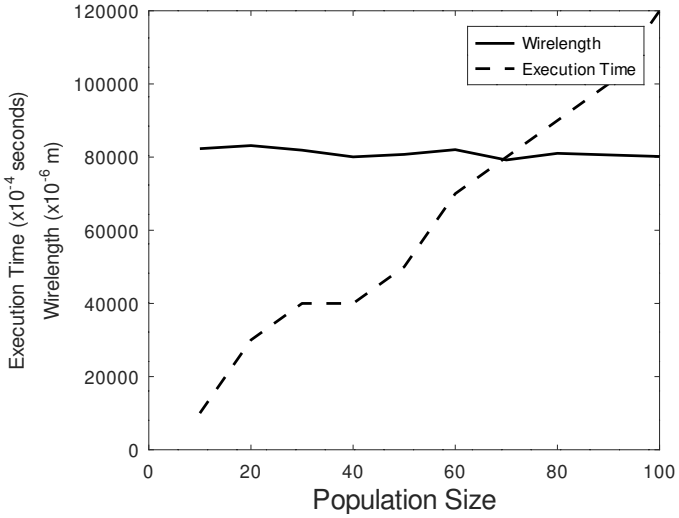


Figure 4 Variation in execution time and total wirelength at different population sizes

it can be expressed as equation (18).

$$F = WL_{total} + N_{tsv} + max(PD_{avg,k});\ \ 0 \leq k \leq K \tag{18}$$

Where K is the number of layers in the 3D-IC while WL_{total} and N_{tsv} are given by equations (14) and (12) respectively.

4.2 | Determination of parameters of the optimization

The size of population for optimization is an important parameter as it is one of the factors that dictates the execution time of the algorithm. In addition to this, the quality of solution can be increased with increase in the size of population. Therefore, choosing an optimum population size is very important for optimization process. The proposed floorplanning is scheme is subjected to GA with different population sizes and results are analyzed carefully. The plot in Fig. 4 shows the variation in total wirelength and execution time resulted from GA with different population sizes. The population size varies from 10 to 100 with a difference of 10 between each pair of consecutive values. It can be observed that with increase in the population size the quality of solution is improved insignificantly. However, the execution time increases significantly in each step increase of population size. The best solution is achieved when the population size is 100 within 12 seconds. While the algorithm results a slightly poorer solution in just 1 second when the population size is 10. Precisely, a 2.6 % improvement is achieved when population size is taken as 100 in comparison with GA with population size equal to 10, however, at the cost of twelve times the execution time. Therefore, the optimization of the proposed floorplanning scheme is performed with population size kept as 10.

In addition to the population size, the percentages of elitism, crossover and mutation are also important parameters of genetic algorithm. Hence, fixing suitable percentages of all operators is necessary. Keeping the population size as 10, different combinations of elitism, crossover and mutation operations are incorporated to GA and results are analyzed carefully. Let us denote the percentages of elitism, crossover, and mutation with E_{pc}, X_{pc} and M_{pc} respectively. Elitism helps in convergence of the algorithm while crossover is responsible for good quality solution. Mutation is incorporated in order to bring sudden chance in some bad chromosomes. We set M_{pc} as 10% in all cases while varying E_{pc} and X_{pc}. Several combinations of E_{pc} and X_{pc} are subjected to GA and results in each of the cases are analyzed and shown in Fig. 5.

As it can be observed from Fig. 5, of all the cases, the combination where E_{pc}, X_{pc} and M_{pc} are 20%, 70% and 10% respectively, demonstrates the best value of the objective function. This concludes our analysis and we set the aforementioned combination for the optimization.

4.3 | Optimization Process

After, the parameters of optimization are fixed, an initial population consisting of 10 sequences, is generated. For each sequence, partitioning and floorplanning are performed. Later, the value of cost function for each chromosome is calculated using equation

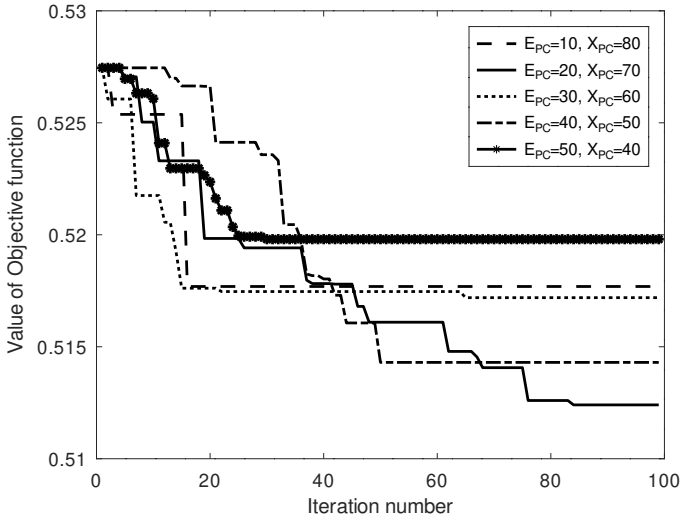


Figure 5 Analysis of difference combinations of operators in GA

(18). The process of optimization is presented in **Algorithm 5** and is explained in the following parts of the subsection. To

Algorithm 5 Algorithm showing the optimization process

```
Get input information
Initialize Solution number, S = 0;
Initialize Population size, P = 10;
Initialize Iteration number, N = 1;
while S < P do
    Generate a solution;
        Generate a sequence;
        Do partitioning;
        Do floorplanning;
    Evaluate objective function value, F; (18)
    S← S+1;
end while
while N < 100 do
    Create next generation solutions;
        Elitism, 20%;
        Position crossover, 70%;
        Complement mutation, 10%;
    N← N+1;
end while
Get optimal solution;
```

generate next generation chromosomes, 2 best chromosomes from the current generation are carried over to the next generation without any alteration. This is elitism. The remaining 8 chromosomes for the next generation are created by applying crossover and mutation operations.

For creating next generation chromosomes with crossover operation, *position crossover* operation[28] is employed. For each generation, 7 pairs of parent chromosomes are randomly selected from the previous generation. Futher, crossover operation is performed for all the pairs to generate one child chromosome from each pair. Thereafter, the remaining one chromosome is generated by mutation operation.

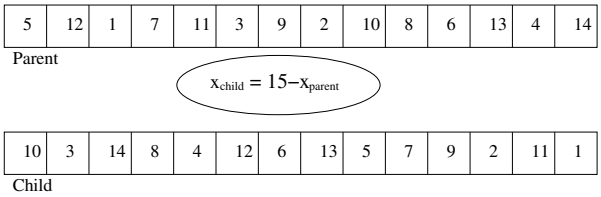


Figure 6 An example showing the complement mutation operation[29]

Table 1 GSRC benchmarks details

Test Circuits	No. of I/Os	No. of Nets	No. of blocks
*n*100	334	885	100
*n*200	564	1585	200
*n*300	569	1893	300

Complement mutation operation[29] is applied to the worst chromosome of the current generation. If N is the total number of units in a chromosome, then each element in the child chromosome sequence is formed by complement the corresponding element in the parent chromosome from $N + 1$. An example of complement mutation operation is shown in Fig. 6. In this example total number of elements is 14. Therefore, each element in the child chromosome sequence is equal to 15 minus the corresponding element in the parent chromosome sequence. After creating all child chromosome sequences, partitioning and subsequently floorplanning is performed for each child sequences. Further, objective function value is evaluated for each chromosome using equation (18). Thus, a new generation of chromosomes is created using the current generation chromosomes by the proposed best operator combination genetic algorithm.

For each generation, the best solution, in terms of the objective function value, is stored. The optimization process ends when 100 generations are completed. The algorithm is evaluated only for 100 generations because further increase in the number of generations does not improve the solution quality in most cases. On the other hand, increase in the number of iterations increases the execution time significantly. The optimal solution is stored as the output of the process. This includes, floorplan layout of each layer as well as the of values of design variables - total wirelength, total number of TSVs, and maximum average power density of all layers in the 3D-IC.

5 | RESULTS AND DISCUSSION

The proposed 3D-IC floorplanning method is implemented using C++ language in Ubuntu Linux platform. It is executed by a system having Core-i3 processor with 8GB RAM and tested GSRC benchmark circuits with power density information - *n*100, *n*200 and *n*300[32]. The details of GSRC benchmarks are shown in Table 1. The proposed 3D IC floorplanning algorithm using GA generates promising values of design variables especially, total wirelength of the entire floorplan. The experimental data for the design are - the aspect ratio α of the fixed-outline is 1 while the dead space percentage γ assumes 30%, 30% and 27% for *n*100, *n*200 and *n*300 respectively. The size of TSV pitch is taken as 1.5 μm x 1.5 μm according to the International Roadmap for Semiconductors 2.0, 2015, Interconnect, available at[33]. Further, the number of layers of 3D-IC is considered to be four. However, the proposed algorithm is adjustable for different values for the number of layers.

In order to find out the best combination of operators for the proposed 3D-IC floorplanning method, the performance and results all crossover operators mentioned in section 4 are analyzed. The comparison is made while employing complement mutation in all cases and is presented in Table 2.

As it can be observed from Table 2, while analyzing the test circuit *n*100, the position crossover method outperforms other crossover methods in terms of wirelength while resulting quite promising values of parameters namely peak temperature and number of TSVs. Further, the position crossover gives better wirelength, peak temperature as well as least number of TSVs when compared to other crossover methods while analyzing *n*200. On the other hand, while analyzing the benchmark circuit *n*300, the best-order crossover gives the best wirelength, order-2 crossover provides the least number of TSVs,and order crossover method results into the lowest peak temperature. However, it is to mention that the position crossover results into quite promising values

Table 2 Comparison of different crossover operators

Test Circuits	Crossover Operators	Total Wirelength	Maximum LPD	No. of TSVs	Computation Time	Peak Temperature
*n*100	Even	85974	5.09	447	1	331.58
	One-point	87963.5	4.91	439	1	332.62
	Two-point	87366.5	4.96	452	1	332.59
	Order	86382	6.35	416	2	333.76
	Order-2	86617	4.20	429	1	328.69
	Best-order	82305	4.76	446	1	335.39
	Position	81763	4.53	424	1	329.00
*n*200	Even	189112	5.14	1048	3	329.92
	One-point	188654	5.14	1048	3	329.32
	Two-point	188137	5.14	1048	4	335.50
	Order	188536	5.14	1045	4	333.90
	Order-2	190243	4.47	1001	3	331.80
	Best-order	182289	5.11	1057	4	328.52
	Position	179394	4.70	989	3	326.74
*n*300	Even	309712	5.28	1320	5	337.31
	One-point	315167	5.05	1312	5	336.76
	Two-point	317544	5.23	1343	5	332.89
	Order	317818	5.07	1313	4	332.08
	Order-2	315034	4.39	1277	5	335.41
	Best-order	303611	5.22	1309	4	335.87
	Position	310056	4.92	1280	4	335.42

Note: Note: Maximum LPD indicates the largest of all layer power densities (average) expressed in $10^5 W/m^2$*. Total Wirelength is expressed in terms of μm, computational time in seconds and peak temperature is presented in Kelvin.*

of all parameters. At the same time, it is observed that variation in the execution time for different operations is insignificant. Overall, it is not too much to state that position crossover provides better values of design variables when compared to other crossover methods in most of the test cases. While some of the parameters are quite acceptable if not the best.

Similarly, various mutation operators are analyzed while position crossover is incorporated and the results are compared and the comparison is demonstrated in Table 3. While analyzing the test circuits *n*100 and *n*200, the complement mutation operator gives better wirelength, number of TSVs as well as lesser peak temperature in comparison with those of other mutation operators. On the other hand, while analyzing *n*300, all the mutation operators provide values of design variables which are very close to each other in comparison. Thus, complement mutation operator is one of the best mutation operators.

Considering the facts demonstrated in Table 2 and Table 3, position crossover and complement mutation operators are employed in genetic algorithm for optimization of the proposed 3D-IC floorplanning scheme. For convenience in further discussions in this article, the proposed method is termed as 3D-IC Floorplanning using genetic algorithm with best operator combination and abbreviated as 3DFLP-BOCGA.

To validate the proposed 3D-IC floorplanning, results, in terms of half-perimeter wirelength, number of TSVs, peak steady state temperature and runtime are compared with well-known existing recent methods of 3D-IC floorplanning. It is presented in Table 4. To increase readability of the table, the best values are made bold and the values generated by the proposed method is presented in italics. While analyzing *n*100 benchmark circuit, the proposed 3DFLP-BOCGA results into the best wirelength, peak temperature and number of TSVs. Moreover, it can also be observed that 3DFLP-BOCGA takes significantly lesser amount of computation time in comparison with all other methods shown in Table 4.

While analyzing *n*200, 3DFLP-BOCGA results the least peak temperature, number of TSVs as well as the execution time although the best wirelenth is presented by 3D-STAF[6]. However, mention may be made that the proposed method results in

Table 3 Comparison of mutation operators

Test Circuits	Mutation Operators	Total Wirelength	Maximum LPD	No. of TSVs	Computation Time	Peak Temperature
*n*100	Inversion	83537.5	4.00	411	1	335.58
	Insert	84456.5	4.28	424	1	330.10
	Twors	85685	4.48	395	1	329.72
	Complement	81763	4.53	424	1	329.00
*n*200	Inversion	187324	4.36	991	3	330.36
	Insert	189818	5.18	975	3	332.71
	Twors	178782	4.36	997	3	329.83
	Complement	179394	4.70	989	3	326.74
*n*300	Inversion	310404	4.70	1252	5	330.94
	Insert	299586	4.92	1288	5	334.54
	Twors	312100	4.87	1260	5	331.78
	Complement	310056	4.92	1280	4	335.42

Note: Note: Maximum LPD indicates the largest of all layer power densities (average) expressed in $10^5 W/m^2$. *Total Wirelength is expressed in terms of μm, computational time in seconds and peak temperature is presented in Kelvin.*

wirelength which is just 6.9% larger than the best value. This may be explained by the fact that the proposed method is having a lesser compact floorplanning solution than 3D-STAF. Since, the design is lesser compact, a 25.33% lesser peak temperature is achieved by the proposed method in comparison with the method 3D-STAF.

Similarly, while analyzing the circuit *n*300, it is observed that the least number of TSVs as well as least peak temperature is acheived by 3DFLP-BOCGA while taking the least amount of execution time. Whereas, the best wirelength is achieved by 3D-STAF while the proposed method results in 30.99% larger wirelength which may be explained by the larger area used in the proposed method. Moreover, a better peak temperature is achieved by the proposed method as compared to 3D-STAF.

Compared to the recent fast thermal analysis for fixed-outline 3D-floorplanning[22], the proposed 3D-floorplanning generates 13.14% shorter wirelength, 39.27% peak temperature, and 34.35% lesser number of TSVs on average, while analysing GSRC thermal benchmark circuits[32]. In addition, the proposed method takes only a few seconds of computation time while the fast thermal analysis of fied-outline 3D-floorplanning takes hundreds of seconds of computational time.

From the above analysis, it is proven that the proposed 3D-IC floorplanning method using genetic algorithm with the best combination of operators provides the lowest peak temperature and least number of TSVs and takes the smallest amount of execution time in all test cases[32] when compared with recent existing methods of 3D-IC floorplanning. In addition to that, it has been demonstrated that the proposed method results into the best or promising values of wirelength if not the best.

6 | CONCLUSION

The proposed 3D-IC floorplanning constitutes two phases - partitioning of functional blocks to different layers and 2D-floorplanning of each layer. The method employs a genetic algorithm with position crossover[28] and complement mutation[29] for optimizing the design variables - the total wirelength, the number of TSVs and maximum average layer power density. The parameters of optimization namely, the population size and percentages of elitism, crossover and mutation are decided systematically with the help of thorough analysis. For efficient thermal management, layers with higher power densities are placed closer to the heat sink to enable efficient heat transfer to the ambient environment. Compared to the recent fast thermal analysis for fixed-outline 3D-floorplanning[22], the proposed 3D-floorplanning generates 13.14% shorter wirelength, 39.27% peak temperature, and 34.35% lesser number of TSVs on average, while analysing GSRC thermal benchmark circuits[32]. In addition, the proposed method takes only a few seconds of computation time while the fast thermal analysis of fied-outline 3D-floorplanning takes hundreds of seconds of computational time. The proposed 3DFLP-BOCGA is proven to be efficient and provide lesser

Table 4 Comparison of the proposed 3D-IC floorplanning with existing methods of 3D-IC floorplanning

Test Circuits	Methods	Total Wirelength	Peak Temperature	TSVs	Computational Time(secs)
*n*100	CBA-T-Fast[2]	111979	495	1156	446
	3D-STAF[6]	91500	429.8	753	341
	IAR-MLFP[9]	121206	NA	859	2.84
	TSV-aware[13]	148748	NA	1171	1306.39
	Heuristic[14]	97022.4	401.06	884	100.7
	Co-place[18]	133980	642	677	2.61
	Two-phase[20]	117502	NA	793	NA
	Fast thermal[22]	93293.19	553.56	746	64.53
	Fast thermal[23]	205159	331.03	NA	1.4
	3DFLP-BOCGA	***81763***	***329.00***	***424***	***1***
*n*200	CBA-T-Fast[2]	203530	515	2058	4474
	3D-STAF[6]	**167800**	437.6	1356	643
	IAR-MLFP[9]	214218	NA	1737	13.68
	TSV-aware[13]	291091	NA	2179	8237.10
	Heuristic[14]	187195.2	402.83	1810	348.75
	Co-place[18]	250829	629	1572	4.97
	Two-phase[20]	211691	NA	1697	NA
	Fast thermal[22]	253513.27	538.15	1564	185.53
	Fast thermal[23]	362153	340.09	NA	6.3
	3DFLP-BOCGA	*179394*	***326.74***	***989***	***3***
*n*300	CBA-T-Fast[2]	326630	481	2350	4953
	3D-STAF[6]	**236700**	441.2	2173	1394
	IAR-MLFP[9]	298337	NA	1825	30.95
	TSV-aware[13]	391694	NA	2730.6	21450.50
	Heuristic[14]	263350	409.47	1914	917.9
	Co-place[18]	350980	650	1758	3.38
	Two-phase[20]	299205	NA	1807	NA
	Fast thermal[22]	303362.81	540.71	1665	316.02
	Fast thermal[23]	504464	343.3	NA	11.2
	3DFLP-BOCGA	*310056*	***335.42***	***1280***	***4***

Note: Results of the compared methods are taken from the original sources. Computation time is not directly comparable. Wirelength is in terms of μm, and temperature is presented in Kelvin.
Bold figures indicate the best values while the italics are results obtained from the proposed method.

number of TSVs as well as lower values of peak temperature with lesser computation time while resulting into acceptable values of wirelength during the analysis of all benchmarks[32] when compared with existing methods of 3D-IC floorplanning methods.

References

1. Knechtel J, Lienig J. Physical Design Automation for 3D Chip Stacks: Challenges and Solutions. In: ; 2016: 3–10.

2. Cong J, Zhang Y. Thermal-aware physical design flow for 3-D ICs. In: ; 2006: 73–80.

3. Adya SN, Markov IL. Fixed-outline floorplanning: Enabling hierarchical design. *IEEE transactions on very large scale integration (VLSI) systems* 2003; 11(6): 1120–1135.

4. Chiang TY, Souri SJ, Chui CO, Saraswat KC. Thermal analysis of heterogeneous 3D ICs with various integration scenarios. In: IEEE. ; 2001: 31–2.

5. Li Z, Hong X, Zhou Q, et al. Efficient thermal via planning approach and its application in 3-D floorplanning. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems* 2007; 26(4): 645–658.

6. Zhou P, Ma Y, Li Z, et al. 3D-STAF: scalable temperature and leakage aware floorplanning for three-dimensional integrated circuits. In: IEEE. ; 2007: 590–597.

7. Li X, Ma Y, Hong X. A novel thermal optimization flow using incremental floorplanning for 3D ICs. In: IEEE. ; 2009: 347–352.

8. Li Z, Hong X, Zhou Q, et al. Hierarchical 3-D floorplanning algorithm for wirelength optimization. *IEEE Transactions on Circuits and Systems I: Regular Papers* 2006; 53(12): 2637–2646.

9. Chen S, Yoshimura T. Multi-layer floorplanning for stacked ICs: Configuration number and fixed-outline constraints. *Integration* 2010; 43(4): 378–388.

10. Murata H, Fujiyoshi K, Nakatake S, Kajitani Y. VLSI module placement based on rectangle-packing by the sequence-pair. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems* 1996; 15(12): 1518–1524.

11. Chen S, Yoshimura T. Fixed-outline floorplanning: Block-position enumeration and a new method for calculating area costs. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems* 2008; 27(5): 858–871.

12. Nain RK, Chrzanowska-Jeske M. Fast Placement-Aware 3-D Floorplanning Using Vertical Constraints on Sequence Pairs. *IEEE transactions on very large scale integration (VLSI) systems* 2011; 19(9): 1667–1680.

13. Tsai MC, Wang TC, Hwang T. Through-Silicon Via Planning in 3-D Floorplanning. *IEEE Transactions on Very Large Scale Integration (VLSI) Systems* 2011(8): 1448–1457.

14. Frantz F, Labrak L, O'Connor I. 3D IC floorplanning: Automating optimization settings and exploring new thermal-aware management techniques. *Microelectronics Journal* 2012; 43(6): 423–432.

15. Knechtel J, Markov IL, Lienig J. Assembling 2-D blocks into 3-D chips. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems* 2012; 31(2): 228–241.

16. LIN JM, HUNG ZX. SKB-Tree: A Fixed-Outline Driven Representation for Modern Floorplanning Problems. *IEEE transactions on very large scale integration (VLSI) systems* 2012; 20(3): 473–484.

17. Cuesta D, Risco-Martin JL, Ayala JL, Hidalgo JI. 3D thermal-aware floorplanner using a MOEA approximation. *Integration* 2013; 46(1): 10–21.

18. Li CR, Mak WK, Wang TC. Fast fixed-outline 3-D IC floorplanning with TSV co-placement. *IEEE Transactions on Very Large Scale Integration (VLSI) Systems* 2013; 21(3): 523–532.

19. Sivaranjani P, Kumar AS. Thermal-aware non-slicing VLSI floorplanning using a smart decision-making PSO-GA based hybrid algorithm. *Circuits, Systems, and Signal Processing* 2015; 34(11): 3521–3542.

20. Xu Q, Chen S, Li B. Combining the ant system algorithm and simulated annealing for 3D/2D fixed-outline floorplanning. *Applied Soft Computing* 2016; 40: 150–160.

21. Lin JM, Yang JA. Routability-driven TSV-aware floorplanning methodology for fixed-outline 3-D ICs. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems* 2017; 36(11): 1856–1868.

22. Xu Q, Chen S. Fast thermal analysis for fixed-outline 3D floorplanning. *Integration* 2017; 59: 157–167.

23. Lin JM, Chen TT, Chang YF, et al. A fast thermal-aware fixed-outline floorplanning methodology based on analytical models. In: IEEE. ; 2018: 1–8.

24. Ni T, Chang H, Zhu S, et al. Temperature-Aware Floorplanning for Fixed-Outline 3D ICs. *IEEE Access* 2019; 7: 139787–139794.

25. Zhang R, Stan MR, Skadron K. Hotspot 6.0: Validation, acceleration and extension. *University of Virginia, Tech. Rep* 2015.

26. Alpert CJ, Mehta DP, Sapatnekar SS. Handbook of algorithms for physical design automation. 2008.

27. Zhang R, Stan MR, Skadron K. Hotspot 6.0: Validation, acceleration and extension. http://lava.cs.virginia.edu/HotSpot; 2015.

28. Andreica A, Chira C. Best-order crossover for permutation-based evolutionary algorithms. *Applied Intelligence* 2015; 42(4): 751–776.

29. Meitei NY, Baishnab KL, Trivedi G. 3D-IC partitioning method based on genetic algorithm. *IET Circuits, Devices & Systems* 2020; 14(7): 1104–1109.

30. Soni N, Kumar T. Study of various mutation operators in genetic algorithms. *International Journal of Computer Science and Information Technologies* 2014; 5(3): 4519–4521.

31. Sivanandam S, Deepa S. Introduction to Genetic Algorithms. 2008.

32. Cong J. Floorplanning Benchmark (GSRC + Power values). http://cadlab.cs.ucla.edu/three_d/3dic.html; 2012.

33. ITRS . ITRS-2.0 Interconnect. https://www.semiconductors.org/wp-content/uploads/2018/06/6_2015-ITRS-2.0_Interconnect.pdf; 2015.