A Novel Thermal Optimization Flow Using Incremental Floorplanning for 3D ICs*

Xin Li Yuchun Ma Xianlong Hong
Department of Computer Science & Technology, Tsinghua University, Beijing 100084, P.R.China
Email: myc@mail.tsinghua.edu.cn

Abstract: Thermal issue is a critical challenge in 3D IC design. To eliminate hotspots, physical layouts are always adjusted by shifting or duplicating hot blocks. However, these modifications may degrade the packing area as well as interconnect distribution greatly. In this paper, we propose some novel thermal-aware incremental changes to optimize these multiple objectives including thermal issue in 3D ICs. Furthermore, to avoid random incremental modification, which may be inefficient and need long runtime to converge, here potential gain is modeled for each candidate incremental change. Based on the potential gain, a novel thermal optimization flow to intelligently choose the best incremental operation is presented. We distinguish the thermal-aware incremental changes in three different categories: *migrating computation, growing unit* and *moving hotspot*. Mixed integer linear programming (MILP) models are devised according to these different incremental changes. Experimental results show that *migrating computation, growing unit* and *moving hotspot* can reduce max on-chip temperature by 7%, 13% and 15% respectively on MCNC/GSRC benchmarks. Still, experimental results also show that the thermal optimization flow can reduce max on-chip temperature by 14% compared to an existing 3D floorplan tool CBA, and achieve better area and total wirelength improvement than individual operations do.

I. INTRODUCTION

With the fast shrinking of device sizes, interconnect delays become the critical bottlenecks of chip performance. Three-dimensional (3D) integration, as figure 1 shows, recently has drawn much attention due to its potential for reducing the interconnect delay and complexity as well as promising high integration density.

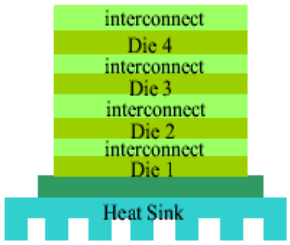
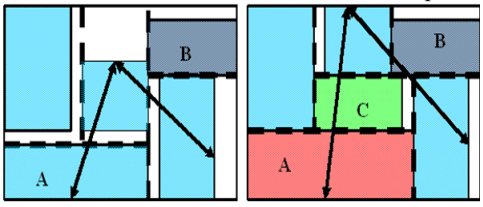


Figure 1 3D IC technology

Though 3D IC has many advantages, there are some significant challenges along with its adoption and further development. With multi-device layers design, the vertically stacked multiple layers of active devices cause a rapid increase of power density and the thermal conductivity of the dielectric layers inserted between device layers for insulation is quite low. Consequently, one extremely important issue in 3D IC design is the thermal problem resulting from both higher power density and lower thermal conductivity.

Recently, several works on thermal optimization during floorplanning for 3D ICs have been proposed [1, 2, 3, 4]. [1] proposed a thermal-driven floorplanning algorithm for 3D ICs. It uses a simulated annealing with an integrated compact thermal model. [2] proposed thermal-aware floorplanning for 3D microprocessors. The power consumption of interconnect is considered during floorplanning. Though the thermal-aware SA-based approaches can indeed distribute heat evenly across the chip to mitigate thermal issue, there is no guarantee to eliminate hotspot completely, sometimes hotspot still exists. To achieve much lower on-chip temperature, minor changes may require a start-over of the floorplanning process, which suffers from long runtime and poor performance scalability. Incremental floorplanning, however, could provide a novel approach: once a good result is obtained, extra thermal improvement can be achieved effectively by eliminating the hotspot incrementally rather than restarting a new general floorplanning.

In the meantime, for an existing floorplan, [5] points out that allocating more die area to blocks especially to hot functional units(*growing unit*) actually has an immediate impact on the temperature. Still, *migrating computation*(MC)[6, 7] provides an attractive way to mitigate thermal issue. It requires a duplicated block of the hot block to share computation tasks, which can efficiently reduce power density of the hot block so as to reduce the max on-chip temperature. Evaluation from [8] shows that migrating computation is surely an efficient technique to decrease max on-chip temperature. Indeed, all these methods can be implemented by effective incremental modifications to avoid random operations.



(a) initial floorplan (b) incremental floorplan
Figure 2 growing units and adding duplicated blocks

Obviously, migrating computation demands a new duplicated block while growing unit will enlarge the hotspot block. In fact, both these methods, say adding block and expanding block, will modify the initial floorplan, which would degrade total wirelength or overall packing area. Take Figure 2 as an illustration, the lines with arrows denote interconnections between blocks. In the initial floorplan as shown in figure 2(a), block B needs a duplicated one to migrate computation and block A needs to grow. Figure 2(b) is the incremental floorplan, where block A is enlarged and block C is a clone of block B which is newly added. After re-placing the blocks, the total wirelength might be increased.

Thus an efficient model is required for incremental modifications to achieve good tradeoff between thermal optimization and other objectives. Especially in 3D IC design, incremental optimization is a promising way to handle multi-objective optimization with complicated constraints and facilitate the design reuse technology. Several works concerned with incremental floorplanning for 2D IC design [9, 10, 11, 12] have been proposed, but none has taken thermal-aware 3D IC design into consideration. [13] proposed a LP based approach to optimize white space to facilitate thermal via insertion, but it is hard to be extended to manage such incremental changes as moving blocks between different layers.

*This work is supported by the NSFC 60606007 and Tsinghua Basic Research Fund JC20070021

978-1-4244-2749-9/09/$25.00 ©2009 IEEE

Additionally, the model formulations alone barely guarantee preeminent results on both runtime and final objectives. For the purpose of optimizing thermal issue during floorplanning, the question is raised: there indeed exist several incremental changes to choose, but which operation is the best one that brings excellent tradeoff? Select randomly or attempt by brute-force? It seems to be not a good idea, for it may be inefficient and need long runtime to converge.

To free designer from this difficult decision-making, in this paper, we propose a novel thermal optimization flow, which can automatically choose the best procedure, based on potential gain for each possible incremental operation. The flow would bring great benefits to designers musing about how to apply incremental methods. Our contributions are summarized as follows:

- MILP based thermal-aware incremental methods. We categorize three different incremental changes in 3D ICs and provide corresponding MILP formulations respectively.
- Simultaneous optimization for chip area, total wirelength and thermal-driven incremental changes. With effective MILP based formulations, our approach can handle multiple objectives and various constraints for 3D ICs at the same time.
- Thermal-aware optimization flow. Most importantly, we propose a novel thermal-aware optimization flow, which chooses the incremental operations automatically rather than manually to cut design cost and attain high-quality results.

II. OVERVIEW OF THERMAL-AWARE INCREMENTAL FLOORPLANNING PROBLEM

The hotspot, with significantly higher temperature than surrounding cooler regions, could reduce chip reliability and lead to catastrophic failure. To effectively eliminate the hotspots, some incremental changes can be used while the original packing does not need to be changed significantly.

Our thermal-aware incremental floorplanning is an iterative optimization flow. The corresponding problem can be described as: Given a multilayer packing with a set of n blocks $M=\{M_1, M_2, \cdots, M_n\}$ in K layers, where w_i and h_i specify the dimensions of block M_i respectively, a set of nets $N=\{N_1, N_2, \cdots, N_m\}$ where N_i, $i=1,2,\cdots, m$ describes the connections between blocks, we want to generate a new packing where the original topological relations between most blocks remain unchanged so that: 1) the max on-chip temperature can be reduced as much as possible; 2) total wirelength and chip area are degraded little compared with the original design.

To mitigate thermal issue, three different incremental floorplanning strategies can be applied:

1. *Growing unit* to allocate more die area around hotspot to reduce max on-chip temperature. In *growing unit*, the power density is decreased proportionally to the increase of the die area, which can effectively reduce temperature increase from the isothermal point according to [5].
2. *Migrating computation* among duplicated blocks. In migrating computation, the hotspot block requires a duplicated one to share computation tasks, which means to halve the power density to reduce the temperature of hotspot block.
3. Moving certain hot block to cooler regions. Why the hotspot comes into being is that the region it locates at usually has high power density and poor heat dissipation. Therefore, moving certain hot block from the hottest region to relatively cooler area, actually can reduce thermal coupling in the hottest region and decrease the max on-chip temperature, since it could reduce power density and improve the heat dissipation of the hottest area,.

It must be noticed that these modifications are just basic incremental changes in 3D floorplanning which waits for the designer to choose. Moreover, just choosing one operation randomly, rather than using an efficient tactic to guide, hardly ensure a desirable result of those multiple objectives. To accomplish a better tradeoff, we roll out an evaluation criterion, say potential gain, to select the most suitable operation to process the iterative flow to mitigate the thermal issue, and optimize area and total wirelength.

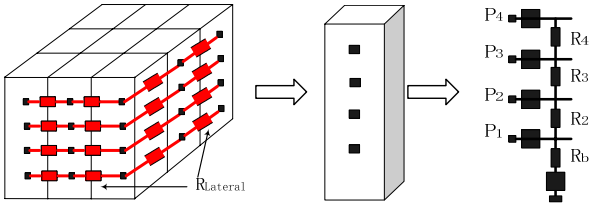
III. THERMAL RESISTANCE MODEL

For temperature profiling, we use the same thermal resistive model as [1]. The 3D circuit is divided by a two-dimensional array of tile stacks, as shown in Figure 3(a). A tile stack is modeled as a resistive network. Each tile stack is composed of several vertically-stacked tiles, as shown in Figure 3(b). These tile stacks are connected by lateral thermal resistances $R_{lateral}$. Within each tile stack, a thermal resistor R_i is modeled for the i-th device layer, while thermal resistance of the bottom layer and silicon substrate is modeled as R_b as shown in Figure 3(c).

The isothermal bases of room temperature are modeled as a voltage source. A current source is present at every node in the network to represent the heat sources. One can spatially discretize the system and solve the following equation to determine the steady-state thermal profile as a function of power profile.

$$T = PA^{-1} \tag{1}$$

where A is an K× K sparse thermal conductivity matrix. T and $P(t)$ are K× 1 temperature and power vectors. K is the number of thermal conduction edges.



(a) Tiles Stack Array (b) Single Tile Stack (c)Tile Stack Analysis

Figure 3 Resistive thermal model for 3D ICs

IV. MILP FORMULATION FOR GENERAL FLOORPLANS

To develop MILP based methods for thermal-aware incremental changes in 3D ICs, multiple objectives and various constraints should be considered at the same time. In this section, we will firstly show how to model these issues in general floorplanning. We use the techniques from [17] in the following subsections A and B.

A. LP model for certain topological Relations

Given a multilayer packing, it is easy to represent the topological relations in linear constraints to prevent overlapping between any pair of rectangular blocks i and j on the same layer, following the techniques in [17]. In the incremental optimization, blocks would deviate from the original packing positions. Let (x_i,y_i) and (x_j,y_j) denote the positions of the lower left corners of block i and j respectively. From the existing floorplan based on certain multilayer representation such as CBA[1] and LTCG[15], we can find the corresponding relative positions of blocks, which keeps unchanged in the optimization process. As a consequence, to prevent overlapping between original blocks i and j on the same layer, one of the following linear inequalities must hold:

$$\begin{aligned} x_i + w_i &\le x_j \quad && i \text{ left to } j \\ x_j + w_j &\le x_i \quad && i \text{ is right to } j \\ y_i + h_i &\le y_j \quad && i \text{ is below } j \\ y_j + h_j &\le y_i \quad && i \text{ is above } j \end{aligned} \tag{2}$$

B. MILP model for uncertain topological relations

If a new block is added to the existing packing, the relative relations between this new block and the old blocks are unknown. To ensure that one of the inequalities in (2) holds such that the new block does not overlap the present blocks, two additional 0-1 integer variables x_{ij} and y_{ij}, which take only either 0 or 1 value, can be introduced as in [17]. Let us define bounding constants B_w and B_h

such that we always have $|x_i-x_j|\le B_w$ and $|y_i-y_j|\le B_h$. Possible choices for B_w and B_h are: $B_w=\sum w_i$ and $B_h=\sum h_i$. Assume block i is the newly added block, we can derive the following constraints:

$$\begin{aligned} x_i + w_i &\le x_j + B_w(x_{ij} + y_{ij}) \\ x_j + w_j &\le x_i + B_w(1 - x_{ij} + y_{ij}) \\ y_i + h_i &\le y_j + B_h(1 + x_{ij} - y_{ij}) \\ y_j + h_j &\le y_i + B_h(2 - x_{ij} - y_{ij}) \end{aligned} \quad (3)$$

Since that x_{ij} and y_{ij} take just either 0 or 1 value, it is obvious that only one of the inequalities in (2) take effect for any combination of values of x_{ij} and y_{ij}. For an instance, if and only if x_{ij}=0 and y_{ij}=0, the first constraint of (2) is applied, which allows block i to be anywhere left to block j. Besides, the remaining three constraints are definitely active for any admissible choices of (x_i,y_i), (x_j,y_j).

In addition, if the new block is allowed to rotate in the incremental floorplanning, an additional binary variable r can be employed to meet this requirement. We can define r as follows: when r=0 the new block is added in its original orientation while r=1 the new block is inserted by rotating 90°. Therefore, (3) can be rewritten as follows:

$$\begin{aligned} x_i + (1-r)w_i + r * h_i &\le x_j + M(x_{ij} + y_{ij}) \\ x_j + w_j &\le x_i + M(1 - x_{ij} + y_{ij}) \\ y_i + (1-r)h_i + r * w_i &\le y_j + M(1 + x_{ij} - y_{ij}) \\ y_j + h_j &\le y_i + M(2 - x_{ij} - y_{ij}) \end{aligned} \quad (4)$$

Where:

$$M = max(B_w, B_h) \quad (5)$$

C. LP formulation of chip area

We propose a new optimization model for chip area. Assume W and H are the width and the height of the original packing respectively, to preserve the initial minimum packing area, additional inequalities for each block i are needed as follows:

$$x_i \ge 0, \quad y_i \ge 0, \quad x_i + w_i \le W, \quad y_i + h_i \le H \quad (6)$$

If the existing floorplan has enough white space to enlarge blocks or add blocks, fixed-outline constraints (6) can be satisfied naturally. However, if the floorplans are tightly packed, the entire chip area might be expanded if some blocks must be enlarged or some new blocks must be inserted. To minimize the area increase, we can introduce two slack variables h_{slack} and w_{slack} as follows:

$$\begin{aligned} &\min \quad w_{slack} * h_{slack} \\ &s.t. \quad w_{slack} \ge 0, \quad h_{slack} \ge 0 \\ &for\ the\ existing\ blocks: \\ &x_i \ge 0, \quad y_i \ge 0, \quad x_i + w_i \le W * w_{slack}, \quad y_i + h_i \le H * h_{slack} \\ &for\ newly\ added\ blocks: \\ &x_i \ge 0, \quad x_i + (1-r)w_i + r * h_i \le W * w_{slack}, \\ &y_i \ge 0, \quad y_i + r * w_i + (1-r)h_i \le H * h_{slack} \end{aligned} \quad (7)$$

where r is the rotation 0-1 variable. Evidently, chip area is minimized if and only if $w_{slack}*h_{slack}$ is minimized. Objective $w_{slack}*h_{slack}$ is not linear, but if we note the inequality:

$$w_{slack} + h_{slack} \ge 2\sqrt{w_{slack} * h_{slack}} \quad (8)$$

which suggests that $(w_{slack}+h_{slack})/2$ is the supremum of $\sqrt{w_{slack}*h_{slack}}$, then we can use the following approximation for area objective: *minimize* $\sqrt{w_{slack}*h_{slack}}$ *(equivalent to minimize* $w_{slack}*h_{slack}$ *here) by minimizing its supremum* $(w_{slack}+h_{slack})/2$. In reality, the difference between the width and the height of the same existing floorplan is so small that this linearization could show good approximation. Even if the constraints allow h_{slack} and w_{slack} to be any large values, the optimal LP solution will ensure that h_{slack} and w_{slack} are set to be the smallest values from these constraints.

D. LP model for wirelength(HPWL)

We model HPWL optimization using model from [5]. Assume the pins are at the center of the corresponding blocks and each net j has four variables x^j_{max}, x^j_{min}, y^j_{max}, and y^j_{min} representing its four boundary edges of the bounding box, (x^i_{pin},y^i_{pin}) represents the pin location of block I, N is the set of nets of the floorplan. Obviously each block i of net j is in the net's bounding box, which ultimately results in LP constraints as follows:

$$\begin{aligned} x^i_{pin} \ge x^j_{\min}, \quad x^i_{pin} \le x^j_{\max} \\ y^i_{pin} \ge y^j_{\min}, \quad y^i_{pin} \le y^j_{\max} \\ x^i_{pin} = x_i + w_i/2, \; y^i_{pin} = y_i + h_i/2 \end{aligned} \quad (9)$$

Total wirelength estimated by HPWL could be minimized as:

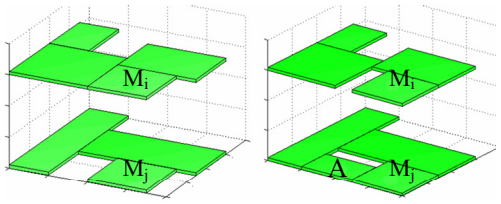
$$\min \quad \sum_{j \in N} (x^j_{\max} - x^j_{\min} + y^j_{\max} - y^j_{\min}) \quad (10)$$

V. MILP BASED THERMAL AWARE INCREMENTAL FLOORPLANNING METHODS

A. Special 3D constrained modifications

Our formulation could provide a flexible way to handle constrained modifications in 3D ICs. Here we demonstrate two kinds of such modifications: adding blocks with alignment constraints and moving blocks between layers.

Adding blocks with alignment constraints: Suppose we have two blocks M_i and M_j in the existing floorplan which must be aligned from high-level design requirement. When a new block is added, this constraint should still be satisfied undoubtedly. As figure 4 shows, new block A does not disturb the alignment of M_i and M_j.



(a) M_i and M_j are aligned (b) insert A while aligning M_i and M_j

Figure 4 adding blocks with alignment constraints

To meet this requirement, we first construct the constraints for uncertain topological relations between the new block and the blocks on target layer. In addition, we set the alignment constraint between M_i and M_j by aligning their lower left corners as:

$$x_i = x_j \quad , \quad y_i = y_j \quad (11)$$

In fact, also we can simultaneously add multiple blocks that have alignment constraints with each other. This constraint is useful especially in such floorplans that have cube blocks, which are packed across several layers in 3D ICs.

Moving blocks between layers: In quite a few cases, blocks should be moved to other layers for better results such as smaller chip area. To handle this modification, we can delete the block from its original layer then construct new constraints for the uncertain topological relations between the moving block and other blocks on target layer. We delete block i by just making its width and height to be zero:

$$w_i = 0 \quad , \quad h_i = 0 \quad (12)$$

B. MILP model for growing unit

Assume the area of block i is enlarged to s_i, allocating more die area to the hot functional unit(*growing unit*) allows w_i and h_i to vary satisfying $w_ih_i=s_i$. This equation in nature is nonlinear and cannot be applied directly in the MILP approach. Similar to [19], we can linearize the nonlinear relation $w_i=s_i/h_i$ by using a set of several linear constraints as shown in Figure 5. Different from [19], which solved LP model iteratively to resize floorplan, we create MILP model to grow the block just once.

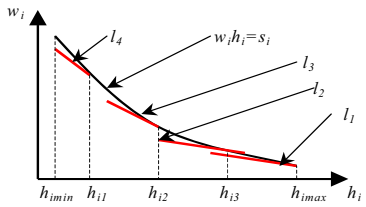


Figure 5 linear approximation of $w_i=s_i/h_i$

Each linear constraint is finished by applying the first two members of the Taylor series for each interval:

$$\begin{aligned} w_i &= \frac{s_i}{h_{i\max}} + (h_{i\max} - h_i)\frac{s_i}{h_{i\max}^2} \qquad if\ h_i \in (h_{i3}, h_{i\max}] \\ w_i &= \frac{s_i}{h_{i3}} + (h_{i3} - h_i)\frac{s_i}{h_{i3}^2} \qquad if\ h_i \in (h_{i2}, h_{i3}] \\ w_i &= \frac{s_i}{h_{i2}} + (h_{i2} - h_i)\frac{s_i}{h_{i2}^2} \qquad if\ h_i \in (h_{i1}, h_{i2}] \\ w_i &= \frac{s_i}{h_{i1}} + (h_{i1} - h_i)\frac{s_i}{h_{i1}^2} \qquad if\ h_i \in (h_{i\min}, h_{i1}] \end{aligned} \tag{13}$$

In order to make sure that only one of above equalities holds in the *growing unit* process, we introduce two additional intermediate 0-1 integer variables a_i and b_i to achieve correspondence between the linear equalities and the intervals of h_i. Firstly we find the one to one correspondence between the intervals and values of a_i and b_i:

$$\begin{aligned} &h_i \le h_{i\max} + M(a_i + b_i) \quad , \quad h_i > h_{i3} - M(a_i + b_i) \\ &h_i \le h_{i3} + M(1 + a_i - b_i) \quad , \quad h_i > h_{i2} - M(1 + a_i - b_i) \\ &h_i \le h_{i2} + M(1 - a_i + b_i) \quad , \quad h_i > h_{i1} - M(1 - a_i + b_i) \\ &h_i \le h_{i1} + M(2 - a_i - b_i) \quad , \quad h_i > h_{i\min} - M(2 - a_i - b_i) \end{aligned} \tag{14}$$

The definition of M is the same as (5). It is easy to see that for each of the four possible choices of (a_i,b_i)=(0,0),(0,1),(1,0),(1,1), only one interval contains h_i, and vice versa. For example, with a_i=0 and b_i=0, only the first couple of inequalities in (14) can take off M, and the remaining constraints are automatically satisfied for any possible value of h_i. That means that h_i can not be located in any other intervals but in $(h_{i3},h_{imax}]$. In the following we could achieve the constraints to satisfy the corresponding relations between intervals and linear functions:

$$\begin{aligned} &w_i \le l_1 + M(a_i + b_i) \quad , \quad w_i \ge l_1 - M(a_i + b_i) \\ &w_i \le l_2 + M(1 + a_i - b_i) \quad , \quad w_i \ge l_2 - M(1 + a_i - b_i) \\ &w_i \le l_3 + M(1 - a_i + b_i) \quad , \quad w_i \ge l_3 - M(1 - a_i + b_i) \\ &w_i \le l_4 + M(2 - a_i - b_i) \quad , \quad w_i \ge l_4 - M(2 - a_i - b_i) \end{aligned} \tag{15}$$

where:

$$\begin{aligned} &l_1 = \frac{s_i}{h_{i\max}} + (h_{i\max} - h_i)\frac{s_i}{h_{i\max}^2} \quad , \quad l_2 = \frac{s_i}{h_{i3}} + (h_{i3} - h_i)\frac{s_i}{h_{i3}^2} \\ &l_3 = \frac{s_i}{h_{i2}} + (h_{i2} - h_i)\frac{s_i}{h_{i2}^2} \quad , \quad l_4 = \frac{s_i}{h_{i1}} + (h_{i1} - h_i)\frac{s_i}{h_{i1}^2} \end{aligned} \tag{16}$$

Now the one-to-one correspondence between intervals and linear functions is created. To demonstrate the relation more clearly, we see a more specific illustration: If h_i is in interval $(h_{i1},h_{i2}]$, due to constraints (14), then the (a_i,b_i) must be (1,0), which suggests that only the third couple of inequalities in (15) take effect, that is $w_i=l_3$. Clearly it is consistent with (13), where if h_i is in interval $(h_{i1},h_{i2}]$, then $w_i=l_3$.

In fact, the accuracy of linear approximation can be improved by using a larger set of linear constraints, which yet consumes longer runtime since more integer variables are required. In our experiment, we have approximated this quadratic function by a set of 4 linear constraints, which still shows both good approximation and runtime performance by simulation results.

C MILP model for moving hot block

Why the hotspots comes into being is that the region it locates at usually has high power density and poor heat dissipation. Therefore, removing the hot block from the hot region to relatively cooler area, which could reduce power density and improve the heat dissipation of the hot area, actually reduce thermal coupling in the hot region thus reasonably can decrease the max on-chip temperature.

The target layer, where the hot block will be moved to, could be any layer including its original layer. This is a special 3D modification discussed previously in subsection V-A: *Moving blocks between layers*. More importantly, the hot block should be separated from the original hot region as far as possible, i.e., to approach the cooler blocks on target layer as close as possible. Here we apply Manhattan distance between the pins to approximate the distance between the hot block and the relatively cooler blocks. Assume C is the set of cooler blocks on coolest region, this process can be formulated as follows:

$$\begin{aligned} \min \quad & E_{hot} = \sum_{i \in C} (hx^i_{\max} - hx^i_{\min} + hy^i_{\max} - hy^i_{\min}) \\ s.t. \quad & x^h_{pin} \le hx^i_{\max}, \quad y^h_{pin} \le hy^i_{\max} \\ & x^h_{pin} \ge hx^i_{\min}, \quad y^h_{pin} \ge hy^i_{\min} \\ & x^i_{pin} \le hx^i_{\max}, \quad y^i_{pin} \le hy^i_{\max} \\ & y^i_{pin} \ge hx^i_{\min}, \quad y^i_{pin} \ge hy^i_{\min} \end{aligned} \tag{17}$$

where (x^h_{pin},y^h_{pin}) and (x^i_{pin},y^i_{pin}) represent the locations of pins of hotspot block and the cooler block, respectively. By minimizing the Manhattan distance, we can make hot block to escape from the hot region. E_{hot} can be used as an estimation for thermal optimization, for it estimates the distance between the block moved and the cooler block.

D. Thermal-aware computation migrating

Migrating computation requires a duplicated unit to share computation tasks with hotspot block, which means half the switching power density so as to reduce the max on-chip temperature[7]. The duplicated block is the clone of the hotspot block. Meanwhile, the nets(paths) that are attached to hotspot block should also be replicated for the new block.

Still, as *moving hotspot* does, when inserted into the chip, the duplicated block also should be separated from the hottest region as far as possible by constraints similar to formulation (17). Besides, the wirelength constraints (9) and (10) should be set up between the duplicated block with those blocks which are among the same nets (paths) with the hotspot block.

E. Simultaneous optimization of multiple objectives

After all relevant MILP models have been completed, we can establish the metrics for these thermal-aware incremental modifications. Multiple objectives, i.e., chip area(A), thermal(E_{hot}), and total wirelength(WL), can be considered as follows:

$$\min \qquad \alpha * A + \beta * E_{hot} + \gamma * WL \tag{18}$$

Where α, β and γ are normalized weight factors.

VI. THERMAL-AWARE OPTIMIZATION FLOW

In order to reduce maximal on-chip temperature as much as possible, we propose a novel thermal-aware incremental optimization flow. An intelligent guidance is applied to pick the best incremental modification to be executed in the iterative flow. To better control the optimization process, we must analyze the temperature contribution from different blocks to the hotspot. Over these temperature increase and block area, we can calculate potential gain for each candidate incremental modification. Finally, it is the operation that could bring the largest potential gain to be selected to carry out.

A. temperature increase from blocks on the same layer

[20] provides an analytical model for thermal profile in 2D chip. This approach models the block as a round region with the same area value as the original block itself, and the radius of the equivalent circularity is estimated by $a=\sqrt{w_ih_i/\pi}$. Based on this assumption, temperature increase from block i to the hotspot could be calculated by:

$$\Delta T_i = \theta(a,r) * P_i * a / k \tag{19}$$

Where,

$$\theta(a,r)=\begin{cases} c_1 I_0(m r^+)+\dfrac{t}{2aB_i} & 0 \le r^+ < 1 \\ c_4 K_0(m r^+) & r^+ > 1 \end{cases} \quad (20)$$

$$c_1=\frac{-t/2aB_i}{I_0(m)+K_0(m)\dfrac{I_1(m)}{K_1(m)}},\ c_4=-c_1\frac{I_1(m)}{K_1(m)}$$

$$r^+=\frac{r}{a},\ B_i=\frac{ht}{k},\ m=a\sqrt{\frac{2h}{kt}}$$

The meanings of the symbols are: r is distance between the hotspot and the center of block i, h is the coefficient of thermal transfer, t is block thickness, k is the thermal conductivity of material per unit volume, P_i is block power, I_0 and I_1 are modified Bessel functions of the first kind while K_0 and K_1 are modified Bessel functions of the second kind. Upon these equations, the thermal effects on the hotspot from those blocks in the same 2D plane can be resolved.

B.Temperature increase from blocks on lower layers

According to the thermal resistance model, blocks in lower layers also affect the thermal distribution in hotspot region. To deal with this case, the estimation for temperature increase would be determined in two successive stages. Firstly, the blocks on the lower layers will be projected to the top layer. In the second stage, temperature increase from the projection block to the hotspot is determined by (19).

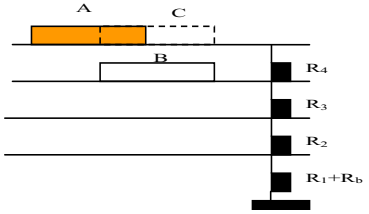


Figure 6 Projection from B to C

Take figure 6 for example, hotspot block A is on top layer, block B is on the sub-top layer while block C, the projection of B on top layer, has the same shape and horizontal position with B. To make these two blocks have the same thermal effects, the power of block C must be determined according to the thermal resistance model as follows:

$$P_C = P_B * \frac{R_b + \sum_{i=1}^{3} R_i}{R_b + \sum_{i=1}^{4} R_i} \quad (20)$$

Now, projection block C is on the same layer with hotspot block A, which indicates that we can use (19) to compute the thermal contribution of block C:

$$\Delta T = P_C * \theta(a,r) * a / k \quad (21)$$

This temperature increase could be considered as contribution from block B.

C. Potential gains for the incremental changes

In the optimization flow, we will consider the following possible incremental modifications as candidate operations: *moving adjacent blocks of the hotspot block, moving the hotspot block, moving the blocks under the hotspot, resizing the hotspot block,* and *migrating computation*. To choose the best candidate that would bring good tradeoff between thermal and chip-area, we must consider the potential gain if certain operation on the block is executed to make it in the ideal location. Since that it is difficult to estimate the influence on wirelengh in this stage, we make total wirelength as one objective to optimize rather than as a criterion for selecting a candidate. Therefore, the potential gain for taking operation on block i could be achieved as follows:

$$g_i = \alpha * \Delta T_i + \beta * \frac{1}{\sqrt{A_i}} \quad (22)$$

Where ΔT is the temperature increase, A_i is the block area(for resizing block, it is the resized area), α and β are normalized weight. This computation will be incorporated into the optimization flow as a guidance to choose the most favorable incremental operation.

D. Optimization flow

In the flow, we first use thermal simulator to find out the position of the hotspot, then calculate the potential gain against those five candidate modification. Afterwards, we do choose the one that can bring maximal potential gain for executing. Mixed integer linear programming(MILP) model is created accordingly. The flow will reiterate above procedures unless objective value cannot be optimized. Following is our iterative flow chart:

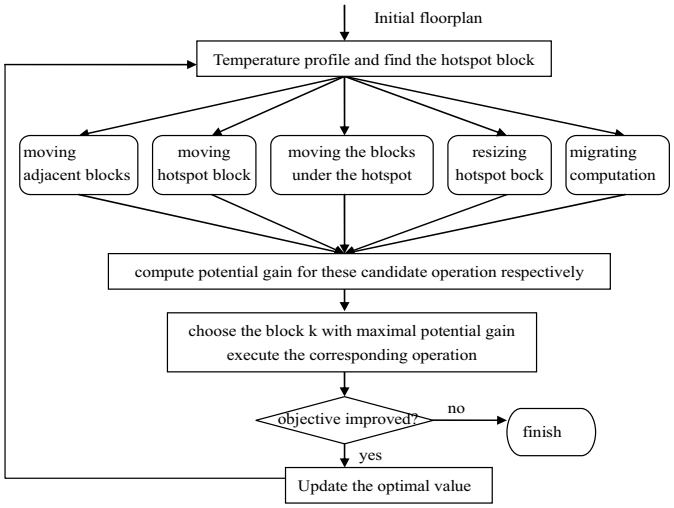


Figure 7 thermal-aware incremental modification flow

VII. EXPERIMENTAL RESULTS

We have implemented the thermal-aware incremental floorplanning in C++ language, and all experiments are performed on a 1.6GHz Intel PC. We solve the MILP problems using a leading LP solver *glpk*[18]. The tile array in each layer is set as 30×30 for each benchmark and the temperature profiles are generated by a fast resistance simulator from[1]. The power dissipations of each block are assigned to the same value with [1](ranging from 10^5W/m^2 to 10^7W/m^2). The ambient temperature is set to be 27°C.

We test the proposed algorithms on MCNC and GSRC benchmarks. All initial floorplans are 4 layers stacked and generated from SA-based 3D floorplanning algorithms CBA[1]. Table 1 shows the parameters of the initial floorplans, including block number, net number, chip area, max temperature and total wirelengh. Many techniques such as *pl2sp*() in Parquet[18] could be used to reconstruct the SP/constraint graphs from the existing placement.

Table 1: Initial Floorplans

	Block#	Net#	Area(um^2)	T_{max}(°C)	WL(um)
Ami33	33	123	342504	462.3	29640
Ami49	49	408	12413700	358.8	187065
N100	100	885	51983	303.7	68899
N200	200	1585	60652	303.4	174921
N300	300	1893	91025	288.7	267944
Avg.			1	1	1

Just as [7] proposed, by migrating computation between two sites, the activity for each block is halved, which means half the switching power density and half the temperature increase from the isothermal point. Hence, to facilitate simulation, we can add a duplicated unit for hotspot block then half the power density of both hotspot block and duplicated block to implement *migrating computation*.

A. Thermal-aware incremental floorplanning methods

We implement those three basic thermal-driven incremental operations: *moving hot block, growing unit and migrating computation*. In the *moving* operation, we only select hotspot block to move since here we just compare these three basic operations and

moving hot blocks of other type will be introduced in subsection B. Each action is executed only once here. Because the initial floorplans are packed tightly, the final packings are all enlarged to facilitate addition and growth of blocks. Table 2 shows the experimental results of our approaches. *Growing unit* enlarges the hotspot block by 3 times on average. As can be seen, *growing unit, migrating computation* and *moving hotspot block* reduce max on-chip temperature by 7%, 13% and 15% respectively. *Moving hotspot* is the best in mitigating thermal issue. *Growing unit* decreases temperature the least, but it brings the slightest area increase and can reduce total wirelength as well. *Migrating computation* can notably reduce max temperature but enlarge total wirelength since the duplicated block introduce extra connections with others blocks.

B. Optimization flow for the 3D chip

We run our flow that includes five possible incremental changes on those floorplans generated from CBA[1]. The flow will not exit until no objective improvement can be achieved, or the chip area is enlarged by more than 10% though it may bring great thermal abatement. Table 3 shows the results of the optimization flow. As can be seen from the table, compared with CBA, our approach can reduce the maximal temperature by about 14%, introducing little time overhead, which shows rapid design convergence. Meanwhile, total wirelength is also decreased by 2%. Because the original floorplans are packed tightly, the chip area is enlarged by 3% and the optimization iterates only a few times according the area constraint. The runtime is mainly spent on solving the MILP formulations rather than invoking the solver. Note that this flow seems to have almost the same thermal optimization effects as *moving hotspot block* does, however, the flow can attain smaller chip area and total wirelength, which shows the effectiveness of the flow to bring better tradeoff.

VIII. CONCLUSION

In this paper, we propose some novel thermal-aware incremental changes to optimize these multiple objectives including thermal issue in 3D ICs. Furthermore, to improve time-to-market via design cycle reduction, incremental design must move from an expert methodology to a mainstream design methodology: one that is automated, integrated, reliable, and repeatable. To avoid random incremental modification, which may be inefficient and need long runtime to converge, here potential gain is modeled for each candidate incremental change. Based on the potential gain, a novel thermal optimization flow to intelligently choose the best incremental operation is presented. We distinguish the thermal-aware incremental changes in three different categories: *migrating computation, growing unit* and *moving hotspot*. Mixed integer linear programming (MILP) models are devised according to these different incremental changes. Experimental results show that *migrating computation, growing unit* and *moving hotspot* block can reduce max on-chip temperature by 7%, 13% and 15% respectively on MCNC/GSRC benchmarks. Still, experimental results also show that the thermal optimization flow can reduce max on-chip temperature by 14% compared to an existing 3D floorplan tool CBA, and achieve better area and total wirelength improvement than individual operations do.

REFERENCES

[1] J.Cong, J. Wei and Y. Zhang, "A Thermal-Driven Floorplanning Algorithm for 3D ICs", in Procceedings of ICCAD, 2004

[2] W. L. Huang, G.M. Link, Y. Xie, N. Vijaykrishnan and M.J Irwin, "Interconnect and Thermal-Driven floorplanning for 3D microprocessors", in Procceedings of ISQED, Mar. 2006

[3] Z.P. Gu, Y. Yang, J. Wang, R.P. Dick and L. Shang, "TAPHS: Thermal aware unified physical-level and high-level synthesis", in Procceedings of ASP-DAC, 2006

[4] P. Zhou, Y. Ma, Z. Li, R.P. Dick, L. Shang, H. zhou, X.L. Hong and Q. Zhou, "3D-STAF: Scalable Temperature and Leakage Aware Floorplanning for Three Dimensional Integrated Circuits", In procceedings of ICCAD, 2007

[5] C.H. Tsai and S.M.S Kang, "Standard cell placement for even on-chip thermal distribution", in Procceedings of ISPD, 1999

[6] K. Skadron, M.R Stan, W. Huang, S. Velusamy, K. Sankaranarayanan D. Tarjan, "Temperature-aware Microarchitecture", in Procceedings of ISCA, 2003.

[7] S. Heo, K. Barr and K. Asanovic, "Reducing power density through activity migration", in Procceedings of ISLPED, Aug., 2003.

[8] T.D. Richardson and Y. Xie, "Evaluation of Thermal-aware design Techniques for Microprocessors", in Proceedings of ASICON, 2005.

[9] J. Cong and M. Sarrafzadeh, "Incremental Physical Design", in Procceedings of ISPD, 2000.

[10] J. Creshaw, M. Sarrafzadeh, P. Banerjee, P. Prabhakaran, "An incremental floorplanner", in Proceedings of GLSVLSI,1999.

[11] S. Liao, M.A. Lopez and D. Mehta, "Constrained Polygon Transformations for Incremental Floorplanning", ACM Trans. On DAES, Vol.6, No.3, July 2001.

[12] X. Tang, R. Tian and M.D.F Wong, "Optimal Redistribution of White Space for Wire length Minimization", In procceedings of ASP-DAC, 2005

[13] X. Li, Y. Ma, X.L. Hong, S. Dong and J. Cong, "LP Based White Space Redistribution for Thermal Via Planning and Performance Optimization in 3D ICs", in procceedings of ASP-DAC, 2008

[14] J. Cong and M. Sarrafzadeh, "Incremental Physical Design", in Procceedings of ISPD, pp.84-92, may, 2000.

[15] H.Y. Jill, E.F.Y Young and R.L.S. Ching, "Block alignment in 3D floorplan using layered TCG", in Procceedings of GLSVLSI, 2006.

[16] www.gnu.org/software/glpk/

[17] S. Sutanthavibul, E. Shragowitz and J.B. Rosen, "An Analytical Approach to Floorplan Design and Optimization", in Procceedings of DAC, 1990.

[18] S.N. Adya, I.L. Markov, "Fixed-outline Floorplanning: Enabling Hierarchical Design", IEEE Trans. On VLSI systems, Vol.11,No.1, pp.1120-1135, Dec.2003.

[19] P. Chen and E.S. Kuh, "Floorplan Sizing By linear Programming Approximation", in Proceedings of DAC, 2000

[20] B. Lall, A. Ortega and H. Kabir, "Thermal Design Rules for Electronic Components on Conducting Boards in Passively Cooled Enclosures", in Proceedings of inter-society Conference on Thermal Phenomerna, 1994

Table 2: Results of different thermal-aware Incremental Floorplannings

	Growing unit				Migrating computation				Moving hotspot block			
	Area(um^2)	T_{max}(°C)	WL(um)	Cpu(s)	Area(um^2)	T_{max}(°C)	WL(um)	Cpu(s)	Area(um^2)	T_{max}(°C)	WL(um)	Cpu(s)
Ami33	364077	409.45	25015	0.3	361883	337.77	33111	1.6s	361883	322.14	30524	1.7
Ami49	12413700	349.63	179368	1.6	12413700	342.15	198711	6.1	12413700	334.39	181969	6.9
N100	53918	290.68	60586	8.6	57431	273.24	72622	21.0	57431	272.78	71043	20.8
N200	65345	273.05	165801	18.4	68362	237.13	179565	85.7	68362	233.37	177421	80.6
N300	93227	275.37	262830	29.0	94050	269.5	257049	256.2	94050	268.42	255435	247.3
Avg.	***1.04***	***0.93***	***0.92***		***1.06***	***0.87***	***1.04***		***1.06***	***0.85***	***1.00***	

Table 3: Results of the iterative optimization flow

Benchmark	CBA						CBA + optimization flow				
	Block#	Net#	Area(um^2)	T_{max}(°C)	WL(um)	Cpu(s)	Area(um^2)	T_{max}(°C)	WL(um)	Cpu(s)	Iteration#
Ami33	33	123	342504	462.3	29640	31	361883	322.1	30524	32.7	1
Ami49	49	408	12413700	358.8	187065	97	12309800	284.6	196600	111	3
N100	100	885	51983	303.7	68899	401	55615	287.6	60946	422.1	1
N200	200	1585	60652	303.4	174921	2273	63736	271.3	165265	2464.9	3
N300	300	1893	91025	288.7	267944	4081	91025	274.0	268107	4592.3	2
Avg.			1	1	1	1	***1.03***	***0.86***	***0.98***	1.09	