On Legalization of Die Bonding Bumps and Pads for 3D ICs

Sai Pentapati
sai.pentapati@gatech.edu
Georgia Institute of Technology
Atlanta, Georgia, USA

Anthony Agnesina
aagnesina@nvidia.com
NVIDIA Corporation
Austin, Texas, USA

Moritz Brunion
moritz.brunion@imec.be
University of Bremen
Bremen, Germany

Yen-Hsiang Huang
yhhuang@gatech.edu
Georgia Institute of Technology
Atlanta, Georgia, USA

Sung Kyu Lim
limsk@ece.gatech.edu
Georgia Institute of Technology
Atlanta, Georgia, USA

ABSTRACT

State-of-the-art 3D IC Place-and-Route flows were designed with older technology nodes and aggressive bonding pitch assumptions. As a result, these flows fail to honor the width and spacing rules for the 3D vias with realistic pitch values. We propose a critical new 3D via legalization stage during routing to reduce such violations. A force-based solver and bipartite-matching algorithm with Bayesian optimization are presented as viable legalizers and are compatible with various process nodes, bonding technologies, and partitioning types. With the modified 3D routing, we reduce the 3D via violations by more than 10× with zero impact on performance, power, or area.

CCS CONCEPTS

• **Hardware** → **Wire routing**; **Placement**; **3D integrated circuits**; *Logic circuits.*

KEYWORDS

3D Integrated Circuits, Face-to-Face Bonding, 3D routing, Via legliza-tion

ACM Reference Format:
Sai Pentapati, Anthony Agnesina, Moritz Brunion, Yen-Hsiang Huang, and Sung Kyu Lim. 2023. On Legalization of Die Bonding Bumps and Pads for 3D ICs. In *Proceedings of the 2023 International Symposium on Physical Design (ISPD '23), March 26–29, 2023, Virtual Event, USA.* ACM, New York, NY, USA, 9 pages. https://doi.org/10.1145/3569052.3578925

1 INTRODUCTION

3D Integrated Circuit Design has been gaining significant momentum commercially in recent years. For example, micro-bumping technology, such as Intel's Foveros [1], aims to improve an SoC's cost and power consumption with block-level 3D IC design implementation. The two tiers are bonded together using micro-bumps of 36 µm pitch to utilize both the high-performance and the low-power nodes without affecting the overall performance. Likewise, with AMD's Ryzen V-Cache 3D IC, hybrid bonding was used to achieve a large L3 cache size and significantly improve system performance [2].

This work is licensed under a Creative Commons Attribution International 4.0 License.

ISPD '23, March 26–29, 2023, Virtual Event, USA
© 2023 Copyright held by the owner/author(s). Publication rights licensed to ACM.
ACM ISBN 978-1-4503-9978-4/23/03…$15.00
https://doi.org/10.1145/3569052.3578925

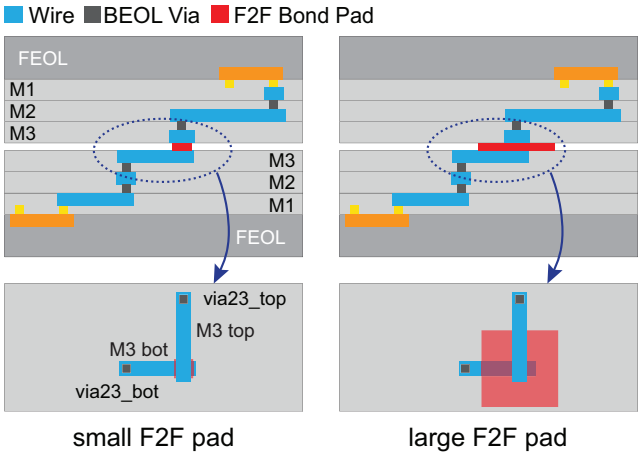

Figure 1: Two cases of Face-to-Face designs with different bond pad sizes. (a) small pads, (b) large pads. The corresponding top-down views are shown at the bottom.

Finer partitioning at an L2 or L1 cache level can transfer such system-level benefits deep within the architecture but require a finer via pitch of 1 µm – 10 µm for the increased connection density [3]. At L2 and L1 level partitioning, the connectivity between the tiers becomes increasingly critical, and flows such as Macro-3D [4] are targeted to optimize such partitioning types.

Monolithic 3D (M3D) IC requires the highest 3D bandwidth of any partitioning type with 3D via pitch of around 0.1 µm [5] and specialized Place-and-Route (PnR) flows such as Pin-3D [6].

A crucial drawback of the current state-of-the-art 3D flows is the routing stage. All the pseudo-3D flows, including the most recent Macro-3D and Pin-3D, assume a Face-to-Face bond pad pitch in the order of 0.1–1 µm. This is mainly because the flows are designed for monolithic 3D integration and are extended to 3D Face-to-Face (F2F) wafer bonding using similar assumptions. However, current research suggests that sub-micron pitch values for 3D wafer bonding pads are not easily realizable and can present yield and manufacturability issues [5, 7, 8].

In works such as [4, 6], the combination of the smaller pitch values and the 28 nm process node used by authors obscures the placement problem of the 3D via due to their large size. Fig. 1 shows how 3D net routing is impacted using a realistic F2F bond pad pitch is used. In Section 2, we further analyze this phenomenon with actual design implementations. The problem arises when the via

Table 1: Terminologies used in this paper.

3D via	Vias connecting the two metal layers to be bonded using micro-bumps or hybrid bond pads. We call them "vias" instead of "bumps" or "pads" because they are added during routing.
Cut Spacing	Edge-to-Edge spacing for a via (cut) = minimum cut spacing across all four via edges.
Pitch	Min required Center-Center distance (=width + spacing) as defined by the technology.
Cut Distance	Center to Center distance between two cuts. Here, L∞-norm (Chebyshev distance) is used as the distance measurement. Given two vias centered at (x_1, y_1) and (x_2, y_2), the cut distance or L∞-norm is $\max(\lvert x_2 - x_1\rvert, \lvert y_2 - y_1\rvert)$.
Cut Overlap	When Cut Distance < Pitch, we call the vias/cuts to be overlapping. This is when cut spacing violation occurs between the two vias.

Table 2: 3D via overlaps generated by state-of-the-art 3D flows. The 3D designs with Macro-3D and Pin-3D have 3K and 50K 3D vias, respectively. OVL/OCC = overlap/occupancy.

Macro-3D [4] Memory-on-Logic			Pin-3D [6] Logic-on-Logic		
Pitch	# OVL	% OCC	Pitch	# OVL	% OCC
1 µm	0	0.3%	0.1 µm	0	0.3%
2 µm	2	1.1%	0.2 µm	0	1.3%
5 µm	1410	7.2%	0.5 µm	0	8.9%
10 µm	11080	28.2%	1.0 µm	5315	32.2%

layer connecting the two 3D ICs has a significantly larger pitch than the connecting metals. Additionally, as metal pitch and overall footprint shrink with process technology node advancement, the cut spacing violations start appearing at smaller pitch values. A more detailed discussion of this effect is done in Section 2.2.

We present two legalization algorithms to remove via overlaps produced by commercial routers. First, Section 4 presents a force-based algorithm that moves only the problematic 3D vias to remove overlaps. Second, Section 5 presents a bipartite-matching algorithm where the vias are optimally assigned to legal locations. Moreover, the parameters that dictate the assignment are tuned with Bayesian optimization to achieve global optimality. These methods, especially the tuned bipartite matching, are robust and applicable to different 3D partitioning, bonding, and technology nodes.

2 ISSUE ANALYSIS

This section discusses the main issue tackled in this paper, and Table 1 provides the terminologies used throughout.[1]

2.1 What Is the Issue? How Serious Is It?

With Macro-3D [4], the application processor is partitioned to have L2 and L1 Data RAMs in the memory tier, and everything else (L1 instruction cache, logic blocks, and other caches) is on the logic tier. Based on the 3D manufacturing methods, the via pitch can vary independently of the technology node. By sweeping the hybrid bond pitch from 1 µm to 10 µm [5], the number of cut overlaps increases gradually, as shown in Table 2. The via utilization is given in the via_{util} column and shows the percentage of vias used compared to the maximum number of vias. Since the maximum number reported here does not consider the timing or the manufacturability impact, it is essential to note that the via utilization should be kept reasonably small (~ 50-60%) as it can affect redundancy and yield of 3D vias.

[1] A special attention is to be paid on how we treat micro-bumps or hybrid bond pads as "vias".

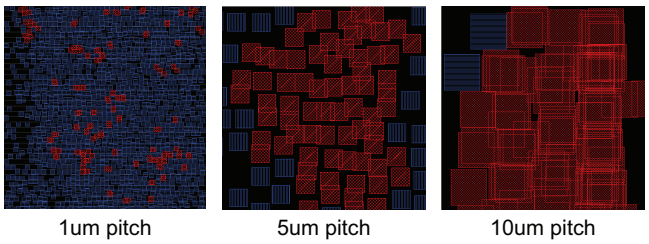

Figure 2: Via overlaps (shown in red) at various pitch values.

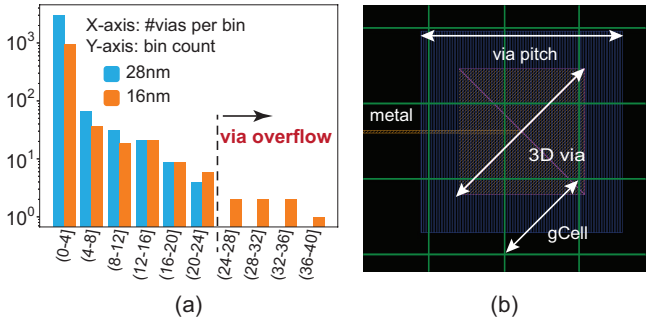

Figure 3: (a) Via distribution of a design in two different process nodes. Each bin is 25 µm × 25 µm, (b) Global cell grid (in green), 3D via, and an M6 metal layer in a 28 nm design.

The Application Processor is implemented up to the CPU level (no L2 cache) with logic-on-logic partitioning using Pin-3D flow [6]. The logic-on-logic partitioning is done at a finer level than memory-on-logic, resulting in many connections between the two tiers. Due to this, the supported via pitch values are an order of magnitude smaller than the memory-on-logic design.

The routing stage in Pin-3D is similar to other flows, such as Compact-2D [9], upon which it is developed. Therefore, the automatically inserted 3D vias that violate cut spacing or cut short design rules also exist in these flows.

Fig. 3(a) shows the difference in via densities of a design implemented in different technology nodes. The 3D via pitch value is set to 5 µm, and the total number of 3D vias is ~ 1200 in both cases as the design and partitioning are the same. Therefore, there can only be 25 vias placed legally in the 25 µm bin. However, we see that the 16 nm design contains bins with significantly more vias shown as via overflow bins in Fig. 3(a).

2.2 What Causes This Issue?

Routing in a physical design using a commercial router is separated into global routing, track assignment, and detail routing. First, the

Table 3: Comparing BEOL dimensions in the 28 and 16 nm nodes. The metal layer (Mx) is directly beneath the 3D via.

Metric (Unit)	28 nm	16 nm
Mx pitch (µm)	0.10	0.08
Via pitch below Mx (µm)	0.10	0.08
3D Via pitch (µm)	5.00	5.00
gCell width (µm)	1.48	1.08

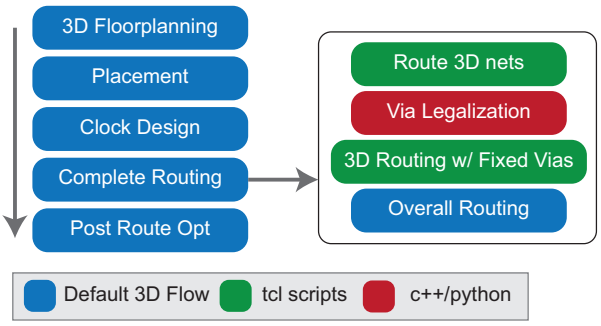

Figure 4: Typical 3D IC design flow, and our modifications to the routing stage for via legalization.

entire footprint is divided into several global cell (gCell) grids during global routing, and nets are assigned to these grids based on the capacity of the grids. This is a much simpler problem than assigning nets to tracks directly due to the much smaller number of gCells. Detail routing then generates an exact physical routing solution for every net by assigning nets to tracks within the gCells.

Generally, 10-15 tracks of lower metal layers are assigned per gCell. The nets are assigned to gCells based on the track utilization of these gCells and the wire length and delay estimations of the nets. During the gCells' track utilization and capacity estimation, commercial routing engines (Innovus/IC Compiler II) only consider the width and spacing of metal layers, ignoring the via dimensions.

In traditional 2D designs, as vias have similar dimensions to the nets, placing the nets and vias during track assignment does not disrupt the global routing solution. However, in Face-to-Face 3D design, the 3D bond pad pitch can be several orders of magnitude larger than the metal pitch and even larger than the gCells. As a result, many violations occur at this stage, which must be resolved during later detailed routing. Fig. 3(b) defines the various routing dimensions, and Table 3 gives their values for 28 nm and 16 nm commercial process nodes. As the gCell grid size decreases with dimensional scaling in advanced technology nodes, the same 3D via pitch would lead to increased routing issues. Similarly, a larger 3D via pitch worsens the 3D via overlaps for a given process node.

3 PROPOSED 3D ROUTING FLOW

In the state-of-the-art 3D flows [4, 6, 9], the implementation environment contains the entire metal layer stack of both 3D tiers and results in a better routing quality than a die-by-die implementation. To remove any possible via overlaps, we modify the routing stage with an added via legalization stage, as shown in Fig. 4.

Instead of routing all nets at once, the 3D nets are first routed to get an initial solution for the 3D via placement. Then, the vias' final

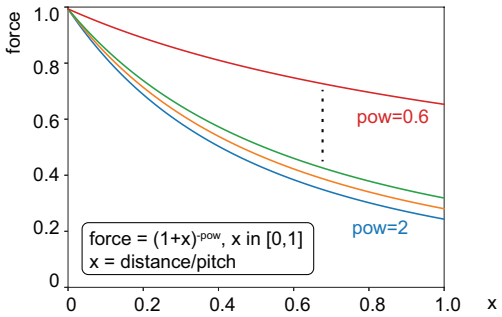

Figure 5: Family of force equations used. The power term is decreased from 2 to 0.6 as iterations increase to amplify the importance of vias near the pitch boundary.

locations are generated in the via legalization step of Fig. 4 using one of the two methods discussed later. Finally, the updated via positions are applied, for example, by using the *editMove* command capability supported by the Cadence Innovus PnR tool. This step only moves the vias without regard to the overall net routing.

The vias are marked as fixed in their new locations, and the 3D nets are rerouted using a commercial EDA native router. This forces the router to use the updated via locations rather than adding new vias. All the other layers of the net would be rerouted to have proper connectivity. Additional cut layer blockage on the 3D via layer is added to discourage the addition of any new 3D vias during routing that would not be legalized. After routing the 3D nets, all the other nets are routed with the cut layer blockage intact.

4 FORCE-BASED VIA LEGALIZATION

Force-directed placement is a popular algorithm for cell placement in the literature [10–12]. Based on this approach, we propose a force-based solution to remove the overlaps in the 3D via layer. Traditional force-based algorithms for global placement move the cells to an equilibrium position by solving for the forces acting on each object. For example, a repelling force moves cells away from each other, while an attractive force allows for wire length minimization and spreads cells toward low-density areas. In our flow, we choose a numerical approach of the force solver by incrementally moving the vias in small discrete steps.

4.1 Algorithm

The force-based legalizer starts with an initial solution from the commercial router. At this stage, we suppress the violation fixing step of the router, as this task is done with the force solver. The router optimizes for various design considerations such as wire length, timing criticality, congestion, and many other metrics in the initial via placement to find a good routing solution. After the 3D net routing, the following actions are performed in each iteration of the force-based solver to remove the overlaps.

4.1.1 Repulsive forces. For each overlapping via pair, we introduce two equal and opposite forces on the vias along the direction of the line joining their centers. Only the vias within the overlap neighborhood of each via are considered for force interactions to minimize the run-time. The overlap neighborhood of a via is

a rectangle of $width = pitch_x$, and $height = pitch_y$ centered at the center of the via. The blue rectangle in the zoomed-in shot of Fig. 3(b) shows the overlap neighborhood of the via. The force $vs.$ distance relation is given as

$$F \propto (1+x)^{-p},\ x = \frac{\text{distance}}{\text{pitch}}$$

where pitch can be varied independently along the two directions. As the iterations progress, the power term (p) gradually decreases from 2 to 0.6, increasing the effect of vias closer to the boundary of the overlap neighborhood. The horizontal and vertical forces are calculated based on the separation along respective directions.

We chose $F \propto (1+x)^{-p}$ rather than the more traditional x^{-p} to avoid infinities. Additionally, the force function is not smooth ($F' \to 0$ as $x \to 1$) as the impact of vias close to the overlap boundaries is significantly reduced with such functions, requiring more iterations and run-time to remove overlaps.

4.1.2 Attractive forces. A small attractive force per via is also added that pulls it towards its initial location. This helps reduce the vias' maximum displacement and reclaim excess spacing between vias. The magnitude of this force is reduced exponentially with the number of iterations so that new overlaps are not formed late in the force run when vias become mainly legalized.

4.1.3 Via movement. Once the forces are calculated, the vias are moved proportionally. Consider an object of mass m starting from rest with a resultant force F acting on it. In a time interval t, it moves a distance of $\frac{1}{2}at^2$ where $a = \frac{F}{m}$. So the displacement is $\propto F$, and the proportionality constant varies with mass m of the object when t is fixed. The mass of each movable signal via is assumed to be the same. Fixed vias and clock net vias are assumed to be 10× heavier than a normal via, so they only move 0.1× the distance of a normal via under the same force. Making these vias heavier rather than leaving them immovable allows us to remove any initial overlaps between fixed vias.

4.1.4 Overall run. The force solver is called multiple times during a run until all the violations are removed. Instead of running the solver with the actual pitch value, we start with a smaller pseudo pitch value of 0.1 µm. Only the vias in a close neighborhood interact and are legalized up to the pseudo-pitch value in the early stages. The pseudo-pitch is then increased by 1.1× until the actual pitch is reached, removing all the true manufacturing violations.

4.1.5 Example. Consider a design with 5000 3D vias and a pitch value of 2 µm. Instead of solving for pitch=2 µm, we start the force solver with pitch=0.1 µm. So, if there are 100 pairs of overlapping vias within a 0.1 µm distance, we run the force-based solver with this pseudo pitch until these violations are under 10. Then, the pseudo-pitch is increased to 0.11 µm and solved until these new violations are almost all resolved. Likewise, the pseudo pitch keeps increasing to 0.121, 0.1331, ..., up to 2 µm (actual 3D via pitch of the design), at which stage we solve to remove the overlaps altogether.

4.1.6 Runtime Analysis. In each iteration of the force solver, we loop over each via as victim, totaling n loops, where n is the number of vias. For each via, we loop over all its neighbors. In the worst case, the number of neighbors is $\propto n$, and each iteration takes $O(n^2)$. In reality, the number of neighbors is much smaller. With an average of m neighbors per via, each iteration takes $O(n \cdot m)$. The number of iterations k varies based on the distribution, severity, and the number of via overlaps in the design. Overall, the run-time can be given by $O(k \cdot n \cdot m)$ or $O(k \cdot n^2)$ in the worst case.

5 MATCHING-BASED VIA LEGALIZATION

While force-based legalization is a more traditional method, it cannot produce a viable solution if the copper pads or bumps are to be arranged on a uniform grid to yield a regular manufacturing bonding pitch. In such cases, we cast the legalization problem into a combinatorial optimization problem of assigning vias to a grid spaced uniformly with the via pitch. The grid intersections are legal via placement points. Starting from an initial solution where vias overlap, we find a legal via assignment that minimizes the total displacement by solving a minimum weighted bipartite matching problem. Even in cases where vias do not need to be assigned on a grid, using a grid is beneficial when the 3D via manufacturing grid differs from that of the design or improves the manufacturability of 3D vias/bond pads.

However, due to a large number of vias and available grid points to assign to, it is computationally and runtime-wise only possible to solve the problem directly by reducing its complexity. Therefore, we propose a windowing technique tuned using Bayesian optimization to reach feasible and close-to-optimal solutions.

5.1 Algorithm

We see legalizing vias to the manufacturing grid while minimizing a cost metric (here, the total via displacement) as an assignment problem on a bipartite graph. Our goal is to uniquely match the set of vias $\mathcal{S}_V$ to the set of grid points $\mathcal{S}_G$, where the cost of matching a particular via v to a particular grid point g is proportional to their Manhattan distance D: $c_{v,g} \propto D(v,g) = |v_x - g_x| + |v_y - g_y| \in \mathbb{R} \cup \{\infty\}$, where (x, y) correspond to the 2D locations of the points in the via layer. Typically, this is an unbalanced problem as $\mathbf{card}(\mathcal{S}_V) < \mathbf{card}(\mathcal{S}_G)$, which adds complexity. However, we transform it into a balanced one by adding enough dummy vias with zero displacement cost to any grid point.

Formally, the goal is to find the matching M minimizing $\sum c_{v,g}$ $\forall (v,g) \in M$. To solve this minimum cost (weight) perfect matching problem, we rewrite it as a linear assignment problem (LAP) as:

$$\begin{aligned} \min & \sum_{v,g} c_{v,g} x_{v,g}, \\ \text{s.t.} & \sum_{g} x_{v,g} = 1, \quad v \in \mathcal{S}_V, \\ & \sum_{v} x_{v,g} = 1, \quad g \in \mathcal{S}_G, \\ & x_{v,g} \geq 0, \quad v \in \mathcal{S}_V, \quad g \in \mathcal{S}_G, \end{aligned} \tag{1}$$

where $x_{v,g} = 1$ if $(v,g) \in M$ and 0 otherwise. We solve this problem using the shortest augmenting path algorithm [13]. Figure 6 depicts the transformation of the geometric problem to LAP represented in a matrix form, input to the shortest augmenting path algorithm.

5.1.1 Extension to Timing Driven. Restricting the legalization displacement of 3D vias on the clock signal is crucial to minimize

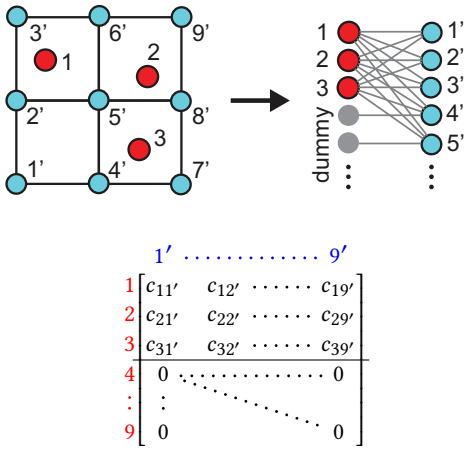

Figure 6: Our high-level grid assignment formulation. Vias (in red) and manufacturing grid points (in blue) are transformed into a bipartite graph, whose pairwise distances form the weight matrix, input to the LAP solver.

Algorithm 1: Windowed Bipartite Matching Algorithm

Data: (x, y) locations of 3D vias from routed design; floorplan boundary; horizontal/vertical pitches of 3D via manufacturing grid; window definition;
Result: A manufacturing grid via assignment minimizing the total timing-driven displacement cost;
for $w \in$ *Windows* **do**
1. Query vias $\in w$ and build grid in that window;
2. Compute pairwise distances, and multiply with pre-computed timing weights to obtain the cost matrix;
3. Solve the LAP with the shortest augmenting path algorithm [13];
4. Apply the assignment solution: update locations of vias and recompute query matrix;

possible PPA degradation. Moreover, vias associated with unconstrained nets connected to, for example, TIE cells, are not as critical as the other vias. Therefore, we propose to weigh the matching cost by the timing-criticality of the connected net based on the net type and static timing analysis. We employ a standard additional net/via weight factor used extensively in timing-driven placement [14]. Per via v, we extract the worst timing path through v and define the weight based on the obtained slack and data arrival time as

$$w(v) = \begin{cases} 2^{\alpha} & \text{if clock net,} \\ \epsilon \ll 1 & \text{if unconstrained net,} \\ 1 & \text{if slack}(v) \geq 0, \\ \left(1 - \frac{\text{slack}(v)}{\text{arrival}(v)}\right)^{\alpha} & \text{otherwise,} \end{cases} \tag{2}$$

where α is the criticality exponent (=2 in our experiments). The new LAP formulation is then updated to use $c_{v,g} = w(v) \cdot D(v, g)$.

5.1.2 2D Windowing. The shortest augmenting path algorithm has a time complexity of $O(\max(\textbf{card}(\mathcal{S}_V), \textbf{card}(\mathcal{S}_G))^3)$. Moreover, the space complexity of the problem is dominated by the size of the cost matrix of $\textbf{card}(\mathcal{S}_V) \times \textbf{card}(\mathcal{S}_G) \times 8$, where 8 is the number of bytes to encode *float64* weights. Thus, even with modern machines, the required memory can easily exceed the RAM capabilities.

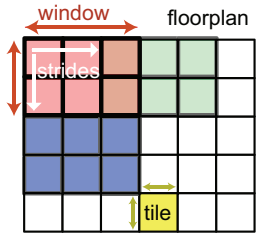

Figure 7: Window-based 2D floorplan grids. In each window, the grid assignment problem is solved optimally. There are multiple legal via locations (grid points) within each tile.

Table 4: The six windowing parameters tuned with Bayesian optimization. The 3D via pitches are noted as p_x, p_y. Window and striding parameters units are set in tile width and height.

Name	Type	Range	Default Value
Tile width (um)	float	[5 p_x, 100 p_x]	30 p_x
Tile height (um)	float	[5 p_y, 100 p_y]	30 p_y
Window size x	int	[3, 10]	3
Window size y	int	[3, 10]	3
Horizontal stride	int	[1,3]	2
Vertical stride	int	[1,3]	2

Therefore, we propose a tiling/windowing method for 2D floorplan/space partitioning to reduce matrix size and solve the LAP locally in each window, following Algorithm 1. This window is slid over the 2D floorplan, similar to a 2D convolution filter. First, we partition the whole floorplan canvas into small tiles. A window is then defined as a rectangle of tiles. Each window will likely contain less than a few thousand vias or grid points for appropriate window sizes, making the problem tractable as we only build the cost matrix and grid points locally. Then, we update the via locations for each window based on the found assignment. Moreover, to counteract the non-optimality introduced by the solutions being only optimal inside the given window, we do not fix the vias after they have been assigned and use striding to allow reassignment of previously derived via locations if it reduces the total displacement.

5.2 Parameter Tuning with Bayesian Optimization

The quality of the assignment significantly depends on the values of the windowing parameters presented in Table 4. These correspond to the window configuration in Figure 7. For example, for a small problem size, the window can be defined to include the entire floorplan, and an optimal solution can be found directly. However, these parameters must be tuned for more complex problems to obtain near-optimal solutions within a reasonable runtime. This objective is realized in the maximization of the following:

$$f(p) = w_C \tanh\left(\frac{C_0}{C(p)}\right) + w_D \tanh\left(\frac{D_{max_0}}{D_{max}(p)}\right) + \tanh\left(\frac{T_0}{T(p)}\right), \tag{3}$$

Table 5: Displacement metrics before and after ten Bayesian optimization iterations. The design has around 6K vias to be legalized on a 5 µm pitch grid. Weights w_C = 20, and w_D = 10.

Metric	Before Tuning	After Tuning
Total cost	9469.1	9249.9 (−2.3 %)
Maximum displacement	21.7	16.2 (−25.3 %)
Runtime (sec.)	1.76	4.02

where p denotes the parameter settings, C denotes the total timing-driven displacement cost, D_{max} is the maximal displacement and T is the runtime. We integrate the maximal displacement to reflect the maximum deviation from the router's initial decision. We set $f(p) = 0$ if the LAP solver crashes due to a runtime exception from the inability to allocate enough memory for the cost matrix or shortest path algorithm. The reference values subscripted with 0 are set based on the default parameter values. The application of the tanh is to squash the differently scaled metrics into $[-1, 1]$ and make them comparable. The weights of each component can be set to realize different trade-offs of optimality vs. speed.

To maximize this objective, we use Bayesian optimization [15]. The Bayesian algorithm sequentially queries the function f and builds a surrogate function interpolating the evaluations. We use the Gaussian process as a surrogate family with a squared exponential kernel. Moreover, we use the Upper Confidence Bound (UCB) acquisition function to pick the next candidate query point. After multiple iterations, we report and use the assignment that maximized the presented objective function. Table 5 shows the positive effect of the tuning on the maximal displacement.

5.2.1 Implementation. We implement the flow in Python, based on Numpy vectorized features, and accelerate the cost matrix calculation with multithreading and SIMD through Numba just-in-time compilation. Moreover, to speed up the query of points in a given window, rather than use traditional 2D spatial query data structures, such as quadtrees or KD-trees, we store the indexes of the list of vias in a 2D matrix Q where $Q[i][j] = \{vias \in \text{tile}(i, j)\}$. Using this matrix Q to query points is much faster than KD-trees due to the regular memory accesses. Moreover, the matrix is quickly updated locally whenever the via locations are changed. In addition, the Bayesian optimization is done using a Python library [16].

6 RESULTS AND ANALYSIS

6.1 Experimental Setup

6.1.1 Technology Setup. To test our via legalizer's efficiency and applicability, we use two commercial PDKs: a 28 nm node and a 16 nm node. Along with the process nodes, the following bonding styles and pitch combinations are also tested: A micro-bump-based 3D IC with 20 µm and 10 µm pitches and a hybrid-bond-based 3D IC with 5 µm and 1 µm pitches.

6.1.2 Place/Route Flows. The micro-bump 3D ICs are designed using a die-by-die flow by pre-assigning the bump locations during the 3D floorplanning stage. Without an initial routing solution for displacement minimization, we start by assigning the bumps to the center of macro pins connected by each 3D net. The displacement is minimized from this center, and the bumps are assigned to a grid. The hybrid bond flows are implemented using Macro-3D flow for memory-on-logic partitioning and Pin-3D flow for the logic-on-logic partitioning, as discussed in Section 2.

6.1.3 Partitioning Types and Benchmark. For the memory-on-logic partitioning, we implement the following circuits: 1. A dual-core application processor (AP1) with 512 kB of L2 cache implemented in the 28 nm node. The memory tier contains the L2 and L1 data cache. 2. A single-core processor (AP2) with 1 MB of L2 with 28 nm PDK. Only the L2 data cache in the memory tier with a cut-size of ∼ 1500. 3. A slight variation of AP2 (referred to as AP3) with 512 kB is implemented in the 16 nm PDK due to the different scaling factors of the memory and logic cells.

The following circuits are designed with logic-on-logic partitioning using the Pin-3D flow. The application processors AP1 and AP2 without the L2 cache (referred to as AP4 and AP5, respectively) are implemented in 28 nm node. And, an NPU with 128 MACs (NP1) is used for the 16 nm node implementation.

6.2 Memory-on-Logic with Hybrid Bonding

The memory-on-logic designs with hybrid bonding are given in Table 6. A hybrid bonding pitch of 5 µm is used for the three designs, with equal width and spacing of 2.5 µm each.

We see from Table 6 that both force-based and bipartite matching algorithms removed nearly all spacing violations in the three cases studied. Furthermore, the overall wire length is largely unaffected, as only a few 3D nets exist with memory-on-logic partitioning. Moreover, the number of vias is smaller in the force, bipartite methods as the modified routing flow in Fig. 4 suppresses the usage of 3D vias by nets connecting to pins within a single tier.

The highlighted path in Fig. 9 shows the critical timing path in the two variations. Both timing paths are across the same hierarchies, and no new timing bottlenecks are created due to the legalization. Similarly, the clock tree layouts in Fig. 10 also show a similar picture between the two routing styles. This further indicates that the via legalization stage does not affect the overall design qualities.

6.2.1 Variations in the via assignment pattern. From Fig. 8(b), we see that the via placement of the force-based legalizer results in a more spread-out solution than Fig. 8(c). This is also supported by comparing the maximum displacement metrics in Table 6. In addition to denser packing with bipartite matching, aligning the vias to a wider 5 µm grid can ease via alignment during manufacturing. On the other hand, force legalization only snaps vias to the smallest manufacturing grid and requires higher alignment accuracy. Even with the wider alignment grid, the bipartite solution could match or improve upon the displacements from the force-based solution. The grid-based solver resulted in a better via legalization solution due to the bipartite matching algorithm's optimality compared to the force solver's heuristic-based displacements.

6.3 Memory-on-Logic with Micro-bumping

Table 7 shows the results of the three designs implemented with the micro bump bonding assumption. As micro bump bonding flows generally require vias to be pre-placed on a custom grid and are not placed by the router, a force-based legalizer cannot be applied. Here we compare the assignment of bumps using bipartite grid

Table 6: Via Legalization results of Memory-on-Logic 3D IC with 5um pitch hybrid bonding.

Implementation Details				Physical Stats			Timing and Power			Legalizer Stats			
Node	Circuit	Legalizer	Freq.	Area	WL	via_{util}	WNS	TNS	Power	#Vias	#Viols	d_{max}	d_{avg}
Units			GHz	mm^2	m	%	ns	ns	mW			µm	µm
28 nm	AP1	None	1.25	1.11	11.62	11.3	-0.119	-202.1	677.0	5014	3538	-	-
		Force	1.25	1.11	11.76	7.4	-0.091	-113.7	678.7	3246	0	29.9	5.3
		Bipartite	1.25	1.11	11.75	7.4	-0.094	-120.4	678.5	3246	0	31.6	3.0
28 nm	AP2	None	1.25	1.89	18.92	3.1	-0.251	-681.1	894.7	2310	893	-	-
		Force	1.25	1.89	18.89	1.8	-0.240	-605.5	895.5	1343	0	12.4	1.3
		Bipartite	1.25	1.89	18.89	1.8	-0.244	-578.9	894.7	1343	0	6.8	2.4
16 nm	AP3	None	1.60	0.605	9.439	5.2	-0.011	-0.057	843.6	1272	2339	-	-
		Force	1.60	0.605	9.455	5.2	-0.013	-0.055	843.7	1243	308	44.2	10.6
		Bipartite	1.60	0.605	9.584	5.2	-0.002	-0.002	844.6	1269	10	15.0	2.8

Table 7: Memory-on-Logic 3D IC with micro-bumping. We use 20um pitch for AP2 benchmark, and 10um for AP1, AP3 due to the smaller footprints. There are no pre-legalization results here, as the micro-bumps are fixed during floorplanning.

Implementation Details				Physical Stats			Timing and Power			Legalizer Stats			
Node	Circuit	Legalizer	Freq.	Area	WL	via_{util}	WNS	TNS	Power	#Vias	#Viols	d_{max}	d_{avg}
Units			GHz	mm^2	m	%	ns	ns	mW			µm	µm
28 nm	AP1	Greedy	1.25	1.11	11.56	53	-0.118	-25.15	796.6	2957	0	601.0	127.5
		Bipartite	1.25	1.11	11.41	53	-0.014	-5.04	794.4	2957	0	385.1	68.1
28 nm	AP2	Greedy	1.25	2.18	20.06	43	-0.323	-1336.4	912.1	1187	0	435.7	160.6
		Bipartite	1.25	2.18	19.92	43	-0.295	-1237.6	907.9	1187	0	198.1	79.3
16 nm	AP3	Greedy	1.60	0.61	10.39	39	-0.155	-881.7	1043	1169	0	326.6	121.1
		Bipartite	1.60	0.61	9.92	39	-0.140	-835.8	1006	1169	0	102.2	44.7

Table 8: Via Legalization results of Logic-on-Logic 3D IC with 1um pitch hybrid bonding. As many violations are unresolved, we see that Face-to-Face is not a viable bump strategy for Logic-On-Logic designs.

Implementation Details				Physical Stats			Timing and Power			Legalizer Stats			
Node	Circuit	Legalizer	Freq.	Area	WL	via_{util}	WNS	TNS	Power	#Vias	#Viols	d_{max}	d_{avg}
Units			GHz	mm^2	m	%	ns	ns	mW			µm	µm
28 nm	AP4	None	1.50	0.25	3.643	23	-0.019	-0.450	350.5	58039	5315	-	-
		Force	1.50	0.25	3.709	21	-0.014	-1.057	351.5	53416	554	5.9	0.6
		Bipartite	1.50	0.25	3.698	21	-0.023	-2.220	350.7	53347	582	4.5	0.5
28 nm	AP5	None	1.50	0.59	9.32	36	-0.032	-15.8	884.3	214123	19992	-	-
		Force	1.50	Not Feasible due to high density of vias, and large number of overlaps.									
		Bipartite	1.50	0.59	10.05	38	-0.122	-438.1	915.0	222465	1900	3.1	0.5
16 nm	NP1	None	1.60	0.12	2.61	40	-0.466	-276.2	390.5	57936	33971	-	-
		Force	1.60	0.12	3.01	36	-0.782	-1157.4	401.5	51414	19573	31.9	6.4
		Bipartite	1.60	0.12	2.77	38	-0.853	-854.5	397.7	54619	17378	28.6	0.9

assignment with a simple priority greedy algorithm based on timing order. Due to the smaller footprints, a bumping pitch of 20 µm is used for the AP2 benchmark and a 10 µm pitch for AP1 and AP3.

A greedy approach creates large displacements for bumps with the lowest assignment priority and is reflected in the max displacement values in Table 7. The optimal placement with a bipartite matching solution provides a much better result, even considering the timing. Compared to a greedy solution, we see a significant improvement in the Total and Worst Negative Slacks with the bipartite matching assignment. This shows our proposed solution's robustness to hybrid bonding and micro-bumping 3D designs.

6.4 Logic-on-Logic with Hybrid Bonding

Finally, Table 8 shows the results for via legalization in logic-on-logic partitioning with a 1 µm 3D via pitch. With a huge 3D via count, the legalization starts to degrade the design quality as the 3D connectivity complexity increases. Due to their extreme pitch requirements, logic-on-logic designs with Face-To-Face bumping

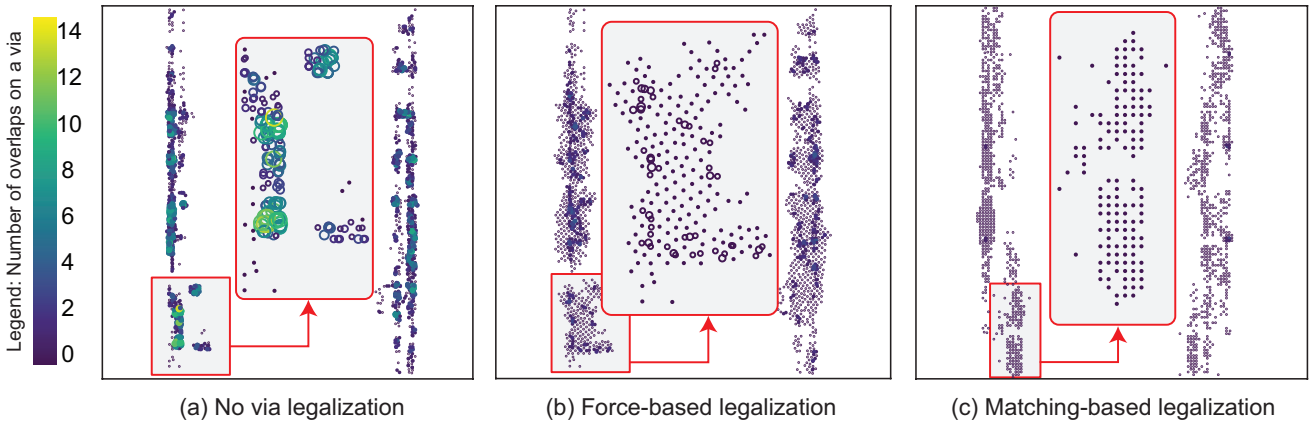

Figure 8: Via assignment under the three different legalization methods for the AP3 benchmark implemented with 16 nm node. The size and color of each via represent the number of overlaps. The gray rectangle shows the zoom-in on a few vias.

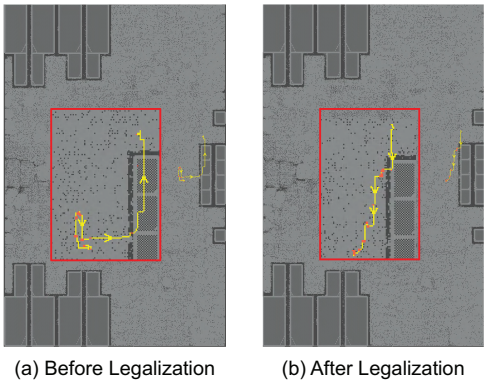

Figure 9: Critical Paths in the Cortex-A7 design. Routing patterns are similar.

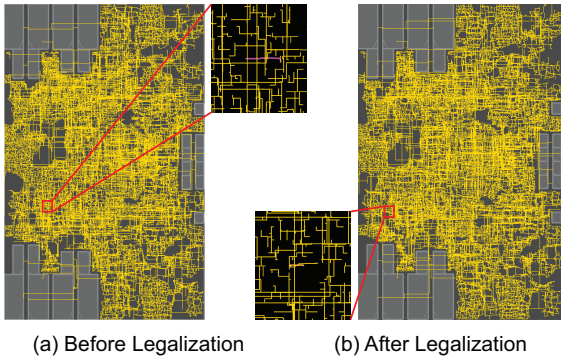

Figure 10: Clock Tree in the Cortex-A7 design. Routing patterns are similar.

are impractical in advanced technology nodes. The high timing criticality and dense connectivity between the dies in this partitioning style do not work well with the bump size and the routing solution proposed here. Therefore, Monolithic 3D (M3D) IC works best for such dense integration [6], and 3D via overlaps would not be an issue with the fine pitch of M3D vias [5].

6.5 Takeaways

Our legalizers can slightly degrade routing quality in designs with large via counts and utilization. Such cases require much finer pitch values that are not practical in the near future and necessitate a fully integrated routing solution to provide a good PPA quality with clean Design Rule Checks.

The two proposed legalization methods work to remove overlaps for more realistic partitioning and 3D bonding types. Our proposed bipartite matching solution is versatile and can be applied for various 3D pitch values, bonding styles, and partitioning types. Compared to more straightforward and traditional approaches like force-based legalization or greedy bump assignment, the bipartite-matching legalization has consistently outperformed in terms of maximum and average via displacements and the overall PPA due to the optimality in its algorithmic implementation. On the other hand, the heuristic-driven force-based legalizer is particularly handy when only a few vias or vias in a small region need to be legalized/displaced without affecting the overall via placement. In such cases, bipartite matching cannot be applicable due to its reliance on a via grid.

7 CONCLUSION

This paper shows that when the 3D via pitch becomes comparable to or larger than the global cell grid, the commercial detail router fails to create a good 3D via assignment without cut short and spacing violations. Fixing these violations early with our legalizer techniques produces better routing quality, fewer DRVs, and reduced total slack with negligible runtime impact, especially for hybrid-bonded 3D ICs. While our proposed method also helps monolithic 3D ICs with large via count and utilization, an improved router would significantly benefit such cases.

REFERENCES

[1] D. B. Ingerly et al. Foveros: 3D Integration and the use of Face-to-Face Chip Stacking for Logic Devices. In *2019 IEEE International Electron Devices Meeting (IEDM)*, 2019.
[2] AMD 3D V-Cache Ryzen Chiplets. https://www.anandtech.com/show/16725, 2018.
[3] Dragomir Milojevic et al. Fine-pitch 3D system integration and advanced CMOS nodes: technology and system design perspective. In *Design-Process-Technology*

Co-optimization XV. SPIE, 2021.

[4] Lennart Bamberg et al. Macro-3D: A Physical Design Methodology for Face-to-Face-Stacked Heterogeneous 3D ICs. *DATE*, 2020.

[5] Eric Beyne. The 3-D Interconnect Technology Landscape. *IEEE Design and Test*, 2016.

[6] Sai Pentapati et al. Pin-3D: A Physical Synthesis and Post-Layout Optimization Flow for Heterogeneous Monolithic 3D ICs. In *2020 IEEE/ACM International Conference On Computer Aided Design (ICCAD)*, 2020.

[7] Imed Jani et al. Characterization of Fine Pitch Hybrid Bonding Pads using Electrical Misalignment Test Vehicle. In *2019 IEEE 69th Electronic Components and Technology Conference (ECTC)*, 2019.

[8] Daniel W. Fisher, Sarah Knickerbocker, Daniel Smith, Robert Katz, John Garant, Jorge Lubguban, Vilmarie Soler, and Norman Robson. Face to face hybrid wafer bonding for fine pitch applications. In *2020 IEEE 70th Electronic Components and Technology Conference (ECTC)*, pages 595–600, 2020.

[9] B. W. Ku et al. Compact-2D: A Physical Design Methodology to Build Two-Tier Gate-level 3D ICs. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 2019.

[10] Hans Eisenmann. Force-Directed Placement of VLSI Circuits. In *Proceedings of the 2015 Symposium on International Symposium on Physical Design*, 2015.

[11] Jingwei Lu et al. EPlace-3D: Electrostatics Based Placement for 3D-ICs. In *Proceedings of the 2016 on International Symposium on Physical Design*, 2016.

[12] B. Goplen and S. Sapatnekar. Efficient thermal placement of standard cells in 3D ICs using a force directed approach. In *International Conference on Computer Aided Design*, 2003.

[13] Roy Jonker and A. Volgenant. A shortest augmenting path algorithm for dense and sparse linear assignment problems. *Computing*, 38:325–340, 2005.

[14] David Z. Pan, Bill Halpin, and Haoxing Ren. Timing-Driven Placement. In *Handbook of Algorithms for Physical Design Automation*, 2008.

[15] Jasper Snoek, H. Larochelle, and Ryan P. Adams. Practical Bayesian Optimization of Machine Learning Algorithms. In *NIPS*, 2012.

[16] Fernando Nogueira. Bayesian Optimization: Open source constrained global optimization tool for Python, 2014–.