Routing-aware Legal Hybrid Bonding Terminal Assignment for 3D Face-to-Face Stacked ICs

Siting Liu
The Chinese University of Hong Kong
Hong Kong SAR

Jiaxi Jiang
The Chinese University of Hong Kong
Hong Kong SAR

Zhuolun He
The Chinese University of Hong Kong
Hong Kong SAR

Ziyi Wang
The Chinese University of Hong Kong
Hong Kong SAR

Yibo Lin
Peking University, Beijing, China
Institute of EDA, Peking University
Wuxi, China

Bei Yu
The Chinese University of Hong Kong
Hong Kong SAR

Martin Wong
Hong Kong Baptist University
Hong Kong SAR

ABSTRACT

Face-to-face (F2F) stacked 3D IC is a promising alternative for scaling beyond Moore's Law. In F2F 3D ICs, dies are connected through bonding terminals whose positions can significantly impact routing performance. Further, there exists resource competition among all the 3D nets due to the constrained bonding terminal number. In advanced technology nodes, such 3D integration may also introduce legality challenges of bonding terminals, as the metal pitches can be much smaller than the sizes of bonding terminals. Previous works attempt to insert bonding terminals automatically using existing 2D commercial P&R tools and then consider inter-die connection legality, but they fail to take the legality and routing performance into account simultaneously. In this paper, we explore the formulation of the generalized assignment in the hybrid bonding terminal assignment problem. Our framework, **BTAssign**, offers a strict legality guarantee and an iterative solution. The experiments are conducted on 18 open-source designs with various 3D net densities and the most advanced bonding scale. The results reveal that BTAssign can achieve improvements in routed wirelength under all testing conditions from 1.0% to 5.0% with a tolerable runtime overhead.

CCS CONCEPTS

• **Hardware** → **Physical design (EDA)**; **3D integrated circuits**.

KEYWORDS

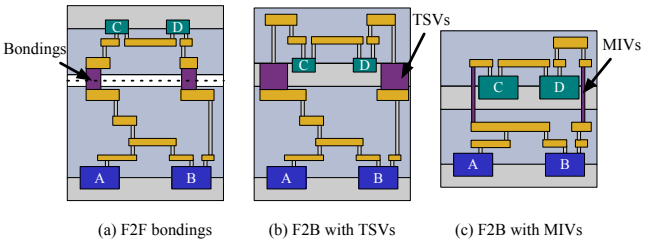
3D ICs; Bonding Terminal Planning; Routing; Generalized Assignment Problem; Divide and Conquer

ACM Reference Format:

Siting Liu, Jiaxi Jiang, Zhuolun He, Ziyi Wang, Yibo Lin, Bei Yu, and Martin Wong. 2024. Routing-aware Legal Hybrid Bonding Terminal Assignment for 3D Face-to-Face Stacked ICs . In *Proceedings of the 2024 International Symposium on Physical Design (ISPD '24), March 12–15, 2024, Taipei, Taiwan.* ACM, New York, NY, USA, 8 pages. https://doi.org/10.1145/3626184.3633322

ISPD '24, March 12–15, 2024, Taipei, Taiwan

ACM ISBN 979-8-4007-0417-8/24/03…$15.00
https://doi.org/10.1145/3626184.3633322



(a) F2F bondings (b) F2B with TSVs (c) F2B with MIVs

Figure 1: 3D stacking techniques for the inter-die connection. TSVs occupy the silicon area. MIVs only allow stacking with the same technology node even though their size is similar to metal vias and enable different track numbers. Bonding terminals can provide heterogeneous technology stacking without space crowding on the silicon layer.

1 INTRODUCTION

3D integration has provided a potential solution to the scaling issue of Moore's Law. Additionally, 3D stacking will lower fabrication costs because stacked dies can be designed and tested separately. Existing 3D integration techniques have mainly two alternatives shown in Figure 1: (1) Face-to-Face (F2F): two pre-fabricated dies are stacked by flipping the top die and stacking up on the bottom die, where the topmost metal layers are bonded together using F2F bondings. (2) Face-to-Back (F2B): the bottom die's topmost metal layer is connected to the top die's silicon layer, which is implemented by through-silicon s (TSVs) or monolithic vias (MIVs).

Compared with the F2B manner, F2F bonding has become a more promising way to continue scaling due to the superiority of F2F bonding terminals to both TSVs and MIVs. To be specific, TSV [1] takes up a significant amount of silicon area, which would make the placement area congested. Even though MIVs [2] are almost the same size as metal vias and Pentapati *et al.* [3] provides the chance for different track number settings, the tiers must be stacked sequentially, and do not permit heterogeneous technology nodes. Hybrid F2F bondings, on the other hand, are much smaller than TSVs

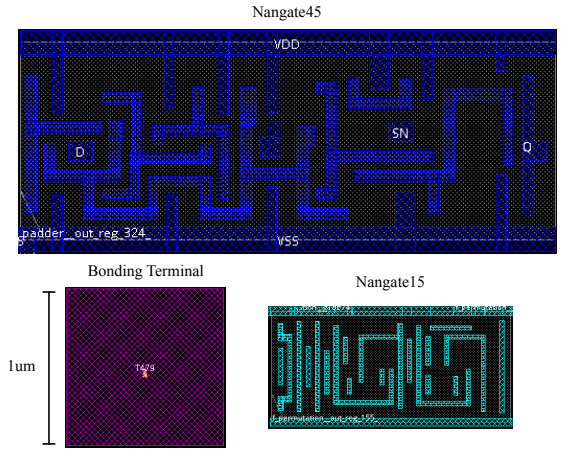


Figure 2: Illustration for the scale among gates in different technology nodes and the most advanced bonding terminal (purple) with 1μm pitch. the gate in blue is the gate in Nangate45 node and the green gate is the gate in Nangate15 node. With the advanced technology, bonding terminals are hard to automatically insert by the traditional routing process due to the non-negligible relative bonding terminal size to the gate. The inner rectangles are pin shapes.

and do not require any space on the silicon layer. Further, bondings enable heterogeneous stacking with different technology nodes. Both academia and industry have explored and demonstrated the superiority of F2F stacked 3D ICs. For instance, Premachandran *et al.* [4] achieved 30% package real estate reduction with F2F stacked heterogeneous 3D integration of MEMS and SoCs. Kim *et al.* [5] reported high memory bandwidth usage of 63.8 GB/s with approximated 4W peak power consumption by utilizing 3.4um-diameter F2F bondings. Intel Foveros [6] enables F2F stacking of heterogeneous dies using hybrid bondings. Furthermore, hybrid bonding is also used in AMD's Ryzen V-Cache 3D IC [7] to achieve a large L3 cache size and significantly improve system performance. All these successful applications highlight the remarkable performance and power benefits of F2F-bonded 3D ICs.

Meanwhile, 3D electronic design automation (EDA) tools developed from the conventional 2D flow have been made possible by the advancement of F2F bonding techniques. Shrunk2D [8] was first proposed to adapt the commercial 2D physical design tool to F2F-stacked 3D ICs. It initially shrinks the cells and routing geometries by 50% and then conducts traditional P&R using the commercial tool. The subsequent die partitioning is performed to determine the die location of the cells with original sizes. Following Shrunk2D, Compact2D [9] increased the floorplan footprint by a factor 2× to overcome the issue that the shrinking operation is not possible for ultimately scaled technology nodes. As a result, cells are linearly mapped back to the original die area after running the commercial 2D P&R flow on the enlarged floorplan. Similarly to Shrunk2D, die partitioning and F2F bonding automatic planning are then performed, followed by post-die-partitioning optimization and incremental routing. Furthermore, Macro-3D [10] was introduced to enable heterogeneous memory-on-logic or sensor-on-logic 3D integration. Moreover, Macro-3D does not need to extend the footprint or cell sizes substantially.

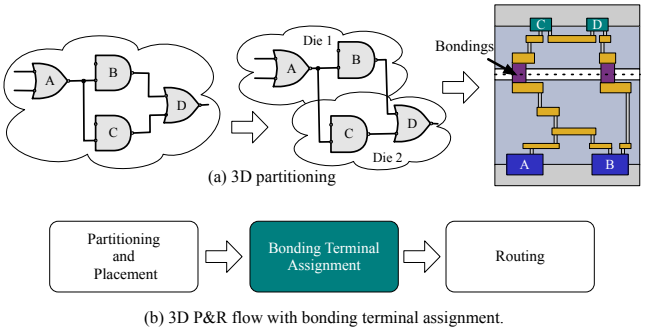


Figure 3: With the partitioned netlist, we stacked the separate dies with bonding terminals in this work. Our routing-aware bonding terminal assignment algorithm is integrated into the routing stage.

Though the above pseudo-3D methods have achieved favorable results, all these flows fail to honor the pitch rules for the bonding terminals [11]. Instead, they simply treat terminals as metal vias and then pin the hope on the 2D commercial flow and additional post-processing stage to insert them automatically. This arouses an overlapping issue due to the growing gap between the via pitch and the bonding terminal pitch requirement. In the case of Nangate15-Nangate45 F2F stacked ICs, the track pitches of the topmost metal layers are 0.28μm and 0.064μm for Nangate45 and Nangate15 respectively. Nevertheless, the pitch of the most advanced bonding technique is around 1μm [12], which is much larger than the advanced metal track pitch. As shown in Figure 2, even in the most aggressive terminal scale [12], the bonding terminal pitch is non-negligible in the modern technology nodes. In particular, the overlapping problem is getting worse with the dimensional scaling in advanced technology nodes, which increases the need for resolving this problem.

To guarantee the legality of modern F2F bonding terminal planning, Pentapati *et al.* [11] has proposed a bipartite-matching algorithm to legalize the bonding terminals with minimal total terminal displacement after inserting bonding terminals by the routing engine. However, considering legality in the post-processing stage will cause detours on the well-optimized routing solution. In addition, the legalization improvement is severely constrained by the initial assignment position.

To overcome the above-mentioned limitations, we explore the generalized assignment formulation on the legal hybrid bonding terminal assignment problem. Further, the proposed routing-aware framework, **BTAssign**, is integrated before the routing stage for more room for improvement as shown in Figure 3. In the BTAssign framework, the adaptive generalized assignment formulation is solved efficiently through an iterative divide-and-conquer algorithm. The codes of BTAssign is available online[1].

The major contributions of this paper are listed as follows,

- To the best of our knowledge, we are the first to explore the generalized assignment formulation for routing-aware legal hybrid bonding terminal assignment to consider the legality and routing performance simultaneously.

[1]https://github.com/Lusica1031/BTAssign

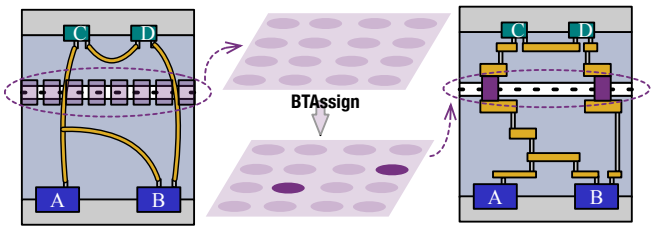


Figure 4: Bonding Terminal Assignment; This example gives three nets (Two 3D nets and one 2D net). The light purple circles represent the evenly distributed terminal candidates. Bonding terminal assignment is to determine terminals, colored with dark purple, for each 3D net.

- We provide an iterative hierarchical bipartite matching algorithm to solve the generalized assignment formulation within an acceptable runtime.
- Compared to current state-of-the-art bonding terminal assignment and legalization works, our proposed framework can obtain an average 2.79% improvement on routed wirelength. Further, The performance enhancement is up to 5.0%. Notably, All the testing cases could gain benefits from it.

2 PRELIMINARIES

2.1 3D Face-to-Face Stacking

To avoid the ever-increasing complexity of technology node scaling, advances in wafer bonding techniques have opened up new possibilities for 3D integration. Modern hybrid bonding terminals offer integration along the z-axis, which enables new architectures with improved performance and compressed die area. Different from traditional 2D architecture with all the pins in the lowest metal layer, 3D integration partitions the gates into different dies. In the F2F bonded design, the topmost metal layers of separate dies are connected by bonding terminals as shown in Figure 3. As a consequence, the inter-die F2F bonding connections would not occupy any placement resource on both dies, which always takes place on the bottom metal layer. On the other hand, the large bonding terminal pitch makes the bonding terminal resource between dies limited and critical, while in the common 2D designs, the routing resources at the lower metal layers are more scarce.

Unlike the large TSVs, F2F bonding terminals make sub-10μm pitch inter-die interconnection possible. To meet different demands, various bonding techniques [12–14] were proposed, among which the hybrid bonding technique has emerged as the promising solution. To enable the fabrication of hybrid bonding terminals, an interconnect surface must be created to link two wafer surfaces together. It enables direct bonding between the back-end-of-lines of pre-fabricated dies [15, 16] using an annealing process at a low temperature ($< 250^{\circ}$C) to strengthen the inter-die bonding. The resulting advanced F2F bonding terminal pitch allows designers to utilize fine-grained and silicon-space overhead-free 3D interconnections in F2F stacked 3D ICs. According to the fabrication process of hybrid bonding terminals described above, early assignment is necessary for hybrid bonding terminals. Since the wafer design symmetry must be maintained and proper connectivity should be ensured, bonding terminals preferably have neat distribution. Therefore, dissimilar to previous bonding terminal assignment works [10, 17], hybrid bonding terminal candidates with even distribution are applied in our proposed framework to enable a better modern fabrication process.

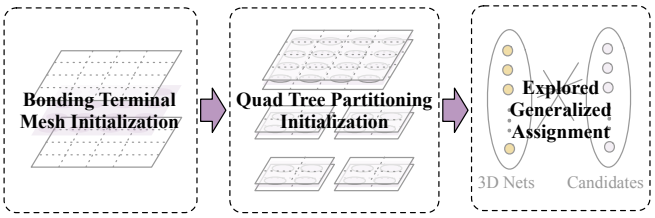


Figure 5: The overall flow of our proposed BTAssign framework can be divided into initialization, partitioning, and the generalized assignment for bonding terminals.

2.2 Problem Formulation

Definition 1 (3D Net). If net e has a set of pins $p = \{p_1, p_2, \cdots\}$ and there exists a pair of pins (p_i, p_j) which are placed in different dies, we call e as a 3D net.

Problem 1 (Routing-aware Bonding Terminal Assignment). Given a set of 3D nets $E = \{e_1, e_2, \cdots, e_n\}$ and bonding terminals $T = \{t_1, t_2, \cdots, t_m\}$, our task is to find a surjection solution $f : T \rightarrow E$ for better routing performance.

The bonding terminal assignment is illustrated in Figure 4.

3 ALGORITHM

3.1 Overall Flow

Before diving into the algorithm details, we first introduce the overall flow of our routing-aware bonding terminal assignment framework as illustrated in Figure 5. The BTAssign can be divided into three parts, the initialization stage, the partitioning stage, and the bonding terminal assignment. We first initialize the legal bonding terminal mesh grids as candidates and partition the mesh to construct a quadtree. Then we formulate the above-mentioned routing-aware bonding terminal assignment problem as a generalized assignment and propose an iterative hierarchical bipartite matching algorithm to solve this NP-complete formulation.

3.2 Explored Generalized Assignment

A set of 3D nets $E = \{e_1, e_2, \cdots, e_n\}$ is included in the 3D F2F stacking integration. Routing for both 3D nets and planar nets in the F2F-bonded designs is called after the partitioning and placement stage. The main difference between the routing process for 2D nets and for 3D nets is the criticality of bonding terminals. Unlike the traditional multi-metal layers global routing, the gates are divided into two different silicon dies in 3D F2F-bonded designs. In this case, the 3D F2F-bonded routing needs to achieve inter-die connections, which are fabricated through the bonding terminals. On the other hand, the bonding terminal pitch is much larger than the metal via pitch with the advanced technology node as described before, limiting the number of bonding terminals that can legally fit on an entire chip. Therefore, the terminal-net assignment becomes an important factor affecting the final routing solution quality.

The terminal assignment can be naturally formulated as a generalized assignment problem, which aims to seek the minimum cost

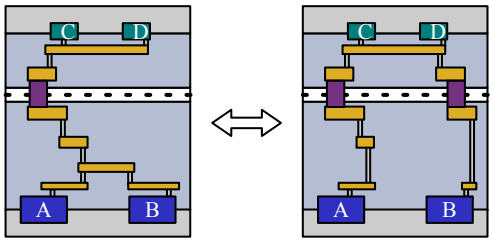


Figure 6: Single vs. Multiple bonding terminal assignment. The multiple assignment may benefit on the metal wirelength compared to the single assignment.

assignment of multiple tasks to multiple agents. Any agent can be assigned to perform any task, incurring costs that vary depending on the agent-task assignment. Besides, each agent has a capacity constraint and the sum of tasks assigned to it cannot exceed the capacity. Mathematically, the generalized assignment problem can be formulated as integer programming [18], which is NP-hard. There exist some linear programming relaxations to give an approximated solution. Especially, when all the agents' capacities C_i are equal to 1, the generalized assignment problem can be reduced to an assignment problem. Alternatively, it can be reduced to the weighted bipartite graph matching problem for minimum costs.

In terms of our problem, we first initial a set of candidate even-distributed bonding terminals $T = \{t_1, t_2, \cdots, t_m\}$. $t_j \sim e_i$ is introduced as an assignment from t_j to n_i. Further, we define $T_i = \{t_j \in T \mid t_j \sim e_i\}$ as the set of bonding terminals assigned to the net e_i. We use $c(T_i)$ to denote the pre-calculated routing cost of the assignment T_i, which is approximated by the minimum spanning tree length between the pins of net e_i and the bonding terminals T_i in our setting.

Each net e_i has a pair of constrain (C_i^l, C_i^u) to limit the minimal/maximal number of matching bonding terminals. The problem aims to find an assignment from bonding terminals to nets to minimize the sum of costs. We use a boolean variable x_{ij} to denote whether the relation $t_j \sim e_i$ is established. Then, the bonding terminal assignment problem can be formulated as in Formula (1),

$$\begin{aligned} \min \quad & \sum_{i=1}^{n} c(T_i), \\ \text{s.t.} \quad & C_i^l \le \sum_{j=1}^{m} x_{ij} \le C_i^u, \\ & \sum_{i=1}^{n} x_{ij} \le 1, x_{ij} \in \{0,1\}, \end{aligned} \tag{1}$$

where the second constraint says that each bonding terminal can be assigned to at most one net.

The first constraint in Formula (1) means that at least one bonding terminal should be assigned to implement the inter-die connection. What's more, there exist possibilities to assign multiple bonding terminals to some multi-pin 3D nets for better performance. As shown in Figure 6, in some cases, more bonding terminals would help to further reduce the total routed wirelength of 3D nets. What complicates our situation is that, in the routing problem, the combined effect of multiple bonding terminals to the final result is not a simple linear accumulation of every single terminal. In other words,

$$c(T_i) \neq \sum^{t_j \sim e_i} c(\{t_j\}). \tag{2}$$

Therefore, we need to enumerate all the bonding terminal combinations to solve the generalized assignment problem in Formula (1). Since a net e_i can have at most $\sum_{z=C_i^l}^{C_i^u} \binom{m}{z}$ possible assignment solutions, we have to precompute a table of $\sum_{i=1}^{n} (\sum_{z=C_i^l}^{C_i^u} \binom{m}{z})$ assignment costs in advance. To reduce problem complexity, we enforce $C_i^u \le 2$. However, m is a huge number for VLSI designs. Directly solving Formula (1) is still impractical in the modern design flow.

Another strategy to constrain the upper bound of terminal amounts is to assign them to nets incrementally. Imagine a bipartite graph, where the first set of vertices is the 3D nets, and the second set of vertices are the candidate bonding terminals. If we solve the bipartite matching problem on this graph, we are effectively assigning one bonding terminal to each 3D net if appropriate. Then, we can update the cost of the remaining terminals. By solving the bipartite matching problem in a fixed number of rounds, we can control the maximum terminal number assigned to each net. We use a superscript $^{(k)}$ to denote the variables after solving k rounds of bipartite matching problems. Specifically, $x_{ij}^{(k-1)}$ is the accumulated assignment solution of previous $k-1$ iterations and $y_{ij}^{(k)}$ denotes the assignment in the k-th iteration. Now, the optimization problem in the k-th iteration becomes in Formula (3).

$$\begin{aligned} \min \quad & \sum_{i=1}^{n} c(T_i^{(k-1)} \cup \{t_j \mid y_{ij}^{(k)} = 1\}), \\ \text{s.t.} \quad & C_i^l \le \sum_{j=1}^{m} (x_{ij}^{(k-1)} + y_{ij}^{(k)}) \le C_i^u, \\ & \sum_{i=1}^{n} (x_{ij}^{(k-1)} + y_{ij}^{(k)}) \le 1, \quad \sum_{j=1}^{m} y_{ij}^{(k)} \le 1, \\ & x_{ij}^{(k-1)} = \{0,1\}, \quad y_{ij}^{(k)} = \{0,1\}. \end{aligned} \tag{3}$$

Based on the second constraint, each net is only assigned at most one bonding terminal in each iteration. Therefore, we only need to calculate $m \times n$ costs at one iteration.

Algorithm 1 Bonding Terminal Assignment

Input: Placed nets E, bonding terminals T, budget K.
Output: Bonding terminal assignment.

```
function assign(nets E, terminals T, budget K)
    for all i ∈ 1, ··· , K do
        for all unassigned terminal t ∈ T do
            for all net e ∈ E do
                Calculate the cost of assigning t to e;
            end for
        end for
        Run bipartite matching between E and T;
        Update terminal assignment;
    end for
end function
```

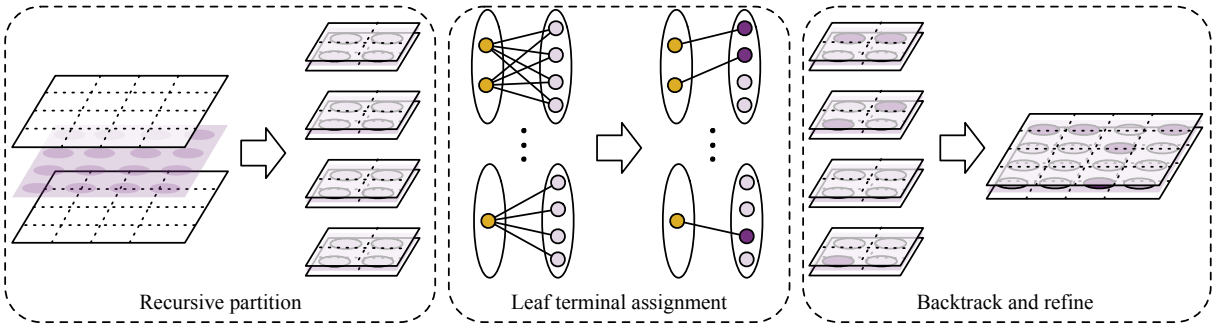


Figure 7: Illustration for the hierarchical bonding terminal assignment framework with the Quad-tree. After each level of terminal assignment, the assigned terminals are masked to the next level as shown in the backtrace and refine stage.

3.3 Hierarchical Bipartite Matching

The iterative optimization problem introduced above is still expensive to solve. In this sense, we propose a hierarchical solution to the problem in a three-step manner (see Figure 7): 1) Recursive partition; 2) Leaf terminal assignment; 3) Backtrack and refine.

Recursive partition. To reduce the problem complexity, our strategy is to recursively partition the routing region into uniform grids, until the covered grid area is smaller than $p \times p$ (p is set as 50μm in our experiments). The partition procedure is similar to constructing a quad-tree [19] of the space. On the top, we partition the whole region into 2×2 grids uniformly. For each sub-region, if the stopping criteria is not met, we recursively repeat the partition procedure. In this way, solving individual problems within the leave sub-regions becomes feasible.

During partition, it is important to carefully deal with the nets, as pins within the same net might scatter around the layout. There are various ways to deal with the objects in the space, as extensively studied in spatial data structures like R-tree [20], R*-tree [21]. In our approach, we assign each net to the quad-tree node that fully encloses the whole net. This ensures the mapping quality of bonding terminal assignment since the corresponding net pins are close enough to the considered bonding terminals.

Leaf terminal assignment. Given a leaf node of the partition quad-tree, bonding terminals are assigned to the nets within the leaf node. To further reduce problem complexity, only bonding terminals located within the region are considered during the assignment. This is intuitive for wirelength minimization. Basically, Formula (3) and Algorithm 1 are utilized to find an assignment that optimizes the estimated wirelength, where the chosen cost function c should be defined by some wirelength model. In our practice, we use the minimum spanning tree (MST) as the wirelength proxy. To calculate the cost of a set of bonding terminals assigned to a net, we assume a clique composed of the net pins as well as the bonding terminals and then run *Prim's algorithm* to obtain the MST of the clique.

Backtrack and refine. After the leaf terminal assignment is done, we have to backtrack along the tree and refine the terminal assignment for large nets. According to our algorithm, these nets do not belong to any leaf node. Instead, they are assigned to some internal tree nodes. Similar to the leaf terminal assignment, we look for bonding terminal assignment to these large nets within the region. The only difference is that the terminals already assigned are masked out. The backtrack and refine step is illustrated in lines 7-13, Algorithm 2.

Algorithm 2 Hierarchical Bonding Terminal Assignment

Input: Placed nets E, bonding terminals T, budget K, partitioning parameter p.
Output: Bonding terminal assignment

```
1:  partition(layout);
2:  for all Leaf subregions s do
3:      E_s ← nets located within s;
4:      T_s ← terminals located within s;
5:      assign(E_s, T_s, K);
6:  end for
7:  for all Nonleaf subregions s from bottom to top do
8:      Mask assigned terminals within s;
9:      Remove assigned nets within s;
10:     E_s ← unassigned nets located within s;
11:     T_s ← unassigned terminals located within s;
12:     assign(E_s, T_s, K);                    ▷ Algorithm 1
13: end for
14: function partition(region)
15:     if region > p × p then
16:         Partition region into 2 × 2 subregions uniformly;
17:         for all subregion do                ▷ Recursive partition
18:             partition(subregion);
19:         end for
20:     end if
21: end function
```

4 EXPERIMENT RESULTS

4.1 Experimental Setting

We evaluate the performance of our proposed framework on six open-source real-world designs by integrating our proposed bonding terminal assignment framework as shown in Figure 3. Three winners in [17] are applied to finish the 3D partitioning and placement stage. We pick cadb1024 since the second place, cadb1015, cannot produce placement solutions in a reasonable runtime. After assigning bonding terminals using the state-of-the-art F2F placer, FastRoute [22] in the latest OpenROAD project is used to generate the routing results, in which each routing tile includes 10 tracks. Note that all the routing results are overflow-clean. Our framework

Table 1: Statistics for six open-source real-world designs.

Design	Die Area (um^2)	#Instances	#Nets
ethmac	136.023 × 134.4	23771	25151
jpegencode	419.753 × 268.8	173469	206046
mor1kx	176.7 × 268.8	57197	38198
or1200	659.562 × 403.2	326845	339318
sha3	76.688 × 134.4	22797	22870
tinyaes	358.629 × 268.8	326136	326728

is developed with the OpenDB database. All the algorithms are implemented in C++. The version of the Gurobi solver used here is 10.0. All experiments ran on a Ubuntu Linux x86 64-bit machine with 40 Intel Xeon CPU cores at 2.40GHz, and 128GB RAM with 8 threads.

4.2 Sampling Scheme

Due to the memory limitation and the runtime requirement, besides quad-tree partitioning, a candidate sampling scheme is proposed to reduce memory usage during each matching. The sampling factor s is used to determine the sampling density of bonding terminal candidates. When there are n 3D nets in the current quad-tree node, we will generate even-distributed $n \times s$ bonding terminal candidates. Besides, the partitioning stop size criteria p decides the covered area size on the leaf nodes. In other words, p decides the levels of our quad-tree. In our experiments, p is set as 50 μm for both viaLegal [11] and ours. Furthermore, s is 8 for the first iteration and 4 for the second in our framework.

4.3 Design Generation

We acquire six open-source designs from OpenCores [23]. The design detailed information is listed in Table 1. Since synthesizing with different technology nodes brings differences in the netlist structure during the optimization process in Cadence Genus, all six designs are first synthesized with Nangate15 to set the netlist. Then, having the netlist information, we extract cell libraries from the Nangate15 and Nangate45 to meet the contest input format (the cell size and pin offset). The heterogeneous face-to-face stacking IC is set as Nangate45 in the bottom die and Nangate15 in the top die [24]. The contest winners' binaries are utilized to generate the partitioning and placement solution for the gates of our 3D designs. As shown in Table 3, these winners can provide various distributions of 3D nets for the same design. In other words, different bonding terminal densities would be allocated. As shown in Table 2, we stack 13 metal layers together with one BT layer for bonding terminals. The BT layer is blocked from planar routing to enable a direct heterogeneous 3D routing process using FastRoute [22]. Note that the pitch of the bonding terminal is set as 1μm here.

4.4 Routing Performance

The comparison is conducted to prove the efficacy of our proposed assignment formulation. To evaluate the routing performance improvement bring by our proposed iterative bonding terminal assignment framework, we compare with the given initialized bonding terminals generated by the 3D placers from the ICCAD Contest [17] and the bonding terminal legalization method proposed in [11]. In Table 4, 'wl' and '# vias' represent the routed wirelength, and

Table 2: Track pitch information of our F2F stacked ICs.

Top Die (Nangate15)		Bottom Die (Nangate45)	
Name	pitch (μm)	Name	pitch (μm)
M1	0.064	BT	Blocked
MINT1	0.064	metal6	0.28
MINT2	0.064	metal5	0.28
MINT3	0.064	metal4	0.28
MINT4	0.064	metal3	0.14
MINT5	0.064	metal2	0.19
MINT6	0.064	metal1	0.14

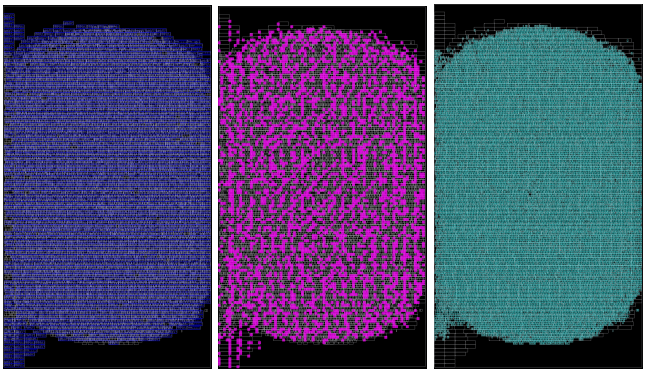
Table 3: Statistics for the number of 3D nets generated by three 3D placers from [17].

Design	#3D Nets		
	cadb1051	cadb1021	cadb1024
ethmac	11311	3724	6570
jpegencode	59487	31681	29379
mor1kx	31234	8445	13254
or1200	105849	45385	154082
sha3	8394	3227	3892
tinyaes	69772	19979	16385

the number of metal vias reported from FastRoute. 'rt' is short for runtime. Note that 'rt-route' only includes the routing runtime and 'rt-all' is the accumulated runtime of bonding terminal legalization/assignment and routing. As shown in Table 4, our proposed iterative bonding terminal assignment framework can obtain up to 5.0% and 2.24% improvement for routed wirelength and vias number respectively on average compared to the winner's assignment solutions. Our method can achieve 2.8% and 0.5% for the routed wirelength and via number on average. As listed in Table 4, the routed performance of all the cases is improved significantly under the same setting compared to the initial bonding terminals from [17] and the legalized on from [11]. It's worth noting that the results show the viaLegal method proposed in [11] is very dependent on the initialized bonding terminal positions, which are given by the winner F2F placers here. Differently, our proposed routing-aware formulation can obtain more flexibility and global information. Further, compared to setting the total via displacement as the legalization target, the routing-aware costs are better for the following routing stage. As for the overall runtime comparison, we only report the routing runtime for the given initialized bonding terminals since the winners' bonding terminal assignment algorithm is integrated into the whole placement stage and the separate runtime profiling cannot be obtained from the binaries provided by [17]. Compared to the original routing runtime, our iterative framework only needs an average 1.436× runtime overhead while the legalization method proposed in [11] takes 5.252× longer since they need to consider all the candidate positions for legalization within the area of a single quad-tree node. Therefore, the runtime overhead is acceptable to obtain up to 5% routed wirelength improvement. To further establish the solution of our framework, we illustrate one case 'sha3 with cadb1021' in Figure 8. The bottom die cells are colored as blue and the cells on the top die are in green. Our assigned bonding terminals are marked as purple. As shown in Figure 8, the bonding

Table 4: Experimental results on real-world open-source designs with different placement solutions from three placers. These placers allocate an initial bonding terminal positions. Compared to the previous bonding terminal legalization work, viaLegal [11], our proposed iterative hierarchical matching algorithm can obtain improvements on both routed wirelength and the number of vias with a tolerable runtime overhead. The best one is marked as bold.

Design		FastRoute [22]			viaLegal [11] + FastRoute [22]			BTAssign + FastRoute [22]		
		wl (μm)	# vias	rt-route (s)	wl (μm)	# vias	rt-all (s)	wl (μm)	# vias	rt-all (s)
cadb1024	ethmac	186.81	281883	**5.99**	189.10	283396	16.08	**178.39**	**279194**	11.41
	jpegencode	1437.11	3104739	**48.08**	1445.62	3109616	336.00	**1394.64**	**3085740**	119.66
	mor1kx	584.20	722950	**24.64**	586.80	726698	67.36	**566.02**	**718771**	44.29
	or1200	4559.60	3010540	**114.60**	4626.86	3062008	933.15	**4397.76**	**2943050**	373.29
	sha3	170.95	373479	**7.98**	171.73	373905	19.16	**162.39**	**368041**	19.42
	tinyaes	1934.92	6706891	**87.36**	1938.77	6710108	173.79	**1910.34**	**6689367**	125.39
cadb1021	ethmac	179.67	480896	12.70	179.88	481134	17.78	**174.11**	**479606**	10.11
	jpegencode	1319.08	4176215	**52.23**	1318.47	4176845	168.79	**1257.06**	**4157969**	95.29
	mor1kx	592.65	1174821	**18.62**	593.19	1175284	51.27	**583.70**	**1171429**	49.58
	or1200	4490.04	6979948	**126.29**	4490.04	6981306	655.61	**4443.75**	**6967324**	228.29
	sha3	161.81	469271	**6.16**	161.83	469205	10.72	**158.37**	**468771**	18.30
	tinyaes	1662.31	7629955	**82.13**	1661.34	7629853	204.48	**1614.66**	**7610934**	124.32
cadb1051	ethmac	172.87	241985	**5.19**	178.71	245379	20.64	**167.90**	**240768**	14.81
	jpegencode	1263.82	3080798	**49.07**	1283.94	3094259	297.98	**1207.32**	**3058096**	171.04
	mor1kx	605.03	**573297**	**24.66**	629.23	588623	143.99	**599.65**	576116	94.14
	or1200	3996.45	5163098	**116.06**	4026.81	5189696	1204.45	**3959.78**	**5154037**	372.53
	sha3	167.14	313924	**7.55**	173.77	317453	36.01	**162.20**	**311601**	69.91
	tinyaes	2093.40	6066197	**95.60**	2119.49	6091509	290.27	**2044.98**	**6041593**	214.06
Ratio		1.000	1.000	**1.000**	1.011	1.005	5.252	**0.972**	**0.995**	2.436



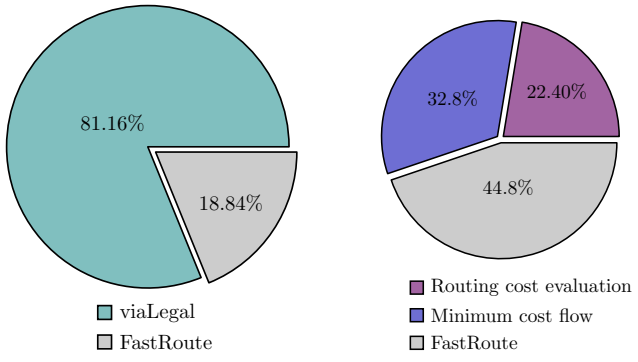
(a) Bottom die **(b) Bonding terminals** **(c) Top die**

Figure 8: Illustration for our bonding terminal assignment solution (sha3 with cadb1021).

terminals selected by our algorithm are strictly distributed on the manufacturing grids and have no overlap.

4.5 Runtime Breakdown

To further profile the runtime overhead, we illustrate the runtime breakdown in Figure 9. As described above, the setting for partitioning parameter p is the same for both viaLegal [11] and ours. Since the objective of viaLegal [11] is to minimize the total displacement of bonding terminals, the cost in viaLegal is simply evaluated by the



(a) viaLegal+FastRoute **(b) BTAssign+FastRoute**

Figure 9: Runtime breakdown. The runtime overhead of our proposed BTAssign is from both the routing cost evaluation step and minimum cost flow to solve the bipartite matching. ViaLegal takes 4× longer than FastRoute.

Manhattan distance between the given bonding terminal to the legal candidates. As shown in Figure 9(a), the legalization process without sampling spends more than 80% of time in the legalization process. As for our iterative assignment framework, As listed in Figure 9(b), the time spent on costs evaluation is about equal to the time spent on minimum cost flow for all the iterations. Furthermore, the overall runtime of our bonding terminal assignment framework is near 1.5× to FastRoute [22].

Table 5: Performance comparison between solving Formula (1) directly using Gurobi and our proposed iterative assignment algorithm under different settings for partitioning parameter p and sampling factor s.

Design	Gurobi Solver for (1)			Iterative Bipartite Matching		
	wl (μm)	# vias	rt-all (s)	wl (μm)	# vias	rt-all (s)
p: 50μm, s: 8 (Default setting as same to Table 4)						
sha3	MEMORY EXCEEDED			158.37	468771	18.30
ethmac	MEMORY EXCEEDED			174.11	482041	9.30
p: 25μm, s: 4						
sha3	159.44	468959	6414.25	159.46	468961	11.58
ethmac	MEMORY EXCEEDED			177.50	480564	9.47
p: 25μm, s: 2						
sha3	161.88	469355	2515.21	161.89	469351	8.89
ethmac	182.45	481975	655.93	182.61	482041	9.20

4.6 Generalized Assignment Formulation Evaluation

With the same partitioning parameter p and sampling factor s to the above experiments, the Gurobi solver for Formula (1) runs out of memory for all cases as shown in the first two lines in Table 5 due to the large number of candidate assignments. To prove the benefits of multiple-to-one bonding terminals assignment, we collect some feasible solutions from Formula (1) by using different parameter settings. Due to the memory limitation and massive memory usage to solve Formula (1) directly, we reduce the sampling rate s and shrink the partitioning parameter p. Then the area of quad-tree leaf node and the number of candidate terminals are reduced to fit the maximum memory quota. Gurobi [25] is applied as the integer programming solver for Formula (1). Although in the case of sampling, the experimental results cannot be proved as the optimal solution. But to a certain extent, we can still compare the effectiveness of our proposed iterative hierarchical bipartite matching algorithm under the same parameter settings.

We list three different configurations in Table 5. The first setting is the same to the default setting in Table 4. Gurobi cannot obtain any feasible solutions even for the smallest designs under this setting. By shrink both p and s to the half, the integer programming solver can attain a result for 'sha3 with cadb1021'. When we further reduce s to 2, two designs, 'sha3' and 'ethmac' with cadb1021 give the results. The experimental results in Table 5 show that the proposed iterative bipartite matching algorithm to solve Formula (3) can obtain comparable routing performance with notable runtime saving.

5 CONCLUSION

In this paper, we have proposed an explored generalized assignment formulation of the 3D IC bonding terminal assignment problem for the first time. Then, an iterative hierarchical bipartite matching framework, BTAssign, is introduced to efficiently solve the problem. Experimental results on real-world designs prove the efficacy of our proposed framework compared to the previous bonding terminal legalization work. Overall, the provided framework in this paper can consider the legality and acquire the global information in advance. Naturally, our proposed BTAssign could be extended to other bonding terminal scale without overlap issue since the bonding terminal mesh is generated first based on the scale. Lastly, this study has raised the importance of routing-aware bonding terminal assignment for improving the final routing performance.

ACKNOWLEDGEMENT

This work is partially supported by The Research Grants Council of Hong Kong SAR (No. CUHK14210723) and The Natural Science Foundation of Beijing, China (No. Z230002).

REFERENCES

[1] M. Motoyoshi, "Through-silicon via (TSV)," *Proceedings of the IEEE*, vol. 97, no. 1, pp. 43–48, 2009.
[2] P. Batude, B. Sklenard, C. Fenouillet-Beranger, B. Previtali, C. Tabone, O. Rozeau, O. Billoint, O. Turkyilmaz, H. Sarhan, S. Thuries *et al.*, "3D sequential integration opportunities and technology optimization," in *Proc. IITC*, 2014, pp. 373–376.
[3] S. S. K. Pentapati and S. K. Lim, "Heterogeneous monolithic 3D ICs: EDA solutions, and power, performance, cost tradeoffs," in *Proc. DAC*, 2021, pp. 925–930.
[4] C. Premachandran, J. Lau, L. Xie, A. Khairyanto, K. Chen, M. E. P. Pa, M. Chew, and W. K. Choi, "A novel, wafer-level stacking method for low-chip yield and non-uniform, chip-size wafers for MEMS and 3D SIP applications," in *Proc. ECTC*, 2008, pp. 314–318.
[5] D. H. Kim, K. Athikulwongse, M. B. Healy, M. M. Hossain, M. Jung, I. Khorosh, G. Kumar, Y.-J. Lee, D. L. Lewis, T.-W. Lin *et al.*, "Design and analysis of 3D-MAPS (3d massively parallel processor with stacked memory)," *IEEE TC*, vol. 64, no. 1, pp. 112–125, 2013.
[6] D. Ingerly, S. Amin, L. Aryasomayajula, A. Balankutty, D. Borst, A. Chandra, K. Cheemalapati, C. Cook, R. Criss, K. Enamul *et al.*, "Foveros: 3D integration and the use of face-to-face chip stacking for logic devices," in *Proc. IEDM*, 2019, pp. 19–6.
[7] "AMD 3D V-Cache Ryzen Chiplets," .https://www.anandtech.com/show/16725.
[8] S. Panth, K. Samadi, Y. Du, and S. K. Lim, "Shrunk-2D: A physical design methodology to build commercial-quality monolithic 3D ICs," *IEEE TCAD*, vol. 36, no. 10, pp. 1716–1724, 2017.
[9] B. W. Ku, K. Chang, and S. K. Lim, "Compact-2D: A physical design methodology to build commercial-quality face-to-face-bonded 3D ICs," in *Proc. ISPD*, 2018, pp. 90–97.
[10] L. Bamberg, A. García-Ortiz, L. Zhu, S. Pentapati, S. K. Lim *et al.*, "Macro-3D: A physical design methodology for face-to-face-stacked heterogeneous 3D ICs," in *Proc. DATE*, 2020, pp. 37–42.
[11] S. Pentapati, A. Agnesina, M. Brunion, Y.-H. Huang, and S. K. Lim, "On legalization of die bonding bumps and pads for 3D ICs," in *Proc. ISPD*, 2023, pp. 62–70.
[12] C. Netzband, K. Ryan, Y. Mimura, S. Ilseok, H. Aizawa, N. Ip, X. Chen, H. Fukushima, and S. Tan, "0.5 μm pitch next generation hybrid bonding with high alignment accuracy for 3D integration," in *Proc. ECTC*, 2023, pp. 1100–1104.
[13] D. W. Fisher, S. Knickerbocker, D. Smith, R. Katz, J. Garant, J. Lubguban, V. Soler, and N. Robson, "Face to face hybrid wafer bonding for fine pitch applications," in *Proc. ECTC*, 2020, pp. 595–600.
[14] P. Ramm, J. J.-Q. Lu, and M. M. Taklo, *Handbook of wafer bonding*, 2011.
[15] P. Morrow, C.-M. Park, S. Ramanathan, M. Kobrinsky, and M. Harmes, "Three-dimensional wafer stacking via Cu-Cu bonding integrated with 65-nm strained-Si/low-k CMOS technology," *IEEE electron device letters*, vol. 27, no. 5, pp. 335–337, 2006.
[16] T. Suga, R. He, G. Vakanas, and A. La Manna, "Direct Cu to Cu bonding and alternative bonding techniques in 3D packaging," *3D Microelectronic Packaging: From Architectures to Applications*, pp. 201–231, 2021.
[17] "ICCAD Contest 2022," .http://iccad-contest.org/2022/Problems.html.
[18] D. G. Cattrysse and L. N. Van Wassenhove, "A survey of algorithms for the generalized assignment problem," *European Journal of Operational Research*, vol. 60, no. 3, pp. 260–272, 1992.
[19] H. Samet, "The quadtree and related hierarchical data structures," *ACM Computing Surveys*, vol. 16, no. 2, pp. 187–260, 1984.
[20] I. Kamel and C. Faloutsos, "Hilbert R-tree: An improved r-tree using fractals," Tech. Rep., 1993.
[21] N. Beckmann, H.-P. Kriegel, R. Schneider, and B. Seeger, "The R*-tree: An efficient and robust access method for points and rectangles," in *Proc. SIGMOD*, 1990, pp. 322–331.
[22] Y. Xu, Y. Zhang, and C. Chu, "FastRoute 4.0: Global router with efficient via minimization," in *Proc. ASPDAC*, 2009, pp. 576–581.
[23] "OpenCores," https://opencores.org.
[24] "Nangate Technology Node," .https://si2.org/open-cell-library/.
[25] "Gurobi," .https://www.gurobi.com.