An Analytical RDL Chiplet-Terminal Co-Placement Methodology for Large-Scale 2.5D IC

ABSTRACT

As the number of chiplets in 2.5D IC continues to increase, existing chiplet placement method faces two main challenges: (1) the combinatorial explosion in the search space, and (2) the difficulty of achieving global optimization through iterativing chiplet and terminal placement. These challenges make it difficult to apply these methods for large-scale 2.5D ICs. To tackle these challenges, we develop an efficient analytical RDL chiplet-terminal co-placement methodology, ensuring synchronous placement of chiplets and terminals. Our methodology employs RDL chiplet-terminal co-placement in three stages: analytical global placement, legalization, and bump-terminal assignment, to achieve legal placement results that comply with design rules constraints. Experimental results demonstrate that our chiplet-terminal co-placement methodology reduces average wirelength by 25.78% compared to prior work for the common testcases, with a maximum speedup of up to 4800× in testcases with more than 10 chiplets. Furthermore, our methodology completes the placement with 64 chiplets within 30 minutes.

KEYWORDS

Chiplet-Terminal Co-Placement, 2.5D IC

ACM Reference Format:

. 2018. An Analytical RDL Chiplet-Terminal Co-Placement Methodology for Large-Scale 2.5D IC. In *Proceedings of Make sure to enter the correct conference title from your rights confirmation emai (Conference acronym 'XX).* ACM, New York, NY, USA, 8 pages. https://doi.org/XXXXXXX.XXXXXXX

1 INTRODUCTION

Traditional approaches to improving chip performance face significant challenges in the post-Moore era. Technologies such as Wafer-Level Chip-Scale Packaging (WLCSP) [18] and Chip on Wafer on Substrate (CoWoS) [8] have been proposed to manufacture 2.5D ICs, which is an important approach for improving chip performance by combining multiple chiplets within a single package. Both of them employ Redistribution Layers (RDLs) for interconnecting chiplets, facilitating high-speed and high-density interconnects. The sectional diagram of a 2.5D IC is shown in Figure 1. The chiplets and terminals are positioned respectively above and below the RDL layer. The chiplet is connected to the RDL via pins(μ-bumps) for interconnection through the RDL, and signals are routed out through terminals(C4-bumps) on the bottom of the RDL.

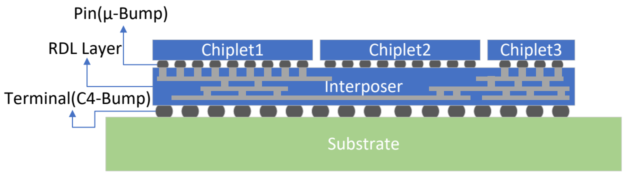
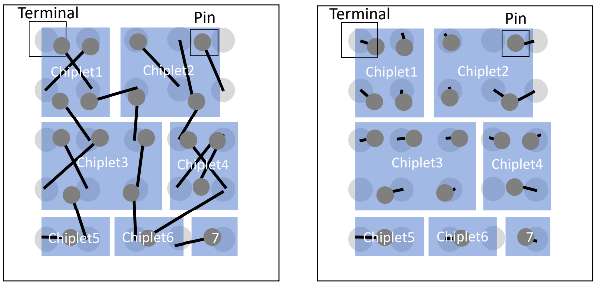


Figure 1: The sectional diagram of a 2.5D IC.



(a) (b)

Figure 2: Comparison of chiplet-terminal co-placement: (a) without co-placement; (b) with co-placement

In the design process of a 2.5D IC, the placement of instances on RDL significantly influences the routability of and the total wirelength after routing. If the placement results are suboptimal, it can lead to increased wirelength and affect system performance. RDL placement is a crucial design step where chiplets are positioned on the RDL layer while conforming to design rules and minimizing wirelength.

Two primary chiplet placement methodologies have been proposed to minimize chiplet interconnect wirelength: a) simulated annealing-based methods, and b) branch-and-bound (B&B) approaches. Ho et al. proposed a three-level HB* tree to represent the placement of chiplets macros, and I/O buffers on RDL and employed a simulated annealing approach to place them [6].[4, 14] took thermal considerations into account during chiplet placement, employing a simulated annealing method to minimize both wirelength and temperature. Osmolovskyi et al. modeled the chiplet placement problem as a constrained satisfaction problem (CSP) and employed a pruned parallel B&B approach for chiplet placement [16]. Chiou et al. proposed a parallel B&B method employing SP-CP sequences to achieve optimal chiplet placement and proposed a post-CP method to reduce operating temperatures with a slight increase in wirelength [3]. However, previous methods face the challenge of combinatorial explosion in the search space, growing exponentially by $O(n!)$ that n represents the number of chiplets, resulting in low efficiency when dealing with large-scale 2.5D IC placement. Despite

Conference acronym 'XX, June 03–05, 2018, Woodstock, NY

ACM ISBN 978-1-4503-XXXX-X/18/06
https://doi.org/XXXXXXX.XXXXXXX

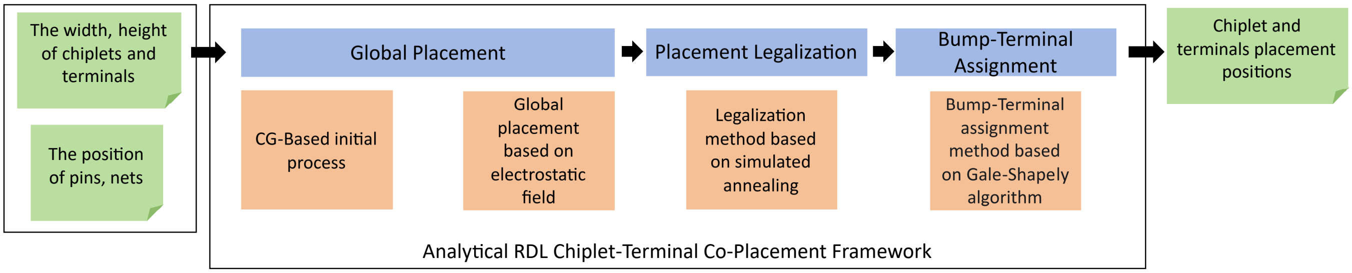

The width, height of chiplets and terminals
The position of pins, nets
Global Placement
CG-Based initial process
Global placement based on electrostatic field
Placement Legalization
Legalization method based on simulated annealing
Bump-Terminal Assignment
Bump-Terminal assignment method based on Gale-Shapely algorithm
Analytical RDL Chiplet-Terminal Co-Placement Framework
Chiplet and terminals placement positions

Figure 3: Overview of proposed RDL placement methodology

employing pruning techniques to speed up the search, their maximum runtime still exceeds 6 hours when the number of chiplets reaches 10 or more. Furthermore, they cannot simultaneously place chiplets and terminals due to efficiency constraints, limiting their ability to optimize results through chiplet-terminal co-placement, as shown in Figure 2.

Analytical techniques based on gradients provide fast convergence, excellent scalability, and are well-suited for optimizing large solution spaces. We apply these analytical methods to the RDL placement problem, enhancing the efficiency of RDL placement for large-scale 2.5D IC and improving the placement quality through chiplet-terminal co-placement, which included analytical global placement, legalization, and bump-terminal assignment three stages. Our methodology models placement density based on eDensity [11] and combines the weighted average wirelength model (WA) [7] in global placement. After global placement, we employ a simulated annealing-based legalization algorithm and a Gale-Shapely-based bump-terminal assignment algorithm to ensure the placement results meet design rules constraints. Our main contributions are as follows,

- We develop an RDL placement methodology to do RDL chiplet-terminal co-placement simultaneously, which includes the analytical global placement stage, legalization stage, and bump-terminal assignment stage to accomplish 2.5D IC RDL placement efficiently.
- We develop an efficient analytical global placement method to place chiplets and terminals simultaneously. We utilize a weighted average (WA) wirelength model combined with eDensity-based [12] density model for placement.
- Our methodology achieves an average wirelength reduction of 25.78% compared to prior work for the common testcases by employing chiplet-terminal co-placement. Additionally, the maximum speedup can reach up to 4800× faster cases with more than 10 chiplets. Furthermore, our method completes the placement for design with 64 chiplets within 30 minutes.

2 PROBLEM FORMULATION

RDL (Redistribution Layer) placement is the process of determining the positions of chiplets and terminals on the RDL. Chiplets are placed above the RDL, containing the core logic circuits of the chips, while terminals are placed below the RDL and connected to the outside world through C4-bumps bonding beneath the RDL. The pins of the chiplet are connected to the RDL via μ-bumps, the RDL constructs wire nets by connecting terminals to pins to achieve interconnects in 2.5D IC systems.

An example of RDL placement is illustrated in Figure 2b. The placement can be represented as a hypergraph $G = (V_c, V_t, E, R)$, where V_c and V_t represent the sets of vertices (chiplets) and vertices (terminals) respectively, and $V = V_c \cup V_t$, E represents the set of hyperedges (nets), and R represents the placement region, which includes chiplet placement regions R_c and terminal placement regions R_t. We use V_m and V_f to denote the sets of movable instances and fixed instances in V. $n = |V_m|$ represents the number of movable placement instances. The objective of the placer is to minimize the total half-perimeter wirelength ($HPWL(\mathbf{v})$) estimate of all the nets.

$$HPWL(\mathbf{v}) = HPWL(\mathbf{v_c}) + HPWL(\mathbf{v_t}) \tag{1}$$

Here $HPWL(\mathbf{v_c})$ and $HPWL(\mathbf{v_t})$ represent the total HPWL lengths calculated separately for the R_c and the R_t.

$$\min_{\mathbf{v}} HPWL(\mathbf{v}) \quad \mathbf{v} \text{ is legal.} \tag{2}$$

Equation 2 defines the RDL placement problem, where $\mathbf{v} = (\mathbf{x}, \mathbf{y})$ represents a placement result. $\mathbf{v}$ is legal, denoting that it satisfies the following requirements.

(1) Chiplets do not overlap with each other or with terminals.
(2) Terminals should be placed according to C4-bump design rules to ensure manufacturability.

3 OVERVIEW OF METHODOLOGY

Figure 3 shows the framework of the proposed RDL placement methodology. First, we input the physical information of chiplets and terminals to the framework, including the width and height of chiplets, the positions of pins, the list of terminals, the width, and height of the placement region of the RDL, and the net information between chiplets' pins and terminals.

In the initial placement stage, the bound-2-bound (B2B) net model [17] is employed to establish the relationship between wirelength estimation and the positions of pins and terminals. The optimization of the HPWL is transformed into a quadratic optimization problem [9] by employing B2B net model and solved by conjugate gradient method to obtain the initial placement results.

Subsequently, we utilize a weighted average wirelength model for smoothing the half-perimeter wirelength estimation [7]. We represent placement density employing an electrostatic field-based density model eDensity [11]. By combining wirelength and density

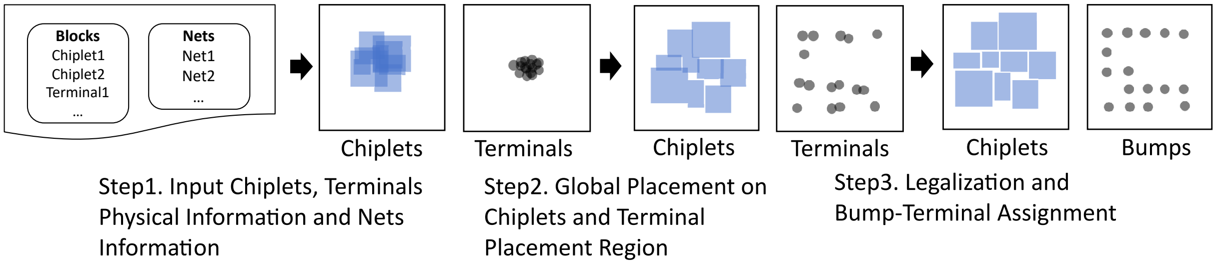

Blocks
Chiplet1
Chiplet2
Terminal1
...
Nets
Net1
Net2
...
Chiplets
Terminals
Chiplets
Terminals
Chiplets
Bumps
Step1. Input Chiplets, Terminals Physical Information and Nets Information
Step2. Global Placement on Chiplets and Terminal Placement Region
Step3. Legalization and Bump-Terminal Assignment

Figure 4: RDL chiplet-terminal co-placement flowchart.

objectives, we construct a placement objective function. Then We employ Nesterov's method to obtain the global placement results.

Next, we process to the placement legalization stage. In this stage, a simulated annealing-based algorithm is utilized to legalize the placement of chiplet and terminal regions simultaneously, ensuring no overlap between chiplets and terminals. Finally, we employ a bump-terminal assignment algorithm based on the Gale-Shapely algorithm to match terminals with bump according to design rules to get the final placement result. The schematic diagram of the various stages of RDL placement is depicted in Figure 4.

4 ANALYTICAL RDL GLOBAL PLACEMENT

4.1 Global Placement Problem Formulation

The placement region is uniformly divided into several rectangular grids (bins) during the global placement phase, denoted as B in the global placement stage. For the RDL placement problem, we uniformly divide the top R_c and bottom R_t placement regions of the RDL into bins, denoted as B_c and B_t respectively. Then, we express B as $B = B_c \cup B_t$. Based on a placement solution $\mathbf{v}$, we calculate $\rho_b(\mathbf{v})$ and $\rho_b(\mathbf{t})$, which is placement density from the Equation 3 of [12] and density target to be achieved in the global placement stage for each bin $b \in B$. As Equation 2 shows, a global placement problem targets a solution $\mathbf{v}$ with minimum total HPWL. To facilitate solving the placement problem using gradient-based analytical methods, we employ a weighted average (WA) wirelength model to smooth the HPWL [7] to make it differentiable and replace HPWL with WA forming $W(\mathbf{v})$ as the optimization objective. The global placement problem can be defined as Equation 3.

$$\min_{\mathbf{v}} W(\mathbf{v}) \text{ s.t. } \rho_b(\mathbf{v}) \leq \rho_t, \forall b \in B. \tag{3}$$

The objective of the global placement problem is to find a solution $\mathbf{v}$ that minimizes the total WA wirelength while satisfying density constraints for all bins, including the chiplet and terminal placement regions. We transform the density constraints into electrical energy term $N(\mathbf{v})$ by modeling the placement density $\rho_b(\mathbf{v})$ in Equation 3 as eDensity [11], where the movable instance i in RDL (including chiplets, terminals and pins) is regarded as a positively charged body, with its charge quantity corresponding to its area A_i. Then, we analogize the density penalty as electrical energy and the gradient of the density penalty as an electric field [11]. To prevent excessive uniform distribution of movable instances in the placement area leading to increased wirelength, we add fillers to the placement region to concentrate movable instances and reduce wirelength [12]. Finally, We transform the global placement problem into an unconstrained optimization problem for simplicity, as shown in Equation 4 [12].

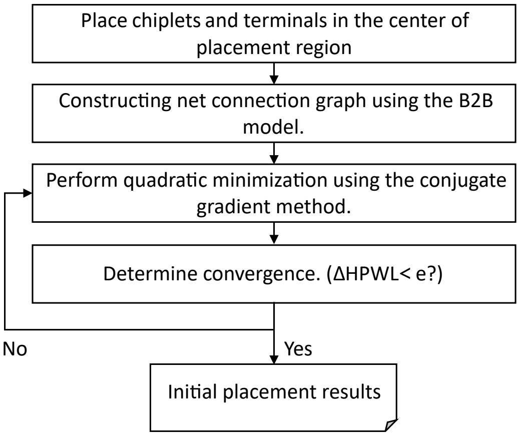

Place chiplets and terminals in the center of placement region
Constructing net connection graph using the B2B model.
Perform quadratic minimization using the conjugate gradient method.
Determine convergence. (ΔHPWL< e?)
No
Yes
Initial placement results

Figure 5: Initial placement flowchart.

$$\min_{\mathbf{v}} f = W(\mathbf{v}) + \lambda N(\mathbf{v}) \tag{4}$$

Here $W(\mathbf{v})$ represents the WA wirelength, which is the sum of WA $W(\mathbf{v_c})$ for the chiplet placement region R_c and WA $W(\mathbf{v_t})$ for the terminal placement region R_t, as shown in Equation 5. f is the objective function for optimization, and λ is the penalty factor.

$$\begin{aligned} W(\mathbf{v}) &= W(\mathbf{v_c}) + W(\mathbf{v_t}) \\ N(\mathbf{v}) &= N(\mathbf{v}_c) + N(\mathbf{v}_t) + \mu N_p(\mathbf{v}) \end{aligned} \tag{5}$$

Here $N(\mathbf{v}_c)$ and $N(\mathbf{v}_t)$ represent the density model established on R_c and R_t respectively. We model the pins of chiplet as charged bodies forming $N_p(\mathbf{v})$ as part of the density penalty to ensure a uniform distribution of pins after placement to prevent congestion during routing and achieve faster convergence, where μ is the penalty factor. Since both $W(\mathbf{v})$ and $N(\mathbf{v})$ in the optimization objective are differentiable, gradient-based optimization techniques can be used for optimization [12].

4.2 Initial Placement

Figure 5 illustrates the placement initial algorithm flow. In the initial placement stage, we first place the chiplets and terminals at the center of their placement region, then employ a B2B net model [17]

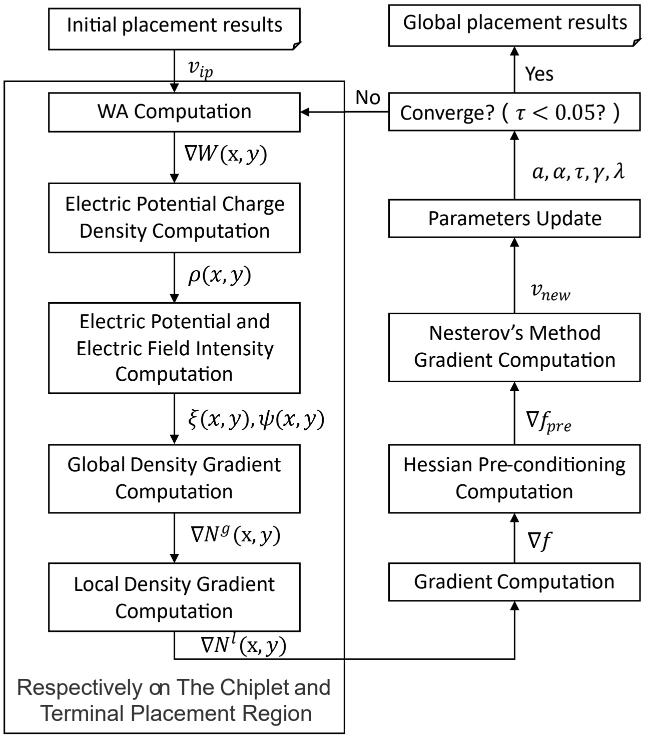

Initial placement results
v_{ip}
WA Computation
∇W(x, y)
Electric Potential Charge Density Computation
$\rho(x, y)$
Electric Potential and Electric Field Intensity Computation
$\xi(x, y), \psi(x, y)$
Global Density Gradient Computation
∇N^g(x, y)
Local Density Gradient Computation
∇N^l(x, y)
Respectively on The Chiplet and Terminal Placement Region
Global placement results
Yes
No
Converge? ($\tau < 0.05$?)
$a, \alpha, \tau, \gamma, \lambda$
Parameters Update
v_{new}
Nesterov's Method Gradient Computation
∇f_{pre}
Hessian Pre-conditioning Computation
∇f
Gradient Computation

Figure 6: Global placement flowchart.

to establish the relationship between HPWL and the position of pins and terminals as a net connection graph. By transforming it into a quadratic optimization problem, We optimize the HPWL iteratively by using the conjugate gradient method until convergence [9] to obtain the initial placement results.

4.3 Global Placement

We modify the algorithm from [2] for RDL global placement. The objective of the global placement algorithm is to compute Equation 4. Cheng et al. proposed a local density penalty in [2] that provides more repulsion for overflowed bins, helping us restrain the increase in suboptimal wirelength caused by the global density penalty factor λ. The optimization objective after adding the local density function is shown in Equation 6.

$$\min_{\mathbf{v}} f = W(\mathbf{v}) + \lambda N^g(\mathbf{v}) + N^l(\mathbf{v}) \tag{6}$$

Here $N^g(\mathbf{v})$ represents the global density, and $N^l(\mathbf{v})$ represents the local density. We partition the placement region into $m_c \times m_c$ bins and $m_t \times m_t$ bins for R_c and R_t respectively. Increase in m leads to increase in the total number of bins, improving accuracy but sacrificing some computational efficiency. As the FFT package from [5] requires the grid dimension m to be a power of 2, we set $m = \lceil \log_2 \sqrt{m'_{\text{set}}} \rceil$, $\lceil \cdot \rceil$ represents the ceiling function and m_{set} is a preset value due to efficiency concerns [12].

Global placement algorithm's detailed flow is shown in Algorithm 1. After dividing the chiplet and terminal placement region into $m_c \times m_c$ and $m_t \times m_t$ bins (line 1), we then initialize λ_0 by Equation 10 in [1] and initialize $\alpha_0 = 0.044 w_b$ where w_b is the bin width (line 2-3).

Algorithm 1 Global Placement Algorithm

```
Input:  initial placement solution $\mathbf{v}_0 = \mathbf{v}_{ip}$
        minimum overflow $\tau_{min}$
        maximum iterations $k_{max} = 500$
        pin density weight $\mu$
Output: global placement solution $\mathbf{v}_{gp}$
 1: $m_c \times m_c$ decomposition over $R_c$, $m_t \times m_t$ decomposition over $R_t$
 2: initialize $\lambda_0$
 3: initialize $\alpha_0^{max} = 0.044wb$
 4: initialize $\alpha^{local} = 1e-12, \beta^{local} = 1e-11, a_0 = 1, \mathbf{u}_0 = \mathbf{v}_0 = \mathbf{v}_{ip}$
 5: for $k = 1$ to $k_{max}$ do
 6:     $f_k = f(\mathbf{v}_k) = W(\mathbf{v}_k) + \lambda_k N^g(\mathbf{v}_k) + N^l(\mathbf{v}_k)$
 7:     for $\{R_c, R_t\}$ do
 8:         compute wirelength gradient $\nabla W$ and density $\rho_k$
 9:         $(\psi_k, \xi_k) = FFTsolver(\phi_k)$
10:         compute energy (global density) gradient $\nabla N_k^g = \mathbf{q}\xi_k$
11:         compute local density gradient $\nabla N_k^l$[2]
12:     end for
13:     compute Sum Gradient $\nabla f = \nabla W + \lambda \nabla N_k^g + \nabla N_k^l$
14:     compute Hessian Pre-conditioning $\nabla f_{pre}$[13]
15:     $\mathbf{u}_{k+1}, \mathbf{v}_{k+1}, a_{k+1} =$
16:     NM[13]$(a_k, \mathbf{u}_k, \mathbf{v}_k, \mathbf{v}_{k-1}, \nabla f_{pre}(\mathbf{v}_k), \nabla f_{pre}(\mathbf{v}_{k-1}))$
17:     update $\lambda_{k+1}$ by Eq. 7
18:     if $\tau_{k+1} \le \tau_{min}$ then
19:         $\mathbf{v}_{gp} = \mathbf{v}_{k+1}$
20:         break
21:     end if
22: end for
23: return $\mathbf{v}_{gp}$
```

We update λ in each iteration according to Equation 7 [13]. We set $\Delta HPWL_{REF}$ in Equation 7 to be 4.46×10^4.

$$\begin{aligned} \lambda_k &= \mu_k \lambda_{k-1} \\ \mu_k &= 1.1^{-(\Delta HPWL_k / \Delta HPWL_{REF})+1.0} \\ \Delta HPWL_k &= HPWL(\mathbf{v}_k) - HPWL(\mathbf{v}_{k-1}) \end{aligned} \tag{7}$$

Then we initialize α^{local}, β^{local}, and a_0 (line 4). The algorithm then enters the optimization iteration. we need to calculate the WA gradients in the x and y directions respectively for the R_c and R_t during iteration. Additionally, we compute the charge density for each chiplet and terminal within their respective placement regions (line 8). Using the spectral method, we solve for the electric potential respectively for the R_c and R_t (line 9). Then we obtain the gradient values for density penalty by calculating the gradient of the electric energy (line 10) and calculate the gradient of the local density penalty ∇N_k^l according to [2] (line 11). Next, we sum up these gradients and compute the hessian pre-conditioning ∇f_{pre}[13] to accelerate the optimization speed (lines 13-14). Following, we employ Nesterov's Method from [13] for gradient optimization until the final overflow reaches $\tau \le \tau_{min}$ (lines 16-21). We set τ_{min} to 0.05, and its calculation formula is shown in Equation 8.

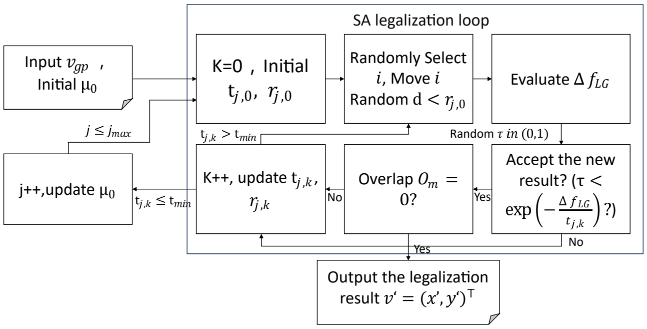

SA legalization loop
Input v_{gp} , Initial μ_0
K=0 , Initial $t_{j,0}$, $r_{j,0}$
Randomly Select i, Move i Random d < $r_{j,0}$
Evaluate Δf_{LG}
Random τ in (0,1)
Accept the new result? ($\tau < \exp\left(-\frac{\Delta f_{LG}}{t_{j,k}}\right)$?)
Yes
Overlap O_m = 0?
No
K++, update $t_{j,k}$, $r_{j,k}$
$t_{j,k} > t_{min}$
$t_{j,k} \le t_{min}$
j++,update μ_0
$j \le j_{max}$
No
Yes
Output the legalization result $v' = (x', y')^{\top}$

Figure 7: Legalization flowchart.

$$\tau = \frac{\sum_{b \in B} \max(\rho'_b - \rho_t, 0) A_b}{\sum_{i \in V_m} A_i}, \tag{8}$$

Here, A_b denotes the area of bin b, A_i denotes the area of movable instance i, ρ'_b denotes the density of bin b solely caused by movable instances, and ρ'_t is the target density [12].

5 LEGALIZATION AND BUMP-TERMINAL ASSIGNMENT

5.1 Legalization

Once the global placement results are obtained, there may still be slight overlaps between movable instances. RDL placement legalization aims to fine-tune the global placement result, eliminating overlaps between movable instances as much as possible without significantly increasing the wirelength. We modify the simulated annealing-based legalization algorithm from [13] to legalize the RDL placement, the objective function for legalization is shown as Equation 9.

$$f_{LG}(\mathbf{v}) = HPWL(\mathbf{v}) + \mu_o O_m(\mathbf{v}) \tag{9}$$

Here $HPWL(\mathbf{v})$ represents the total HPWL by Equation 1, and $O_m(\mathbf{v})$ represents the overlapping area between movable instances. The placement legalization is divided into two layers of iterative algorithms. The inner iteration uses a simulated annealing-based legalization algorithm to move the movable instances, and the outer iteration updates the optimization objective weight μ_o, with j and k representing the indices of the outer and inner iterations respectively.

The objective function's coefficient μ_o is initialized to 1.0, and μ_o is multiplied by β in each outer iteration, making the legalizer more proactive in reducing overlaps. In the inner loop, the legalization algorithm randomly selects a movable element and randomly determines its x and y-direction movement vectors within the movable radius $r_{j,k}$. It then computes the change in the objective function Δf, generates a random number $\tau \in (0, 1)$, and determines whether the new placement result is accepted by checking $\tau < \exp(-\Delta f / t_{j,k})$ [13].

5.2 Bump-Terminal Assignment

Upon completing the legalization process, both the chiplet placement region and terminal placement region are now free of overlaps.

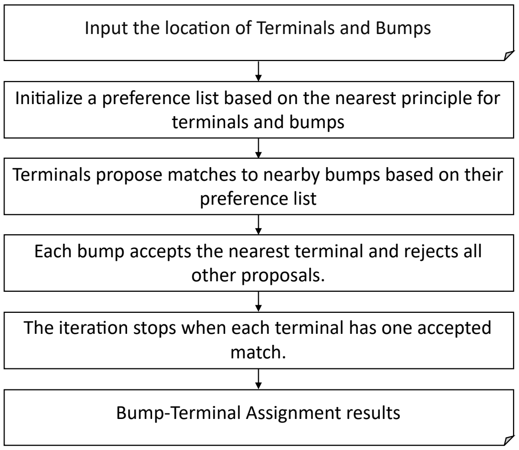

Input the location of Terminals and Bumps
Initialize a preference list based on the nearest principle for terminals and bumps
Terminals propose matches to nearby bumps based on their preference list
Each bump accepts the nearest terminal and rejects all other proposals.
The iteration stops when each terminal has one accepted match.
Bump-Terminal Assignment results

Figure 8: Bump-terminal assignment flowchart.

However, during packaging manufacturing, the C4-bumps under the RDL should conform to specific design rules, such as the size of the C4-bumps and the spacing between them. While the legalization process has ensured non-overlapping placements among terminals, it did not consider the precise arrangement of terminals according to the design rules for C4-bumps. Therefore, it is necessary to establish a correspondence between C4-bumps and terminals, ensuring compliance with C4-bump design rules while minimizing the HPWL.

We propose a bump-terminal assignment algorithm based on the Gale-Shapley algorithm. The Gale-Shapley algorithm is a stable marriage matching algorithm that ensures stable matches between two groups of individuals, allowing each person to find an optimal match. In this paper, we consider bumps as males and terminals as females and perform bump-terminal assignment employing the Gale-Shapley algorithm. The flowchart of the algorithm is shown in Figure 8.

The algorithm involves calculating the Euclidean distance between terminals and C4-bumps and storing these distances in a two-dimensional vector *distances* (lines 1-3). Then, for each terminal, the distances to bumps and the indices of these bumps are stored in a two-dimensional vector *disSort*, which is used for the subsequent bump-terminal matching (lines 4-6). Next, apply the Gale-Shapley algorithm to match terminals with C4-bump. This algorithm iteratively finds the best C4-bump match for each terminal and ensures that each C4-bump is matched with only one terminal(lines 7-19). Finally, based on the matching results, update the positions of the terminals to their corresponding C4-bump position. bump-terminal assignment algorithm's detailed flow is shown in Algorithm 2.

6 EXPERIMENTAL RESULTS

We implement the proposed framework in C++ language with compiler gcc 9.4.0 (-O3 optimization flag) and execute it on a Linux workstation with AMD CPU Ryzen 7 5800U with eight cores at 1.9GHz. We validated our algorithms using three benchmark sets (Table 2): (i) a modified version of the ISPD08 benchmarks from [10] with four and six chiplets (ii) the well-known floorplanning

Algorithm 2 bump-terminal Assignment Algorithm

```
Input: $V_t$, the placement results of terminals
       $V_b$, the locations of bumps
Output: $M_{b,t}$, the matches of bump $b$ to terminal $t$
for $v$ in $V_t$ and $b$ in $V_b$ do
    compute $distances[v][b]$ = Euclidean Distance$(v, b)$
end for
for $v$ in $V_t$ and $d$ in $distances[v]$ do
    Sort $d$ as $d_s$ ,$disSort[v] = d_s$
end for
while $v$ in $V_t$ not matched and $b$ in $disSort[v]$ do
    if $b$ doesn't match $v$ then
        match $b$ and $v$
        break
    else
        if $dis[v][b] < dis[M_{b,t}[b]]$ then
            unmatch $M_{b,t}[b]$
            match $b$ and $v$
            break
        end if
    end if
end while
return $M_{b,t}$
```

Table 1: Comparison with baseline methods

Method	C-P	T-P	C4-B-T/A
[16] method	Y	N	N
[3] method	Y	N	N
ours	Y	Y	Y

Table 2: Physical information of test cases

Case Name	N	C	T	W(μm)	H(μm)
ispd08m_4die_small_a1	1808	4	789	16324	16324
ispd08m_4die_big	12265	4	1033	27750	27850
ispd08m_4die_mid_a4	5326	4	1174	25542	25707
ispd08m_6die_big	1720	6	639	25542	25707
ispd08m_6die_mid_b1	14264	6	1192	25575	25740
ispd08m_6die_small_a2	7123	6	1162	12485	12485
mcnc_apte	97	9	73	10500	10500
mcnc_hp	83	11	45	4928	4200
mcnc_xerox	203	10	2	5831	6412
mesh_4_1	60	4	60	16000	19000
mesh_8_1	110	8	60	32000	19000
mesh_8_2	90	8	40	45000	33000
mesh_16_1	200	16	80	32000	38000
mesh_32_1	380	32	120	60000	38000
mesh_64_1	720	64	160	60000	76000

benchmarks MCNC with nine, ten, and eleven chiplets [15], and (iii) modified testases from (i) and extend up to 64 chiplets, constructed with the mesh network. The physical information for each testcase is shown in Table 2. N, C, and T respectively represent the number of wire nets, the number of chiplets, and the number of terminals. W and H represent the width and height of the RDL, respectively. We have open-sourced the code of our methodology.

6.1 Effectiveness of Chiplet-Terminal Co-Placement.

To the best of our knowledge, previous work only focused on placement chiplets based on fixed terminal locations, without achieving simultaneous placement of terminals and chiplets. The comparison of our method with the methods from [16] and [3] is shown in Table 1.

We compare our approach with the method proposed by Osmolovskyi [16] and [3], both of them use a Branch-and-Bound (B&B) algorithm to find the optimal placement result. The comparison results are shown in Table 3. Since [16] method and [3] method cannot simultaneously place terminals and chiplets, the test results in Table 3 remain consistent with their original work. However, our method includes chiplet-terminal coordinated placement in the testing process. The results show HPWL has an average improvement of 25.78% with chiplet-terminal co-placement in common testcases. The B&B-based methods are more efficient when the number of chiplets is small due to the manageable search space and the incorporation of pruning techniques. On the other hand, analytical methods still require a certain number of iterations for solving. However, as the number of chiplets increases, B&B-based methods face the challenge of combinatorial explosion in the search space. When the chiplet number reaches 10 (xerox), the execution time of [16] method exceeds 12 hours. There is some improvement in speed in [3] compared to [16], but the execution time for this case still exceeds 6 hours. In contrast, our method completes this case in just 9 seconds, with the highest run time in common cases only 11.18 minutes.

We design 6 large-scale chiplet testcases to evaluate placement performance in large-scale 2.5D IC designs, including 4, 8, 16, 32, and 64 chiplets. Due to [16] requires terminal positions to be fixed, we use the terminal positions obtained from our method. The comparison results with [16] method are shown in Table 4. When the number of chiplets reached 16, [16] method runtime exceeded 12 hours, while our method completed it in 4 minutes. Even when the number of chiplets reached 64, our method still finished in 23 minutes.

6.2 Effectiveness of Pin Density and Bump-Terminal Assignment

To improve the efficiency of our placement algorithm, we add pin density penalty to the optimization objective (Equation 4). Table 5 presents the comparison results. Here, W/BA&Wo/PD denotes that pin density is not included in the optimization objective but bump-terminal assignment is performed, while W/BA&W/PD denotes that pin density was included in the optimization objective and bump-terminal assignment is performed. The "Compare PD" column shows the comparison experimental results, showing that after adding pin density, the average runtime is reduced by 7%, and the HPWL remained mostly unchanged in the majority of cases.

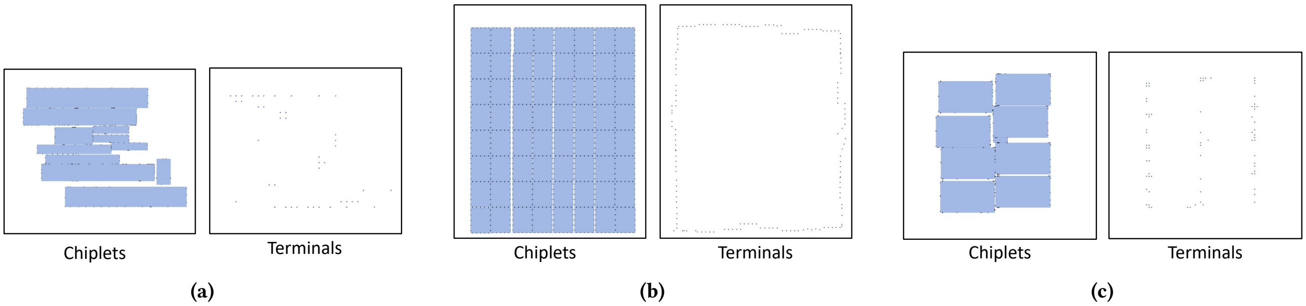


Figure 9: Partial RDL placement results: (a) hp; (b) mesh_64_1; (c) apte.

Table 3: Compare with B&B method algorithm results

Case	[16] method		[3] method*		Ours		Compare with [16]		Compare with [3]	
Name	HPWL (μm)	time (s)	HPWL (μm)	time (s)	HPWL (μm)	time (s)	Diiff. HPWL	Speedup (×)	Diff. HPWL	Speedup (×)
ispd08m_4die_big	58917800	2	58917800	1	45538312	338	0.77	0.005	0.77	0.003
ispd08m_4die_mid_a4	38143400	2	38143400	1	24198463	354	0.63	0.005	0.63	0.003
ispd08m_4die_small_a1	10871000	1	10871000	1	7099386	65	0.65	0.015	0.65	0.015
ispd08m_6die_big	62711300	3	62711300	1	57515199	671	0.92	0.004	0.92	0.001
ispd08m_6die_mid_b1	33771600	1	33771600	1	27279522	122	0.81	0.008	0.81	0.008
ispd08m_6die_small_a2	9005520	1	9005520	1	3766693	49	0.42	0.02	0.42	0.02
mcnc_apte	437506	1170	437510	187	131208	29	0.3	40.344	0.3	6.448
mcnc_hp	150258	11270	150140	349	131687	4	0.88	2817.500	0.88	87.250
mcnc_xerox	375150**	>12h	364300	22187	487639	9	1.3	/	1.3	2465.222

* Represents results from their paper as reference data for comparison.

** Represents the best result obtained within twelve hours.

Table 4: Large-scale cases compare with B&B method results

Case	[16] method		Ours		Diff.
Name	HPWL (μm)	Time (s)	HPWL (μm)	Time (s)	HPWL
mesh_4_1	20945	1	21866	47	1.04
mesh_8_1	50316	1	54280	187	1.08
mesh_8_2	107019	4	117815	225	1.10
mesh_16_1	/	>12h	97034	223	/
mesh_32_1	/	>12h	227406	777	/
mesh_64_1	/	>12h	362652	1348	/

During bump-terminal assignment, the bump diameter is fixed at 50μm, and the bump pitch is set to 100μm. The experimental results for comparison of the bump-terminal assignment algorithm are shown in Table 5. Here, Wo/BA&W/PD denotes without bump-terminal assignment but including pin density penalty, while W/BA&W/PD denotes with bump-terminal assignment and pin density penalty. The "Compare BA" column displays the comparison experimental results, indicating that the average increase in HPWL after executing the bump-terminal assignment algorithm is less than 1%, with the time overhead being approximately 1%. The experimental results confirm the effectiveness of the bump-terminal assignment algorithm. Due to space constraints, we present partial placement results in Figure 9.

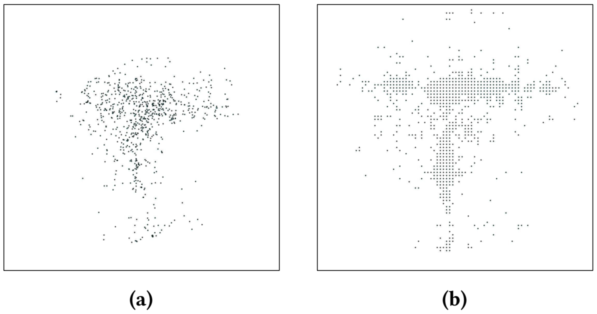


Figure 10: Comparison of BA results: (a) ispd08m_4die_small_a1 without BA; (b) ispd08m_4die_small_a1 with BA.

7 CONCLUSIONS

We develop an efficient chiplet-terminal co-placement methodology tailored for large-scale 2.5D IC Design. This methodology includes the analytical global placement, legalization, and bump-terminal

Table 5: Comparison of bump-terminal assignment & Pin Density algorithms results

Case	Wo/BA&W/PD		W/BA&Wo/PD		W/BA&W/PD		Compare BA		Compare PD	
Name	HPWL	Time	HPWL	Time	HPWL	Time	Diff.	Speedup	Diff.	Speedup
	(μm)	(s)	(μm)	(s)	(μm)	(s)	HPWL	(×)	HPWL	(×)
ispd08m_4die_big	45490436	376	45537544	428	45538312	338	1.00	0.88	1.00	1.27
ispd08m_4die_mid_a4	24196326	393	24198463	406	24198463	354	1.00	0.97	1.00	1.15
ispd08m_4die_small_a1	7086677	64	7099386	69	7099386	65	1.00	0.93	1.00	1.06
ispd08m_6die_big	57512771	791	64317485	873	57515199	671	1.00	0.91	0.89	1.30
ispd08m_6die_mid_b1	27262688	171	27279522	113	27279522	122	1.00	1.51	1.00	0.93
ispd08m_6die_small_a2	3752213	49	3766693	57	3766693	49	1.00	0.86	1.00	1.16
mcnc_apte	129521	28	131208	28	131208	29	1.01	1.00	1.00	0.97
mcnc_hp	130775	3	131687	4	131687	4	1.01	0.75	1.00	1.00
mcnc_xerox	487639	10	487639	9	487639	10	1.00	1.11	1.00	0.90
mesh_4_1	20807	50	21866	56	21866	47	1.05	0.89	1.00	1.19
mesh_8_1	52705	197	54280	190	54280	187	1.03	1.04	1.00	1.02
mesh_8_2	117815	225	118700	216	118700	226	1.01	1.04	1.00	0.96
mesh_16_1	96644	223	79786	227	97034	223	1.00	0.98	1.22	1.02
mesh_32_1	227586	771	227406	799	227406	777	1.00	0.96	1.00	1.03
Average							1.01	0.99	1.01	1.07

Assignment stages. In the global placement stage, we model placement density based on eDensity and add pin density penalty to enhance algorithm efficiency and quality. Next, we employ a simulated annealing-based legalization approach and a Gale-Shapely-based bump-terminal assignment to ensure placement compliance with design constraints. Results demonstrate that our chiplet-terminal co-placement methodology reduces average wirelength by 25.78% compared to prior work for the common testcases, with a maximum speedup of up to 4800× in testcases with more than 10 chiplets. Moreover, our method can complete the placement testcase includes 64 chiplets within 30 minutes.

REFERENCES

[1] Tung-Chieh Chen, Zhe-Wei Jiang, Tien-Chang Hsu, Hsin-Chen Chen, and Yao-Wen Chang. 2008. NTUplace3: An analytical placer for large-scale mixed-size designs with preplaced blocks and density constraints. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems* 27, 7 (2008), 1228–1240.

[2] Chung-Kuan Cheng, Andrew B Kahng, Ilgweon Kang, and Lutong Wang. 2018. Replace: Advancing solution quality and routability validation in global placement. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems* 38, 9 (2018), 1717–1730.

[3] Hong-Wen Chiou, Jia-Hao Jiang, Yu-Teng Chang, Yu-Min Lee, and Chi-Wen Pan. 2023. Chiplet Placement for 2.5D IC with Sequence Pair Based Tree and Thermal Consideration. In *2023 28th Asia and South Pacific Design Automation Conference (ASP-DAC)*. 7–12.

[4] Ayse Coskun, Furkan Eris, Ajay Joshi, Andrew B Kahng, Yenai Ma, Aditya Narayan, and Vaishnav Srinivas. 2020. Cross-layer co-optimization of network design and chiplet placement in 2.5-D systems. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems* 39, 12 (2020), 5183–5196.

[5] Hayguen. [n. d.]. Ooura's General Purpose FFT (Fast Fourier/Cosine/Sine Transform) Package. https://github.com/hayguen/OouraFFT. GitHub repository.

[6] Yuan-Kai Ho and Yao-Wen Chang. 2013. Multiple chip planning for chip-interposer codesign. In *Proceedings of the 50th Annual Design Automation Conference*. 1–6.

[7] Meng-Kai Hsu, Yao-Wen Chang, and Valeriy Balabanov. 2011. TSV-aware analytical placement for 3D IC designs. In *Proceedings of the 48th Design Automation Conference*. 664–669.

[8] Shin-Puu Jeng. 2014. CoWoS™ technologies. In *Proceedings of Technical Program - 2014 International Symposium on VLSI Technology, Systems and Application (VLSI-TSA)*. 1–1.

[9] Myung-Chul Kim, Dong-Jin Lee, and Igor L Markov. 2011. SimPL: An effective placement algorithm. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems* 31, 1 (2011), 50–60.

[10] Wen-Hao Liu, Min-Sheng Chang, and Ting-Chi Wang. 2014. Floorplanning and signal assignment for silicon interposer-based 3D ICs. In *2014 51st ACM/EDAC/IEEE Design Automation Conference (DAC)*. 1–6.

[11] Jingwei Lu, Pengwen Chen, Chin-Chih Chang, Lu Sha, Dennis Jen-Hsin Huang, Chin-Chi Teng, and Chung-Kuan Cheng. 2013. FFTPL: An analytic placement algorithm using fast fourier transform for density equalization. In *2013 IEEE 10th International Conference on ASIC*. 1–4.

[12] Jingwei Lu, Pengwen Chen, Chin-Chih Chang, Lu Sha, Dennis J-H Huang, Chin-Chi Teng, and Chung-Kuan Cheng. 2014. ePlace: Electrostatics based placement using Nesterov's method. In *Proceedings of the 51st Annual Design Automation Conference*. 1–6.

[13] Jingwei Lu, Hao Zhuang, Pengwen Chen, Hongliang Chang, Chin-Chih Chang, Yiu-Chung Wong, Lu Sha, Dennis Huang, Yufeng Luo, Chin-Chi Teng, et al. 2015. ePlace-MS: Electrostatics-based placement for mixed-size circuits. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems* 34, 5 (2015), 685–698.

[14] Yenai Ma, Leila Delshadtehrani, Cansu Demirkiran, José L Abellan, and Aiav Joshi. 2021. TAP-2.5 D: A thermally-aware chiplet placement methodology for 2.5 D systems. In *2021 Design, Automation & Test in Europe Conference & Exhibition (DATE)*. 1246–1251.

[15] Theodore W. Manikas. [n. d.]. MCNC Benchmark Netlists for Floorplanning and Placement. https://s2.smu.edu/~manikas/Benchmarks/MCNC_Benchmark_Netlists.html. Accessed: 2024.

[16] Sergii Osmolovskyi, Johann Knechtel, Igor L. Markov, and Jens Lienig. 2018. Optimal die placement for interposer-based 3D ICs. In *2018 23rd Asia and South Pacific Design Automation Conference (ASP-DAC)*. 513–520.

[17] Peter Spindler, Ulf Schlichtmann, and Frank M Johannes. 2008. Kraftwerk2—A fast force-directed quadratic placement approach using an accurate net model. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems* 27, 8 (2008), 1398–1411.

[18] Chien-Fu Tseng, Chung-Shi Liu, Chi-Hsi Wu, and Douglas Yu. 2016. InFO (wafer level integrated fan-out) technology. In *2016 IEEE 66th Electronic Components and Technology Conference (ECTC)*. 1–6.