Physical Design Final Report

3D Placement with D2D Vertical Connections

Team 2
R10943182 黃旭鈺
R10943147 李彥緯
R10943093 謝秉翰

- **Problem Description:**

The problem is 3D placement which focuses on 2-die face-to-face configuration with std cells only. Figure 1 shows the face-to-face vertically connected with hybrid bonding terminals. The contestants need to develop a 3D placer engine to place and optimize the 2 dies std cell placement and the hybrid bonding terminal placement to minimize the total HPWL of the 2 dies. The center point of the hybrid bonding terminals needs to be included in the HPWL calculation. All the cells need to be placed on row and cannot overlap with each other. All the cells would be single row height of the corresponding technology of the die. Also, all the given constraints need to be satisfied. [CADContest 22]

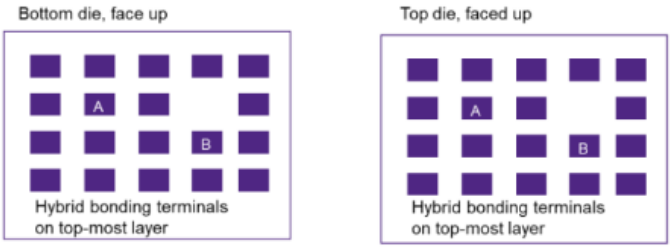


Figure 1: Face-to-face vertically connected with hybrid bonding terminals

The evaluation constraints are shown below, [CADContest 22]

1. Max placement utilization constraint must be satisfied
2. All the given instances must be placed on either top die or bottom die
3. All the instances must be on row without overlap
4. Hybrid bonding terminal spacing constraint must be satisfied
5. Crossing-die nets must have 1 and only 1 hybrid bonding terminal
6. The minimum resolution of all the coordinate values is integer
7. Runtime limit is 1hr for each case in the evaluation machine

The final evaluation score is like equation (1). Both the HPWL of top die and HPWL of bottom die should consider the bonding terminal as figure 2 shown.

$$Evaluation\ score = HPWL\ of\ top\ die + HPWL\ of\ bottom\ die \qquad (1)$$

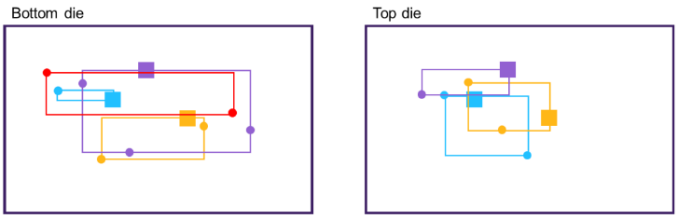


Figure 2: HPWL of top/bottom die(s)

- ## Proposed Techniques:

  - **Overall Flowchart:**

    The overall flowchart is shown in Figure 3, and the detail of each stage will be described in the following paragraphs.

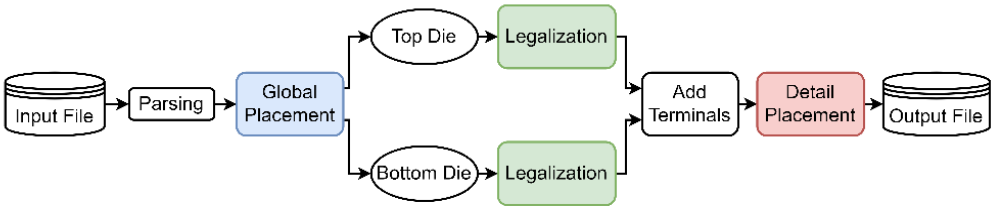


Figure 3: Overall flowchart

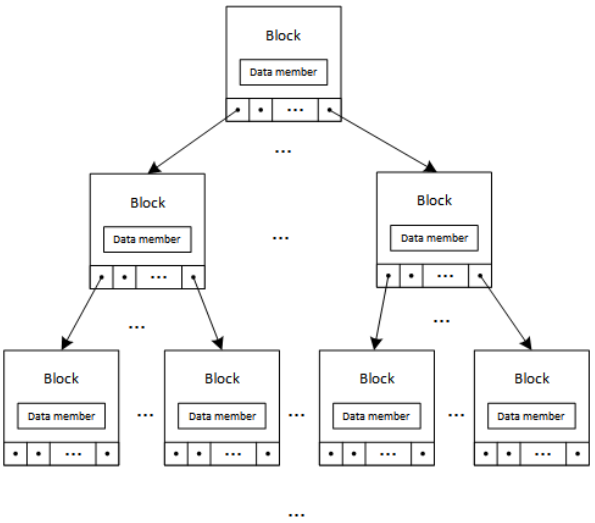
  - **Data Structure:**

    The data structure is modified from Programming Assignment 3 of Physical Design Course, the following are the modifications,

    1. Save a variable in Inst.h to save the die information of each instance.
    2. Save x_offset and y_offset for each pin in Pin.h.

    (More details can be explored in the source code)

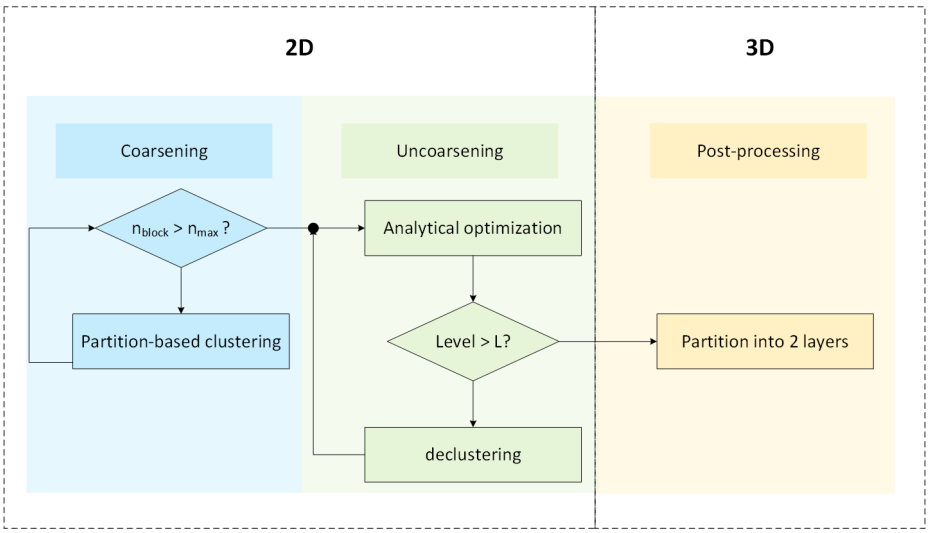
    The multilevel framework deploys the following tree structure. The number of children in a block doesn't exceed a user-defined number n_{max}. The analytical optimization is performed on the blocks at the same level (the vertices at the same level in the tree structure).



    There are modules for partitioning as well, and the data structure is nearly the same as that of Programming Assignment 1, which is based on FM heuristics.

  - **Global Placement:**

    - **Flow Chart**



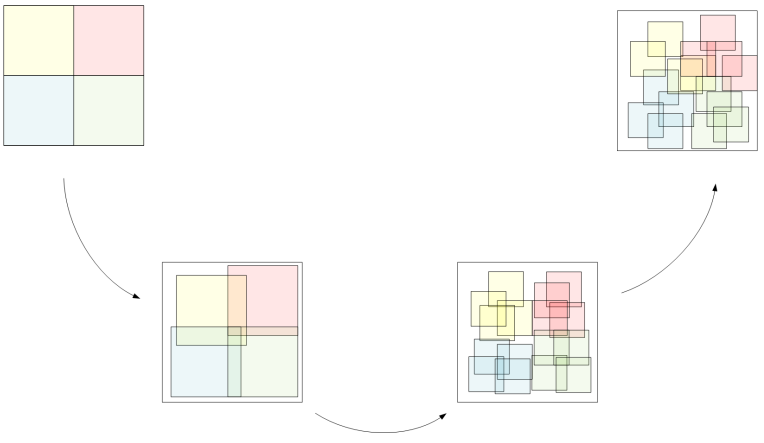
- **Multilevel Framework**

  n_{max}: the maximum number of blocks clustered

  l_{max}: the minimum layer to perform analytical optimization

  If the number of blocks in the current level is greater than n_{max}, we partition the blocks into n_{max} groups and cluster the ones in the same group. If the number of blocks in a groups is still greater than n_{max}, we further do the clustering recursively in each group.

  After clustering blocks properly, we can perform analytical optimization in levels no lower than l_{max}. Note that the coarsest stage is of the highest level by our definition. We move on to the optimization in the next level after declustering the blocks in the current level. The process is illustrated in the following figure.



- **Analytical Placement Model (refer to the lecture note)**

  Objective function: $W(x,\ y)\ +\ \lambda\Sigma(D_b\ -\ M_b)^2$, where

  D_b: density of bin b, M_b: max. density of bin b

$$W(x,\ y)\ =\ \Sigma_{e\in E}\left(\frac{\Sigma x_i\ exp(x_i/\gamma)}{\Sigma exp(x_i/\gamma)}-\frac{\Sigma x_i\ exp(-x_i/\gamma)}{\Sigma exp(-x_i/\gamma)}+\frac{\Sigma y_i\ exp(y_i/\gamma)}{\Sigma exp(y_i/\gamma)}-\frac{\Sigma y_i\ exp(-y_i/\gamma)}{\Sigma exp(-y_i/\gamma)}\right)$$

$$D_b = \Sigma_{v \in V} O_x(b, v) O_y(b, v)$$

$$O_x = p(l - x)p(l + x), O_y = p(l - y)p(l + y),\ l = \frac{binW}{2}$$

$$p(x) = 1,\ 0.5 \le ax \qquad \text{[Hsu 14]}$$

$$1 - 2(ax - 0.5)^2,\ 0 \le ax \le 0.5$$

$$2(ax + 0.5)^2,\ -0.5 \le ax \le 0$$

$$0,\ ax \le -0.5$$

- **Pseudo-code (format reference: Fig,3 in [Chen 05])**

Multilevel Global Placement

INPUT:

C_0: a circuit with 2-layer dies

n_{max}: the largest # of blocks clustered in each level

l_{max}: the lowest level to do uncoarsening

OUTPUT:

(x_*, y_*): optimized positions of each instance without considering overlap and bonding terminals

$level \leftarrow 0$

While $number(C_{level}) > n_{max}$

$C_{level} \leftarrow PartitionClustering(C_{level})$

$level \leftarrow level + 1$

Initialize block positions by partition-based constructive method

for $currentLevel \leftarrow level$ down to l_{max}

Initialize bin grid size

$\lambda_0 \leftarrow 1$

$iter_{max} \leftarrow \sqrt{number(C_{level})}$

for $i \leftarrow 1$ to $iter_{max}$

solve min $W(x, y) + \lambda_i \Sigma (D_b - M_b)^2$

$\lambda_i \leftarrow 2 \times \lambda_i$

if $blocks\ scattered\ enough$

break

$Decluster(C_{currentLevel})$

- **Legalization:**

The purpose of legalization is to eliminate all the overlaps by perturbing the modules as little as possible [PDF PD]. The target is to place instances at the position with smallest displacement with a given range.
Our implementation is based on a three-stage legalizer with Diffusion-based method, Tetris-like legalization, and Single-row optimization (Figure below).

- **Diffusion-based method [Ren 07]**

This stage is used to avoid the high density placement situation on the die. In order to sparse the instances properly on the die. We cut the die into multiple bins and calculate the density of each bin. If there are any high density bins, we will move some instances from the high density bins to the nearby low density bins. Please know that this movement is updated by considering all the nearby density and this operation will be done iteratively until all the high density bins are fixed.

- **Tetris-like legalization [Oikonomou 18]**
  The legalization method we choose is Tetris-like legalization. First, we sort all the instances by their x-coordinate for the priority of legalization. Second, we set the distance constraint for the new location, which means the new position cannot be too far away from the original one. Next, we start to do the legalization according to the priority of each instance. This step will be done iteratively until all the instances are legalized. However, if the legalization fails, we will loosen the distance constraint and redo the legalization again.

- **Single-row optimization**
  This stage is used to further optimize the wirelength after legalization. We try to slightly move each instance on the row to check whether the wirelength can be improved or not. Since this process is done by greedy algorithms, the results can only be improved slightly.

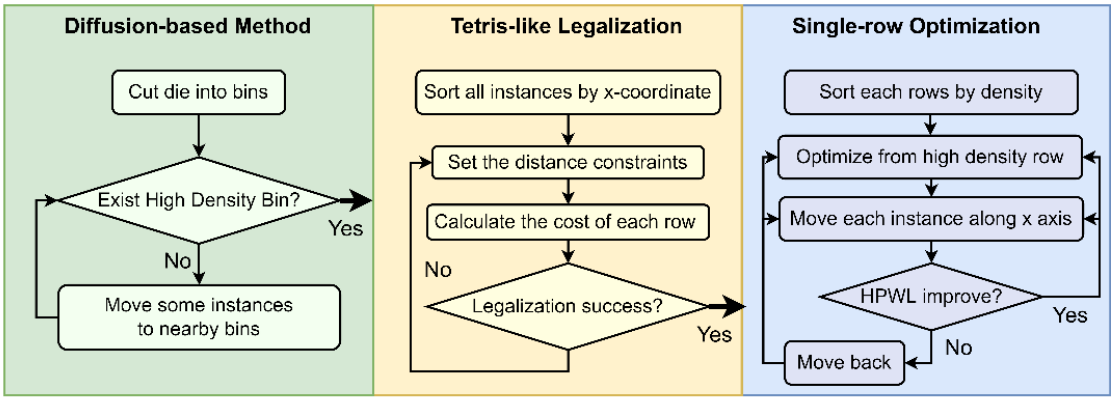


Figure: Three stages for legalization

- **Bonding Terminal Placement:**

We try to build the bonding terminal after the legalization stage. By considering the information of terminal sizeX, terminal sizeY, and terminal spacing, we can split the die into grids (Figure). Each grid is a possible position for the terminal to place. Our current method can be separated into three steps. First, we determine which net needs to add a bonding terminal. Second, we randomly select several grid positions to place the terminal, and calculate the wire length. Third, we choose the best position with shortest wire length and greedily place the terminal there. Please know that in order to use

this method, the case should satisfy the equation below, which means the number of grid positions should be larger or equal to the number of terminals (nets).

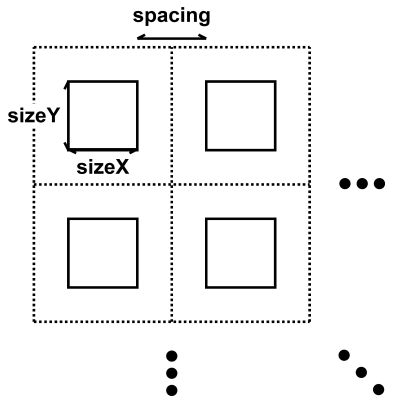


Figure: Grid-based terminal placement

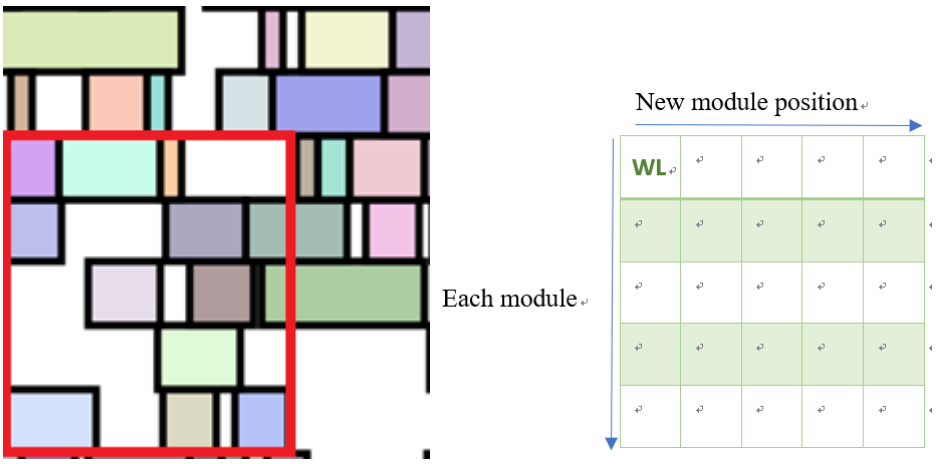
$$\#\ Nets < \left(\frac{Die_{width}}{Teminal_{sizeX} + Terminal_{spacing}}\right) \times \left(\frac{Die_{height}}{Teminal_{sizeY} + Terminal_{spacing}}\right)$$

- **Detail Placement:**

In this problem, we use NTUplace3**[Chen 08]** as reference code, but few adjustments are required to complete this problem. First, we split cell matching into top and bottom dies separately, second, when calculating the net length for the matching problem, we need to consider wirelength towards the terminal, third, both the parameters and data structure need to be changed completely, below we will introduce the detailed part of cell matching, which are from the NTUplace3 code.

- **Detailed code flow**

First we set up the window size, while the window will scan through the chip iteratively starting from left bottom coordinate of the die, next we save the modules and the rows which are in the window, and sort the modules by its width decreasingly, then for the modules whose width that is equal or smaller than the largest width, we get their position and moduleID, for the cases that are smaller, we also subtract their width and get their new position since their will still be empty spaces that can be placed. Then before calculating the bipartite matching problem, we have to put the empty spaces back and remove the modules we choose from the rows. Finally we solve the problem with the shortest augmenting path algorithm. First build a matrix, one row at a time, we calculate the modules to put on different positions and calculate their wirelength, also we set a new position for the module with the shortest wire length in case the modules aren't independent, then we start another new row. At last we calculate the position for each module with the total shortest wire length (weight for matching problem). Below are two figures of the window scan process and the matrix that saves wirelength in order to solve the bipartite problem.



- **Results:**

This result is from our calculation of HPWL:

	case1	case2	case3	case4
HPWL	148	6,363,041	167,932,940	2,306,581,027
Time (sec)	0	2	40	52

Table1: Wire length and runtime of case1~case4

- **case1** (# Insts = 8, # Nets = 6)

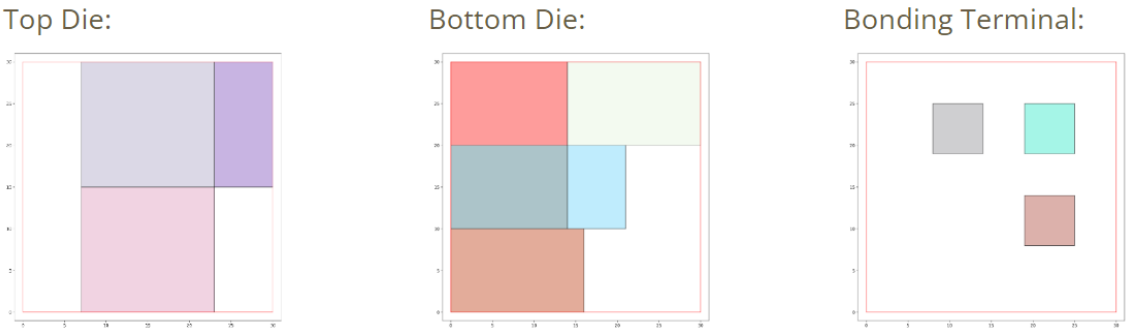


Figure: Visualize result for case1

```
[TopDie]  Area(used/max): 670/900 Util: 74.44 Max Util: 80
[BottomDie] Area(used/max): 585/900 Util: 65.00 Max Util: 90
Total HPWL for this design is 135
```

Figure: Result from Evaluator for case1

- **case2** (# Insts = 2,735, # Nets = 2,644)

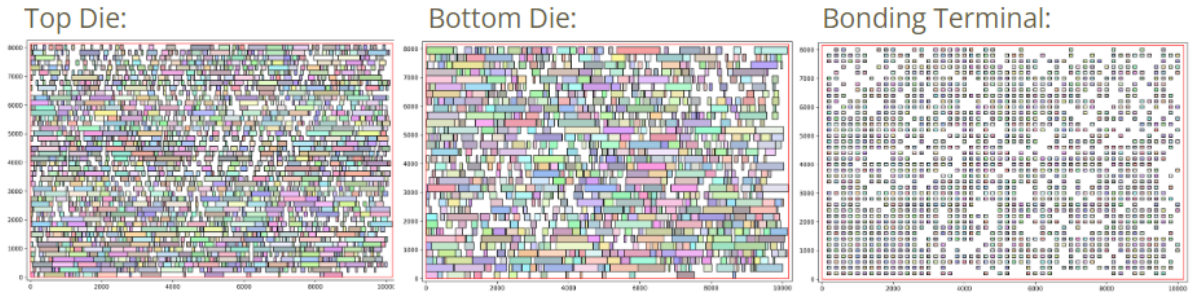


Figure: Visualize result for case2

```
[TopDie]  Area(used/max): 53491504/82936425 Util: 64.50 Max Util: 70
[BottomDie] Area(used/max): 60911676/82936425 Util: 73.44 Max Util: 75
Total HPWL for this design is 6362089
```

Figure: Result from Evaluator for case2

- **case3** (# Insts = 44,764, # Nets = 44,360)

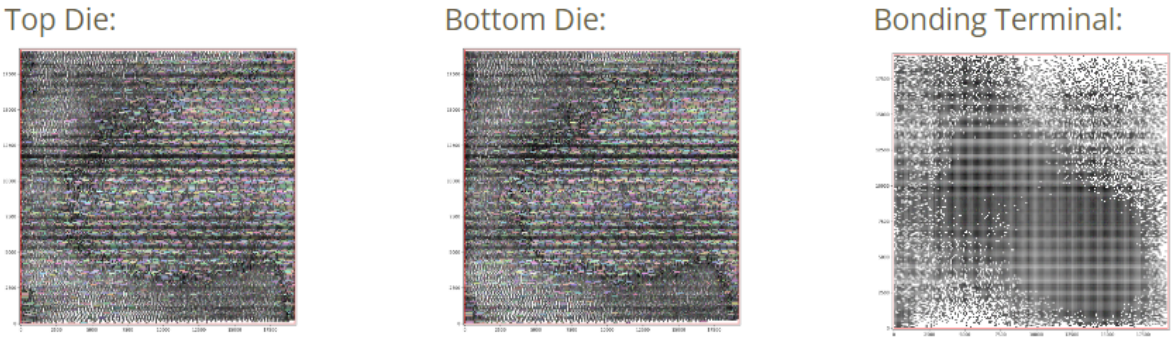


Figure: Visualize result for case3

```
[TopDie]  Area(used/max): 278216970/369254080 Util: 75.35 Max Util: 78
[BottomDie] Area(used/max): 288017730/369254080 Util: 78.00 Max Util: 78
Total HPWL for this design is 167938173
```

Figure: Result from Evaluator for case3

- **case4** (# Insts = 220,845, # Nets = 220,071)

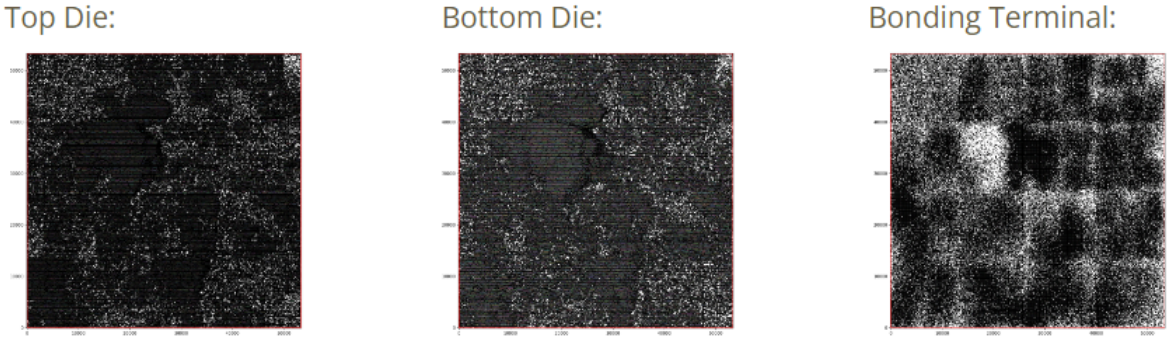


Figure: Visualize result for case4

```
[TopDie]  Area(used/max): 1840438932/2838171970 Util: 64.85 Max Util: 66
[BottomDie] Area(used/max): 1954095985/2838171970 Util: 68.85 Max Util: 70
Total HPWL for this design is 2307283610
```

Figure: Result from Evaluator for case4

- **Future Work:**
  - Consider the top and bottom dies together to optimize the results.
  - Move the bonding terminal placement to the global placement stage to improve the wire length. (Consider the case in figure below, if a 2-pin net is separated on both dies, we might get the worst case shown on the left side. Therefore, considering terminal in the early stage may solve this problem.)

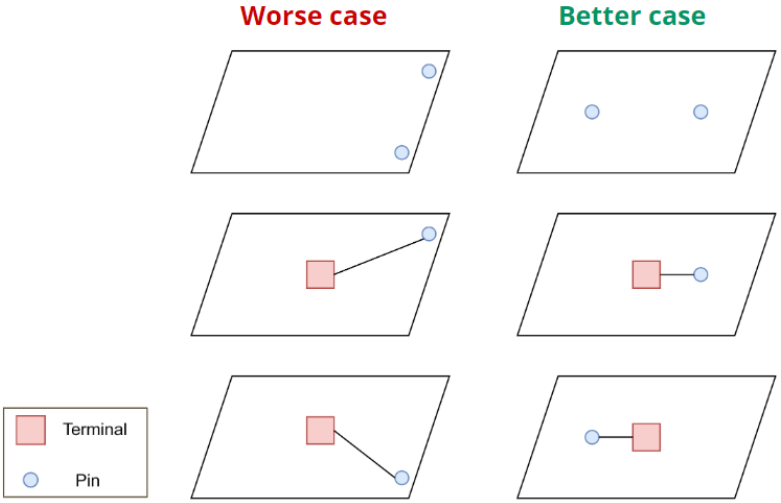


Figure: Different results for the 2-pin net on both dies

  - Combine with NTUplace3 code to get a better performance.
  - Analyze the netlist to improve the performance. (Table 2)

case1	case2	case3	case4
# 2-Pins Net = 3 # 3-Pins Net = 3	**# 2-Pins Net = 1663** # 3-Pins Net = 534 # 4-Pins Net = 180 # 5-Pins Net = 123 # 6-Pins Net = 41 # 7-Pins Net = 19 # 8-Pins Net = 17 … # 31-Pins Net = 3 # 32-Pins Net = 2 # 33-Pins Net = 6 # 34-Pins Net = 1 # 36-Pins Net = 1 # 37-Pins Net = 1 # 38-Pins Net = 1 # 60-Pins Net = 1 # 65-Pins Net = 1	**# 2-Pins Net = 27523** # 3-Pins Net = 8478 # 4-Pins Net = 3622 # 5-Pins Net = 1293 # 6-Pins Net = 470 # 7-Pins Net = 450 # 8-Pins Net = 339 … # 26-Pins Net = 4 # 27-Pins Net = 9 # 28-Pins Net = 8 # 29-Pins Net = 13 # 30-Pins Net = 9 # 31-Pins Net = 102 # 32-Pins Net = 49 # 33-Pins Net = 55 # 3165-Pins Net = 1	**# 2-Pins Net = 151300** # 3-Pins Net = 25833 # 4-Pins Net = 12540 # 5-Pins Net = 7919 # 6-Pins Net = 3964 # 7-Pins Net = 2599 # 8-Pins Net = 1864 … # 28-Pins Net = 173 # 29-Pins Net = 160 # 30-Pins Net = 178 # 31-Pins Net = 182 # 32-Pins Net = 212 # 33-Pins Net = 557 # 34-Pins Net = 458 # 35-Pins Net = 269 # 36-Pins Net = 514

Table 2: Pin information of case1~case4

  - Follow the upcoming schedule:
    - Alpha test submission: 2022/06/13 17:00:00 (GTM+8)
    - Beta test submission: 2022/07/22 17:00:00 (GTM+8)
    - Final submission: 2022/08/30 17:00:00 (GTM+8)

- ## References:

[CADContest 22]
http://iccad-contest.org/Problems/CADContest_2022_Problem_B_20220215.pdf
[Chen 05] Tung-Chieh Chen, Tien-Chang Hsu, Zhe-Wei Jiang, and Yao-Wen Chang. 2005. NTUplace: a ratio partitioning based placement algorithm for large-scale mixed-size designs. In Proceedings of the 2005 international symposium on Physical design.
[Chen 08] T. Chen, Z. Jiang, T. Hsu, H. Chen and Y. Chang, "NTUplace3: An Analytical Placer for Large-Scale Mixed-Size Designs With Preplaced Blocks and Density Constraints," in IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, 2008.
[Ho 10] Tsung-Yi Ho and Sheng-Hung Liu, "Fast legalization for standard cell placement with simultaneous wirelength and displacement minimization," 2010 18th IEEE/IFIP International Conference on VLSI and System-on-Chip.
[Hsu 14] M.-K. Hsu, Y.-F. Chen, C.-C. Huang, S. Chou, T.-H. Lin, T.-C. Chen, and Y.-W. Chang, "NTUplace4h: A novel routability-driven placement algorithm for hierarchical mixed-size circuit designs," IEEE Trans. Computer-Aided Design of Integrated Circuits and Systems (TCAD), 2014.
[PDF PD] http://cc.ee.ntu.edu.tw/~ywchang/Courses/PD_Source/EDA_placement.pdf
[Ren 07] H. Ren, D. Z. Pan, C. J. Alpert, P. G. Villarrubia and G. Nam, "Diffusion-Based Placement Migration With Application on Legalization," in IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, 2007.
[Oikonomou 18] P. Oikonomou, A. N. Dadaliaris, T. Loukopoulos, A. Kakarountas and G. I. Stamoulis, "A Tetris-based legalization heuristic for standard cell placement with obstacles," 2018 7th International Conference on Modern Circuits and Systems Technologies (MOCAST), 2018.

(Refer to NTUplace3 code)

- ## Work Distribution:

Global Placement	r10943182 黃旭鈺
Legalization	r10943093 謝秉翰
Detail Placement	r10943147 李彥緯