3D Placement with D2D Vertical

組員:

台灣科技大學 資訊工程系 B10715041 馬孝傑

台灣科技大學 資訊工程系 B10732027 吳秉寰

台灣科技大學 資訊工程系 B10732014 何雨澂

台灣科技大學 資訊工程系 B10732040 李宇哲

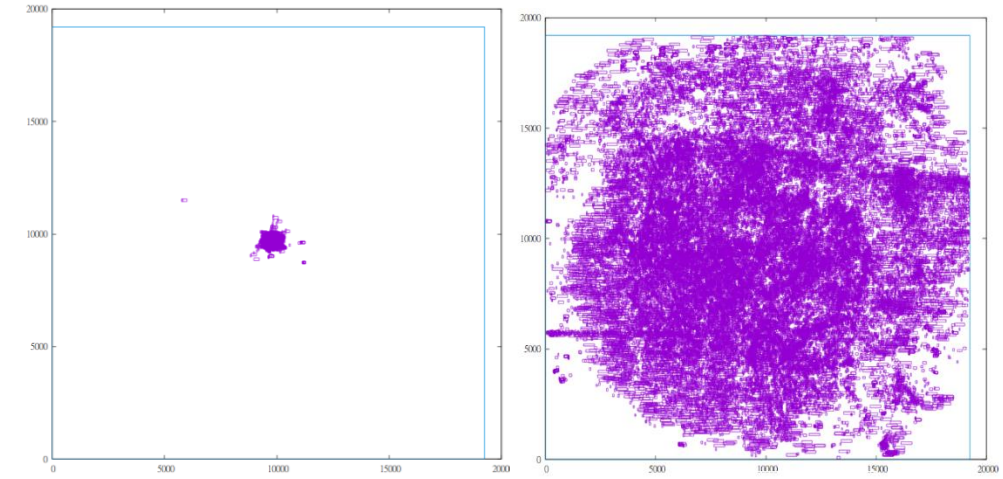
Table of Contents

1. Global placement

在此階段會將所有的 instance cell 在一個 die 上的面積進行 placement。目標為最小化 HPWL 的條件下，降低 placement 的密度。Objective function 如下:

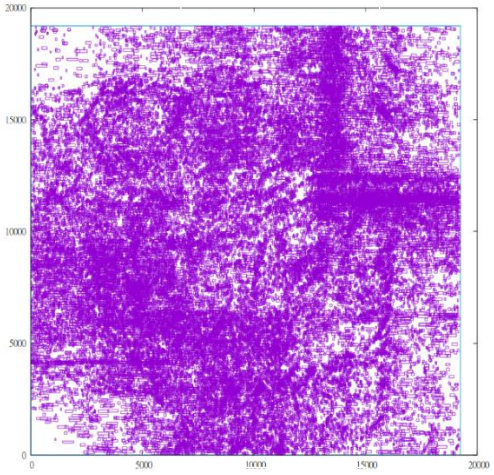
$$\min \underbrace{W(x,y)}_{\text{wirelength}} + \lambda \underbrace{\sum (D(x,y) - Mb)^2}_{\text{density}}$$

在 objective function 中 wirelength 的部分用 HPWL 來計算，而 density 部分則是對每個 bin 來計算密度，而 λ 係數調整 density 在 objective function 所佔的比重。為了讓 objective function 可微分，wirelength 的部分會使用 LSE model 來估算，density 部分則會採用 Bell-shaped model 來估算。剛開始，λ 係數會設為 0，讓 design 以 HPWL 為主來做 placement。接著調整 λ 係數，讓 wirelength 和 density 在 objective function 中所貢獻的比重相同，之後 λ 係數設為前一輪的兩倍，使 cell 慢慢地散開來，以此降低 density。圖 1 為示意圖。

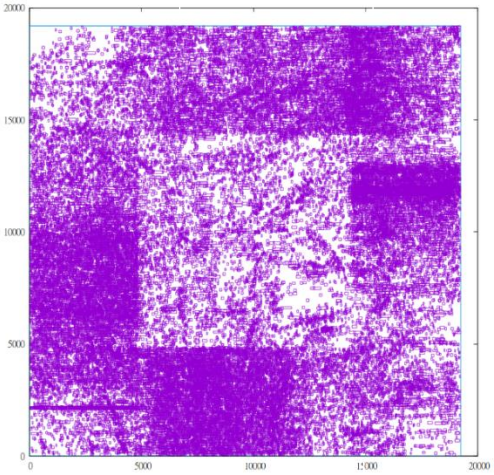


(a) Round 1

(b) Round 2



(c) Round 3



(d) Round 4

圖 1。

2. Cell Legalization

因 global placement 是將所有 cell 都先放在同一塊 die 上面做擺放，因此在做 legalization 之前得先做 partition 來決定每一 cell 要放在哪一塊 die 上。我們使用 FM 來做 partitioning，目標為使 hybrid bonding terminals 的數量越少越好，同時滿足兩塊 die 的 cell 總面積不超過 max utilization。每個 FM pass 若有多個 gain 一樣的 cell 則優先從空位較小的 die 開始挑選(將此 cell 放到空位較大的 die)。空位的定義如下:

Max Utilization * Die 的大小 - 已擺放的 cell 總面積。

之後開始進行 cell legalization，我們使用 abacus 的做法來實作。將 cell 放到 row 上並尋找最佳 x 做標的行為可用底下 quadratic program 來描述:

$$\min \sum_{i=1}^{N_r} e(i)\,[x(i)-x'(i)]^2 \qquad (1)$$

$$\text{s.t.} \quad x(i)-x(i-1) \geq w(i-1) \quad i=2,...,N_r \qquad (2)$$

其中 e(i)為第 i 個 cell 的權重，可為 pin 的數量等，w(i)為第 i 個 cell 的寬度，x(i)為第 i 個 cell 的左下角 x 座標。此式希望最小化 weighted 的總移動距離平方能最小化，同時 row 上每個 cell 不互相 overlap 且保持順序，移動距離指的是放上 row 後的位置和 global placement 結果位置的距離。因權重均為正，此式會是 convex。但要解有>=限制的 quadratic program 會很花時間，因此假設所有 cell 均相鄰，則式 2 的>=可改為=，且每個 cell 的座標可由第一個 cell 的座標 x(1)來算出，如下式:

$$x(i) = x(1) + \sum_{i-1}^{i-1} w(k) \quad i=2,...,N_r \qquad (3)$$

將此式代回 1 式會得到變數只剩 x(1)的 quadratic function，此時 function 的最小值只需求出微分後等於 0 的解則為最佳位置 x(1):

$$\underbrace{\sum_{i=1}^{N_r} e(i)}_{e_c}\, x(1) - \underbrace{\left[e(1)x'(1) + \sum_{i=2}^{N_r} e(i) \left[x'(i) - \sum_{k=1}^{i-1} w(k) \right] \right]}_{q_c} = 0 \qquad (4)$$

由此可知，要求出最佳 x(1)只需算 qc/ec 這一 linear equation，非常迅速。

但實際上一條 row 不會全部的 cell 都相鄰，因此假設 row 上有多個 cluster，每個 cluster 中就都是相鄰的 cell，並且只要計算 qc/ec 就能得出此 cluster 的最佳位置。若有 cluster 計算完位置後與前面 cluster 發生 overlap 則合併所有 cell 為一 cluster 並重新計算最佳位置。

在實作過程中，有改進原論文的演算法的幾個地方，其一為論文中的 place row 分成 trial mode 和 final mode，trial mode 只用來尋找最佳 Y 座標，因此不希望改到 row 上 cluster 和 cell 的座標，因此他們的 placerow 每次都重新計算

row 上所有 cell 的座標，但實際上因為在擺放的時候所有 cell 是依照 x 座標的順序來放，因此只需在 trial mode 的 place row 前將原值紀錄後，place row 中便只需擺放最新的 cell，結束後再將記錄的值寫回，將提升不少速度。其他改進點還有原論文的 Algorithm 2: PlaceRow 中，第三行如下:

3 | **if** $i = 1$ **or** $x_c(c) + w_c(c) \leq x'(i)$ **then**

若要放的 cell 為此 row 上的第一個 cell 或是跟上個 cluster 的距離>=上個 cluster 的寬度則新建一 cluster，此處應把等號拿掉，因若等號成立，表示此時新 cell 跟上個 cluster 是相鄰的，那只需將此 cell 加入上個 cluster 就好，不需建立新的 cluster。

還有一點一樣為 Algorithm 2: PlaceRow 中，第八行後面，應也要加上 Collapse，因若加入的新 cell 的位置是違法的(超出 die 邊界)，且之後沒有別的 cell 加入此 cluster，則不會進行違法檢查。若在第八行後加上 Collapse，就能對此 cluster 進行邊界檢查，並在合法化(移回界內)後遞迴檢查有無與之前的 cluster 發生 overlap，若有就進行 cluster 合併。

3. Detailed Placement

在 Global placement 後執行 Detailed placement，可以使得 design 的 HPWL 更近一步縮減，圖 2 為 Detailed placement 之 flow。

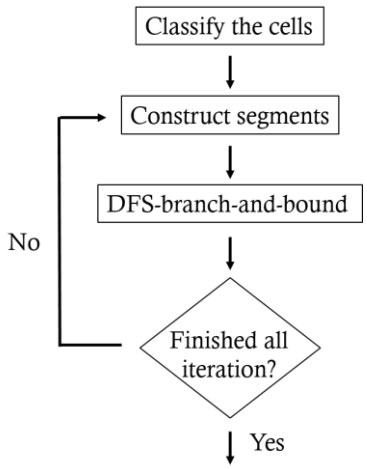


圖 2。Detailed placement 的流程圖

3.1 Classify the cell

我們建立一個類似於 hash table 的 array，同一個 bucket 中的 slots 所存放之 instance cells，皆位於同一個 die 和同一個 row 上。並且同一個 bucket 內的 instance cells 會經由 x 座標排序。

3.2 Construct segments

在每一個 row 上，我們都會建立複數個 segments，方法為：從 row 的左側開始，挑出連續(但不必相連)三個 instance cell 組成一個 segment；segments 之間不會有重複的 instance cells。若是一個 row 中剩餘的 instance cells 不足三個，則以剩餘的 instance cells 組成一 segment。

Segment 的高度同該 row 之高度，x 方向上的 boundary 則採用以下規則：

(1) 若此 segment 為該 row 之最左側，則其左側的 boundary 為此 row 之最左座標，否則為此 segment 之最左的 cell instance 與前一個 segment 最右的 cell instance 的中間座標。

(2) 若此 segment 為該 row 之最右側，則其右側的 boundary 為此 row 之最右座標，否則為此 segment 之最右的 cell instance 與後一個 segment 最左的 cell instance 的中間座標。

為了 timing 的考量，我們會對 segment 在 x 方向上的 boundary 進行調整：

(1) 若一 segment 中的兩個連續的 cell instances 相距太遠，則會將此 segment 拆分為二。

(2) 若一 segment 中的最左或最右 cell instance 與 boundary 相距太遠，則會將 boundary 朝 cell instance 挪近。

圖 3 為建構 segment 的一個例子，圖中藍色長方形代表 instances cell，而涵蓋 instance cells 的黑色外框則代表 row 的範圍。圖 4 則為調整 segment boundary 的例子，圖中藍色長方形代表 instances cell，而涵蓋 instance cells 的黑色外框則代表 segment 的範圍。

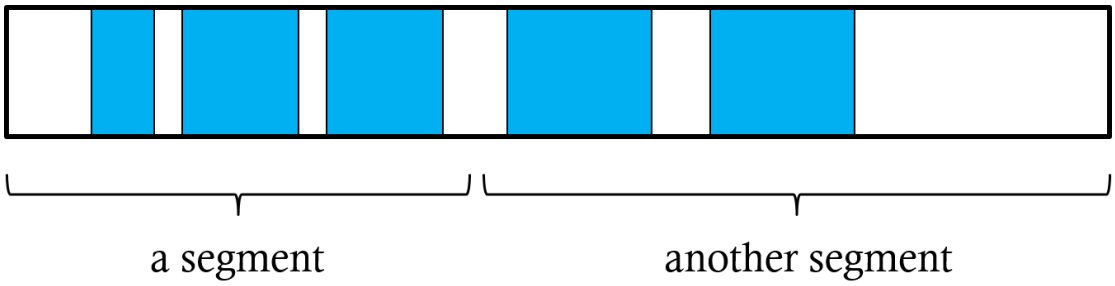


圖 3。建構 segment 的一個例子。

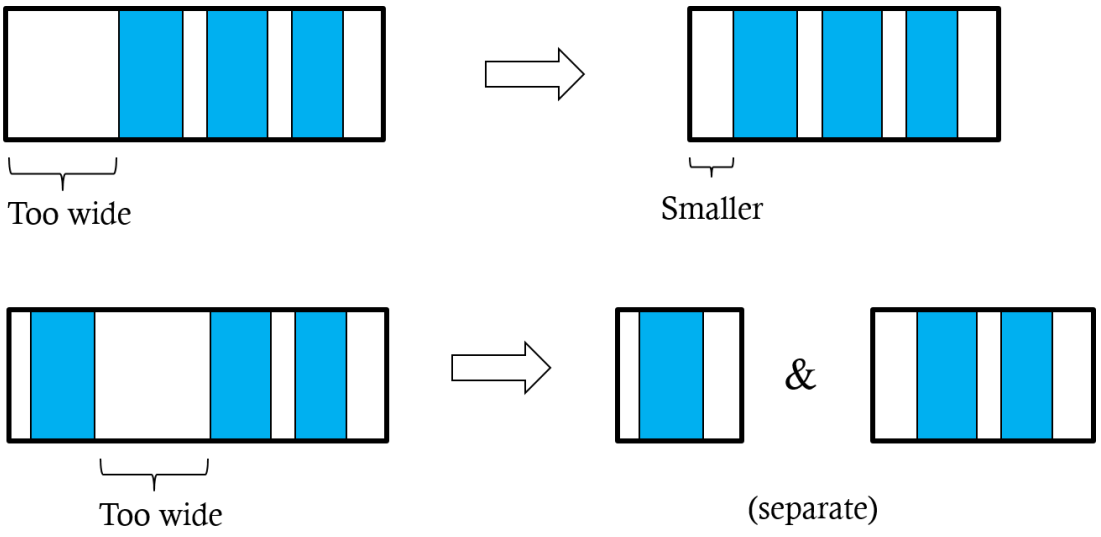


圖 4。調整 segment boundary 的例子。

3.3 DFS-branch-and-bound

建構出 segment 後，會對這些 segment 依次進行 DFS-branch-and-bound：

(1) 將 segment 中的第一個 instance cell(順序自訂)擺放在 free space 的最左側，segment 中其餘 instance cell 則擺放在剩下的 free space 上。

(2) 每個 instance cell 一開始擺放都是放置於 free space 的最左側。

(3) 一個 instance cell 在擺放後若想移動位置，則向右移動一個 step size，同一 die 上的 segment 之 step size 相同，數值自訂。

(4) 若一個 cell 嘗試過所有可擺放的位置後，則將其拔除並移動上一個 instance cell 之位置。

(5) 若是某一 instance cell 擺放至某一位置而使得整體 HPWL 大於目前最佳解，則不必擺放後續的 instance cell，直接將此 instance cell 移動至下一位置。此為 branch-and-bound 的部分。要注意的是，計算 HPWL 十，尚未擺放上去的 instance cell 並不會被列入考量。

(6) 若是此 segment 中的所有 instance cells 皆完成擺放，且使得 design 的整體 HPWL 小於目前最佳解，則記下 instance cells 之座標，並記錄當前 HPWL 為目前之最佳解。

4. Terminal Placement

4.1 Terminal Insertion

當一條 net 的 pin 點跨越上下兩個 die 時，就需要插入一個 terminal 來連接上下兩條 net，我們先令 net 在下層 die 上 pin 點的 bounding box 為 b0，在上層 die 上 pin 點的 bounding box 為 b1，terminal 插入的初始位置可以分成三種情況來看：

(1) 若 b0 和 b1 的 x、y 軸都沒有重疊，則 terminal 插入在 b0 和 b1 兩矩形中最接近的兩個點所形成區域(如圖 7 紅色區域)中的任一位置，如圖 5。

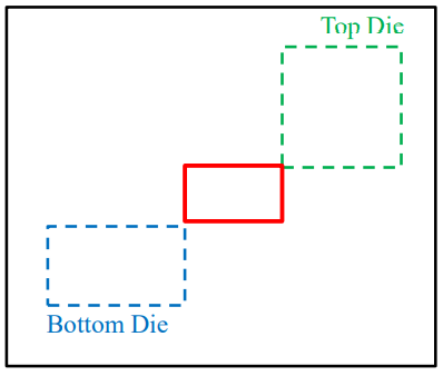


圖 5。

(2) 若 b0 和 b1 的 x、y 軸有其中一軸重疊，則 terminal 插入在 b0 和 b1 間 channel 通道中的任一位置，如圖 6。

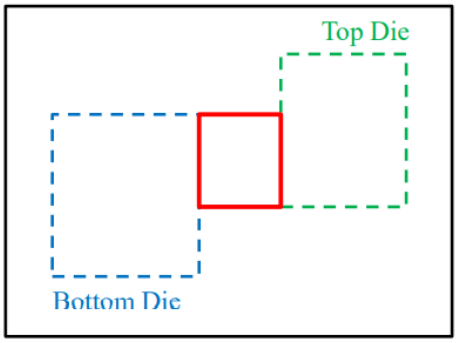


圖 6。

(3) 若 b0 和 b1 的 x、y 軸皆重疊，則 terminal 插入在 b0 和 b1 重疊區域中的任一位置，如圖 7。

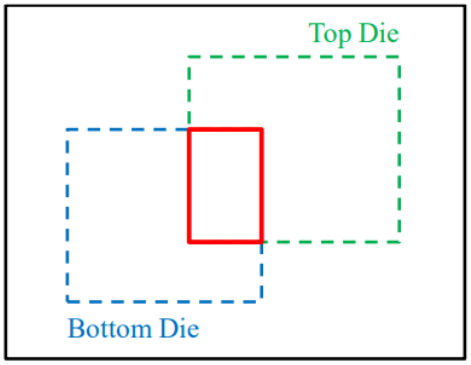


圖 7。

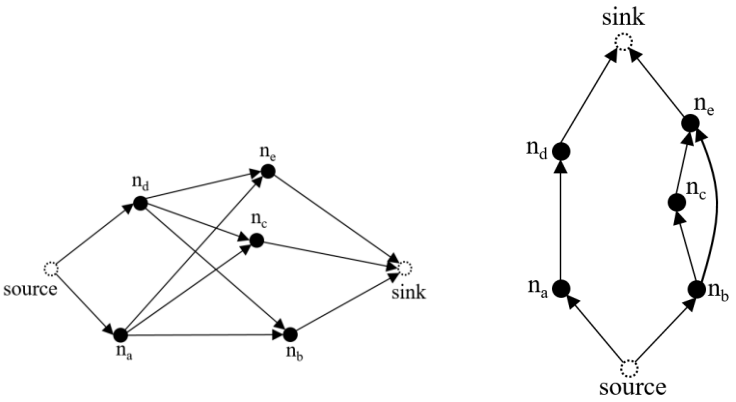
按照以上三個情況決定 terminal 的初始位置可以保證 local HPWL 的最小解。

4.2 Terminal Legalization

決定好每一個 terminal 的初始位置後，terminal 之間可能會重疊，因此這一步要來消除重疊，我們利用 TCG（Transitive Closure Graph）和線性規劃（Linear Programming）技術，讓 terminal 間彼此不重疊，並且最小化 terminal 的總位移量。之所以要先建立 TCG 是因為單純用不重疊限制（non-overlapping constraint）來解線性問題的話非常耗時，所以我們先利用 TCG 將 terminal 之間的相對位置關係限制住，達到縮小可行解區域（solution space）的目的，有效降低執行時間。

(1) TCG （Transitive Closure Graph）

將所有 terminal 的相對位置關係置轉換成水平和垂直的 TCG，分別表示成 G^H 和 G^V，如圖 8，TCG 的頂點 n_i 代表 terminal，邊的權重 e_{ij} 代表邊的兩個端點不重疊至少要有的距離長度（non-overlapping distance）（要注意的是，這個距離必需額外加上 terminal spacing constraint，以符合最小通道距離限制），所以此 TCG 同時也是 CG（Constraint Graph），每一條邊都對應一組 non-overlapping constraint。



（a） （b）

圖 8（a）G^H 。（b）G^V。

(2) LP （Linear Programming）

將 TCG 的邊轉成 ILP 問題的約束條件（Constraint）（3），以最小位移（Displacement）總和作為 LP 問題的目標函數（Objective function）（4），有了約束條件和目標函數後，便可將它們代入 LP solver (使用 lp_solve 5.5)來決定 terminal 的實際座標位置。

$$\min \quad \sum_{i=1}^{n} d_{x_i} + d_{y_i} \qquad (3)$$

$$\text{s.t.} \quad x_j' - x_i' \geq e_{ij} \quad if \ \exists e_{ij} \in G^H$$

$$y_j' - y_i' \geq e_{ij} \quad if \ \exists e_{ij} \in G^V \qquad (4)$$

最後即可得到彼此沒有重疊的 terminal 擺放結果。

5. Experiment results

testcase	TopDie Area(used/max)	BottomDie Area(used/max)	Total HPWL	Run time
case1	620/900 Util: 68.89 MaxUtil: 80	630/900 Util: 70 MaxUtil: 90	123	0.215s
case2	52838368/82936425 Util: 63.71 MaxUtil: 70	62201916/82936425 Util: 75 MaxUtil: 95	10085526	39.014s
case3	283122985/369254080 Util: 76.67 MaxUtil: 78	283111715/369254080 Util: 76.67 MaxUtil: 78	285721297	14m29s
case4	1819446556/2838171970 Util: 64.11 MaxUtil: 66	1986719990/2838171970 Util: 70 MaxUtil: 70	2451499704	52m46s

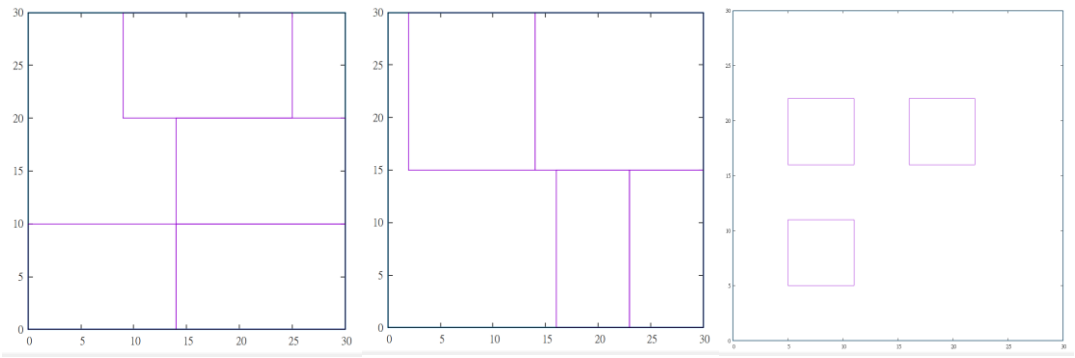


圖 8 case1 (a)TopDie cell (b)BottomDie cell (c)Terminal

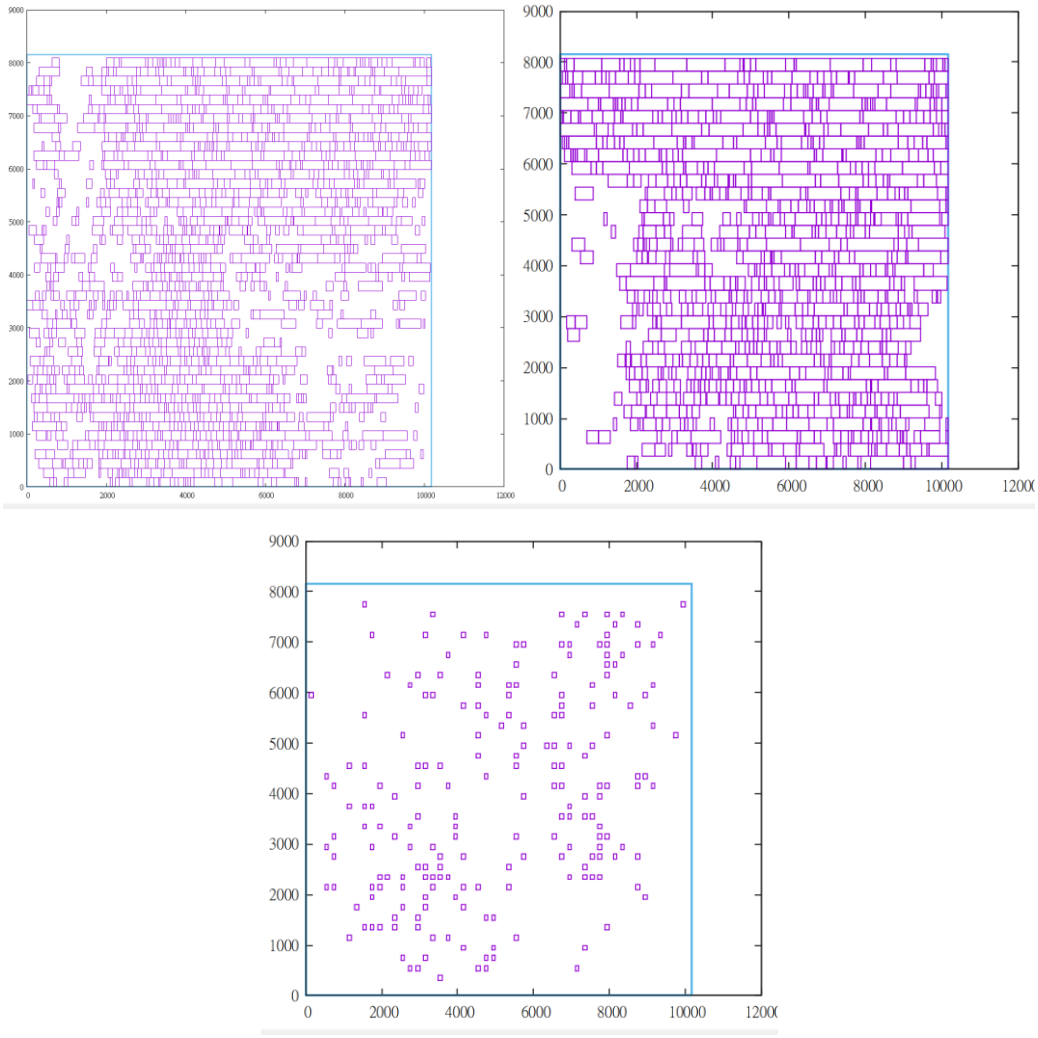
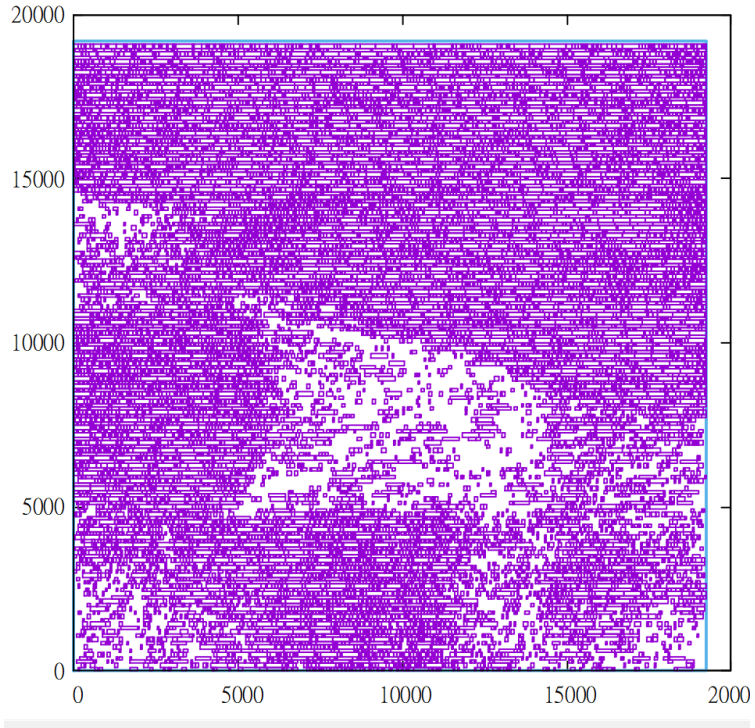


圖 9 case2 (a)TopDie cell (b)BottomDie cell (c)Terminal



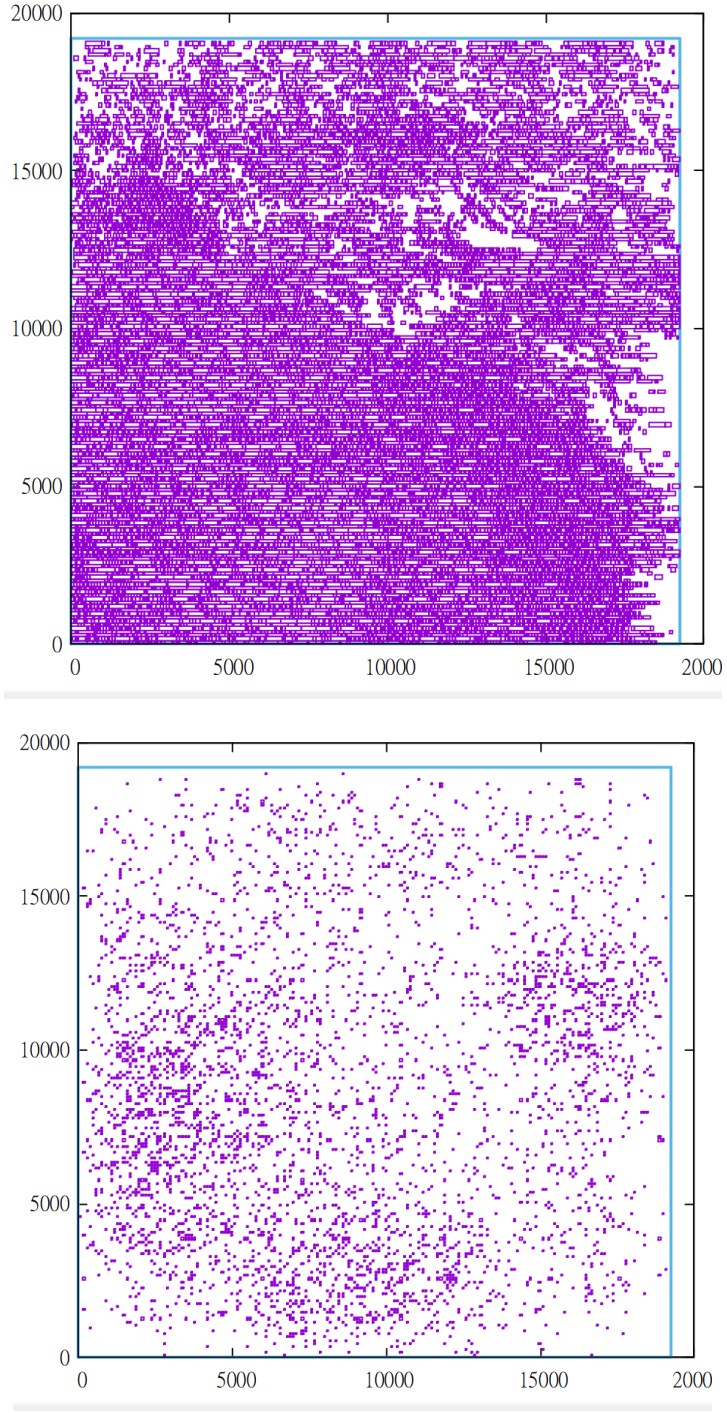
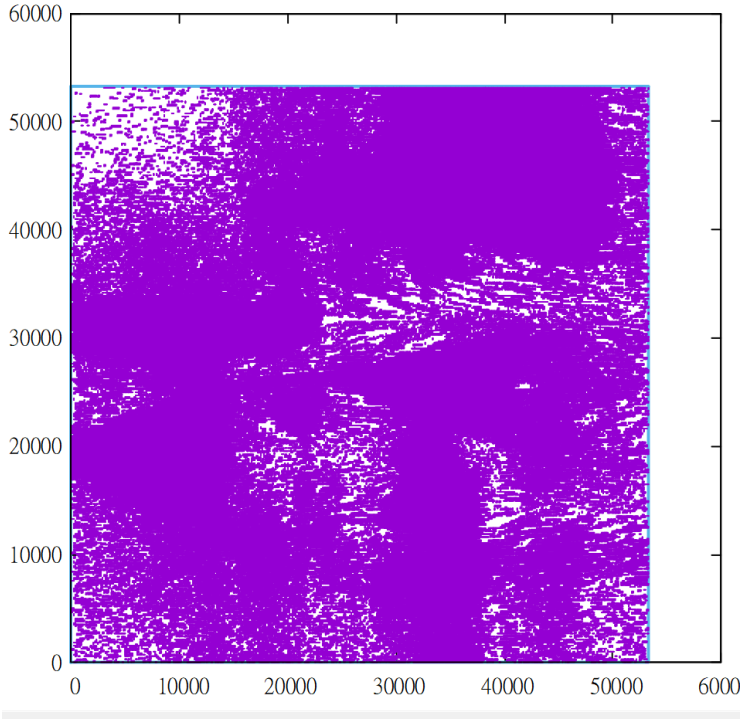
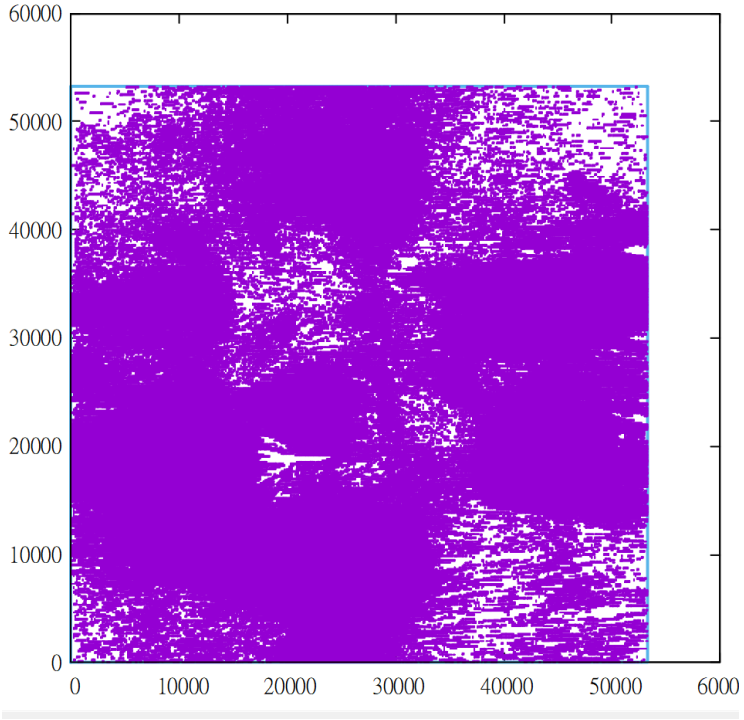


圖 10 case3 (a)TopDie cell (b)BottomDie cell (c)Terminal


60000
50000
40000
30000
20000
10000
0
0
10000
20000
30000
40000
50000
6000


60000
50000
40000
30000
20000
10000
0
0
10000
20000
30000
40000
50000
6000

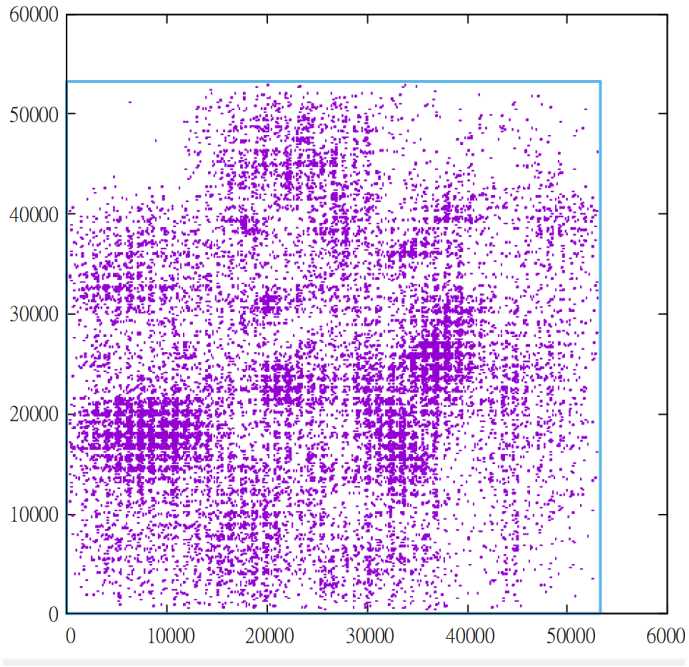


圖 11 case4 (a)TopDie cell (b)BottomDie cell (c)Terminal

6. Reference

5.1 Global placement

(1) NTUplace3: An Analytical Placer for Large-Scale Mixed-Size Designs With Preplaced Blocks and Density Constraints Tung-Chieh Chen, Student Member, IEEE, Zhe-Wei Jiang, Student Member, IEEE, Tien-Chang Hsu, Hsin-Chen Chen, Student Member, IEEE, and Yao-Wen Chang, Member, IEEE

5.2 Cell legalization

(1) source code of NTUplace3

5.3 Detailed placement

(1) http://cc.ee.ntu.edu.tw/~ywchang/Courses/PD_Source/EDA_placement.

(2) http://ntur.lib.ntu.edu.tw/bitstream/246246/141412/1/40.pdf

(3) source code of NTUplace3

5.4 Terminal placement

(1) Michel Berkelaar, Kjell Eikland, and Peter Notebaert. "lp_solve." http://lpsolve.sourceforge.net/5.5/ (accessed July 20, 2021.

(2) J.-M. Lin and Y.-W. Chang, "TCG: A transitive closure graph-based representation for non-slicing floorplans," in *Proceedings of the 38th annual Design Automation Conference*, 2001, pp. 764-769.