**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

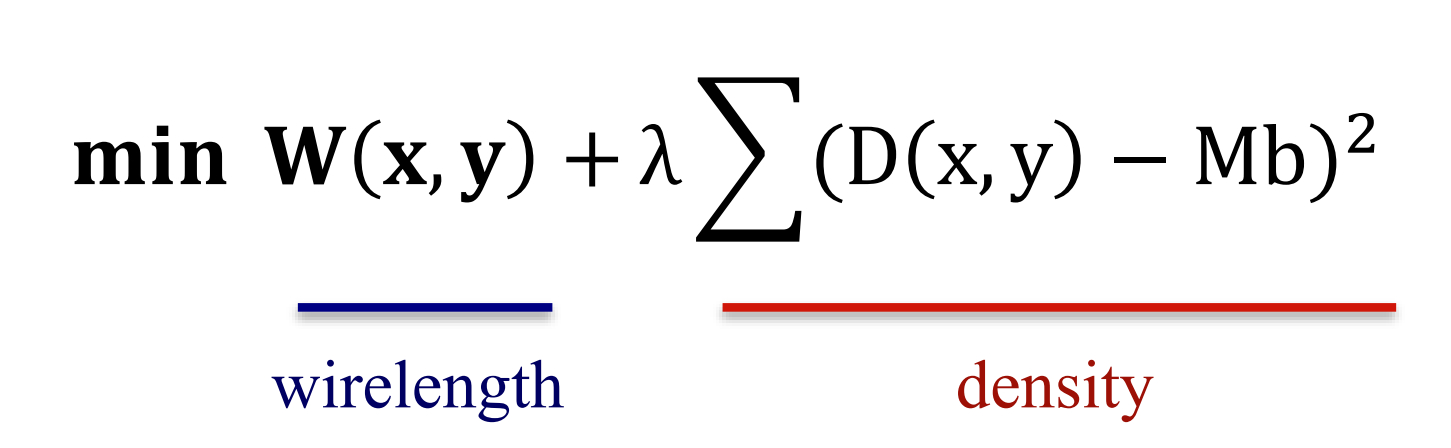
台灣科技大學 資訊工程系 B10732014 何雨澂

台灣科技大學 資訊工程系 B10732040 李宇哲

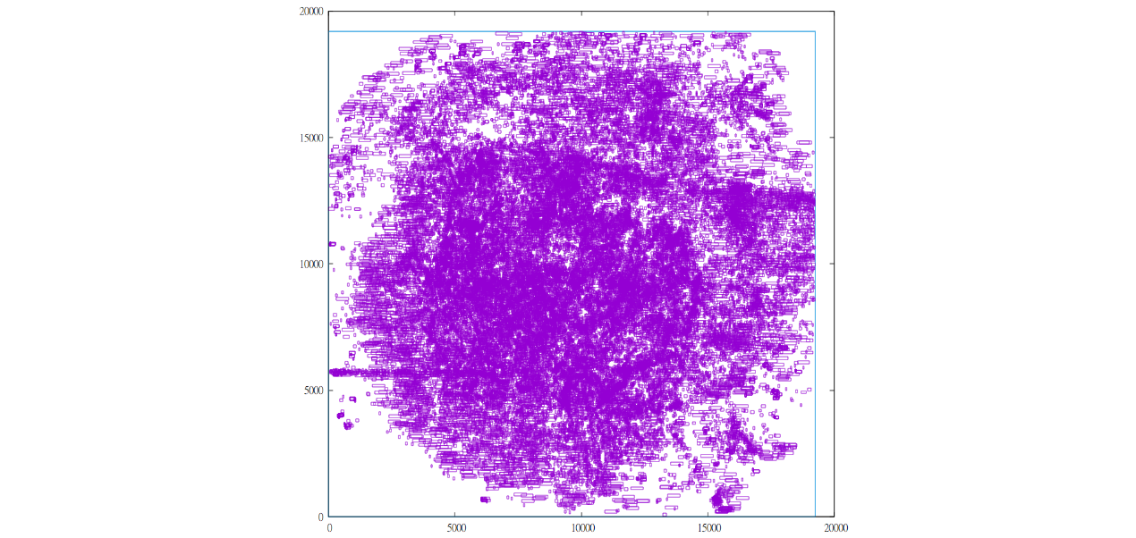
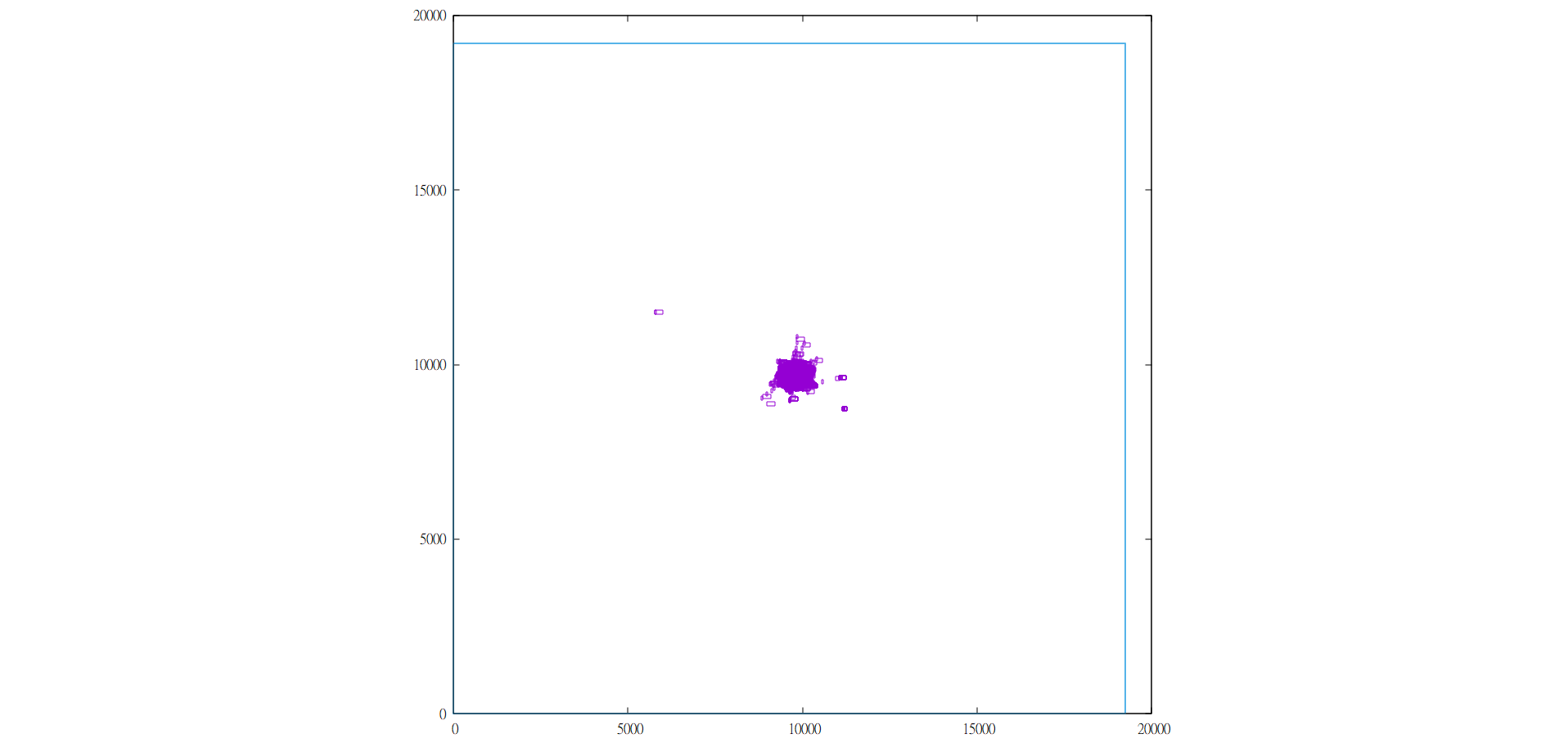
**Table of Contents**

1. Global Placement ……………………………………………………………………..1
2. Cell Legalization ………………………………………………………………………..4
3. Detailed Placement …………………………………………………………………..5
4. Terminal Placement ……………………………………………………………..……8
5. Experiment Results …………………………………..……………………………...10
6. Reference ……………………………………………………………………………..…..14
7. **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:

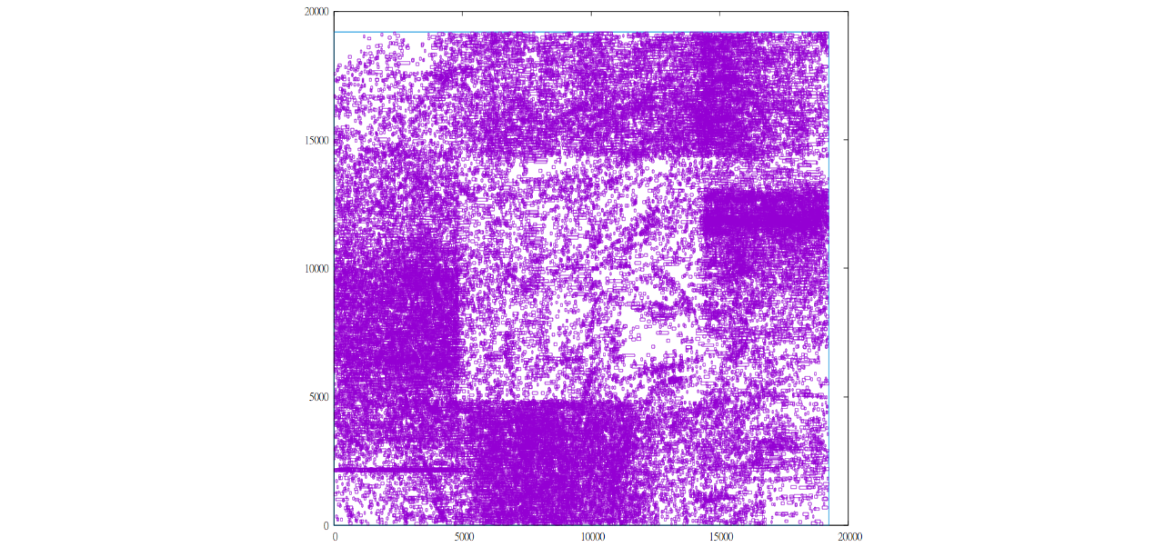
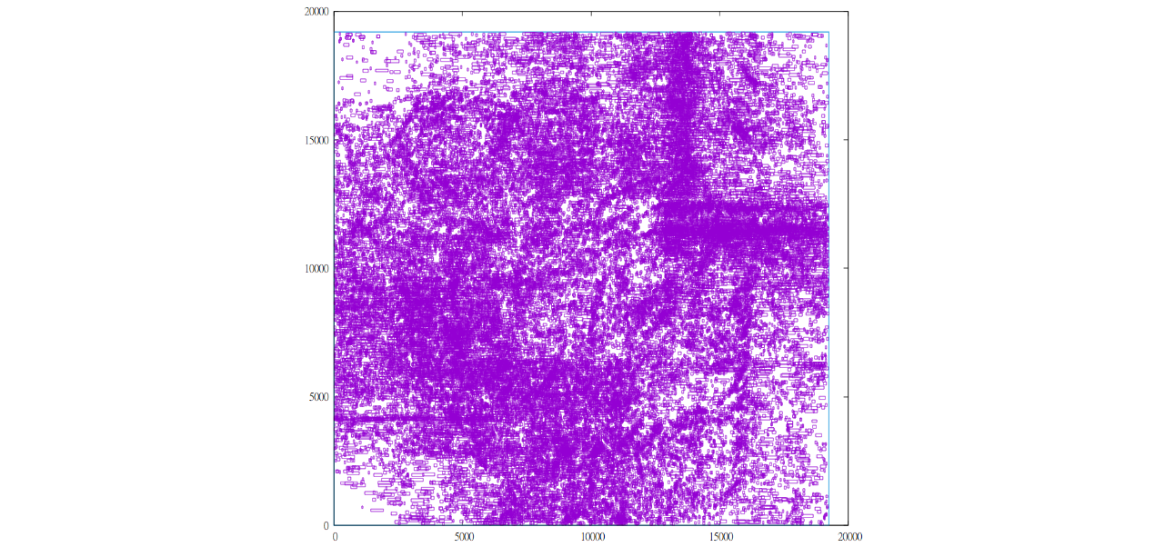


在objective function中wirelength的部分用HPWL來計算，而density部分則是對每個bin來計算密度，而λ係數調整density在objective function所佔的比重。為了讓objective function可微分，wirelength的部分會使用LSE model來估算，density部分則會採用Bell-shaped model來估算。剛開始，λ係數會設為0，讓design以HPWL為主來做placement。接著調整λ係數，讓wirelength和density在objective function中所貢獻的比重相同，之後λ係數設為前一輪的兩倍，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



(b) Round 2

1. Round 1



(d) Round 4

(c) Round 3

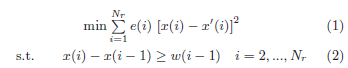
圖1。

1. **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。

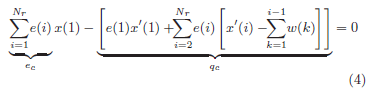
之後開始進行cell legalization，我們使用abacus的做法來實作。將cell放到row上並尋找最佳x做標的行為可用底下quadratic program來描述:



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:

https://lh3.googleusercontent.com/GYtLB223z1CKhKTV4hPj83IHduim5T8gB3_DZqCxQy9VelWmvz-wqMl5NlhuZS_F5kONxgYg1e2y35qFYL1raar4LwfOQe4CAjQMUzHXIOYP-TUeIHSScwCoXA7VjqcboTr66lqDMaOX_1jbUA

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):



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:

https://lh6.googleusercontent.com/jJAdtFmzBmnK4gZRmQeAj8ongpCHKRO23qVCYuRxGRc24xf-LhgUXW2BTnQyNkATpC39K2bH7RdmZyTPgXJRKLNQK6Rh7cfP4IqDWzRe5cX0_VuxdyTkw7V_otjjOqMu5cQKzc2w8bBOHx1oIQ

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。

1. **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。

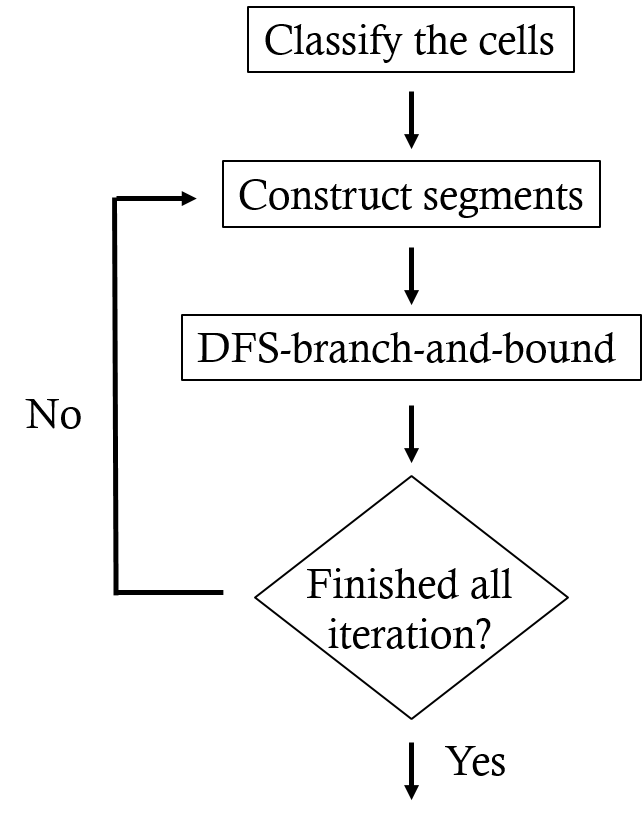


圖2。Detailed placement的流程圖

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。

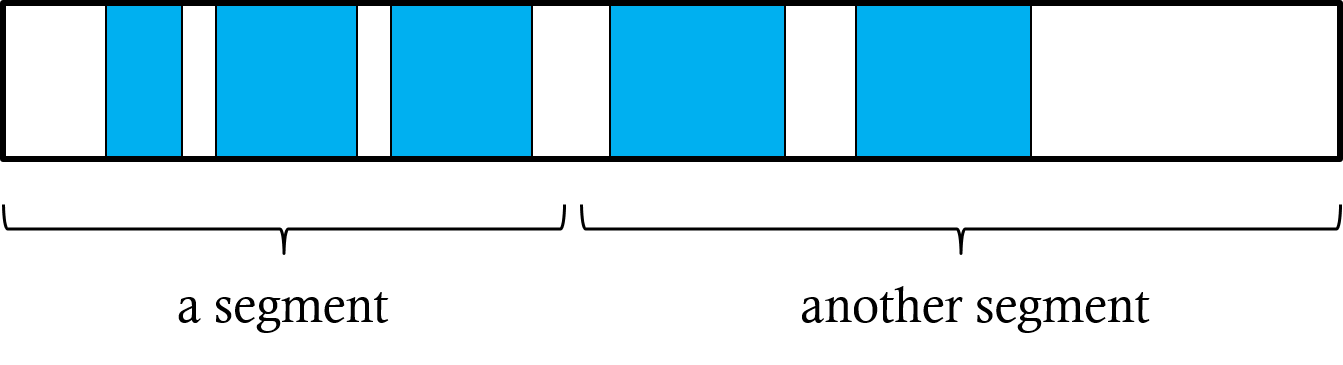


圖3。建構segment的一個例子。

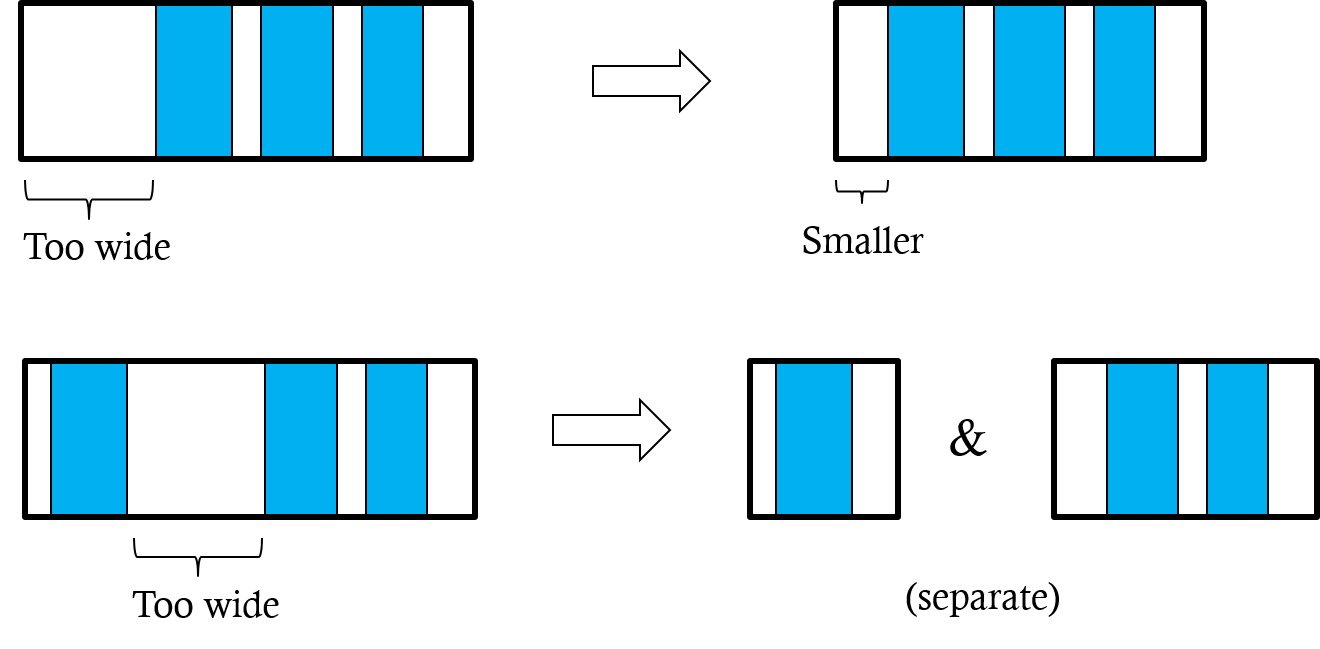


圖4。調整segment boundary的例子。

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。

* 1. 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。

1. **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。

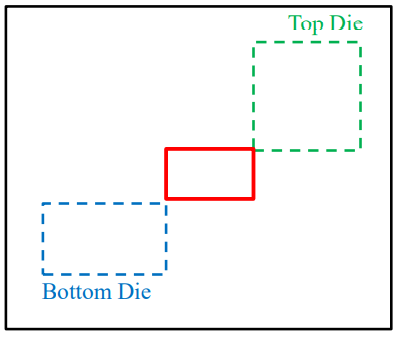


圖5。

1. 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。

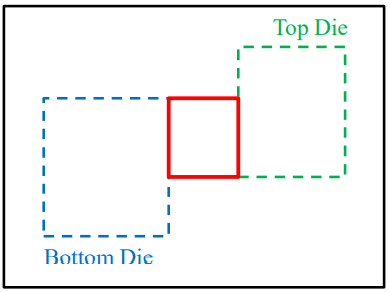


圖6。

1. 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。

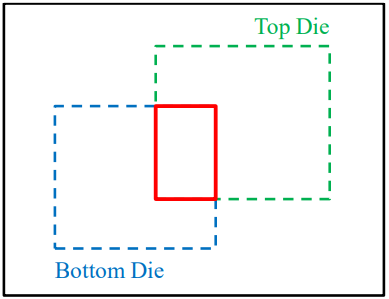


圖7。

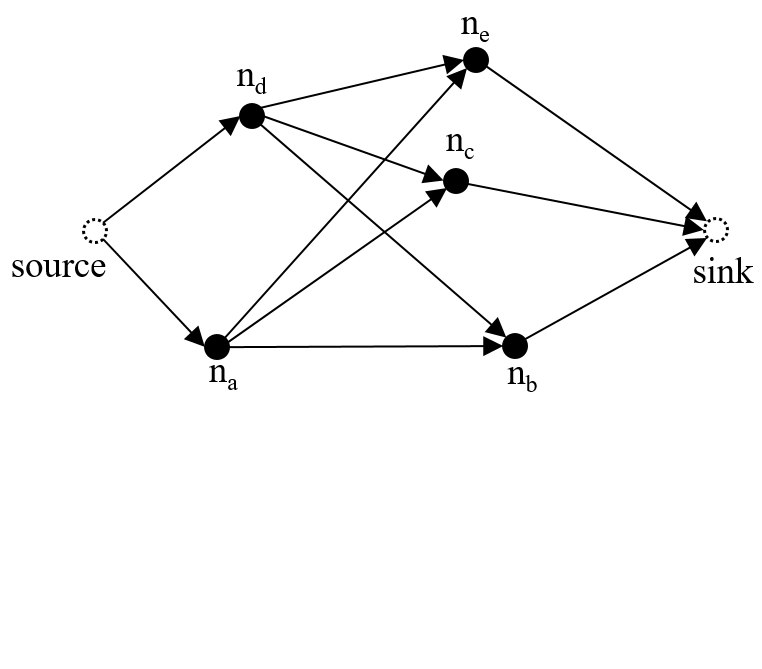
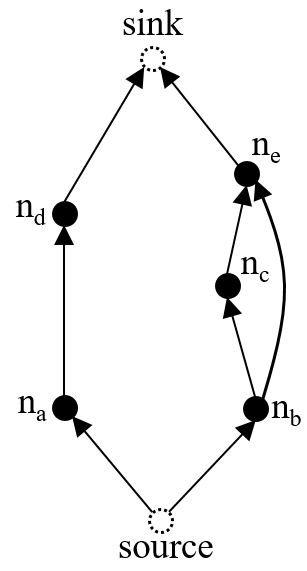
按照以上三個情況決定terminal的初始位置可以保證local HPWL的最小解。

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*GH*和*GV*，如圖8，TCG的頂點*ni*代表terminal，邊的權重*eij*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）*GH* 。（b）*GV*。

1. 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。

（3）

（4）

最後即可得到彼此沒有重疊的terminal擺放結果。

1. **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 |

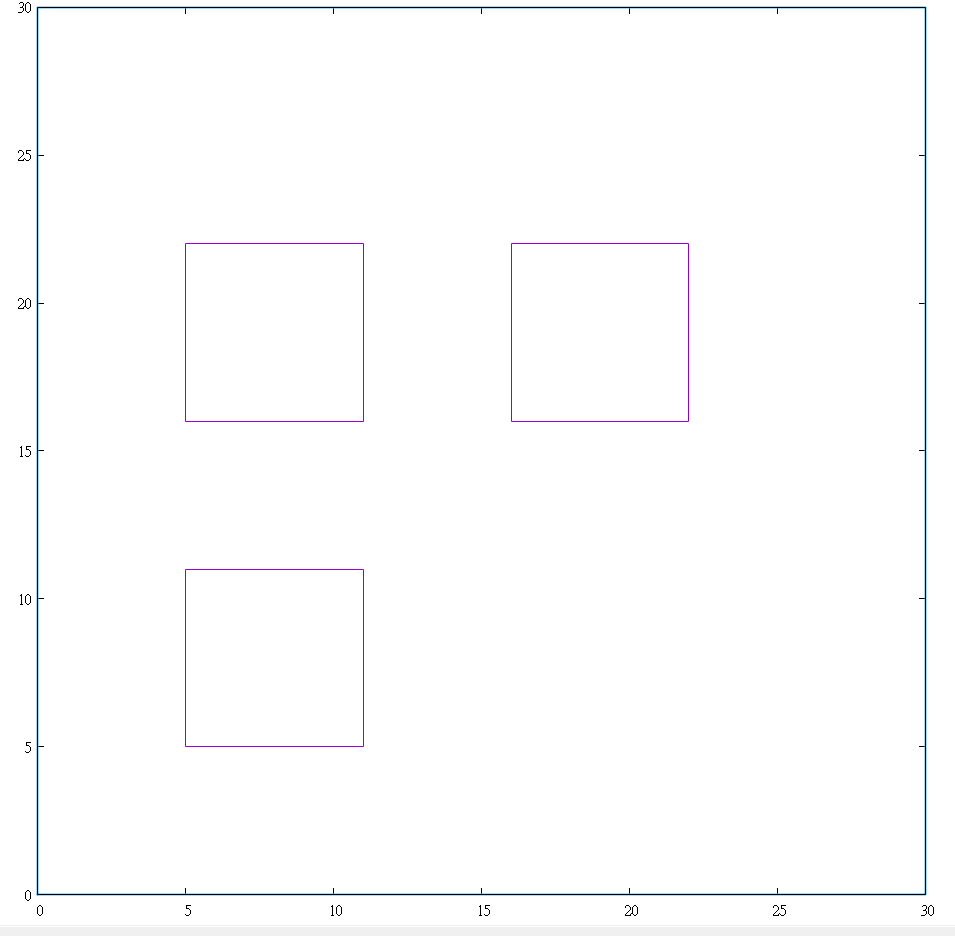
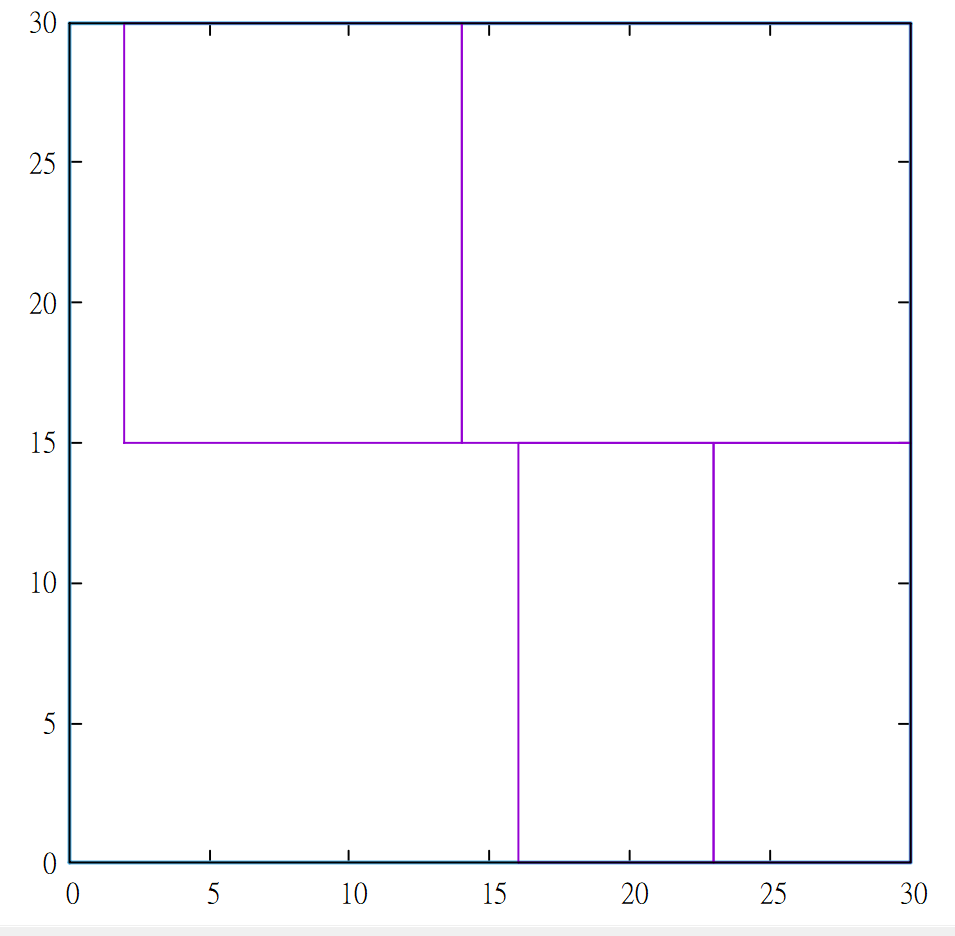
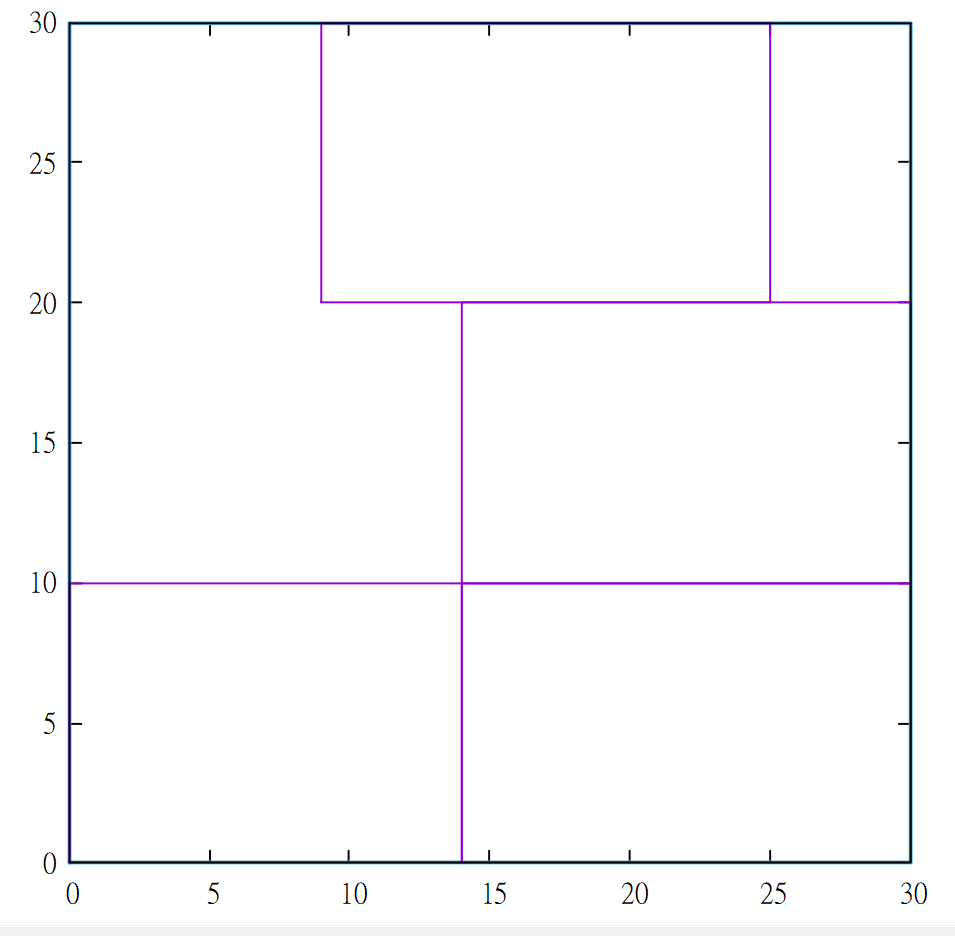


圖8 case1 (a)TopDie cell (b)BottomDie cell (c)Terminal

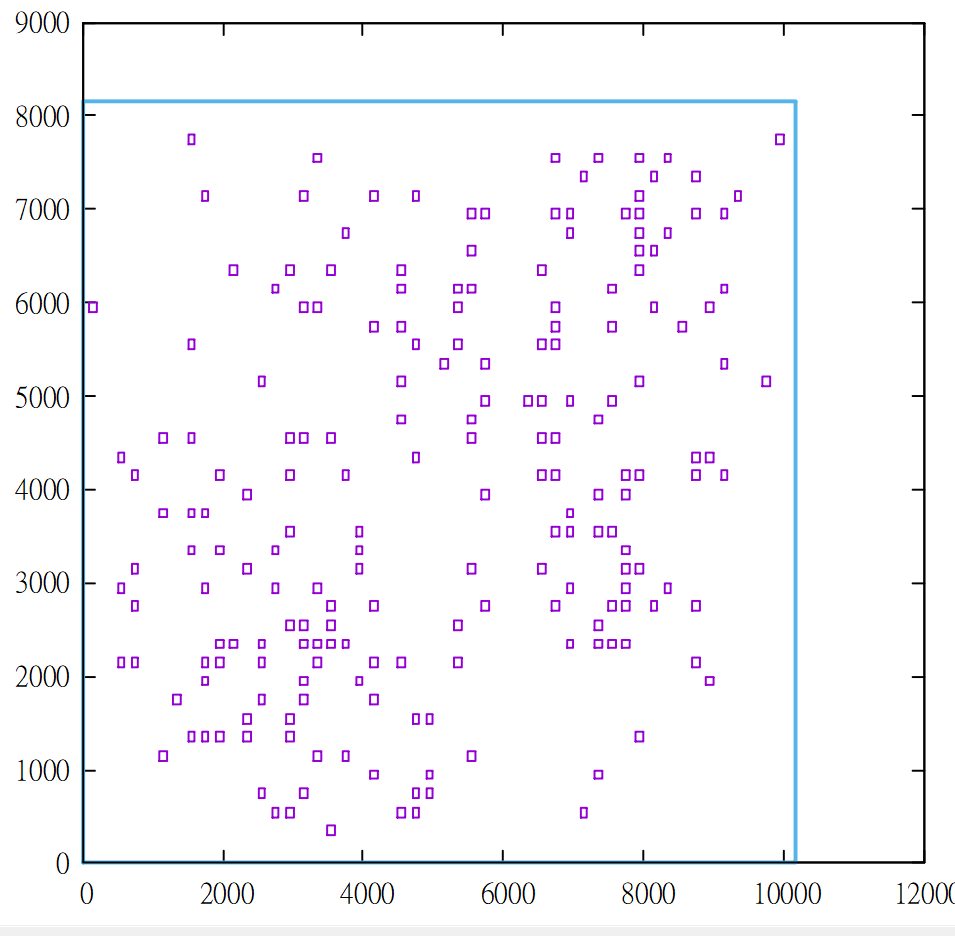
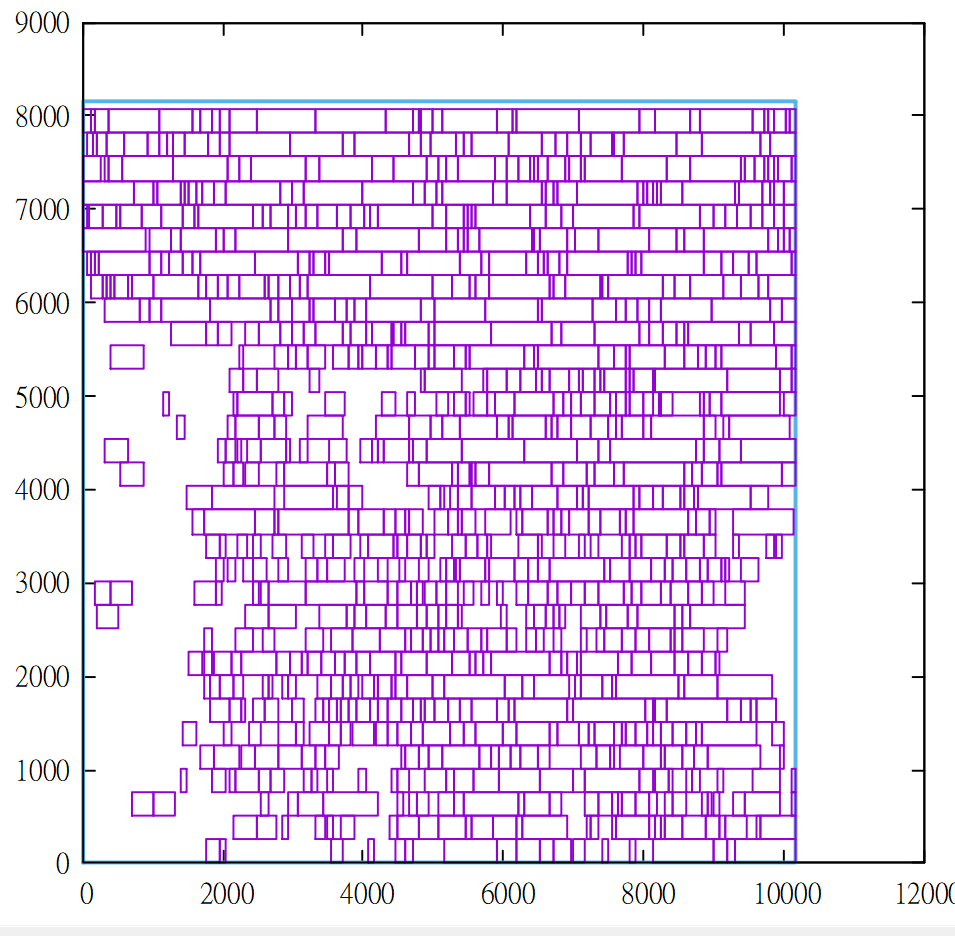
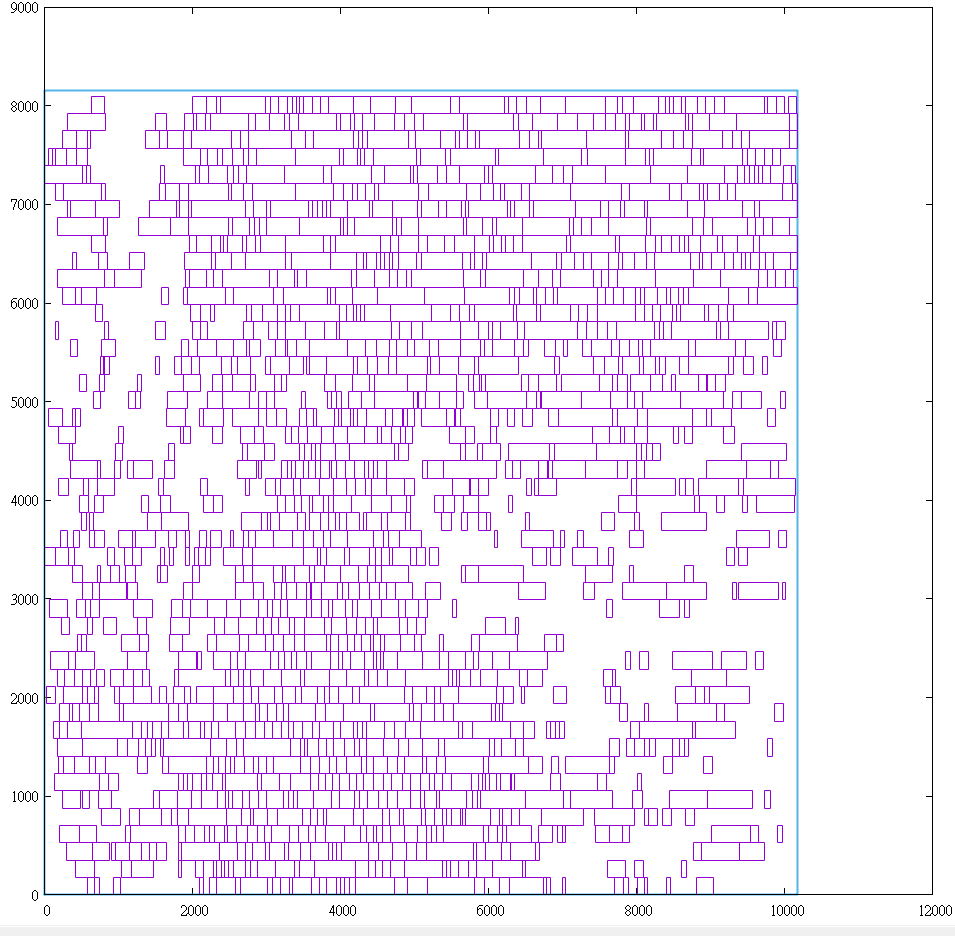
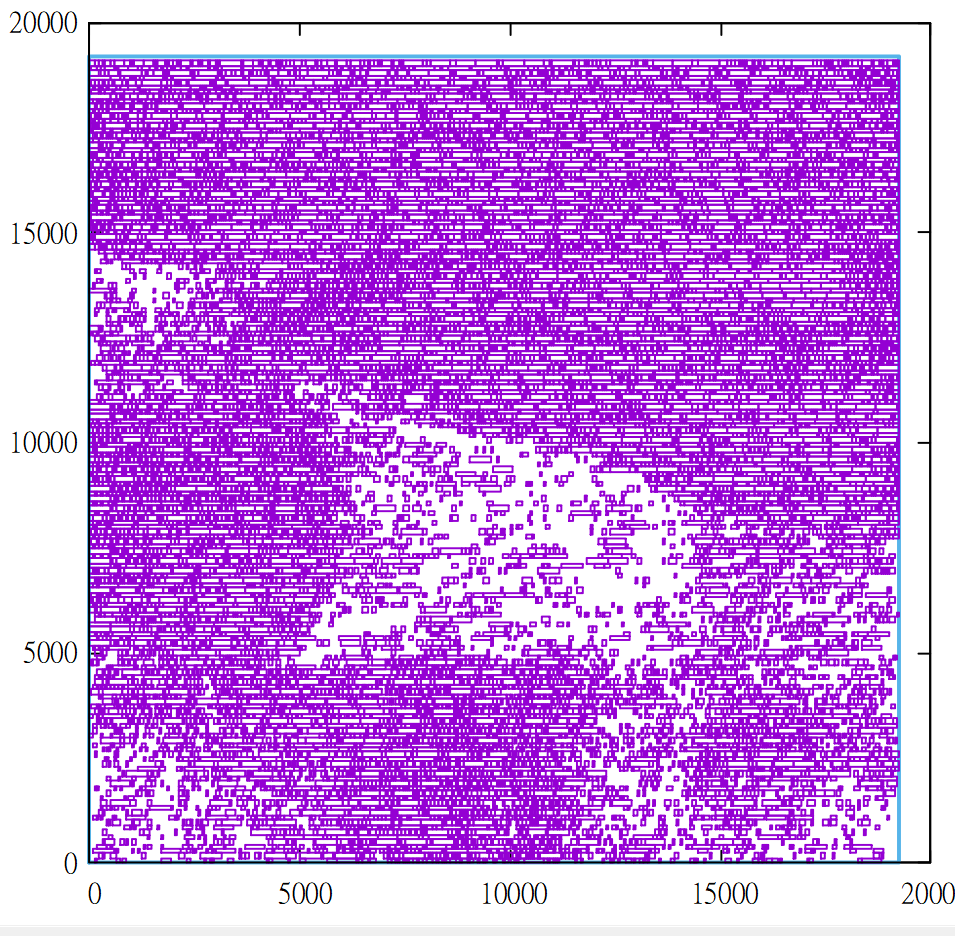
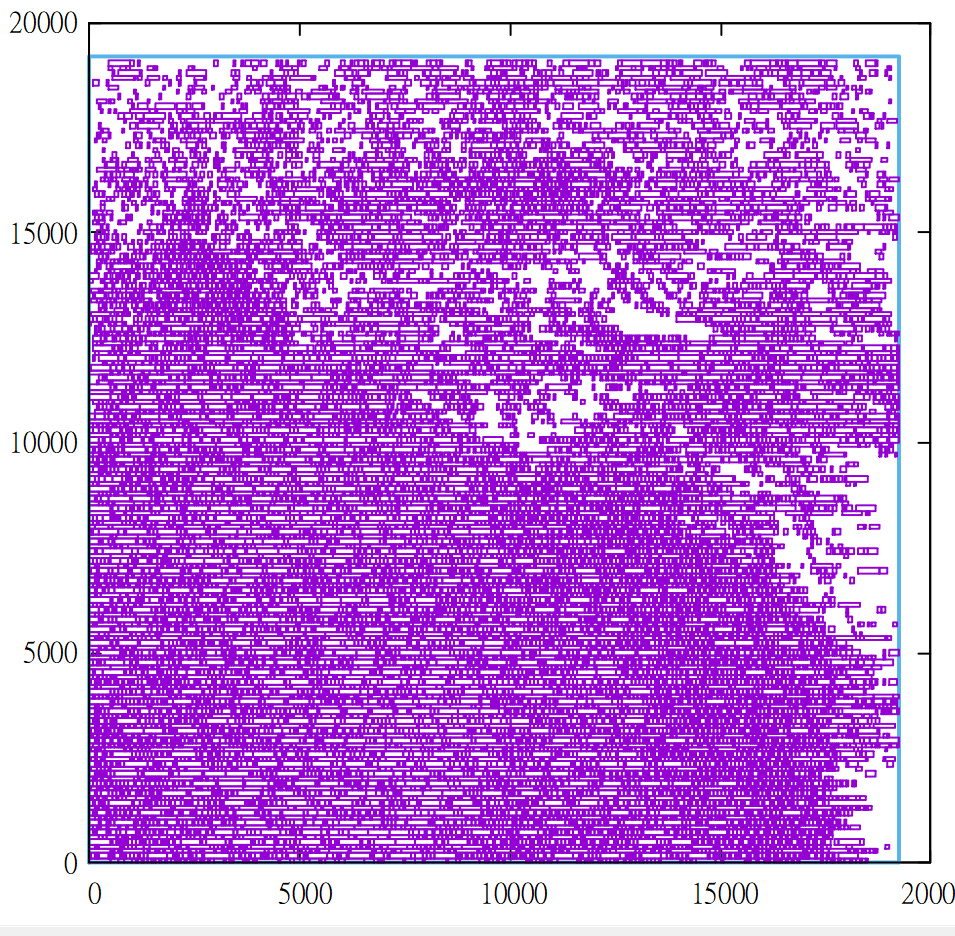


圖9 case2 (a)TopDie cell (b)BottomDie cell (c)Terminal

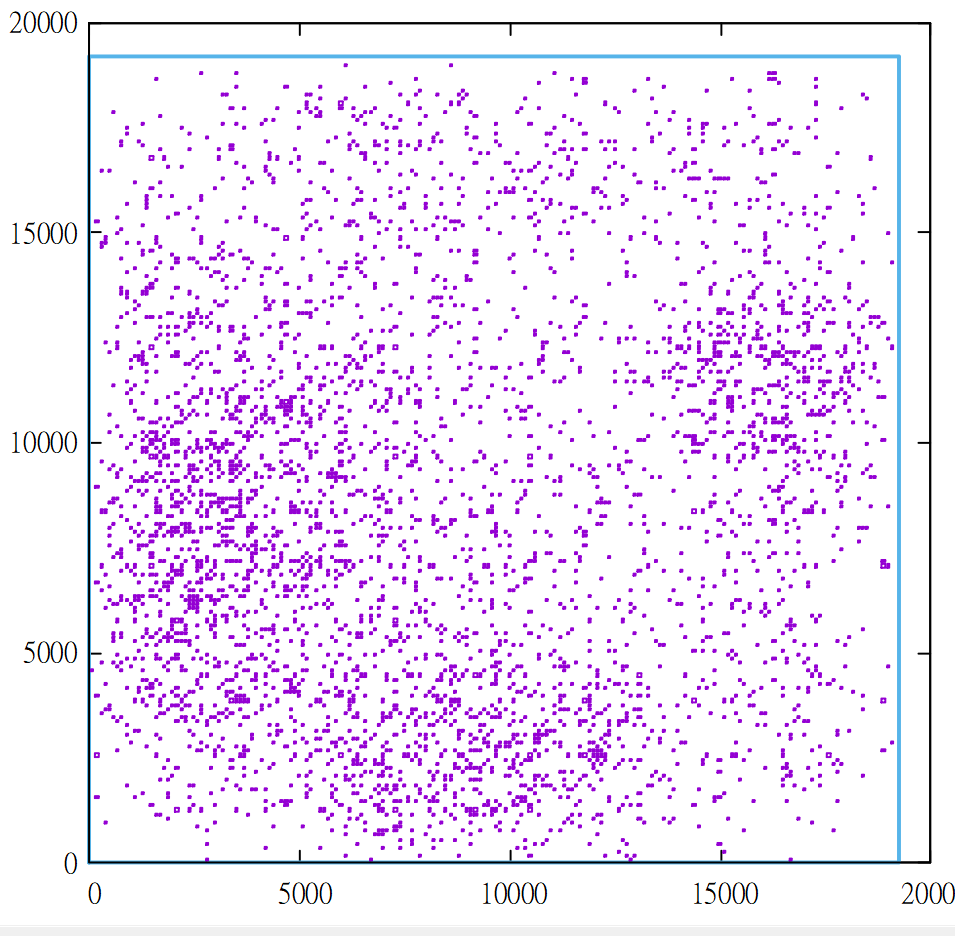
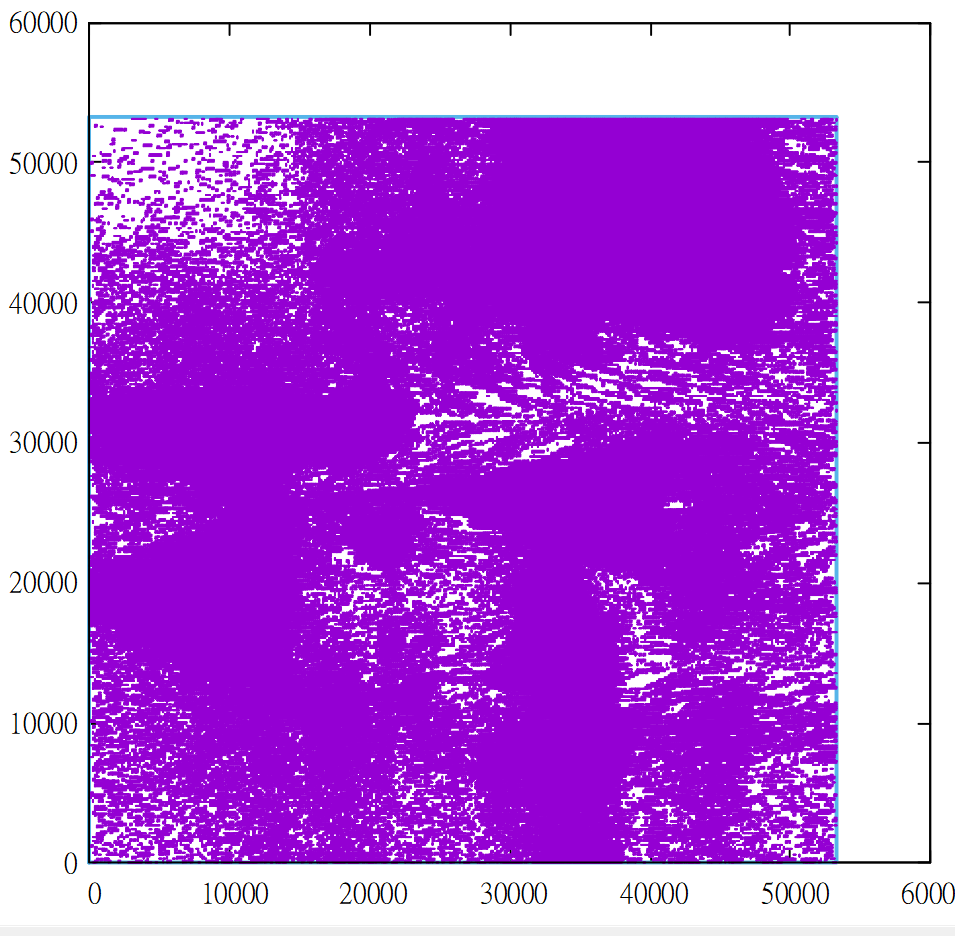


圖10 case3 (a)TopDie cell (b)BottomDie cell (c)Terminal



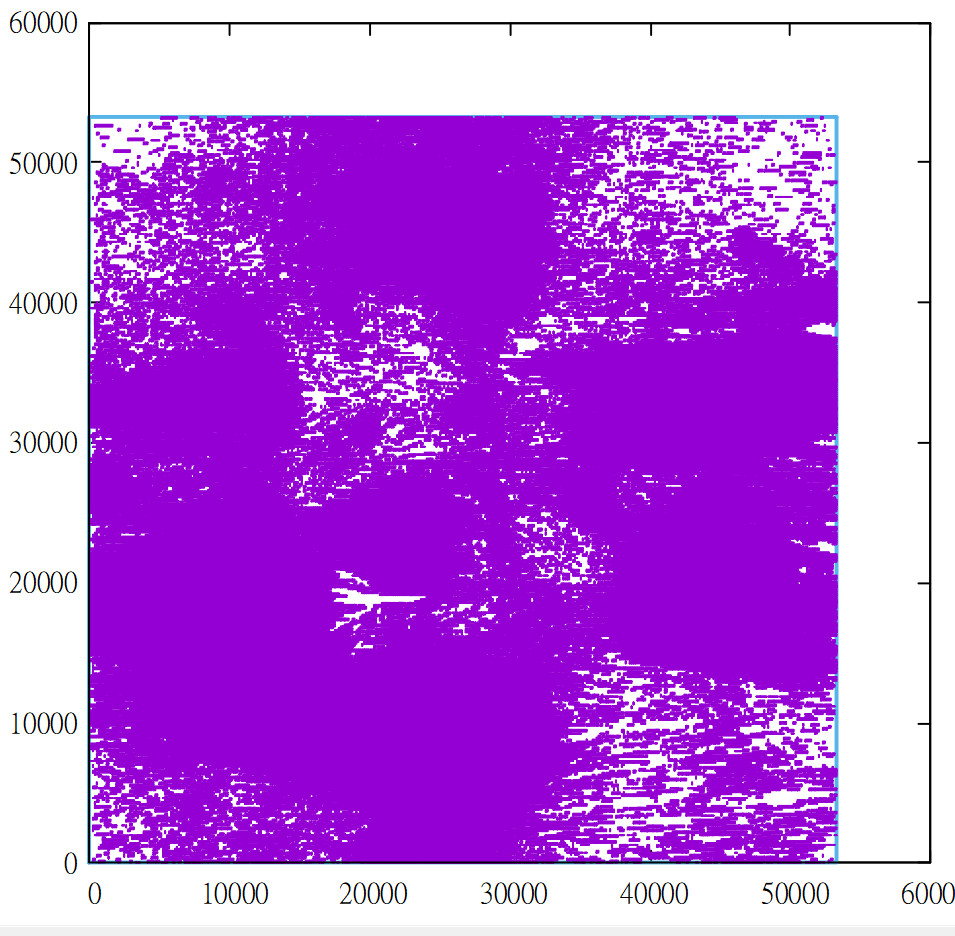
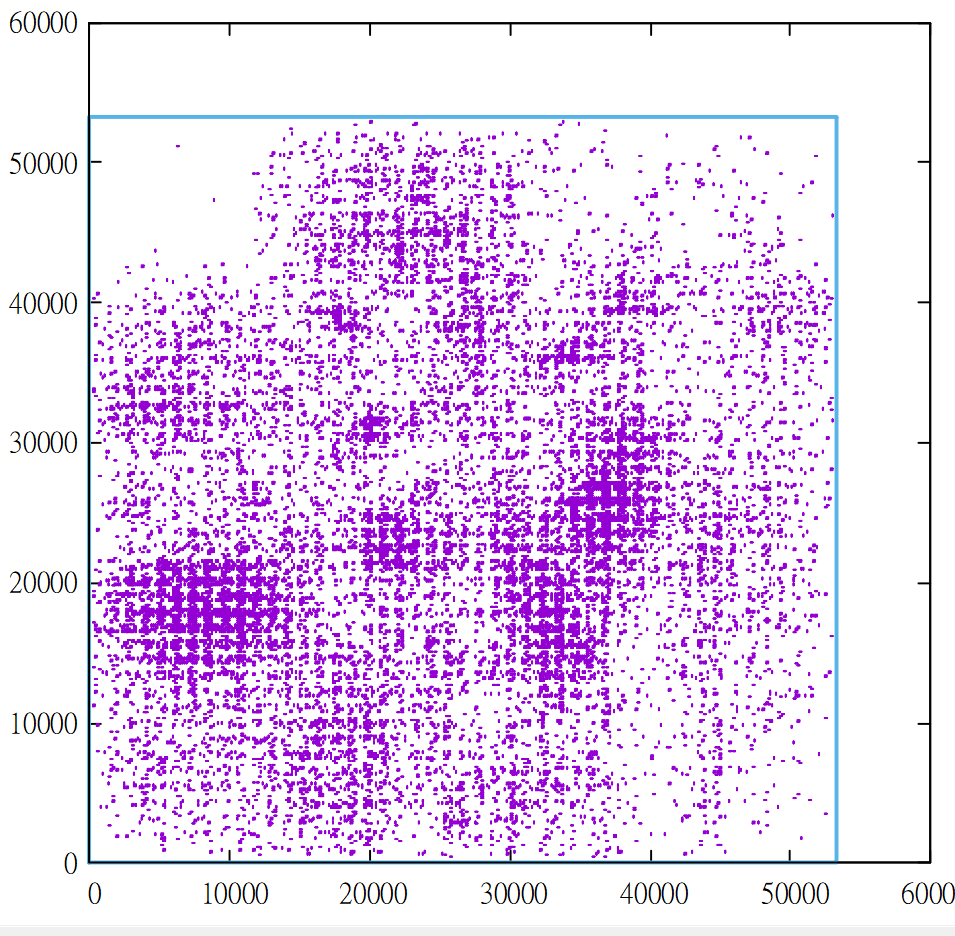
 

圖11 case4 (a)TopDie cell (b)BottomDie cell (c)Terminal

1. **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.