專題研究報告

3D Placement with Macros

團隊： 國立清華大學資訊工程學系 余伽璇、林晨、陳美晴
指導教授： 王廷基教授

摘要

本團隊的專題研究以參與 ICCAD 2023 CAD Contest 為主，我們選擇的題目為 Problem B : 3D Placement with Macros，旨在根據給定兩個 die 的 technology，以及面積、種類、位置、距離等諸限制，將 standard cells 跟 macros 合法分配並擺置在兩個 die 上，也包括對橫跨兩個 die 的 net 擺置 hybrid bonding terminal。目的是找出具最短 Half-Perimeter Wirelength（HPWL）的合法擺置結果 [1]。

剛開始我們的表現並不如預期，不僅耗時甚鉅也不符限制。然而經過反覆的討論與實驗後，我們藉由改變 partition 後的 legalize 算法、優化 hybrid bonding terminal 的位置，及找到妥善擺置 macros 的方法，得到顯著的進步成果，也成功通過了所有公開測資。比賽最終得獎名單還未公布，我們的最終成績在官方釋出的 Beta 版本測試結果中位列第六名，我們與其它多由碩博生組成的隊伍相比也有出色的表現。期盼未來我們能根據後述的討論，再深入改善目前的策略，以得到更好的結果。

關鍵詞：3D Placement、EDA

一、 研究背景

在目前小晶片（chiplet）的時代，設計趨勢走向將一個大的 die 分割成兩個或以上較小的 die，再將這些小的 die 垂直互連，即為 3D 晶片。運用這項技術，可以得到更好的良率（yield）、更佳的性能（距離縮短，時序加快）、更低的成本（每個 die 可依不同適應類型與需求來以不同 technology 製造），以及迎來更快的上市時間。我們此次參與的 ICCAD 2023 CAD Contest 的 B 題便是在解決 3D Placement 的問題，與來自世界各地的大學生、碩博生切磋最佳解。

二、 研究目的

此問題旨在找到兩個面對面垂直連接的 die 中， standard cells 跟 macros 的合法擺置方法。給定資訊包括上下兩個 die 的面積、最大使用率限制、 row 長寬、不同 technology 的 standard cells 或 macros 資訊（長、寬、 pin 位置）、 instances 所屬 standard cells 或 macros 、 pin 位置、 netlist ，以及 hybrid bonding terminal 與邊界和與 hybrid bonding terminal 間的最小距離限制。我們須根據指定條件將 instances 在兩個 die 上擺置，並在符合距離限制的狀況下對橫跨兩個 die 的 net 進行 hybrid bonding terminal placement 。最終目的是在產出合法擺

置結果的同時獲得最小的 Half-Perimeter Wirelength (HPWL) ，也考慮 hybrid bonding terminal cost 跟 runtime cost。

三、 研究方法

圖 1 是我們的演算法整體流程。

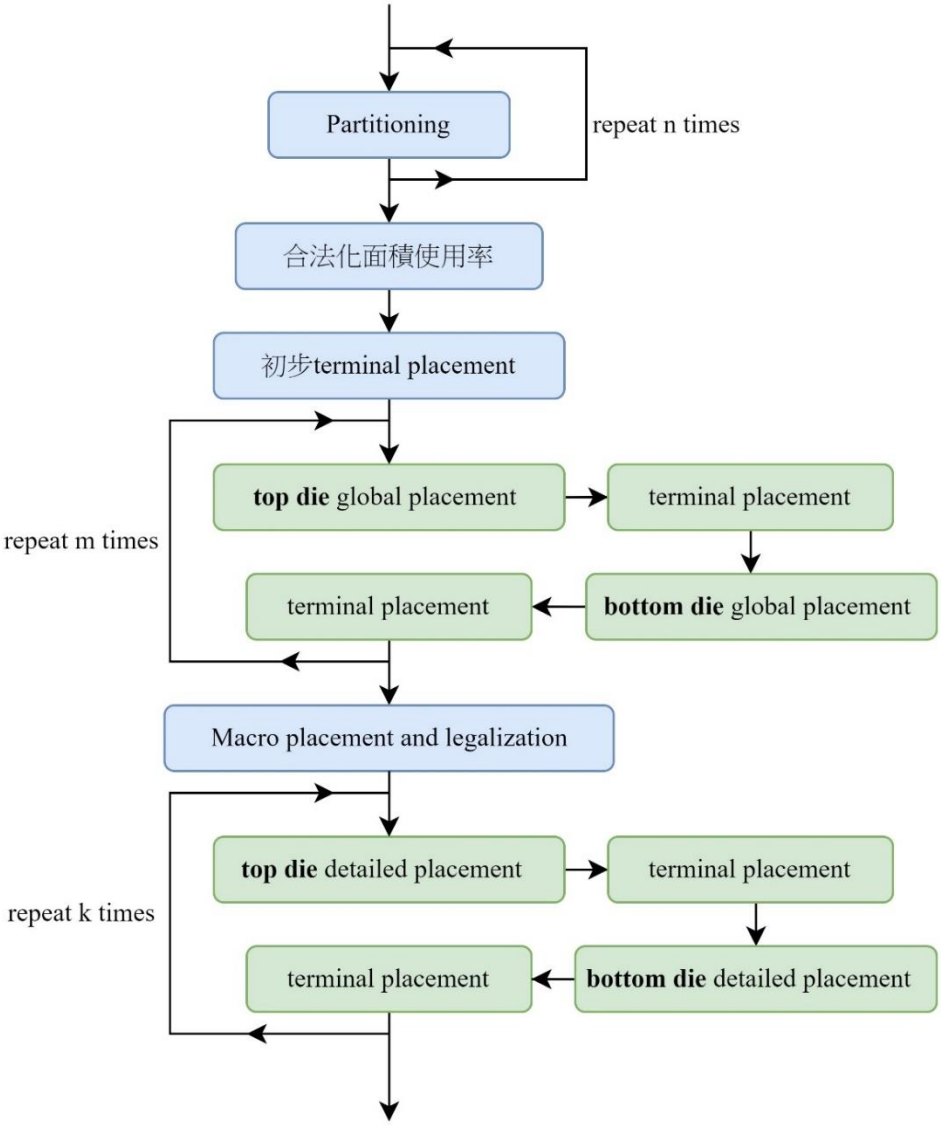

圖 1. 整體流程

1. 使用 hMETIS 初步 partition

a. 我們選擇開源的軟體套件 hMETIS [2] 中的 hmetis 程式，hmetis 是 multilevel hypergraph partitioning 演算法，目標是得到 cut size 最小的 partition。而應用在此題目的初步分割中，hmetis 將 partitioning 分為三個步驟： coarsening phase 、 initial partitioning phase 、 uncoarsening and refinement phase。在 coarsening phase 裡做

multilevel clustering (分群)，讓最後的 vertex 數量減少到 100 以下，在 initial partitioning phase 對很少量的 vertices 做 partition 就可以很快完成；相較之下，Fiduccia-Mattheyses (FM) [3] 只適用於 vertex 數量少的 partition，在 vertex 數量很大時，會非常耗時。因此我們選擇使用 hmetis 來大幅增加 partition 的速度。

b. 為了讓 hmetis 產生的結果盡可能符合面積使用率，我們透過實驗調整 hmetis 的 input vertex weight 和參數之一的 UBfactor ，方法如下：令 vertex weight 取值為 instance 在兩個 die 上的面積平均；UBfactor 則由兩個 die 的面積使用率限制和兩個 die 上的所有 instances 的面積所計算而得。又 UBfactor 的定義為：若 UBfactor = x，則一個 die 的 vertex weight 總和界在 (50 - x)% ~ 50% ，另一個 die 的 vertex weight 總和界在 50% ~ (50 + x)% 。 其中計算 UBfactor 所使用公式如下(3.1)和(3.2)式。

$$utilize\ ratio = \frac{Max\ utilization\ ratio_{top\ die}}{Max\ utilization\ ratio_{bottom\ die}} \times \sum_{i \in \forall instance} \frac{area\ on\ bottom\ die_i}{area\ on\ top\ die_i} \div instance\ count \quad (3.1)$$

$$UBfactor = 100 \times \frac{utilize\ ratio - 1}{utilize\ ratio + 1} \quad (3.2)$$

utilize ratio 是預期的兩個 die 使用面積比，因此正比於 top die 對 bottom die 的面積使用率比，反比於所有 instance 在 top die 對 bottom die 的面積比之算術平均。

c. 重複執行 hmetis 數次後，取 cut size 最小的結果。

2. 調整 partition 結果，合法化兩個 die 的面積使用率

a. hmetis 產生的結果無法每次皆滿足面積使用率，原因是 hmetis 無法直接限制兩個 die 的使用面積，也無法對 instance 在不同的 die 上賦值不同的 vertex weight ，來動態調整面積。因此僅使用 UBfactor 仍無法確保兩個 die 的面積使用率都滿足限制。

b. 我們將超過面積限制之 die 上的 instance 移動到另一個 die 上，直到兩個 die 都符合面積使用率限制。移動原則為計算每個 instance 的 FM gain 值和 area_difference_ratio ，並每次優先移動 gain 值最大的 instance；若遇 instances 的 gain 值相同，則優先移動 area_difference_ratio 最大者。而將 area_difference_ratio 納入條件的理由為：經過我們測試發現，當兩個 die 上的使用面積總和愈小，愈能降低之後 placement 的執行時間並優化 placement 執行結果。其中計算 area_difference_ratio 所使用公式如下(3.3)式。

$$area\ difference\ ratio_i = \frac{instance\ area_i\ 在移出的die上}{instance\ area_i\ 在移入的die上} \quad (3.3)$$

c. 為了加快 legalize 的速度，我們設計特定的機率，使得在發生該情況下必須移動 macro ，這是由於一個 macro 的面積必然比一個 standard cell 大，被選擇的話便可以減少移動 instance 的總次數。而選擇 macro 的方式同前述 b 點。然而一個 macro 上的 pin 也必然較 std cell 多，移動 macro 便會對 cut size 產生較大影響，因此設計移動 macro 的機率時，需考量 cut size 結果以取得平衡。最後經過我們不斷測試，決定必須移動 macro 的機率如下表 1。

Instance count	Must move macro probability
<= 300000	0.33
> 300000	0.01

表 1. 移動 macro 機率

d. 若 top die 的實際使用面積大於 top die 的最大可使用面積，與 bottom die 的實際使用面積大於 bottom die 的最大可使用面積的情況同時發生，而導致無法再移動任何 cell 時，則從超出面積使用率的 die 中移出 macro 到另一個 die，接著繼續重複前述 b 點。

e. 每一個 instance 最多都只會被移動一次，以避免重複移動造成超時的狀況。

f. 如 e 點所述，我們已避免執行時間超出題目限制。萬一仍有發生該情況的風險，我們還有最後的解決方案，即紀錄當前已執行時間，並依此判斷是否更改 partition 方法。更換的方法為：重新執行 hmetis 並再次 legalize。與首次相比，我們在 hmetis 的 input 內容中新增一項 fixfile ，目的是在執行 hmetis 前先根據面積比例分配 macros 於兩個 die 上，使其在 hmetis 執行過程中為固定狀態而不受影響，此時僅 standard cells 為可分配對象。接著在超過使用面積限制的 die 中，選擇移動面積變化最小的 standard cell 到另一個 die 。若發生如前述 d 點情況，則移動 macro ，直到其中一個 die 符合面積限制時改回移動 standard cell 。

3. 初步 hybrid bonding terminal placement

a. Placement 時會固定 hybrid bonding terminal 的位置來擺放 standard cells 和 macros，以得到 HPWL 的最佳結果。因此初步 terminal placement 可以有效讓 RePlAce 的 mixed size global placement 結果變更好。

b. 為了確保 terminal 彼此之間的間距都滿足 minimum terminal spacing ，我們把 die 切割成數個可以擺放 terminal 的格子，並使格子之間的間距為給定的 minimum terminal spacing 。

c. 我們將 terminal 盡可能擺在中間的格子，再將其位置固定，以作為不可移動的元件。此時 RePlAce 會參考已固定的 hybrid bonding terminal 位置，將 instance 擺置得較為分散。

4. 重複執行 mixed size global placement 和 hybrid bonding terminal placement

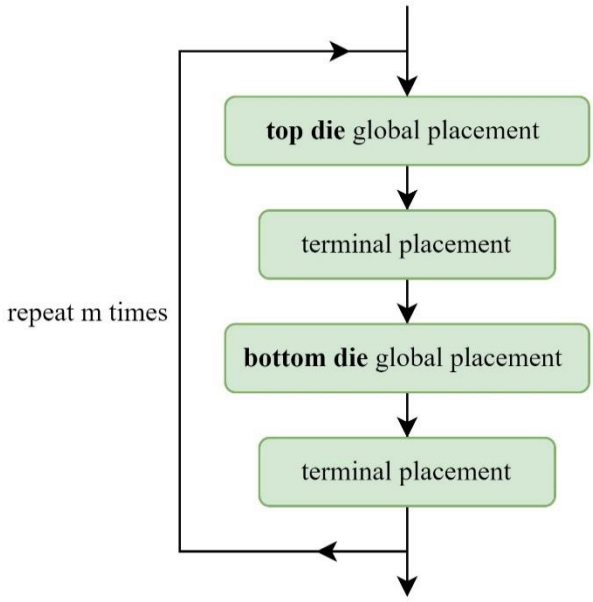

圖 2. global placement 流程

a. 與教授討論後，我們決定選擇 RePlAce 來進行 placement 。因為不論在擺置時間、HPWL 結果這兩方面，它都優於許多現有的 placement 方法，並且可以處理 mixed size global placement ，符合比賽題目要求。
b. 我們首先修改了 RePlAce 的源代碼，讓它只要做完 mixed size global placement 就先回傳擺放位置資訊。之後每做完一個 die 的 global placement ，我們都會調整 terminal 的位置，再固定 terminal 位置並進行另一個 die 的 global placement 。這個重複執行的過程可有效降低 placement 的 HPWL 。
c. 對所有橫跨兩個 die 的 net ，我們傳入另一個 die 上的 HPWL 範圍的中心點，使得每個 net 的 HPWL 範圍跟另一個 die 上的 HPWL 範圍儘量重疊，便能優化 terminal 的擺放位置，降低擺放 terminal 前後的 HPWL 差距。

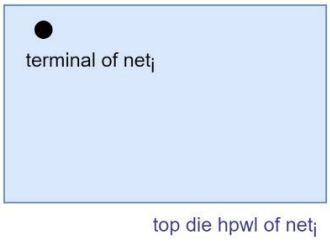

圖 3. HPWL 範圍示意圖
(不加另一個 die 上的 HPWL 中心點)

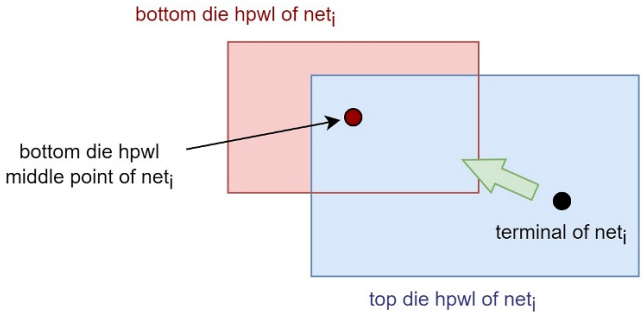

圖 4. HPWL 範圍示意圖
(加另一個 die 上的 HPWL 中心點)

假設此時在做 top die 的 mixed size global placement，對於一個橫跨兩個 die 的 net i (以下以 N 指稱 net i)，圖 3 是不傳入 N 在 bottom die 上的 HPWL 中心點，只有傳入固定位置的 terminal of N，藍長方形會是假設得到的 top die HPWL of N。圖 4 則是傳入 N 在 bottom die 上的 HPWL 中心點和 terminal of N，得到的 top die HPWL of N 會往 N 在 bottom die 上的 HPWL 中心點移，做完 top die 的 mixed size global placement 後調整 terminal 的位置時，terminal of N 也會往綠色箭頭(左上角)移動。經過多次迭代，N 在 bottom die 的 HPWL 會和 top die 上的 HPWL 漸漸移到相近的平面上位置，當 N 在兩個 die 上的 HPWL 在平面上位置越近，擺放 N 的 terminal 前後的 HPWL 增加量就會越小。

5. 合法化 Macro Placement

a. 由於 RePlAce 只是先做完 mixed size global placement，對 macro 與 standard cell 進行粗略的擺放，所以仍存在 macro overlap 的問題。我們採用 B*-tree [5] 和 Simulated Annealing 的 non-slicing floorplanning algorithm 來處理 macro overlap。

b. 選擇 B*-tree 是因為它比現存的其他 non-slicing floorplans 表示法更具有高效性與靈活性，並且能適應不同的布局而給出近乎最佳的面積使用結果。若要將一個 admissible placement 的結果轉換成 B*-Tree 表示，每個節點的左小孩會是位於右側最低的 macro ，而右小孩則是與自己左側座標對齊的上方 macro 。我們先將進行完 mixed size global placement 的擺置結果中的 macros 轉換成 B*-tree，接著在每次的 SA 迭代中，從「旋轉一個 block 」、「交換兩個 block 」與「移動一個 block 」 三種方法中，隨機選擇其中一種做轉換，同時更新 B*- tree，不斷地進行迭代，直到找出最佳解。

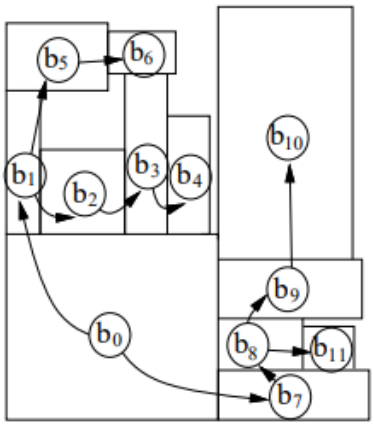

圖 5. An admissible placement[5]

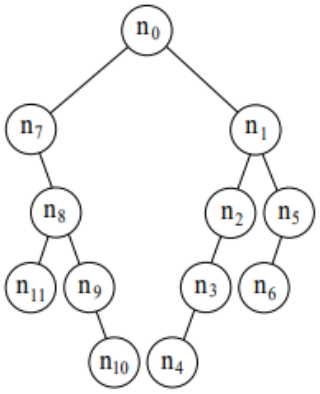

圖 6. The(horizontal) B*-tree representing the placement[5]

c. 在 open source code 中， SA 的一次迭代有三種轉換方式，分別是旋轉一個 block 、交換兩個 block 、移動一個 block ，又對 floorplanning 來說， block 只有兩種方向：R0 跟 R90 (R90 == R270，R0 == R180)。然而 macro 上有很多 pin ， macro 四個旋轉方向產生的 HPWL 都不同，因此我們對每個 block 加上 R180 和 R270 兩種方向，藉此找到 macro 的最佳旋轉角度。

d. 我們也增加每個 block 的 pin 位置，並同時傳入當前 die 上的 standard cell ，以及他的 pin 位置和 terminal 位置，使得擺放 macro 的同時考慮 standard cell 和 terminal 的位置，以找到 macro 的最佳擺放位置。

e. 我們做的是 fixed-outline floorplanning，因為上下 die 都有面積限制，必須在有限的空間內擺完所有 instance 且不能 overlap 。計算 outline 我們分成以下 (3.4) 式和 (3.5) 式兩種方法。

$$outline = \sum_{i \in \forall blocks} area_i \times dead\ space\ ratio \qquad (3.4.1)$$

$$dead\ space\ ratio = 1.0 \qquad (3.4.2)$$

如 (3.4) 式是第一種公式，我們將 dead space ratio 設定成 1.0 ，因為讓 macro 能夠分散一點且不 overlap 是我們的主要目標。

$$outline = die\ area \qquad (3.5)$$

如 (3.5) 式是第二種公式，只有在第一種公式算完的 outline ，超過 die 的面積時才會使用。

在計算 cost function 時，如果這次的擺置會超出 die 的水平或垂直邊界，我們將使它產生一定的成本，以反映擺置不符合的現象。為確保最後的結果都能讓 macro 符合 die 的面積限制，cost function 的公式如 (3.6) 式。

$$\left(\sum_{i \in \forall invalid\ placed\ blocks} Area_i\right) \times 1000 + now\ HPWL \quad (3.6)$$

6. 重複數次 standard cell global placement、detailed placement、terminal placement

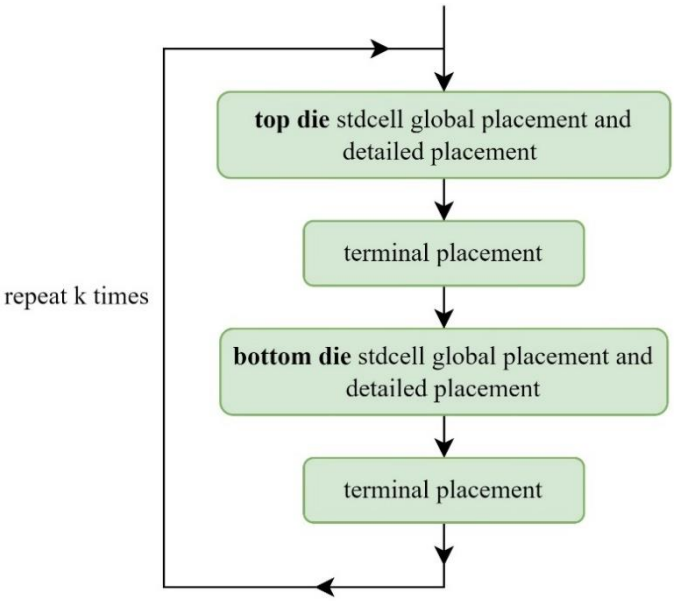

圖 7. Detailed placement 流程

a. 做完 macro legalization 後，我們已經得到 macro 不會 overlap 的擺置結果。但因為還沒進行 detailed placement ，仍存在 standard cells 不在 row 上與重疊的問題。故我們將 macro 與 hybrid bonding terminal 位置固定，作為不可移動之元件，與 standard cells 再讓 RePlAce 做一次擺置。這裡也有修改 RePlAce 的源代碼，讓它先對 standard cell 做 global placement ，才進行所有 instance 的 detailed placement 。和之前一樣，每做完一個 die 的 placement ，都要根據擺好的 instance 位置再調整一次 hybrid bonding terminal ，接著進行下一次 detailed placement 。
b. 我們選擇用 ntuplace3[6] 來做 detailed placement ， ntuplace3 的程式裡會先擺放出一個 instance 間都不會 overlap 且 standard cells 在 row 上的合法結果，接著微調 standard cells 的位置，找到 HPWL 最小的 placement 。
c. 同 4. d ，對所有橫跨兩個 die 的 net ，傳入另一個 die 上的 HPWL 範圍的中心點，使得每個 net 的 HPWL 範圍 跟另一個 die 上的 HPWL 範圍儘量重疊，便能優化 terminal 的擺放位置，降低擺放 terminal 前後的 HPWL 差距。經過測試，我們在 global placement 和 detailed placement 都加上這個步驟，可以優化約 15% 的 HPWL 。

四、 研究結果與討論

此題共有 4 個公開測資，我們成功通過所有公開測資，並且取得了不錯的表現，實驗結果如表 2 所示。我們盡可能降低了 partition 和 macro legalization 所需的時間，並且改成使用 fread 、 fwrite 來讀寫 file ，因此最後的執行時間，case 4 從原本需耗時 6 個小時到最後僅需不到 1 小時。藉由降低兩個 die 的總使用面積（研究方法 2. b ）以及在 placement 時考慮另一個 die 上的 instance 位置（研究方法 4. d 和 6. c ）， HPWL 從最初的合法結果進步了 20% 。但我們的作法仍有改善的空間，以下是我們認為可以改善的地方。

1. hMetis 是 non-deterministic，每次 partition 的結果都不一樣。如果能夠在執行 hMetis 時加入平行程式的技術，就能省下更多時間，或是能在同樣的時間內取得更多次 hMetis 後的結果並選擇最佳 cut。
2. 我們在 partition 完後就不再移動 instance 至另一個 die，也許嘗試在 placement 時搬動 instance，或是一開始就以 3D placement 的結果來決定如何 partition instance 到兩個 die ，會得到更好的結果。
3. 我們使用 B*-tree algorithm 來進行 macro legalization ，讓 macro 彼此間不重疊也不超過邊界。但我們的 B*-tree 在擺放時只考慮 macro 是否重疊，如果能把 standard cell 的位置也考慮進去，並降低兩者的重疊面積，也許可以讓 B*-tree 中計算的 HPWL 更接近最後 detailed placement 完的結果，使 macro legalization 的結果更準確。

Benchmark	# Instance	HPWL	Runtime(sec)
Case1	8	133	316
Case2	13907	38383724	181
Case3	124265	154108743	568
Case4	740243	1528482794	4458

表 2. 實驗結果

五、 團隊合作方式

一開始，我們三人各自研讀幾篇相關論文，以探討如何解決問題。隨後，我們將我們所讀的內容與彼此討論，並共同決定實施的方法。然後，我們分為兩組，一組負責處理分區（Partition）成兩組的部分，而另一組負責處理放置（Placement）的部分。最後，針對不合法的結果和超時的問題，我們共同討論並一起解決。

貢獻度
余伽璇 — 36.5%
林晨 — 36.5%
陳美晴 — 27%

六、 結論

此次參與 ICCAD 2023 Contest 是我們三人第一次接觸 EDA 相關比賽，與教授討論後，我們認為很難在短時間內實作出能夠超越現有 partitioning 與 placement 演算法的表現。因此在反覆測試後，決定使用開源工具 hMetis 處理 partitioning、RePlAce 處理 placement，佐以 B*-tree 與 Simulated Annealing 解決元件重疊的問題，再搭配一些自己實作的優化方法取得更好的 3D placement 結果。最後我們成功通過了所有公開測資，並且最終成績在官方釋出的 Beta 版本測試結果中位列第六名，即使是與其它多由碩博生組成的隊伍競爭，仍獲得令人滿意的成果。未來我們也志在深入改善優化目前的策略，以得到更好的結果。

七、 參考文獻

[1] 3D Placement with Macros. [Online]. Available: http://iccad-contest.org/Document/Problems/2023 Problem B - 3D Placement with Macros.pdf

[2] G. Karypis, R. Aggarwal, V. Kumar, and S. Shekhar, “Multilevel Hypergraph Partitioning: Application in VLSI Domain,” In *Proc. Design Automation Conference*, pp. 526-529, USA, 1997.

[3] C. M. Fiduccia and R. M. Mattheyses, “A Linear Time Heuristic for Improving Network Partitions,” In *Proc. Design Automation Conference*, pp, 175–181, 1982.

[4] B* Tree. [Online]. Available: https://github.com/jacky860226/CS6135-VLSI-PDA-HW3/tree/master

[5] Yun-Chih Chang, Yao-Wen Chang, Guang-Ming Wu, and Shu-Wei W, “B*-Trees: a new representation for non-slicing floorplans,” *Proc. of ACM/IEEE Design Automation Conference,* pp. 458-463, June 2000.

[6] T.-C. Chen, Z.-W. Jiang, T.-C. Hsu, H.-C. Chen, and Y.-W. Chang, “Ntuplace3: An analytical placer for large-scale mixed-size designs with preplaced blocks and density constraints,” IEEE TCAD, vol. 27, no. 7, pp. 1228–1240, 2008.

指導教授簽名： 

日期：2023年 10 月 8 日