

電子設計自動化導論 - 期末報告

Introduction to Electronic Design Automation - Final Project

B00902051

B01901046

B01901065

李益昌

謝沅廷

陳韻竹

指導教授：江介宏

摘要

這份期末報告以「ICCAD Contest 2015 Problem C : Incremental Timing-driven Placement」為題，實現一個自動化的 placement 程式，在不改變 global placement 特性（密度、繞線複雜度等等）的前提下進行 local placement，試圖達到最大的 total negative slack (TNS) 和 worst negative slack (WNS)。本競賽題限制了每個 cell 的最大移動距離，以及 cell 擺放的 constraint；此外，也要考慮 local clock buffer (LCB) 和 flip-flop (FF) 的連接關係。這份程式執行之後，WNS 和 TNS 可降低 5%。

關鍵字：Timing-driven Placement、Local Clock Buffer、WNS、TNS。

I. 題目背景

傳統的 placement 大多試圖優化 total wire length，並沒有將每條 path 的 timing 資訊納入考量；但實際上每個電路都會有一條或多條耗時最久的 critical path 讓整體 clock cycle time 沒辦法壓得更低，這些 critical path 如果在 placement 時就設計的較短，則能夠在不影響 total wire length 的前提下優化電路運算速度。這種 placement 的方法稱為 timing-driven placement (TDP)，必須同時考量 timer 在每個 iteration 提供的 timing 資訊，試圖找到一個 WNS 和 TNS 最大（即負的最少）的 placement。TDP 實作上最大的困難點在於：1. timer 的資訊不夠精準。2. 調整 cell 可能同時降低其中一條 critical path 却新增了另一條。

II. 題目限制 (Constraints)

本競賽目標即：維持 global placement 的 quality (placement density、pin density、routing congestion)，並在最大可移動範圍 (maximum movement) 的限制下，優化 timing violation (包含 Δ TNS 和 Δ WNS)。

Constraint 依照強制程度分成：若違反則依照程度扣分的 soft constraint，以及完全不容許違反的 hard constraint。

Soft constraints:

1. Similar routing congestion
2. Pin/Placement density

Hard constraints:

1. Maximum displacement
2. Legality

將題目目標及 hard constraint 以公式的方式表示：

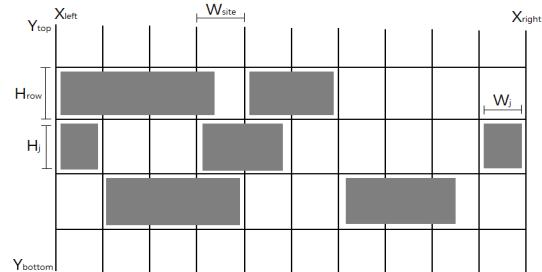


圖 1. 符合 Legality 的 cell 擺放示意圖。

$$\text{Max : tns}^L + \text{tns}^E \quad (1)$$

S.t. :

$$X_{\text{left}} \leq x_j \leq X_{\text{right}} - W_j \quad (2)$$

$$Y_{\text{bottom}} \leq y_j \leq Y_{\text{top}} - H_j \quad (3)$$

$$x_j = \left\lfloor \frac{x_j}{W_{\text{site}}} \right\rfloor \times W_{\text{site}} \quad (4)$$

$$y_j = \left\lfloor \frac{y_j}{H_{\text{row}}} \right\rfloor \times H_{\text{row}} \quad (5)$$

$$x_j + W_j \leq x_{j+1} \quad (6)$$

$$y_j + H_j \leq y_{j+1} \quad (7)$$

$$\max_{j \in C} (|x_j - x_j^0| - |y_j - y_j^0|) \leq D_{\text{max}} \quad (8)$$

其中 x_j 和 y_j 為第 j 個 cell 的位置， D_{max} 為題目限制的最大可移動距離， C 是所有 cell 的集合。其他參數如圖 1 所示。本題 standard cell 的擺放均在特定橫排上，每個 cell 等高但不同寬。特別要注意的是，每個 row 都切割成若干個等寬的 site，cell 擺放時必須從整數個 site 的位置開始擺起，也就是所謂的「可擺放空間」必須是整個都空著的 site。

III. Local Clock Buffer (LCB)

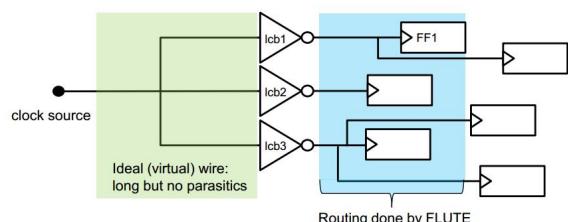


圖 2. FF 和 LCB 的連接關係圖。

除了 cell 的擺放之外，本題還有另一個可以更改的 placement 設定，即 local clock buffer 和 flip flop (FF) 的連接關係。

通常整個電路會使用同一個 clock，換句話說 clock 接出去的線會拉很長，除了用 clock tree 來平衡每個 FF 收到 clock 時的 delay time 之外，本題多加了一個 LCB 的機制。如圖 2 所示，每個 FF 都必須連接到一個 LCB，且只能連接到一個 LCB (但一個 LCB 可以向後連接到數個 FF)，在 LCB 之前的 wire 都假設成沒有 delay 的理想接線，LCB 之後則依然用 FLUTE 繞線，並且必須考慮 timing 資訊。

本題可以更改的即 LCB 和 FF 的連接關係，例如可將 FF1 改成連接到 lcb2，在 placement 時所做的更改都必須寫在 .ops 檔內。

IV. 檔案格式

輸入：
 .iccad2015：列出測資所需檔案
 .v：cell 間的連接關係
 .lef：cell library
 .def：cell 的擺放位置
 .sdc：初始 timing 條件
 .lib：delay 的時間資訊
 輸出：
 .def：調整過的 cell 位置
 .ops：有更動 LCB 和 FF 連接關係

V. 解題架構

首先將 Timer 回傳的每條線上的 timing 資訊轉成 net-weight，方便接下來移動 cell 時知道每一條 path 在整個電路上是否為 critical path (及其相對重要性)；此方法稱為 net-based TDP，分為靜態及動態兩種。

靜態的作法是呼叫一次 Timer 之後，一直使用這組 timing 換算的 net-weight 直到 placement 結束，好處是不用每次 iteration 都要呼叫 Timer，因為呼叫 Timer 很花時間；動態的作法則是每次移動 cell 的 iteration 之前都要呼叫一次 Timer，也就是 net-weight 是會動態更新的。雖然相對而言動態比較花時間，但每次 iteration 之後 timing 的資訊也會一直更動，使用靜態 TDP 到後面會嚴重失真，故最後還是採用動態的作法。

接著，用 Lagrangian Relaxation 來處理本題的 hard constraints，將 Lagrange multiplier (LM) λ^L 和 λ^E 作為 net-weight，其值會根據 Timer 回傳的 timing 資訊動態更新。定義 late arrival time a_j^L 及 early arrival time a_j^E 如下：

$$a_j^L = \max_{c_i \in F_j} (a_i^L + d_{i,j}^L) \quad (9)$$

$$a_j^E = \min_{c_i \in F_j} (a_i^E + d_{i,j}^E) \quad (10)$$

其中 F_j 為第 j 個 cell 的 fanin set， $d_{i,j}^L$ 及 $d_{i,j}^E$ 為從 c_i

的輸出到 c_j 的輸出之間的 late delay 和 early delay，這兩個 delay 的資訊來自 timer。定義 late slack s_j^L 及 early slack s_j^E 如下：

$$s_j^L = r_j^L - a_j^L \quad \forall j \in PO \quad (11)$$

$$s_j^E = a_j^E - r_j^E \quad \forall j \in PO \quad (12)$$

其中 PO 為 primary output set， r_j^L 和 r_j^E 分別為 late required time 及 early required time，根據整個電路的 clock 頻率而定。本題 timing 上的目標及 constraint，以 Lagrangian Relaxation formula 表示如下：

$$\text{Min: } - \sum_{j \in PO} s_j^L - \sum_{j \in PO} s_j^E \quad (13)$$

S.t:

$$s_j^L \leq 0, \quad s_j^E \leq 0 \quad \forall j \in PO \quad (14)$$

$$r_j^L - a_j^L \geq s_j^L, \quad a_j^E - r_j^E \geq s_j^E \quad \forall j \in PO \quad (15)$$

$$a_i^L + d_{i,j}^L \leq a_j^L \quad a_i^E + d_{i,j}^E \geq a_j^E \quad \forall j \in C \quad (16)$$

$$(2) \text{ to } (8) \quad (17)$$

其中 slacks s_j^L 及 s_j^E 必須確保非正數。綜合 objective function 和 constraint，Lagrangian function 為：

$$\begin{aligned} L_\lambda : & - \sum_{j \in PO} s_j^L - \sum_{j \in PO} s_j^E + \sum_{j \in PO} \lambda_j^L s_j^L + \sum_{j \in PO} \lambda_j^E s_j^E \\ & + \sum_{j \in PO} \lambda_j^L (s_j^L - r_j^L + a_j^L) + \sum_{j \in PO} \lambda_j^E (s_j^E - a_j^E + r_j^E) \\ & + \sum_{j \in C} \left(\sum_{i \in F_j} \lambda_{i,j}^L (a_i^L + d_{i,j}^L - a_j^L) + \sum_{i \in F_j} \lambda_{i,j}^E (a_j^E - a_i^E - d_{i,j}^E) \right) \end{aligned} \quad (18)$$

簡化後的 L_λ 為

$$L_\lambda : \sum_{j \in C} \left(\sum_{i \in F_j} \lambda_{i,j}^L (d_{i,j}^L) + \sum_{i \in F_j} \lambda_{i,j}^E (-d_{i,j}^E) \right) \quad (19)$$

解 Lagrangian problem 即反覆的解兩個子問題：在給定的 LM 下最小化 objective function 的 Lagrangian relaxation subproblem (LRS) 和隨時更新 LM 使 LRS 的解最大化 Lagrangian dual problem (LDP)：

$$\text{LRS: } Q_\lambda = \min_{(x_j, y_j) \forall j \in C} L_\lambda \quad (20)$$

$$\text{LDP : } \max_{\lambda \geq 0} Q_\lambda \quad (21)$$

在這個演算法下，移動每個 cell 的順序依照一個自定義的 specific order：從 primary input 開始，用橫向優先搜尋（breadth first search，BFS）排好每個 cell 的順序，最後再將此順序反向，也就是最終會變成從 primary output 開始 visit。

此外，LRS 可再細分為兩個部分：容許 cell overlap 的 discrete local search，及 overlap removal。

VI. 演算法

第一步先設定 λ^L 和 λ^E 的初始值：每個 primary output 的 λ 均為 1、每個 cell 其 fanin 的 λ 總和等於其 output 的 λ ，且 fanin 的 λ 均相等。Discrete local search、overlap removal 和 LM update (LDP) 的虛擬程式碼分別如下：

Discrete local search

Input : Placed circuit with timing violation, λ^L, λ^E

Output: Timing optimized circuit

for each movable cell $c_j \in C$ in specific order **do**

 best_cost $\leftarrow \infty$

for each candidate location ρ within the

 fixed-size window **do**

 place c_j in the coordinate of location ρ

$$\Delta C_{Lj} \leftarrow (L_j - \hat{L}_j)c$$

$$\cos t^L \leftarrow \sum_{i \in F_j} (\hat{d}_{i,j}^L + A_1 \Delta C_{Lj}) \lambda_{i,j}^L$$

$$\cos t^E \leftarrow \sum_{i \in F_j} (\hat{d}_{i,j}^E + A_1 \Delta C_{Lj}) \lambda_{i,j}^E$$

for each fanout c_k **do**

$$\Delta \tau_{j,k} \leftarrow \left(K_D \cdot r \cdot l_{j,k} \left(\frac{c \cdot L_j}{2} + c_{pin_k} \right) \right) - \hat{\tau}_{j,k}$$

$$\cos t^L \leftarrow \cos t^L + (\hat{d}_{j,k}^L + \Delta \tau_{j,k}) \lambda_{j,k}^L$$

$$\cos t^E \leftarrow \cos t^E + (\hat{d}_{j,k}^E + \Delta \tau_{j,k}) \lambda_{j,k}^E$$

end

$$\cos t_\rho \leftarrow \alpha \cos t^L + \omega \cos t^E$$

if $\cos t_\rho < \text{best_cost}$ **then**

 (best_rho, best_cost) $\leftarrow (\rho, \cos t_\rho)$

end

 place c_j in the coordinate of best_rho

end

其中 C_L 為電容變化量、 L 為半周長導線長度 (HPWL)、 c 為單位微米上的電容、 A_1 為一常數、 τ 為 Elmore delay、 K_D 為一常數、 r 為單位微米上的電阻、 $l_{j,k}$ 為 c_j 到 c_k 的曼哈頓距離，window size 指的則是每個 cell 在移動範圍內可以選擇的新位置數量。

因為這步驟是容許 overlap 的，故下一步必須將移動完後有 overlap 的情況移除掉。

Overlap removal 概念上很簡單，就是將每個移動後有 overlap 的 cell 再移到最近的空格（空的 site）內。但由於 cell 的寬度都不同，剩餘空格也要用大小來分類。

首先建一個紅黑樹紀錄所需的各種空格大小 (size of site)，接著再建立數個紅黑樹記錄相對應大小的剩餘空格位置（每個紅黑樹代表一種空格大小）；遇到 overlap 的時候，cell 會依據大小去搜尋適當的紅黑樹，並在位置離自己最近的前一個與後一個空格內，選一個距離較短的當作更新後的位置。使用紅黑樹的好處是：搜尋時間可以從 n^2 縮短為 $n \log(n)$ 。

最後是 LDP，也就是更新 LM 的方法。參考[2]裡面的數學模型後，LM 更新的方式如下：

$$\lambda_j^L = \lambda_j^L \left(\frac{a_j^L}{r_j^L} \right), \lambda_j^E = \lambda_j^E \left(\frac{r_j^E}{a_j^E} \right), \forall j \in PO \quad (22)$$

$$\lambda_{i,j}^L = \lambda_{i,j}^L \left(\frac{a_i^L + d_{i,j}^L}{a_j^L} \right), \lambda_{i,j}^E = \lambda_{i,j}^E \left(\frac{a_j^E}{a_i^E + d_{i,j}^E} \right), \forall j \in C \quad (23)$$

LM 更改的順序也是依照上述的 specific order。

VII. 執行結果

目前以兩個簡單的測資來測試整份程式運作的結果。競賽給的評分程式會依據微調過的 placement 來給分，主要是考量 cell 的最大移動距離和 TNS、WNS。

```

Analyzing placement ...
max displ. (um) : 0
numBins : 1 ( 1 x 1 )
bin dimension : 50x50 x 30780
target util : 1
ABU_2,5,10,20 : 0, 0, 0, 0
ABU penalty : 0
alpha : 1

Analyzing timing ..
FLUTE: Total 0 internal Steiner points are found.
Slicing wire segments: 9 --> 9 (< 10 um)
HPWL, STWL (um) : 36.4645, 31.957
Scaled STWL : 31.957 ( 0 % )
Clock period : 80
early WNS, TNS : 0, 0
late WNS, TNS : -103.734, -103.734

Checking legality ...
Testing ILLEGAL_TYPE 1 ... 0
Testing ILLEGAL_TYPE 2 ... 0
Testing ILLEGAL_TYPE 3 ... 0
Testing ILLEGAL_TYPE 4 ... 0
Testing ILLEGAL_TYPE 5 ... 0
Testing ILLEGAL_TYPE 6 ... 0
Placement is LEGAL.

```

圖 3. 第一筆測資的原始跑分結果。

```

Analyzing placement ...
max disp. (um) : 1.33
numBins : 1 ( 1 x 1 )
bin dimension : 30780 x 30780
target util : 1
ABU [2,5,10,20] : 0, 0, 0, 0
ABU penalty : 0
alpha : 1

Analyzing timing ...
FLUTE: Total 0 internal Steiner points are found.
Slicing wire segments: 9 --> 9 (< 10 um )
HPWL, STWL (um) : 35.5745, 31.0667
Scaled STWL : 31.067 ( 0% )
Clock period : 80
early WNS, TNS : 0, 0
late WNS, TNS : -102.277, -102.277

Checking legality ...
Testing ILLEGAL_TYPE 1 .. 0
Testing ILLEGAL_TYPE 2 .. 0
Testing ILLEGAL_TYPE 3 .. 0
Testing ILLEGAL_TYPE 4 .. 0
Testing ILLEGAL_TYPE 5 .. 0
Testing ILLEGAL_TYPE 6 .. 0
Placement is LEGAL.

```

圖 4. 第一筆測資微調後的跑分結果 (Result One 為例)。

圖 4. Overlap removal 前後的 cell 擺放示意圖，其中 0

代表空格、1 代表放置一個 cell、2 以上代表有 overlap。

Result	iteration	window size	WNS	TNS	max displ. (μm)	overlap
Original	—	—	-103.734	-103.734	0	0
One	1	7	-102.277	-102.277	1.33	0
Two	1	10	-102.711	-102.711	1.9	0
Three	1	32	-104.302	-104.302	3.99	1
(after overlap removal)	1	32	-111.145	-111.145	3.42	0
Four	3	7	-100.172	-100.172	1.33	0
Five	5	7	-100.172	-100.172	1.33	0

表 1. 第一筆測資位置微調後的相關數據。

Result	iteration	window size	WNS	TNS	max displ. (μm)	overlap
Original	—	—	-111.917	-111.917	0	0
One	1	7	-106.078	-106.078	1.33	0
Two	1	10	-103.351	-103.351	1.9	0
Three	1	32	-104.302	-104.302	5.7	1
(after overlap removal)	1	32	-111.145	-111.145	2.47	0
Four	3	7	-106.078	-106.078	1.33	0
Five	5	7	-106.078	-106.078	1.33	0

表 2. 第二筆測資位置微調後的相關數據。

由上表可以看出，window size 越大（候選位置越多）可以有比較多的選擇空間，但因為 Timer 每次 iteration 約給的 timing 資訊是針對原始位置的，所以移動的越遠則 timing 的失真程度會越高，調整後的結果可能反而不如預期。Iteration 的部分，跑越多次顯然調整的結果會越好，且因為每次 iteration 之前都會再重新呼叫一次 Timer，故沒有失真的問題；但缺點是跑越多輪，程式執行的時間就會越長。

VIII. 未來展望

目前的進度做到可以確實移動 cell、移除 removal，但要讓最後 WNS、TNS 表現更好，還有一些可以改進的方向：

- (1) 使用 Hybrid TDP，同時考慮 Path-based TDP
 - (2) 找一個更好的 net-weight 模型（目前 LM 初始值的設定可能不盡理想）
 - (3) 優化 removal 的演算法，目前的演算法雖然可以確實移除 overlap，但 WNS、TNS 也會顯著上升
 - (4) 考慮 LCB 的連接關係，目前的進度尚未同時把 LCB 和 FF 納入考量

分工

Paper survey: 謝沅廷、陳韻竹

C++ implement: 謝沅廷、李益昌

Report Writing: 陳韻竹

致謝

感謝黃于真助教和實驗室學長提供的參考資料和討論指教。

參考文獻

- [1] Chrystian Guth et al. Timing-Driven Placement Based on Dynamic Net-Weighting for Efficient Slack Histogram Compression
 - [2] H. Tennakoon et al. Nonconvex gate delay modeling and delay optimization