以偶角模組列表示法解決在叢聚限制下不可切割結構之平面規劃問題

Non-Slicing Floorplan with Clustering Constraints Using Corner Block List

吳光閔
南華大學資訊管理學系碩士班

黃嘉政
南華大學資訊管理學系碩士班

摘　要

幾年來半導體產業蓬勃發展，造就了台灣經濟的起飛，但是現今積體電路晶片之設計，慢慢趨向智慧財產權（IP, Intellectual Property）的方向來設計。所以現在大部份積體電路在設計上，都會由一種以上的 IP 所組成，因此平面規劃（floorplan）這方面的問題，就顯得更加地重要。

在平面規劃的問題上，基於電路設計上的需求，某些模組在最後封裝的時候，必須滿足一些限制條件。而在我們的論文中，我們是利用偶角模組列（Corner Block List）表示法，配合我們的演算法，來解決在叢聚限制下之平面規劃問題，最後實驗結果，也証明我們的演算法有很好的效能。

關鍵字：智慧財產權（intellectual property）、平面規劃（floorplan）、電路設計（circuits design）

ABSTRACT

Since integrated circuits design are tending to intellectual property (IP) mode. Circuits are composed of IP modules. Therefore, the floorplan problems are become important.

In floorplan, some modules are required to satisfy some constraints in the final packing. In this thesis, we propose an algorithm based on Corner Block List for the floorplan with clustering constraints. Experiment results show that our algorithm is very efficient.

Keywords: intellectual property, floorplan, circuits design

壹、前言

近幾年來半導體產業蓬勃發展，造就了台灣經濟的起飛，但是現今積體電路（IC）晶片之設計，慢慢趨向智慧財產權（IP, Intellectual Property）的方向來設計，由於和半導體產業有很大的關連，所以也有人把它稱爲矽智產。現在大部份的積體電路（IC）大多由一種以上的 IP 所組成，因此平面規劃（floorplan）這方面的問題，就顯得更加的重要。各功能的 IP 模組，就好像是一塊塊大小不同的拼圖一般，所以在積體電路上，對於不同功能的 IP 組合，如果能在平面規劃的問題上，得到最佳 IP 模組之組合，那麼對於這個積體電路上之效能，或是對製造電路整體之成本，會造成相當大之影響，亦因爲如此平面規劃在積體電路之設計流程中，是一個相當重要的研究範疇。

在平面規劃（Floorplan）的問題中，我們可以依據模組之間的排列結構，把問題分成「可切割的結構（Slicing）」，以及「不可切割的結構（Non-Slicing）」兩類。前者可切割的結構（Slicing Structure），是在所組成的模組之中，可以用垂直的切線（Vertical lines），以及水平的切線（Horizontal lines），把模組切割成不會重疊的數個區塊，如圖 1(a)所示，圖中 V_1 這條垂直的切線，可以把整個模組切成左右兩邊，而在左邊的模組，又可以用 H_1 這條水平切線，切成單一的 a、c 兩個模組，另外在右邊也可以先用一條水平切線 H_2，分割成上下兩部份，再用 V_2 這條垂直切線，將其完全分割成單一模組 b、d，以及 e，所以可以用這種階層式分割法，將一平面規劃的排列結構，切成單一模組，這種結構稱爲可分割結構；後者不可切割的結構（Non-Slicing Structure），則是不能用垂直的切線（Vertical lines），以及水平的切線（Horizontal lines），把模組切割成不會重疊的數個區塊，換句話說，就是不屬於可切割結構的問題，其結構如圖 1(b)所示，圖中 V_1、V_2 代表垂直的 2 條切線，而 H_1、H_2、H_3 代表水平的 3 條切線，不管從哪一條切線開始分割，皆不能把模組分割開，這種結構稱爲不可切割的結構。

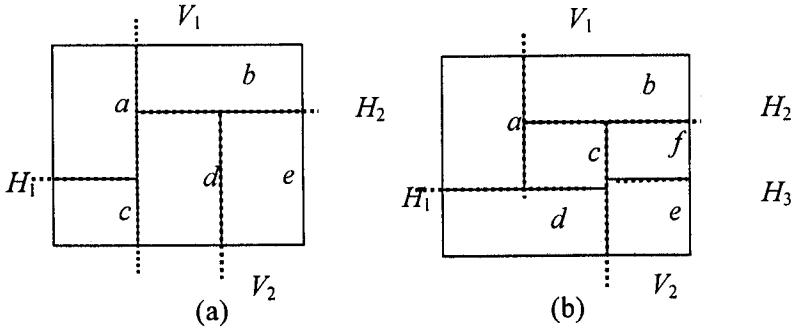


圖 1: 平面規劃模組間的結構 (a)可分割結構（Slicing Structure）；(b)不可分割結構

（Non-Slicing Structure）

目前提出來解決平面規劃的表示方法，主要分成可切割的結構（Slicing）與不可切割的結構（Non-Slicing）兩類。前者有分割樹（Slicing tree）[16]和波蘭表示法（Polish Expressing）[17]兩種；後者有序列組表示法（Sequence-Pair）[12]、BSG 表示法（Bound-Sliceline-Grid）[14]、O-Tree 表示法[3]、B*-tree 表示法[1]、以及 TCG 表示法[10]；最近在 2001 年提出了一種新的表示法，這種表示法稱爲 CBL（corner block list）[4, 5]表示法，是一種利用在平面規劃右上角的位置，以水平或垂直的方向加入或移除模組，而且可以控制加入平面規劃空間時，與其要加入方向切面上，所接觸到的模組數。在這些表示法中，以不可切割結構（Non-Slicing）中的 CBL 表示法最新，而且 CBL 擁有以下的優點，首先，在建構平面規劃時，只需要在線性時間內就可以完成；再者是 CBL 表示法的組合數，亦比 SP 與 BSG 表示法來的少；還有其所占的編碼也比較少。

在處理平面規劃（Floorplan）的問題時，就模組的形狀通常可分爲以下三種模組，第一種是固定模組（Hard Module），其長寬都固定，但是其座標與擺置可以變動的；第二種是彈性模組（Soft Module）[6, 13]，其面積不變，不但長寬比率可以在某一範圍內變動，另外座標可以變動；第三種是直線區域模組（Rectilinear Module）[2, 7, 8, 15, 18]，，所組成的模組，必須在長寬固定的矩形範圍限制內，因此所組成的形狀是固定的，但是所組成的模組之方位與位置，是允許它們變動的。

但是在積體電路的設計上，除了上述的模組型態外，工業界於設計上提出了一些不同的限制條件，所以在元件的平面規劃問題上，就產生了這些限制式，常見的限制有以下四種：(一)預置模組限制式（Pre-placed Module Constraints），其限制式是指某些模組的座標，因爲特定的須求，所以是不可以變動的，目前已經提出的方法，是運用 SP 表示法[12]，以及 BSG 表示法[15]來表示；(二)範圍限制（Ranges Constraints），其意指的就是，有些固定的模組，必須擺在一固定的區域範圍，目前已經提出的方法，有運用波蘭表示法（Polish Expressing） [19] 表示；(三)邊界限制（Boundary Constraints），其意是指在擺置模組的時候，有些模組必須擺置在整個晶片的邊緣，如下方、上方、左方，或者是右方，原因是這些模組要做主要輸出、輸入之途，目前已經提出的方法，有波蘭表示法（Polish Expressing）[20]，以及用 CBL 表示法[11]表示；(四)叢聚限制（Clustering Constraints），其意指的是有些相同的模組，可能因爲功能的需求，因此必須放在一起，以利設計之需求，以及爾後繞線的成本，所以是一個很重要的限制問題，但是目前叢聚限制（Clustering Constraints），只有使用過波蘭表示法（Polish Expressing）[21]，來解決過叢聚限制之問題，然而波蘭表示法只是一種僅能處理可分割結構問題之表示法，但如果最佳解沒有落在可分割結構上，就不能求到較好的解，因爲這時候這個較好的解，就

是落在不可分割結構性上。

本篇論文要解決的是叢聚限制（Clustering Constraints）上之問題，因爲目前這個問題，只有做到可切割性結構上，可是平面規劃（Floorplan）的最佳解，不一定只落在可切割性結構上，這個最佳解也很有可能會落在不可切割性結構中，因此這就是我們所要解決的問題，而我們採用的表示法，是運用 CBL 表示法，因爲它在解決上述的邊界限制（Boundary Constraints）上，有非常好的效果，另外在平面規劃轉擺置上之時間複雜度，只需要 O(n)的時間就可以完成，是一種相當好的表示法。所以本篇論文在綜合以上 CBL 表示法的優點，我們決定運用 CBL 表示法，再配合模擬退火（SA, Simulated annealing）演算法[9]，來解決在積體電路規劃上的叢聚限制（Clustering Constraints）問題，並從中尋找比較好的解。

在實驗上，針對一些 MCNC 所建構的範例，經過我們將演算法撰寫成程式，其實驗求得的結果，顯示我們所提出來以偶角模組列（CBL）爲基礎的表示法，來解決叢取限制的問題，能求得一個很好的解。接下來我們在第二章會進行本研究的問題描述；在第三章會介紹 CBL 表示法；第四章則說明我們的演算法；第五章會顯示實驗的結果；在最後的第六章，會做結論與對未來之發展。

貳、問題描述

我們將積體電路在處理平面規劃問題時，其所設置的叢聚限制（clustering constraint）問題定義如下，對於有 n 個模組之集合 S，$S = \{s_1, s_2, s_3, \ldots, s_n\}$，$S = \{ F, C \}$，其中 $F=\{f_1, f_2, \ldots, f_p\}$，且 $f_i \in S$，$0 \le i \le p$ ，F 爲一般模組之集合，集合內任何一個模組沒有叢聚之限制，所以這些模組爲可以任意擺放的模組集合。

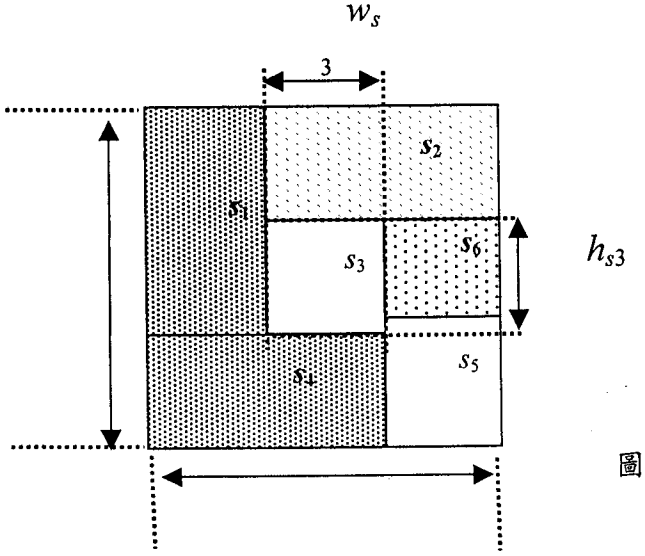


圖 2: 叢聚限制的範例

定義一：叢聚限制

本篇論文所定義的叢聚限制（clustering constraint），其指的是任一參與叢聚之模組，至少會與另一個也是相同群聚之模組，與其幾合學相鄰接（geometrically adjacent），如圖 2 所示，s_1 與 s_4 爲同一叢聚集合，所以兩個模組之間有相鄰，因此符合叢聚限制之條件。

而叢聚限制的問題定義如下，$C = \{C_1, C_2, \ldots, C_q\}$，$C_i \subset S$ ，$0 \le i \le q$，其中 C_q 表示在 C 叢聚集合中，其中的一個叢聚（對於任一個 $c_{qi} \in C_q$ 中，必定會有至少另一個 $c_{qi} \in C_q$，與其幾合學相鄰接），所以我們以 $|C_q|$ 表示 C_q 中參與相同群聚模組的個數，因此 $0 \le |C_1|+|C_2|+|C_3|+\ldots+|C_q| \le n\text{-}p$，此外，對於每個模組 $s_i \in S$，其寬度（width）與高度（height）分別爲 w_i 與 h_i，而面積爲 $a_i=w_i\,h_i$，模組 s_i 皆可以自由的旋轉方向（當 s_i 旋轉 90°的時候，等於 w_i 與 h_i 的值互換），而我們的問題定義如下所示：

問題定義：

平面規劃合理的擺置方式（feasible placement），其必要的條件是使每一個 $C_i \subset C$ 滿足叢聚限制的條件，而且在 S 中的任兩個 s_i 不可以重疊，若沒有滿足這個條件，則爲不合理的解，而最終的目的，就是我們要求平面規劃問題時，加上叢聚限制條件，在滿足合理的擺置方式內，求得最小面積 $\min(A)=HW$。

以圖 2 爲例，模組的集合爲 $S=\{s_1, s_2, s_3, s_4, s_5, s_6\}$，這個集合包含了六個模組，而在這些集合之中，$C=\{C_1(s_1, s_4), C_2(s_2, s_6)\}$，$F=\{s_3, s_5\}$，這個例子中在叢聚限制的模組，皆在相鄰在一起，所以是一個合理的平面規劃之排列結構。

參、CBL（Corner Block List）表示法

各表示法的組合數目（The number of combinations）與轉成擺置（Placement）的複雜度，如表 1 所示；從表中我們可以發現到雖然 O-tree 有不錯的結果，但是它是只能呈現一種特別的表示方法，就是結合壓縮左方與下方的晶片，所組成的一種擺置方式（Placement），所以我們採用了 CBL[4, 5]（Cornet Block List），這個表示方法，它的組合數（The number of combinations）只有 $O(n!\, 2^{3n-3}/n^{1.5})$而已，另外在平面規劃（Floorplan）要轉成擺置（Placement）的時間複雜度上，亦只需要 O(n)的時間，就可以完成，因此我們選擇 CBL 這個表示法，做爲本研究的表示法。

表 1: 平面規劃各表示法之間的分析表

模組結構	表示法	組合數	轉換成擺置的時間複雜度
Slicing	Binary tree	$O(n!\,2^{3n-3}/n^{1.5})$	O(n)
	Polish expression	$O(n!\,2^{3n-3}/n^{1.5})$	O(n)
Non-Slicing	SP	$O((n!)^2)$	$O(n^2)$
	BSG	$n!\,(n^2, n)$	$O(n^2)$
	O-tree	$O(n!\,2^{2n-2}/n^{1.5})$	O(n)
	B*-tree	$O(n!\,2^{2n-2}/n^{1.5})$	O(n)
	TCG	$O(n!\,2^{2n-2}/n^{1.5})$	$O(n^2)$
	CBL	$\mathbf{O(n!\,2^{3n-3}/n^{1.5})}$	**O(n)**

3.1 CBL 的模組與平面規劃

平面規劃（Floorplan）是把積體電路中所需要的元件，以水平與垂直的分割線，在一個矩形的空間中，把這些積體電路元件，規劃在矩形空間之中，而 CBL（Corner Block List）正是其中一種最新的表示法。

3.1.1 限制圖（Constraint Graph）

一個平面規劃（Floorplan）的限制圖，可以用 $G=(V, E)$來表示，其中 V 是限制圖中的節點，在平面規劃中，所代表的是模組切面；而 E 是限制點的邊（Edge），在平面規劃中，所代表的則是每個模組的名稱，如圖 1(b)中的 $a, b, \ldots, f$ 這些模組。

然而所代表的限制圖，共有兩種類型，第一種是西向東的限制圖，所以用最左邊的邊界爲限制圖之起始點，用 W（west）來表示它；而最右邊的邊界爲限制圖之終點，所以用 E（east）來表示它，再把中間所代表的節點與切線連接，就是水平的限制圖（HCG, Horizontal constraint graph），如圖 3 所示；另外一種是南向北的限制圖，所以相對的也是由最下面的切線爲限制圖之起點，所以用 S（south）來表示它；而最上邊的邊界爲限制圖之終點，所以用 N（north）來表示它，再把中間所代表的節點與切線連接，就是垂直限制圖（VCG, vertical constraint graph），如圖 4 所示。總之，利用這個切線與模組名稱的交互混合使用，就能簡單且明瞭的表示限制圖。

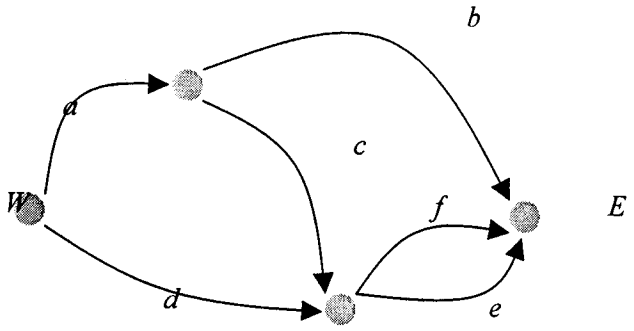


圖 3: 水平的限制圖

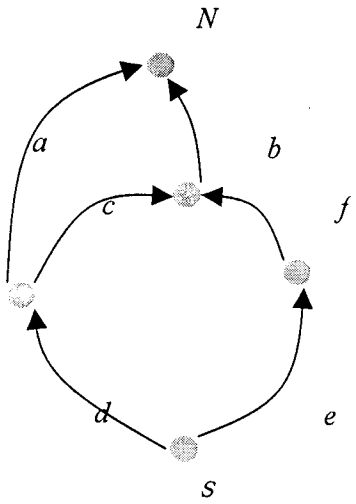


圖 4: 垂直的限制圖

3.1.2 偶角的邊（edge）與模組（Block）

其偶角模組的邊，是指在水平限制圖中，指向 E 節點的邊，另外在垂直限制圖中，指向 N 節點的邊，這兩者結果之集合，就是其偶角模組的邊，若以圖 3 水平限制圖爲例，指向 E 節點的邊爲 “b” 、 “f” ，以及 “e” ，以圖 4 垂直限制圖爲例，指向 N 節點的邊爲 “a” ，以及 “b” ，而兩者之集合爲 “a” 、 “b” 、 “f” ，以及 “e” ，所以這四個邊就是偶角（corner）的邊；而偶角（corner）的模組，就是會同時出現在水平限制圖，以及垂直限制圖的邊，就稱爲偶角（corner）模組（Corner Block），就好比在圖 3 的水平限制圖，與在圖 4 的垂直的限制圖，可以得知 “*b*” 是偶角（corner）模組，但是要注意的是，在利用 CBL 表示平面規劃（floorplan）的時候，CBL 表示法中只會有一個偶角（corner）模組，如圖 1(b)，如果 “*b*” 模組被刪除，那麼 “*f*” 將變成爲偶角（corner）模組。

3.1.3 偶角（corner）模組的擺置方位（Orientation）

在定義偶角（corner）模組加入平面規劃的空間時（floorplan room），是以整個 CBL 表示法之中的偶角（corner）模組爲依據，也就是以偶角（corner）模組左下角的 T 型接點（T -junction）之方位，來判斷這個偶角（corner）加入平面規劃空間時的方位爲垂直（Vertical）或水平（Horizontal）方向。

在圖 5（a）中，“*b*”爲偶角（corner）模組，所以用“*b*”模組的左下角 T 型接點（T -junction）之方位來判斷，其 T 型接點是逆時鐘轉 90 度，所以代表這個偶角（corner）模組是垂直方向來加入平面規劃空間，並以“0”來表示這個偶角（corner）模組是垂直方向來加入平面規劃空間；但在圖 5（b）中，“*e*”爲偶角（corner）模組，所以用“*e*”左下角 T 型接點（T-junction）之方位來判斷，而這個 T 型接點是逆時鐘轉 180 度，所以代表這個偶角（corner） 模組是水平方向加入平面規劃空間，並以“1”來表示這個偶角（corner）模組是水平加入。

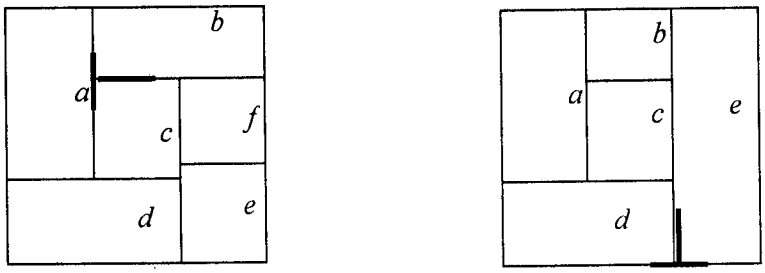


圖 5: 偶角（corner）模組的方位圖 (a) “*b*”爲偶角模組；(b) “*e*”爲偶角模組

3.2 偶角（corner）模組的插入與刪除

偶角（corner）模組，是以水平和垂直的方向來插入新的模組，所以在刪除偶角（corner）模組的時候，也是一樣以原來的水平和垂直方向來刪除新的模組，並改變最後之偶角（corner）模組之名稱，詳細說明如下所示：

3.2.1 刪除偶角（corner）模組

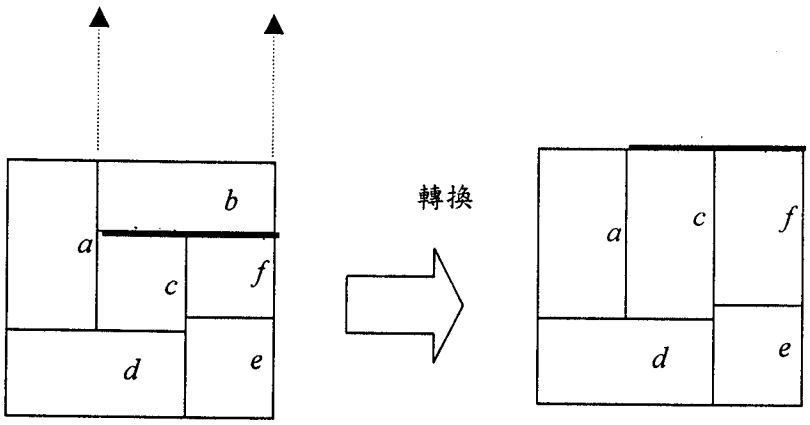


圖 6: CBL 刪除模組轉換圖

在刪除偶角（corner）模組的時候，首先要先從這個偶角（corner）模組的左下角之 T 型接點，判斷這個偶角（corner）模組在加入進來時，是以垂直的方向，或者是水平的方向（T 型接點逆時鐘轉 90 度，代表垂直加入；若是逆時鐘轉 180 度是水平加入），得知加入方向後，再依其原來之方向退出，若是垂直方向就把在偶角（corner）模組下方的水平切線，彈出到最上緣的邊界，如圖 6 所示；反之，若是水平方向，就把偶角（corner）模組左邊的垂直切線，彈出到最右緣的邊界，使其平面規劃空間，又成爲矩形。

除了把偶角（corner）模組刪除之外，必須還要改變垂直與水平之限制圖，所以在刪除 "*b*" 模組之後，因爲 "*b*" 模組是垂直方向刪除，因此在水平限制圖中，要把路徑 "*b*" 刪除，而在垂直限制圖中，除了要刪除一點節點之外，另外要把 "*c*" 與 "*f*" 兩條路徑，由原來刪除的節點，連接到代表最上緣的節點 *N*，這樣就完成了整個刪除的程序，改變後的限制圖，如圖 7 所示；相反，若是刪除的偶角（corner）模組是水平方向的模組，也是必須完成相同的動作。

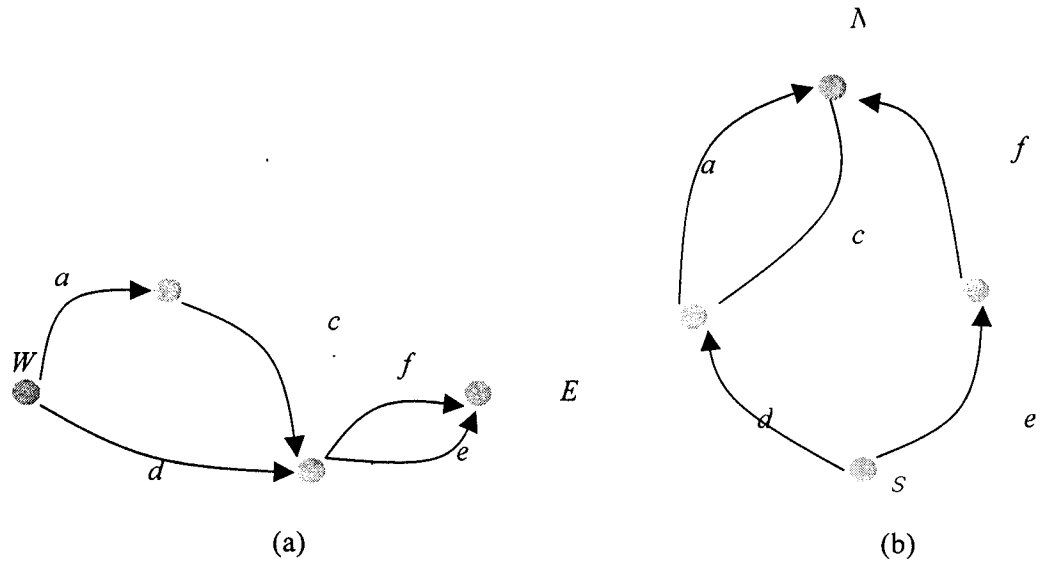


圖 7: 圖 6 中刪除 "*b*" 模組後的限制圖 (a) 水平限制圖；(b) 垂直限制圖

3.2.2 插入偶角（corner）模組

在插入偶角（corner）模組的時候，因爲插入模組有兩種方式，一種是垂直方向插入，另外一種則是水平方向插入。假設要插入的模組是垂直方向插入平面規劃的空間，則可以想像有一條水平的切線由上邊緣壓入平面規劃空間，當然這條水平的切線，是可以決定壓下幾個模組，而壓入之後又形成原來的矩形平面規劃之空間，如圖 8 所示；相反的，如果是決定以水平的方向插入平面規劃空間，其不同的是這條切線變成垂直切線，壓入的方向也變成了從右邊緣壓入平面規劃空間。

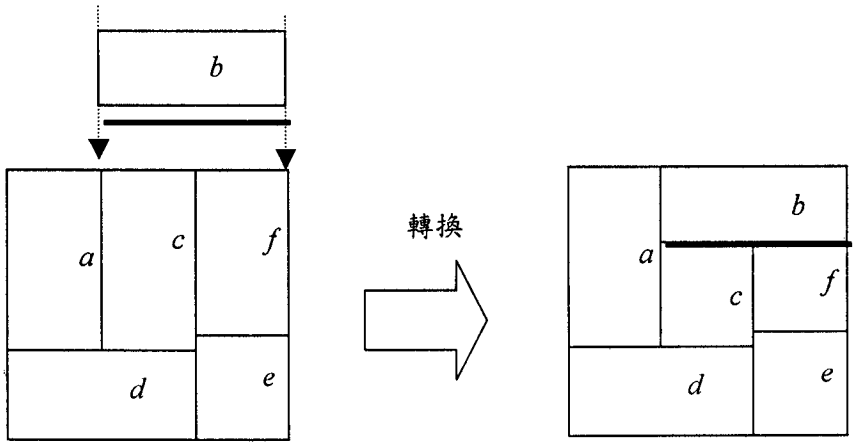


圖 8: CBL 加入模組轉換圖

插入“*b*”模組之後，亦要改變限制圖的結構，也就是說要在水平的限制圖中加入一條路徑“*b*”，而在垂直限制圖中，必須加入一個新的節點，與一條新的路徑“*b*”，其限制圖轉換就如同從圖 7 的限制圖，轉換成圖 3 與圖 4 的限制圖。

3.3 CBL 的參數

CBL 是以遞迴（Recursive）的方式，由後向前的一種記錄結構，除了初始值的模組只有記錄模組名稱外，僅接在初始模組後之模組，除了也有記錄組除名稱外，另外還有記錄二個參數，偶角（corner）模組的方位（Corner block orientation），與模組的切線，共壓入幾個 T 型接點（number of T-junctions uncovered）。因此綜合上面所描述的參數，CBL 共有三個參數，第一個是記錄模組的名稱（*S*），第二個是記錄模組垂直或水平加入平面規劃空間的方位（*L*），第三個則是記錄模組的切線，共壓入幾個 T 型接點（*T*），要注意的是，這些參數是從後面遞迴的方式記錄回來，但是最後會只剩下一個模組，沒有記錄 *L* 和 *T* 參數，也就是 CBL 的初始模組。

第一個參數“*S*”是直接記錄模組名稱，第二個參數“*L*”是以“0”來表示垂直方向加入模組，以“1”來表示水平方向來加入模組；但是，“*T*”這個參數和其它參數比較不一樣的記錄方式，“*T*”是以“1”的個數來表示所壓入的 T 型接點的個數，並且在這些“1”後面，加一個“0”，而加“0”的目的，是爲了要與後者做分隔，因此在從後面讀取“*T*”這個參數的時候，如果遇到“0”，要先再判斷前一個參數是不是“1”，如果答案是的話，那就是有 T 型接點，否則者沒有任何的 T 型接點與偶角（corner）模組的切線相鄰。

若以圖 6 爲例子的話，其“*b*”模組要被刪除，可以從記錄中知道 *S, L, T* 這三個參數資訊，而我們可以知道“*b*”模組記錄的資訊是（*b*, 0, 10），而接下來的順序是“*f*”、“*e*”、“*c*”、“*a*”，以及“*d*”模組，上面是以遞迴的方式由後向前，但是 CBL 資訊的記錄方式（*S, L, T*）分別爲 *S*=（*d, a, c, e, f, b*），*L*=（01100），與 *T*=（0010010）。

3.4 成本函數

我們要求 CBL 表示法的成本函數時，要先把 CBL 表示法轉換成圖 3 的水平限制圖，與圖 4 的垂直限制式，在這兩個限制圖形中分別求出其限制圖的最長路徑，再求得最長路徑後，前者圖 3 的水平限制圖，它的最長路徑是代表積體電路擺置的寬，而後者圖 4 的垂直限制圖，它的最長路徑是代表積體電路擺置的高，把兩者相乘，則可以得到我們想要的成本函數（cost function），也就是積體電路擺置的面積。

我們定義水平限制圖的最長路徑，爲 max{$W \rightarrow E$}；而垂直限制圖的最長路徑，爲 max{$S \rightarrow N$}，$W(p)$代表其權重值，所以我們的成本函數爲：

$$\text{Cost}=(\max\{w_1(p):W \rightarrow E\})*(\max\{w_2(p):S \rightarrow N\})$$

3.5 平面規劃與模組擺置演算法

根據模擬退火（SA, Simulated annealing）演算法，可以知道要求鄰近解，就要試圖改變鄰近的參數，看看結果有沒有預期的好，因此表示法就要有一種本身的擾動方式，而 CBL 本身也有自己的擾動方式，如下所示：

1.交換擾動模組和模組之間的順序

隨機取任兩個模組，但是不包括初始模組，再把這兩個模組交換位置，做爲擾動方式。

2.隨機擾動 ***L*** 這個參數的加入方向（垂直/水平）：

把“0”改變爲“1”（垂直改水平），或者是把“1”改變爲“0”（水平改垂直）。

3.隨機擾動 ***T*** 這個參數的 **T** 接點數目：

把“1”改變成“0”，也就是說原本只接觸一個 T 型接點的，改變成沒有；相反的，或者把“0”改變成“1”，就是原本沒有接觸 T 型接點的，改變成接觸到一個 T 型接點。

4.把模組轉換角度：

可以把模組在原來的狀態下，轉 90 度、180 度，或者是 270 度，來擾動組合方式。

5.將模組倒轉：

把模組在垂直與水平之間的位置，倒轉模組的位置，來擾動組合的方式。

6.置換初始的模組：

隨機取一個不是初始的模組，把這個模組當做是代替最初始的模組，兩者交換替代，做爲一種擾動方式。

在上面第 4 點和第 5 點的擾動方式，和其它比較不同的是，它不是藉由移動或改變其方向，而是利用轉動模組，讓其長寬變動，來尋找出最佳化的解。

肆、我們的演算法

在本研究中，我們使用 CBL 表示法，與模擬退火（SA, simulated annealing）演算法，來發展出我們的演算法，用來解決積體電路在進行平面規劃上之叢聚限制（Clustering Constraints）問題。然而運用模擬退火法與 CBL 表示法，來解決叢聚限制（Clustering Constraints）的問題，要得到最佳解，就需要注意鄰近解與合理解、叢聚的限制式、成本函數，以及模擬退火演算法之流程，而更詳盡地介紹如下所示。

4.1 鄰近解與合理解

在模擬退火演算法之中，有一個重要的流程，就是要擾動表示法之中的參數，利用這種擾動的方式，產生一個鄰近解，然後再進行判斷，比較這個新產生的鄰近解，是否有比原來的解更好，如果有就直接接受，但是若沒有的話，還有一個機制會以機率的方式，來決定要不要接受這個新解，不過隨著溫度的降低，這個機率也會跟著降低，以致求到最佳解，但是 CBL 在進行擾動的時候，可能所產生的鄰近解，並不是一個合理解，所以若其鄰近解，不是一個合理解，我們想了一些機制，使其轉換成合理解，以確保每一次擾動皆能產生效果。

4.1.1 CBL 產生鄰近解的擾動方式

在本研究對 CBL 表示法的擾動方式，我們利用了 CBL 的三組參數（S, L, T），以及模組轉向 90 度，這四種的擾動方式，來產生鄰近的解，而其擾動方式，如下所示：

1.模組位置的擾動：

利用隨機亂數，任意取在平面規劃空間中的任兩個模組，也就是說在 S 參數中的任兩個模組，再將這兩個模組互換，但是不改變模組的加入方向與 T 接點的數目，以利鄰近解之產生。

如圖 9 所示，假定由隨機亂數，選定兩個模組 "b" 與 "d" ，這兩個模組原來平面規劃空間中，在兩個不同的位置，但是爲了要擾動產生鄰近解，我們將兩個模組的位置互換，但是並不改變 CBL 另外兩個參數 L 與 T，以確保產生的解是鄰近解。

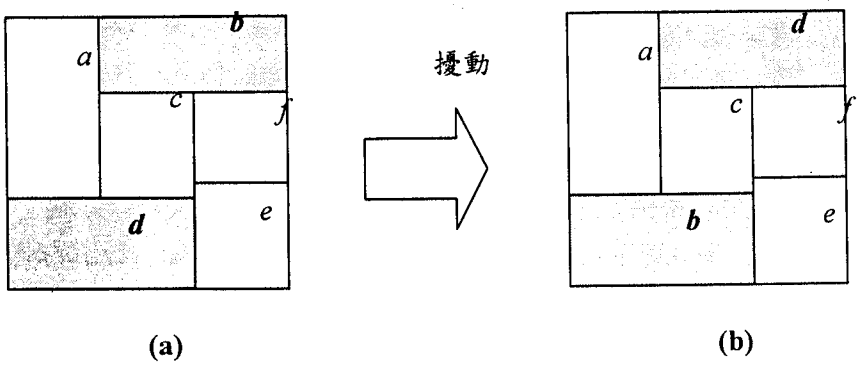


圖 9: CBL 模組位置的擾動圖 (a)擾動 "*b*" 和 "*d*" 模組之前；(b)擾動 "*b*" 和 "*d*" 模組之後

2.模組方向的擾動：

利用隨機亂數，選定一個模組，若是這個模組，原來是垂直方向加入平面規劃空間（L=0），則把它改成水平方向加入規劃空間（L=1）；相反的若是原來爲水平方向加入規劃空間（L=1），則把它改成垂直方向加入平面規劃空間（L=0），利用這種方式來擾動產生鄰近解，但是這種擾動方式，可能會產生不合理解，再不合理解部份，以下會再詳盡說明。

如圖 10 所示，假設隨機亂數取到 "*b*" 模組，原來的 "*b*" 模組是垂直方向加入平面規劃空間（L=0），但是我們爲了要擾動產生鄰近解，則把 "*b*" 模組改成水平方向加入規劃空間（L=1），但是並不改變 CBL 其它的 S 與 T 兩個參數，以確保產生的解是鄰近解。

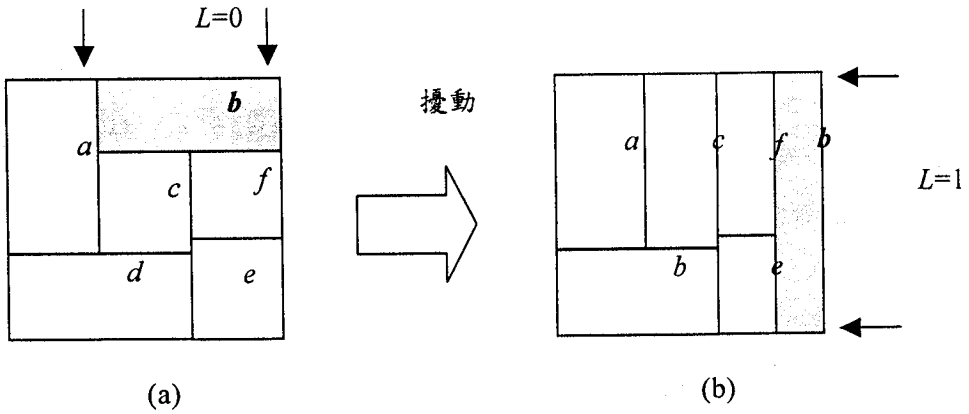


圖 10: CBL 模組方向的擾動圖 (a) 模組 "*b*" 方向擾動前；(b) 模組 "*b*" 方向擾動後

3.T 接點數目的擾動：

利用隨機亂數，選定一個模組，再擾動這個模組與平面規劃空間接觸切線之 T 型接點的數目，就好比原來如果這個切線上沒有任何的 T 型接點，那麼就把它改變爲有 T 型接點；相反的，如果這個切線上已經有 T 型接點，那麼就把它改變成沒有 T 型接點，利用這種方式來擾動產生鄰近解，和模組方向的擾動方式一樣，可能會產生不合理解，不合理解部份，會再以下做詳盡說明。

如圖 11 所示，假設隨機亂數取到 "*b*" 模組，那麼就先判斷這個模組與平面規劃空間接觸的 T 型接點的數目，我們可以知道 "*b*" 模組共有一個 T 型的接點，也就是在這個參數中 T=10，所以爲了擾動產生鄰近解，我們把 "*b*" 模組的 T 這個參數，改變成 T=0，也就是與切線間沒有任何 T 型的接點，但是並不改變其它兩個參數 S 與 L，以確保產生的解是鄰近解。

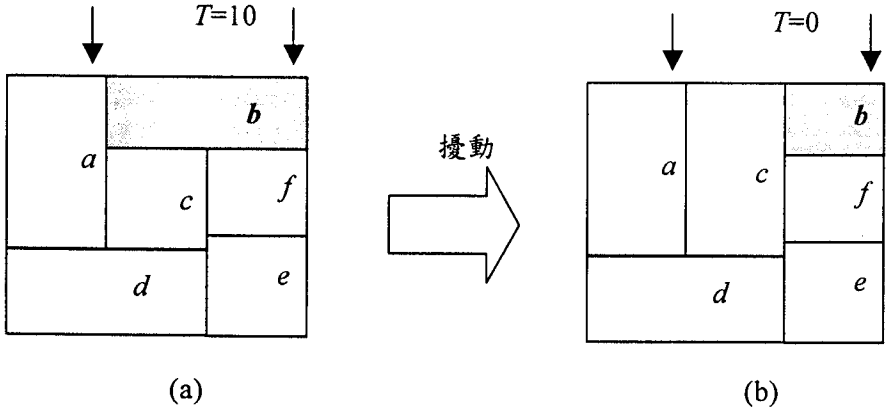


圖 11: CBL 模組 T 接點數目的擾動圖 (a) 模組 "*b*" 的 T 接點擾動前；(b) 模組 "*b*" 的 T 接點擾動後

4.模組轉向 **90** 度的擾動方式：

利用隨機亂數，選定一個模組，然後再將此模組轉向 90 度，其做法是將此模組的長寬互換，利用這種擾動的方式，求一個新的鄰近解。

4.1.2 合理解

在上述四種擾動方式中，分別爲模組位置的擾動、模組方向的擾動、T 型接點數目的擾動，以及模組轉向 90 度擾動四種。模組位置擾動與模組轉向 90 度擾動方式，不會產生不合理解的情況發生；但是模組方向的擾動與 T 型接點數目的擾動二種，則會有不合理的情況發生，如下所示：

1.擾動模組方向會產生的不合理解：

我們在擾動模組方向之後，也就是水平方向改變成垂直方向，或者是水平方向改變成垂直方向加入平面規劃空間時，若是在 CBL 中這個模組的 T=01，代表在原來方向的切線有一個以上的 T 型接點，但是在另外一個方向的切線，卻是只有一個 T 型接點，那麼在這裡就會產生不合理性，若發生這種情況，我們爲了確保每次都能正確有效的擾動，所以如果發生擾動方向後，其模組欲加入的 T 型接點數，大於加入切線上之 T 型接點，在這裡我們會把模組的 T 型接點數，減至小於切線上之 T 型接點數，以致確保在每一次擾動，都能得到一個正確的鄰近解，如圖 12(a)所示，原本垂直加入模組，後來改變成水平加入模組，但是在垂直加入時，壓入二個 T 型接點，但是改成水平的時候，卻因爲右邊緣的 T 型接點只有一個，所以產生不合理解，爲了讓擾動動作確實，我們解決的方式就是減少壓入的 T 型接點，讓其擾動產生的解合理，以達擾動之目的。

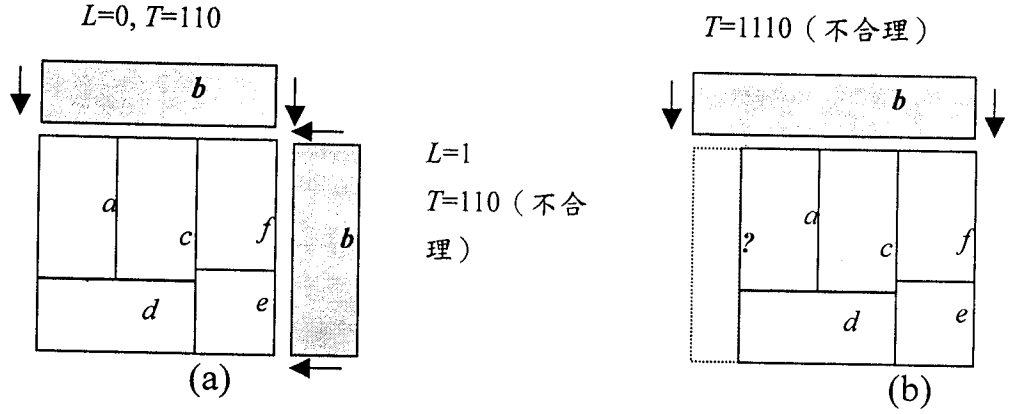


圖 12: CBL 模組不合理之擾動圖 (a) 擾動 L 產生之不合理解；(b) 擾動 T 產生之不合理解

2.擾動 T 型接點數目會產生的不合理解：

同樣的，若是我們擾動這個 CBL 中的 T 這個參數，也可能會產生不合理解，這道理是相同的，在擾動 T 型接點數目之後，其模組欲加入的 T 型接點數，大於加入切線上之 T 型接點，在這裡和處理上述的不合理情形相同，我們會把模組的 T 型接點數，減至小於切線上之 T 型接點數，以致於確保每一次擾動，都能得到一個正確的鄰近解，如例圖 12(b)所示，原本 "b" 模組如 12(a)在加入的時候是壓入二個 T 型接點，但是在我們擾動之後，變成了要壓入三個 T 型接點，可是在欲加入方向之 T 型接點數只有兩個，所以在這裡爲了擾動之確實，因此我們把欲壓入的 T 型接點數目減少，以達擾動之目的。

4.2 叢聚的限制式

在積體電路上，有些元件必須設計在鄰近的位置上，這也是本研究所要利用 CBL 表示法，來表示並滿足業叢聚限制，因此除了上述擾動的結果必須是合理的鄰近解之外，還是滿足叢聚的限制式，才是我們所要的擾動之後的合理鄰近解。

爲了判斷某一個模組和必須設計在一起的模組，在 CBL 表示法之中，是否有相鄰，以致於滿足叢聚的限制，因此我們必須記錄 CBL 表示法中，其任一個模組，的上、下、左，以及右邊的兩條水平與兩條垂直切線，所有相鄰的模組，並把這四個方向切線所相鄰的模組，分別記錄在四個參數之中。

其叢聚限制式如圖 12 之演算法所示，adjacent(S, L, T)演算法有三個參數，分別爲 S、L，以及 T 三個參數，在圖 12 中，2-4 行中分別對三個參數做定義，S 爲每一個 s_i 模組之集合；L 爲每一個 s_i 要加入時，所要依據的方向集合，其方向值有 0 和 1 兩種，“0”代表 s_i 是垂直加入，而“1”代表 si 是水平加入；T 爲每一個 s_i 要加入時，所要依據的 T 型接點相鄰數，其方向有 0、10 與 110，而第一個 0 代表沒有與任何 T 型接點相鄰，但是後兩者中的 1 代表與幾個 T 型接點相鄰，並用 0 來做分隔。在第 5 行的時候，把我們要記錄的上、下、左、右四個陣列清爲空集合（s_j_Top=null; s_i_Bottom= null; s_i_Left= null; s_j_Right = null;），以利程式中之重新記錄。

如第 6 行所示，程式是對 S 集合中，每一個 s_i 來做，其構加分成二大部份，第一部份爲第 7-16 行，模組爲垂直方向加入時之程序，第二部份爲 17-26 行，模組爲水平力入時之程序，以下分別對這二大部份做更請盡的說明。

第一部份，爲模組加入時，以垂直方向加入（l_i=0），如圖 10(a)所示，所以在第 8 行的時候，我們要判斷其加入時，共與幾個 T 型接點相鄰，這影響到這個加入的新偶角（corner block），與幾個上邊緣的模組相鄰，所以會依據 T 型接點數與模組的寬，來判斷執行幾次迴圈，而 9-11 行就是分別記錄新模組的下邊緣，與上邊緣模組的上邊緣，之間的相鄰記錄。在加入之後，這一個新加入的偶角，會與其左邊的模組有左右關係，而 12-16 就是處理與左邊模組的左右關係，但是在最後這個新加入的偶角模組，除了會影響到上邊緣的排列外，也會影響到右邊緣模組的排列，所以我們除了把新偶角模組放入上邊緣堆疊（Top_Stack）外，也要把這個新加入的偶角模組放入右邊緣的堆疊（Right_Stack）中，以利爾後模組若是以水平方向加入時之處理。

第二部份，爲模組加入時，以水平方向加入（l_i=1），如圖 10(b)所示，在程序上和第一部份非常類似，只是在記錄的方向由垂直的上下方向，改水平的左右方向，所以在第 18 行的時候，我們要判斷其加入時，共與幾個 T 型接點相鄰，這影響到這個加入的新偶角（corner block），與幾個右邊緣的模組相鄰，所以會依據 T 型接點數與模組的高，來

判斷執行幾次迴圈，而 19-21 行

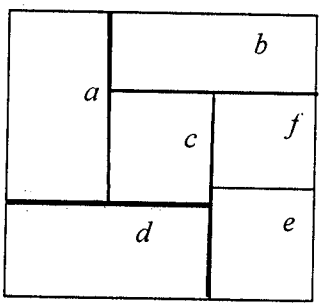
```
1.Algorithm Adjacent(S, L, T)
2.  S={S1, S2, …, Sn}, where si is each rectangular module
3.  L={l1, l2, …., ln}, where li is each rectangular module inserted according to its orientation. (li=1 or li=0)
4.  T={t1, t2, …, tn}, where ti is each rectangular module inserted according to its T-junctions. (ti=0 , ti=01 or ti=011)
5.   Si_TOP=NULL; Si_BOTTOM=NULL; Si_LEFT=NULL; Sj_RIGHT=NULL
6.   For each block Si in S do
7      if li=0 then     /* li=0, vertical orientation*/
8.                for n=0 to ti
                          9. sj←POP and delete from
                          Top_Stack
10                          si_Bottom←si_Bottom+sj
11                    sj_Top←sj_Top+si
12                    sj←Pop from Top_Stack
13                    si_Left←si_Left+sj
14                    sj_Right←sj_Right +si
15                    PUSH sj from Top_Stack
16                    PUSH si from Top_Stack and Right_Stack
17.               else   /* li=1, horizontal orientation*/
18.                   for n=0 to ti
                          19.        sj←POP and delete from
                          Right_Stack
20.                         si_Left←si_Left+sj
21.                         sj_Right←sj_Right+si
22.                   sj←POP from Right_Stack
23.                   si_Bottom←si_Bottom+sj
24.                   sj_Top←sj_Top +si
25.                   PUSH sj from Right_Stack
26.                   PUSH si from Right_Stack and Top_Stack
```

圖 13: 叢聚限制演算法

就是分別記錄新模組的左邊緣，與右邊緣模組的右邊緣，之間的相鄰記錄。在加入之後，這一個新加入的偶角，會與其下邊的模組有上下關係，而 24-26 就是處理與下邊模組的上下關係，但是在最後這個新加入的偶角模組，除了會影響到右邊緣的排列外，也會影響到上邊緣模組的排列，所以我們除了把新偶角模組放入右邊緣堆疊外，也要把這個新加入的偶角模組放入上邊緣的堆疊中，以利爾後模組若是以垂直方向加入時之處理。

如圖 13 所示，將模組四邊的切線，每一個切線所相鄰的模組，全部記錄在圖 13 的表中，這樣的話我們就可以清楚的找到每個模組，是否有和某個模組相鄰在一起，例如模組 "*c*" ，我們從圖 13 中，就可以很快的找出其相鄰的模組，與 "*c*" 在上切線相鄰的有 "*b*" ，在下切線相鄰的有 "*d*" ，在左切線相鄰的有 "*a*" ，以及在右切線相鄰的有 "*e*" 與 "*f*" 。我們用一個四維陣列 abut(*T*, *B*, *L*, *R*)的方式，把圖 13 內的表中資訊，儲存在四維陣列中，以列判斷模組和模組間是否有符合叢聚的關係，以達到我們所以的合理鄰近解，當然若沒有符合叢聚的關係，那我們一樣要拋棄這個不合理的鄰近解，再從新尋找一個合理的鄰近解。

	上	下	左	右
a	-	*d*	-	*b, c*
b	-	*c, f*	*a*	-
c	*b*	*d*	*a*	*e, f*
d	*a, c*	-	-	*e*
e	*f*	-	*c, d*	-
f	*b*	*e*	*c*	-



-：代表沒有任何模組相鄰

圖 14: CBL 各模組間的相鄰模組

4.3 模擬退火演算法之理論與流程設計

在傳統的最佳解方法中，大部份都是以梯度爲基礎（Gradient based）之局部插尋法，而這種梯度爲基礎的尋找法是利用目標函數（objective function）與限制函數（constraint function）的梯度關係，來決定其搜尋的方向，但是這種搜尋方式，屬於非線性規劃（Non-linear Programming）之方法，所以基本上這種搜尋法，僅能求得距起始設計點，最近的一相對最佳解，也就是最後的解，可能會落在局部的最佳解。

但是在演擬退火法（SA, simulated annealing），在設計變數的種類，以及在設計問題的設計空間形式上，並沒有做任何的限制，而且在設計若是在初始狀態（溫度高的時候），會有較大的機率接受能量變大的設計，因此有跳脫局部最佳解的能力，所以模擬退火法，是一種屬於全域搜尋法，如圖 14 所示，在使用上只須要利用限制函數與目標函數，就可有效的使用在各種不同的最佳化問題中，求得想要的最佳解。

4.3.1 模擬退火法之基本理論

摸擬退火法的基本理論，是發展自統計力學方面而來，它之所以能應用於解全域最佳解的問題，源自於在物理現象中，金屬物質在退火的過程，與在解全域最佳解問題，兩者之間的求解過程非常的相似。在進行退火處理的金屬物質，會先把金屬加至高溫，使其金屬結構中的原子，其這些原子間的金屬鍵變弱，讓這些金屬原子能自由的運動，當然在這些金屬原子移動後，會重新的排列組合，而金屬的結構也會隨著增加或減少其位能，但是金屬原子在的自由運動，大部份會尋求最低能量之狀態。而隨著溫度的下降，金屬原子的自由運動也會隨之下降，而使金屬呈現最低能量狀態，但是有一要件，就是退火度下降速率不能太快。基本上模擬退火演算法，和上述的金屬退火的情況非常的類似。模擬退火演算法，在解最佳解的問題上，會有一個合理解的區域，在這個區域內有很多組的合理解，利用各種表示法的擾動方式，產生一個鄰近的合理解，這個解可能會比目標函數小，或者大；比目標函數小的話，就會接受這個新解，但是，初始在溫度高的時候，即使擾動後的目標函數值增加，會有一個機率值，來判定接受與否，但是隨著溫度的下降，慢慢這個接受的機率就會減低，使最後求得的結果爲全域最佳解。

4.3.2 模擬退火法之流程

本研究在利用模擬退火法時，其程式流程步驟如下所示：

1. 流程一：參數設定
2. 流程二：產生初始解
3. 流程三：計算目標函數
4. 流程四：產生合理新解
5. 流程五：計算新解目標函數
6. 流程六：進入 Metropolis 抽樣演算法
7. 流程七：是否滿足降溫條件
8. 流程八：將最佳解的結果輸出

4.3.3 模擬退火法之參數

模擬退火法主要是發展於物理現象－金屬物質退火過程的方式而來，在物理現象，會隨著各種不同的金屬材料，而在退火溫度的範圍，以及退火過程也會有所相異，因此在模擬退火演算法中，也要設定初始溫度來代表金屬退中過程中，加至最高溫的溫度，另外，也要設計溫度的下降速率，來表示金屬退火過程中，溫度的下降，當溫度完全下降時，也就是代表退火程序完成。

經過以上的論述，可以知道模擬退火演算法，最基本的參數爲初始溫度、結束溫度，以及溫度下降速率。另外模擬退火演算法，仍然須要一個產生新的合理解時，與舊的合理解做比較的機制，這個機制是，在新的鄰近合理解，比舊的合理解低時，則接受新的鄰近合理解；相反的，若新的鄰近合理解，沒有比舊的合理解來的低，那麼就要透過 $\exp(-\Delta E/KT)$所產生的數值，以機率來判定能不能接受這個較差的新鄰近合理解，當然這個接受的機率，會隨著溫度的下降，慢慢的減低。而以下就是我們在運用模擬退火演算法各參數之變數。

1. Ts：初始溫度，也就是金屬加熱中的最高溫度
2. Te：結束溫度
3. ζ：降溫係數
4. L：擾動次數
5. T：最大疊代次數
6. K：波茲曼常數（在本研究令 K=1，所以模擬退火過程中，不會出現）

各參數的設定值如下表所示

表 2: 模擬退火演算法參數設計表

編號	參數	參數名稱	參數值
1	Ts	初始溫度	1000
2	*Te*	結束溫度	1
3	ζ	降溫係數	0.95
4	*L*	擾動次數	1000
5	*T*	最大疊代次數	135
6	*K*	波茲曼常數	1

伍、實驗結果

本研究的實驗樣本，來自於 MCNC 所建構的樣本，其主要的樣本有 xerox、hp、apet、ami33，以及 ami49 五個實驗樣本。這些樣本有不同的模組個數與面積，其樣本的特徵描述，如表 3 所示。

而我們的實驗的方式，分成單一叢聚集合，以及多個叢聚集合兩種方式來進行。在參數方面，經過我們多次的實驗結果發現模擬退火法的初始溫度設定在 1000 度，每個溫度擾動次數設在模組個數的 400 倍，當擾動完成後每次溫度下降 0.95 的倍數，再重覆模組個數的 400 倍次之擾動，直到低於 1 度，或者每個溫度成本下降幅度低於 0.05%，則停止程式。在實驗平台方面，我們的演算法是以 Visual Basic 6 來撰寫，作業環境則是在 Windows 2000 的作業系統下，CPU 是 Pentium III　300，以及記憶體 128M Bytes 上的機器來進行實驗。

表 3: MCNC 建構的樣本特徵描述表

樣本名稱	模組個數	樣本面積(mm^2)
xerox	10	19.350296
hp	11	8.830584
apet	9	46.561628
ami33	33	1.156449
ami49	49	35.445424

首先，在單一叢聚集合的方式，是從每一個實驗樣本中，以隨機的方式，選取 20% 的模組，以這些選取的模組，當做叢聚限制的模組集合，如表 3 所示，就好比在 ami33 中選取 33 個模組中的 20%，也就是 7 個模組來當作叢聚限制之集合；而在 ami49 中選取 49 個模組中的 20%，也就是 10 個模組來當作叢聚限制之集合。對於每一個樣本，我們做三次的實驗，而每次的實驗，我們會重新選取不同模組當作叢聚集合，其實驗的結果如表 4 所示。

表 4: 單一叢聚限制的實驗結果

樣本	叢聚集合	模組個數	叢聚個數	閒置空間(%)		時間(分)
				個別	平均	
xerox	c1	10	2	5.315	5.845	25
	c2	10	2	5.693		22
	c3	10	2	6.529		23
hp	c1	11	3	6.072	6.312	30
	c2	11	3	6.432		35
	c3	11	3	6.432		32
apet	c1	9	2	2.077	2.077	16
	c2	9	2	2.077		18
	c3	9	2	2.077		16
ami33	c1	33	7	8.533	8.771	60
	c2	33	7	8.787		50
	c3	33	7	8.995		55
ami49	c1	49	10	8.575	8.783	90
	c2	49	10	8.805		85
	c3	49	10	8.969		92

在表 4 之中，我們分別對每一個樣本做三次實驗，然後在把三次的實驗結果做平均，其 xerox 在單一叢聚，叢聚的模組個數爲 2 的限制下，平均的閒置空間是 5.845%；hp 在單一叢聚，叢聚的模組個數爲 3 的限制下，平均的閒置空間是 6.312%；apet 在單一叢聚，叢聚的模組個數爲 2 的限制下，平均閒置空間是 2.077%，ami33 在單一叢聚，叢聚的模組個數爲 7 的限制下，平均閒置空間是 8.771%，ami49 在單一叢聚，叢聚的模組個數爲 10 的限制下，平均閒置空間是 8.783%；這些樣本經過實驗，都可以得到很好的結果。

其次，在第二個實驗中，我們則選取多個叢聚限制集合，其做法是在前三個實驗樣本中，由於 xerox、hp，以及 apet 的樣本數較少，所以只選出兩個叢聚集合做實驗，叢聚集合是以隨機的方式選取不同的叢聚集合模組，然後進行三次的實驗；後二個樣本 ami33，以及 ami49，我們分別選取有 3 個叢聚集合、4 個叢聚集合，以及 5 個的叢聚集合的多個叢聚限制，來作爲實驗的叢聚限制，其實驗結果如表 5 所示。

表 5: 多個叢聚限制的實驗結果

樣本	叢聚集合	模組個數	叢聚個數	閒置空間(%)		時間(分)
				個別	平均	
xerox	m1	10	2(2, 2)	6.529	6.658	26
	m2	10	2(2, 2)	6.637		30
	m3	10	2(2, 2)	6.81		28
hp	m1	10	2(2, 2)	8.429	8.599	38
	m2	10	2(2, 2)	8.94		39
	m3	11	2(2, 2)	8.429		36
apet	m1	9	2(2, 2)	2.077	2.077	20
	m2	9	2(2, 2)	2.077		18
	m3	9	2(2, 2)	2.077		15
ami33	m1	33	3(4, 4, 3)	9.063	9.197	65
	m2	33	4(3, 3, 3, 2)	9.122		70
	m3	33	5(3, 2, 2, 2, 2)	9.406		67
ami49	m1	49	3(6, 5, 5)	9.035	9.271	100
	m2	49	4(4, 4, 4, 4)	9.197		96
	m3	49	5(4, 3, 3, 3, 3)	9.538		95

在表 5 之中，我們分別對每一個樣本做三次實驗，然後在把三次的實驗結果做平均，其 xerox 在多個叢聚，叢聚的個數爲 2 的限制下，平均的閒置空間是 6.658%；hp 在多個叢聚，叢聚個數爲 2 的限制下，平均的閒置空間是 8.599%；apet 在多個叢聚，叢聚個數爲 2 的限制下，平均閒置空間是 2.077%，ami33 在多個叢聚，叢聚個數分別爲 3、4、5 的限制下，平均閒置空間是 9.197%，ami49 在多個叢聚，叢聚個數分別爲 3、4、5 的限制下，平均閒置空間是 9.271%；這些樣本經過實驗，都可以得到很好的結果。

陸、結論與未來發展

6.1 結論

本研究利用偶角模組列（CBL），來解決在叢聚限制下的不可切割結構的平面規劃問題，以 MCNC 所建構的 xerox、hp、apet、ami33，以及 ami49 五個樣本爲例，配合模擬退火法的處理流程來求解，在求解的過程中，我們運用多種不同的擾動方式，使擾動更加的確實，而在擾動的方式中，有兩種的擾動方式，會產生不合理解，在這方面我們亦運用了一些機制，使我們每次遇到不合理解的時候，皆能使它轉換成合理解，使模擬退火法的擾動不會受到不合理解的影響，降低了擾動的成效，所以運用此方式可以有效地提昇處理程序的效率；接下來以我們本論文的叢聚限制演算法，來判斷所得到的解，是否符合叢聚限制的條件，在這裡我們分別做了兩個實驗來驗証，一個是單一叢聚限制，另外一個是多個叢聚限制，最後我們實驗証明，我們所得到的解，其閒置空間都在 9% 以下，因此可以証明利用偶角模組列（CBL）表示法可以求到一個很好的解。

平面規劃問題已經被証明是一個 NP 問題，而且已經有很多學者利用不同的表示法來解這個問題，但是在叢聚限制下的平面規劃問題上，本論文是最先利用可以表示不可分割結構的偶角模組列（CBL）表示法，來解決這個 NP 問題，並証明有很好之成效。

6.2 未來發展

在未來發展方面，因爲本論文實驗的模組都是固定模組，因此可以在模組上加入彈性模組，或者利用我們的叢聚限制演算法，來處理類似的限制問題，相信也會有很好的結果。

柒、參考文獻

1. Y.-C. Chang, Y.-W. Chang, G.-M. Wu and S.-W. Wu, "B*-trees: A New Representation for Non-Slicing Floorplans," in Proceedings of the 37th ACM/IEEE Design Automation Conference, pp. 458-463, 2000.
2. K. Fujiyosi and H. Murata, "Arbitrary Convex and Concave Rectilinear Block Packing," in Proceedings International Symposium on Physical Design, pp. 103-110, 1999.
3. P. N. Guo, C. K. Cheng, and T. Yoshimura, " An O-tree Representation of Non-Slicing Floorplan and Its Applications," in Proceedings of the 36th ACM/IEEE Design Automation Conference, pp. 268-273, 1999.

4. Xianlong Hong, Sheqin Dong, Gang Huang, Yuchun Ma, Yici Cai, Chung-Kuan Cheng and Jun Gu, "A Non-Slicing Floorplanning Algorithm Using Corner Block List Topological Representation," in Proceedings ASP-DAC, pp. 833-836, 2000.
5. Xianlong Hong, Sheqin Dong, Gang Huang, Yuchun Ma, Yici Cai, Chung-Kuan Cheng and Jun Gu, "Corner Block List: An Effective and Efficient Topological Representation of Non-Slicing Floorplan," in Proceedings of ACM/IEEE Design Automation Conference (DAC-2000), pp. 8-12, 2000.
6. M. Z. W. Kang and W. W. M. Dai, "General Floorplanning with L-Shaped, T-Shaped and Soft Blocks Based on Bounded Slicing Grid Structure," in Proceedings ASP-DAC, pp. 265-270, 1997.
7. M. Z. W. Kang and W. W. M. Dai, "Topology Constrained Rectilinear Block Packing for Layout Reuse," in Proceedings International Symposium on Physical Design, pp. 179-186, 1998.
8. M. Z. W. Kang and W. W. M. Dai, "Arbitrary Rectilinear Block Packing Based on Sequence-Pair," in Proceedings International Conf. on Computer-Aided Design, pp. 259-266, 1998.
9. S. Kirkpatrick et al. "Optimization by simulated annealing," Science, Vol. 220, pp. 671-680, 1983.
10. J.-M. Lin and Y.-W. Chang, "TCG: A Transitive Closure Graph Based Representation for Non-Slicing Floorplans," in Proceedings of ACM/IEEE Design Automation Conference (DAC-2001), pp. 764-769, 2001.
11. Yuchun Ma, Sheqin Dong, Xianlong Hong, Yici Cai, Chung-Kuan Cheng and Jun Gu, "VLSI Floorplanning with Boundary Constraints Based on Corner Block List" in Proceedings ASP-DAC, pp. 509-514, 2001.
12. H. Murata, K. Fujiyoshi and M. Kanedo, "VLSI/PCB Placement with Obstacles Based on Sequence-Pair," in Proceedings International Symposium on Physical Design, pp. 26-31, 1997.
13. H. Murata and E. S. Kuh, "Sequence-pair Based Placement Method for Hard/Soft/Pre-Placed Modules," in Proceedings International Symposium on Physical Design, pp. 167-172, 1998.
14. S. Nakatake, K. Fujiyoshi, H. Murata and Y. Kajitani, "Module Placement on BSG-Structure and IC Layout Applications," in Proceedings of the IEEE/ACM

International Conference on Computer-Aided Design, pp. 484-493, IEEE Computer Society Press, Nov. pp. 10-14 1996.

15.S. Nakatake, M. Furuya and Y. Kajitani, "Module Placement on BSG-Structure with Pre-Placed Modules and Rectilinear Modules," in Proceedings ASP-DAC, pp. 571-576, 1998.

16.R. H. J. M. Otten, "Automatic Floorplan Design," in Proceedings of the 19th IEEE/ACM International Conference on Computer-Aided Design, pp. 261-267, 1982.

17.D. F. Wong and C. L. Liu, "A New Algorithm for Floorplan Design," in Proceedings of the 23rd ACM/IEEE Design Automation Conference, pp. 101-107, IEEE Computer Society Press, June 1986.

18.J. Xu, P. N. Guo and C. K. Cheng, "Rectilinear Block Placement Using Sequence-Pair," in Proceedings International Symposium on Physical Design, pp. 173-178, 1998.

19.F. Y. Young and D. F. Wong, "Slicing Floorplans with Range Constraint," in Proceedings International Symposium on Physical Design, pp. 97-102, 1999.

20.F. Y. Young and D. F. Wong, "Slicing Floorplan with Boundary Constraint," in Proceedings ASP-DAC, pp. 17-20, 1999.

21.W. S. Yuen and F. Y. Young, "Slicing Floorplan with Clustering Constraints," IEEE Asia South Pacific Design Automation Conference, pp. 503-508, 2001.