多組史坦納樹演算法

A Multiple Steiner Trees Routing Algorithm

詹景裕	李明哲	李欣冀	呂紹偉
國立臺北大學	銘傳大學	國立臺灣海洋大學	國立臺灣海洋大學
電機工程研究所	資訊傳播工程研究所	電機工程研究所	電機工程研究所
gejan@mail.ntpu.edu.tw	leemc@mcu.edu.tw	M96530018@mail.ntou.edu.tw	B0119@mail.ntou.edu.tw

***摘要*—**多組史坦納樹連結演算法是指在一個 X 架構網路中，利用垂直(vertical segments)、水平(horizontal segments)以及斜線(oblique segments)線段，將各組節點以最短的連接路徑相互連接後所形成多組史坦納樹。史坦納樹演算法在時間複雜度上屬於 NP-complete [6]、[8]，所以目前多組演算法之研究報告尚不多見。多組史坦納樹可應用於電子電路中，如 VDD、GND 等。本文利用 2D 八向式史坦納樹之試誤型演算法來處理各組節點連接問題，然後利用各節點與關鍵節點的距離來作一個依序連結的順序，以求得多組史坦納樹。其時間複雜度為 $O(p^2N)$、空間複雜度為 $O(pN)$，其中 p 為欲連結節點數目，q 為組數，N 為二維矩陣中節點的數目。

***關鍵詞*—**多組，史坦納樹，繞線

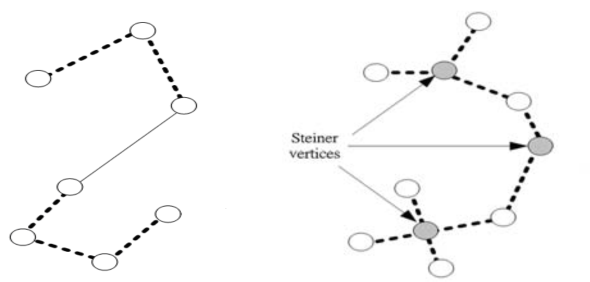
Abstract - In an X-Grid, a Multiple Steiner Trees Routing algorithm connects each group node via vertical, horizontal and oblique segments. So far the researches in this topic are few since the problem is NP-complete. The Multiple Steiner Trees Routing algorithm can be used to solve many well-known problems, such as the VDD-GND routing problem in VLSI. This paper presents a novel Multiple Steiner Trees Routing algorithm, which adopts 4-geometry Steiner Trees Routing Heuristic algorithm to connect each group. The space and time complexities are $O(pN)$ and $O(p^2N)$, respectively, where N is the number of nodes in the 2-D matrix and p is the number of terminal nodes, q is the number of groups.

Keywords: Multiple, Steiner Trees, Routing

一、緒論

網路中的史坦納樹問題(Steiner problem in networks; SPN)是指在歐式平面(Euclidean plane)上，找出可展開(spanning)給定節點集合(set of vertices)的最短路徑網路。史坦納樹問題與最小伸展樹(Minimal Spanning Tree; MST)的問題相當類似，不同點為最小伸展樹中所構成路徑的邊集合(set of edges)中的邊(edges)只可由節點(vertices)連接至節點，而史坦納樹路徑的邊除了可由節點連接至節點的邊構成外，也可由其它任意自由節點來當作輔助節點，使連接整個節點後的總長度最小，因而稱為史坦納樹節點(Steiner vertices)。圖一中的圓點表示欲相互連接之節點，實線及虛線則表示連接該節點集合的邊集合。圖一(a)表示最小伸展樹所允許的路徑連接方法將給定的節點連接後所形成之連接圖(connected graph)；而圖一(b)則為給定與圖一(a)相同的節點集合，以最小史坦納樹定義的路徑形態所得出之連接圖形。

比較圖一虛線部份，可以明顯看出最小伸展樹與史坦納樹之差異。最小伸展樹只可由節點連結至節點的路徑形態，而史坦納樹則可由



(a) Minimal Spanning Tree　　(b) Steiner Minimal Tree

圖一、最小伸展樹與史坦納樹之差異示意圖

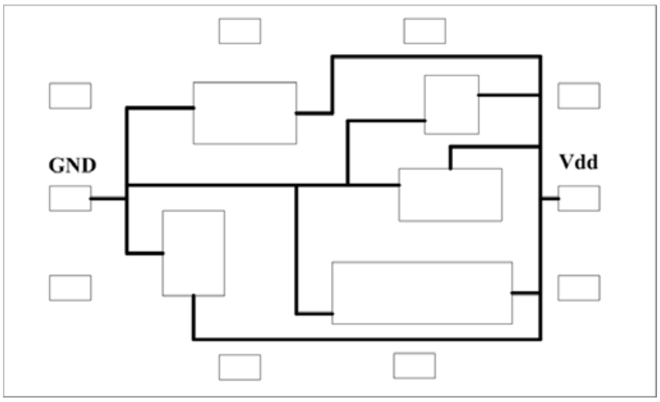
其它自由節點連結至現有路徑之路徑形態。

史坦納樹問題之正式的數學定義為：若給定一個具有$|V|$個節點的節點集合 V、$|E|$個邊的邊集合E以及成本函數(cost function) C: $E\rightarrow R$的無向網路圖 $G=(V, E, C)$，以及 V 的子集合 Z，Z 包含於 V，Z 為具有 p 個欲相互連接成一網路的節點所成的集合。則史坦納樹問題可以表示為:求解 G 的子網路 G_Z 使得在 Z 中的每一節點對(Z_i,Z_j)，$0<i, j\leq p$，均存在一條路徑，使 G_Z 總成本 $C(G_Z)$為最小。

G_Z 中共有兩種節點型態，包含於 Z 中的節點稱為 Z 節點(Z-vertices)，其餘的節點則稱為 S 節點(S-vertices)。子網路 G_Z 則稱為於 G 上對 Z 而言的最小史坦納網路。如圖一(b)中之史坦納節點即為 S 節點，其餘則為 Z 節點。

史坦納樹問題有許多確切型演算法(exact algorithm)已經被提出，如 Melzak [11], Cockayne [5], Boyce 及 Seery [4], Boyce [3]和 Winter [13]等。不過由於史坦納樹的問題太過於複雜，使得上述的確切型演算法只能在合理的時間內解決連接節點數 p 較少的問題，當連接節點數過多時，所需的處理時間將為天文數字而令人無法接受。Karp [8]和 Foulds [6]分別證明了史坦納樹問題是屬於 NP-complete。因此，為了解決較多的連接節點數，試誤型演算法(heuristic algorithm)變得相當重要。目前已有許多大量針對史坦納樹問題而提出的試誤型演算法[1][2]。其中獲得較佳結果且執行效率也較佳的史坦納樹試誤型演算法是由 Lin 等人所提出 [12][10]。

史坦納樹問題原先是由數學延伸而來的，近年來一直被應用在計算機科學領域中。它的應用範圍相當廣泛，如多重廣播路徑之找尋、印刷電路版(Printed Circuit Board, PCB)的電流網路連接網狀架構(mesh)上的容錯路徑，以及超大型積體電路內的路由路徑(VLSI、ULSI routing)找尋等，均為史坦納樹問題的應用領域。



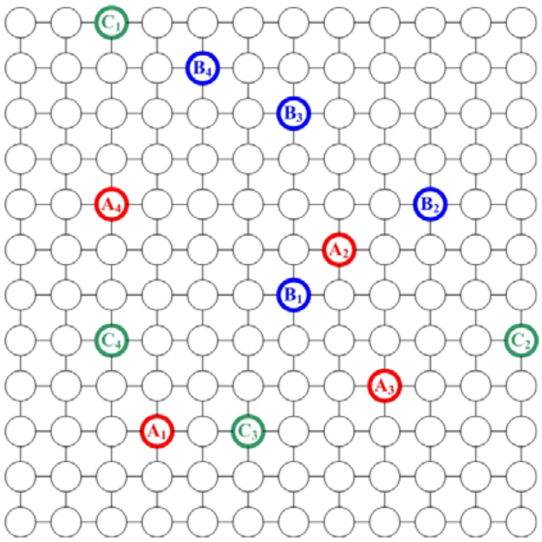
圖二、電源與接地繞線問題

過去數十年來，許多數學家、資訊科學領域的研究人員，均對史坦納樹問題投入不少心力，但是多組史坦納樹問題的研究報告尚不多見，大部分都是針對單組連結路徑來做改進，對於應用在電子電路上來說是相當不實用的。舉例而言，在一般電路板的應用上，可能不只單組線路，通常會有多組線路組合而成。圖二為電路板中常見的電源(Vdd)與接地(GND)配置圖。因為電源與接地線所承載的電流比其他的線路要多，線材本身也較寬，因此通常必須在同一層完成繞線。有鑑於此，多組史坦納樹連結在電子電路的設計上有其必要與重要性。

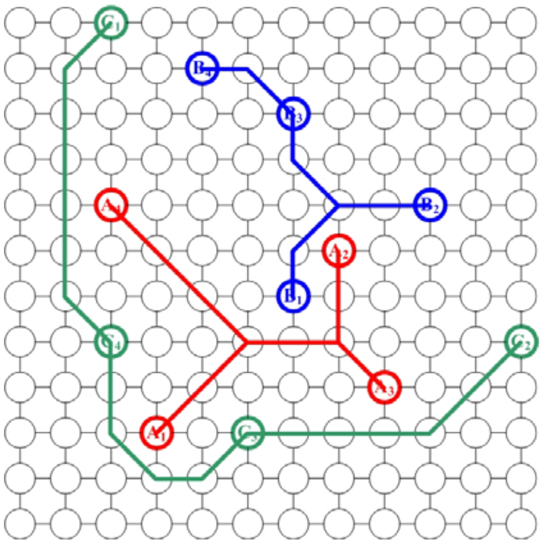
本文之後將介紹的部份說明如下：第二節將問題定義及介紹所需的變數名稱，第三節為多組史坦納樹演算法；演算法效能分析則安排在第四節，第五節則是簡單的範例，最後本文在第六節總結。

二、問題定義與變數介紹

多組史坦納樹演算法問題是由單組延伸出來，因為單組已經不適用在目前的 PCB 板上，而是要有多組以上之路徑才符合需求，所以才會提出多組史坦納樹演算法。圖三是問題定義，圖三(a)表示起始圖，有 A、B、C 三組節點，每組節點有 4 個節點，即 A1-A2-A3-A4、B1-B2-B3-B4、C1-C2-C3-C4。各組對間兩兩不得相交。



(a) 起始圖



(b) 完成圖

圖三、多組史坦納樹問題定義

為了方便索引及參考，本論文所使用之變數符號之定義如表一與表二所示。

三、多組史坦納樹演算法

過去 Lee 提出的最短路徑連接演算法 (簡稱 Lee 演算法) 是一個被廣泛運用的網格式空間資料結構演算法 [9]，在網格平面上給定起點、終點及若干障礙物，於起點至終點間存在一條最短且具有防碰撞障礙物之路徑集合，Lee 演算法可保證於 $O(N)$時間內找到該路徑集合中的其中一條最短路徑，其中 N 是網格平面上自由網格節點之數目(number of free cells)。最近 Jan 利用 Lee 演算法的觀念 [7]，被稱其為 Higher Geometry Maze Routers (HGMR) algorithm，將原本的平面四向發展到平面八向。

多組史坦納樹連結演算法是利用 HGMR 演算法中的洪泛程序來處理各組節點的等距離圖，並求得各組之距離累加圖，再求得各組之關鍵節點，接著利用排序觀念，將各組節點離各組的關鍵節點加以排序，排序第一的節點連結關鍵節點(路徑)，其它節點再從頭開始依序以上步驟，直至所有節點都連結完畢。

表一、空間資料結構說明

參數	資料型態	說明
$v_{i,j}$	整數二維陣列	vertices，儲存 $m \times n$ X 架構網路內的節點狀態，數值介於{0, 1, 2 }之間。數值為 0 時表示該節點為自由節點，為 1 表示為障礙節點，為 2 則表示欲相互連接的節點。其中(i, j)為二維陣列之索引值，$0 \leqq i \leqq m$，$0 \leqq j \leqq n$。
$D^{local}(k, l)$	整數二維陣列	distance for local Z vertices，儲存以第 k 組 Z 中第 l 個節點之位置為起點，經由 HGMR 演算法之洪氾程序後所得之等距離圖，$0 \leqq l < p$。$D^{local}(k, l)$陣列之大小為 $m \times n$。當以 $D^{local}_{i,j}(k, l)$表示時，表其為存取 $D^{local}(k, l)$上索引值(i, j)位置上之值，$0 \leqq i \leqq m$，$0 \leqq j \leqq n$。
$D^{global}(k)$	整數二維陣列	global distance，儲存網狀網路內各組所有 Z 節點之等距離累加圖。$D^{global}(k)$，陣列之大小為 $m \times n$。當以 $D^{global}_{i,j}(k)$表示時，表其為存取 $D^{global}(k)$上索引值為第 k 組(i, j)位置上之值，$0 \leqq i \leqq m$，$0 \leqq j \leqq n$。

表二、輔助運算之資料結構

變數	資料型態	說明
$LL^{z}_{k,l}$	Linked Lists	儲存記錄 $v_{i,j}$ 中值為 2 之節點，即 Z 節點。$LL^{z}_{k,l}$係指取出第 k 組第 l 個節點 $v_{i,j}$。
$e_{v_{i,j},\, v_{i',j'}}$	字串	edges，儲存由 $v_{i,j}$ 連結至 $v_{i',j'}$的邊。
$LL^{SMT}_{k,l}$	Linked Lists	為儲存計算後多組史坦納路徑的鏈結串列。其內之資料型態由邊 $e_{v_{i,j},\, v_{i',j'}}$所組成。$LL^{SMT}_{k,l}$係指取出第 k 組第 l 個邊 $e_{v_{i,j},\, v_{i',j'}}$。
index	整數變數	儲存計算時所用的索引值。
$*v_{critical}$	指標變數	儲存關鍵節點之位址。
$*v_{temp}$	指標變數	儲存計算時所用的暫存節點 $v_{i,j}$ 之位址。
temp	整數變數	記錄計算時所用的暫存值。
$*\underline{\underline{D}}^{critical}$	指標變數	為指向記錄 critical vertex 的等距離圖二維陣列 $D^{local}(c, k)$，c 為組別，k 為關鍵節點於 Z 中的索引值。當以$*\underline{\underline{D}}^{critical}_{i,j}$ 表示時，是表示存取$*\underline{\underline{D}}^{critical}$ 指標所指向之 $D^{local}(c, k)$上索引值為(i, j)位置上之值，其中 $0 \leqq i \leqq m$，$0 \leqq j \leqq n$。
$Len(l)$	整數變數	線路連結順序，$1 \leqq l < p$。
min	整數變數	儲存計算時所用的最小暫存值。

演算法：

BEGIN {多組史坦納樹連結演算法}

STEP 1 呼叫**初始化**副函式

STEP 2 計算距離

STEP 2.1 呼叫**計算等距離圖**副函式。利用 HGMR 洪氾演算法計算各節點個別之等距離圖

STEP 2.2 呼叫**等距離累加圖**副函式

STEP 2.3 呼叫**搜尋關鍵節點**副函式

STEP 2.4 呼叫**最短距離**副函式

STEP 3 計算多組史坦納樹

STEP 3.1 連結離關鍵節點(路徑)最近路徑

STEP 3.2 執行 STEP 2

STEP 4 結果輸出

輸出 $LL^{SMT}_{k,l}$，即為所求之多組史坦納樹。

END {多組史坦納樹連結演算法}

副函式 1 初始化:

BEGIN

依序搜尋 $v_{i,j}$ 為 Terminal node 之節點，則將該節點的陣列索引值對(i, j) 加至鏈結串列 $LL^{z}_{k,l}$末端。

新增一個新的等距離圖 $D^{local}_{i,j}(k, l)$，其中 k 代表組別，l 表其為第 l 個 Z 節點，$0 \leqq l < p$，$0 \leqq i \leqq m$，$0 \leqq j \leqq n$。

END

副函式 2 計算等距離圖:

BEGIN

STEP 1 設定 $index \leftarrow 0$。

STEP 2 由 $LL^{z}_{k,l}$中取出第 p 組中第 *index* 個節點 $v_{i,j}$。

STEP3 在 $D^{local}(index)$中，根據 $v_{i,j}$ 中座標對(i, j)之值，設定(i, j)位置上之 $D^{local}_{i,j}(index) = 0$ 。

STEP 4 在 $D^{local}(index)$中，設定座標對(i, j)位置上之節點為起點，經由 Jan 八向演算法中之洪氾階段演算法，計算由起點至各

節點的等距離圖，直至所有自由節點均被探索過為止。Jan 之洪氾階段演算法計算方式是四個四向連結的鄰格+1，四個對角的鄰格+$\sqrt{2}$ 網格的最短距離可在+1 和+$\sqrt{2}$ 兩者中取最小者。

STEP 5 $index = index + 1$，重複 STEP 2，直至 $LL^{z}_{k,l}$中的所有節點 $v_{i,j}$ 均被計算過。

END

副函式 3 等距累加圖:

BEGIN

STEP 1 設定 $index = 0$。

STEP 2 $D^{global}_{i,j}(k) = D^{local}_{i,j}(k) + D^{global}_{i,j}(index)$，$0 \leqq i \leqq m$，$0 \leqq j \leqq n$。

STEP 3 $index \leftarrow index + 1$，重複 STEP 2 直至所有的 $D^{local}(k, l)$均已被計算過。

END

副函式 4 搜尋關鍵節點:

BEGIN

線性搜尋 $LL^{z}_{k,l}$找出座標值(i, j)於 $D^{global}_{i,j}(k)$中值最小的各組關鍵節點：

STEP 1 設定 $index = 0$。

STEP 2 從 $LL^{z}_{k,l}$中取出第 k 組第 $index$ 個節點 $v_{i,j}$。
$min \leftarrow D^{global}_{i,j}(k)$，
$*v_{critical} \leftarrow \&v_{i,j}$，
$temp \leftarrow index$。

STEP 3 $index \leftarrow index + 1$。

STEP 4 從 LL_{Z} 中取出第 $index$ 個節點 $v_{i,j}$，如果 $D^{global}_{i,j}(k) < min$，則：
$min \leftarrow D^{global}_{i,j}(k)$，
$*v_{critical} \leftarrow \& v_{i,j}$，
$temp \leftarrow index$。

STEP 5 重覆 STEP 1.3 直至$LL^{z}_{k,l}$中的所有節點 $v_{i,j}$ 均被計算過。

END

副函式 5 最短距離:

BEGIN

STEP 1 計算$LL^{z}_{k,l}$節點離$D^{global}_{i,j}(l)$的距離跟座標，存入 $Len(l)$ 中。

STEP 2 找出與關鍵結點距離最短的節點存入 $Len(l)$中。

END

四、演算法效能分析

(一) 空間複雜度分析

多組史坦納樹連結演算法的空間複雜度分析，本演算法使用了一個 $m \times n$ 的二維陣列 $v_{i,j}$ 來儲存 X 架構網路中各個節點的狀態、p 個 $m \times n$ 二維陣列 $D^{local}(k)$來儲存各個 Z 節點分別的等距離圖、q 個 $m \times n$ 的二維陣列 D^{global} 來儲存等距離累加圖以及數個輔助計算用的資料結構。其中二維陣列的空間需求為 N 即 $m \times n$ 個。又因為組數 q 會比欲連結的端點數 p 為少，所以本演算法的空間複雜度(space complexity) S_{total} 為 ：

$$
\begin{aligned}
S_{total} &= S_{initial} + S_{D\_local} + S_{D\_global} \\
&= O(N + pN + qN) \\
&= O((1 + p + q)N) \\
&= O(pN)
\end{aligned}
$$

其中，$q < p < N$。

(二) 時間複雜度分析

多組史坦納樹連結演算法的時間複雜度分析，可分為幾個步驟來探討。第一部分為初始化(Step 1)，判斷每個節點是否為欲連結的端點，以及所屬的組別。這一步驟需 $O(N)$的時間。第二階段包含 Step 2 以及 Step 3。其中 Step 2.1 為計算每個終端節點至其他點的 HGMR 洪氾，需要 $O(pN)$。Step 2.2 呼叫等距離累加副函式，每個點須累加自己到所有組對的終端節點的洪氾值，所以時間為 $O(pN)$。Step 2.3 為搜尋關鍵節點副函式，由於要計算 q 組 $LL^{z}_{k,l}$，$LL^{z}_{k,l}$的長度最多不超過 p 個，因此時間複雜度 $O(pq)$。Step 2 中最後 2.4 呼叫最短距離副函式，選出各組的 Z 節點距離關鍵節點最短的距離，需要 $O(p)$。因

此 Step 2 總共需要:

$$T_{Step2} = O(N) + O(pN) + O(pq) = O(pN+pq)$$

又因為 $q < p < N$，所以 $T_{Step2} = O(pN)$

Step 3 為連結離關鍵節點(路徑)最近的路徑，要找到該節點須搜尋 $LL^{z}_{k,l}$，而 $LL^{z}_{k,l}$ 的長度最多不超過 p 個，接著迴圈呼叫 Step 2 直到結束。所以總和 Step 2 與 Step 3，總共需花費 $O(p^2N)$ 的時間複雜度。最後的步驟為輸出 $LL^{SMT}_{k,l}$，其長度不超過 N，所以時間亦限制在 $O(N)$以內。總結上述的步驟，本演算法的時間複雜度為 :

$$T_{total} = T_{initial} + T_{Step2\&Step3} + T_{Output}$$
$$= O(N) + O(p^2N) + O(N) = O(p^2N)$$

五、演算法範例說明

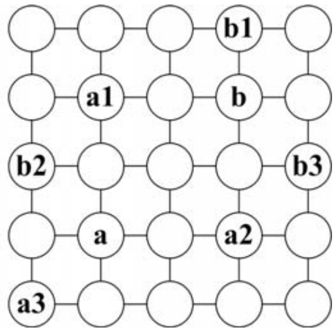
本範例以 5×5 網格大小來做說明，如圖四(a)為預設起始圖，有 a、b 兩不同群組的節點。首先要完成各節點洪泛步驟，四向連結的鄰格+1，四個對角的鄰格+$\sqrt{2}$，網格的最短距離可在+1 和+$\sqrt{2}$ 兩者中取最小者。接著依序更新表格並選取具有最短距離的線段連接，如圖四所示。其中每完成一組線段後，須重新計算關鍵節點距離各端點的距離。最後結果如圖四(g)所示。

六、結論

本論文提出了一個可適用於 X 架構網路上的多組史坦納樹演算法，其空間複雜度為 $O(pN)$，時間複雜度為 $O(p^2N)$，其中 p 表示網路中欲相互連結的節點總數，q 表示為網路中欲連結節點的組數，N 則表示為網路上所有節點的個數。

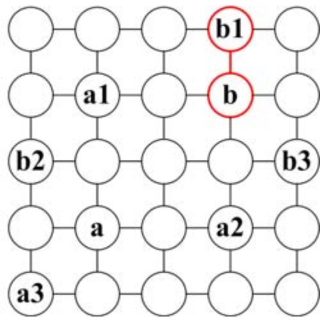
在未來的研究領域上，可將本論文中所使用的多組史坦納樹連結演算法朝向 3D 做為發展，如此一來，能更加深入應用在印刷電路版上，能夠更符合在 EDA 領域中的需求以及研究發展之應用層面。

	a	b
1	2	1
2	2	3.4
3	1.4	1.4



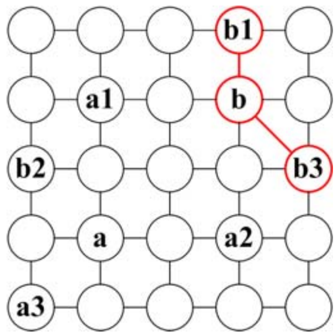
(a) 關鍵節點與端點距離與各組關鍵節點

	a	b
1	2	1
2	2	3.4
3	1.4	1.4



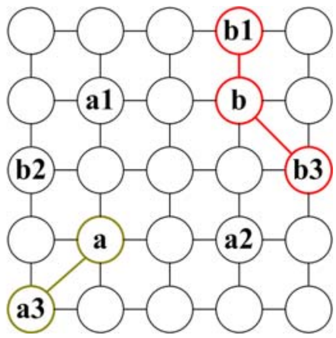
(b) 選取最短距離與第一條線段

	a	b
1	2	1
2	2	3.4
3	1.4	1.4



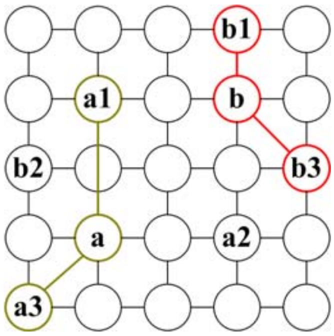
(c) 更新後的表格與第二條線段

	a	b
1	2	1
2	2	3.4
3	1.4	1.4



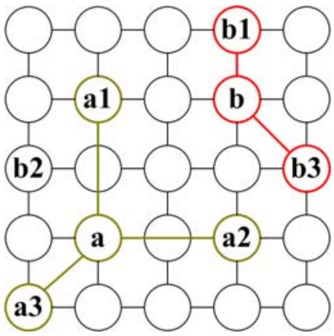
(d) 更新後的表格與第三條線段

	a	b
1	2	1
2	2	~~3.4~~ 4.4
3	1.4	1.4



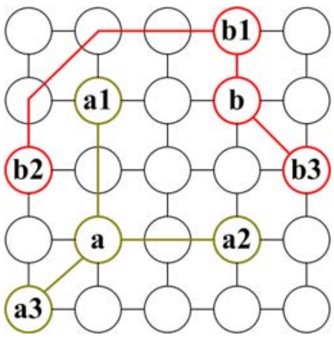
(e) 更新後的表格與第四條線段

	a	b
1	2	1
2	2	4.4
3	1.4	1.4



(f) 更新後的表格與第五條線段

	a	b
1	2	1
2	2	4.4
3	1.4	1.4



(g) 更新後的表格與最後完成圖

圖四、演算法範例

參考文獻

[1] 范啟明，“網狀網路上可容錯之直線式史坦那樹試誤型演算法”，*國立台灣海洋大學資訊科學系碩士論文*，中華民國九十一年七月。

[2] 羅仲志，“試誤型史坦那樹及單層多組線路連結演算法”，*國立台灣海洋大學資訊工程學系碩士論文*，中華民國九十三年七月。

[3] W. M. Boyce, “An improved program for the full Steiner tree problem,” *ACM Trans. on Math.* Software 3, pp. 194-206, 1977.

[4] W. M. Boyce and J. B. Seery, “STEINER 72: An improved version of the minimal network problem,” *Tech. Rep.*, No. 35, Comp. Sci. Res. Ctr. Bell Laboratories, Murray Hill, N.J.

[5] E. J. Cockayne and D. G. Schiller, “Computation of Steiner minimal trees in Welsh and Woddall (Eds.)”, *Combinatorics*, Inst. Math. Appl. pp. 52-71, 1972.

[6] L. R. Foulds and R. L. Graham, “The Steiner problem in phylogeny is NP-complete,” *Adv. Appl. Math.*, vol. 3, pp. 43-49, 1982.

[7] G. E. Jan, K. Y. Chang, S. Gao, and I. Parberry, “A 4-geometry maze router and its application on multiterminal nets,” *ACM Transactions on Design Automation of Electronic System*, vol. 10, pp. 116-135, Jan., 2005.

[8] R. M. Karp, “Reducibility among combinatorial problems,” in R. E. Miller, J. W. Thatcher (eds.), *Complexity of Computer Computations*, Plenum Press, New York, pp. 85-84, 1972.

[9] C. Y. Lee, “An algorithm for path connections and its applications,” *IRE Trans. Electron. Computer*, vol. EC-10, pp. 346-365, Sept. 1961.

[10] C. W. Lin, S. Y. Chen, C. F. Li, Y. W. Chang, and C. L. Yang, “Obstacle-Avoiding Rectilinear Steiner Tree Construction Based on Spanning Graphs,” IEEE Trans. Computer-Aided Design Integr. Circuits Syst., vol. 27, no. 4, pp. 643-653, April 2008.

[11] Z. A. Melzak, “On the problem of Steiner,” *Canad. Math. Bull.*, vol. 4, pp. 143-148, 1961.

[12] J. M. Smith, “An $O(n \log n)$ heuristic for Steiner minimal tree problems on the Euclidean metric,” *Networks*, vol. 11, pp. 23-39, 1981.

[13] P. Winter, “An algorithm for the Steiner problem in the Euclidean plane,” *Networks*, vol. 15, pp. 323-345, 1986.