# Independent Study Project

## NET OPEN LOCATION FINDER WITH OBSTACLES

Chia-Lin, Chen & Xiang-Yi, Liu
committee : Dr. Wei-Kai Cheng

# Overview

- Background
- Description
- Process Of Thinking
- The Flow Chart
- Results And Comparison
- Conclusion
- Reference

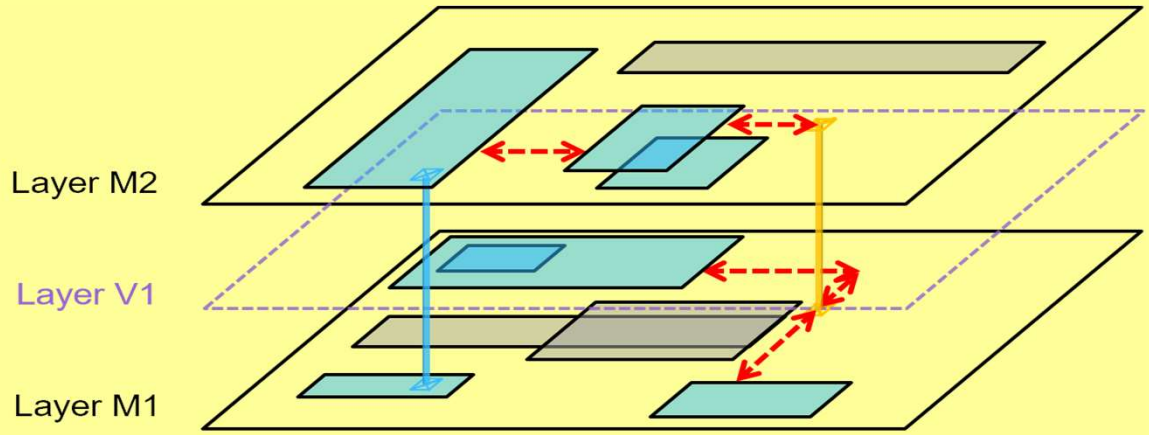
# Background

While people designed IC, it's important to make the connection avoid some obstacles,

and it's also important to make overall cost as little as possible.

For this reason, we considered a lot of mathematic methods, using some algorithms and data structures to do it well.

# Description



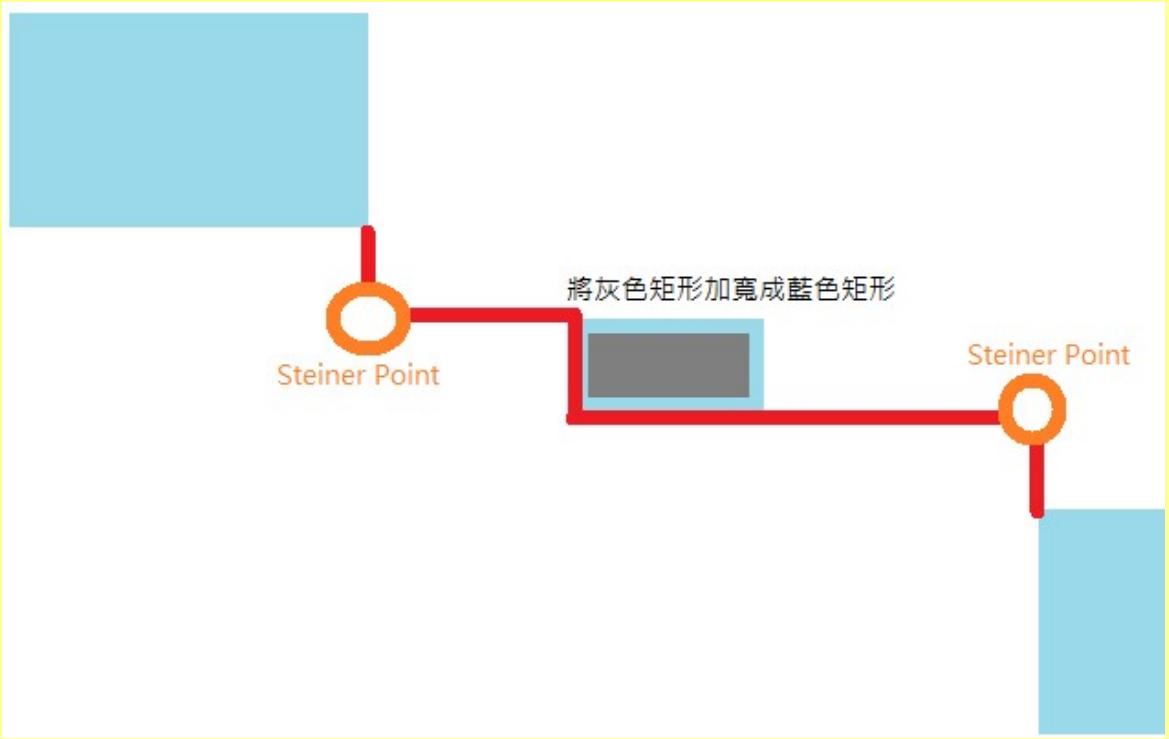

- Make all the above blue rectangles be connected, BUT AVOID TO PASS THROUGH THE GREY RECTANGLES(OBSTACLES).
- Via used on layers' connection
- Goal : according to the formula, let the overall cost as little as possible.

Overall Cost = $\sum Cost(P_{path}) + \sum Cost(P_{via})$

- $Cost(P_{path})$ : path of V-line or H-line
- $Cost(P_{via})$ : path of Via

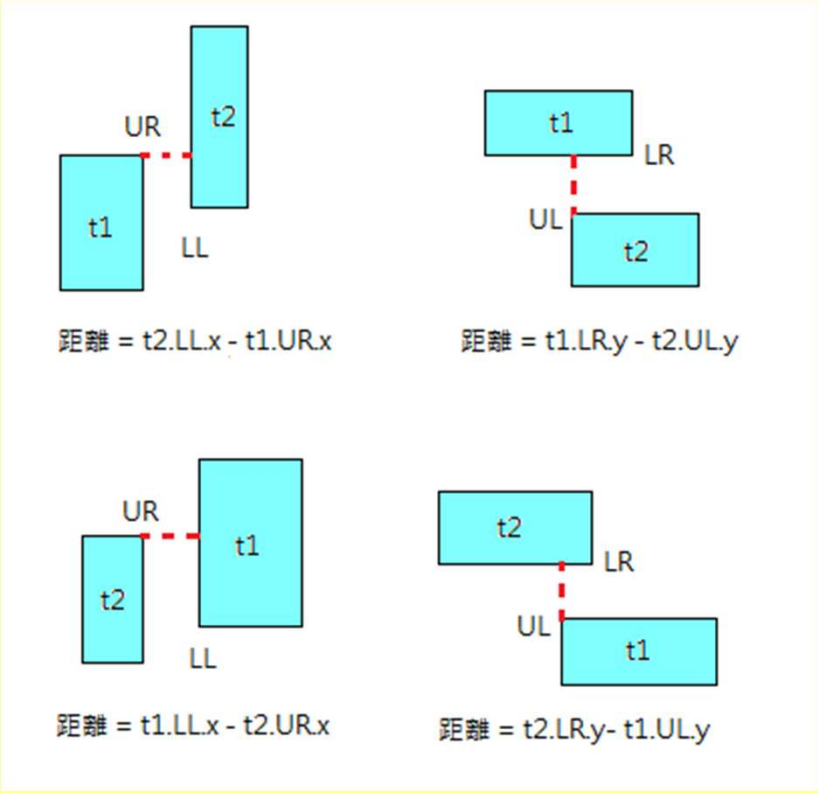
# Process Of Thinking

**The “Big Bad Wolf” method**

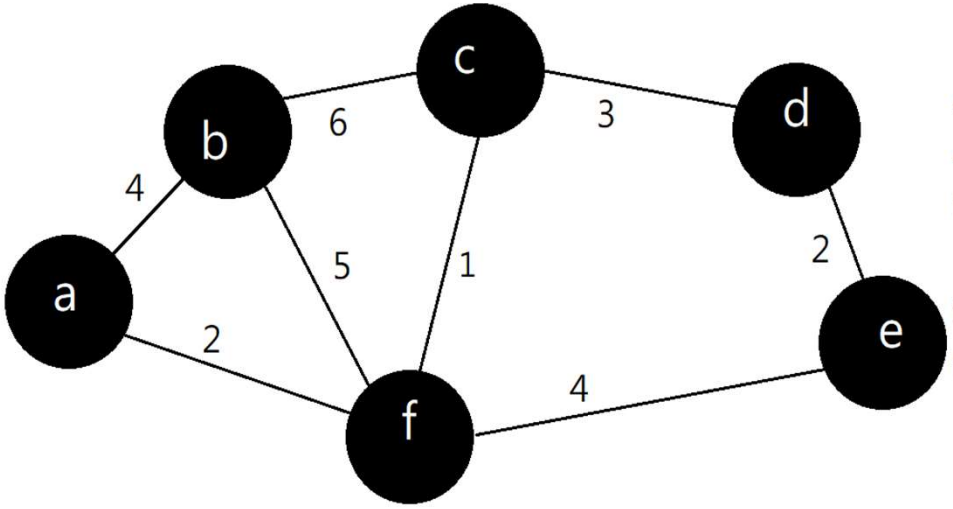


# Process Of Thinking

**FindDistance**



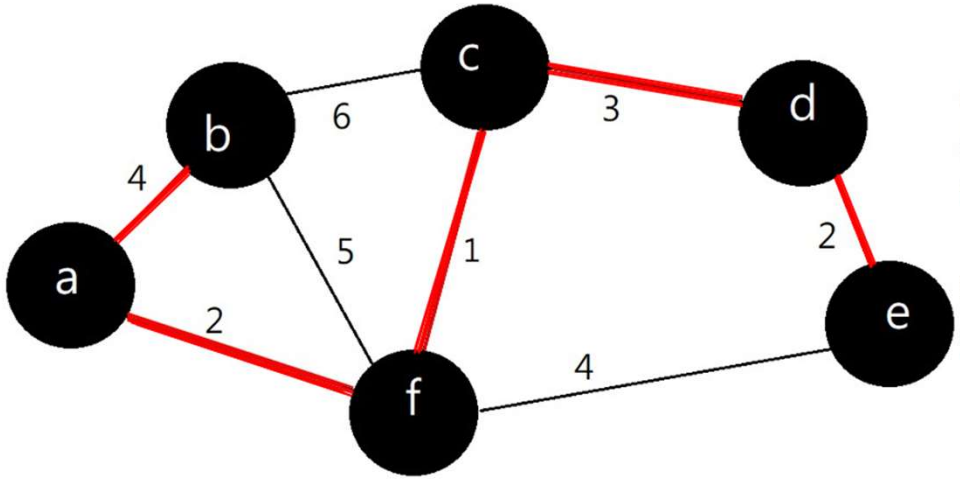
# Process Of Thinking




**Kruskal's Algorithm**

A={}, Kruskal(V,E)
A=∅
**foreach** v ∈ V :
  Make-disjoint-set(v)
Sort E by weight increasingly
**foreach** (v1,v2) ∈ E:
 **if** Find(v1)≠Find(v2):
  A=A∪{(v1,v2)}
  Union (v1,v2)
return A

# Process Of Thinking

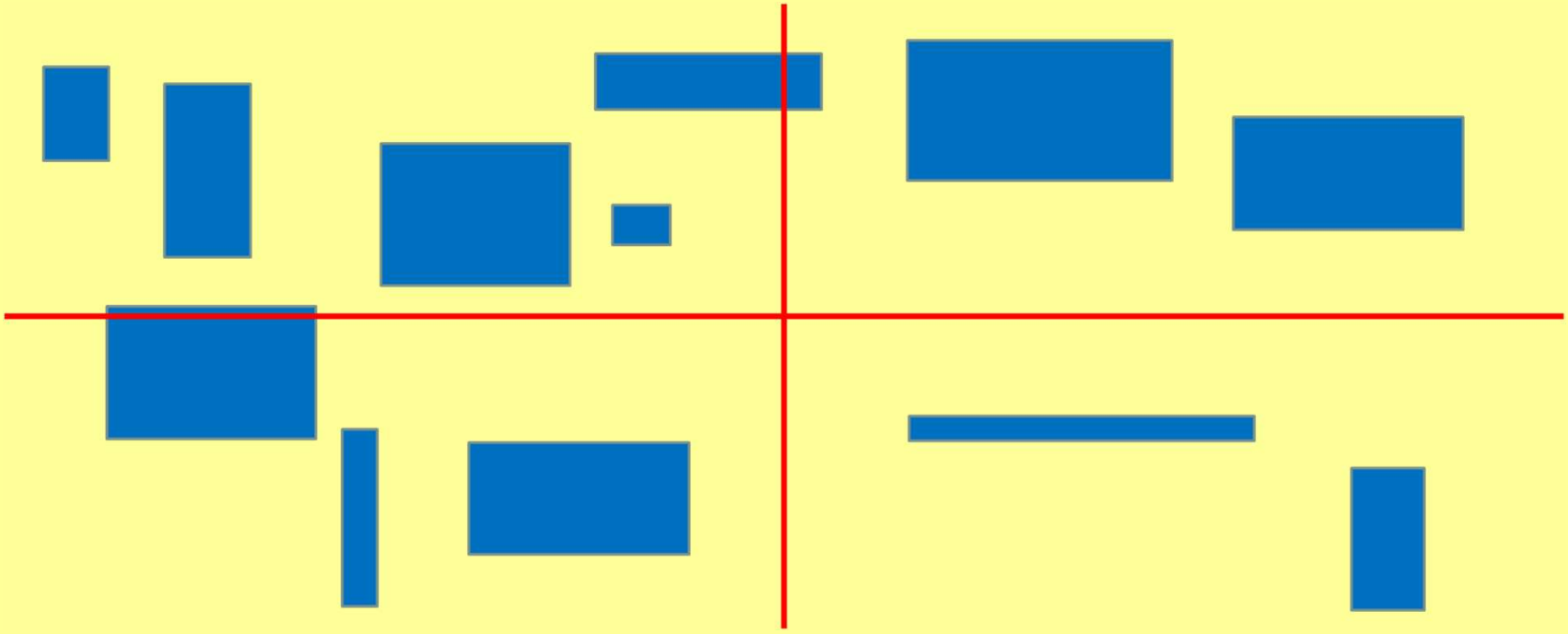



Kruskal's Algorithm

```
A={}, Kruskal(V,E)
A=ø
foreach v ∈ V :
   Make-disjoint-set(v)
Sort E by weight increasingly
foreach (v1,v2) ∈ E:
 if Find(v1)≠Find(v2):
   A=A∪{(v1,v2)}
   Union (v1,v2)
return A
```

# Process Of Thinking

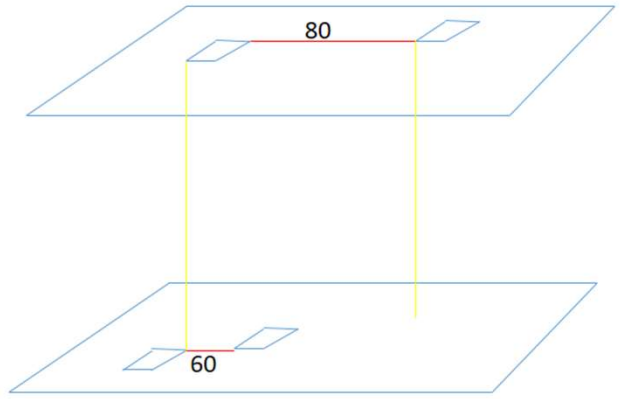
**The "Divide-by-four" method**

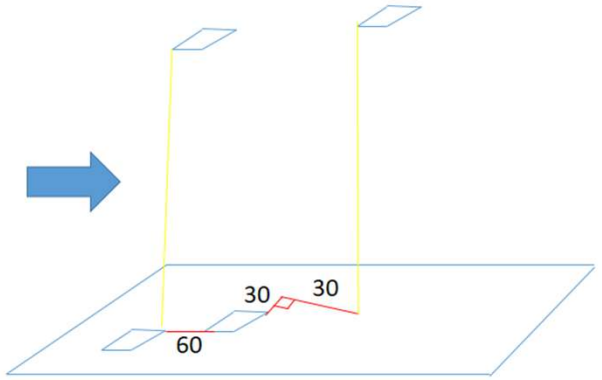


# Process Of Thinking

**The “Projection” method**

註:黃線為via，此例via之cost為10



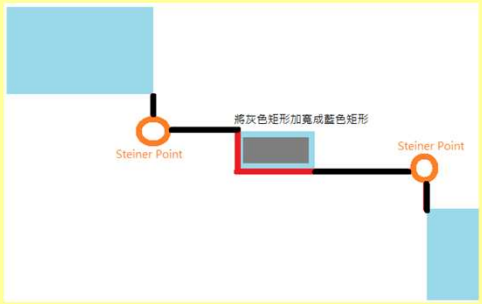

使用最初方法得Overall cost = 60+80+10*1=150

使用投影法後，圖中4個藍色矩形兩兩相連，
用Kruskal演算法，得需用得最短三條線分別為
10、60、70(30+30+10)，所以總cost=10+60+70=140

# The Flow Chart

1. input
2. Broaden all grey rectangles to be blue Rectangles.
3. Divide all rectangles to four quadrants in each layers
4. Do Projection
5. Find distances between every two rectangles
6. Check if it is a Best Case?
   - No → Do Quick Sorting
   - Yes → Do Bubble Sorting
7. Kruskal' s algorithm
8. Add 'L' paths
9. Delete the overlaped paths
10. output

Four quadrants，do four times



# Results And Comparison

- **case1** T… e cost … via is 450), execute…
- **case2** T… he cos… of via is 5580), execute…
- **case5** c… ost of … ia is 4505), execute…
- (The ab… WIND… ™ i5-4210U (ASUS …

net_open_finder_output - 記事本

檔案(F) 編輯(E) 格式(O) 檢視(V) 說明(H)

```
H-line M1 (3525,1631) (3491,1631)
V-line M1 (3458,1602) (3458,1501)
H-line M1 (3458,1501) (3357,1501)
V-line M1 (3525,1602) (3525,1395)
H-line M1 (3525,1395) (3702,1395)
V-line M1 (5652,2776) (5652,2775)
V-line M1 (3662,2744) (3662,2743)
V-line M1 (4115,2798) (4115,2797)
V-line M1 (6041,2774) (6041,2773)
V-line M1 (3406,2870) (3406,2869)
V-line M1 (3667,2743) (3667,2742)
V-line M1 (4110,2472) (4110,2471)
V-line M1 (4220,2185) (4220,2184)
V-line M1 (4304,2184) (4304,2182)
V-line M1 (3671,2759) (3671,2757)
V-line M1 (6796,2656) (6796,2654)
V-line M1 (4159,2964) (4159,2962)
V-line M1 (5611,2784) (5611,2782)
V-line M1 (5393,2566) (5393,2564)
V-line M1 (4954,2180) (4954,2178)
V-line M1 (5158,2409) (5158,2407)
H-line M1 (6824,2416) (6822,2416)
V-line M1 (4856,2367) (4856,2364)
V-line M1 (4433,1833) (4433,1830)
H-line M1 (5953,1752) (5950,1752)
```

case1.txt - 記事本

檔案(F) 編輯(E) 格式(O) 檢視(V) 說明(H)

```
V-line M3 (5041,253) (5041,245)
H-line M3 (3731,893) (3723,893)
V-line M3 (6094,1422) (6094,1414)
H-line M3 (6634,1155) (6625,1155)
H-line M3 (6722,1184) (6713,1184)
V-line M3 (3616,437) (3616,428)
V-line M3 (4225,35) (4225,26)
V-line M3 (4152,393) (4152,383)
H-line M3 (4739,704) (4729,704)
Via  V1 (5303,2431)
Via  V1 (5253,2407)
Via  V1 (5611,2582)
Via  V1 (3562,1994)
Via  V1 (1896,1781)
Via  V1 (2957,1679)
Via  V1 (570,1148)
Via  V1 (514,1160)
Via  V1 (1714,820)
Via  V1 (3343,174)
Via  V1 (294,1411)
Via  V1 (2462,787)
Via  V1 (5063,1340)
Via  V2 (6419,2275)
Via  V2 (6320,1993)
```

# Results And Comparison

- **Results and comparison in case1**

| version | layers | The amount of blue rectangles | The amount of grey rectangles | Costs of paths | The total amount of via that we use | Costs of vias (In this case is 10) | Overall cost= Costs of paths + Costs of vias | The time of execution |
|---|---|---|---|---|---|---|---|---|
| **The first** | 3 | 1503 | 414 | 53095736 | 2 | 20 | 53095756 | 204secs. |
| **The last** | | | | 4326 | 45 | 450 | 4776 | 94secs. |

# Results And Comparison

- **Results and comparison in case2**

| version | layers | The amount of blue rectangles | The amount of grey rectangles | Costs of paths | The total amount of via that we use | Costs of vias (In this case is 30) | Overall cost= Costs of paths +Costs of vias | The time of execution |
|---|---|---|---|---|---|---|---|---|
| **The first** | 5 | 4518 | 4773 | 515899298 | 4 | 120 | 515899418 | 30570secs. (about 8.5 hrs.) |
| **The last** | | | | 39806 | 186 | 5580 | 45386 | 1019secs. |

# Results And Comparison

- **Results and comparison in case5**

| version | layers | The amount of blue rectangles | The amount of grey rectangles | Costs of paths | The total amount of via that we use | Costs of vias (In this case is 5) | Overall cost= Costs of paths +Costs of vias | The time of execution |
|---|---|---|---|---|---|---|---|---|
| **The first** | 5 | 4450 | 4762 | 405798912 | 4 | 20 | 405798932 | 28185secs. (about 7.8hrs.) |
| **The last** | | | | 13691 | 901 | 4505 | 18196 | 939secs. |

# Conclusion

- The above cases showed that if we use the “Projection Method”, although the costs of all vias MUST GROW UP, the overall costs WILL CUT DOWM when the given cost of via is less than 100 and the given amount of layers are more than 3.

# Reference

- Results in case1

1.txt - 記事本

檔案(F) 編輯(E) 格式(O) 檢視(V) 說明(H)

```
ViaCost = 10
Spacing = 8
Boundary = (0,0) (7000,3000)
#MetalLayers = 3
#RoutedShapes = 1503
#RoutedVias = 3
#Obstacles = 414
RoutedShape M1 (2244,2426) (2309,2436)
RoutedShape M1 (2263,2415) (2267,2442)
RoutedShape M1 (6360,2379) (6380,2400)
RoutedShape M1 (6351,2387) (6388,2394)
RoutedShape M1 (6277,1307) (6293,1332)
RoutedShape M1 (6262,1328) (6320,1343)
RoutedShape M1 (548,2252) (671,2279)
RoutedShape M1 (600,2250) (611,2257)
```

C:\Users\Lana\Desktop\專題\算專題的cost\net_open_finder_cada082.e

```
Please input a file name, [0]Quit :
1.txt

Process returned 0 (0x0)   execution time : 94.544 s
Press any key to continue.
```

C:\Users\Lana\Desktop\專題\算專題的cost\計算出cost.exe

```
Please input a file name, [0]Quit :
case1.txt
所有用到的H-line/V-line之path加起來的cost為: 4326
總via數: 45條
總via cost為: 450
總path之cost再加上via cost為: 4776
請按任意鍵繼續 . . .

Process returned 0 (0x0)   execution time : 8.834 s
Press any key to continue.
```

# Reference

- Results in case2

2.txt - 記事本

檔案(F) 編輯(E) 格式(O) 檢視(V) 說明(H)

```
ViaCost = 30
Spacing = 8
Boundary = (0,0) (40000,15000)
#MetalLayers = 5
#RoutedShapes = 4518
#RoutedVias = 34
#Obstacles = 4773
RoutedShape M1 (24891,3089) (24894,3129)
RoutedShape M1 (22231,2090) (22239,2153)
RoutedShape M1 (4502,12563) (4508,12625)
RoutedShape M1 (262,10567) (365,10570)
RoutedShape M1 (27050,8439) (27465,8469)
RoutedShape M1 (22105,8043) (22125,8202)
RoutedShape M1 (21907,8117) (22353,8141)
RoutedShape M1 (11858,5908) (12520,5949)
```

C:\Users\Lana\Desktop\專題\算專題的cost\net_open_finder_cada082.exe

```
Please input a file name, [0]Quit :
2.txt

Process returned 0 (0x0)   execution time : 1019.480 s
Press any key to continue.
```

C:\Users\Lana\Desktop\專題\算專題的cost\計算出cost.exe

```
Please input a file name, [0]Quit :
case2.txt
所有用到的H-line/V-line之path加起來的cost為: 39806
總via數: 186條
總via cost為: 5580
總path之cost再加上via cost為: 45386
請按任意鍵繼續 . . .

Process returned 0 (0x0)   execution time : 8.253 s
Press any key to continue.
```

# Reference

- Results in case5

5.txt - 記事本

檔案(F) 編輯(E) 格式(O) 檢視(V) 說明(H)

```
ViaCost = 5
Spacing = 14
Boundary = (0,0) (35000,20000)
#MetalLayers = 5
#RoutedShapes = 4450
#RoutedVias = 31
#Obstacles = 4762
RoutedShape M1 (14263,4378) (14267,4465)
RoutedShape M1 (9166,8541) (9397,8550)
RoutedShape M1 (1268,12213) (1579,12240)
RoutedShape M1 (9604,19145) (10109,19168)
RoutedShape M1 (12400,5352) (12535,5377)
RoutedShape M1 (23944,7783) (23964,8040)
RoutedShape M1 (23674,17257) (23988,17282)
RoutedShape M1 (5596,18807) (6291,18855)
```

C:\Users\Lana\Desktop\專題\算專題的cost\net_open_finder_cada082.exe

```
Please input a file name, [0]Quit :
5.txt

Process returned 0 (0x0)   execution time : 939.481 s
Press any key to continue.
```

C:\Users\Lana\Desktop\專題\算專題的cost\計算出cost.exe

```
Please input a file name, [0]Quit :
case5.txt
所有用到的H-line/V-line之path加起來的cost為: 13691
總via數: 901條
總via cost為: 4505
總path之cost再加上via cost為: 18196
```


Thank you