tags: **PD**

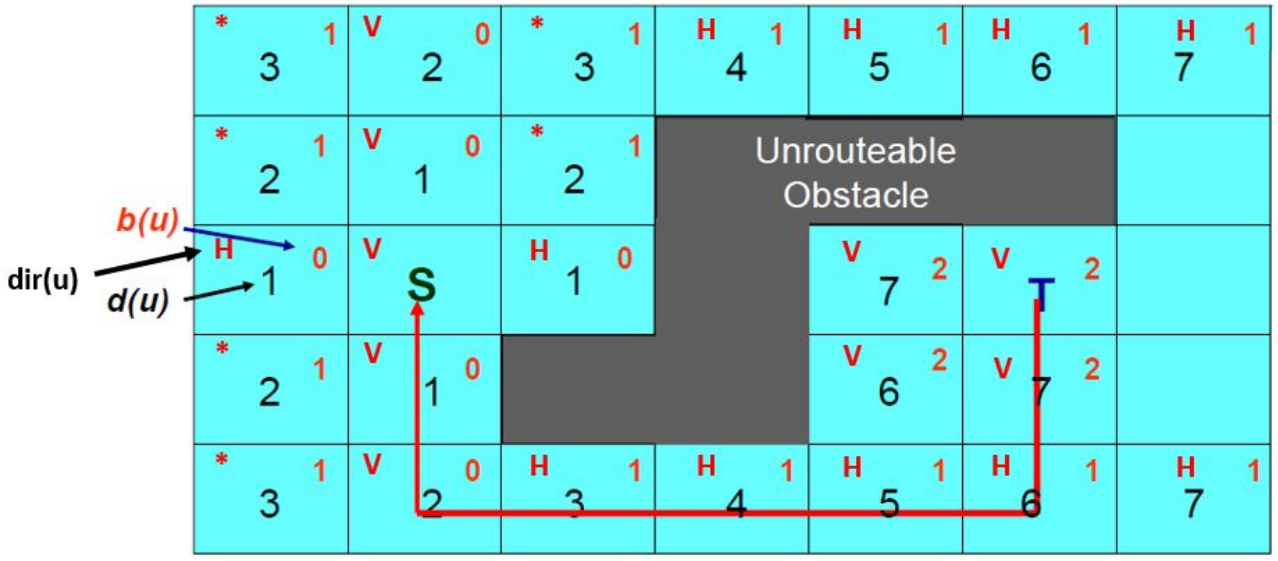
Physical Design HW4

Problem 1 [Collaborator: None]

Also traversing the grid by BFS with increasing distanve, but apart from the recorded distance $d(u)$, we need to record two additional elements: $b(u)$ be the minimum bend to node u, and $dir(u)$ be the last traverse direction of the min. bend path to node u. Notice that there are three directions: H, V, $*$ where $*$ denotes either Horizontal and Vertical direction exists a min. bend path.

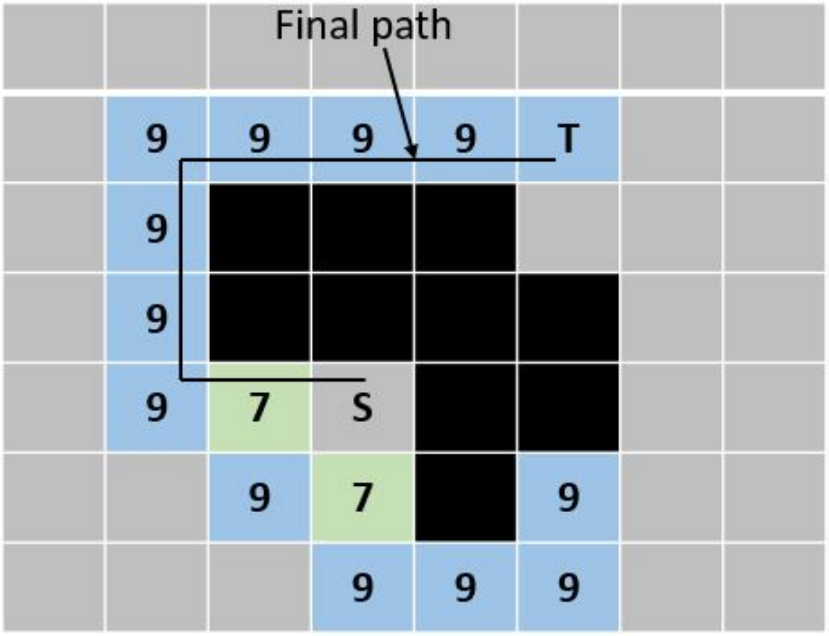
When a new node is traversed, we we must check the direction of the new traverse node corresponding to its parent nodes. If it is the same as $dir(.)$ of parent node, then the bend is the same as its parent node; otherwise, the bend is increase by 1. Notice that direction $*$ is defined to be the same as either H or V.

A simple example is shown in the figure below:



Problem 2 [Collaborator: None]

The procedure of A* search is shown in the figure below.



Problem 3 [Collaborator: None]

a.

For each node, we find the neareat node in four directions R1, R2, R3, R4 and form the table (Fig. 2). Then we can construct the spanning graph based on this table (Fig. 3).

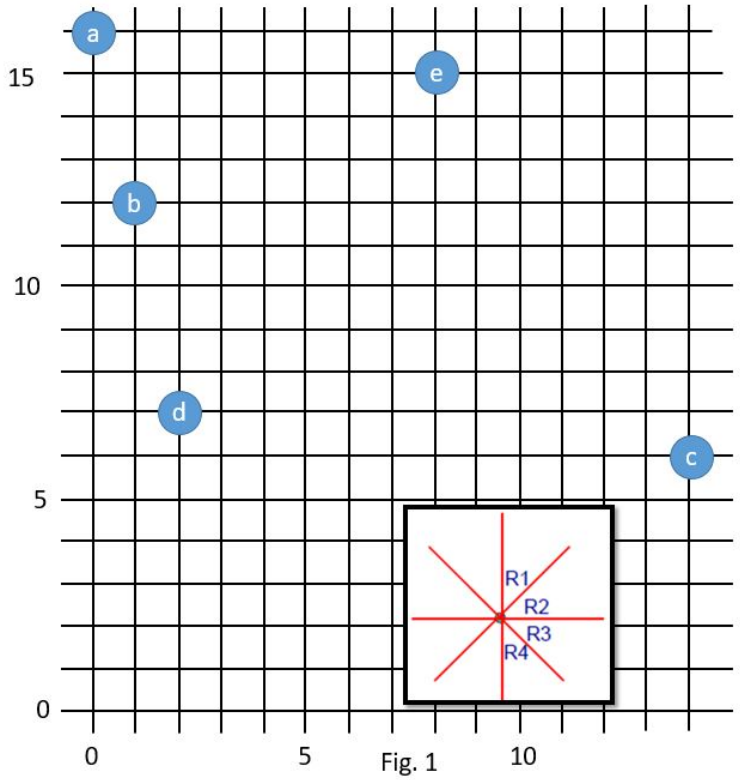


Fig. 1

	a	b	c	d	e
R1				e	
R2		e			
R3	e	c		c	
R4	b	d			c

Fig. 2

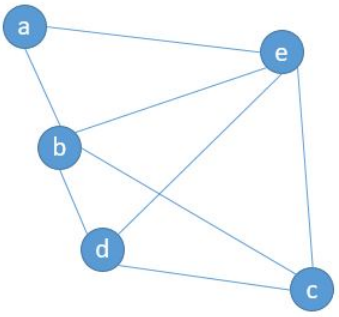


Fig. 3

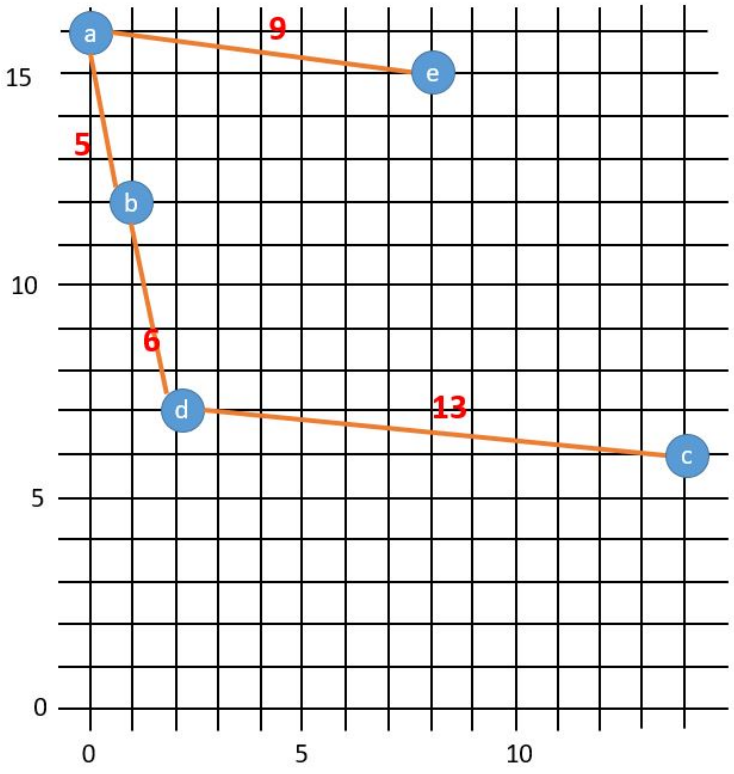
b.

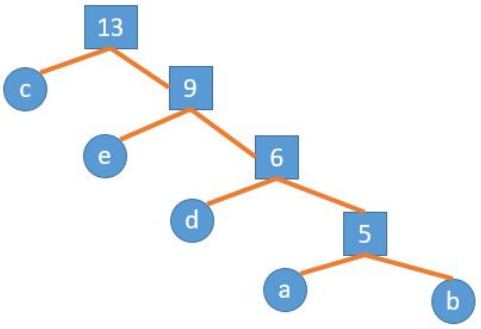
[Ref. Hai Zhou, "Efficient Steiner Tree Construction Based on Spanning Graphs," ISPD 03]

First, build the minimum spanning tree and merging binary tree by applying Kruskal's algorithm.

Edge	Wirelength	Chosen oreder
ab	5	1
ae	9	3
bc	19	
bd	6	2
be	10	
cd	13	4
ce	15	
de	14	

The resulting minimum spanning tree and merging tree is shown in the figure below.





Then the point edge pairs is shown is the table below:

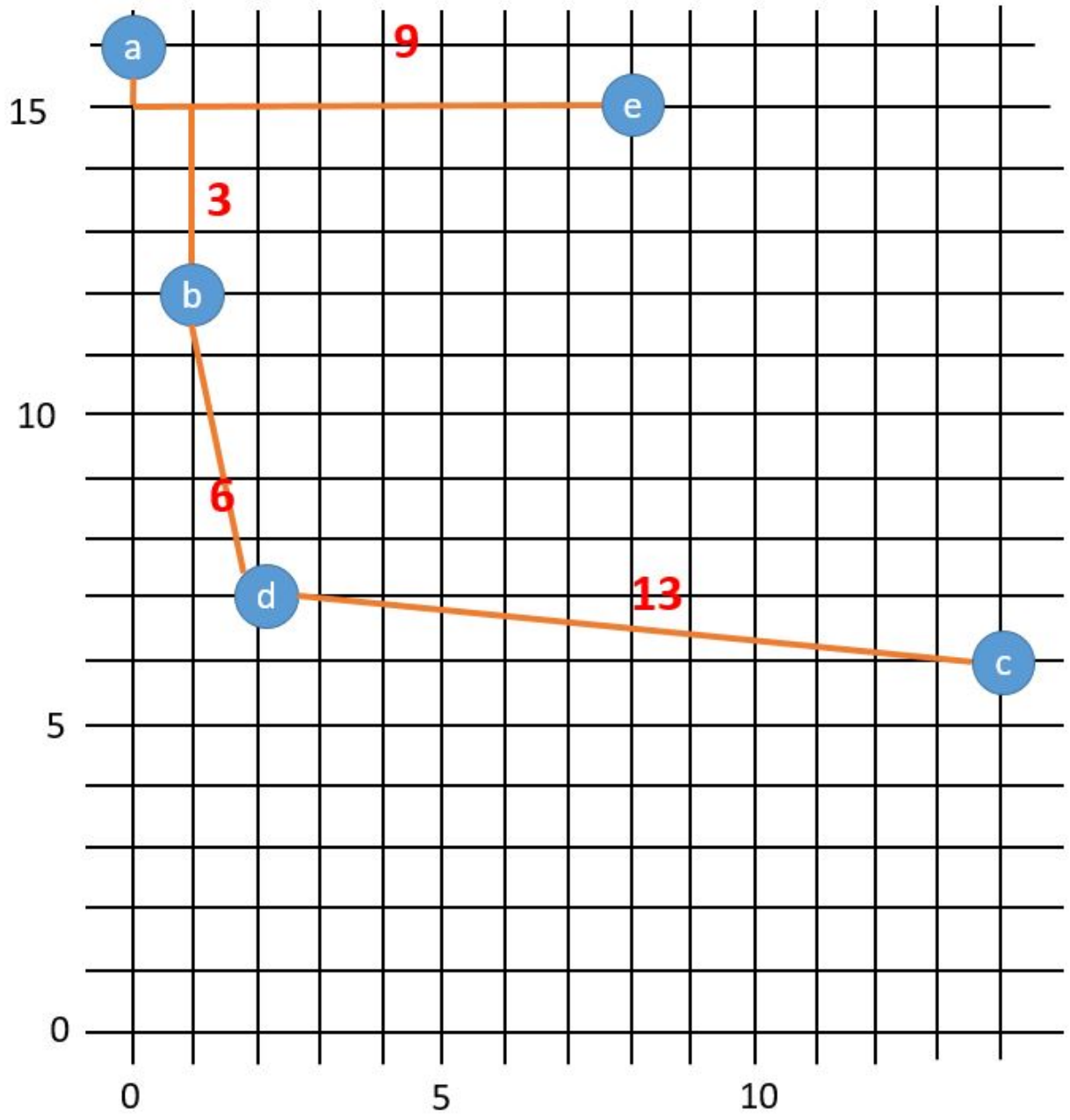
Point	Edge	Delete edge	Gain
b	ae	ab	2
e	ab	ea	2
e	cd	ae	1
d	ab	db	0
a	bd	ab	0
c	bd	cd	0
b	cd	bd	0
e	bd	ae	-1
c	ae	cd	-2
c	ab	cd	-6

The positive gain point edge pair is:

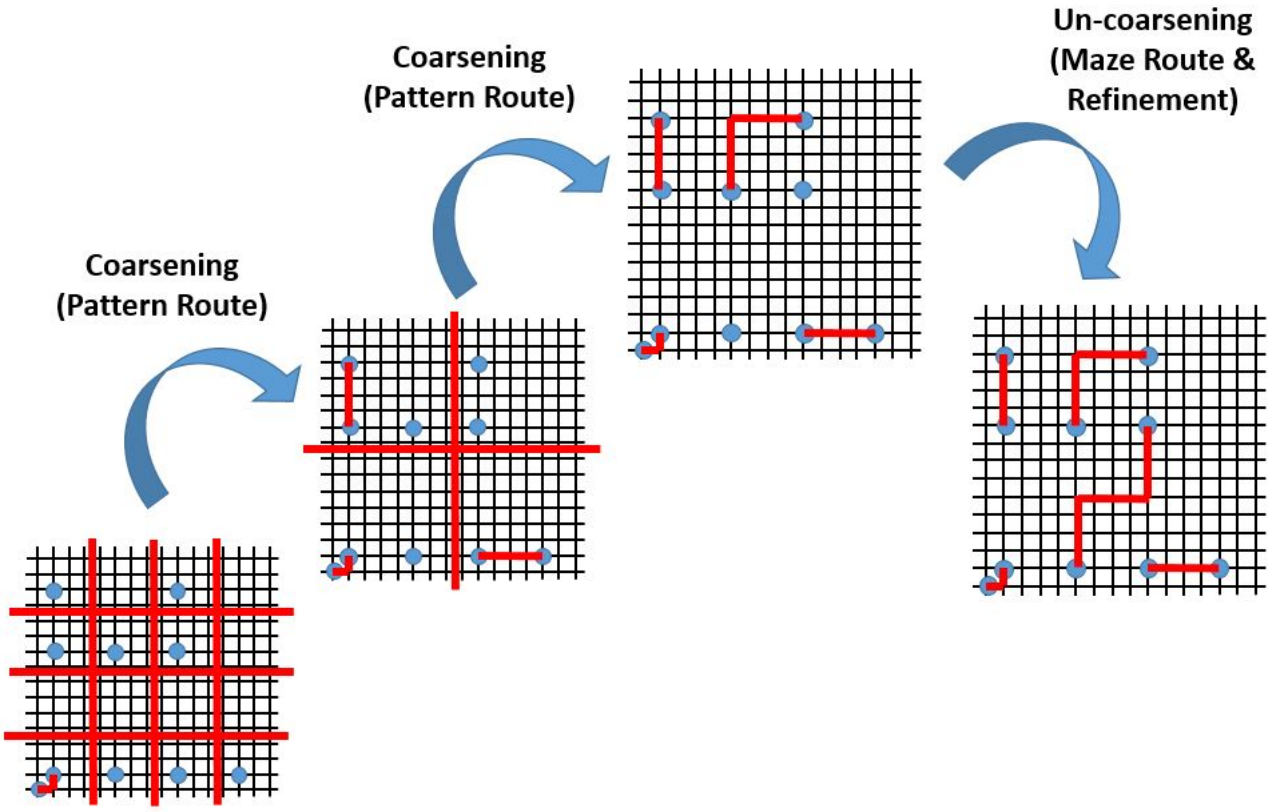
Point	Edge	Delete edge	Gain
b	ae	ab	2
e	ab	ea	2
e	cd	ae	1

Choose b to ae, and delete the related pairs.

Therefore, the final solution is as follows:

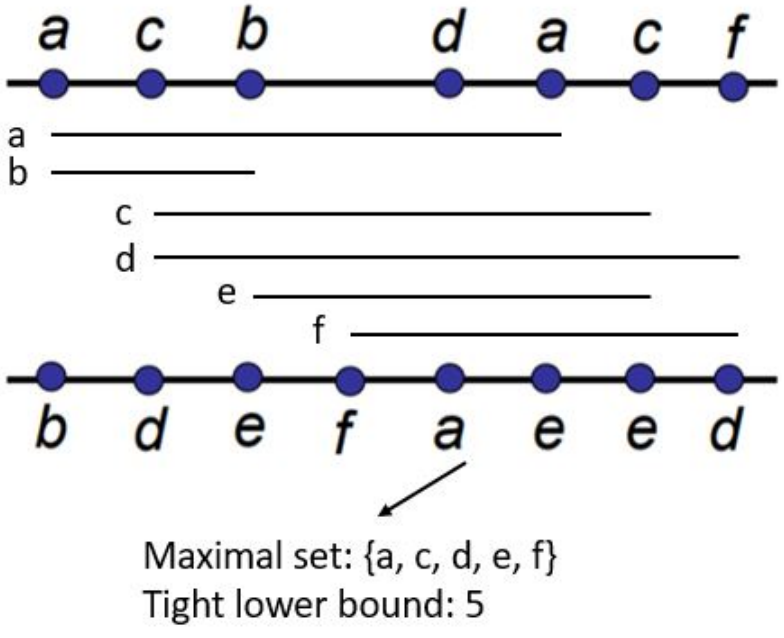


Problem 4 [Collaborator: None]



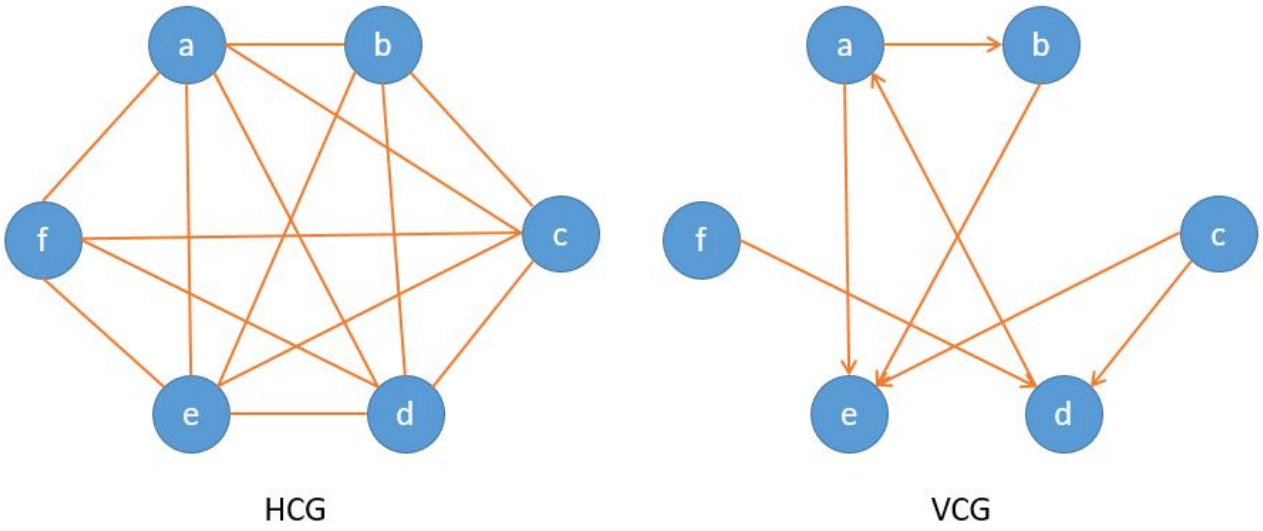
Problem 5 [Collaborator: None]

a.



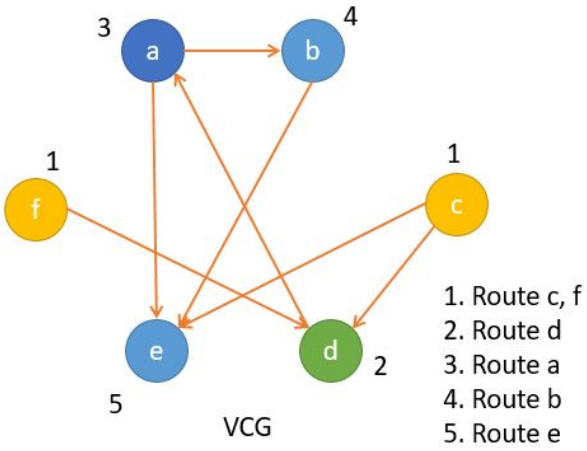
Maximal set: {a, c, d, e, f}
Tight lower bound: 5

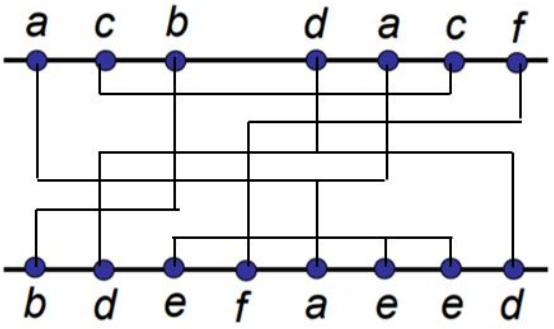
b.



c.

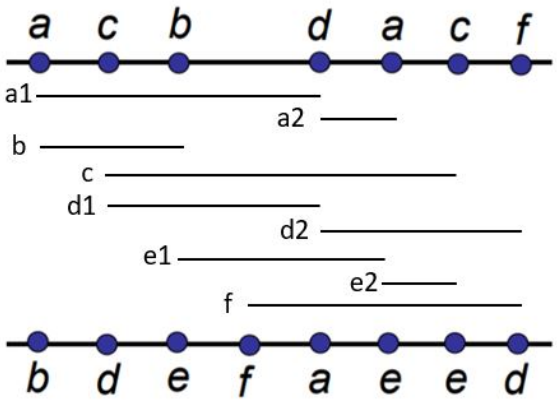
Yes, the constrained left edge algorithm can be applied to this routing instance, and the routing result is shown in the figure below.

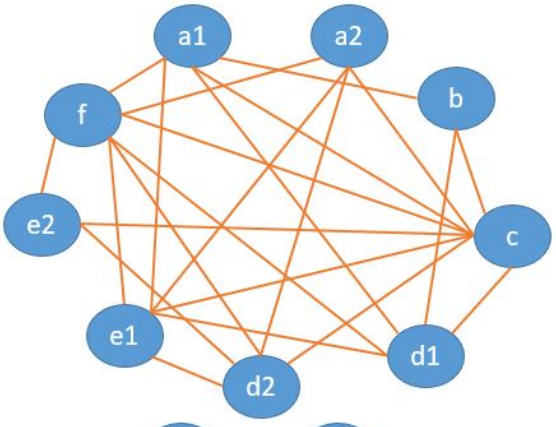




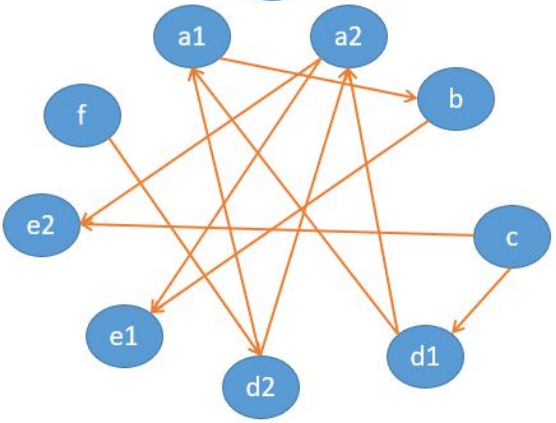
d.

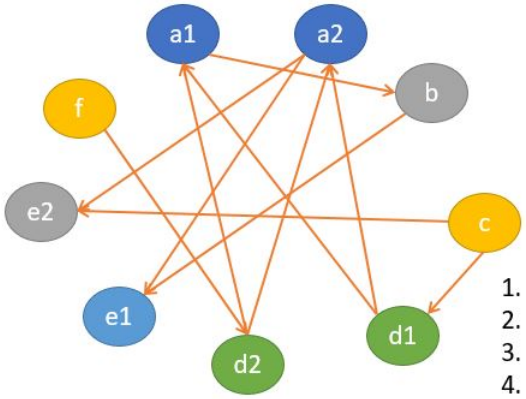
Yes, the dogleg channel router can be applied to this instance, and the procedure and result is shown in the figure below.



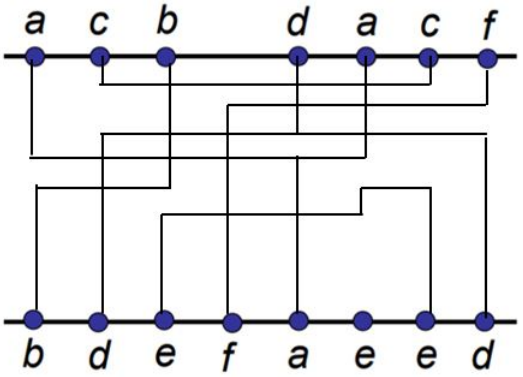


1. Find multiple pin nets: a, d, e
2. Split a into a1, a2; d into d1, d2 ; e into e1, e2
3. Forms HCG & ECG
4. Apply constrained left edge algorithm





1. Route c, f
2. Route d1, d2
3. Route a1, a2
4. Route b, e2
5. Route e1



Problem 6 [Collaborator: None]

a.

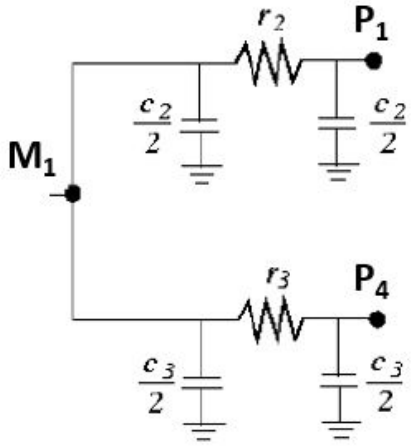
[Ref: Jason Cong, Andrew B. Kahng, Gabriel Robins, "Matching-Based Methods for High-Performance Clock Routing," IEEE TRANSACTIONS ON COMPUTER-AIDED DESIGN OF INTEGRATED CIRCUITS AND SYSTEMS, VOL 12, NO. 8, AUGUST 1993]

$P_1 : (20, 80),\ P2 : (10, 10),\ P3 : (50, 0),\ P4 : (60, 90)$

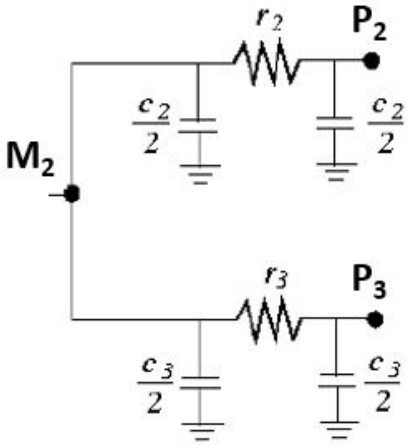
	P1	P2	P3	P4
P1	-	80	110	50
P2	-	-	50	130
P3	-	-	-	100
P4	-	-	-	-

Minimum cost matching: $(P_1, P_4),\ (P_2, P_3)$.
Connect all pairs of point and represent it by its balanced point, which located in the "straight line" between the two nodes as illustrated in the paper.
The remain balanced point is $M_1 = (40, 85)$ and $M_2 = (30, 5)$
The length of M_1 to M_2 is 90, and it can achieve **zero clock skew** by connecting the clock signal at $(35, 45)$. Though H-flipping does not increase clock skew, it increase the total tree cost; therefore, it is not applied in this case.

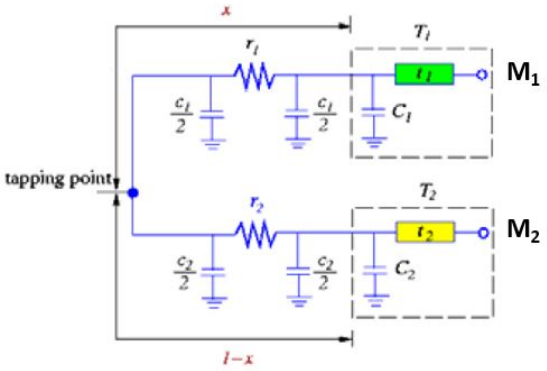
b.
Same as a., we find the minimum cost matching: $(P_1, P_4),\ (P_2, P_3)$.
For (P_1, P_4), The tapping point is its mid point $M_1 = (40, 85)$, and the tapping point for $M_2 = (P_2, P_3)$ is $(30, 5)$.
By applying the pi-model, we can calculate the delay as follows:



r2 = r3 = 25 * 0.1 = 2.5
c2 = c3 = 25 * 0.2 = 5
Cap. = 5 + 5 + 2 = 12
Delay = 2.5 * (2.5 + 1) = 8.75 (s)



r2 = r3 = 25 * 0.1 = 2.5
c2 = c3 = 25 * 0.2 = 5
Cap. = 5 + 5 + 6 = 16
Delay = 2.5 * (2.5 + 3) = 13.75 (s)



t1 = 8.75, C1 = 12
t2 = 13.75 , C2 = 16

x = [(5)+0.1*90*(16+90*0.2/2)]
/ [0.1*90*(0.2*90+12+16)]
= 0.5556

0.5556 * 90 = 50
Let clk at (35,40)

r1 = 50 * 0.1 = 5
c1 = 50 * 0.2 = 10
r2 = 40 * 0.1 = 4
c2 = 40 * 0.2 = 8

Delay M1
= 5 * (5 + 12) + 8.75
= 93.75
Delay M2
= 4 * (4 + 16) + 13.75
= 93.75

Therefore, the resulting clock skew is 0(s), and delay 93.75(s).

c.
For adding buffer in the solution of a., we can't move the coocrdinate of the clk signal and the wire.

It is obvious that the buffer should be insert between clk and M_1 and between clk and M_2.

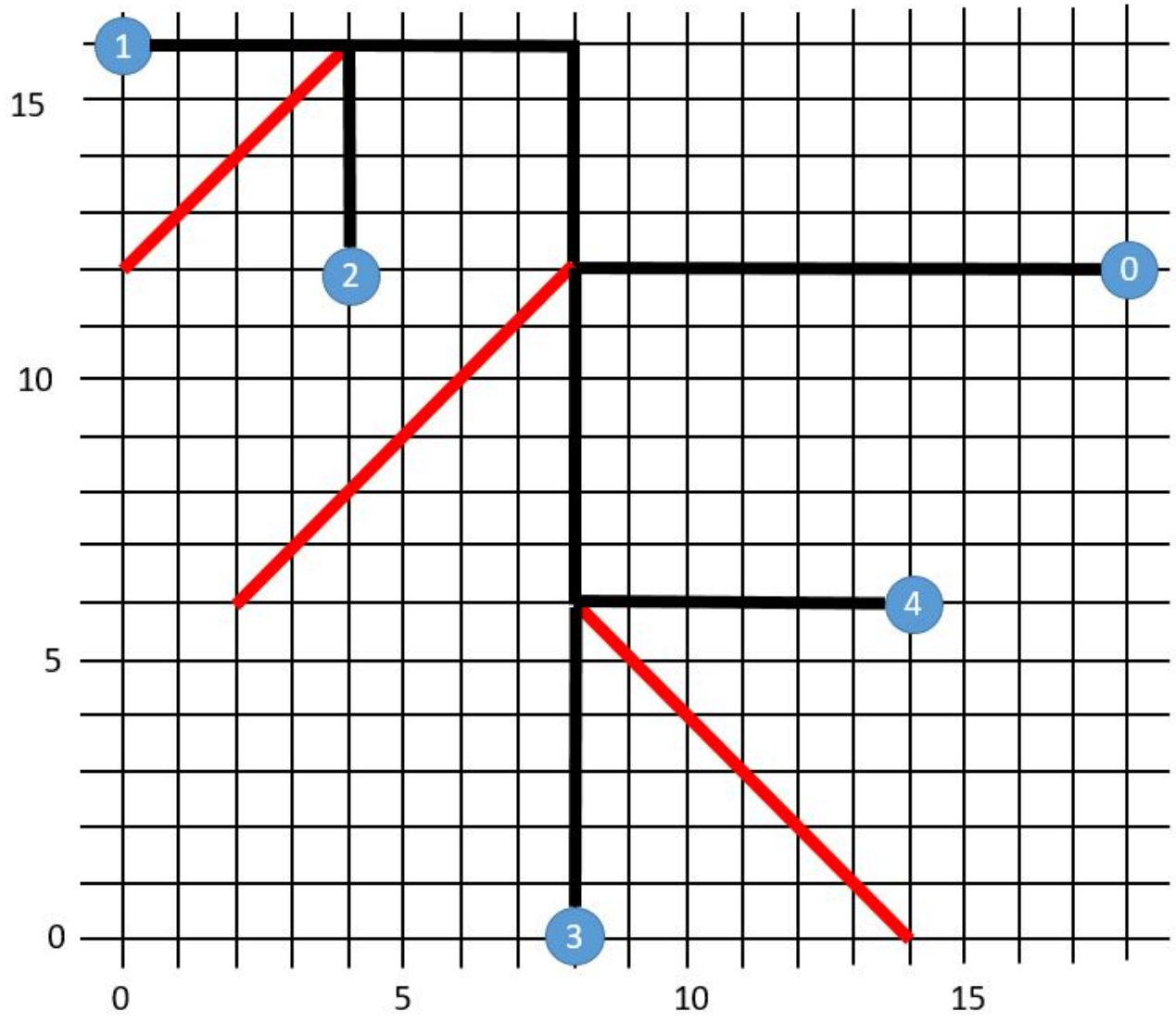
Supposed the buffer is insert at distance x from clk signal between clk and M_2, then we can formulate the delay as follows:
$delay = 0.1x(0.1x + 0.2) + 0.1(25 - 0.2x) + (4.5 - 0.1x)(20.5 - 0.1x) + 13.75$
$= 0.02x^2 - 2.5x + 108.5$
So, choose $x = 45$, and the resulting delay is 36.5.

Simlarly the delay of M_2 side is:
$delay = 0.1x(0.1x + 0.2) + 0.1(21 - 0.2x) + (4.5 - 0.1x)(16.5 - 0.1x) + 8.75$
$= 0.02x^2 - 2.1x + 85.1$
Here we choose $x = 35$ to minimize the clock skew.
The resulting delay is 36.1.

Therefore, the overall clock delay is 36.5(s) and clock skew is 0.4(s).

Problem 7 [Collaborator: None]

The red lines are merging segments, and the black lines are clock tree.

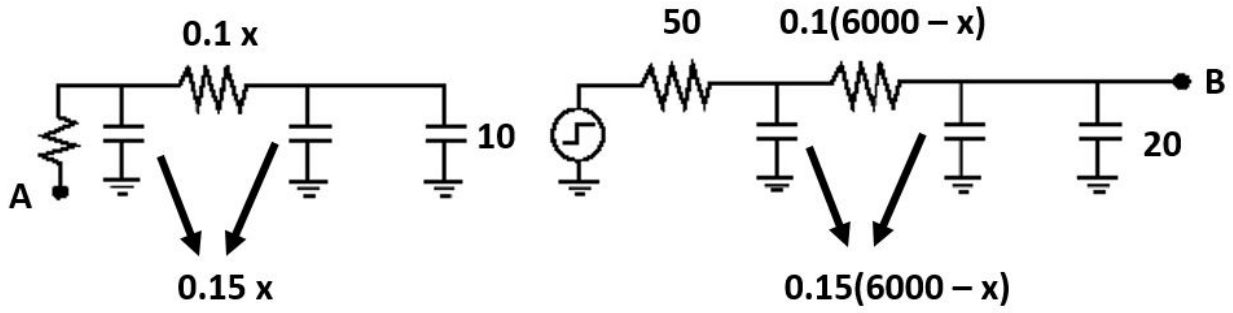


Problem 8 [Collaborator: None]

a.

Assume that the unit size buffer is inserted at distance x from a.

Then the resulting model is shown in the figure below:



The delay

$t_{ab} = 100(0.3x + 10) + 0.1x(0.15x + 10) + 50[0.3(6000 - x) + 20] + 0.1(6000 - x)[0.15(6000 - x) \cdot$

$= 0.03x^2 - 166x + 644000$

$\frac{\partial(0.03x^2 - 166x + 644000)}{\partial x} = 0.06x - 166 = 0$

So chosing $x = 2766.667$ can get the minimum delay, and the min. delay is:

$0.03x^2 - 166x + 644000|_{x=2766.667} = 414366.66(fs)$

b.

Let the width be c (um). Then, the delay by applying pi model is:

$t_{ab} = 100(0.3 \times 6000 \times c) + 6000(1/c)(0.1)(6000 \times c \times 0.15 + 20) = 600(900 + 20/c + 300c)$

By choosing $c = 1$, we can get the min. delay: $t_{ab} = 600 \times 1220 = 732000(fs)$

It is obviously that inserting buffer is more effective for delay optimization.

Problem 9 [Collaborator: None]

a.

Resistor per unit length: $r_0 = \rho/(H_i W_i) \propto S^2$

Capacitance per unit length: $c_0 = \epsilon W_i / t_{ox} \propto 1$

Local interconnection length: $l_l \propto 1/S$

Therefore,

Local RC delay: $r_0 c_0 l_l^2 = 1$

Global RC delay: $r_0 c_0 l_g^2 = S^2 D^2$

b.

As the feature size sclaing down, the global interconnection RC delay will dominate.

Problem 10 [Collaborator: None]

First we define the dimension of the variables:

$\mathbf{c} \in R^{1 \times n}$

$\mathbf{x} \in R^{n \times 1}$

$\mathbf{A} \in R^{m \times n}$

$\mathbf{b} \in R^{m \times 1}$

$\lambda \in R^{1 \times m}$

a.

$\mathbf{cx}$ and $\mathbf{Ax}$ must be positive coefficient polynomials.

b.

The KKT condition of the Lagrange function is:

$\frac{\partial L}{\partial \mathbf{x}} = c^T + A^T \lambda^T = 0$

$Ax - b \leq 0$

$\lambda(Ax - b) = 0$

$\lambda > 0$

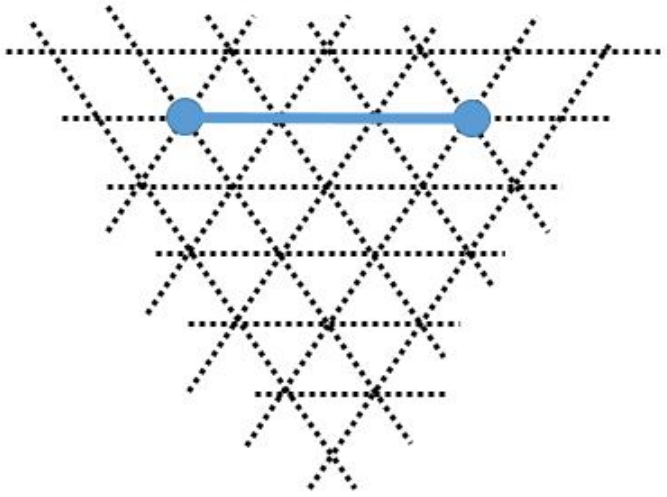
c.

If the current solution satisfy the KKT condition, then it is the optimal solution of the problem.

Problem 11 [Collaborator: None]

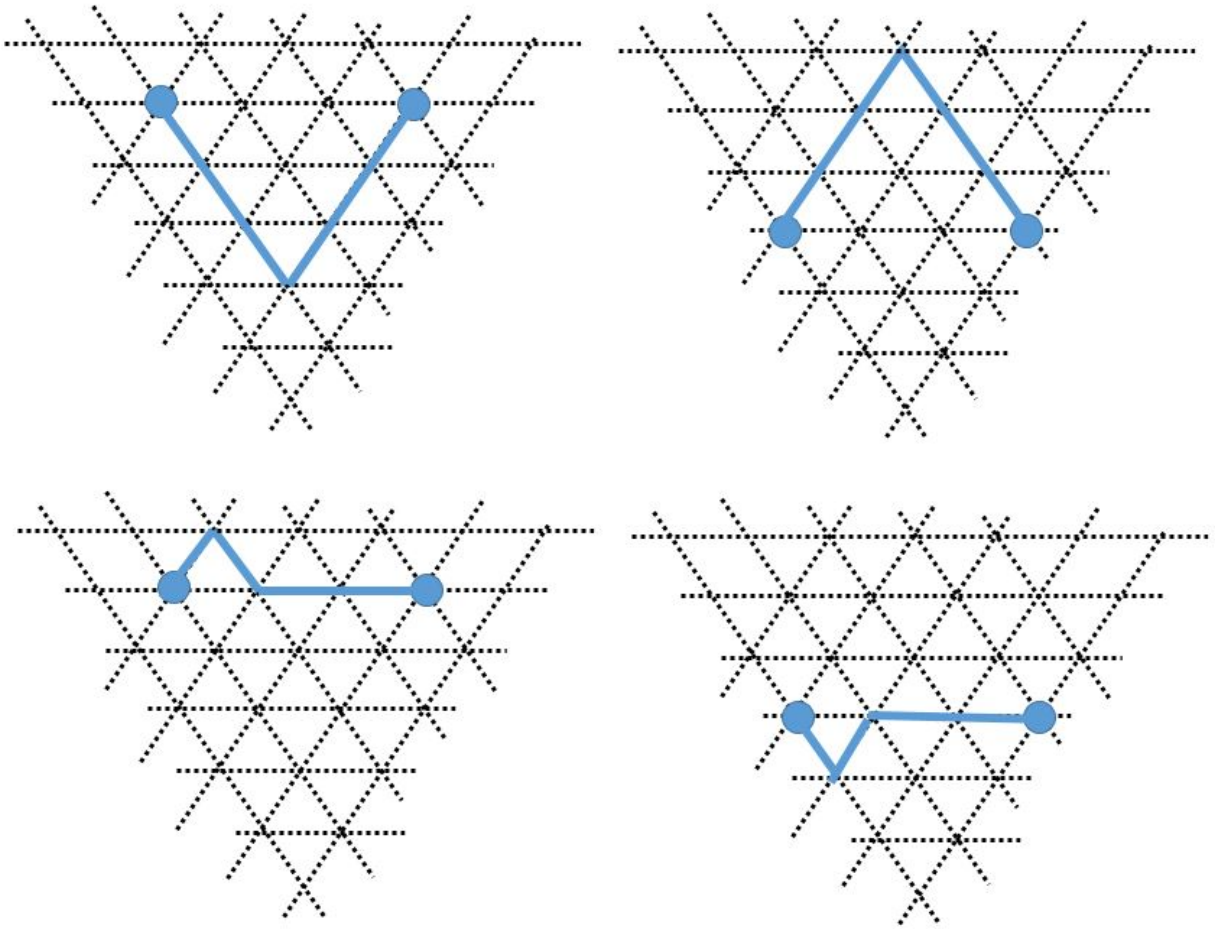
a.

Three types of conencting with 0 bends are shown in the figure below:



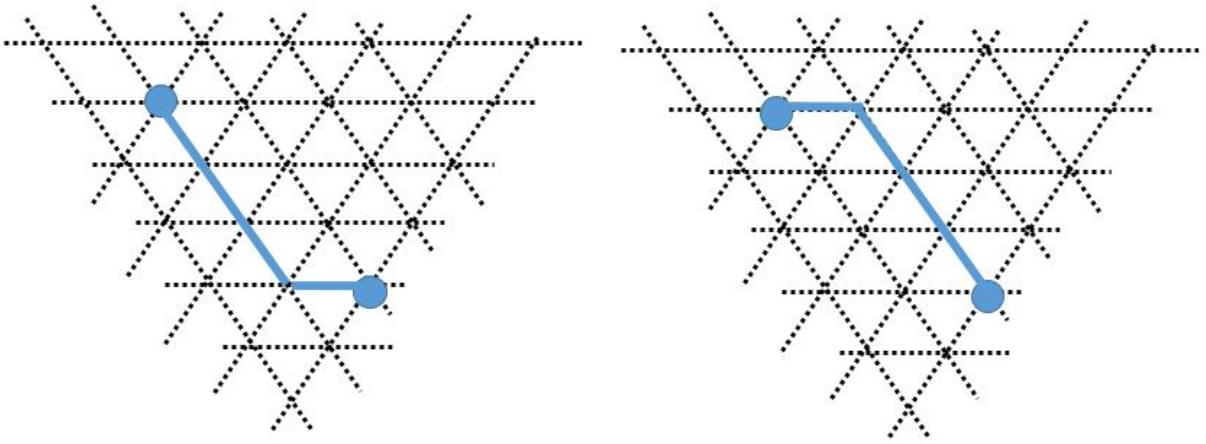
b.

Two types of conencting with 1 bends & 2 bends are shown in the figure below:



c.

Two possible routing topologies is shown in the figure below.

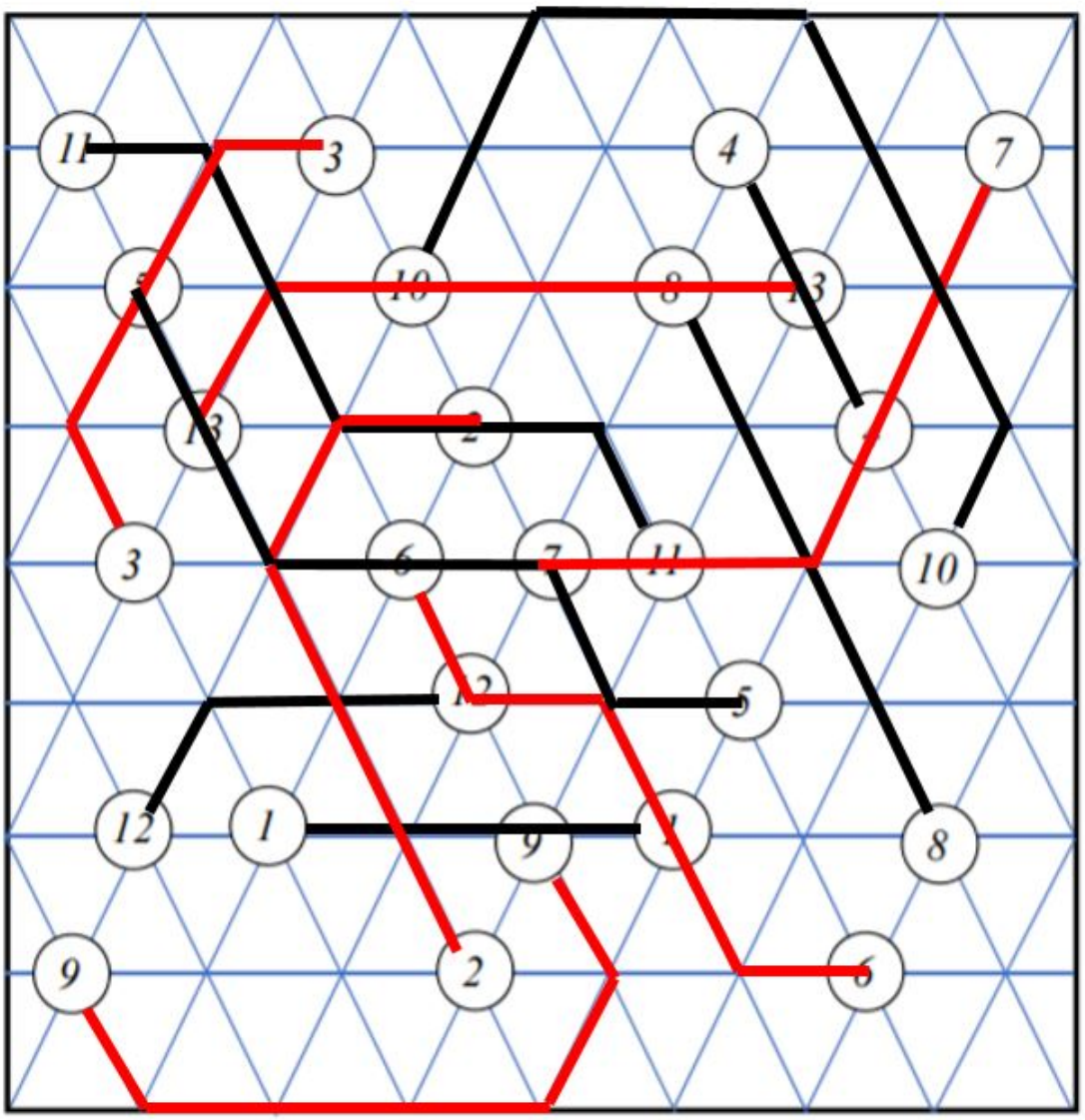


d.

The min. routing wire length for directly connecting each 2 pin net is 55.

We can easily proved that routing in only one layer is impossible by finding a cutline.

The following figure shows a solution with 2 routing layers.
And the resulting wirelength is 7 more than min. routung wire length, where net 9 contributes 2, net 10: 3, net 3: 1 and net 2: 1.
Therefore, the total wirelength is 62.



Problem 12 [Collaborator: None]

We come up with an $O(nlgn)$ idea for constructing a steiner tree.
Though it is not as accurate as minimum steiner tree, it forms a really high accuracy as RMST, which can be very useful in the placement stage due to its low complexity.
The psuedo code is as follows:

```
glob_color = 0
construct linked list Y based on increasing y order
construct linked list X based on increasing x order
construct_steiner(number of nodes, linked list X, linked list Y){
    ++glob_color
    if(number of nodes <= 3)
        connect the three point by simple steiner tree.
    Find range(x) and range(y) by simply traverse through the two linked list
    // We split the larger range direction and the two component is coonected by the
    //     geometry of the two clusters
    if(split in x direction)
        maintain two pointers mid_pointer, end_pointer
        while(end_pointer != end of list)
            end_pointer += 2 // jump to next of next node in the list
            mid_pointer += 1
            all nodes traversed by mid pointer is colored glob_color
        Cut linked list X into X1 and X2 by the mid pointer
        traverse through linked list Y and formed the colored node
        into link list Y1 and the remain to Y2
    else
        ... // similar to splitting in z direction
    record num_nodes_cluster1, num_nodes_cluster2
    construct_steiner(num_nodes_cluster1, X1, Y1)
    construct_steiner(num_nodes_cluster2, X2, Y2)
    // return the outest node in 4 direction by simply comparing all traverse
    // node
    return four outest node in 45, 135, -45, -135 direction
}
Run construct_steiner(number of nodes, X, Y)
```

The complexity of this algorithm can be calculate by master's thm:
$T(n) = 2T(n/2) + O(n)$
So, $O(nlgn)$ complexity.

A simple comparison table is shown below.

	Complexity	**Accuracy**
RMST	V^2lgV	high
Steiner	exp.	highest of all
Our	$VlgV$	high

A simle example is shown in the figure below: