

# Lecture 10: Floorplan – II

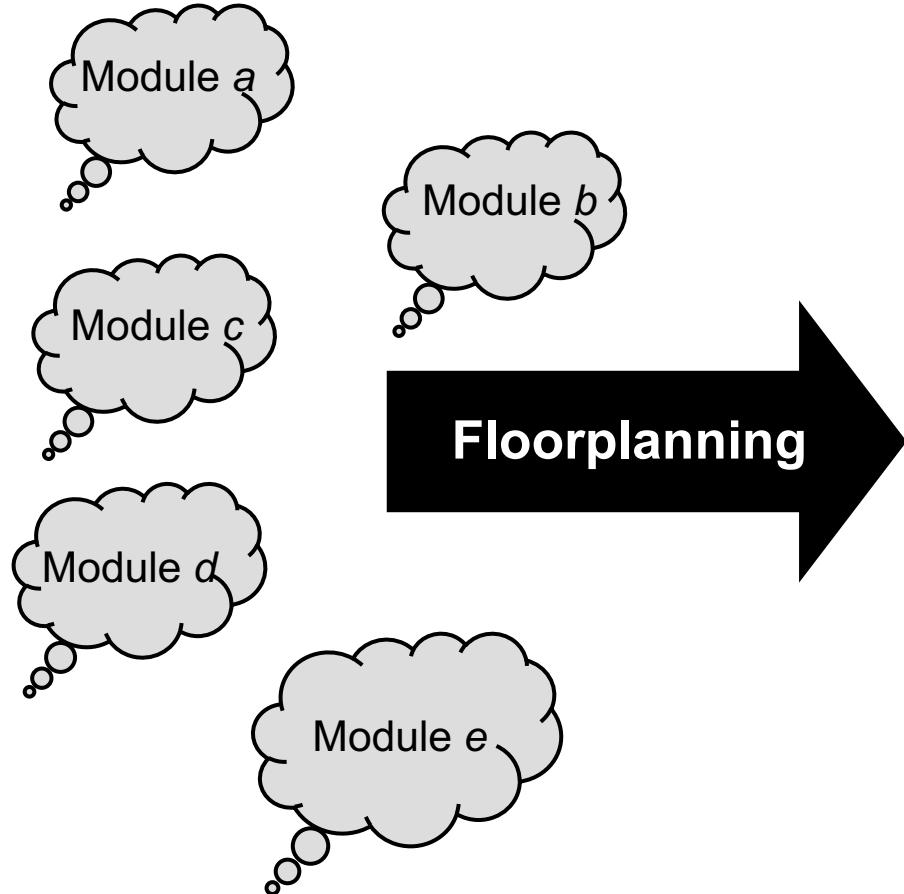
---

Tsung-Wei (TW) Huang

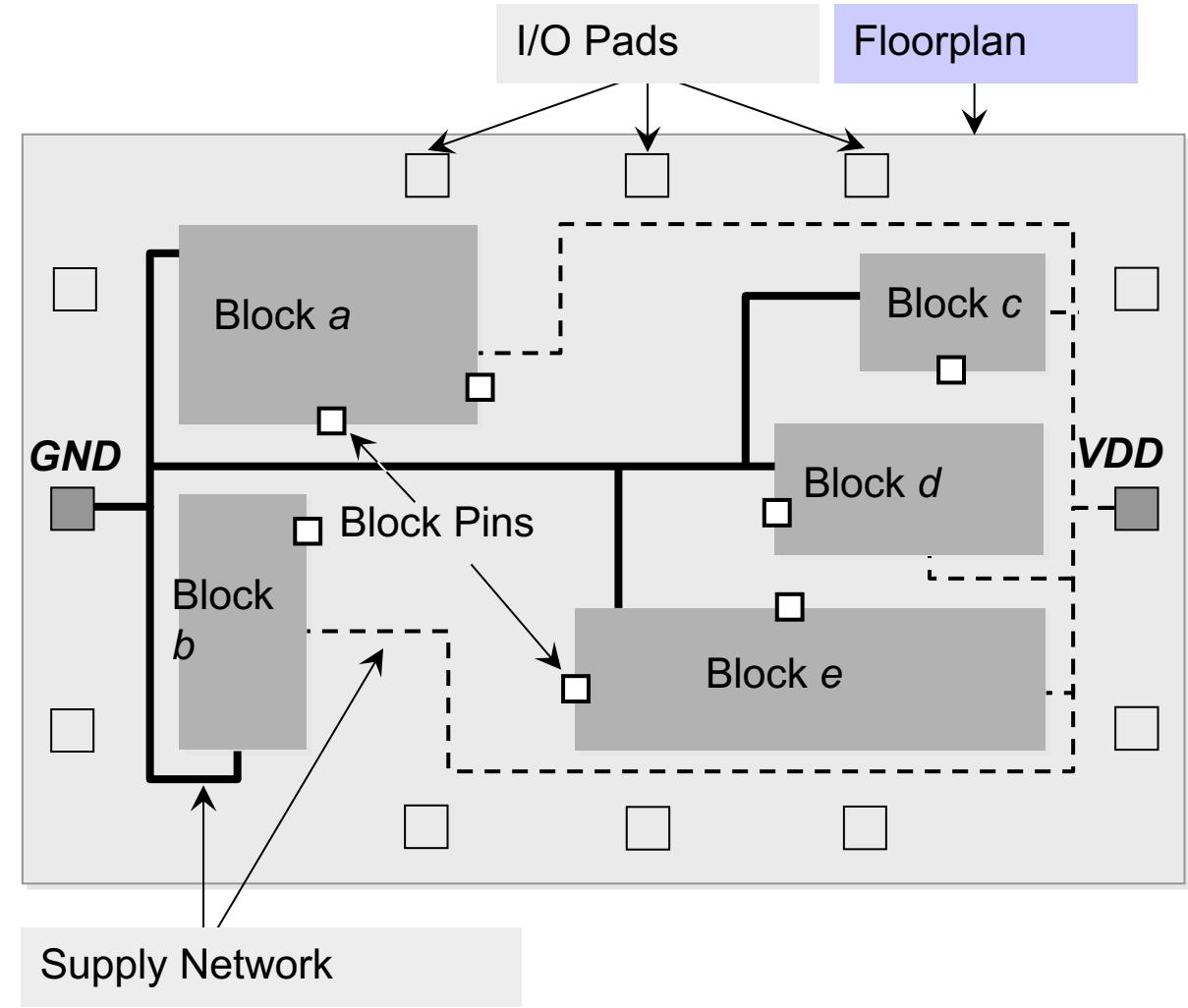
Department of Electrical and Computer Engineering  
University of Utah, Salt Lake City, UT



# Recap: Floorplanning



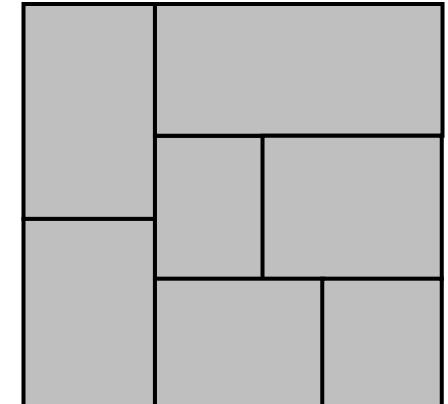
Partitioned modules (e.g., FM algorithm)



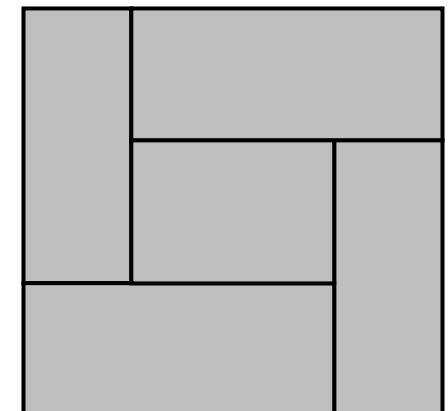
# Recap: Slicing Floorplan

---

- **Slicing floorplan**
  - One that can be obtained by repetitively subdividing (slicing) rectangles horizontally or vertically
- **Non-slicing floorplan**
  - One that may not be obtained by repetitively subdividing alone
- **Slicing floorplans are much easier to handle**
  - Efficient data structures exist for efficient computational manipulations



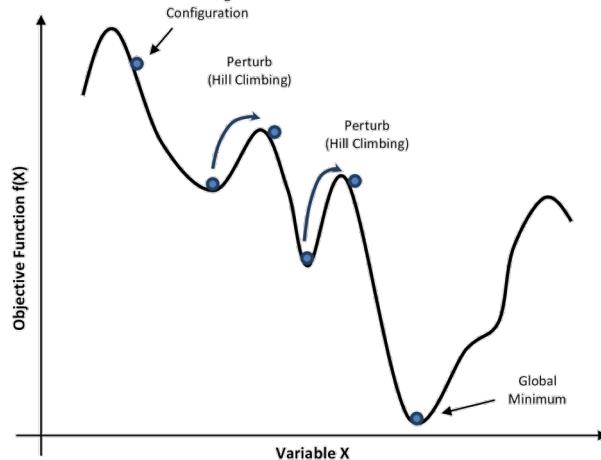
Slicing Floorplan



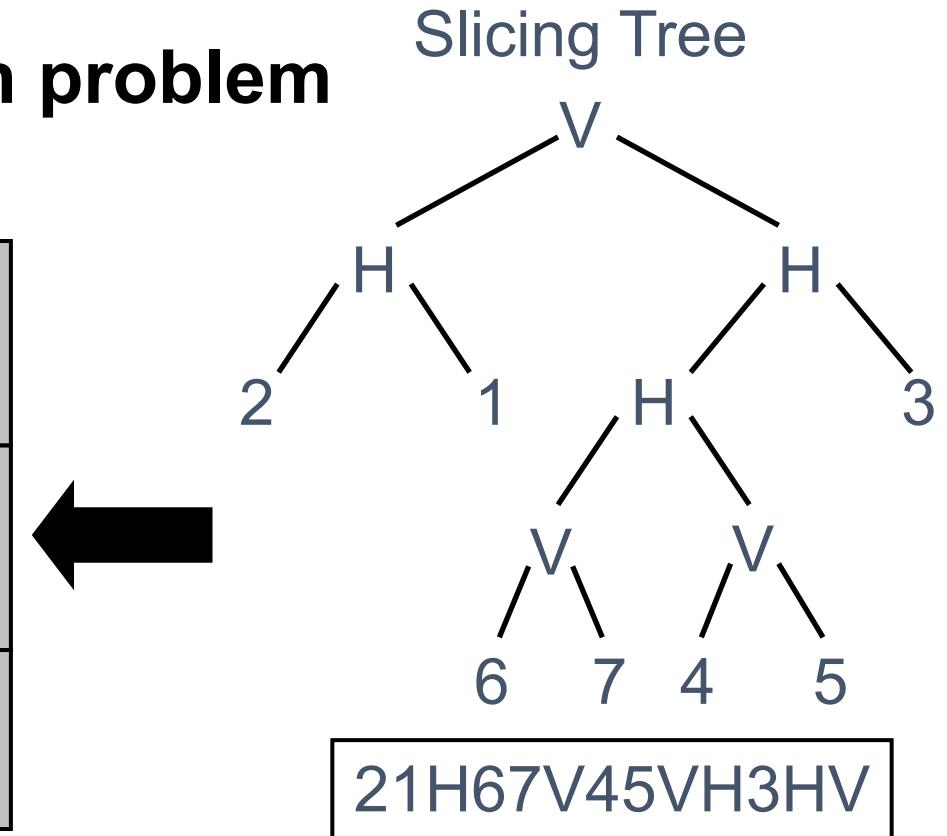
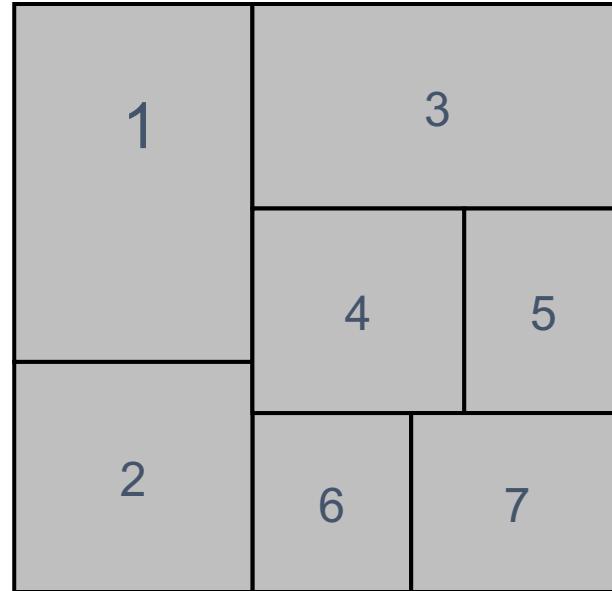
Non-slicing Floorplan

# Big Idea in Floorplan Optimization

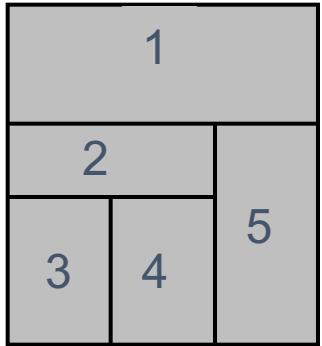
- Design efficient linear state representation for 2D layout
  - Polish expression (PE) for slicing floorplan
- Apply SA to optimize state search problem



Slicing Floorplan

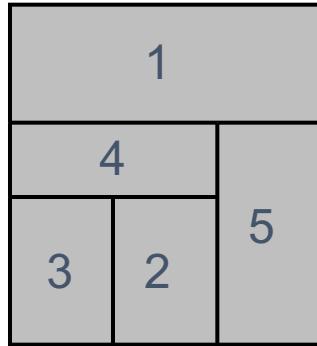


# Simulated Annealing-based Optimization

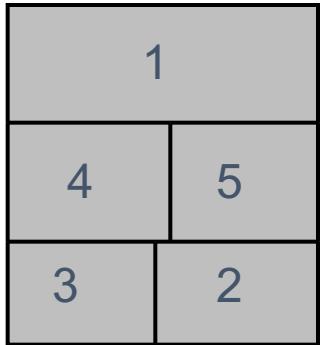


34V2H5V1H

M1  
→



32V4H5V1H



32V45VH1H

M2  
←

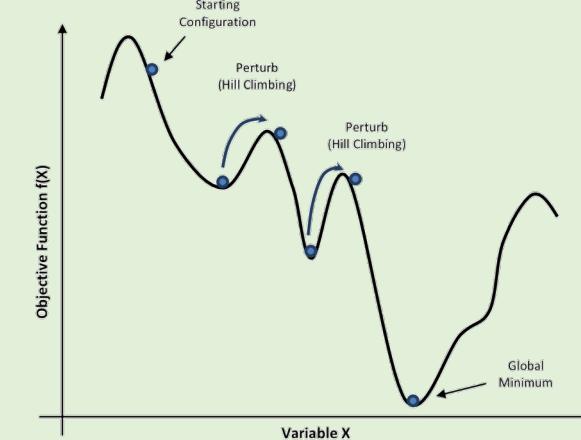


32V45HV1H

M3  
↷

Each move gives a new floorplan result

Key idea: Leverage simulated annealing to optimize moves



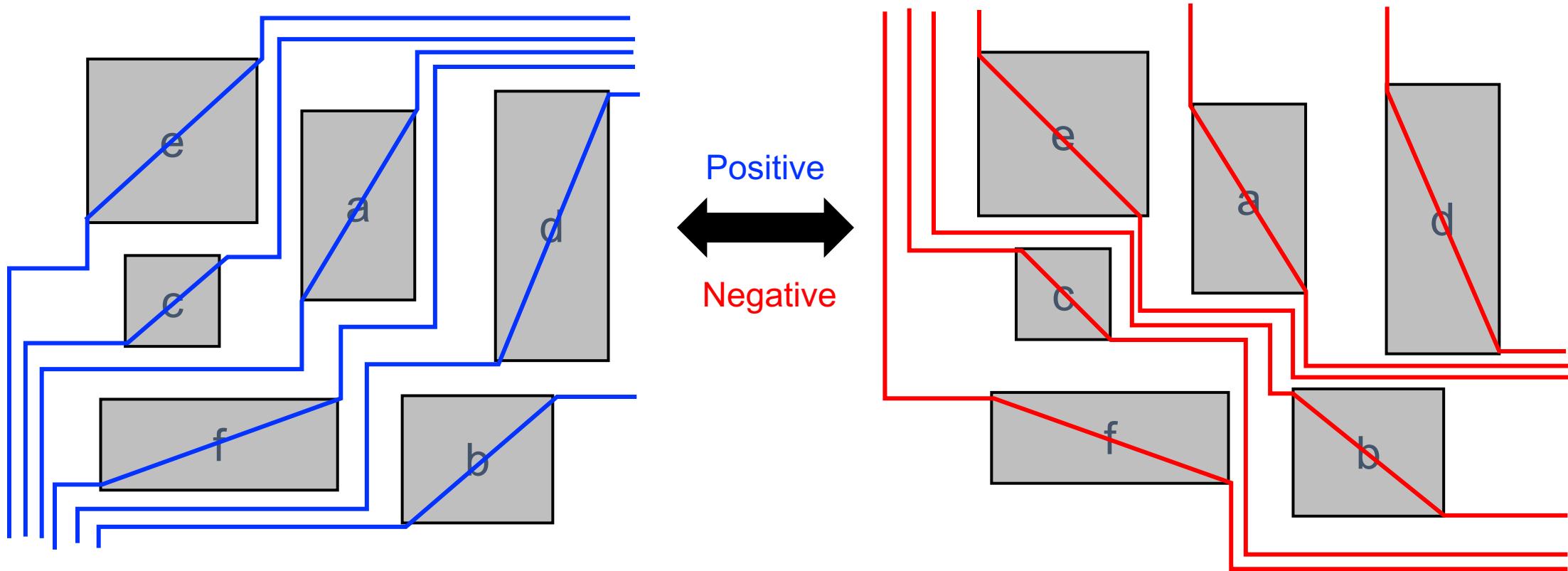
# Sequence Pair

---

- H. Murata, K. Fujiyoshi, S. Nakatake and Y. Kajitani, “VLSI module placement based on rectangle-packing by the sequence-pair,” in **IEEE TCAD, vol. 15, no. 12, pp. 1518-1524, Dec. 1996**
  - One of the most influential papers in the computer design industry
  - Widely used by industrial tools
- **Sequence pair is a succinct representation of non-slicing floorplan**
  - Just like PE for slicing floorplan
- **Leverage SA to find a good sequence pair for floorplan optimization**

# Sequence Pair (cont'd)

- Positive sequence: lower-left to upper-right, i.e., ecafdb
- Negative sequence: upper-left to lower-right, i.e., fcbead



# Positive Sequence Construction

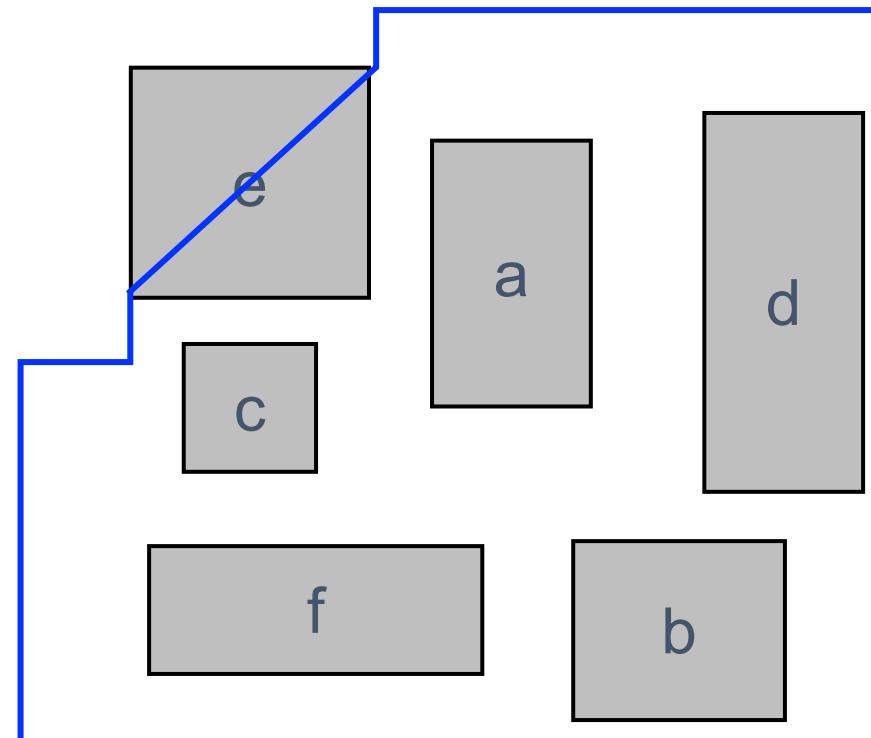
---

- To go from a floorplan to its sequence pair, we first draw **up-right steplines** for each module as follows:
  - Starting from the upper right corner of a module until we reach the upper right corner of the floorplan, we draw vertical lines (going up) and horizontal lines (going right) in an alternative fashion so that we don't cross any module boundary in the floorplan and any previously drawn line
- We also draw **down-left steplines** in a similar way but in an opposite direction
- Negative Sequence can be constructed in a similar way

# Positive Sequence Construction

---

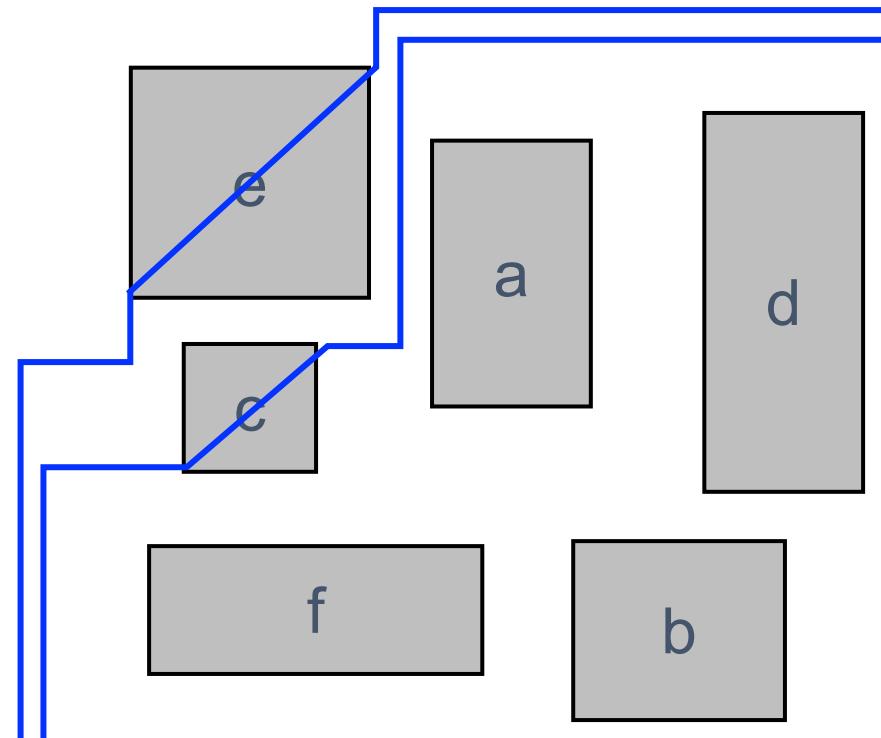
- Positive stepline sequence: e



# Positive Sequence Construction

---

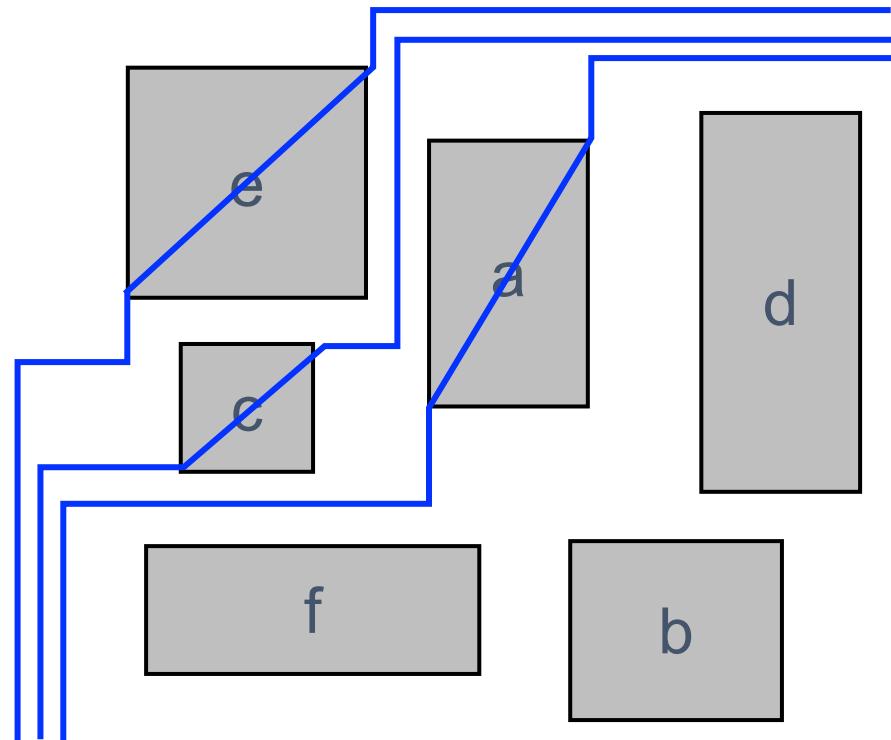
- Positive stepline sequence: ec



# Positive Sequence Construction

---

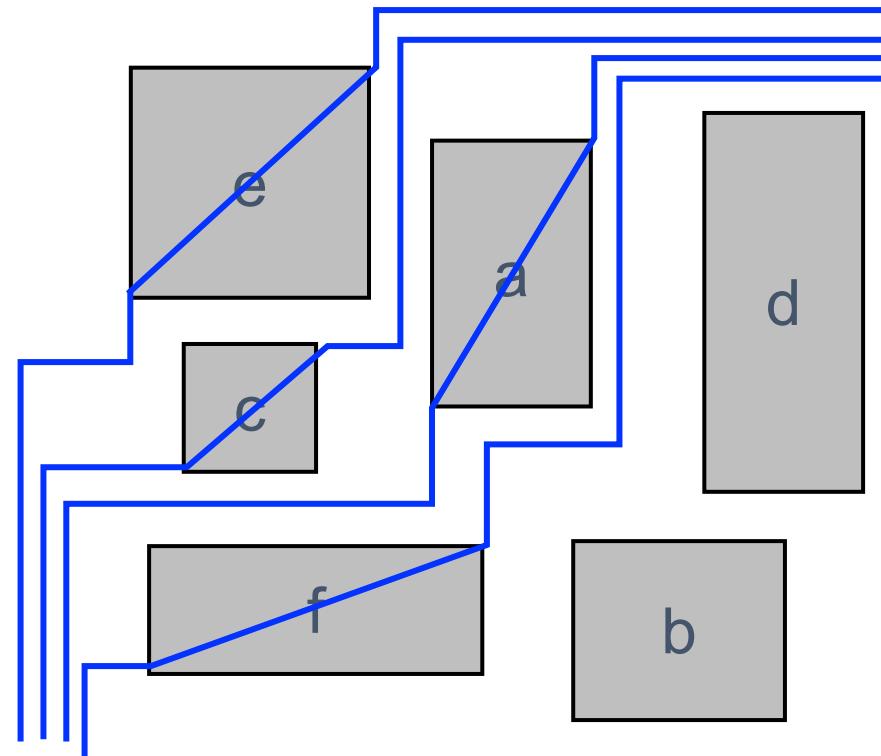
- Positive stepline sequence: eca



# Positive Sequence Construction

---

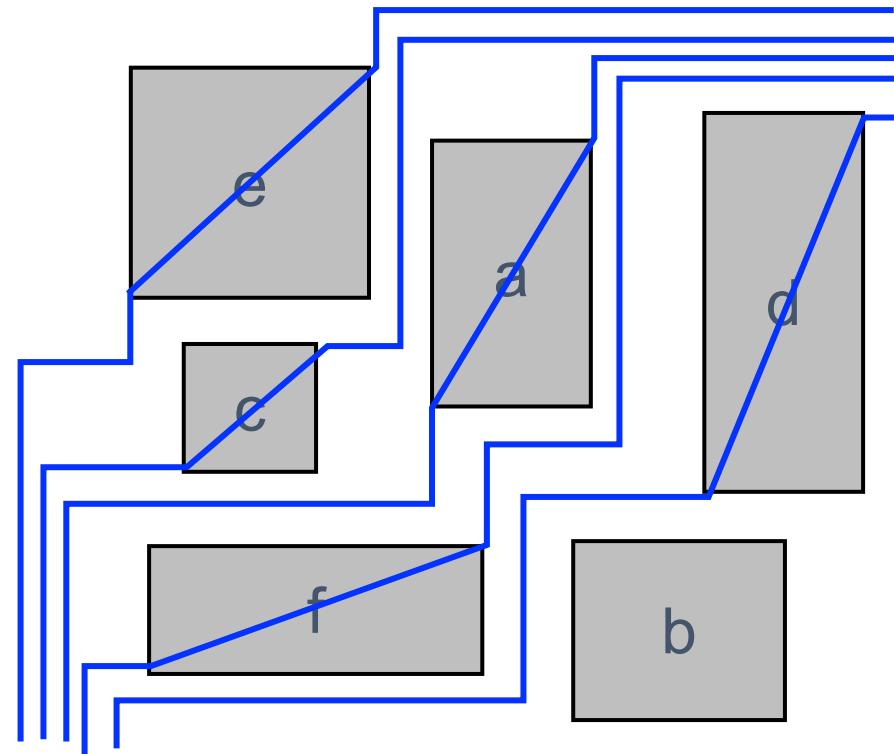
- Positive stepline sequence: ecaf



# Positive Sequence Construction

---

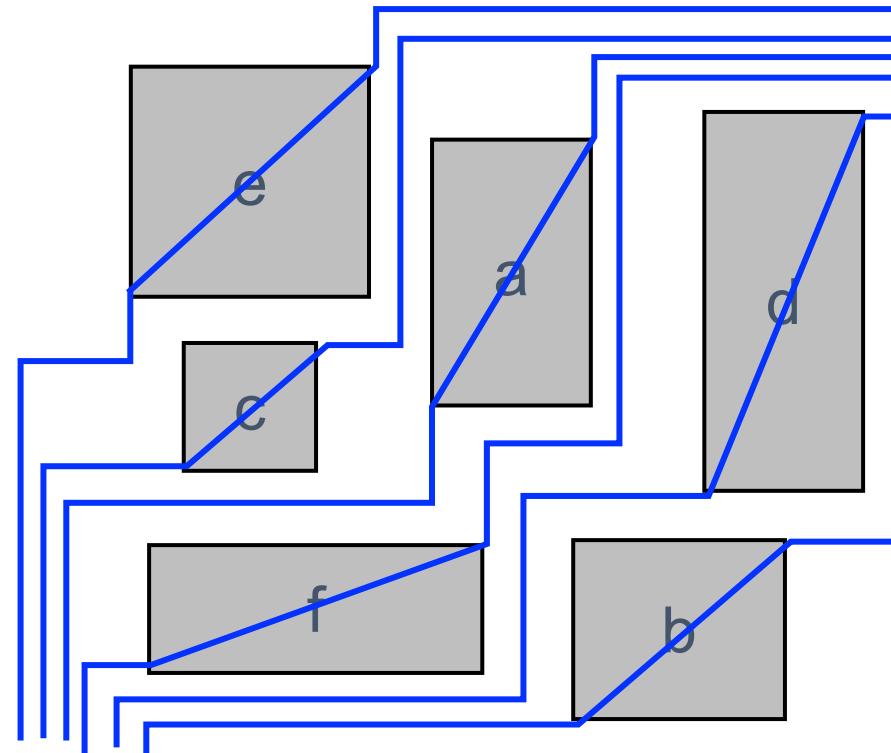
- Positive stepline sequence: ecafdf



# Positive Sequence Construction

---

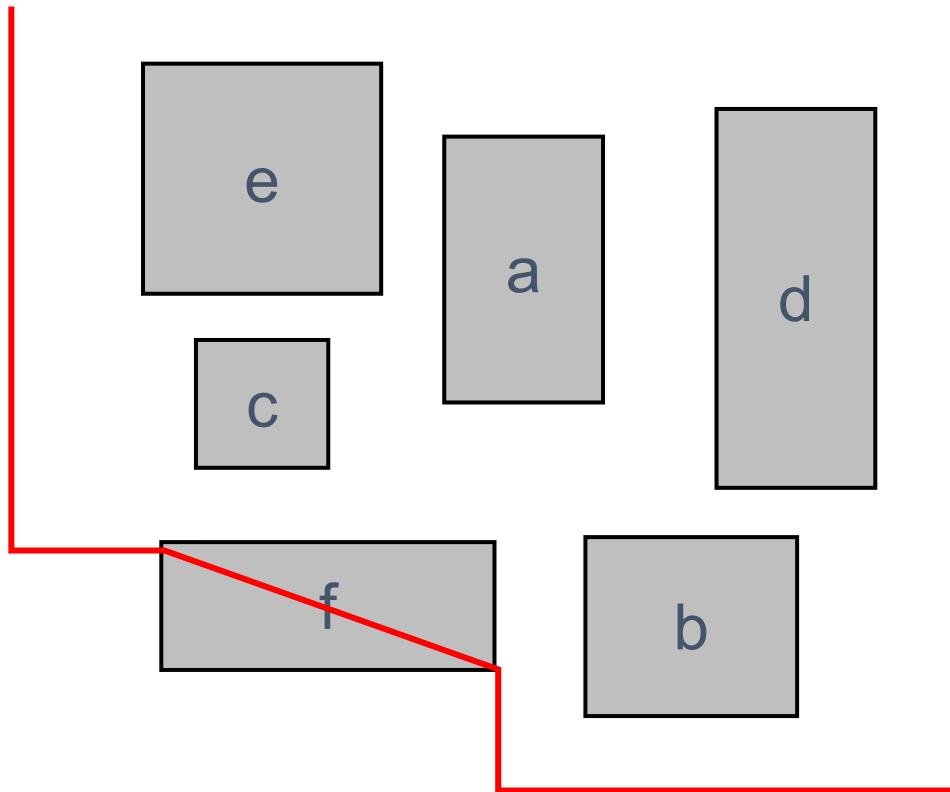
- Positive stepline sequence: ecafdb



# Negative Sequence Construction

---

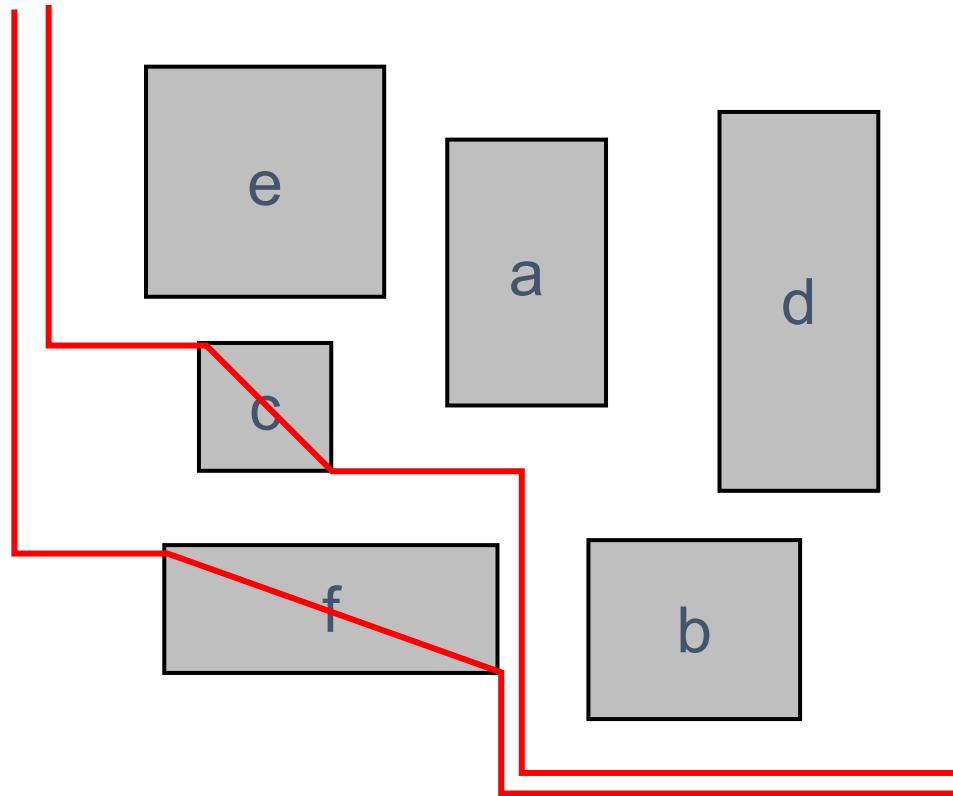
- Negative stepline sequence: f



# Negative Sequence Construction

---

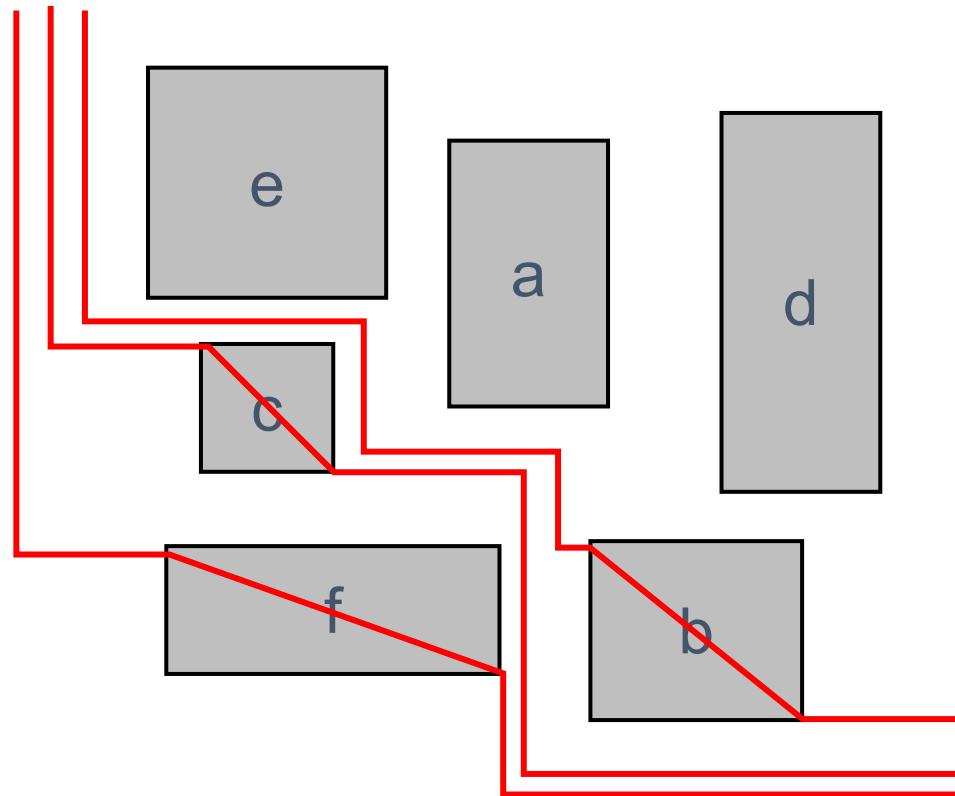
- Negative stepline sequence: fc



# Negative Sequence Construction

---

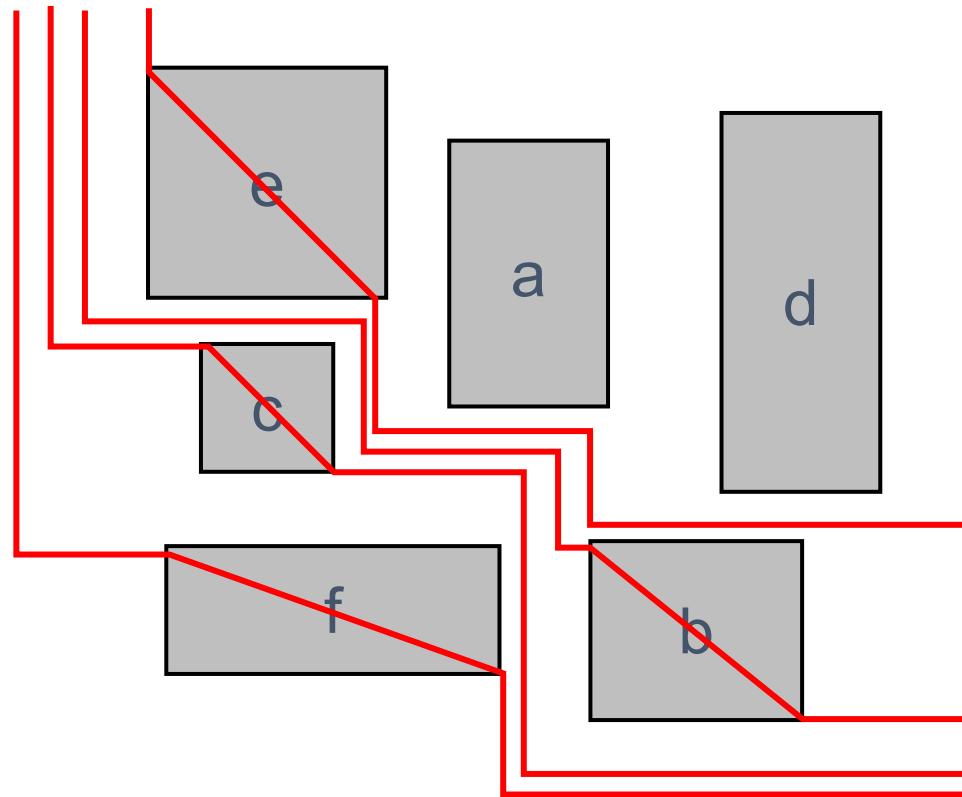
- Negative stepline sequence: fcb



# Negative Sequence Construction

---

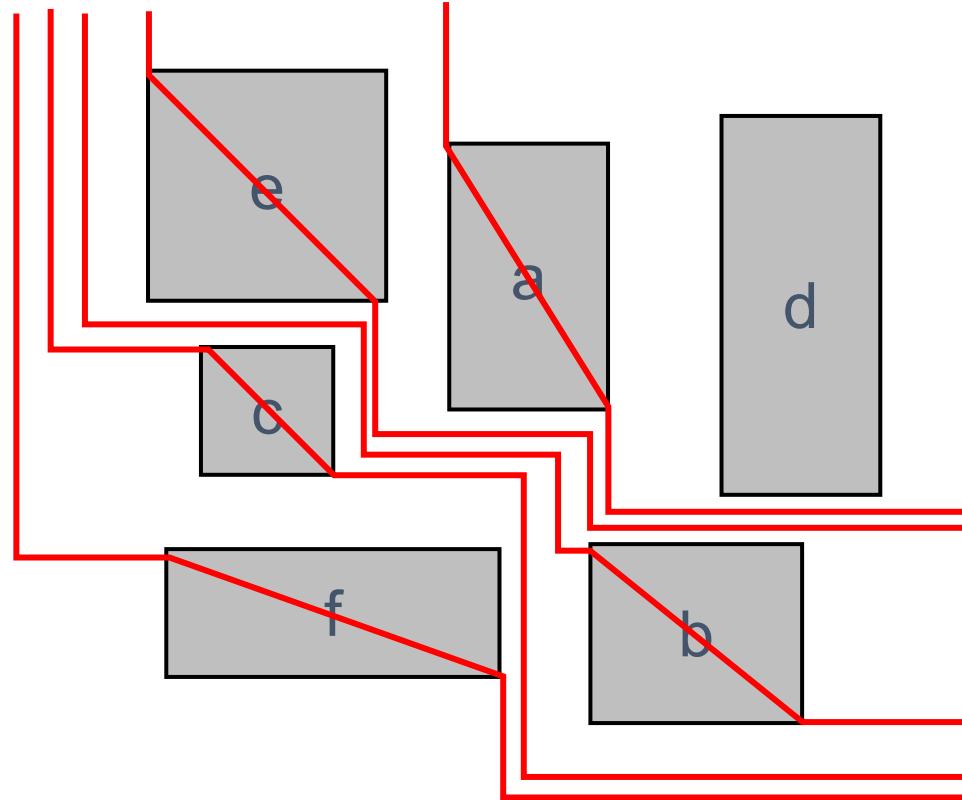
- Negative stepline sequence: fcbe



# Negative Sequence Construction

---

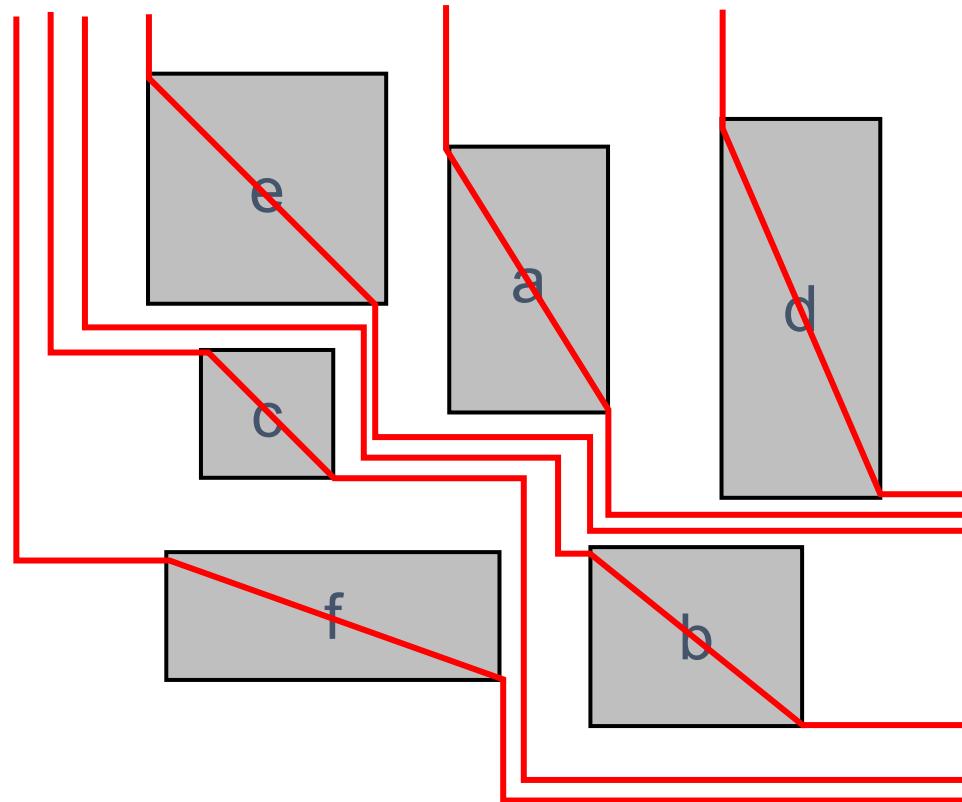
- Negative stepline sequence: fcbea



# Negative Sequence Construction

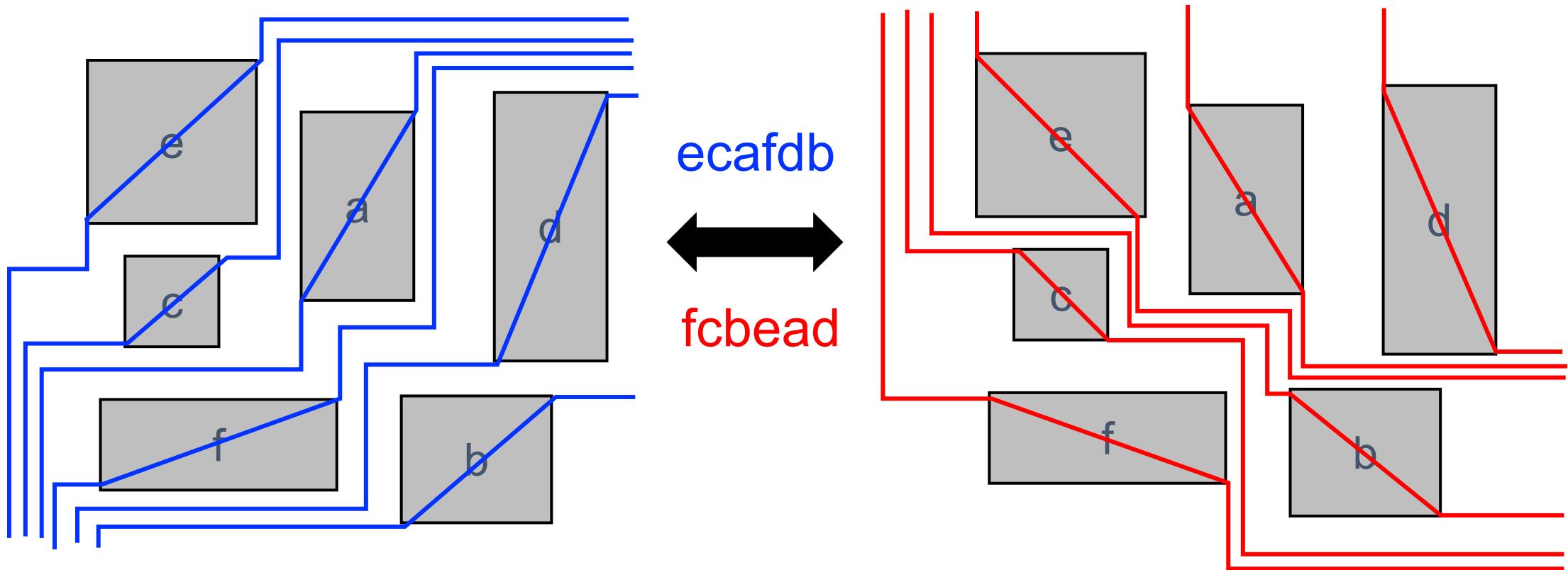
---

- Negative stepline sequence: fcbead



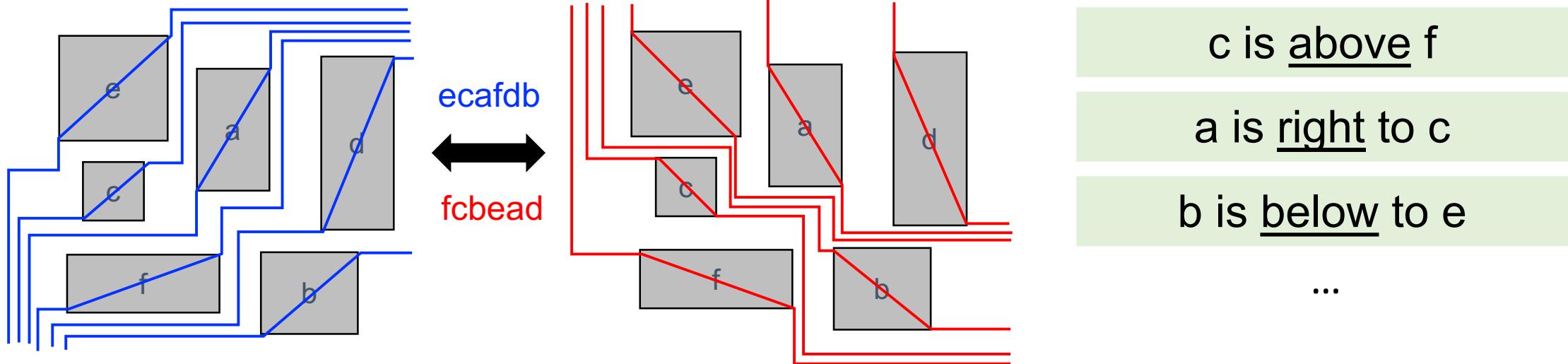
# Sequence Pair Property

- The authors showed that the positive/negative sequence of modules always exist and do not cross each other



# Geometric Info of a Sequence Pair

- Given a sequence pair ( $P, N$ )
  - $a$  is right to  $b$ , iff  $a$  is after  $b$  in both  $P$  and  $N$
  - $a$  is left to  $b$ , iff  $a$  is before  $b$  in both  $P$  and  $N$
  - $a$  is above  $b$ , iff  $a$  is before  $b$  in  $P$  and  $a$  is after  $b$  in  $N$
  - $a$  is below  $b$ , iff  $a$  is after  $b$  in  $P$  and  $a$  is before  $b$  in  $N$



# Solution Perturbation

---

- **M1: swap a random pair of modules in the first sequence**
  - (ecafdb, fcbead) → (eacfdb, fcbead)
- **M2: swap a random pair of modules in both sequences**
  - (ecafdb, fcbead) → (edafcb, fdbeacc)
- **M3: rotate a randomly selected module**
  - (ecafdb, fcbead), (W<sub>c</sub>, H<sub>c</sub>) → (ecafdb, fcbead), (H<sub>c</sub>, W<sub>c</sub>)
- **Wait ... how to recover the floorplan from a sequence pair?**

# Relative Locations

a is right to b, iff a is after b in both P and N  
a is left to b, iff a is before b in both P and N  
a is above b, iff a is before b in P and a is after b in N  
a is below b, iff a is after b in P and a is before b in N

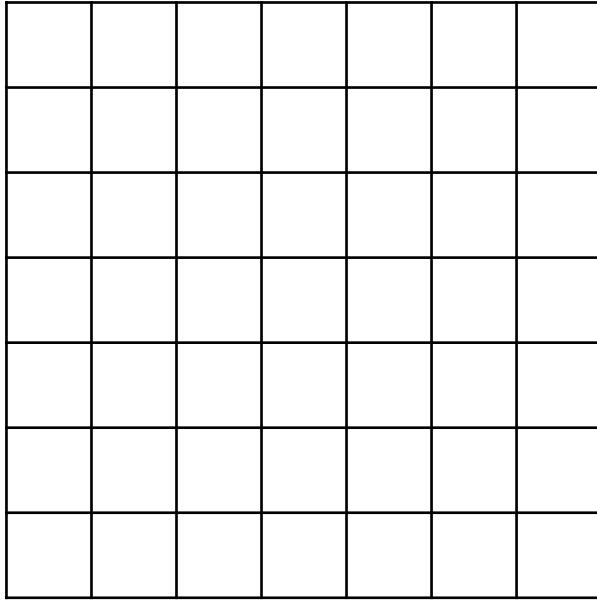
- SP = (17452638, 84725361)

- Dimensions (W, H): (2,4), (1,3), (3,3), (3,5), (3,2), (5,3), (1,2), (2,4)

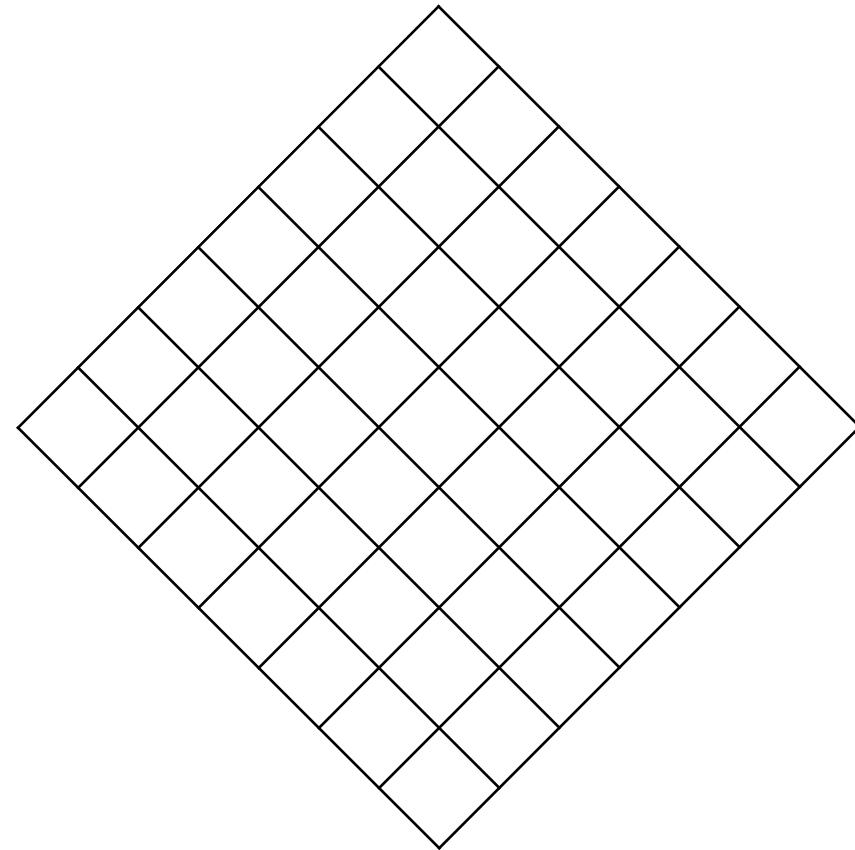
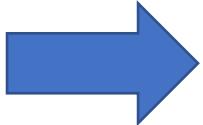
module	right-of	left-of	above	below
1	∅	∅	∅	{2, 3, 4, 5, 6, 7, 8}
2	{3, 6}	{4, 7}	{1, 5}	{8}
3	∅	{2, 4, 5, 7}	{1, 6}	{8}
4	{2, 3, 5, 6}	∅	{1, 7}	{8}
5	{3, 6}	{4, 7}	{1}	{2, 8}
6	∅	{2, 4, 5, 7}	{1}	{3, 8}
7	{2, 3, 5, 6}	∅	{1}	{4, 8}
8	∅	∅	{1, 2, 3, 4, 5, 6, 7}	∅

# Module Locations on a Grid

---



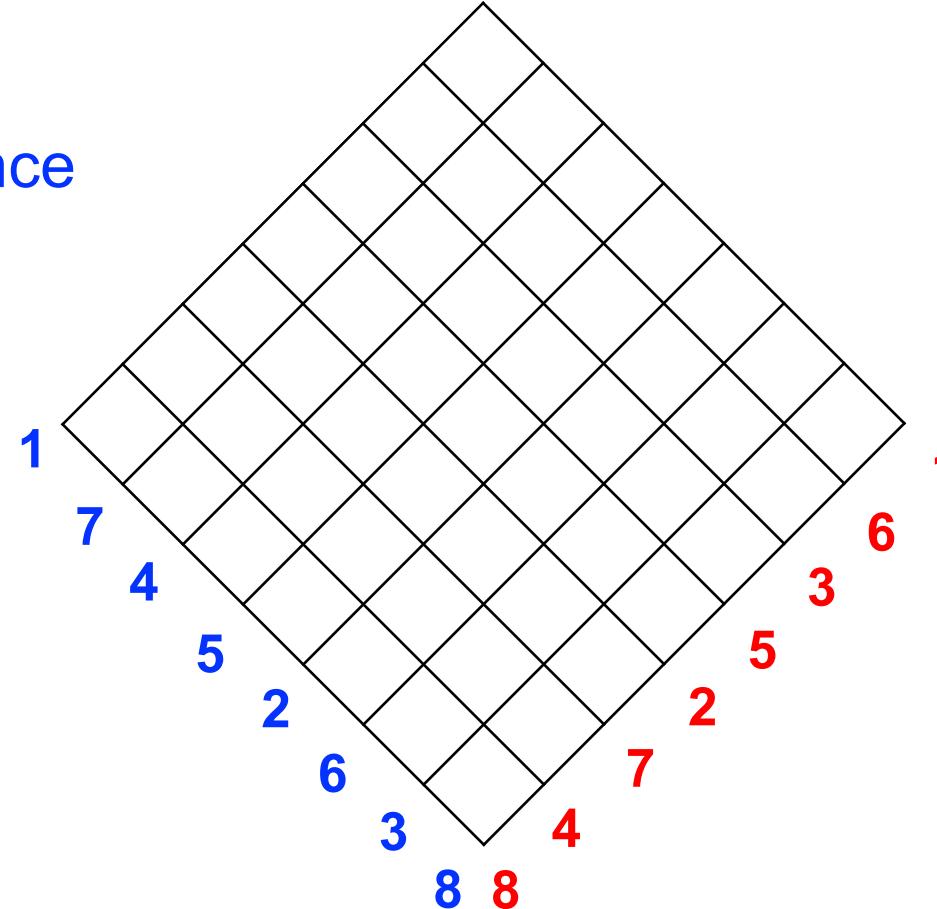
Rotate 45 degree



# Module Locations on a Grid

---

Positive sequence  
(17452638)

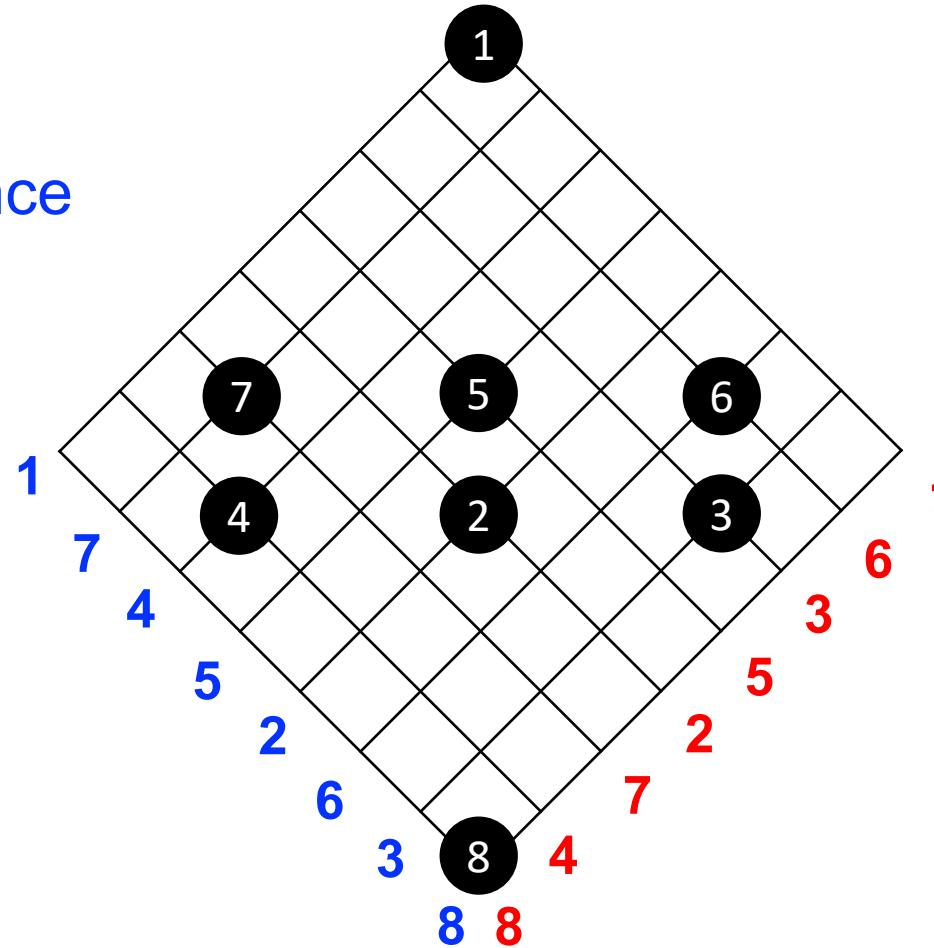


Negative sequence  
(84725361)

# Module Locations on a Grid

---

Positive sequence  
(17452638)



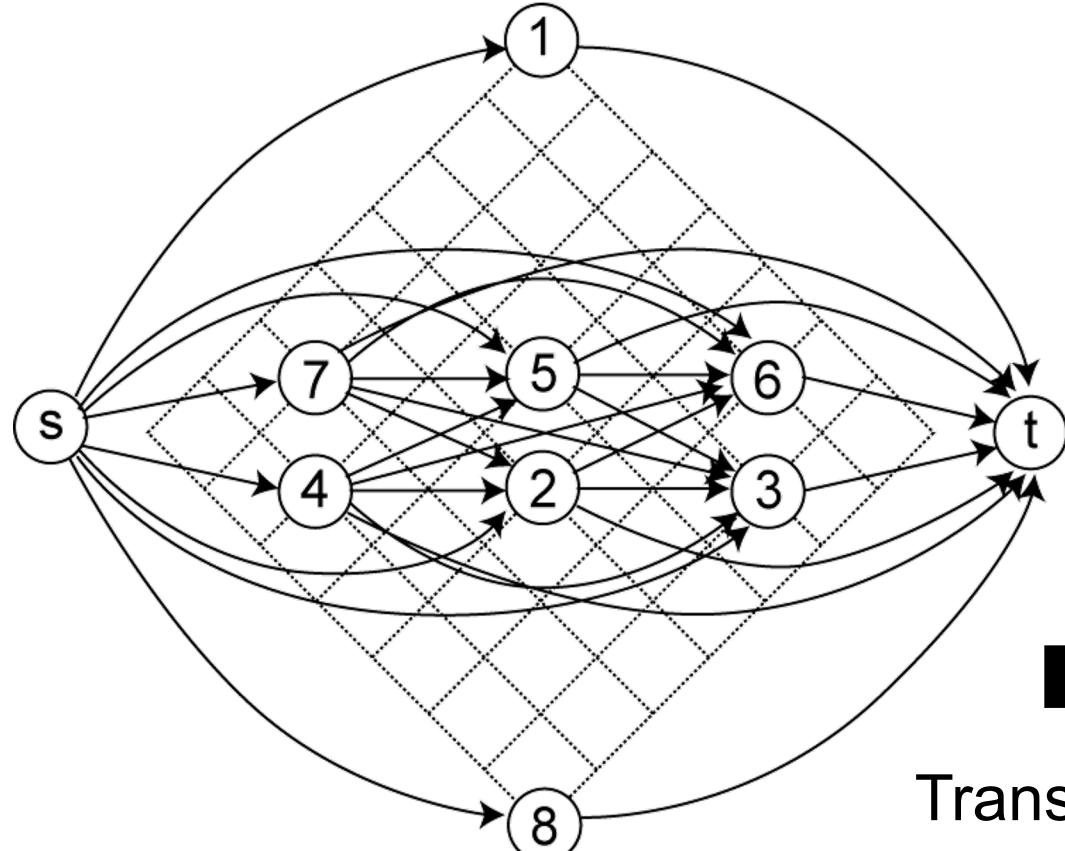
Negative sequence  
(84725361)

# Horizontal Constraint Graph (HCG)

- Add a directed edge  $m_1 \rightarrow m_2$  if  $m_2$  is right to module  $m_1$

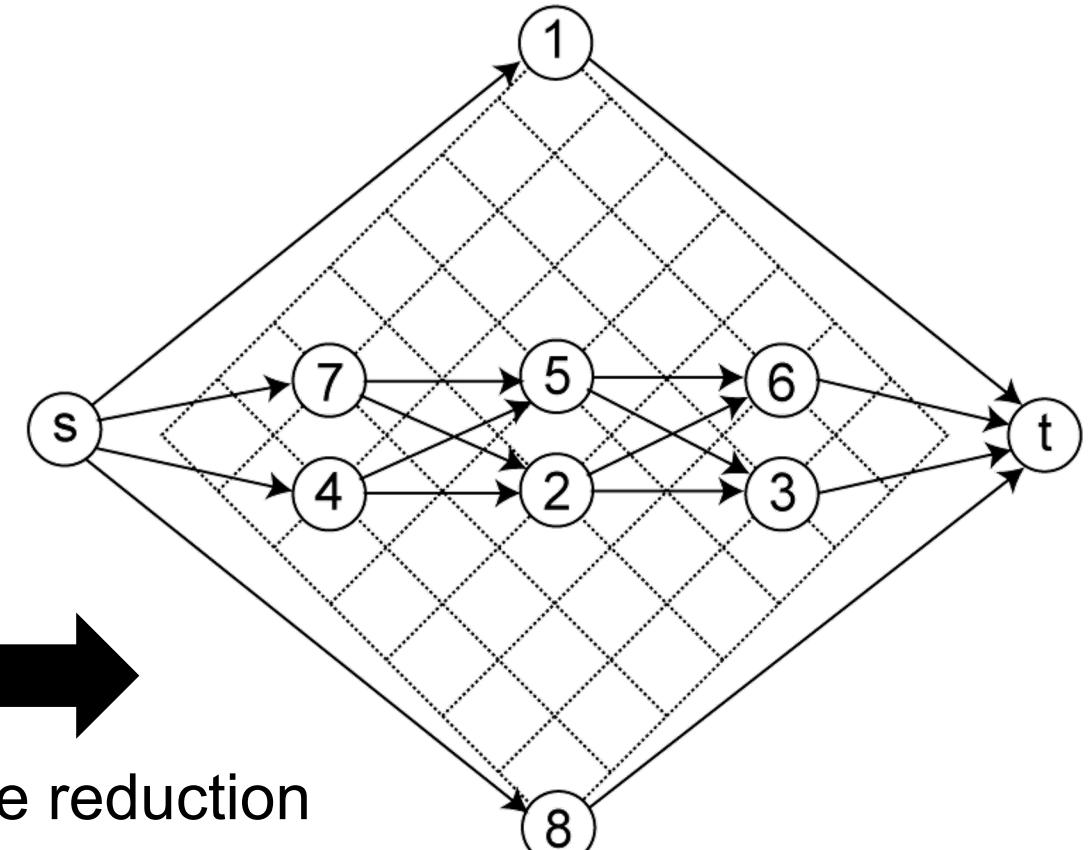
module	right-of	left-of	above	below
1	$\emptyset$	$\emptyset$	$\emptyset$	{2, 3, 4, 5, 6, 7, 8}
2	{3, 6}	{4, 7}	{1, 5}	{8}
3	$\emptyset$	{2, 4, 5, 7}	{1, 6}	{8}
4	{2, 3, 5, 6}	$\emptyset$	{1, 7}	{8}
5	{3, 6}	{4, 7}	{1}	{2, 8}
6	$\emptyset$	{2, 4, 5, 7}	{1}	{3, 8}
7	{2, 3, 5, 6}	$\emptyset$	{1}	{4, 8}
8	$\emptyset$	$\emptyset$	{1, 2, 3, 4, 5, 6, 7}	$\emptyset$

# Horizontal Constraint Graph (cont'd)



(a)

Transitive reduction



(b)

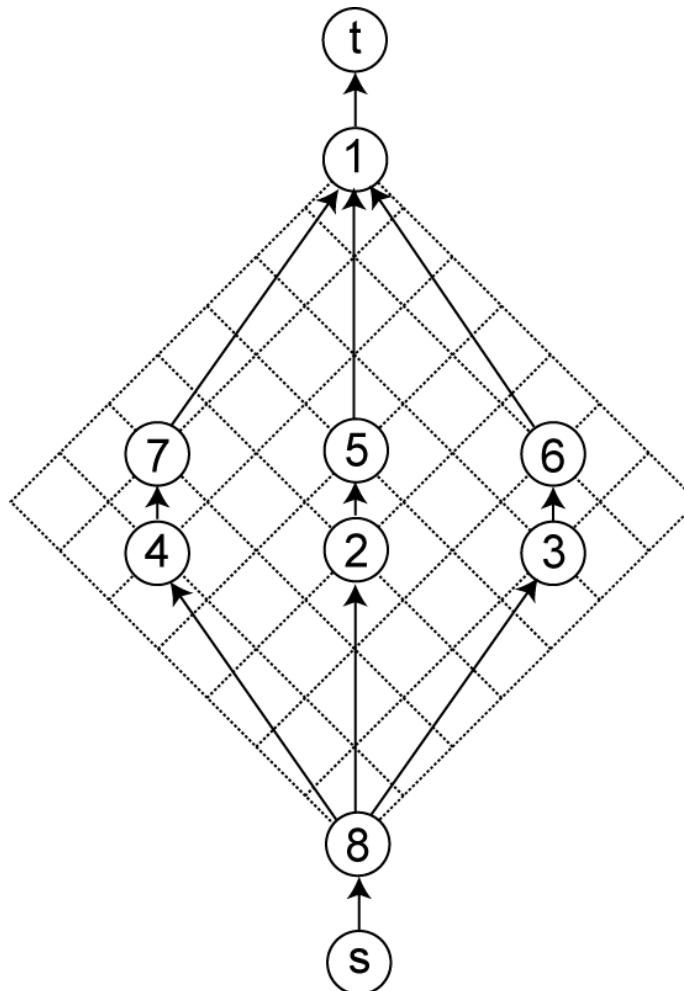
# Vertical Constraint Graph (VCG)

- Add a directed edge  $m_1 \rightarrow m_2$  if  $m_2$  is above module  $m_1$

module	right-of	left-of	above	below
1	$\emptyset$	$\emptyset$	$\emptyset$	$\{2, 3, 4, 5, 6, 7, 8\}$
2	$\{3, 6\}$	$\{4, 7\}$	$\{1, 5\}$	$\{8\}$
3	$\emptyset$	$\{2, 4, 5, 7\}$	$\{1, 6\}$	$\{8\}$
4	$\{2, 3, 5, 6\}$	$\emptyset$	$\{1, 7\}$	$\{8\}$
5	$\{3, 6\}$	$\{4, 7\}$	$\{1\}$	$\{2, 8\}$
6	$\emptyset$	$\{2, 4, 5, 7\}$	$\{1\}$	$\{3, 8\}$
7	$\{2, 3, 5, 6\}$	$\emptyset$	$\{1\}$	$\{4, 8\}$
8	$\emptyset$	$\emptyset$	$\{1, 2, 3, 4, 5, 6, 7\}$	$\emptyset$

# Vertical Constraint Graph (cont'd)

---



# Computing Floorplan Width and Height

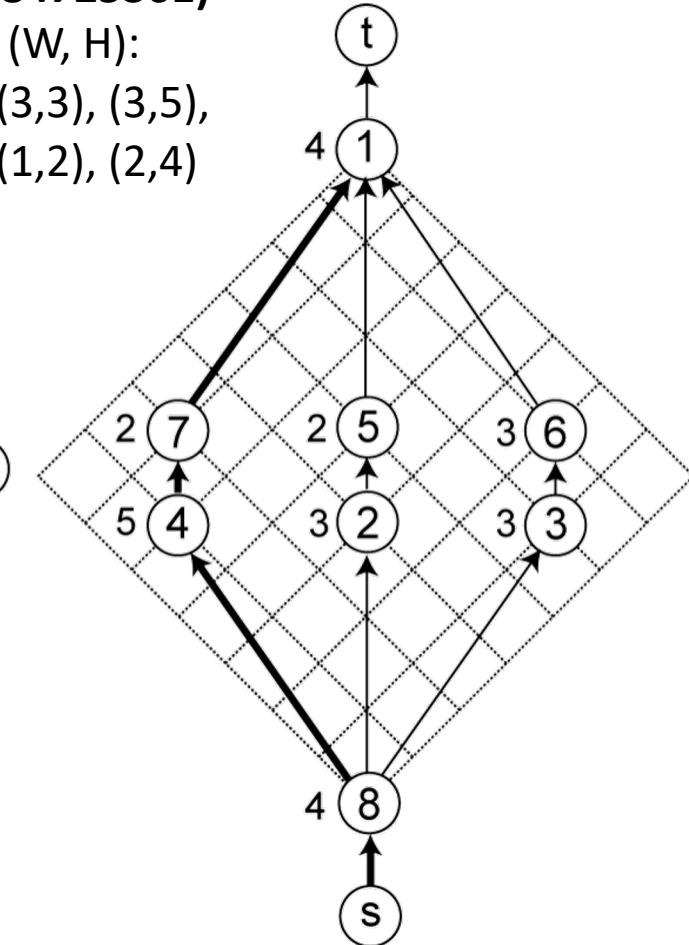
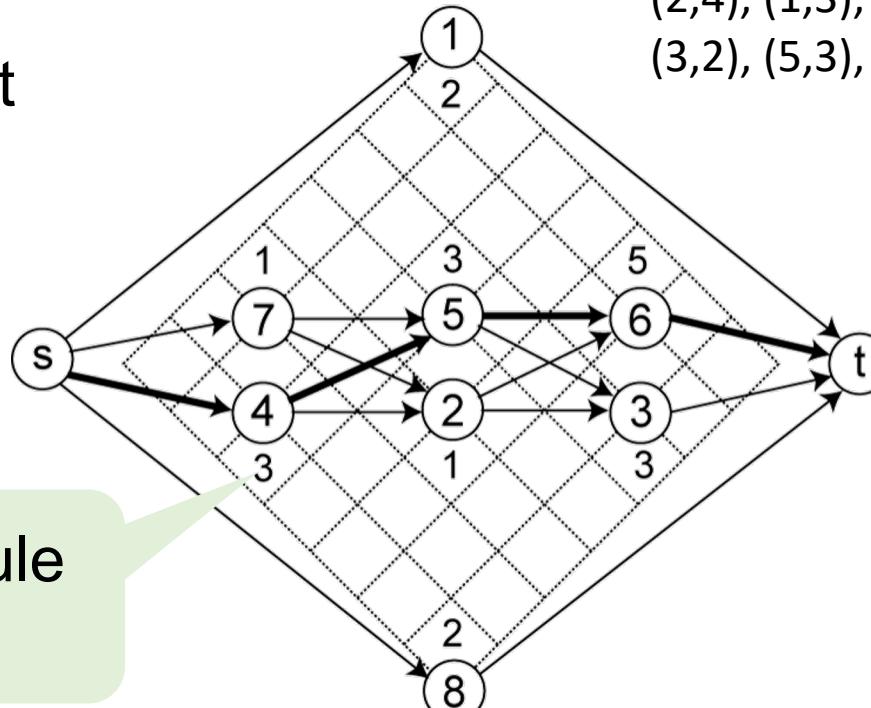
- Longest s-t path in
  - HCG: chip width
  - VCG: chip height

$SP = (17452638, 84725361)$

Dimensions (W, H):

(2,4), (1,3), (3,3), (3,5),  
(3,2), (5,3), (1,2), (2,4)

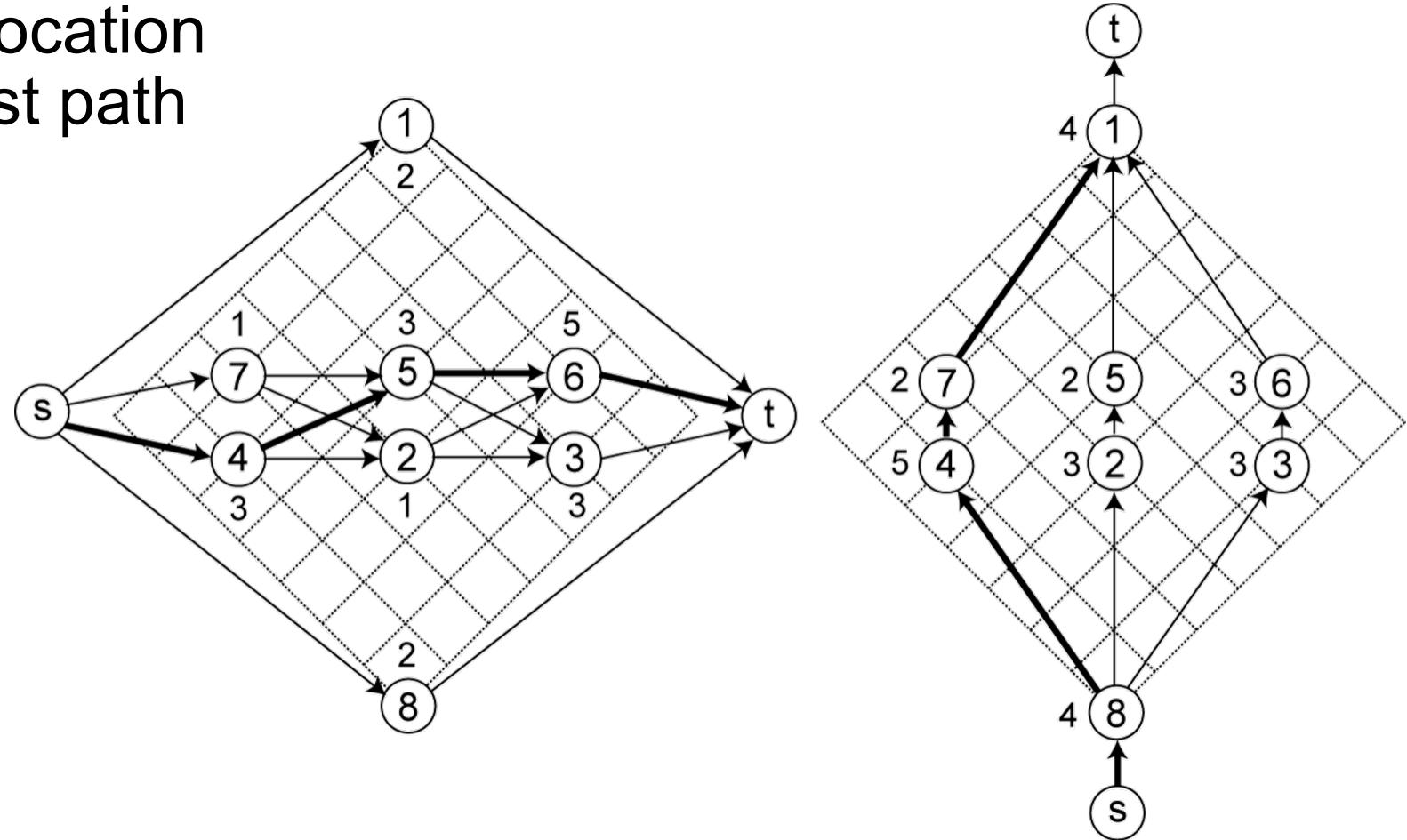
Node weight = module  
width/height!



# Computing Module Locations

Lower-left corner location  
maps to the longest path  
to each module

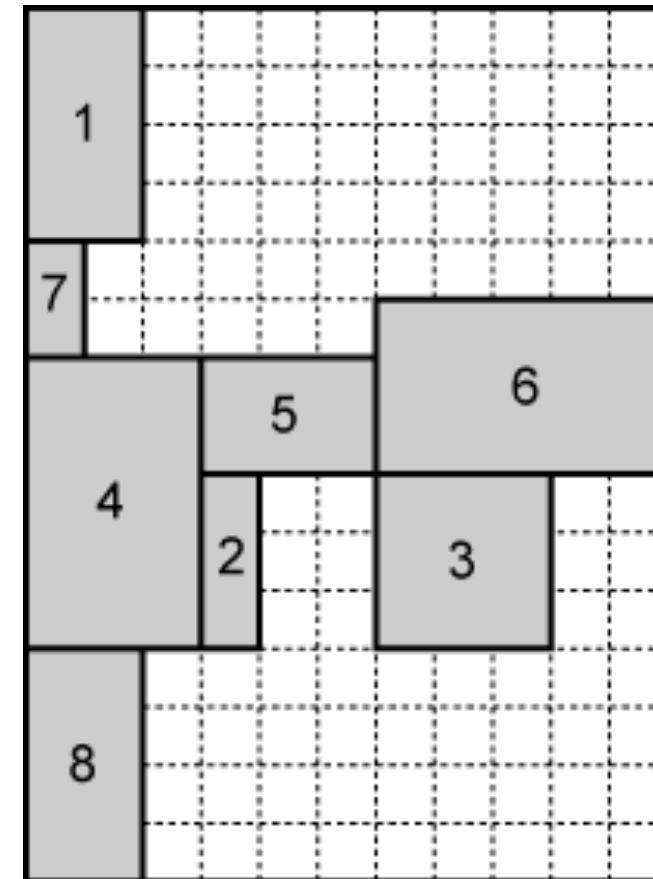
module	HCV	VCG
1	0	11
2	3	4
3	6	4
4	0	4
5	3	7
6	6	7
7	0	9
8	0	0



# Floorplan Result

module	HCV	VCG
1	0	11
2	3	4
3	6	4
4	0	4
5	3	7
6	6	7
7	0	9
8	0	0

11×15

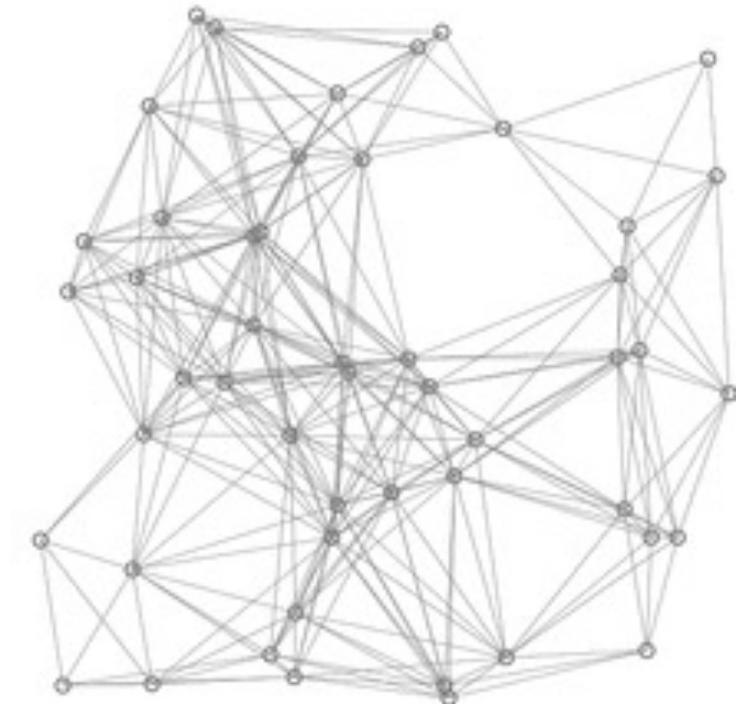


# How to Find Longest Paths?

- **Shortest-path-faster algorithm (SPFA)**
  - Improved version of Bellman-Ford algorithm
  - Add a minus sign “-” to the edge weight

```
procedure SPFA( $G, s$ )
1 for each vertex  $v \neq s$  in  $V(G)$ 
2    $d(v) := \infty$ 
3    $d(s) := 0$ 
4   push  $s$  into  $Q$ 
5 while  $Q$  is not empty do
6    $u := Q.pop()$ 
7   for each edge  $(u, v)$  in  $E(G)$  do
8     if  $d(u) + w(u, v) < d(v)$  then
9        $d(v) := d(u) + w(u, v)$ 
10      if  $v$  is not in  $Q$  then
11        push  $v$  into  $Q$ 
```

Relaxation happens  
only at active vertices  
(in queue) – *largely  
reduced redundant  
relaxations!*



red lines are shortest path covering  
blue lines are relaxations

# Move 1

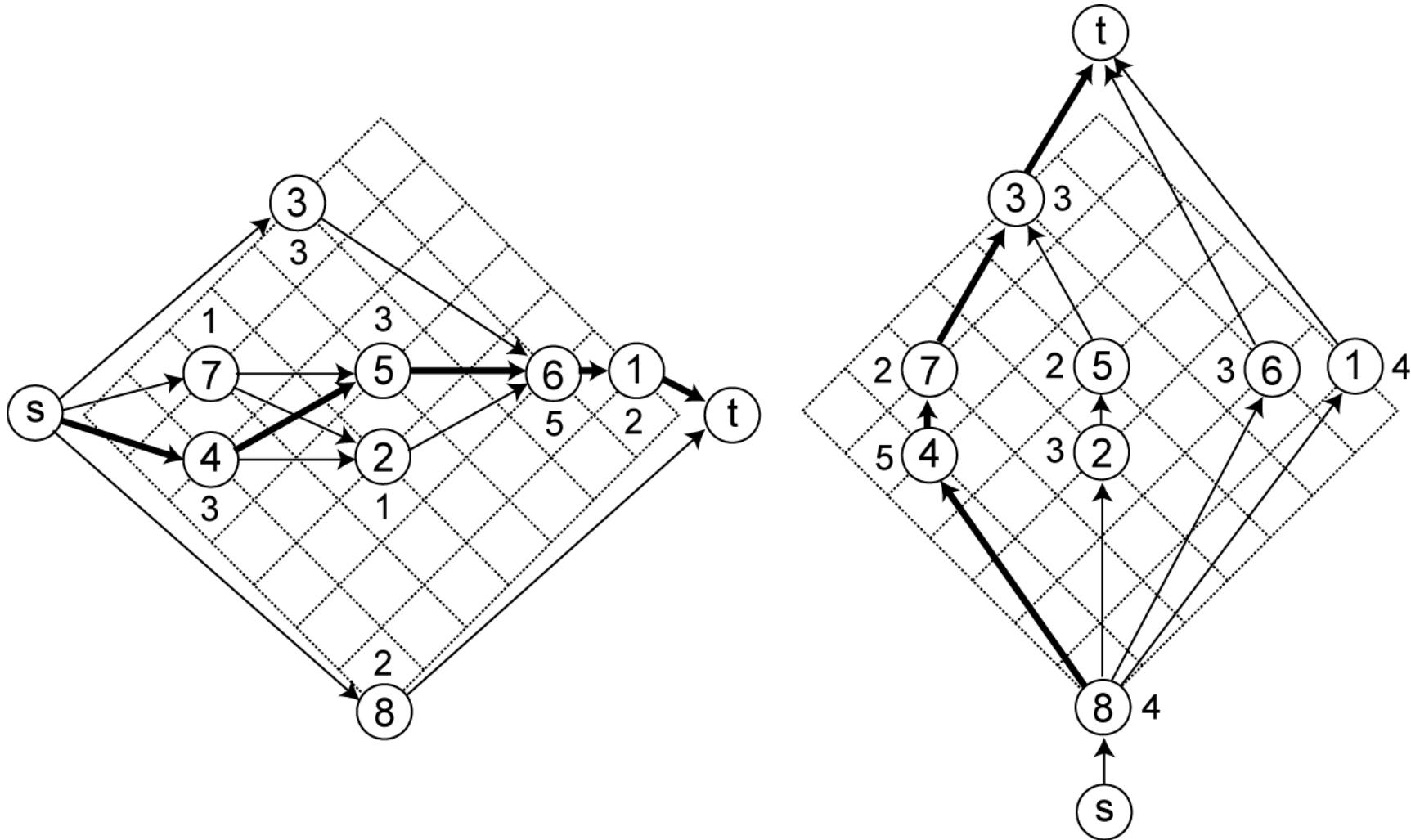
---

- Swap 1 and 3 in positive sequence of  $\text{SP}_1$ 
  - $\text{SP}_1 = (\underline{1}74526\underline{3}8, 84725361); \text{SP}_2 = (\underline{3}74526\underline{1}8, 84725361)$

module	right-of	left-of	above	below
1	$\emptyset$	$\{2, 3, 4, 5, 6, 7\}$	$\emptyset$	$\{8\}$
2	$\{1, 6\}$	$\{4, 7\}$	$\{3, 5\}$	$\{8\}$
3	$\{1, 6\}$	$\emptyset$	$\emptyset$	$\{2, 4, 5, 7, 8\}$
4	$\{1, 2, 5, 6\}$	$\emptyset$	$\{3, 7\}$	$\{8\}$
5	$\{1, 6\}$	$\{4, 7\}$	$\{3\}$	$\{2, 8\}$
6	$\{1\}$	$\{2, 3, 4, 5, 7\}$	$\emptyset$	$\{8\}$
7	$\{1, 2, 5, 6\}$	$\emptyset$	$\{3\}$	$\{4, 8\}$
8	$\emptyset$	$\emptyset$	$\{1, 2, 3, 4, 5, 6, 7\}$	$\emptyset$

# Constraint Graphs after Move 1

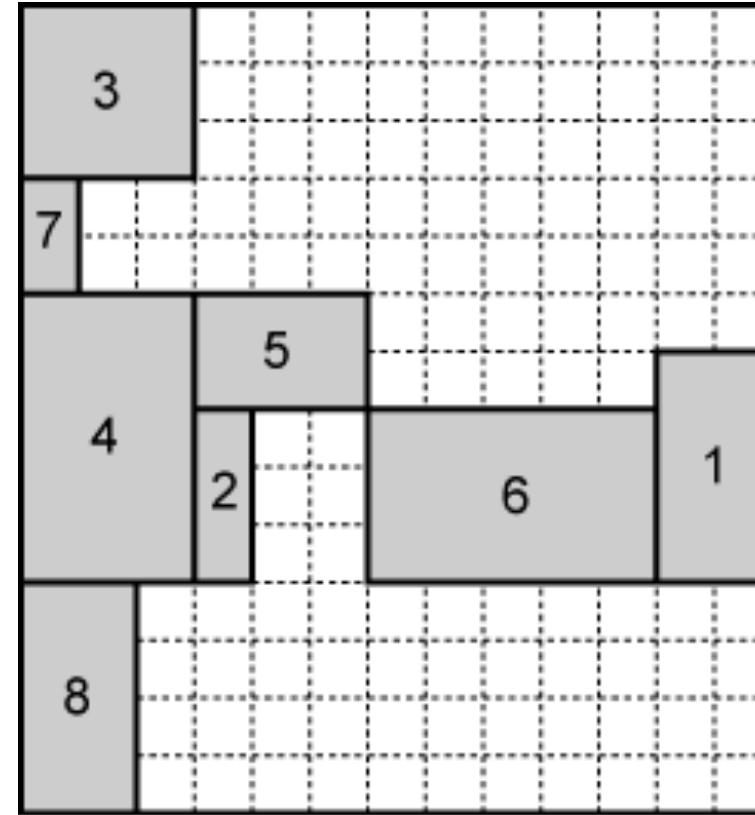
---



# Constructing Floorplan after Move 1

module	HCV	VCG
1	11	4
2	3	4
3	0	11
4	0	4
5	3	7
6	6	4
7	0	9
8	0	0

13×14



# Move 2

---

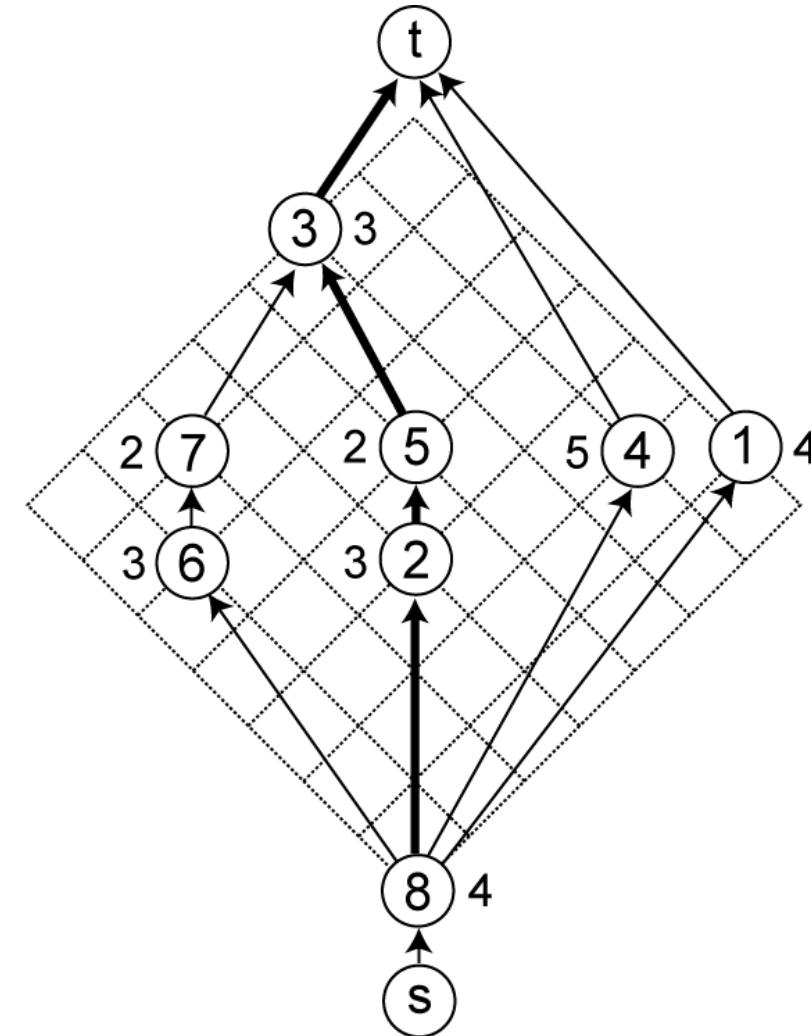
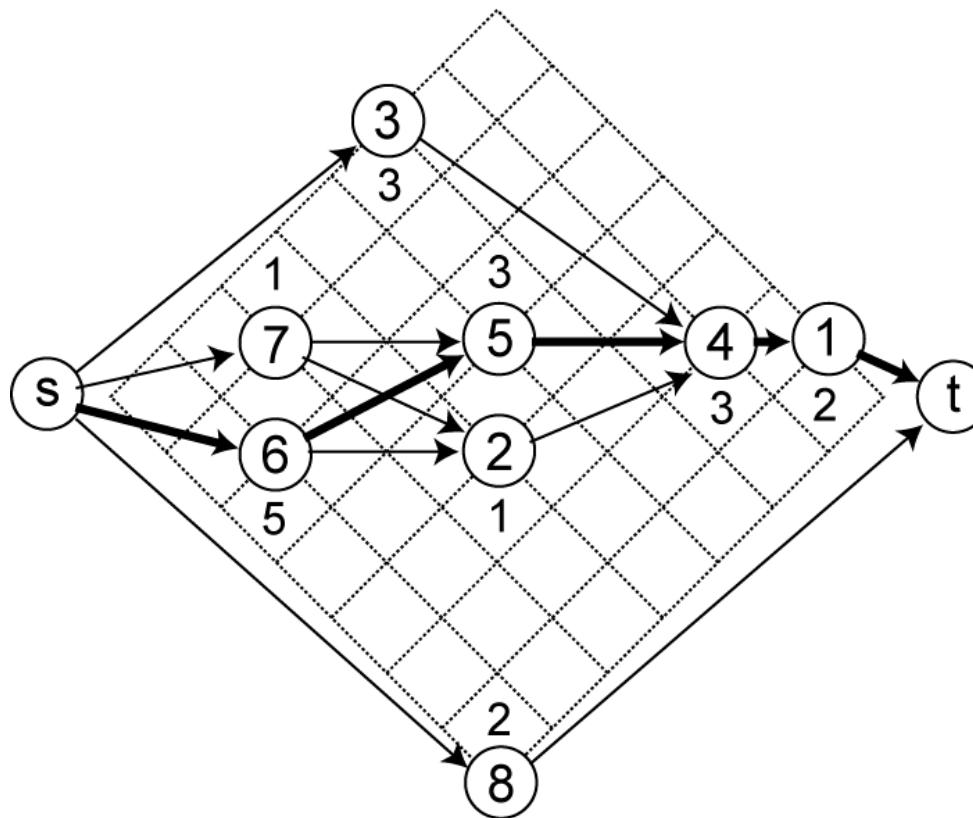
- Swap 4 and 6 in both sequences of  $\text{SP}_2$

- $\text{SP}_2 = (37\underline{4}52\underline{6}18, 8\underline{4}7253\underline{6}1); \text{SP}_3 = (37\underline{6}52\underline{4}18, 8\underline{6}7253\underline{4}1)$

module	right-of	left-of	above	below
1	$\emptyset$	$\{2, 3, 4, 5, 6, 7\}$	$\emptyset$	$\{8\}$
2	$\{1, 4\}$	$\{6, 7\}$	$\{3, 5\}$	$\{8\}$
3	$\{1, 4\}$	$\emptyset$	$\emptyset$	$\{2, 5, 6, 7, 8\}$
4	$\{1\}$	$\{2, 3, 5, 6, 7\}$	$\emptyset$	$\{8\}$
5	$\{1, 4\}$	$\{6, 7\}$	$\{3\}$	$\{2, 8\}$
6	$\{1, 2, 4, 5\}$	$\emptyset$	$\{3, 7\}$	$\{8\}$
7	$\{1, 2, 4, 5\}$	$\emptyset$	$\{3\}$	$\{6, 8\}$
8	$\emptyset$	$\emptyset$	$\{1, 2, 3, 4, 5, 6, 7\}$	$\emptyset$

# Constraint Graphs after Move 2

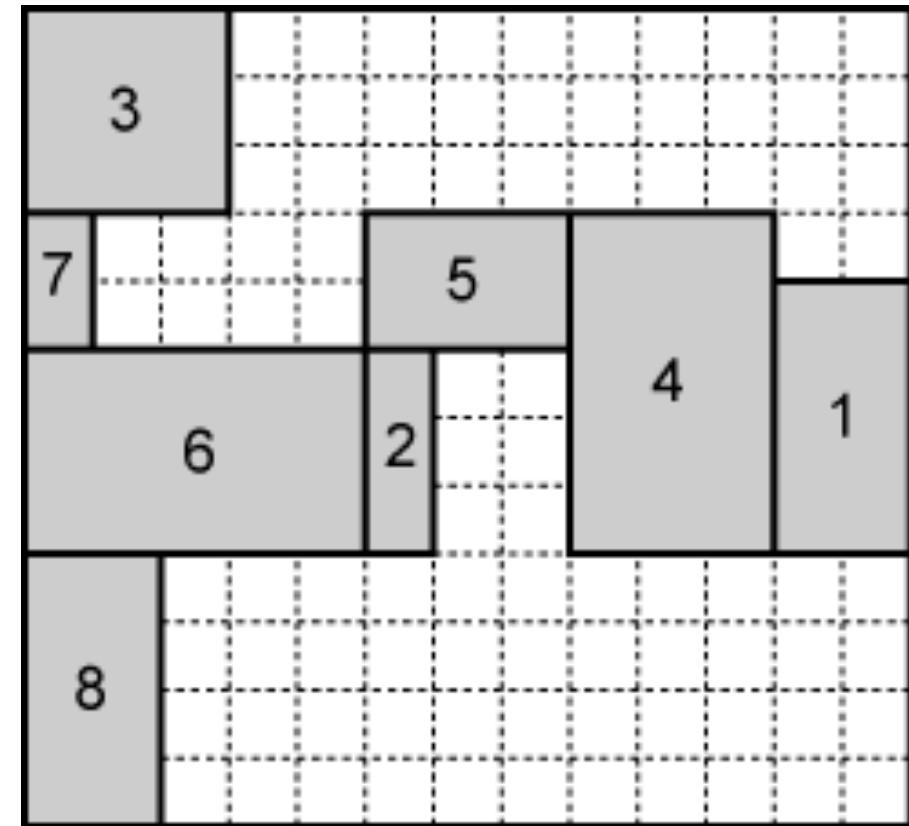
---



# Constructing Floorplan after Move 2

module	HCV	VCG
1	11	4
2	3	4
3	0	11
4	0	4
5	3	7
6	6	4
7	0	9
8	0	0

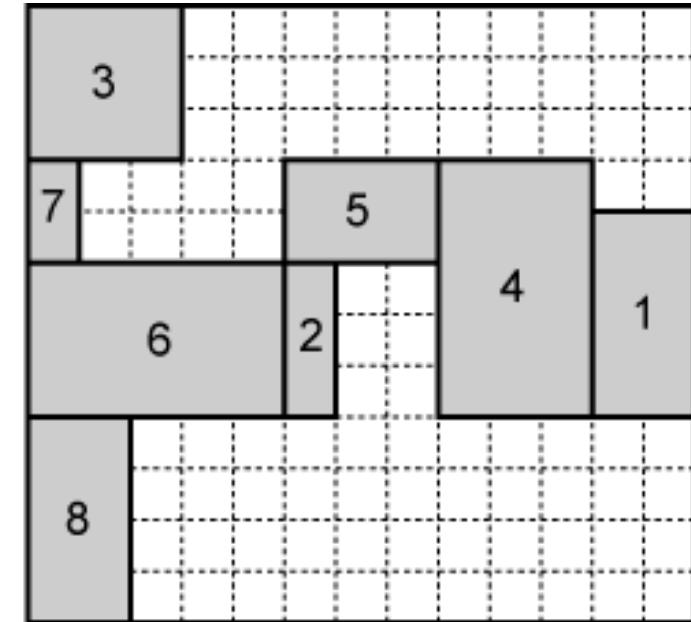
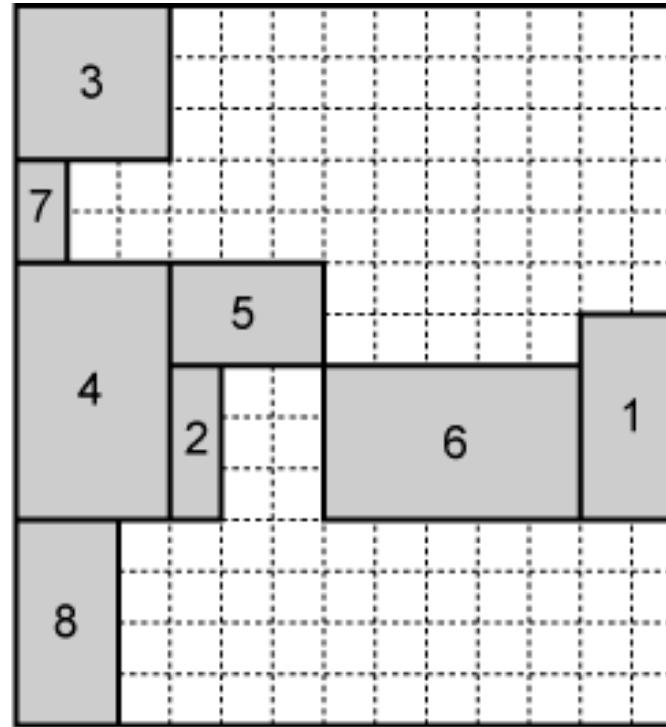
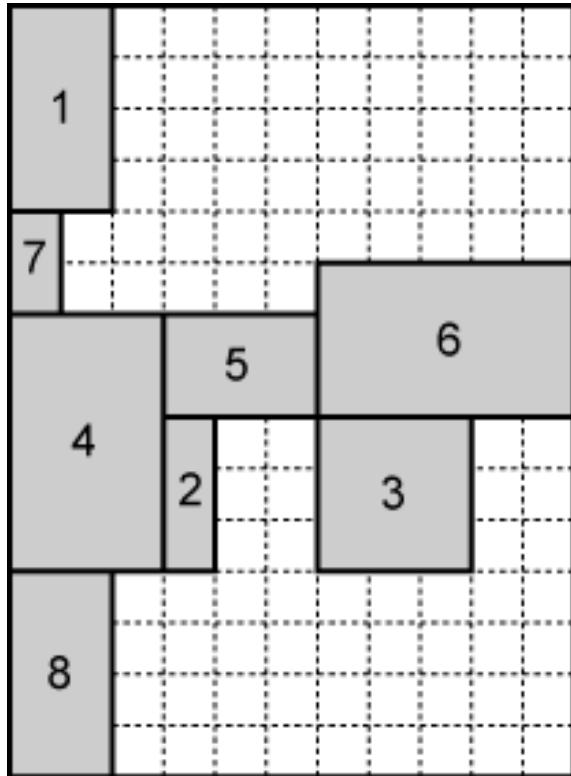
13×12



# Impacts of Move 1 and Move 2

---

- Floorplan dimension changes from  $11 \times 15$  to  $13 \times 14$  to  $13 \times 12$



# Simulated Annealing (SA) Algorithm

```
1 begin
2 Get an initial solution  $S$ ;
3 Get an initial temperature  $T > 0$ ;
4 while not yet “frozen” do
5   for  $1 \leq i \leq P$  do
6     Pick a random neighbor  $S'$  of  $S$ ;
7      $\Delta \leftarrow cost(S') - cost(S)$ ;
8     /* down hill move */
9     if  $\Delta \leq 0$  then  $S \leftarrow S'$ 
10    /* uphill move */
11    if  $\Delta > 0$  then  $S \leftarrow S'$  with probability;
12     $T \leftarrow rT$ ; /* reduce temperature */
13 return  $S$ 
14 end
```

- $T_i = \alpha T_{i-1}$  where  $\alpha=0.85$
- At  $T_i$ , try  $k \times n$  moves
  - $k$  is around 5 to 10
- Stop the annealing when
  - # of accepted moves  $< 5\%$
  - or the temperature is low
  - or until a max # of iterations
- Cost metric:  $\alpha A + \beta L$ 
  - Area + Wirelength

# Programming Assignment #2

---

- **Implement a fix-outline floorplanning algorithm**
  - <https://github.com/tsung-wei-huang/ece5960-physical-design/tree/main/PA2>
- **Two checkpoint dues, 10/5 and 10/19 23:59 PM**
  - <https://github.com/tsung-wei-huang/ece5960-physical-design/issues/2>
- **Final Due on 10/26 (Wed) 23:59 PM**
  - Upload your solutions to twhuang-server-01.ece.utah.edu
    - Account: ece6960-fall22
  - Place your source code + README under PA2/your\_uid/
  - README should contain instruction to compile & run your code

# Programming Assignment #2 (cont'd)

---

- **In addition to source code + README, upload a report with:**
  - A table showing your results of each benchmark
  - A section discussing what challenges you encounter
  - A section discussing how you overcome those challenges
    - Also discuss unsolved challenges
- **The report needs to be just a one- or two-page pdf**
  - No need to be lengthy ...
- **Upload your report to the class GitHub page**
  - <https://github.com/tsung-wei-huang/ece5960-physical-design/issues/3>
  - Due 10/26 23:59 PM

# In-class Presentation: 10/19

---

- **Floorplan research presentation on 10/19 (in class)**
  - Xiaoping Tang and D. F. Wong, "FAST-SP: a fast algorithm for block placement based on sequence pair," *IEEE/ACM Asia and South Pacific Design Automation Conference*, 2001
  - Jackey Z. Yan and Chris Chu, "DeFer: Deferred Decision Making Enabled Fixed-Outline Floorplanning Algorithm," *IEEE Trans. Comput. Aided Des. Integr. Circuits Syst.* 29(3): 367-381, 2010
- Upload your pptx to <https://github.com/tsung-wei-huang/ece5960-physical-design/issues/10> before presentation

# Summary

---

- **We have discussed sequence pair floorplan optimization**
  - Positive and negative sequences to describe module locations
  - Can represent generic floorplan (non-slicing layout)
- **We have discussed floorplan recovery from sequence pair**
  - Horizontal constraint graph (HCG)
  - Vertical constraint graph (VCG)
  - Longest path in HCG represents the floorplan width
  - Longest path in VCG represents the floorplan height