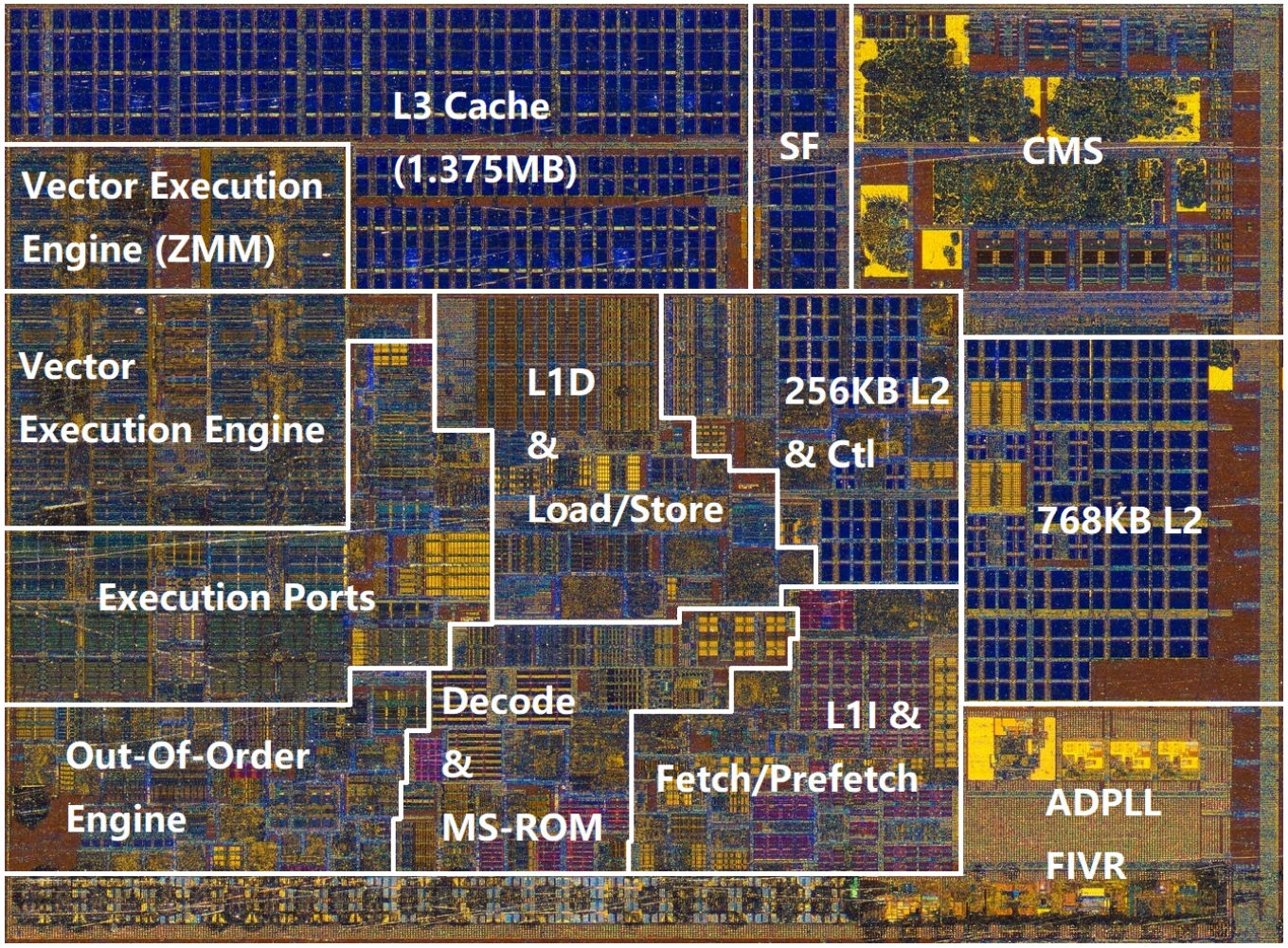
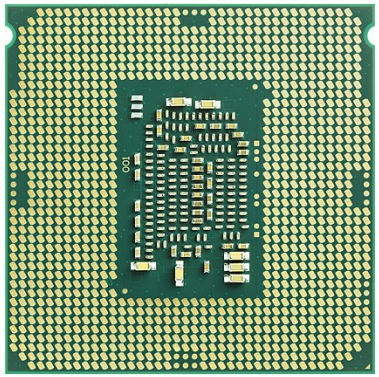
Floorplanning

ECE6133
Physical Design Automation of VLSI Systems

Prof. Sung Kyu Lim
School of Electrical and Computer Engineering
Georgia Institute of Technology

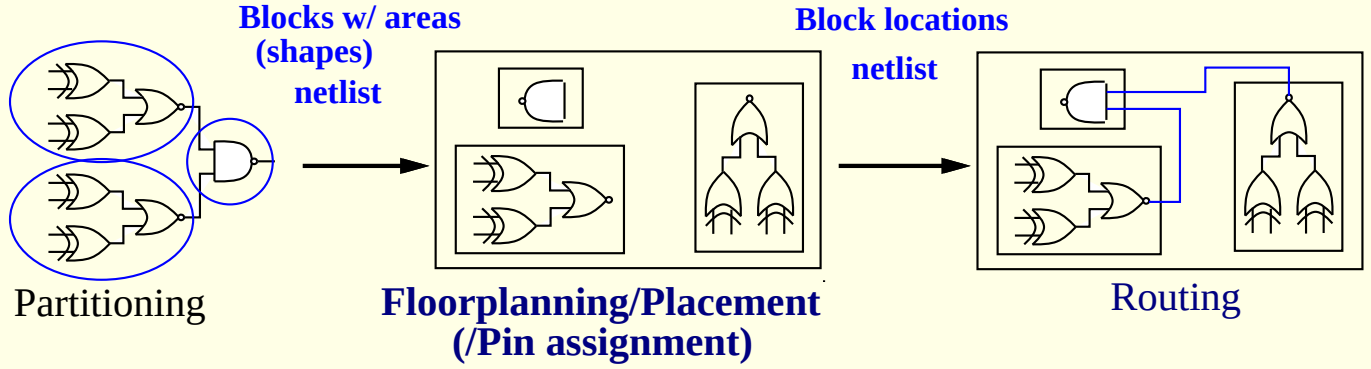
Intel i7 Skylake Floorplan (14nm, 2015)





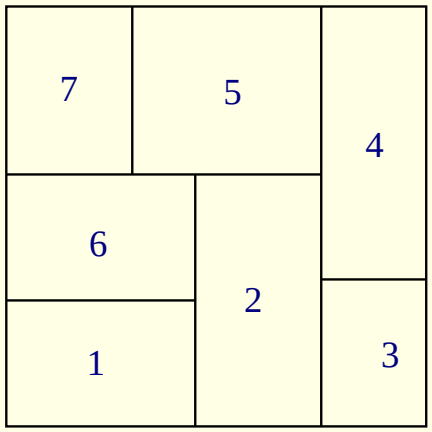
Floorplanning, Placement, and Pin Assignment

- Partitioning leads to
  - Blocks with well-defined **areas and shapes** (fixed blocks).
  - Blocks with approximated areas and no particular shapes (flexible blocks).
  - A **netlist** specifying connections between the blocks.
- Objectives
  - Find **locations** for all blocks.
  - Consider shapes of flexible block, pin locations of all the blocks.

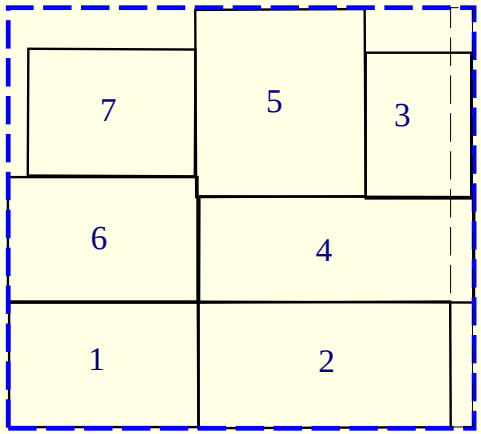


Floorplanning

- Inputs to the floorplanning problem:
  - A set of blocks, fixed or flexible.
  - Pin locations of fixed blocks.
  - A netlist.
- Objectives: **Minimize area**, **reduce wirelength** for (critical) nets, **maximize routability**, determine shapes of flexible blocks

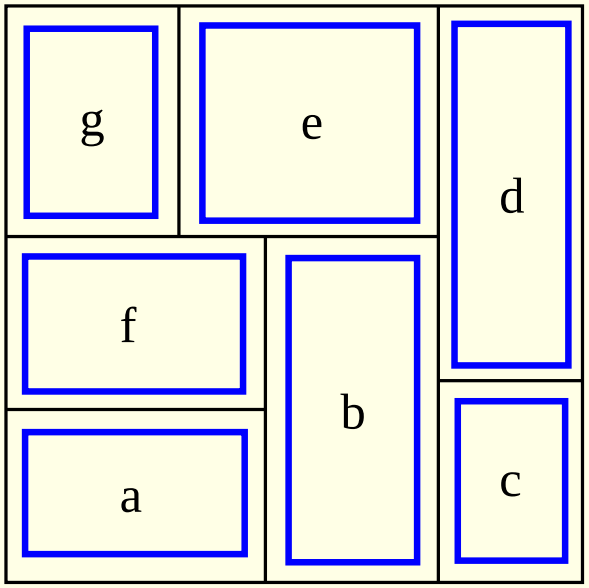


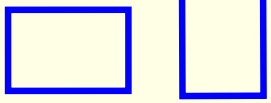
An optimal floorplan, in terms of area

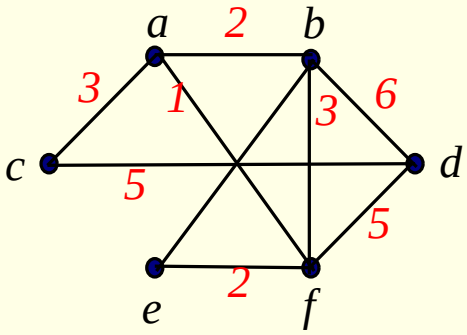


A non−optimal floorplan

Floorplan Design

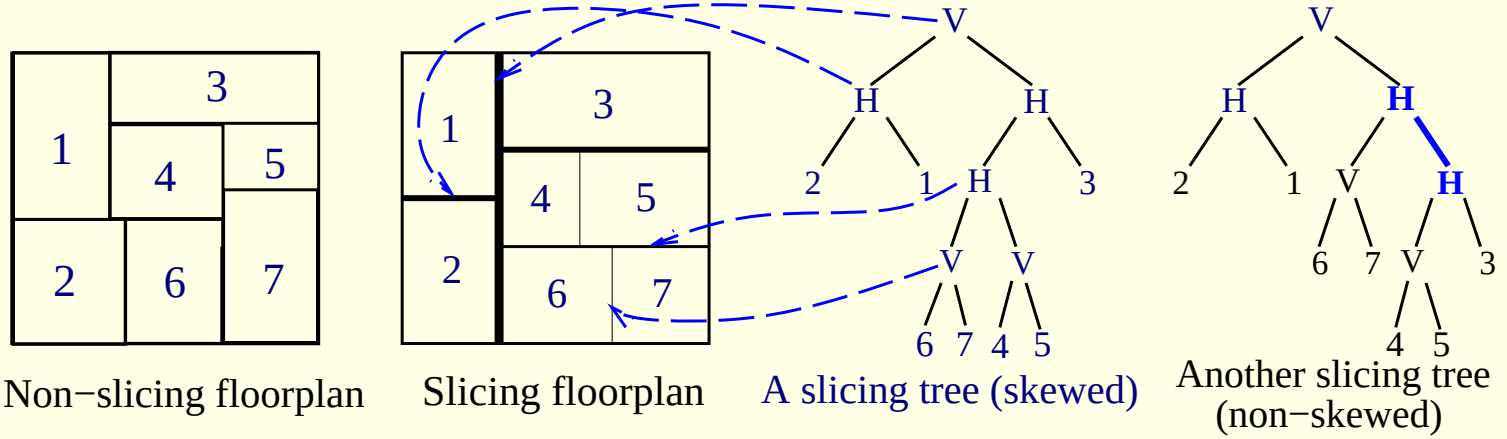


- *Modules:*  *x* *y*
- *Area: A=xy*
- *Aspect ratio: r <= y/x <= s*
- *Rotation:* 
- *Module connectivity*



Floorplanning: Terminology

- **Rectangular dissection:** Subdivision of a given rectangle by a finite # of horizontal and vertical line segments into a finite # of non-overlapping rectangles.
- **Slicing structure:** a rectangular dissection that can be obtained by repetitively subdividing rectangles horizontally or vertically.
- **Slicing tree:** A binary tree, where each internal node represents a vertical cut line or horizontal cut line, and each leaf a basic rectangle.
- **Skewed slicing tree:** One in which no node and its **right** child are the same.



Non−slicing floorplan

Slicing floorplan

A slicing tree (skewed)

Another slicing tree (non−skewed)

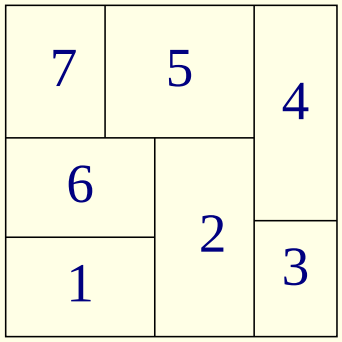
Solution Representation

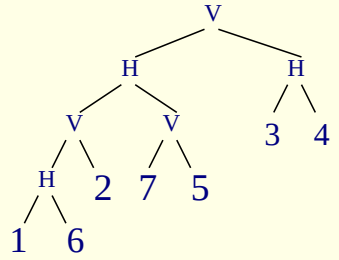
- An expression $E = e_1 e_2 \ldots e_{2n-1}$, where $e_i \in \{1, 2, \ldots, n, H, V\}, 1 \leq i \leq 2n-1$, is a **Polish expression** of length $2n-1$ iff
  1. every operand j, $1 \leq j \leq n$, appears exactly once in E;
  2. (**the balloting property**) for every subexpression $E_i = e_1 \ldots e_i, 1 \leq i \leq 2n-1$, #operands $>$ #operators.

1 6 H 3 5 V 2 H V 7 4 H

# of operands = 4 ..... = 7
# of operators = 2 ..... = 5

- Polish expression $\longleftrightarrow$ Postorder traversal.
- ijH: rectangle i on bottom of j; ijV: rectangle i on the left of j.



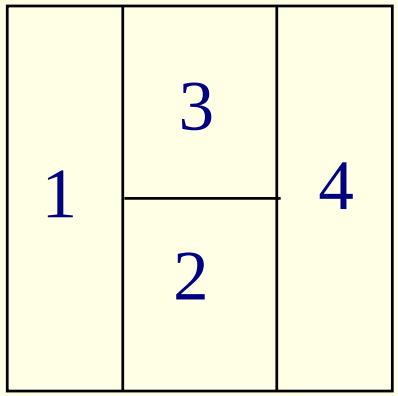


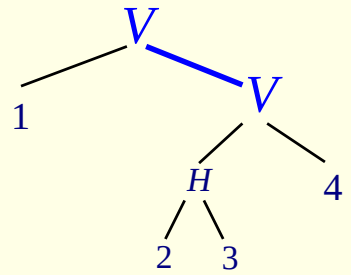
E = 16H2V75VH34HV

*E = 16+2*75*+34+**

Postorder traversal of a tree!

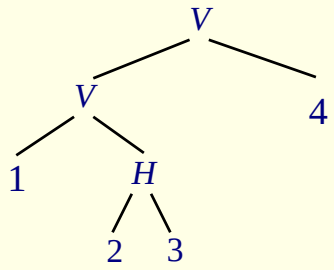
Solution Representation (cont'd)





E = 123H4VV

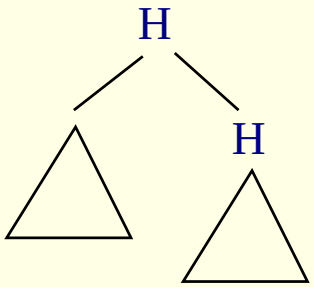
non−skewed!



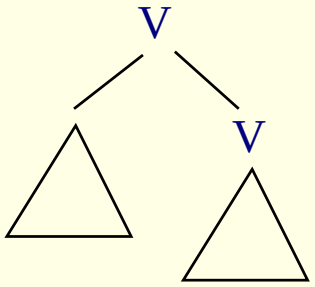
E = 123HV4V

skewed!

Non−skewed cases



..... HH .....

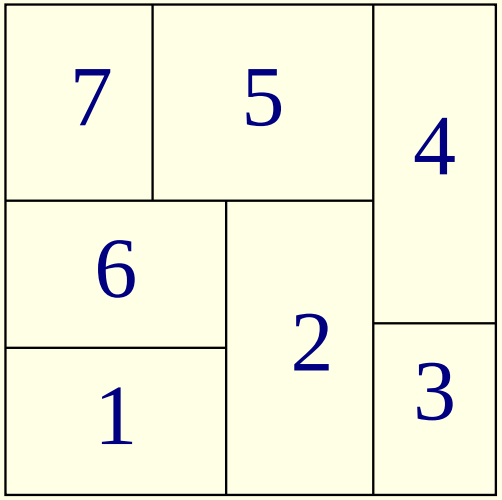


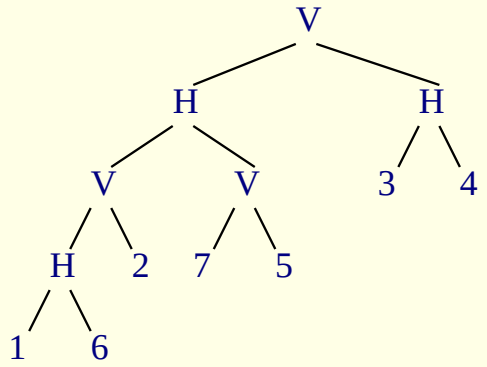
..... VV .....

- **Question:** How to eliminate ambiguous representation?

Normalized Polish Expression

- A Polish expression $E = e_1 e_2 \ldots e_{2n-1}$ is called **normalized** iff E has no consecutive operators of the same type (H or V).
- Given a **normalized** Polish expression, we can construct a **unique** rectangular slicing structure.

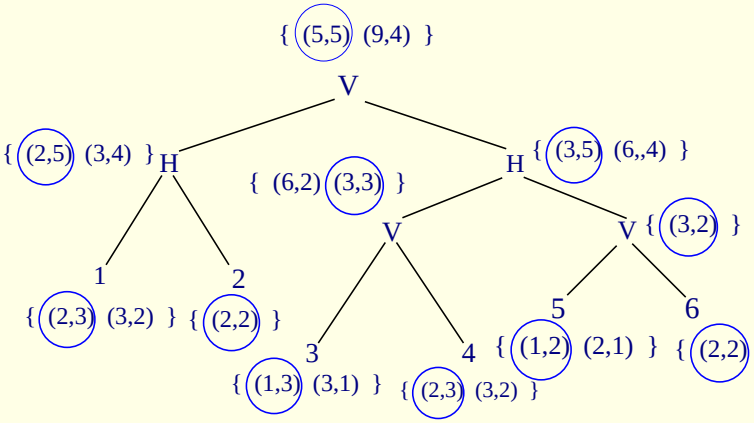




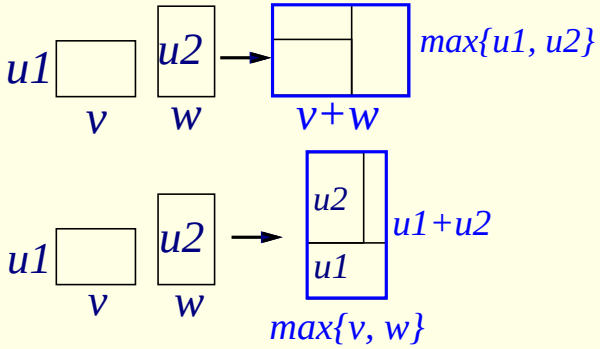
E = 16H2V75VH34HV

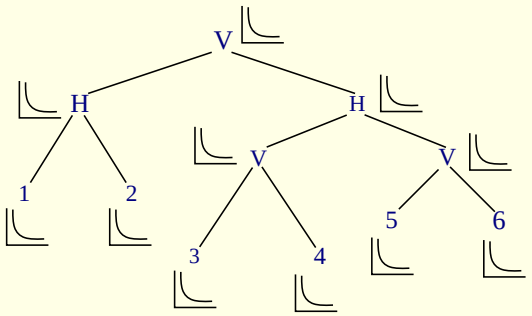
A normalized Polish expression

Area Computation



	2	1	2
2	2	5	6
3	1	3	4





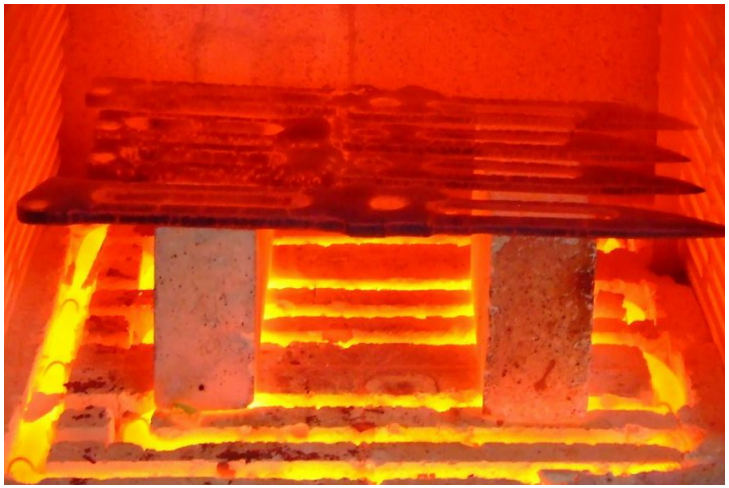
- Wiring cost?

Floorplan Design by Simulated Annealing

- Related work
  - Wong & Liu, "A new algorithm for floorplan design," DAC'86.
    - Consider slicing floorplans.
  - Wong & Liu, "Floorplan design for rectangular and L-shaped modules," ICCAD'87.
    - Also consider L-shaped modules.
  - Wong, Leong, Liu, Simulated Annealing for VLSI Design, pp. 31–71, Kluwer academic Publishers, 1988.
- Ingredients: solution space, neighborhood structure, cost function, annealing schedule?

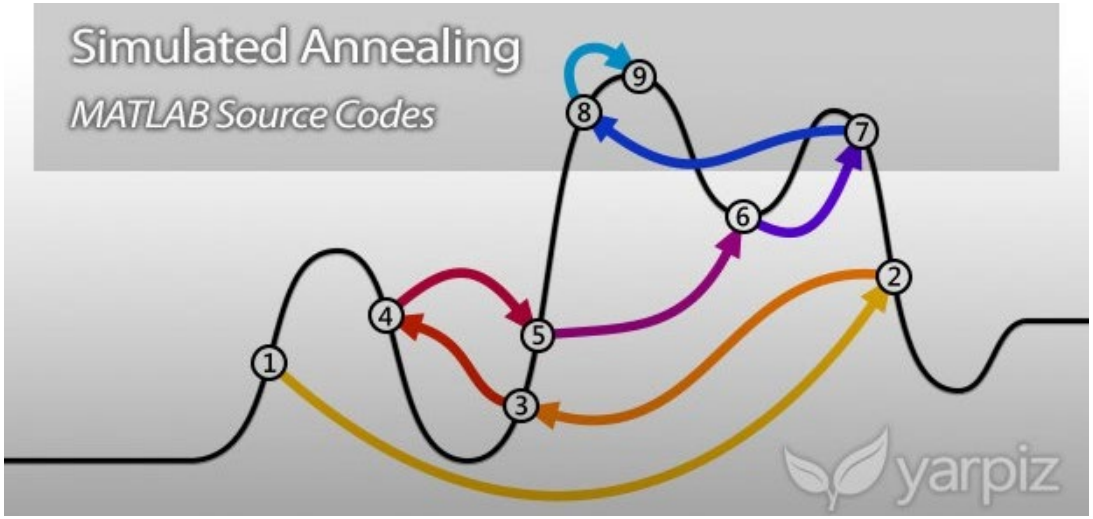
Annealing (Wikipedia)

- **Annealing (metallurgy)**
  - a heat treatment that alters the microstructure of a material, causing changes in properties such as strength, hardness, and ductility
- **Simulated annealing**
  - a numerical optimization technique for searching for a solution in a space otherwise too large for ordinary search methods to yield results



Simulated Annealing Algorithm

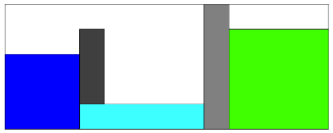
- **A random initial solution is available as the input**
  - **A new solution is generated by making a RANDOM perturbation**
  - **If the solution improves, the move is always accepted**
  - **If not, the move is accepted with a probability that decreases with the decrease in a parameter called "annealing temperature" T.**



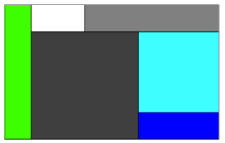
Kirkpatrick, Gelatt, Vecchi, "Optimization by Simulated Annealing". Science, 1983.

Slicing Floorplanning Examples

- **Modi and Gupta (Spring 2013)**
  - **5 blocks**

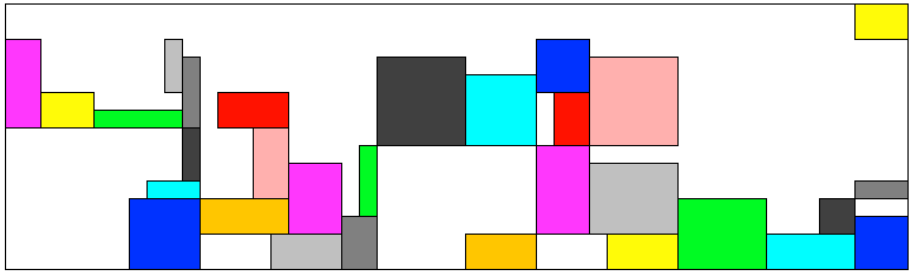


before (area = 65)

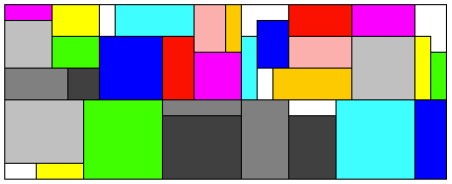


after (area = 40)

  - **30 blocks**



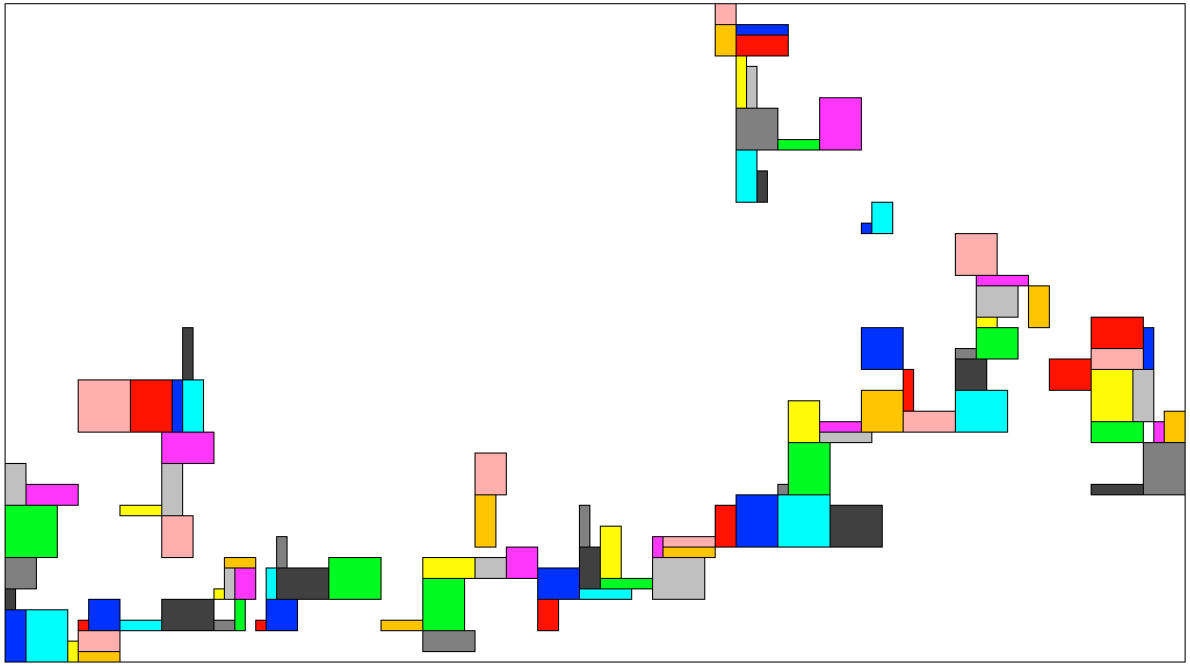
before (area = 1075)



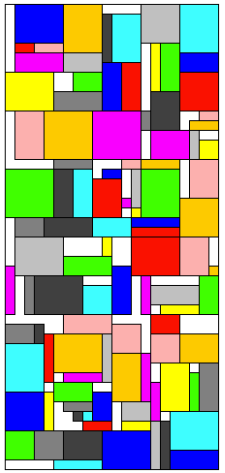
after (area = 308)

Slicing Floorplanning Examples

- **Modi and Gupta (Spring 2013)**
  - **100 blocks (took 0.5min)**



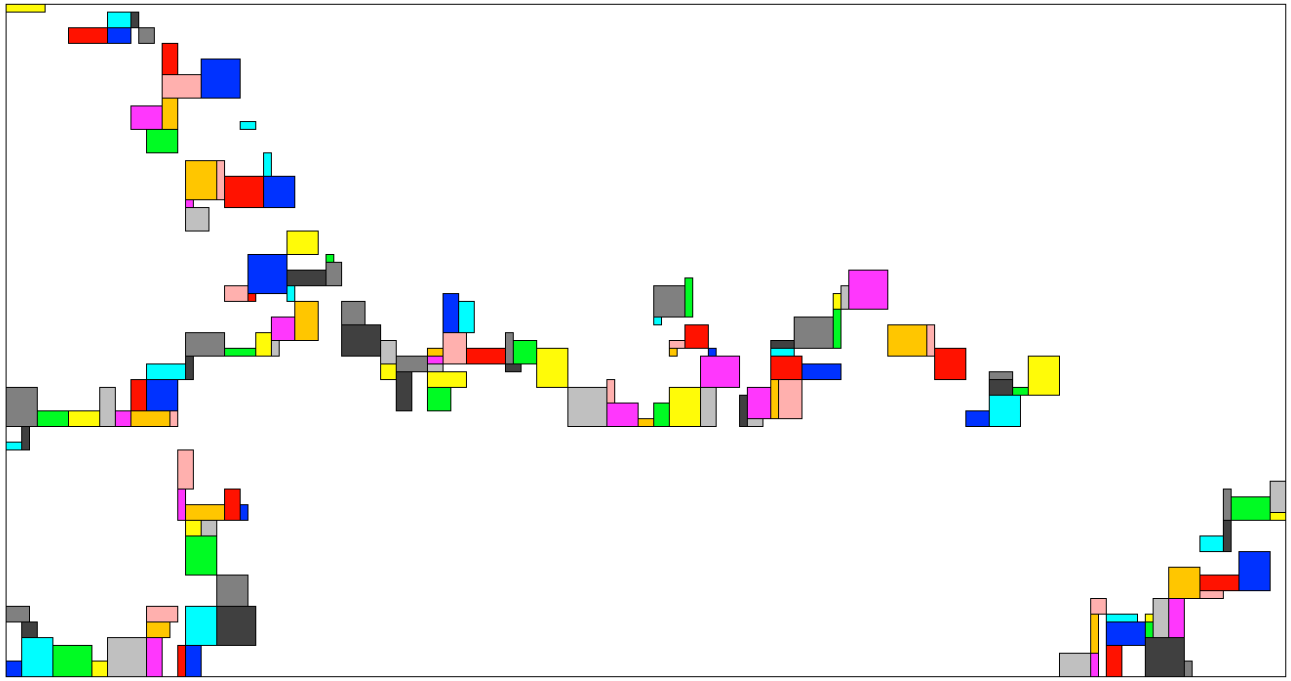
before (area = 7119)



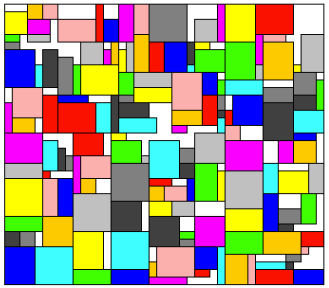
after (area = 1056)

Slicing Floorplanning Examples

- **Modi and Gupta (Spring 2013)**
  - **150 blocks (took 8min)**



before (area = 14104)

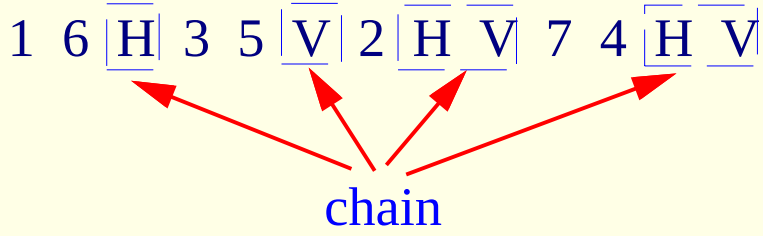


after (area = 1554)

Slicing Floorplanning Examples

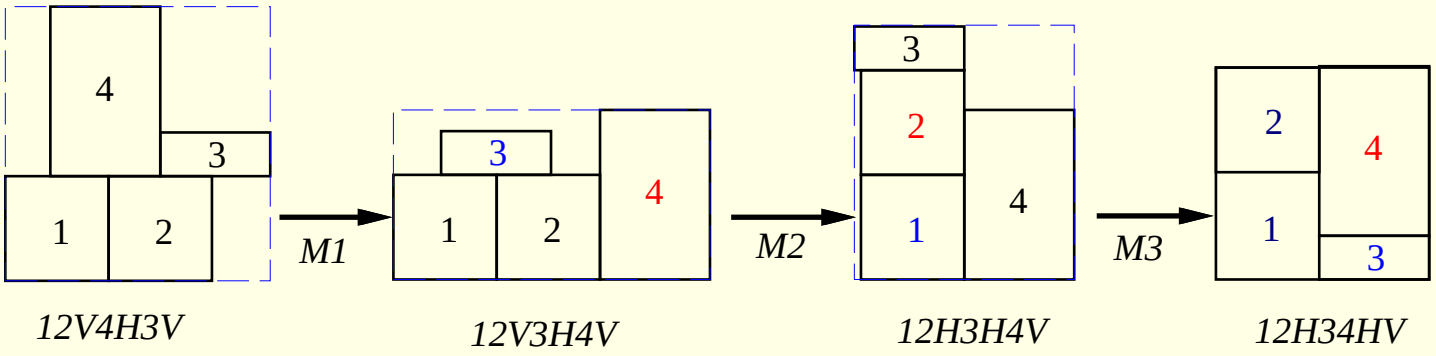
Neighborhood Structure

- **Chain:** $HVHVH\ldots$ or $VHVHV\ldots$



- **Adjacent:** 1 and 6 are adjacent operands; 2 and 7 are adjacent operands; 5 and V are adjacent operand and operator.
- 3 types of moves:
  - $M1$ **(Operand Swap)**: Swap two adjacent operands.
  - $M2$ **(Chain Invert)**: Complement some chain ($\overline{V} = H, \overline{H} = V$).
  - $M3$ **(Operator/Operand Swap)**: Swap two adjacent operand and operator.

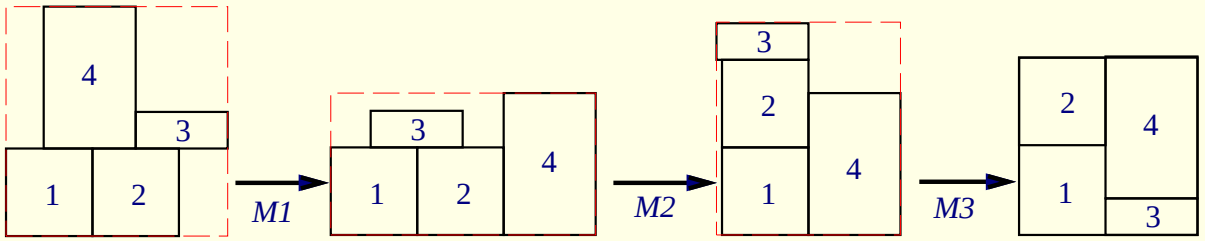
Effects of Perturbation



- **Question:** The balloting property holds during the moves?
  - $M1$ and $M2$ moves are OK.
  - **Check the $M3$ moves!** Reject "illegal" $M3$ moves.
- **Check $M3$ moves:** Assume that the M_3 move swaps the operand e_i with the operator e_{i+1}, $1 \leq i \leq k-1$. Then, the swap will not violate the balloting property iff $2N_{i+1} < i$.
  - N_k: # of operators in the Polish expression $E = e_1 e_2 \ldots e_k, 1 \leq k \leq 2n-1$.

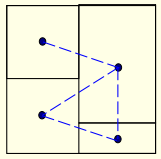
Cost Function

- $\Phi = A + \lambda W$.
  - A: area of the smallest rectangle
  - W: overall wiring length
  - λ: user-specified parameter



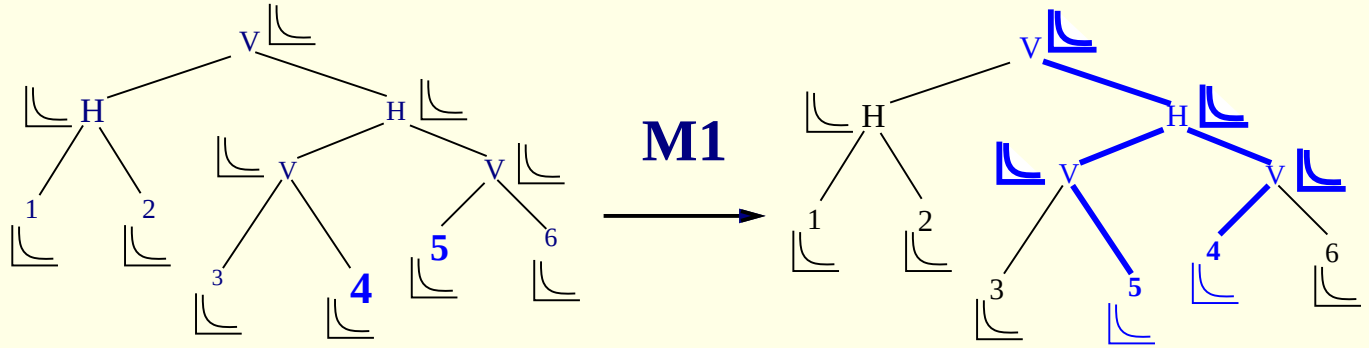
A: 12H34HV

- $W = \sum_{ij} c_{ij} d_{ij}$.
  - c_{ij}: # of connections between blocks i and j.
  - d_{ij}: center-to-center distance between basic rectangles i and j.



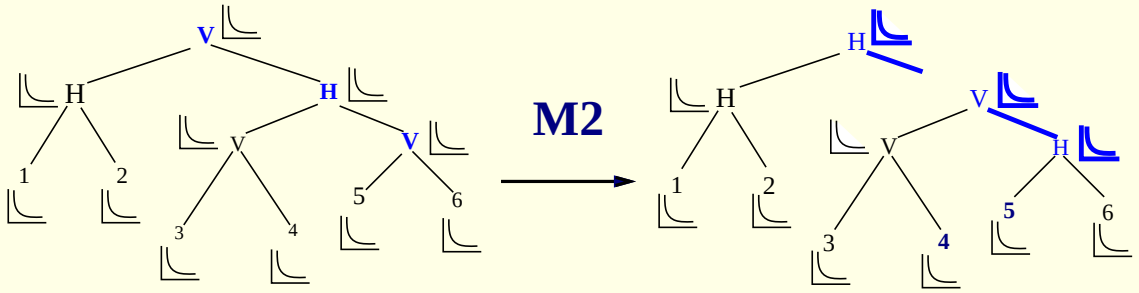
Incremental Computation of Cost Function

- Each move leads to only a minor modification of the Polish expression.
- At most **two paths** of the slicing tree need to be updated for each move.

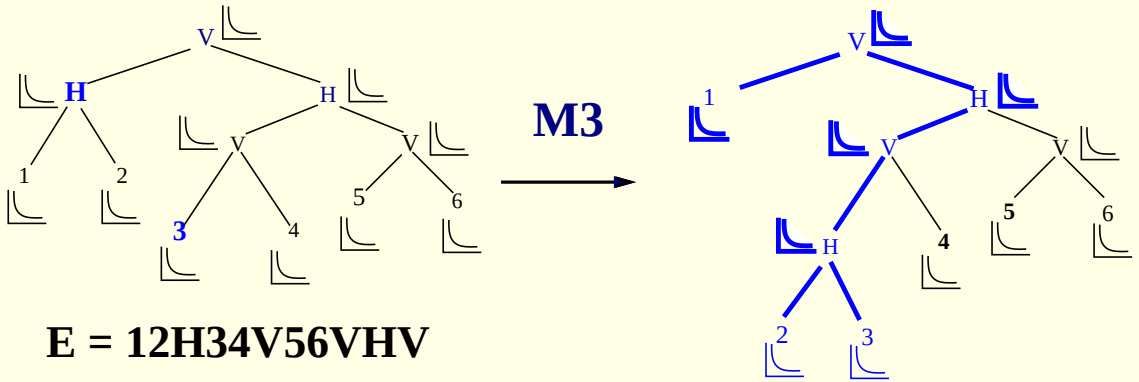


E = 12H34V56VHV **E = 12H35V46VHV**

Incremental Computation of Cost Function (cont'd)



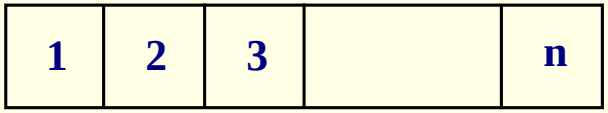
E = 12H34V56VHV **E = 12H34V56HVH**



E = 12H34V56VHV **E = 123H4V56VHV**

Annealing Schedule

- Initial solution: $12V3V \ldots nV$.



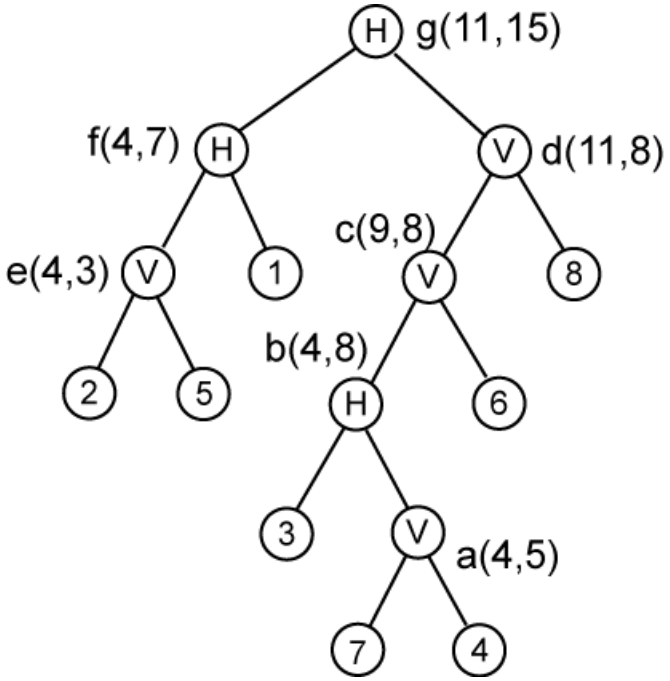
- $T_i = r^i T_0, i = 1, 2, 3, \ldots$; $r = 0.85$.
- At each temperature, try kn moves ($k = 5$–10).
- Terminate the annealing process if
  - # of accepted moves $< 5\%$,
  - temperature is low enough, or
  - run out of time.

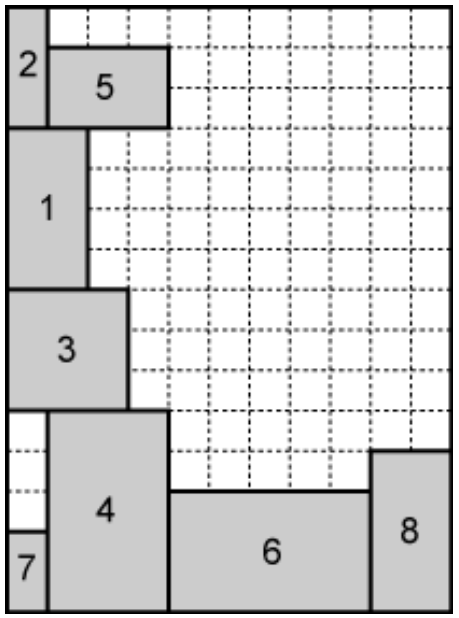
Algorithm: Simulated_Annealing_Floorplanning(P, ϵ, r, k)

```
1  begin
2  E ← 12V3V4V...nV; /* initial solution */
3  Best ← E; T_0 ← Δ_avg / ln(P); M ← MT ← uphill ← 0; N = kn;
4  repeat
5    MT ← uphill ← reject ← 0;
6    repeat
7      SelectMove(M);
8      Case M of
9      M_1: Select two adjacent operands e_i and e_j; NE ← Swap(E, e_i, e_j);
10     M_2: Select a nonzero length chain C; NE ← Complement(E, C);
11     M_3: done ← FALSE;
12       while not (done) do
13         Select two adjacent operand e_i and operator e_{i+1};
14         if (e_{i-1} ≠ e_{i+1}) and (2N_{i+1} < i) then done ← TRUE;
15       NE ← Swap(E, e_i, e_{i+1});
16     MT ← MT + 1; Δcost ← cost(NE) − cost(E);
17     if (Δcost ≤ 0) or (Random < e^{−Δcost/T})
18     then
19       if (Δcost > 0) then uphill ← uphill + 1;
20       E ← NE;
21       if cost(E) < cost(best) then best ← E;
22     else reject ← reject + 1;
23 until (uphill > N) or (MT > 2N);
24 T = rT; /* reduce temperature */
25 until (reject/MT > 0.95) or (T < ε) or OutOfTime;
26 end
```

Normalized Polish Expression

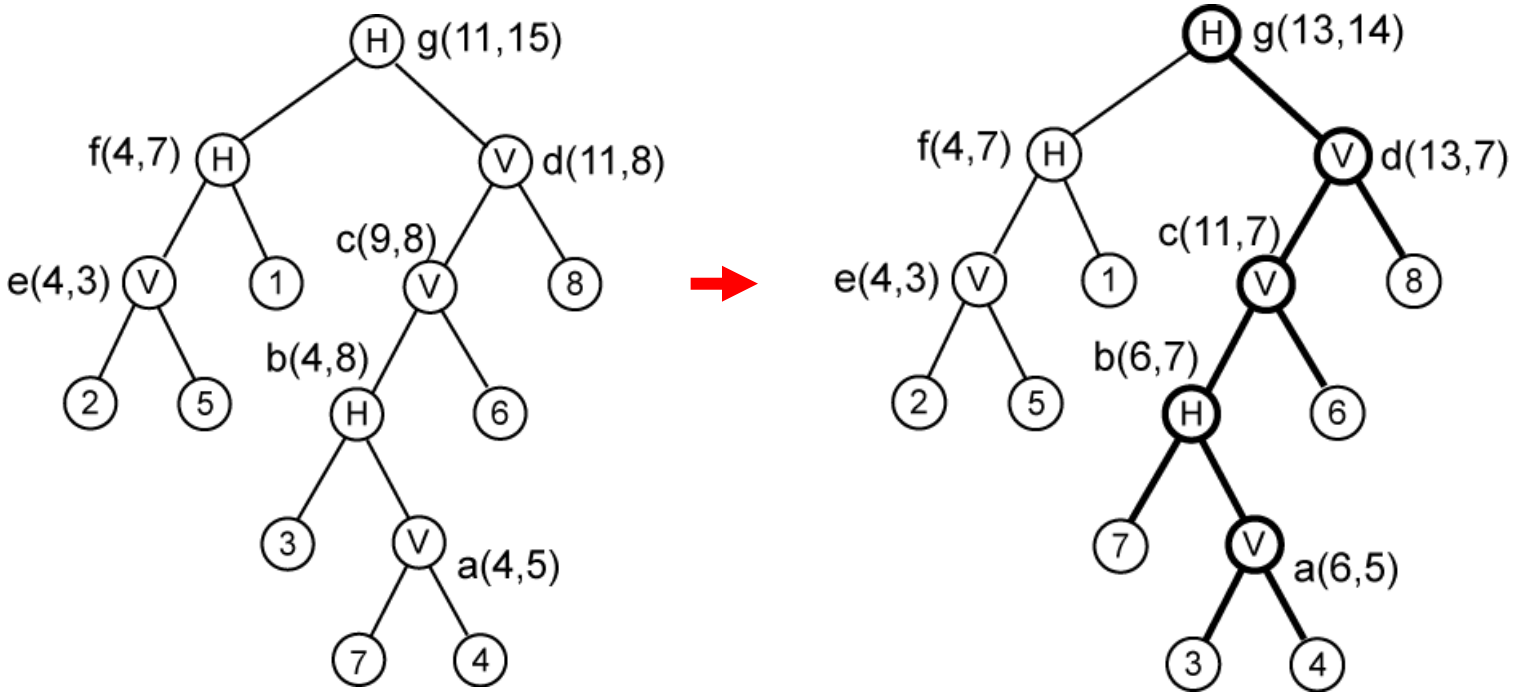
- Draw slicing floorplan based on:
  - Initial PE: P_1 = 25V1H374VH6V8VH
  - Dimensions: (2,4), (1,3), (3,3), (3,5), (3,2), (5,3), (1,2), (2,4)



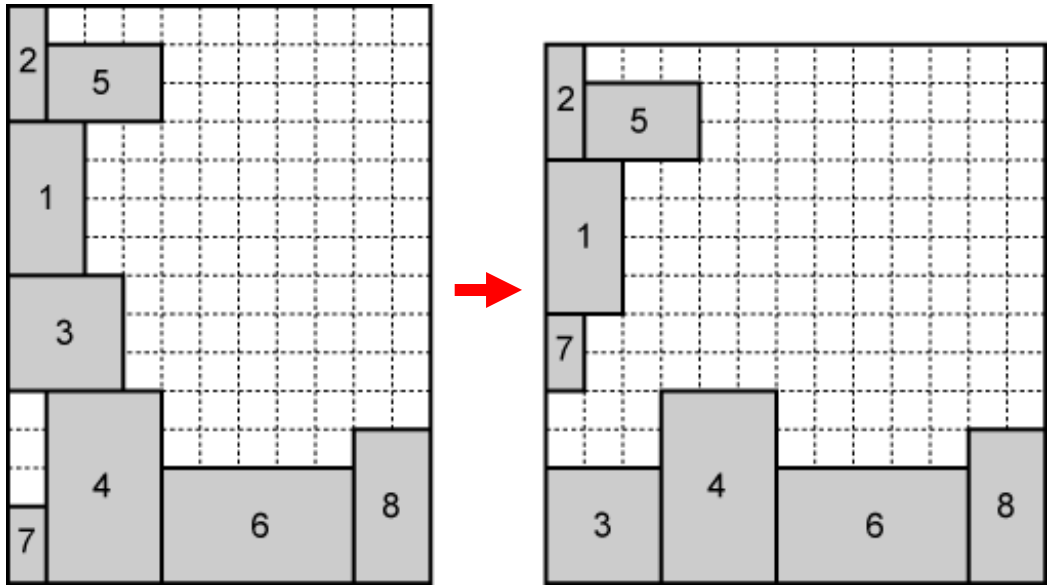


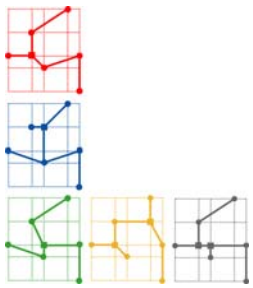
M1 Move

- Swap module 3 and 7 in P_1 = 25V1H374VH6V8VH
  - We get: P_2 = 25V1H734VH6V8VH
  - Area changed from 11 × 15 to 13 × 14



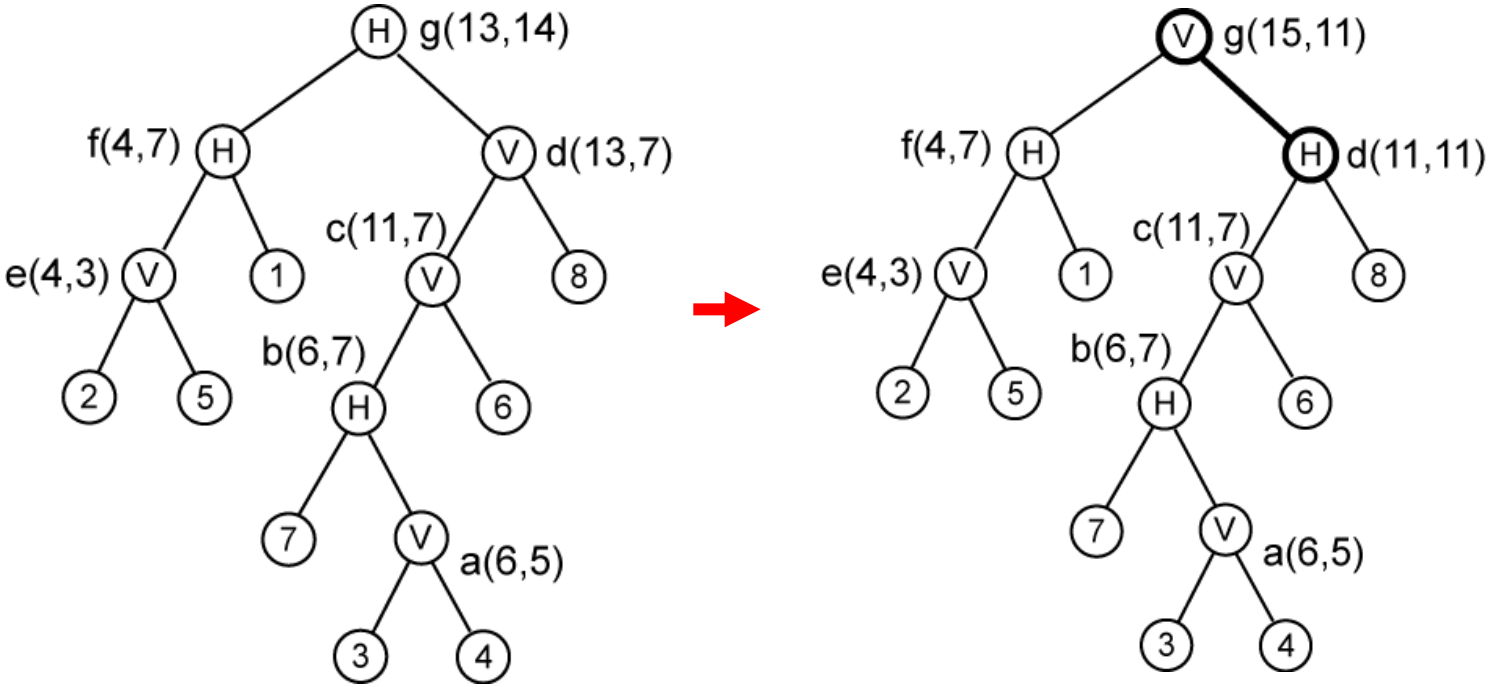
Change on Floorplan



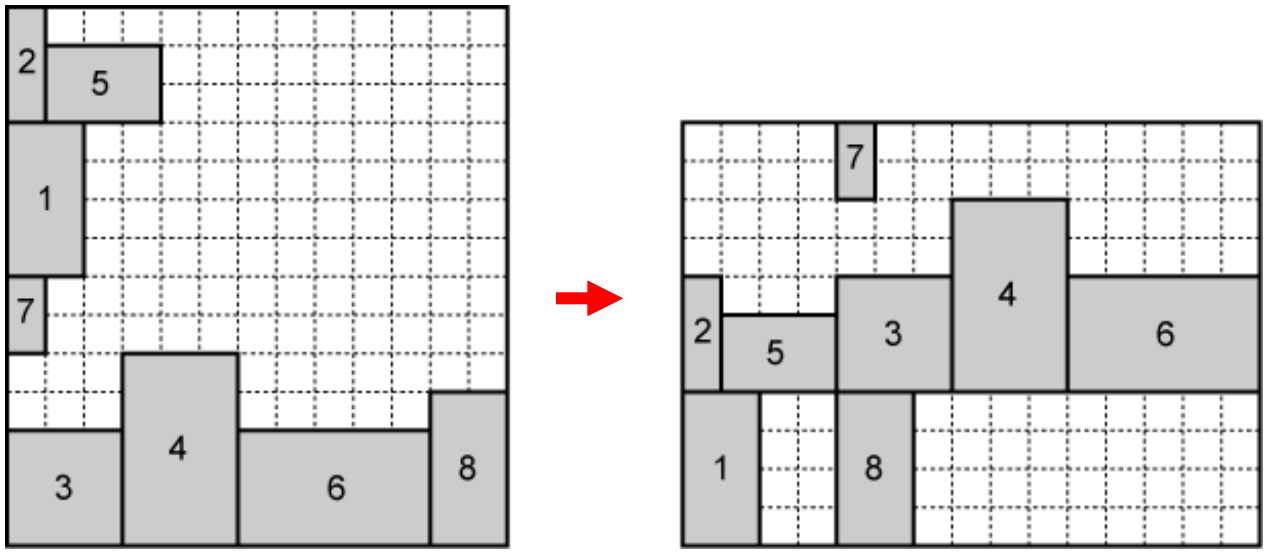


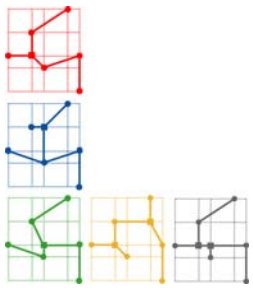
M2 Move

- Complement last chain in P_2 = 25V1H734VH6V8VH
  - We get: P_3 = 25V1H734VH6V8HV
  - Area changed from 13 × 14 to 15 × 11



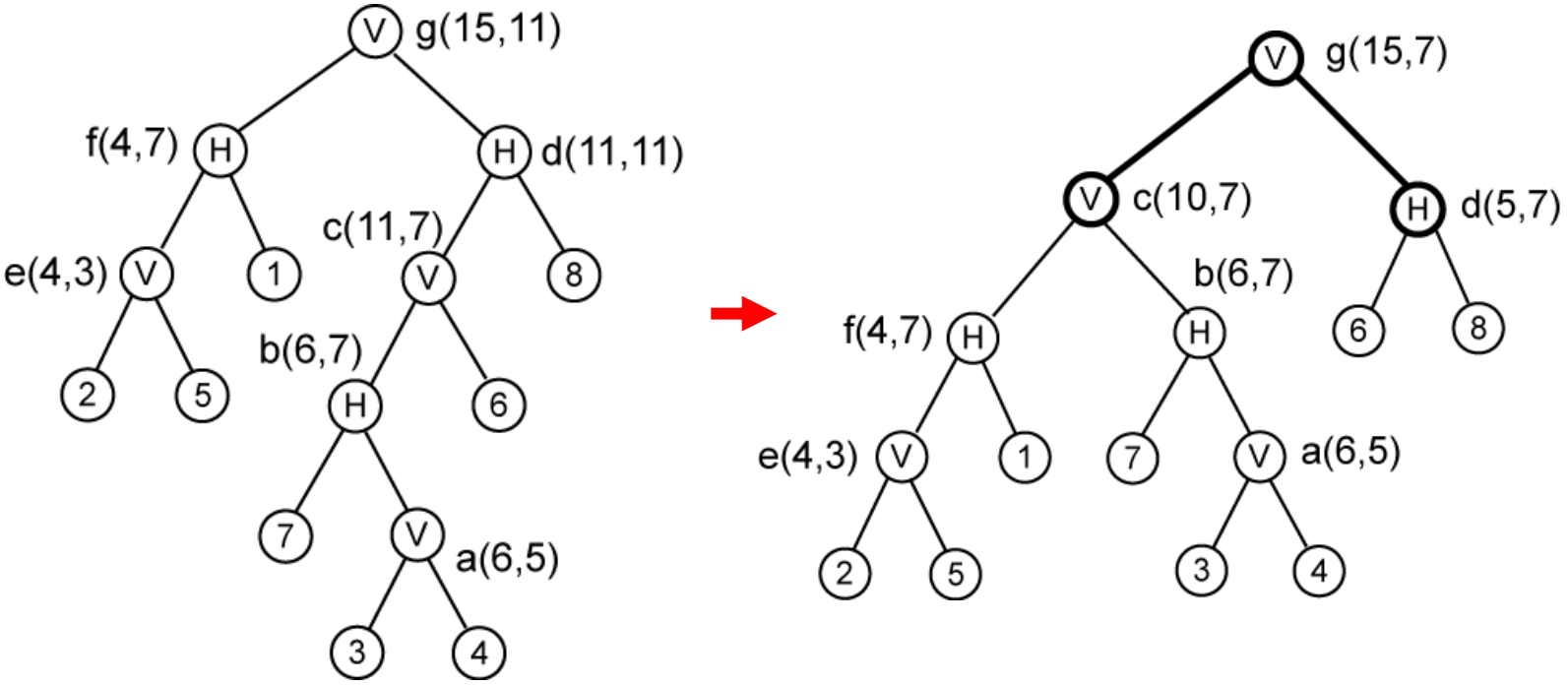
Change on Floorplan



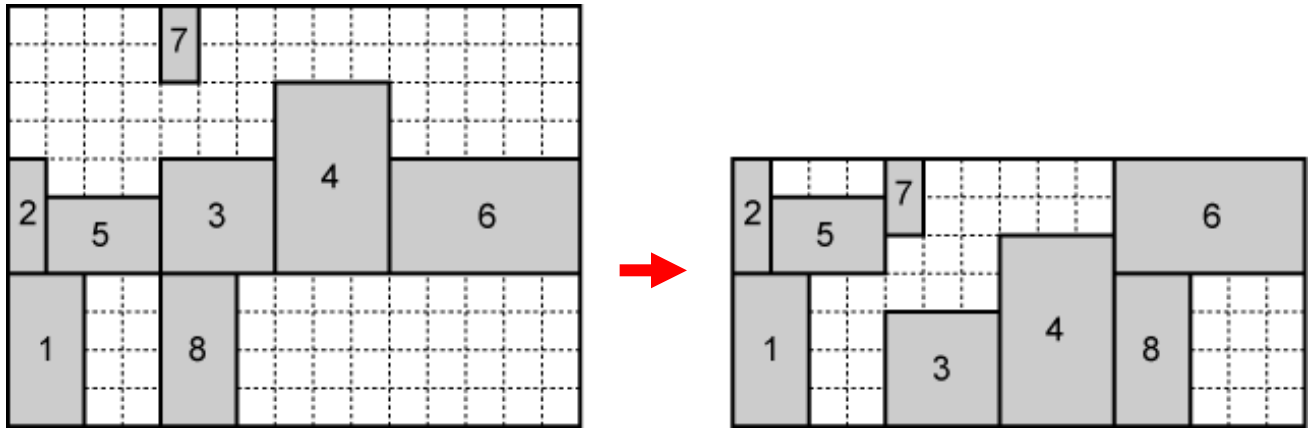


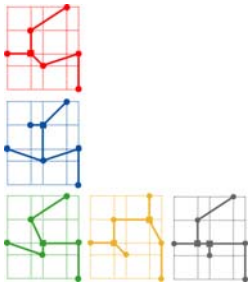
M3 Move

- Swaps 6 and V in P_3 = 25V1H734VH6V8HV
  - We get: P_4 = 25V1H734VHV68HV
  - Area changed from 15 × 11 to 15 × 7



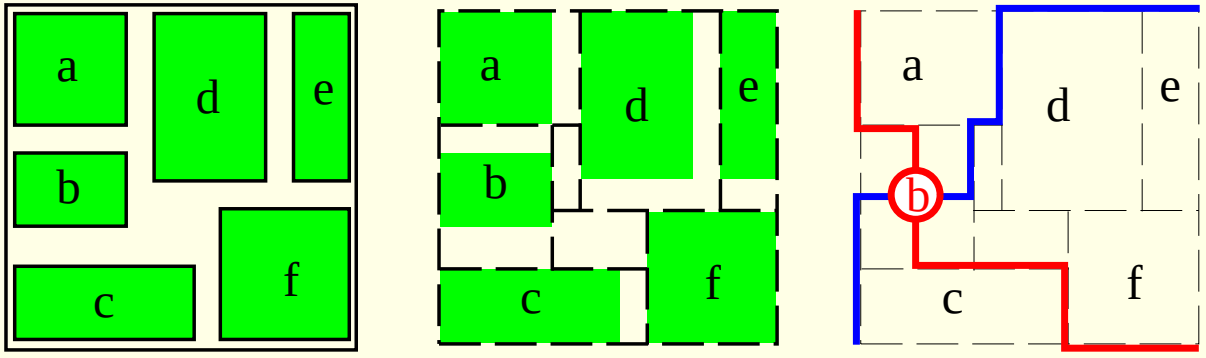
Change on Floorplan





Sequence-Pair Based Floorplanning/Placement

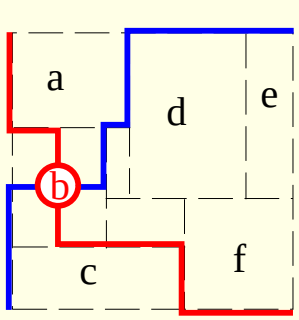
- Murata, et al, ICCAD-95; Nakatake, et al, ICCAD-96; Murata, et al, ISPD-97; Murata and Kuh, ISPD-98; Xu, et al, ISPD-98; Kang and Dai, ISPD-98, ICCAD-98.
- Represent a packing by a pair of module-name sequences (e.g., $(abdecf, cbfade)$).
- Correspond all pairs of the sequences to a P-admissible solution space.
- Search in the P-admissible solution space (typically, by simulated annealing).



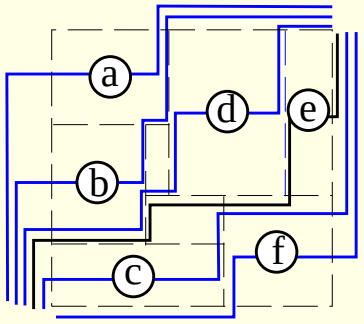
A floorplan Loci of module b

Relative Module Positions

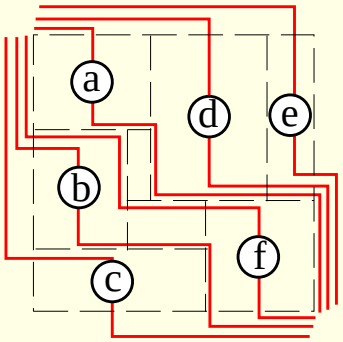
- A floorplan is a partition of a chip into **rooms**, each containing at most one block.
- **Locus** (right-up, left-down, up-left, down-right)
  1. Take a non-empty room.
  2. Start at the center of the room, walk in two alternating directions to hit the sides of rooms.
  3. Continue until to reach a corner of the chip.
- **Positive locus:** Union of right-up locus and left-down locus.
- **Negative locus:** Union of up-left locus and down-right locus.



Loci of module b



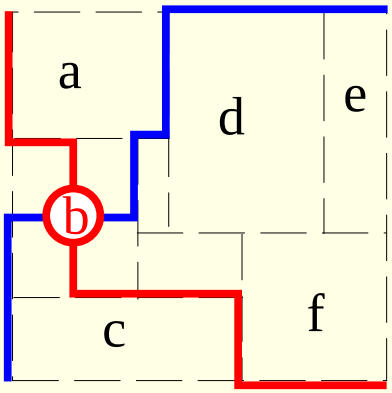
Positive loci: abdecf



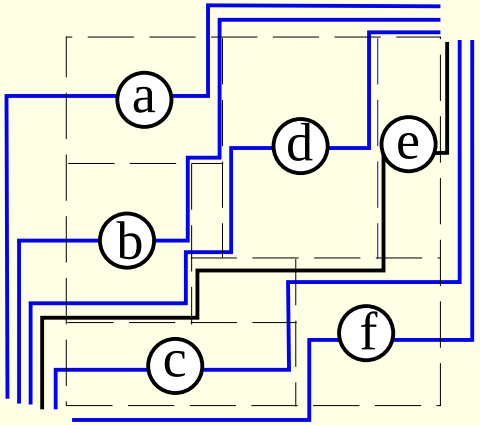
Negative loci: cbfade

Geometrical Information

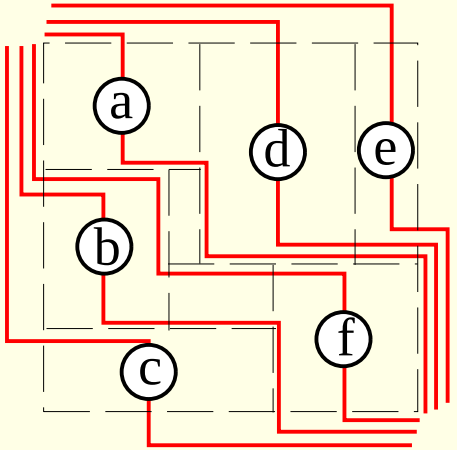
- No pair of positive (negative) loci cross each other, i.e., **loci are linearly ordered.**
- **Sequence Pair** (Γ_+, Γ_-): Γ_+ is a module-name sequence representing the order of positive loci. (Exp: $(\Gamma_+, \Gamma_-) = (abdecf, cbfade)$)
- x' is **after** (**before**) x in both Γ_+ and $\Gamma_- \Longrightarrow x'$ is **right** (**left**) to x.
- x' is **after** (**before**) x in Γ_+ and **before** (**after**) x in $\Gamma_- \Longrightarrow x'$ is **below** (**above**) x.



Loci of module b



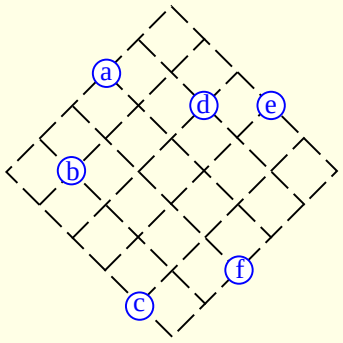
Positive loci: abdecf



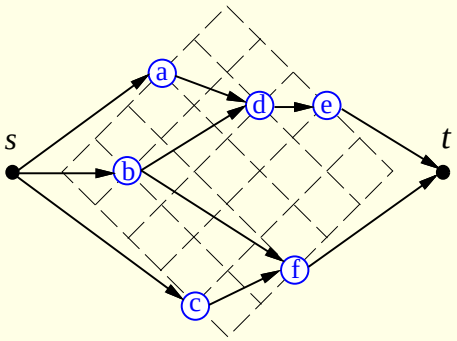
Negative loci: cbfade

(Γ_+, Γ_-)-Packing

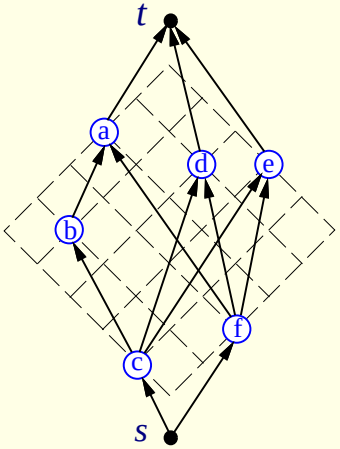
- For every sequence pair (Γ_+, Γ_-), there is a (Γ_+, Γ_-) packing.
- **Horizontal constraint graph** $G_H(V, E)$ (similarly for $G_V(V, E)$):
  - V: source s, sink t, m vertices for modules.
  - E: (s, x) and (x, t) for each module x, and (x, x') iff x must be left-to x'.
  - **Vertex weight:** 0 for s and t, **width** of module x for the other vertices.



Packing for sequence pair: (abdecf, cbfade)



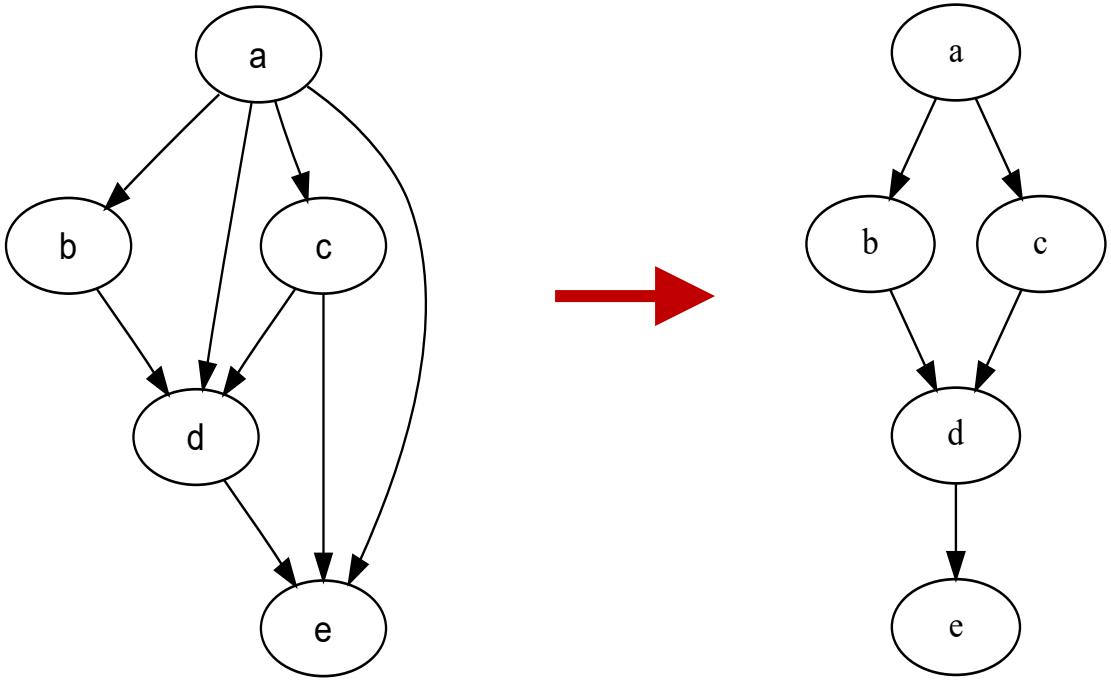
Horizontal constraint graph (Transitive edges are not shown)



Vertical constraint graph (Transitive edges are not shown)

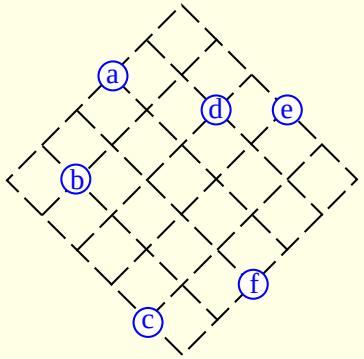
Transitive Reduction

- **Our HCG/VCG are DAG**
  - **Longest path from the source in terms of # of hops**
  - **Then remove the edges not on the longest paths**
  - **This can be done in linear time! use topological sorting**

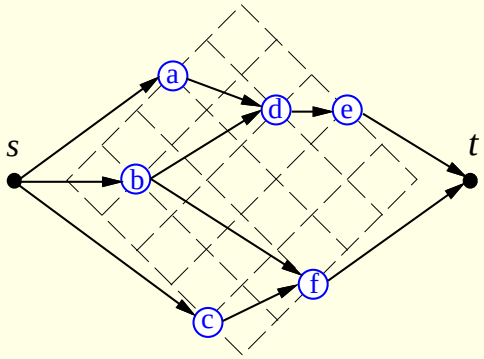


Optimal (Γ_+, Γ_-)-Packing

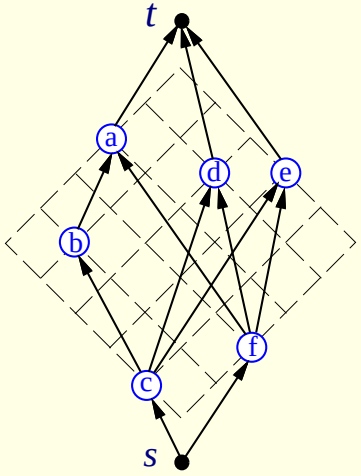
- **Optimal** (Γ_+, Γ_-)-**Packing** can be obtained in $O(m^2)$ time by applying a longest path algorithm on a vertex-weighted directed acyclic graph.
  - G_H and G_V are independent.
  - The X and Y coordinates of each module are determined as the minimum by assigning the longest path length between s and the vertex of the module in G_H and G_V, respectively.
- The set of all sequence pairs is a P-admissible solution space.



Packing for sequence pair: (abdecf, cbfade)



Horizontal constraint graph (Transitive edges are not shown)

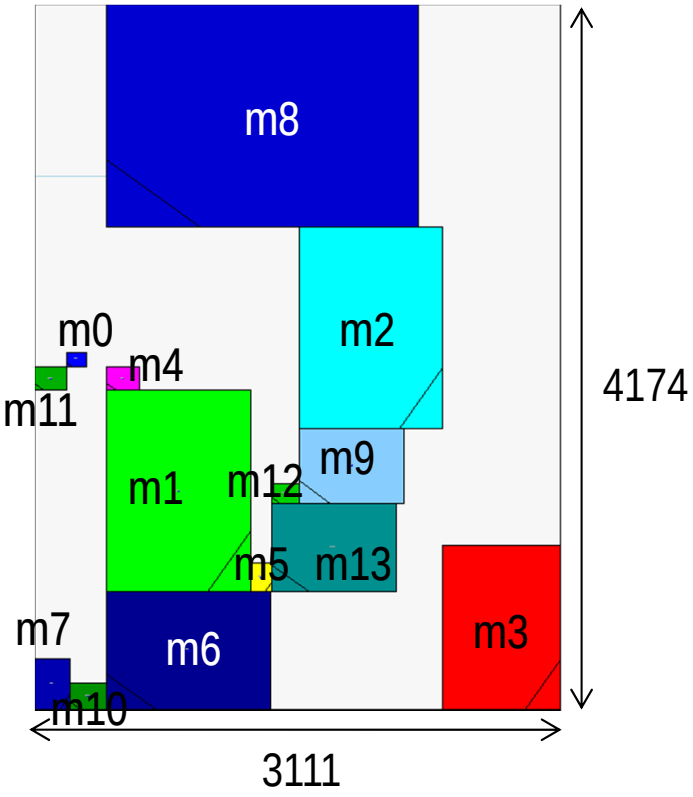


Vertical constraint graph (Transitive edges are not shown)

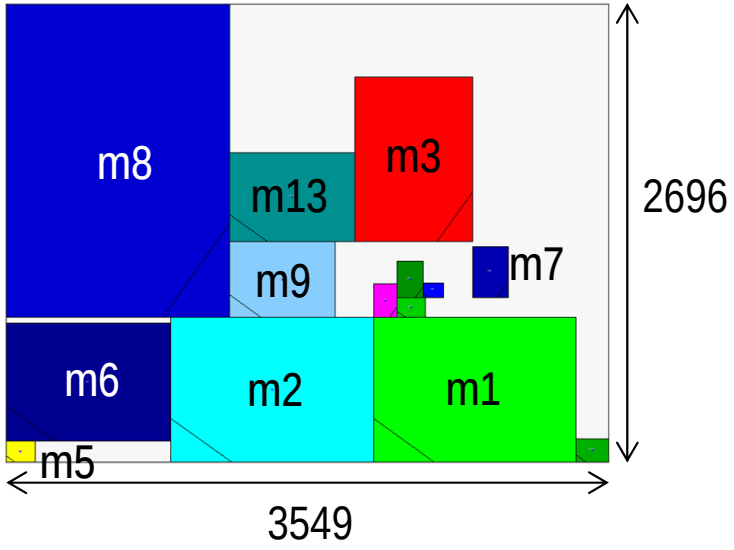
Sequence Pair

- **Final chip area?**
- **Solution space size?**
  - **Without rotation vs with rotation**
- **Optimization: Simulated Annealing**
  - **Initial solution: $\Gamma_+ = \Gamma_-$**
  - **Swap two modules in Γ_+**
  - **Swap two modules both in Γ_+ and Γ_-**
  - **Rotate**
- **Results: produces highly packed non-slicing floorplans**

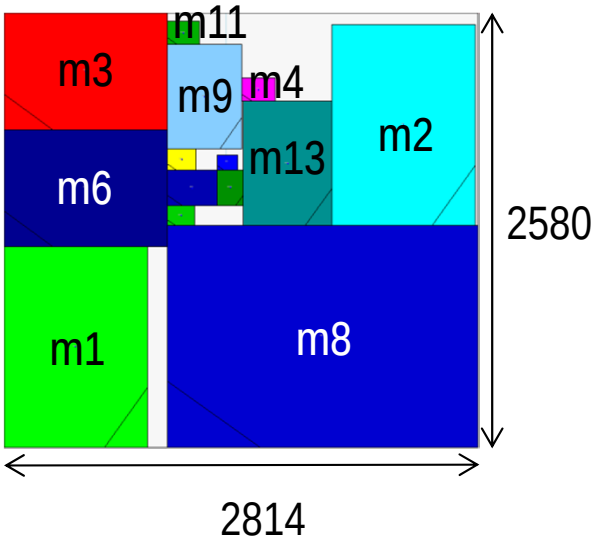
Annealing Temperature vs. Floorplan Quality



(a) temperature: 2000
area: 12985314



(b) temperature: 1000
area: 9568104



(c) temperature: 20
area: 7260120

-26.3%

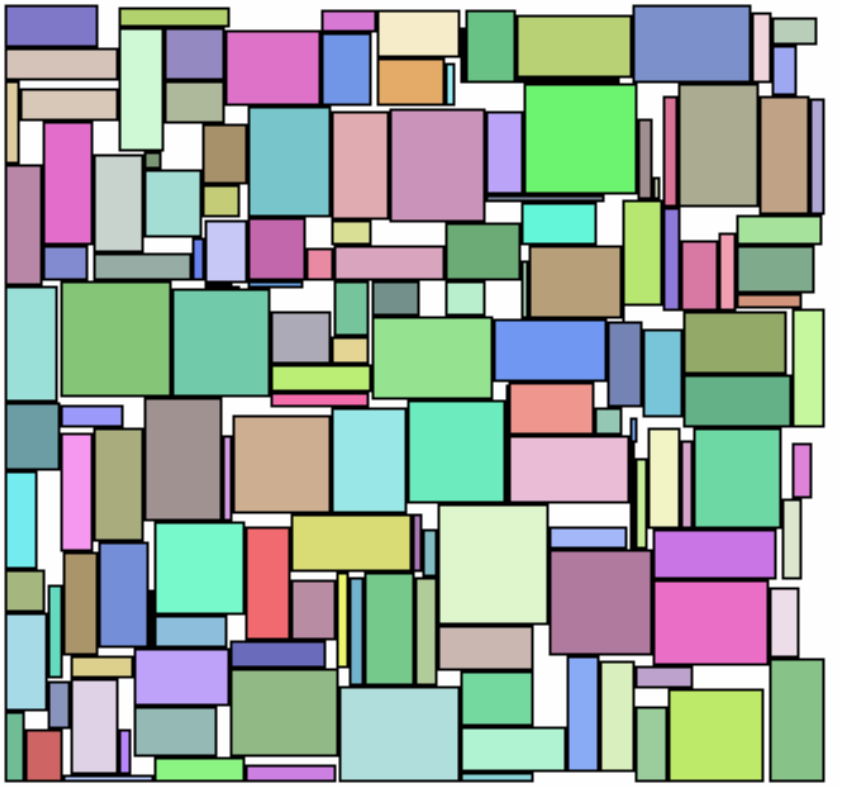
-44.1%

Sequence Pair

- Floorplan (a)
  - S1: 11 0 7 10 8 4 1 5 12 2 9 13 6 3
  - S2: 7 10 6 1 11 5 4 0 13 12 9 2 3 8
- Floorplan (b)
  - S1: 8 6 13 3 9 5 2 4 10 0 7 12 1 11
  - S2: 5 6 2 1 8 9 13 4 12 10 11 0 3 7
- Floorplan (c)
  - S1: 3 11 6 9 5 4 7 0 10 12 13 1 2 8
  - S2: 1 6 8 12 3 7 5 10 0 9 11 13 4 2

Non Slicing Floorplan

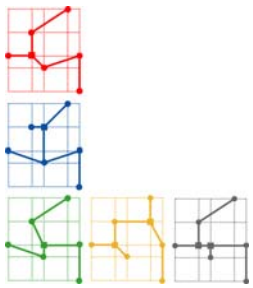
- **Sequence Pair + SA by Adam & Todd (class project)**



Sequence Pair Representation

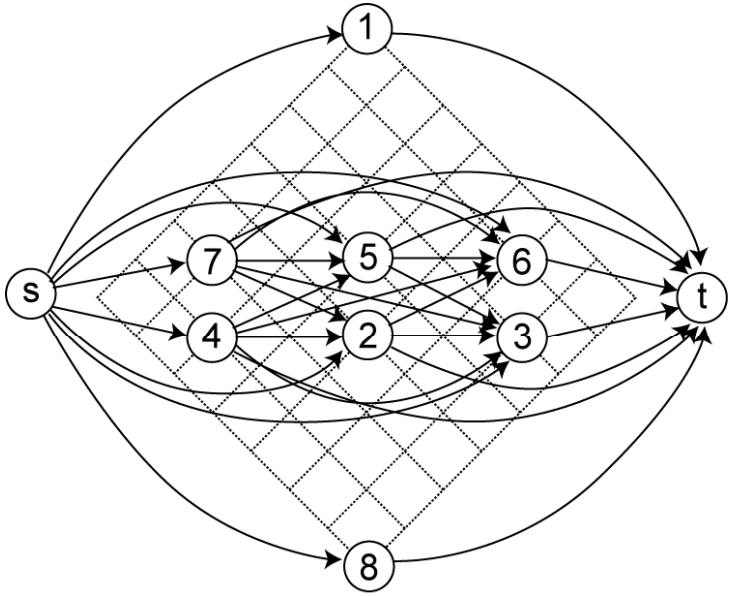
- Initial SP: SP_1 = (17452638, 84725361)
  - Dimensions: (2,4), (1,3), (3,3), (3,5), (3,2), (5,3), (1,2), (2,4)
  - Based on SP_1 we build the following table:

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	$\emptyset$	$\emptyset$	$\{1, 2, 3, 4, 5, 6, 7\}$	$\emptyset$

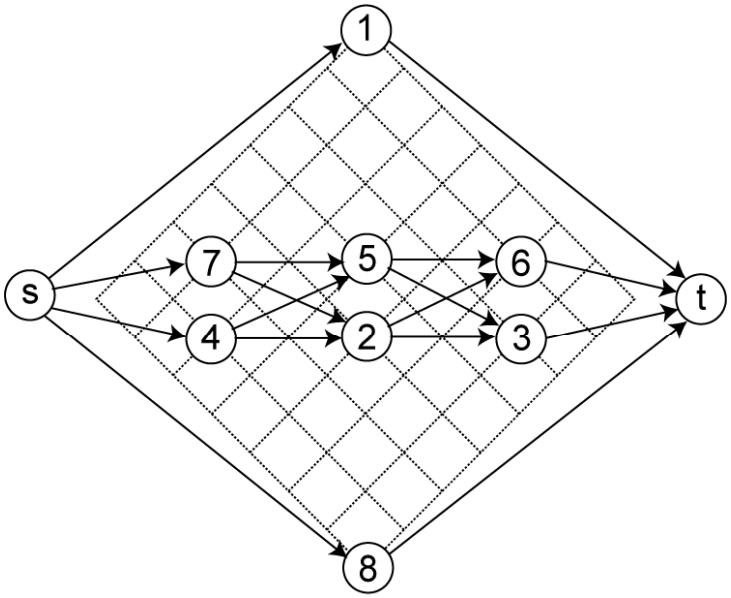


Constraint Graphs

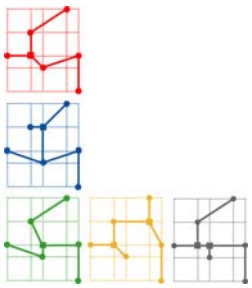
- Horizontal constraint graph (HCG)
  - Before and after removing transitive edges



(a)

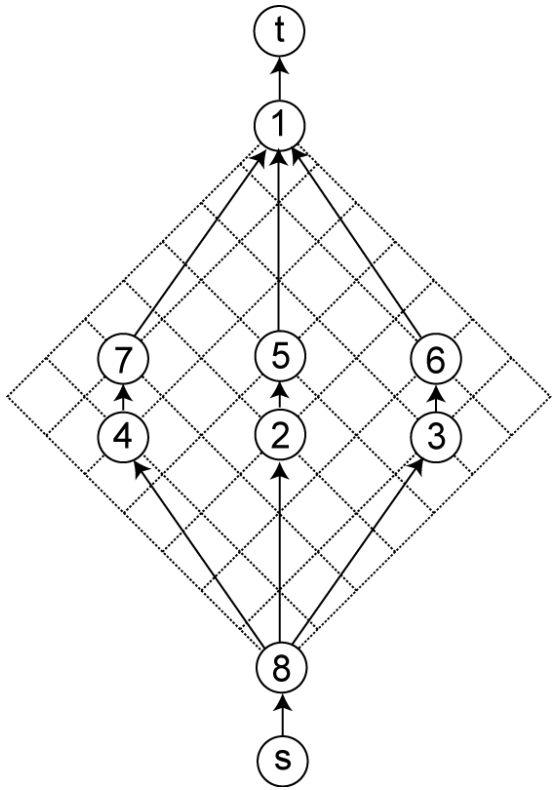


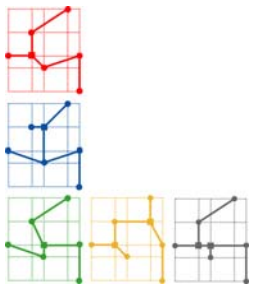
(b)



Constraint Graphs (cont)

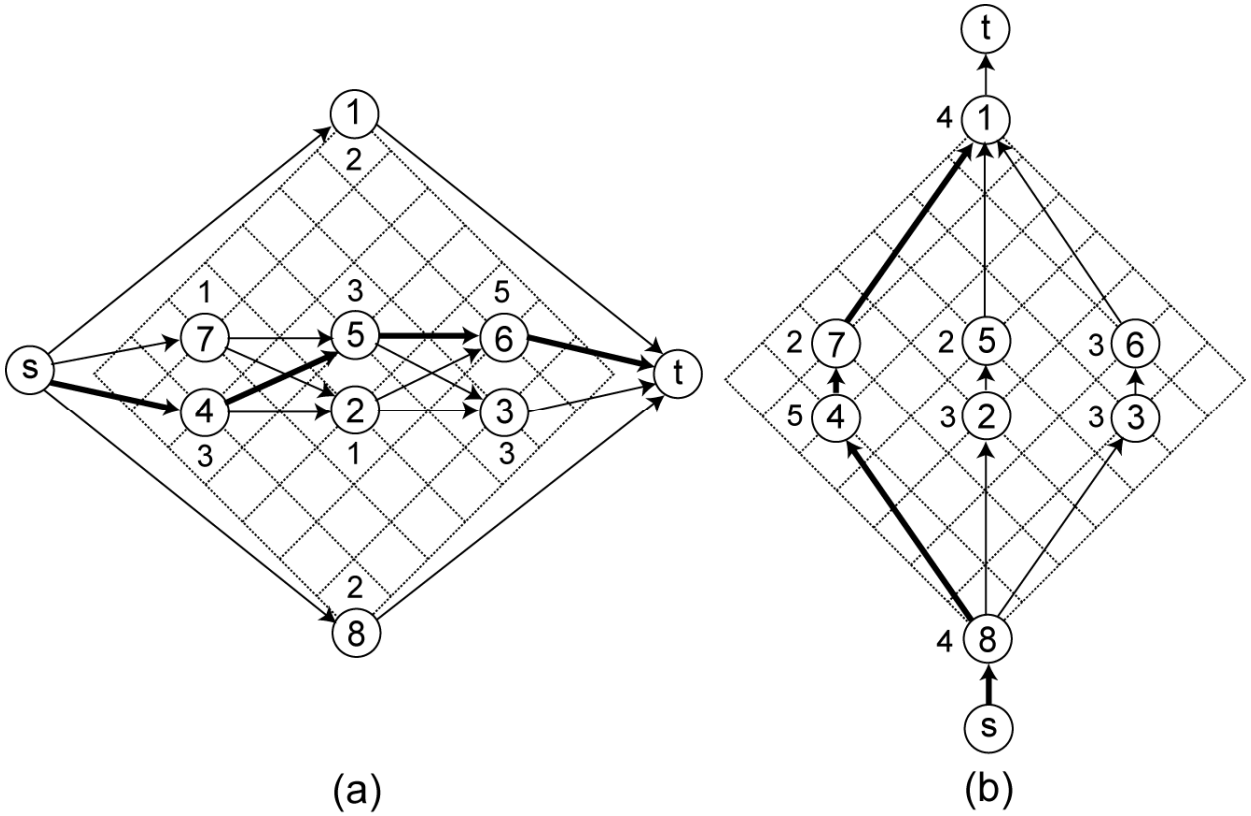
- Vertical constraint graph (VCG)





Computing Chip Width and Height

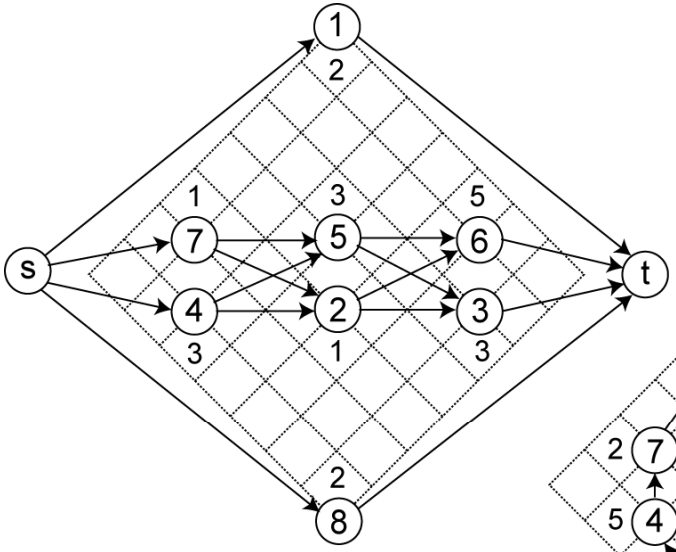
- Longest source-sink path length in:
  - HCG = chip width, VCG = chip height
  - Node weight = module width/height



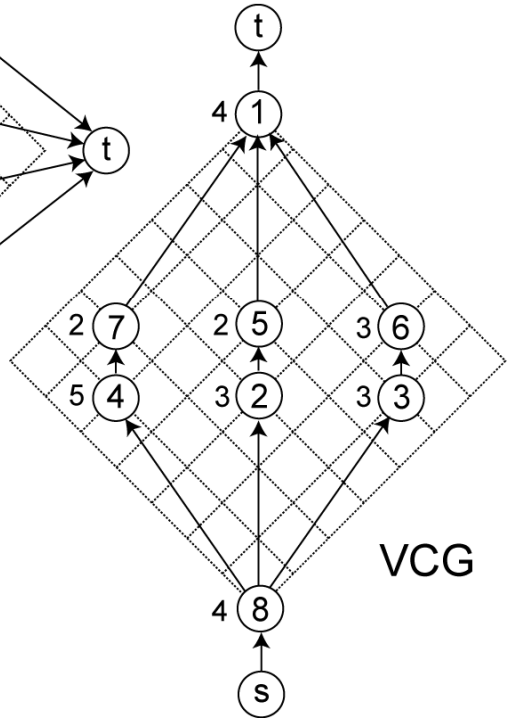
Computing Module Location

- Use longest source-module path length in HCG/VCG
  - Lower-left corner location = source to module input path length

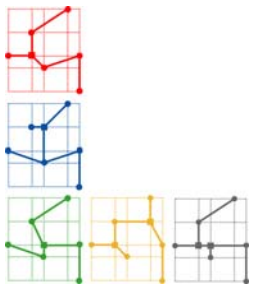
module	HCV	VCG
1	0	11
2	3	4
3	6	4
4	0	4
5	3	7
6	6	7
7	0	9
8	0	0



HCG

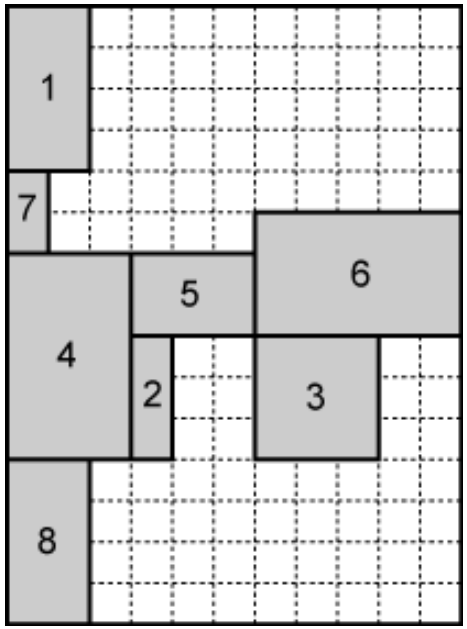


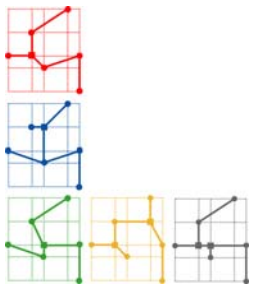
VCG



Final Floorplan

- Dimension: 11 × 15

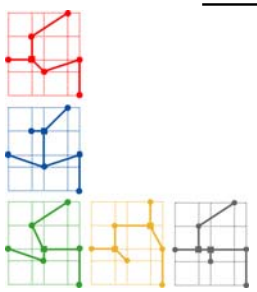




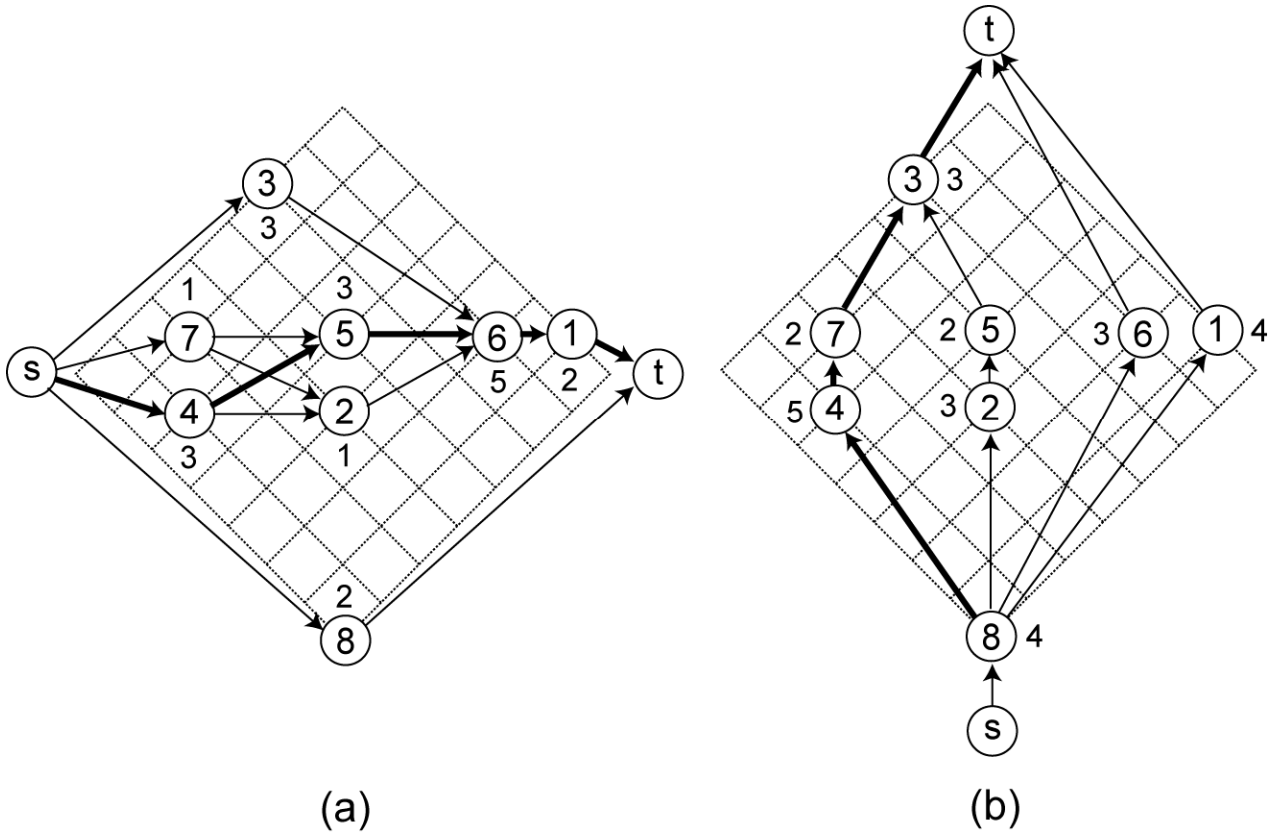
Move I

- Swap 1 and 3 in positive sequence of SP_1
  - SP_1 = (17452638, 84725361)
  - SP_2 = (37452618, 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	$\emptyset$	$\emptyset$	$\{1,2,3,4,5,6,7\}$	$\emptyset$



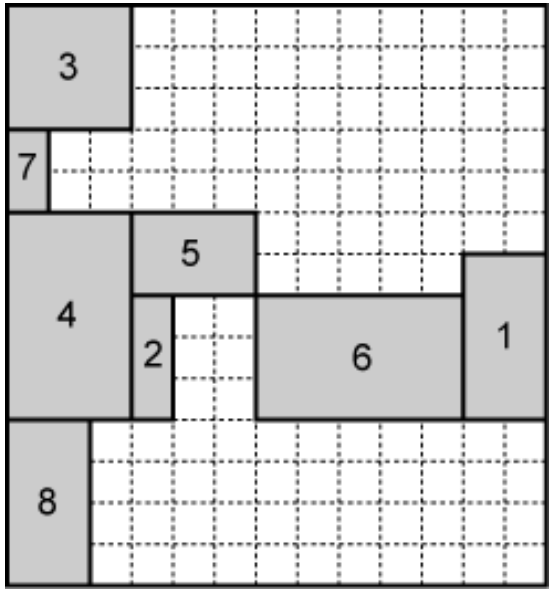
Constraint Graphs

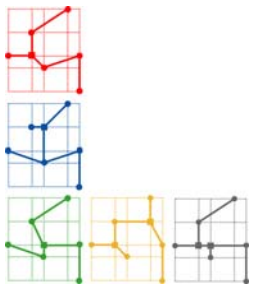


Constructing Floorplan

- Dimension: 13 × 14

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

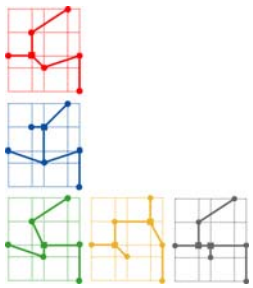




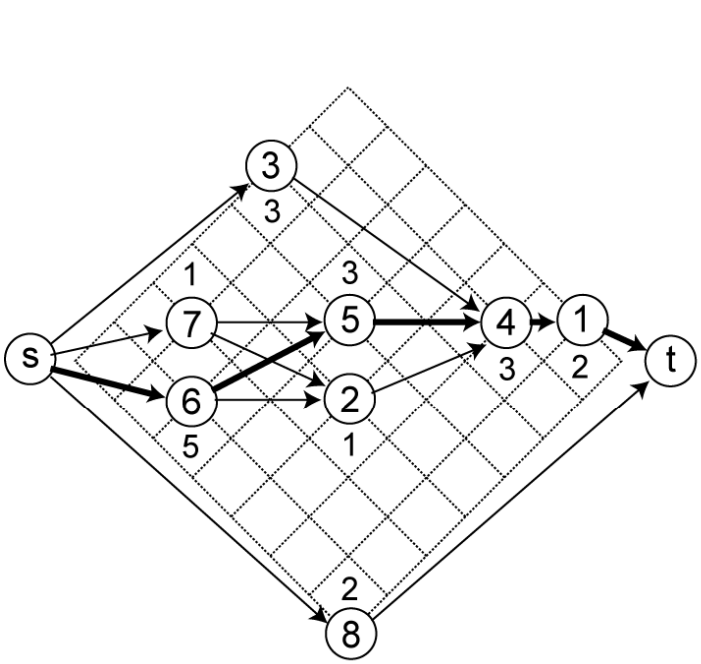
Move II

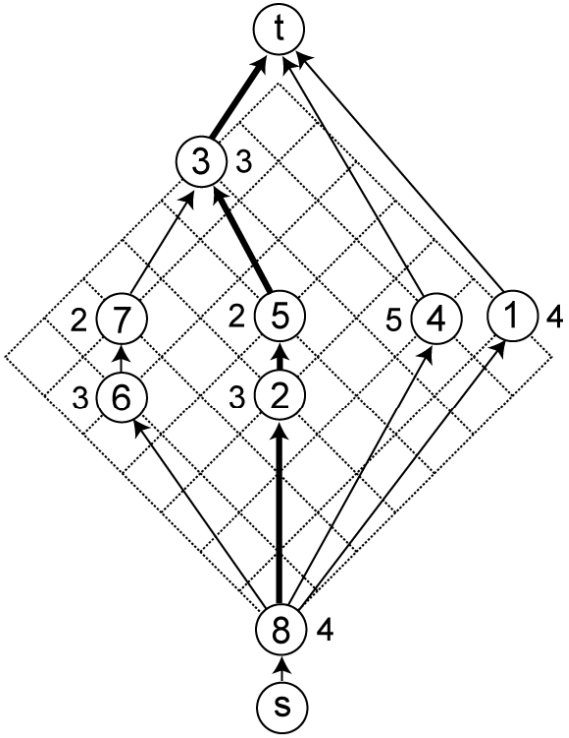
- Swap 4 and 6 in both sequences of SP_2
  - SP_2 = (37452618, 84725361)
  - SP_3 = (37652418, 86725341)

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

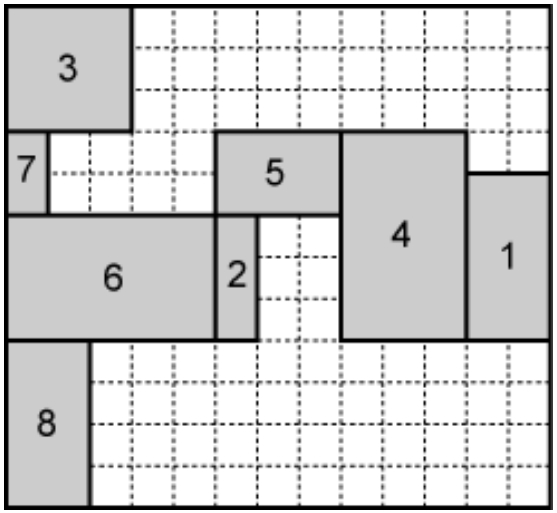


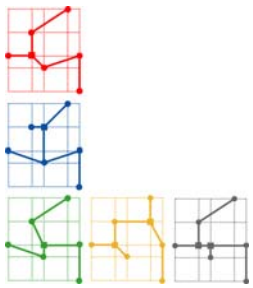


Constructing Floorplan

- Dimension: 13 × 12

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





Summary

- Impact of the moves:
  - Floorplan dimension changes from 11 × 15 to 13 × 14 to 13 × 12