Physical Design of Digital Integrated Circuits (EN0291 S40)

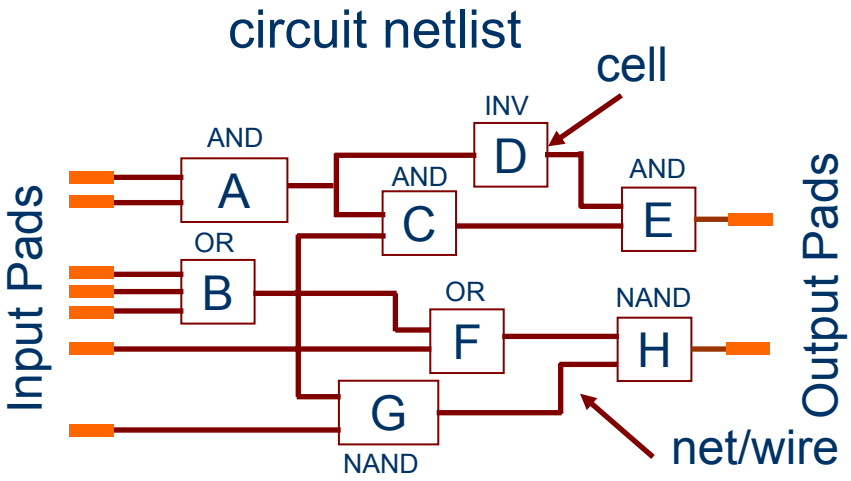
Sherief Reda
Division of Engineering, Brown University
Fall 2006

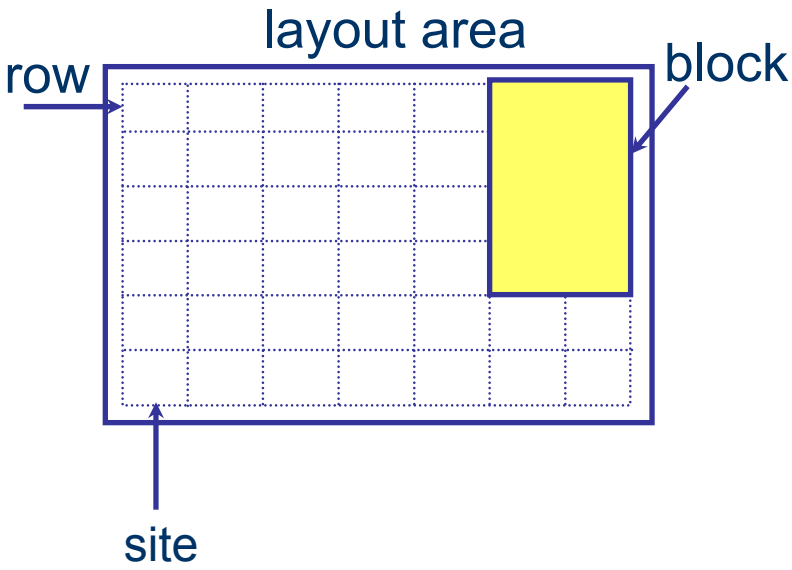
BROWN

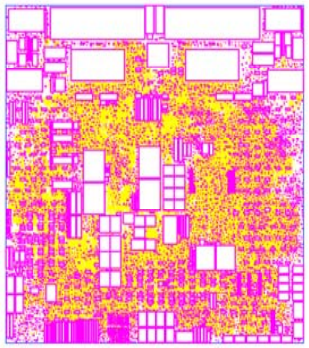
Lecture 07: Floorplanning and Placement

- Introduction to Placement and Floorplanning
- Global Placement
- Legalization and Detailed Placement
- Floorplanning

Placement maps cells onto the layout area such that the total wirelength is minimized







IBM BigBlue4 — 2.1 million gates

Why placement and wirelength are important?

BROWN

As technology scales, wires dominate devices in determining performance and power

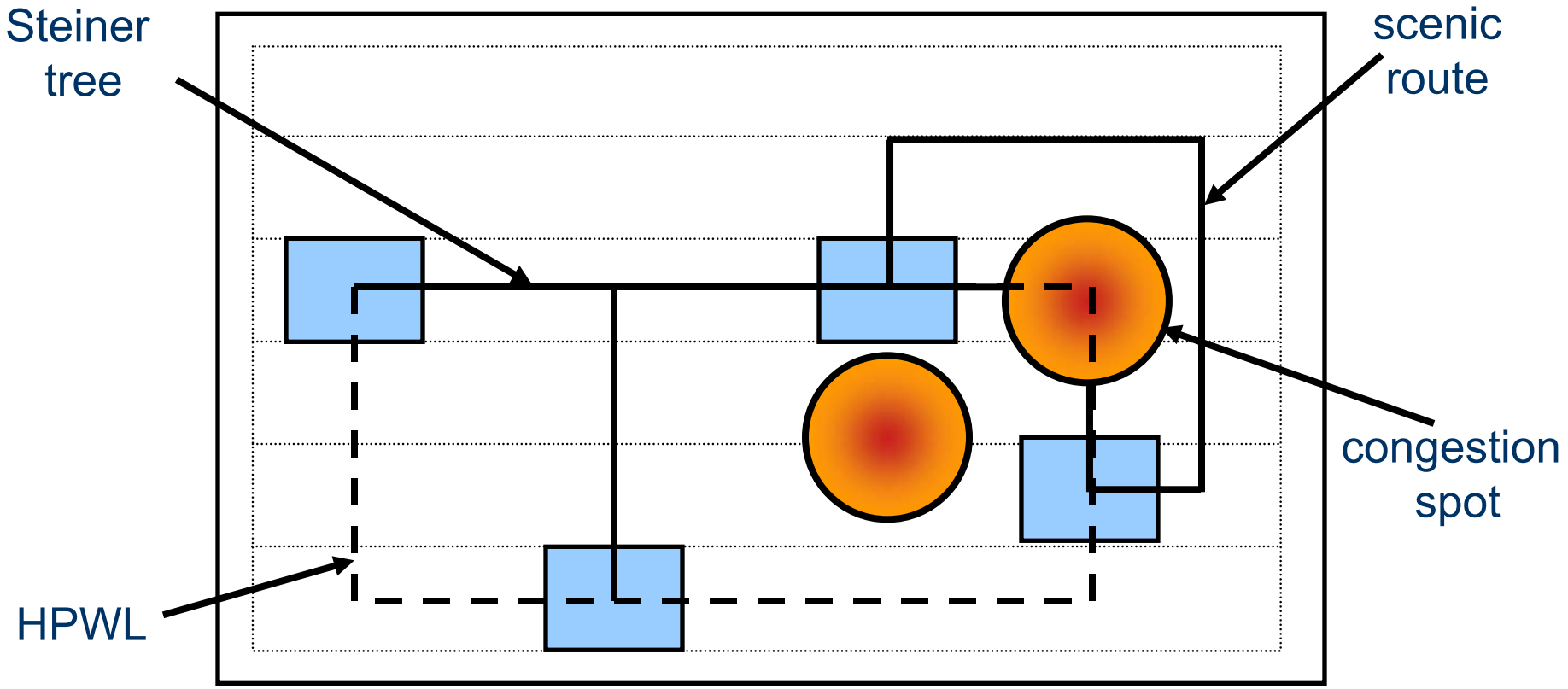
- Scaling → more devices and wires → design complexity
- Wire delays relative to device delays have increased by ~1000× in a span of 8 generations (250nm → 32nm)
  ⇒ reducing wirelength
  → reduces wire delay
  → improves performance

- Reducing wirelength → capacitance → power

How to measure wirelength?

BROWN

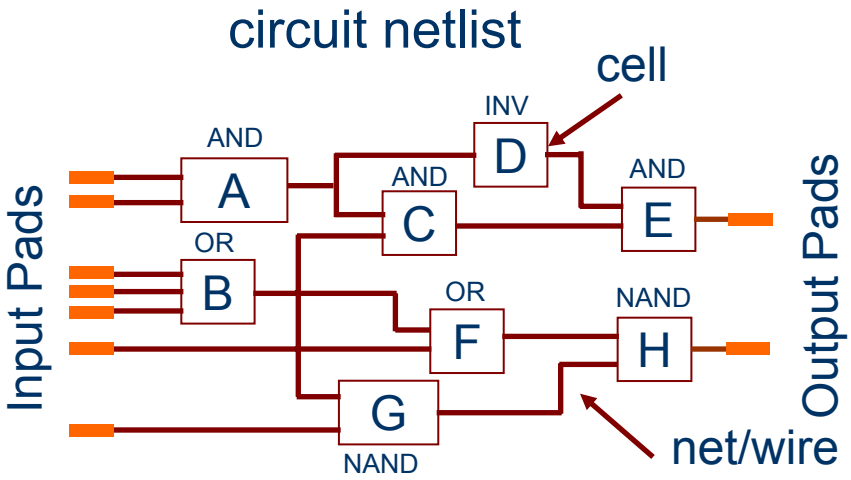
How can we calculate the wirelength?

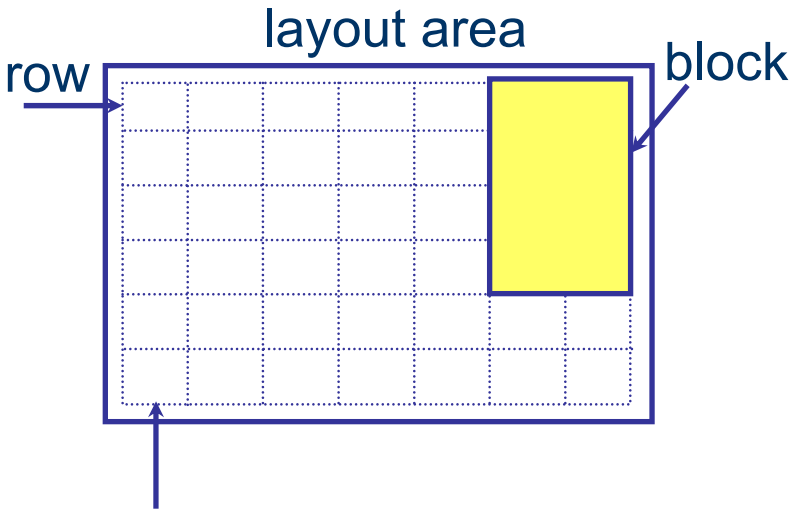


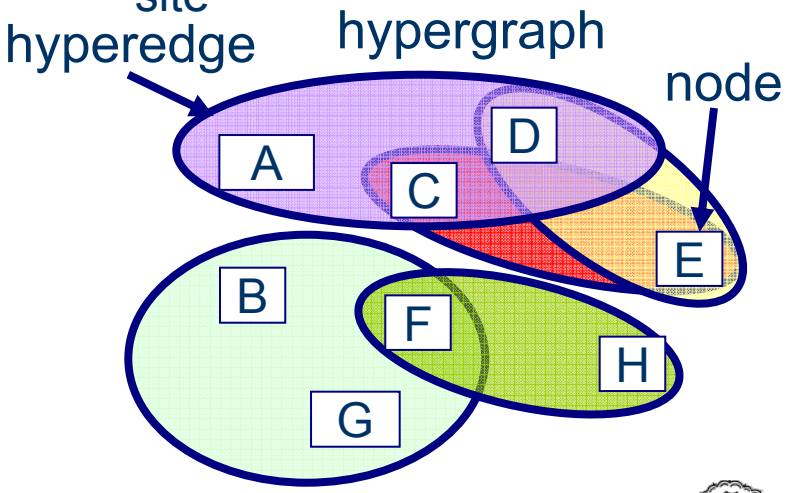
- HPWL = Half Perimeter Wirelength (more practical)
- HPWL ≤ Steiner length ≤ routing length

BROWN

The placement problem stated formally







Placement Problem: Given a hypergraph $H=(V, E)$, find a non-overlapping mapping π for the nodes of the hypergraph onto the layout area such that the total HPWL (Steiner length) is minimized.

BROWN

The placement problem is notoriously hard

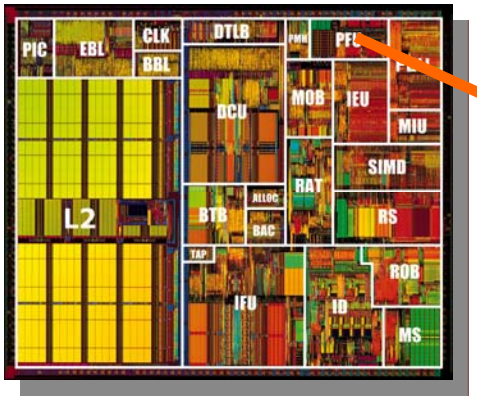
- NP-hard [Sahni76]
- Optimal placements can be practically found only for a few components (~20 – 30)
- No approximation algorithms unless *P* = *NP* [Queyranne86]

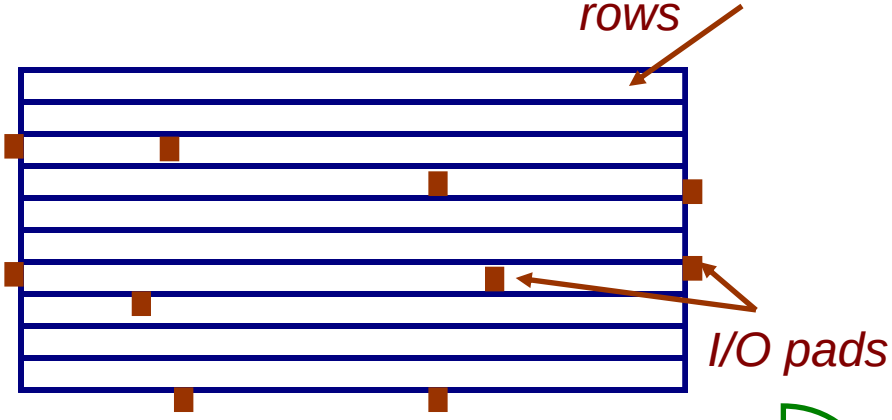
$\Rightarrow$ must rely on heuristic algorithms

$\Rightarrow$ 40+ years of research

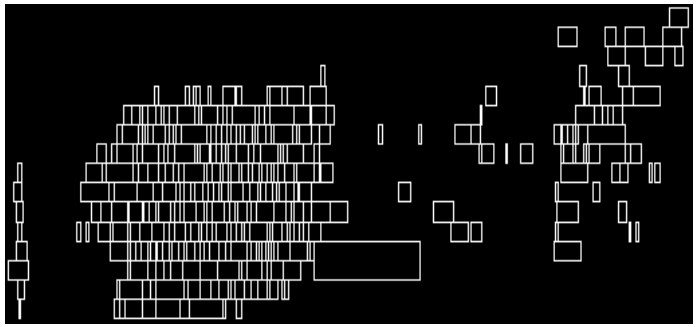
BROWN

Floorplanning and placement

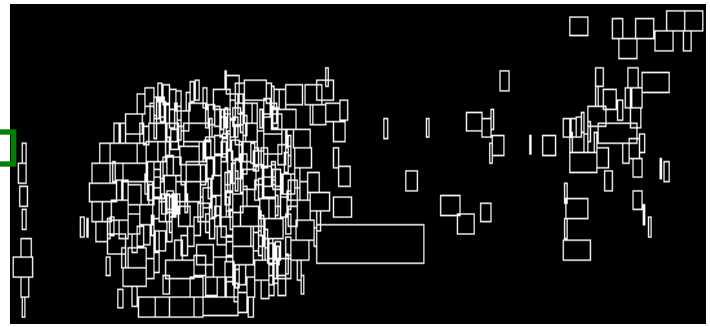




Floorplanning is a small-scale placement problem with large modules



[legal placement]



[illegal placement]

Placement of standard cells is a large-scale 2-D assignment problem

$\Rightarrow$ determines positions for thousands/millions of standard cells

BROWN

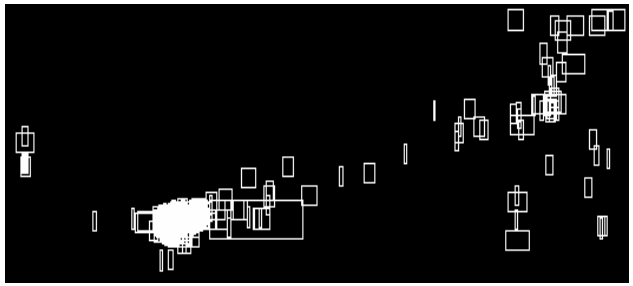
Lecture 07: Floorplanning and Placement

- Introduction to Placement and Floorplanning
- Global Placement
- Legalization and Detailed Placement
- Floorplanning

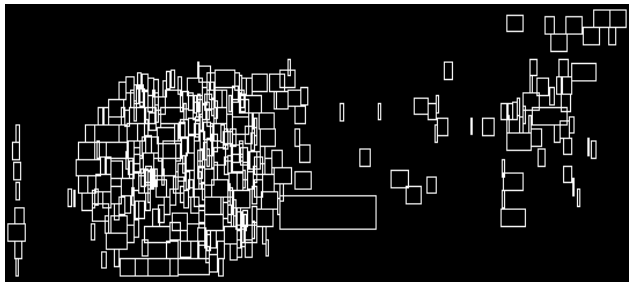
BROWN

Placement algorithms' two central questions

- *How to model and minimize wirelength?* (i.e., how to model the interconnections between cells)

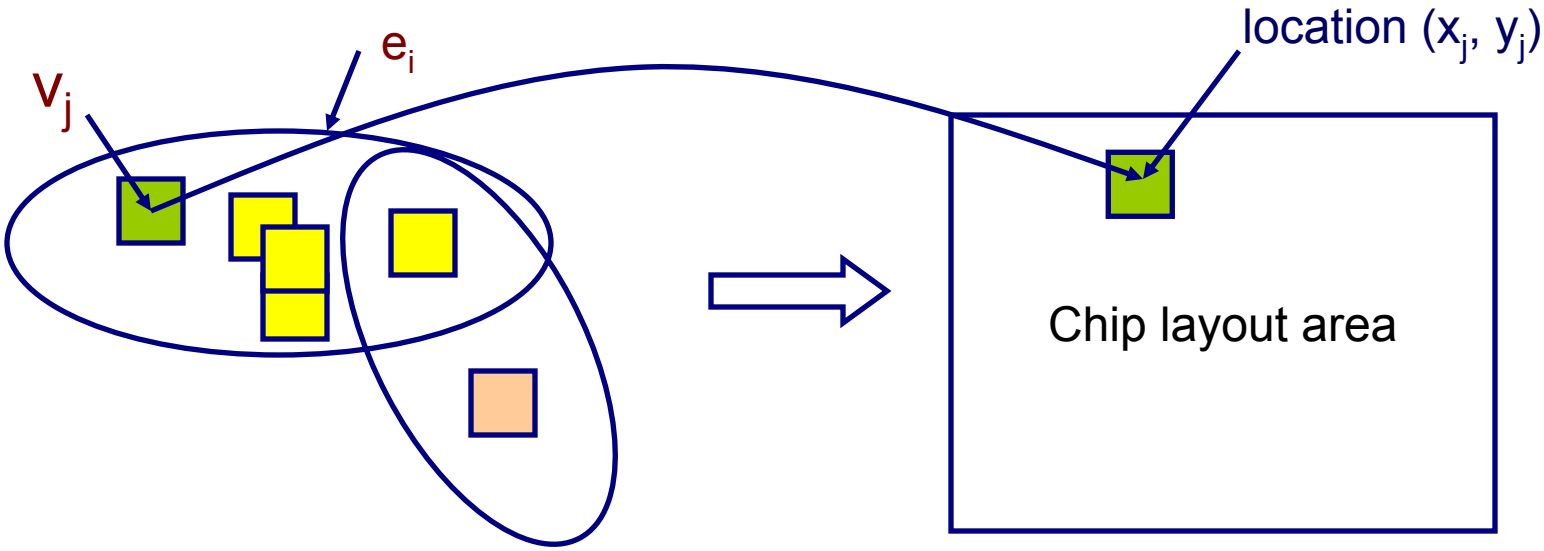


- *How to spread the cells?* (how to avoid collapsing of the cells)



BROWN

Ideal modeling of wirelength



The total wirelength minimization problem: find a position for each cell a position such that the total wirelength (measured by the half-perimeter wirelength) is minimized

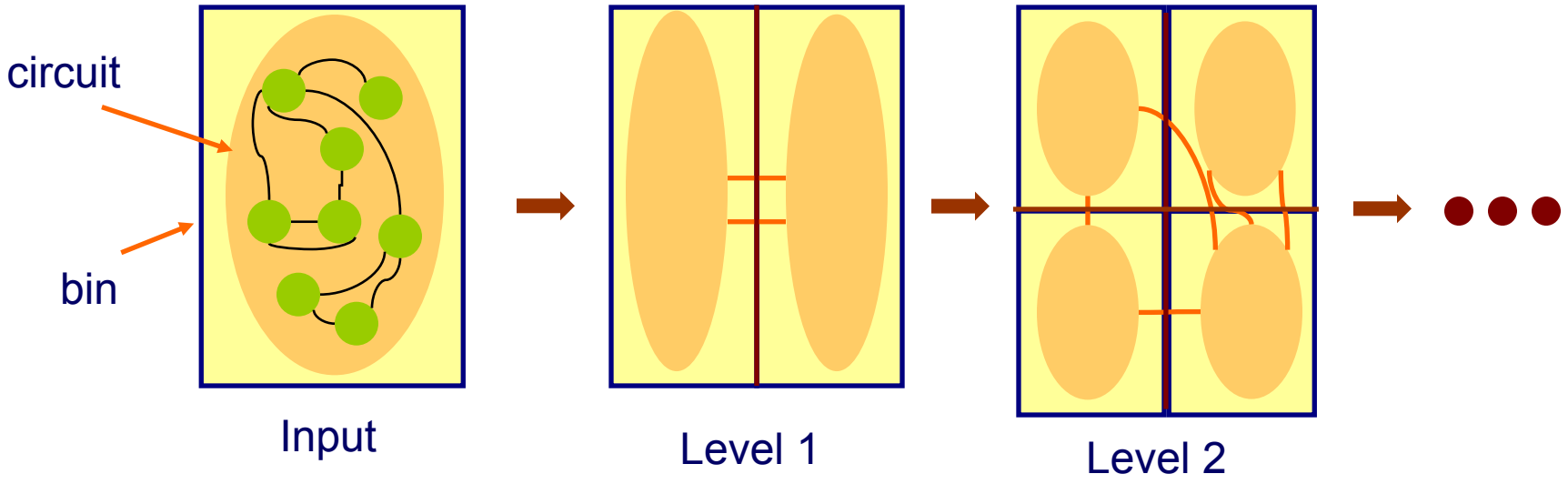
$$z = \sum_{e_i \in E} (\max_{v_j \in e_i} x_j - \min_{v_j \in e_i} x_j + \max_{v_j \in e_i} y_j - \min_{v_j \in e_i} y_j)$$

BROWN

We will focus on three main placement technologies

I. Placers based on recursive min-cut partitioning
II. Placers based on simulated annealing
III. Placers based on analytical techniques

I. Placers based on recursive min-cut paritioning



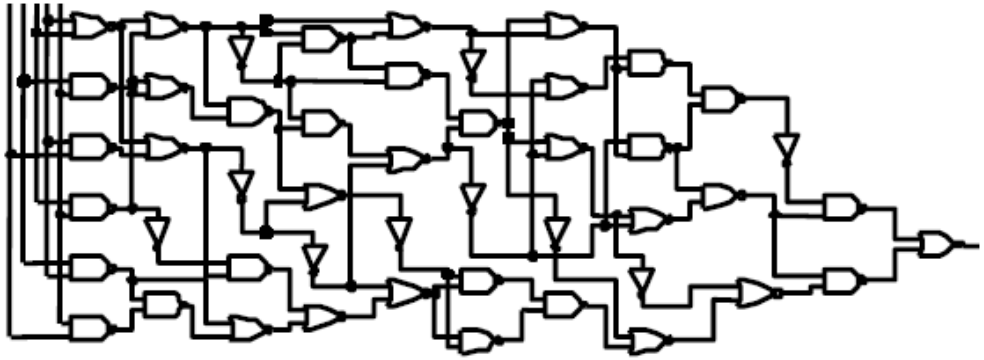
- Two main questions:

Q_1: How to partition a hypergraph?

Q_2: How to propagate net connectivity information from one block to another?

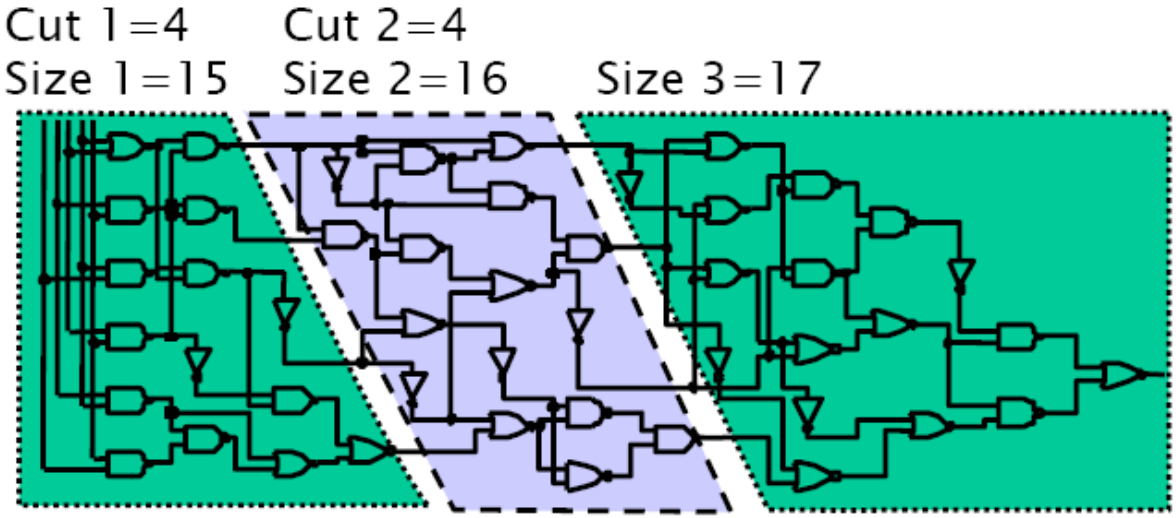
BROWN

Q_1. Min-cut placers: how to partition a circuit?



Size: 48

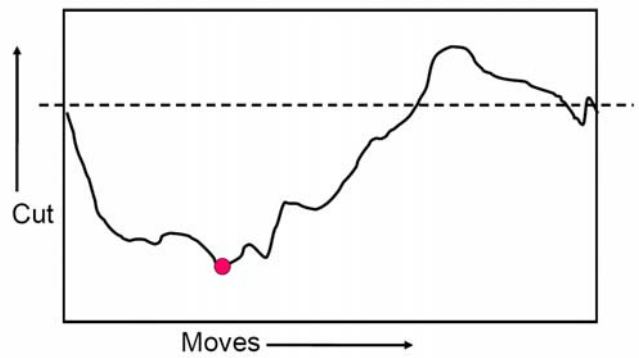
How?



[source: I. Markov]

BROWN

A_1. Fiduccia-Mattheyses (FM) method for finding a min-cut partition



Pass:

– start with all vertices free to move (*unlocked*)
– label each possible move with immediate change in cost that it causes (*gain*)
– iteratively select and execute a move with highest gain, *lock* the moving vertex (i.e., cannot move again during the pass), and update affected gains
– best solution seen during the pass is adopted as starting solution for next pass

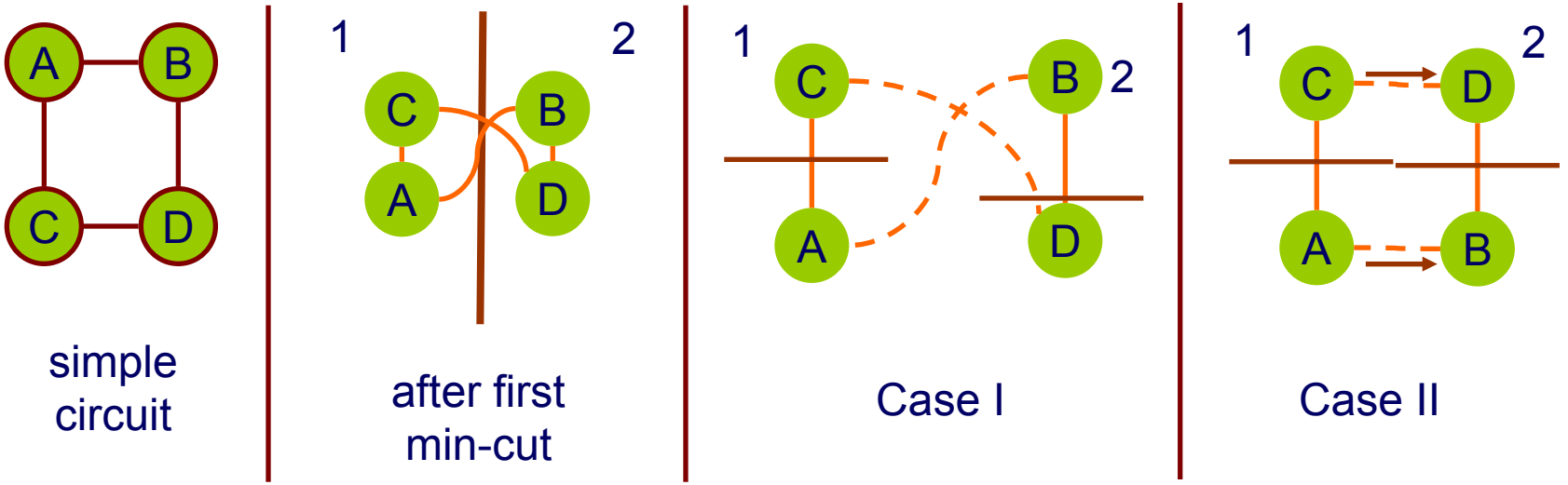
⇒ FM:

– start with some initial solution
– perform passes until a pass fails to improve solution quality

[source: I. Markov]

BROWN

Q_2: Partitioning subcircuits in isolation does not lead to a global optimal; how to fix that?

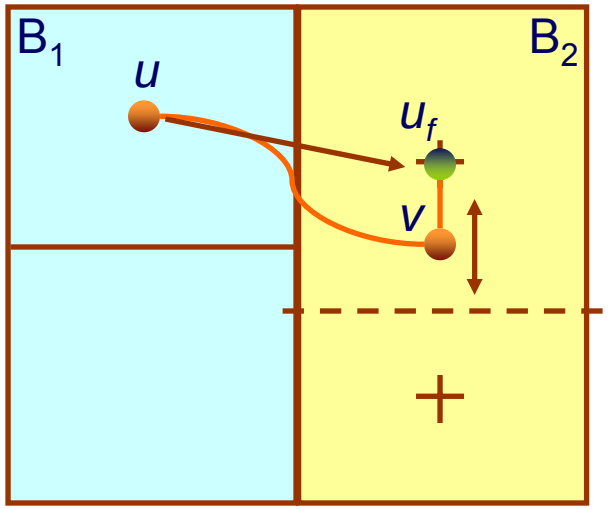


- Case I: Blocks are partitioned in isolation → optimal local partitioning results but far from optimal global results

- Case II: Information about cells in one block are accounted for in the other block → local partitioning results are translated to global wirelength results

BROWN

A_2: Terminal propagation is good technique to capture global connectivity

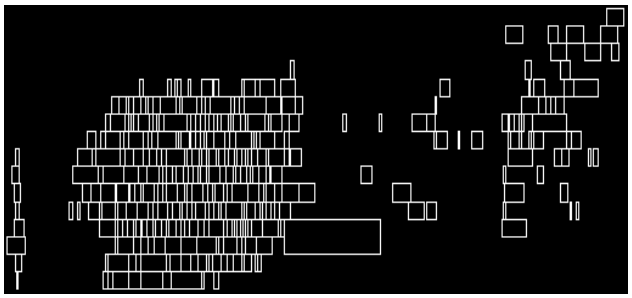


- B_1 has been partitioned; **B_2 is to be partitioned**
- *u* is propagated as a fixed vertex u_f to the subblock that is closer
- u_f biases the partitioner to move *v* upward

BROWN

II. Placers based on simulated annealing

Suppose you have a placement that you want to improve. What can you do?

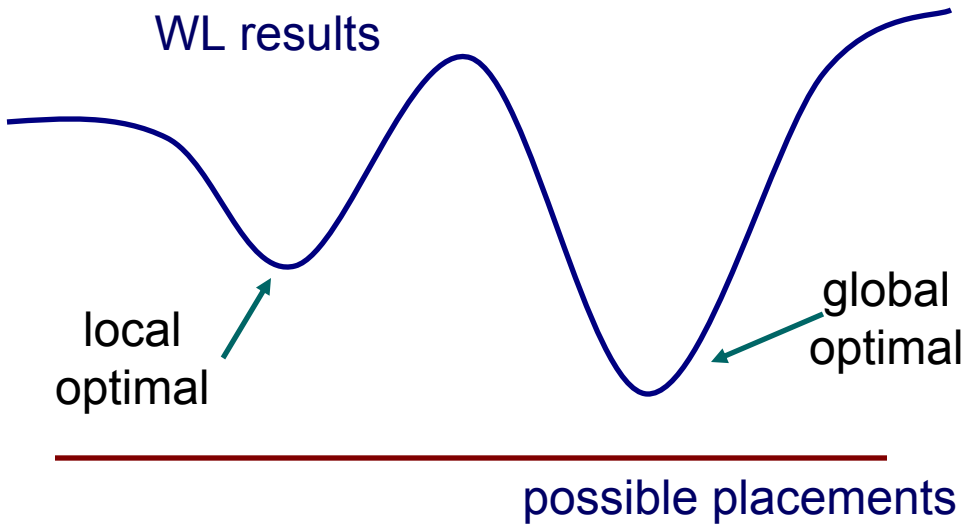


Possible algorithm:

- Pick a pair of cells and swap their locations if this leads to reduction in WL

What's wrong with the previous greedy algorithm?

$\Rightarrow$ It can simply get stuck in a local optimal result

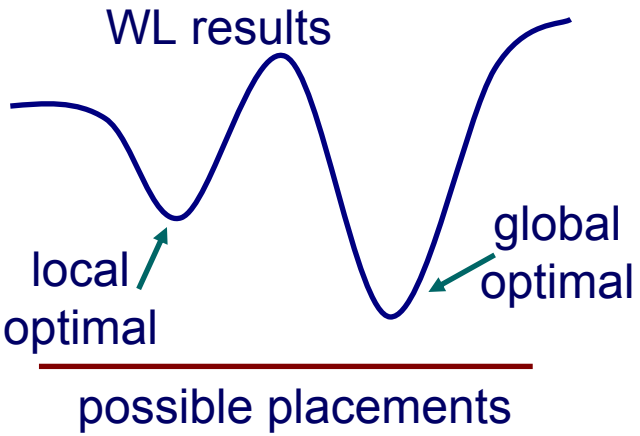


BROWN

Simulated annealing allows us to avoid getting trapped in a local minima

Modified algorithm

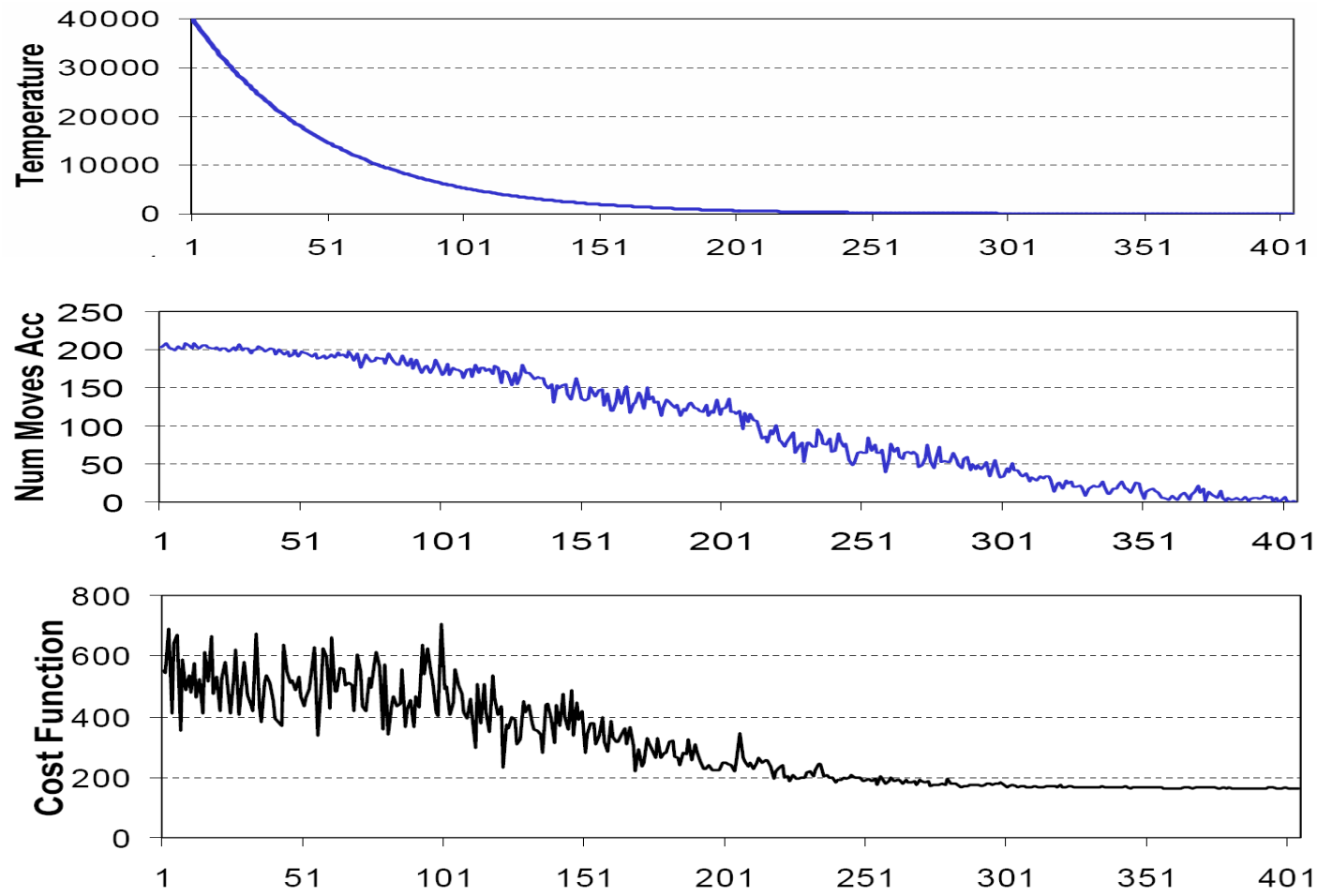
- Generate a random move (say a swap of two cells)
  - calculate the chance in WL (ΔL) due to the move
  - if the move results in reduction ($\Delta L < 0$) then *accept*
  - else reject with probability $1\text{-}e^{-\Delta L/T}$



- T (temperature) controls the rejection probability
- Initially, T is high (thus avoiding getting trapped early in a local minima) then the temperature *cools down in a scheduled* manner; at the end, the rejection probability is 1
- With the right "slow-enough" cooling scheduling, simulated annealing is guaranteed to reach the global optimal

BROWN

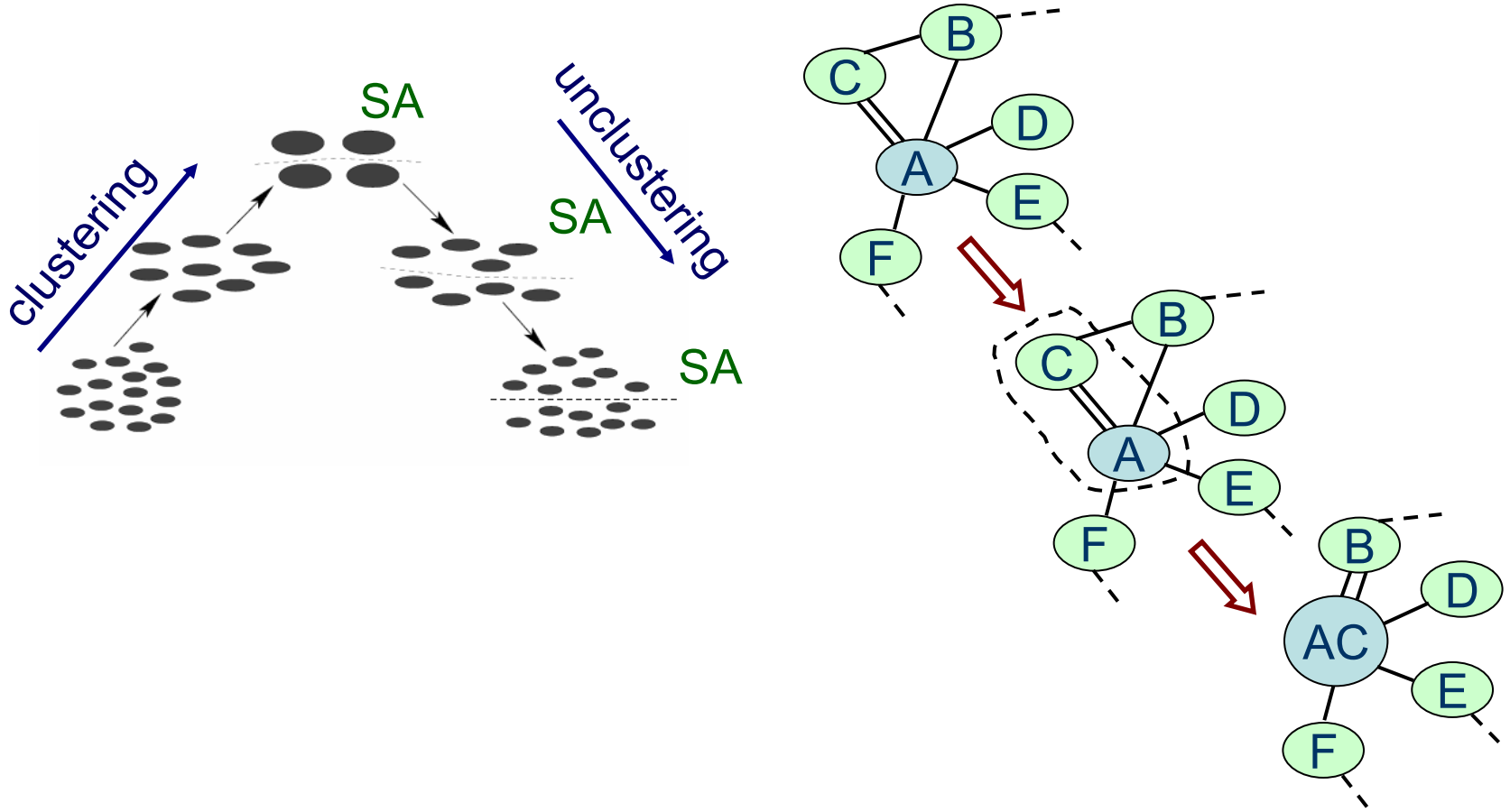
How do the cooling scheduling and corresponding cost functions look like?



[source: I. Markov]

BROWN

Clustering reduces SA runtime

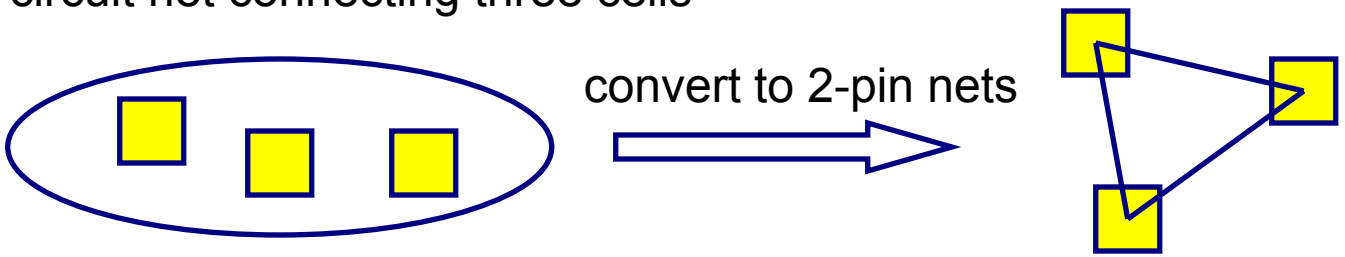


- Challenge: how to do the clustering without hurting the final wirelength/performance results?

BROWN

III. Placers based on analytical techniques

A circuit net connecting three cells



$$z_1 = \frac{1}{2}\sum_{i=1}\sum_{j=1} c_{ij}(|x_i - x_j| + |y_i - y_j|)$$

Hard to solve analytically, but can be approximated as

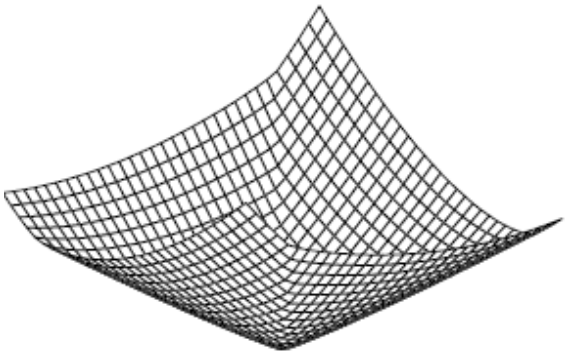
$$z_2 = \frac{1}{2}\sum_{i=1}\sum_{j=1} c_{ij}((x_i - x_j)^2 + (y_i - y_j)^2)$$

$\Rightarrow$ Don't forget that some cells are actually I/O pads that have fixed positions

BROWN

The quadratic program is convex → its optimal solution can be readily found

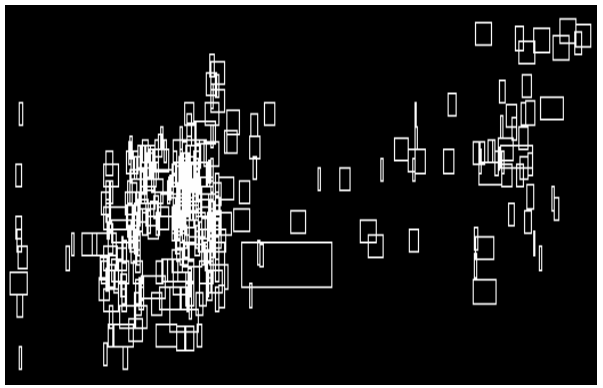
$$z_2 = \frac{1}{2} \sum_{i=1} \sum_{j=1} c_{ij}((x_i - x_j)^2 + (y_i - y_j)^2)$$



convex surface

- Quadratic programs are convex, thus the global optimal can be generally found "quickly" (e.g., using conjugant-gradient methods)

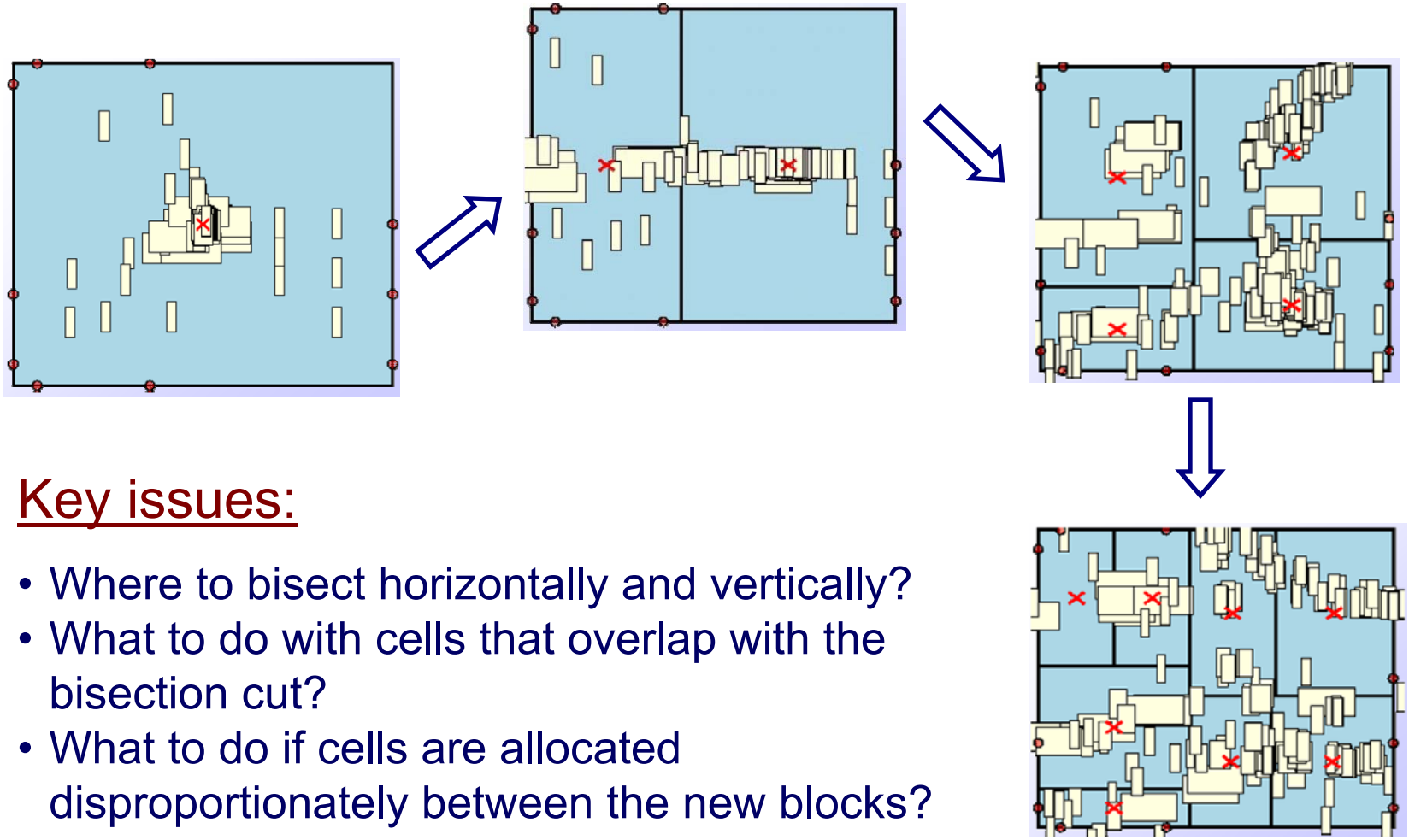
- However, the resultant placement are highly overlapping.
  - How to spread cells and remove the overlap?



results of solving a quadratic placement program

BROWN

Spreading cells by using top-down partitioning



Key issues:

- Where to bisect horizontally and vertically?
- What to do with cells that overlap with the bisection cut?
- What to do if cells are allocated disproportionately between the new blocks?

BROWN

Timing-driven placement

- Timing-driven placement emphasizes (prioritizes) the minimization of the lengths of wires that belong to the critical paths of the circuit
- For example, instead of solving the following purely wirelength-driven function

$$z = \sum_{e_i \in E} (\max_{v_j \in e_i} x_j - \min_{v_j \in e_i} x_j + \max_{v_j \in e_i} y_j - \min_{v_j \in e_i} y_j)$$

- Solve the following weighted function

$$z_T = \sum_{e_i \in E} w_i (\max_{v_j \in e_i} x_j - \min_{v_j \in e_i} x_j + \max_{v_j \in e_i} y_j - \min_{v_j \in e_i} y_j),$$

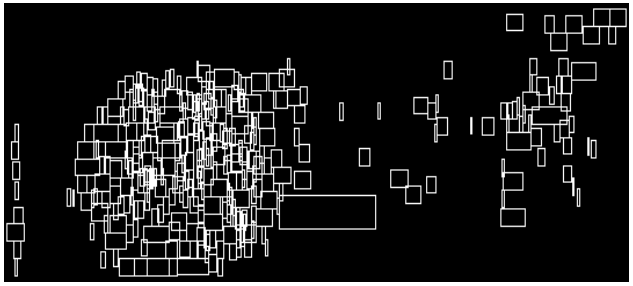
where w_i is a weight that reflect the *criticality* of the wire respect to timing (the criticality is figured out from STA)

BROWN

Lecture 07: Floorplanning and Placement

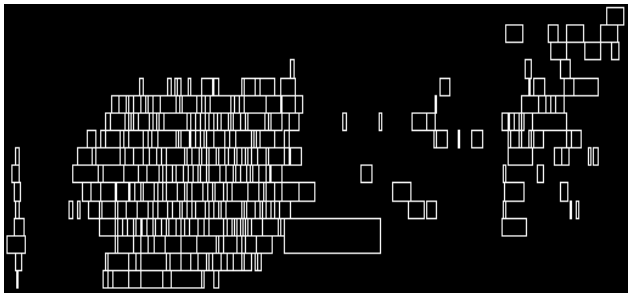
- Introduction to Placement and Floorplanning
- Global Placement
- Legalization and Detailed Placement
- Floorplanning

Placement legalization snaps cells on the layout rows



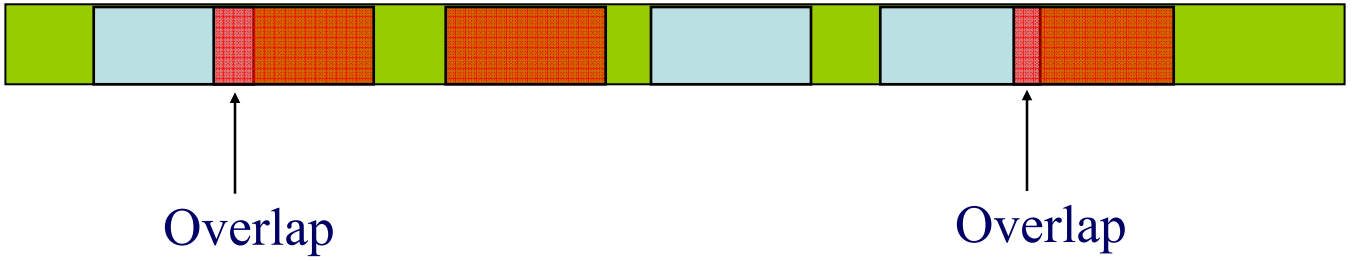
an illegal placement

How to do this w/o harming WL/timing?



legalized placement

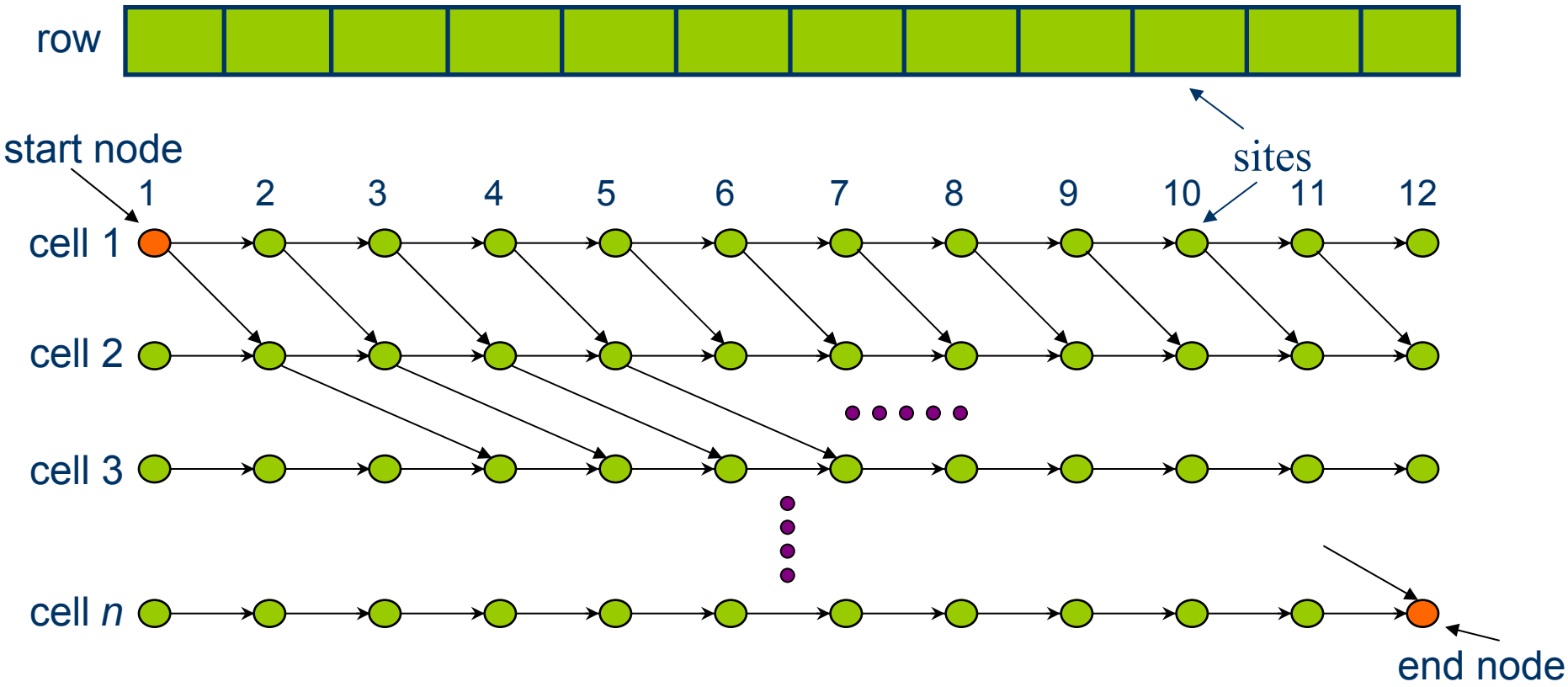
Let's look at a simple case, where we have just one row placement



How to remove this overlap with minimum total movement (perturbation)?

BROWN

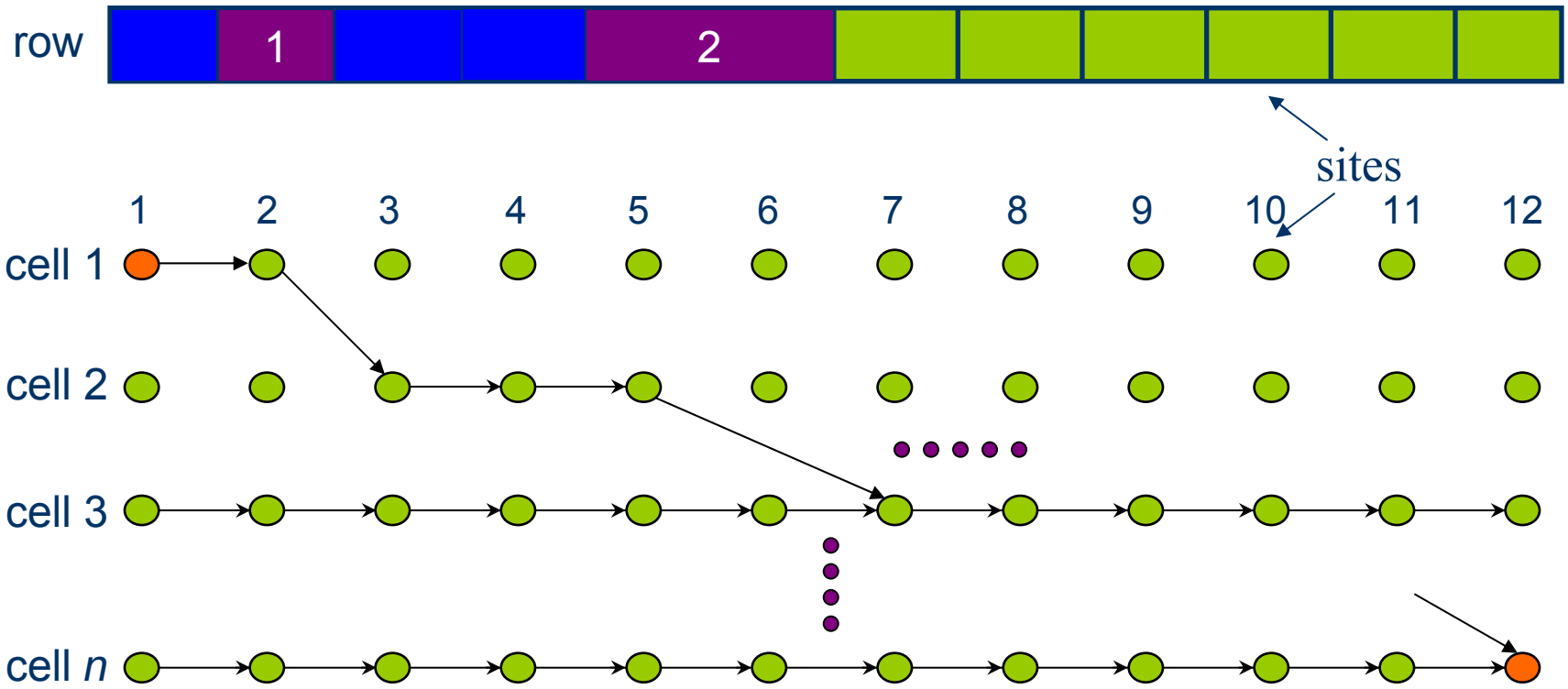
Overlap Removal Using Dynamic Programming



- Each chain represents the possible sites that a cell can be placed at
- The order of chains correspond to the order of cells from left to right in a row

BROWN

Overlap Removal Using Dynamic Programming

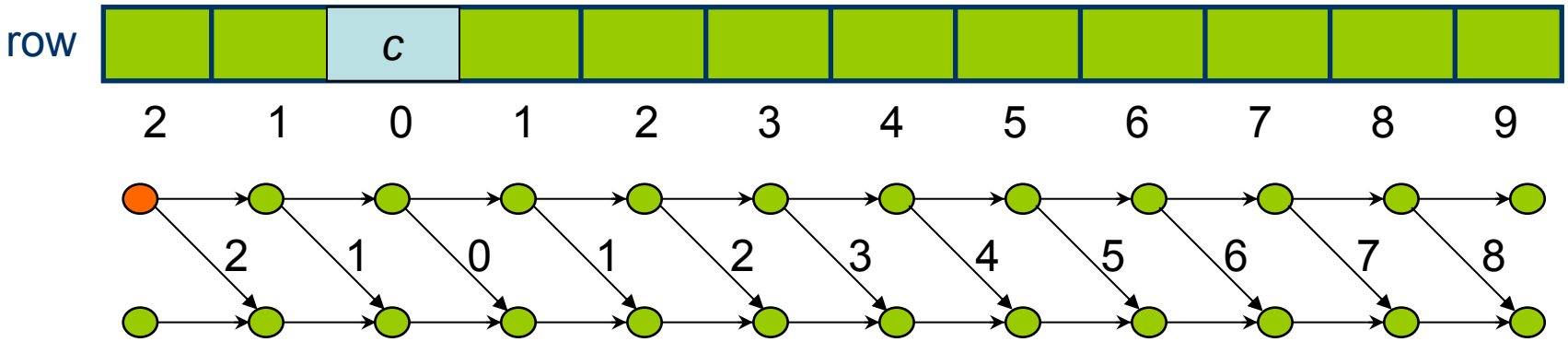


- Start and end sites
- Sites that cell will be placed at
- Empty sites
- Sites not included in calculation

There are many paths from the start and end nodes → select the one that optimizes one of our objectives

BROWN

Min Total Distance Overlap Removal



1. Label a diagonal edge starting at some column *j* and chain *c* by the difference in distance between *j* and current location of cell *c.*

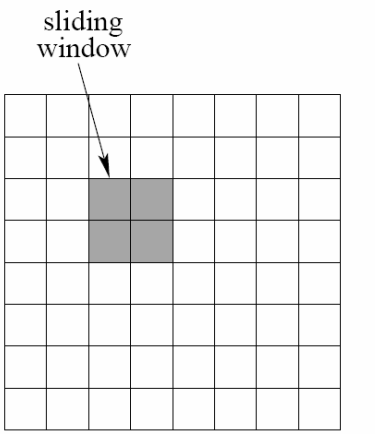
2. Label all horizontal edges by cost 0

3. Find the shortest path from start to end nodes using lexicographical sorting.

BROWN

Detailed placement improves a given legal placement in an incremental fashion

Detailed placement (vs global placement) is of very-small scale combinatorial nature



Simple example of detailed placement:

- Slide a window over the layout area, and for each window location:
  - Find the cells that lie within the window
  - Try all possible layout permutation of the cells
  - Choose the permutation that leads to largest improvement in WL

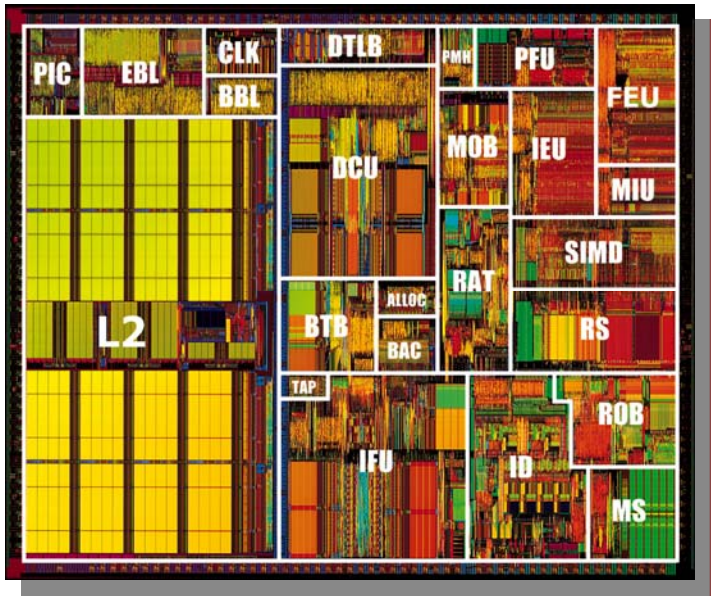
$\Rightarrow$ Can only be feasible for small window size

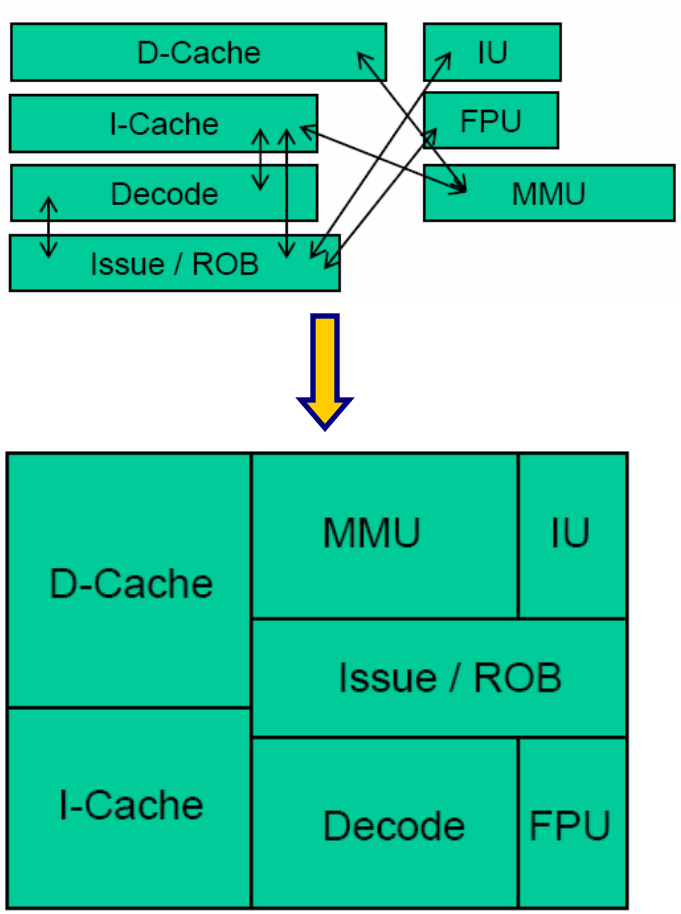
BROWN

Lecture 07: Floorplanning and Placement

- Introduction to Placement and Floorplanning
- Global Placement
- Legalization and Detailed Placement
- Floorplanning

Floorplanning objectives



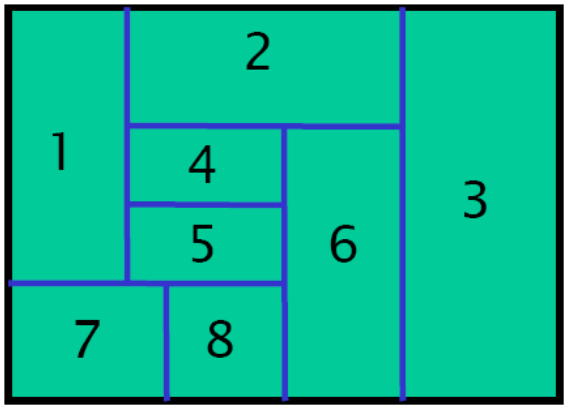


Floorplanning problem: given design modules (with estimates of their areas), find an outline for the chip that minimizes wirelength and area and maximizes performance.

BROWN

Using simulated annealing to solve the floorplanning problem

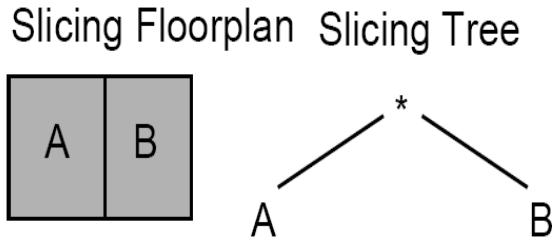
- Floorplanning is a small-scale optimization problem → ideal to solve using simulated annealing
- Simulated annealing is based on swapping moves.
- Key question:
  - How to capture the solution search space in representations (or configurations) that are amenable to swaps?
  - Many representations available. We will only study slicing trees



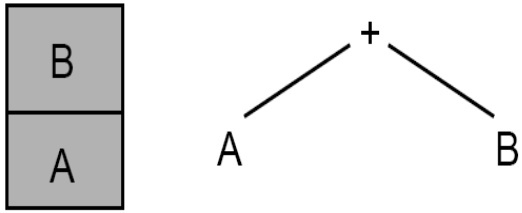
BROWN

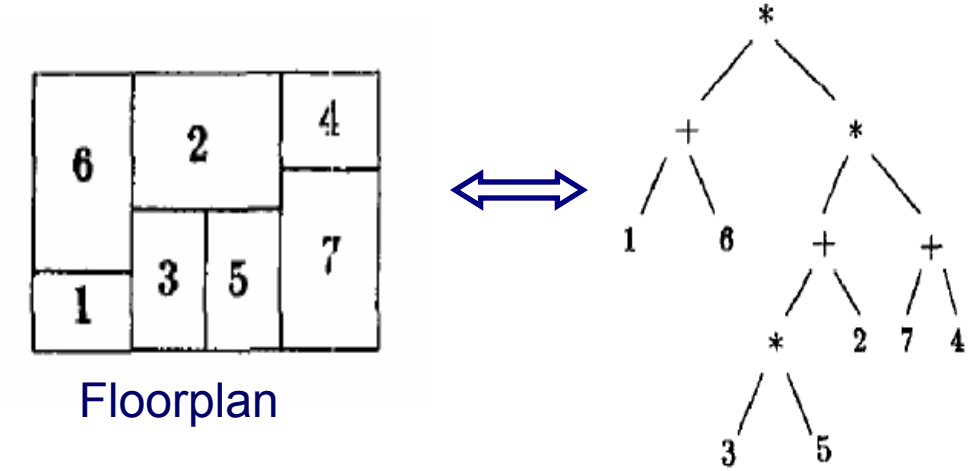
Slicing tree representation for floorplans

Two operators available:
- Slice vertically *
- Slice horizontally +

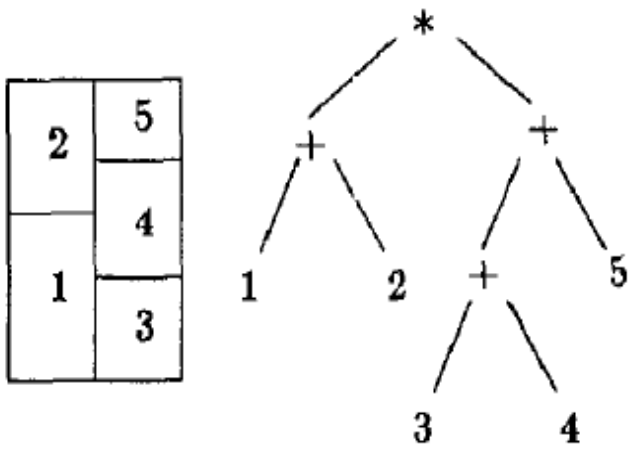


How can we use these two operators to build complex floorplan trees?





A Slicing tree can be written as a string

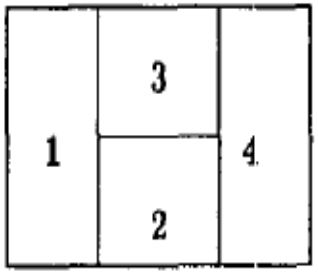


- Slicing tree = (1+2)*((3+4)+5)
- or in *post order traversal*: 12+34+5+*

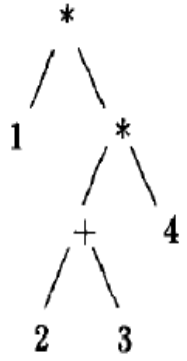
BROWN

Floorplan-Slicing tree interactions

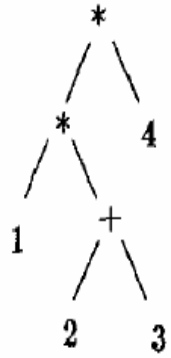
- For a given floorplan, is the slicing tree always unique?



Floorplan

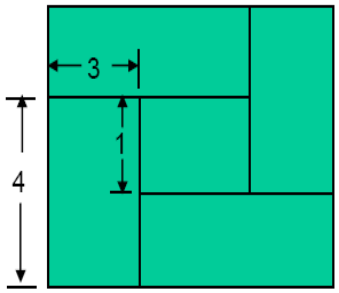


Slicing tree



Another slicing tree

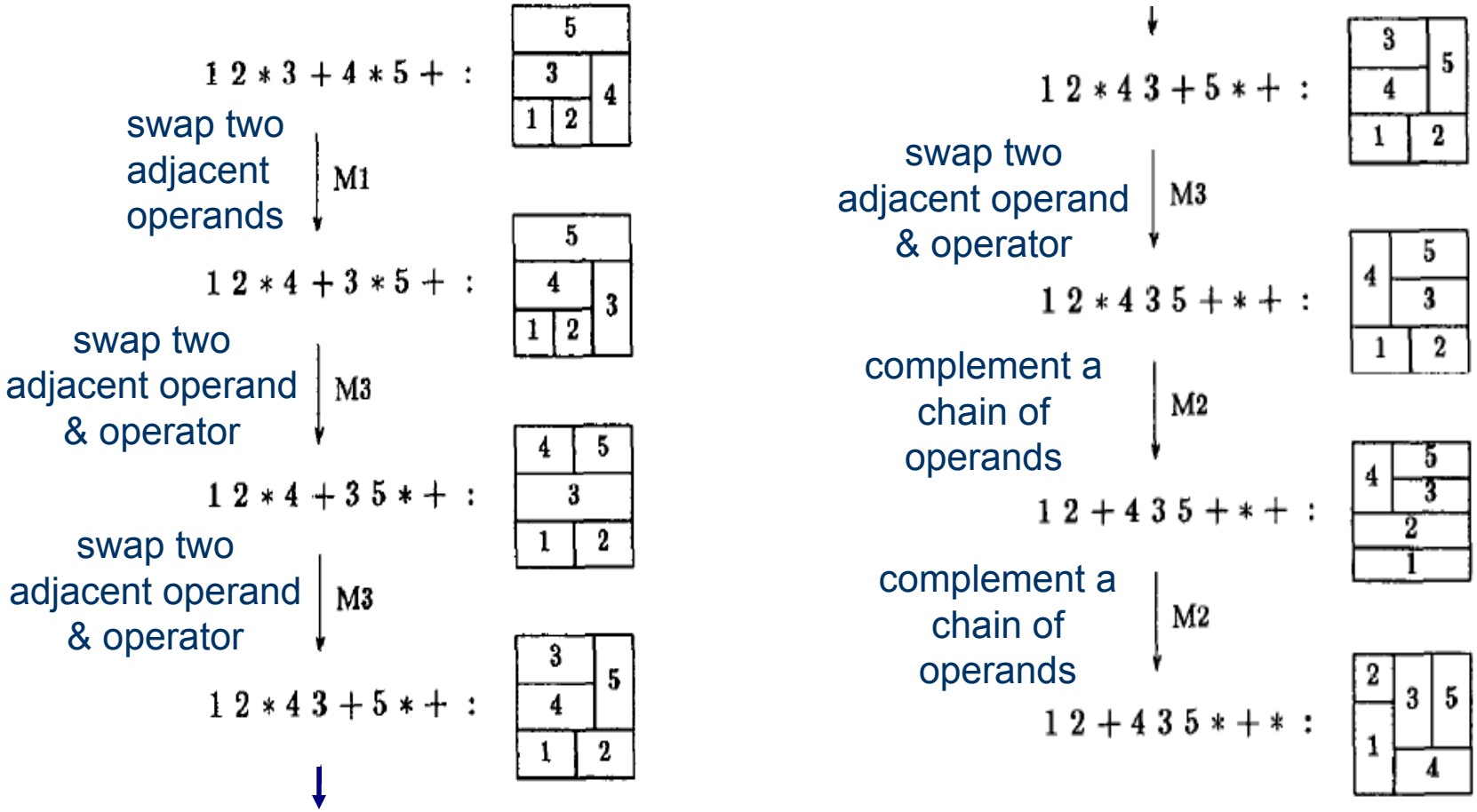
- For a given floorplan, is there always a slicing tree?



- This structure can't be represented by a slicing tree
- Any slicing tree for the above modules must take larger area

BROWN

There are three possible operators that allow to change the floorplan



How to using slicing trees in a SA optimization-based framework?

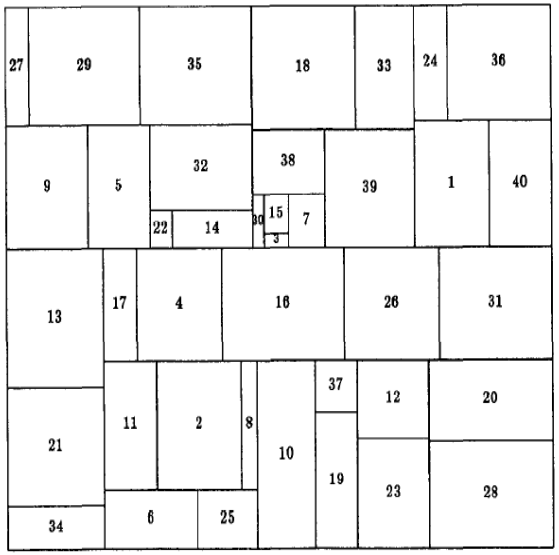
- Generate a random move

  (either: M1: swap two adjacent operands; M2: swap two adjacent operand and operator; or M3: complement a chain)

  – calculate the chance in the floorplan WL (ΔL) due to the move

  – if the move results in reduction ($\Delta L < 0$) then *accept*

  – else reject with probability $1\text{-}e^{-\Delta L/T}$



BROWN

Assignment Readings

- Floorplanning: VLSI Module Placement Based on Rectangle-Packing by the Sequence-Pair
- Placement: Algorithms for Large-Scale Flat Placement

BROWN