CS130B – Data Structures And Algorithms II

Discussion Section Week 10

Final Exam

Tuesday, June 13, 2017

8 AM – 11 AM

Phelps 3515

CUMULATIVE

but focused on material

after midterm

Practice Problem 1

Longest Common Substring Problem

Given two strings X and Y of lengths n and m, respectively, find the longest common substring.

Example:

$$X = philosophy$$

$$Y = philology$$

$$\text{Longest Common Substring } = philo$$

Practice Problem 1

Longest Common Substring Problem

Given two strings X and Y of lengths n and m, respectively, find the longest common substring.

Solution:
Can by solved by dynamic programming

$$
\begin{aligned}
L(i,j) &= \text{length of longest common substring} \\
&\quad \text{of strings } X_1 \ldots X_i \text{ and } Y_1 \ldots Y_j \\
&\quad \text{ending at } X_i \text{ and } Y_j \\
&= \begin{cases} 1 + L(i-1, j-1) & \text{if } X_i = Y_j \\ 0 & \text{otherwise} \end{cases}
\end{aligned}
$$

Practice Problem 1

Longest Common Substring Problem

Example:

$$X = photograph$$

$$Y = tomography$$

Practice Problem 1

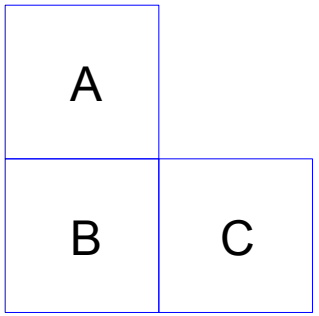
Longest Common Substring Problem

Example:

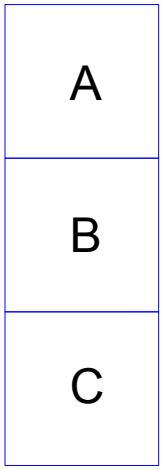
	ϵ	t	o	m	o	g	r	a	p	h	y
ϵ	0	0	0	0	0	0	0	0	0	0	0
p	0	0	0	0	0	0	0	0	1	0	0
h	0	0	0	0	0	0	0	0	0	2	0
o	0	0	1	0	1	0	0	0	0	0	0
t	0	1	0	0	0	0	0	0	0	0	0
o	0	0	2	0	1	0	0	0	0	0	0
g	0	0	0	0	0	2	0	0	0	0	0
h	0	0	0	0	0	0	3	0	0	0	0
a	0	0	0	0	0	0	0	4	0	0	0
p	0	0	0	0	0	0	0	0	5	0	0
h	0	0	0	0	0	0	0	0	0	6	0

Practice Problem 2

Block Configuration Problem



Initial State

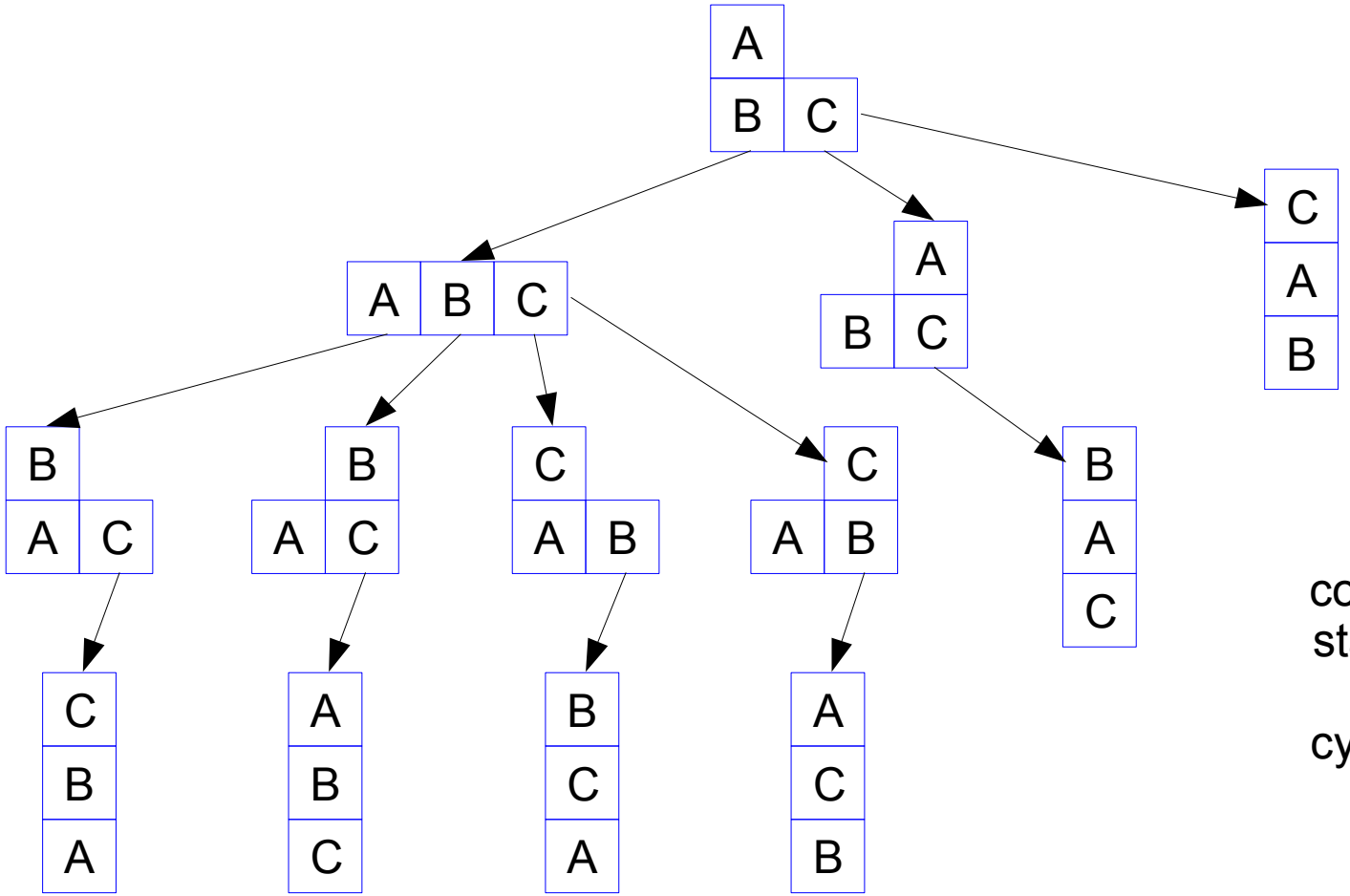


Goal State

Go from initial state to goal state by moving the blocks around, one at a time. Constraint: if a block X is being moved, then there can be no block on top of block X; if moving a block X to top of block Y, there can be no block on top of block Y (can always move a block to a table); lastly, no configuration of blocks can be repeated.

Block Configuration Problem

Example State Space Tree

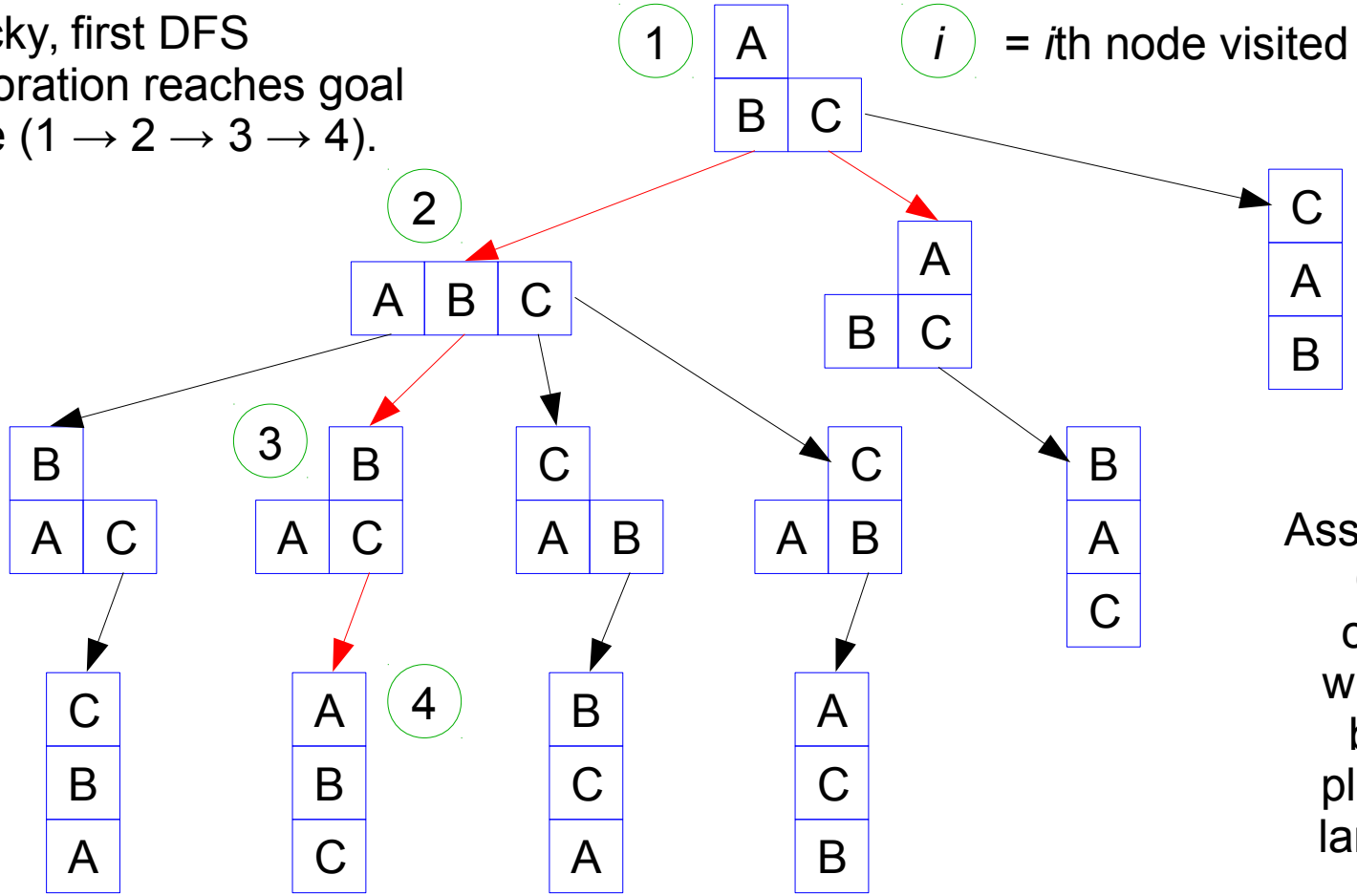


No repeated configurations in state space tree (otherwise a cycle appears in the tree).

Block Configuration Problem

DFS (Implicit Bounding Constraints)

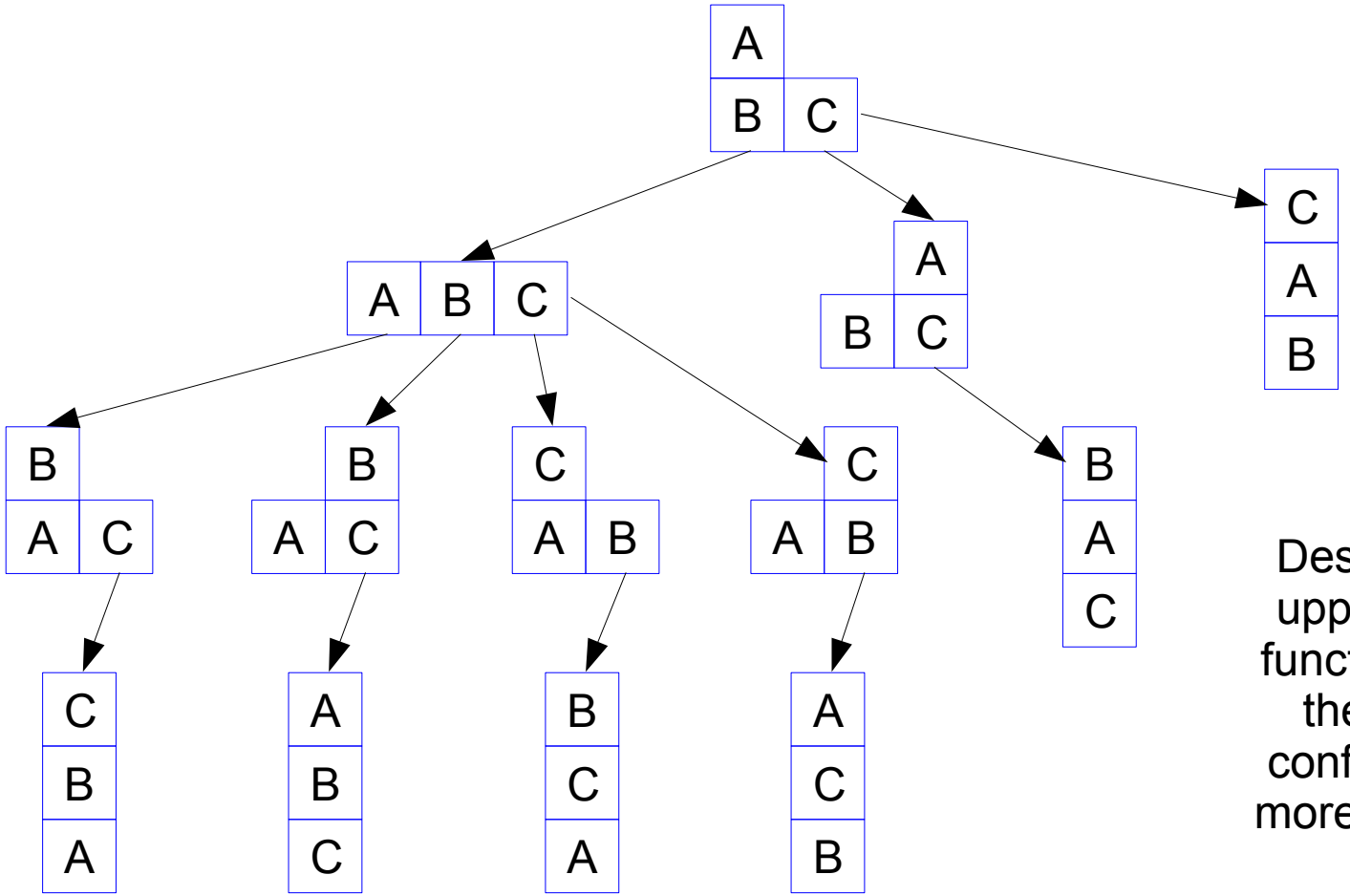
If lucky, first DFS exploration reaches goal state (1 → 2 → 3 → 4).



Constraints:
Assume A < B < C.
Only explore configurations where a smaller block may be placed on top of larger block (red arrows).

Block Configuration Problem

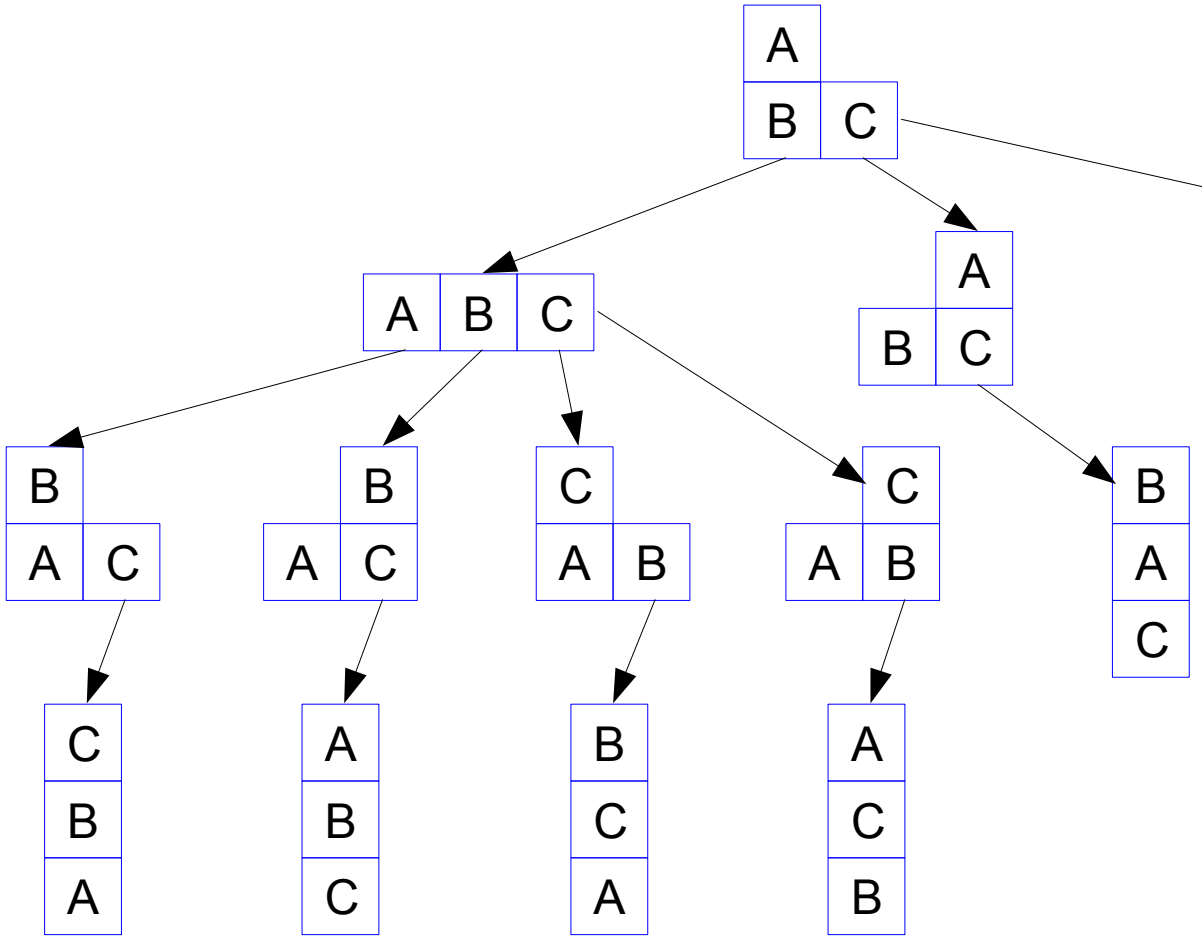
BFS (with bounding)



Design lower and upper bound cost functions based on the moves and configuration (little more troublesome).

Block Configuration Problem

BFS (with bounding)



Cost function construction intuition: penalize more if moving a larger block on top of smaller block; penalize less for moving a block to table and for smaller block on top of larger block.

Block Configuration Problem

BFS (with bounding)

Assume $a > 1$.

$$P(\text{move}) = \begin{cases} a & \text{A top of B or B top of C} \\ a^3 & \text{block to table} \\ a^5 & \text{A top of C} \\ a^7 & \text{B top of A, C top of B} \\ & \text{or C top of A} \end{cases}$$

$$Cost([\text{seq moves}]) = \sum_{m \in \text{seq}} P(m)$$

Block Configuration Problem

BFS (with bounding)

Sequence of moves for upper bound cost:

$$upper([\text{seq}]) = \text{move as many blocks to table as possible w/o repeating configuration, then for each pair of second largest and largest block such that largest block does not have anything on top, stack second largest on top of largest}$$

Block Configuration Problem

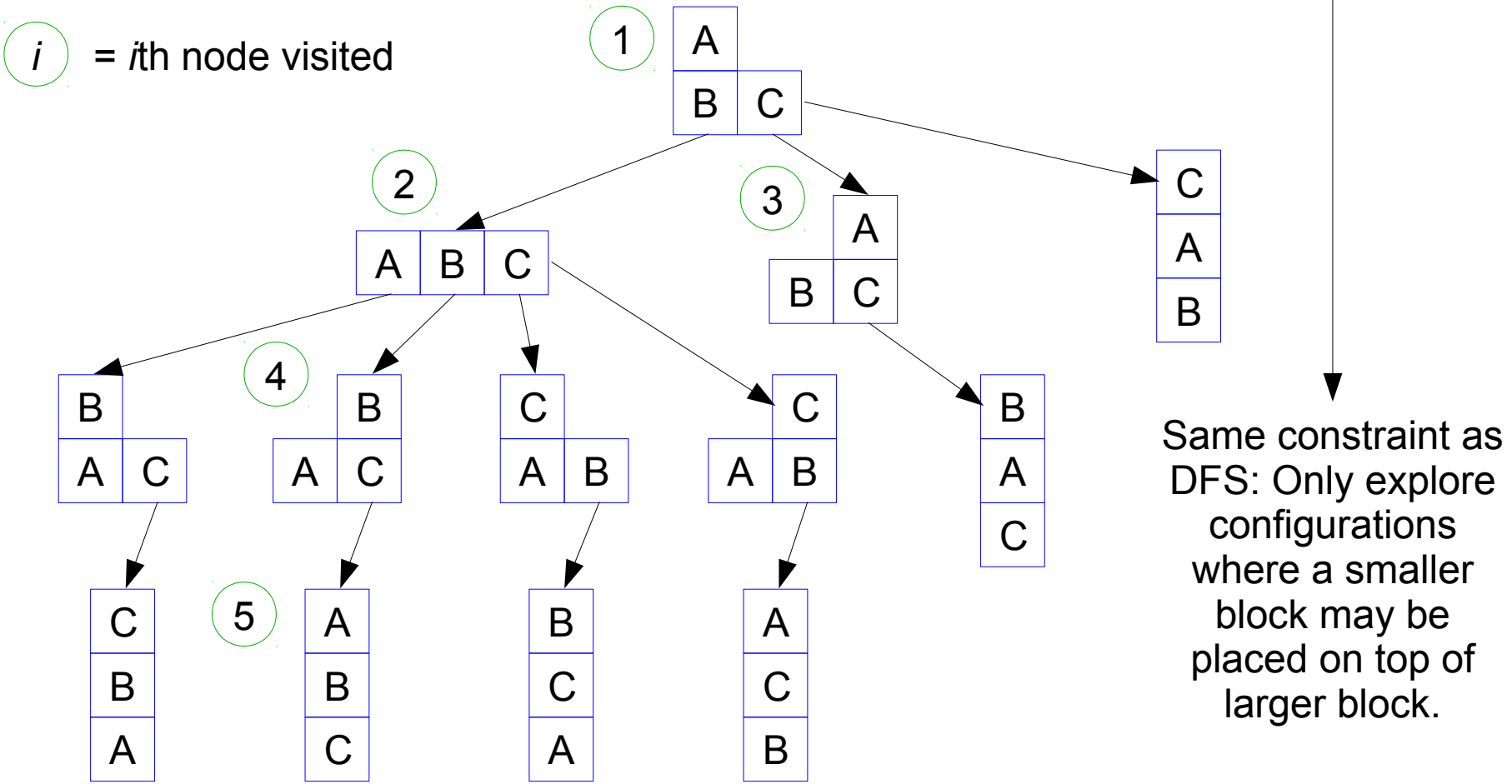
BFS (with bounding)

Sequence of moves for lower bound cost:

$$lower([\text{seq}]) = \text{for each pair of second largest and largest block such that largest block does not have anything on top, stack second largest on top of largest}$$

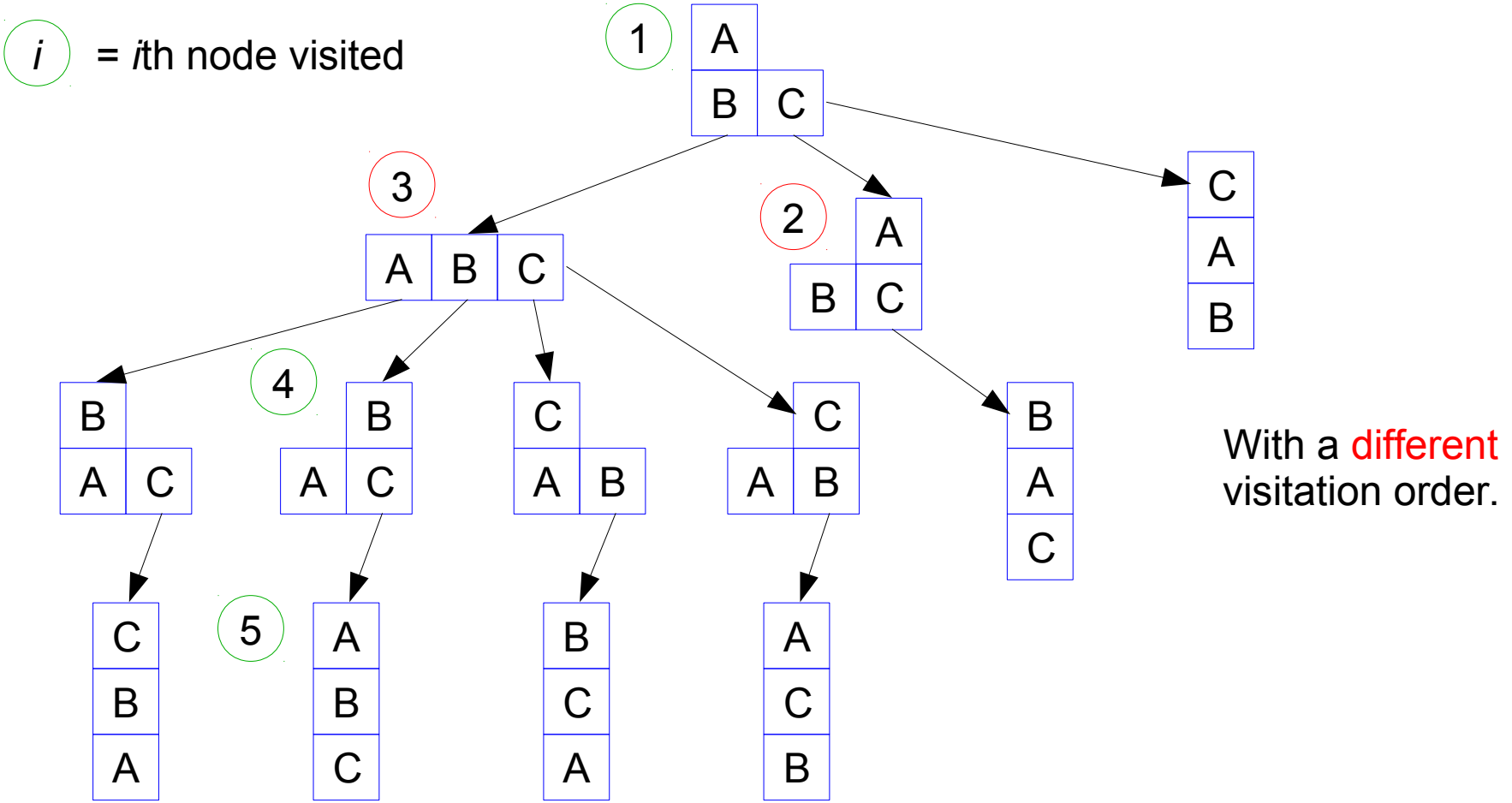
Block Configuration Problem

BFS (with bounding, w/o cost function)



Block Configuration Problem

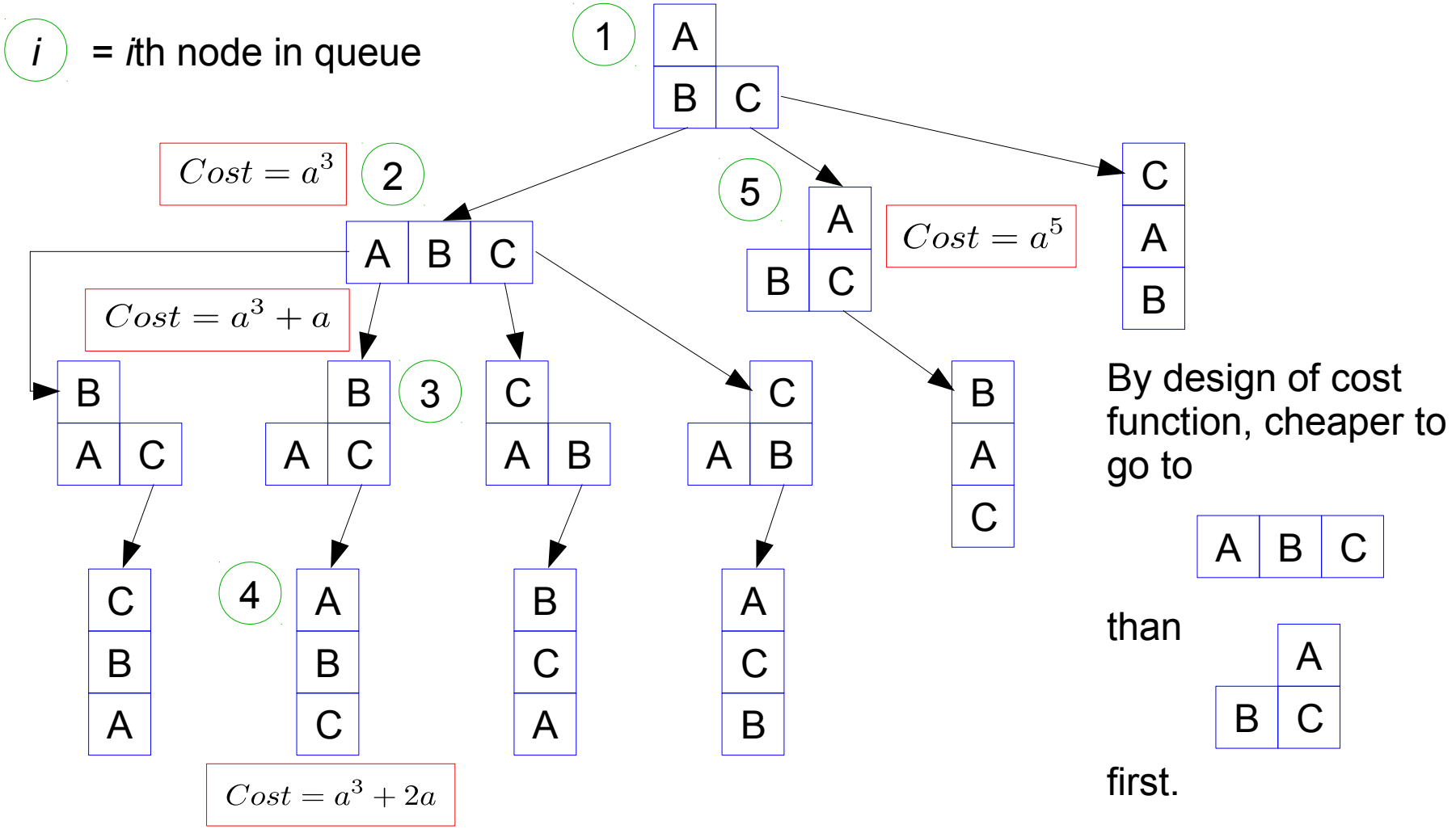
BFS (with bounding, w/o cost function)



With a different visitation order.

Block Configuration Problem

BFS (bounding w/ cost function), aka LC Search



P/NP Problems Review

To be safe, remember the problems that were mentioned in lecture slides:

- 2-SAT (solvable in poly-time), 3-SAT, SAT
- Vertex Cover, Edge Cover (solvable in poly-time)
- Hamiltonian Cycle
- 3D-Matching
- Sudoku
- 3-Coloring of Graph
- Traveling Salesperson
- Independent Set
- K-Clique in a Graph