4.栈与队列

（c3）栈应用：栈混洗

邓俊辉

deng@tsinghua.edu.cn

栈混洗

❖ 考查栈 $A = \langle a_1, a_2, ..., a_n]$、$B = S = \varnothing$ //$\boxed{\text{左}}$端为栈顶

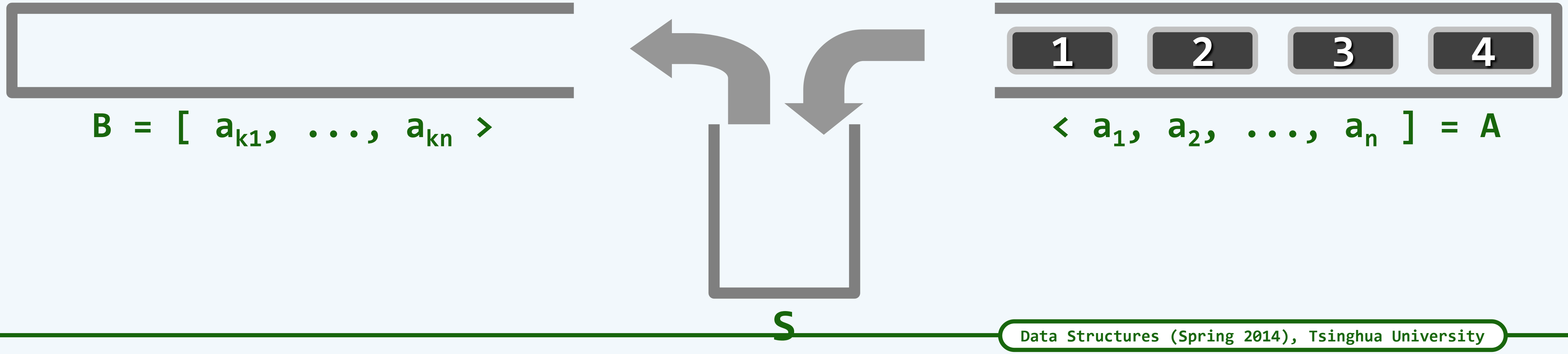
❖ 只允许 将$\boxed{A}$的顶元素弹出并压入$\boxed{S}$，或 //`S.push( A.pop() )`

将$\boxed{S}$的顶元素弹出并压入$\boxed{B}$ //`B.push( S.pop() )`

❖ 若经过一系列以上操作后，A中元素全部转入B中

$B = [a_{k1}, ..., a_{kn} \rangle$ //$\boxed{\text{右}}$端为栈顶

则称之为A的一个栈混洗（stack permutation）

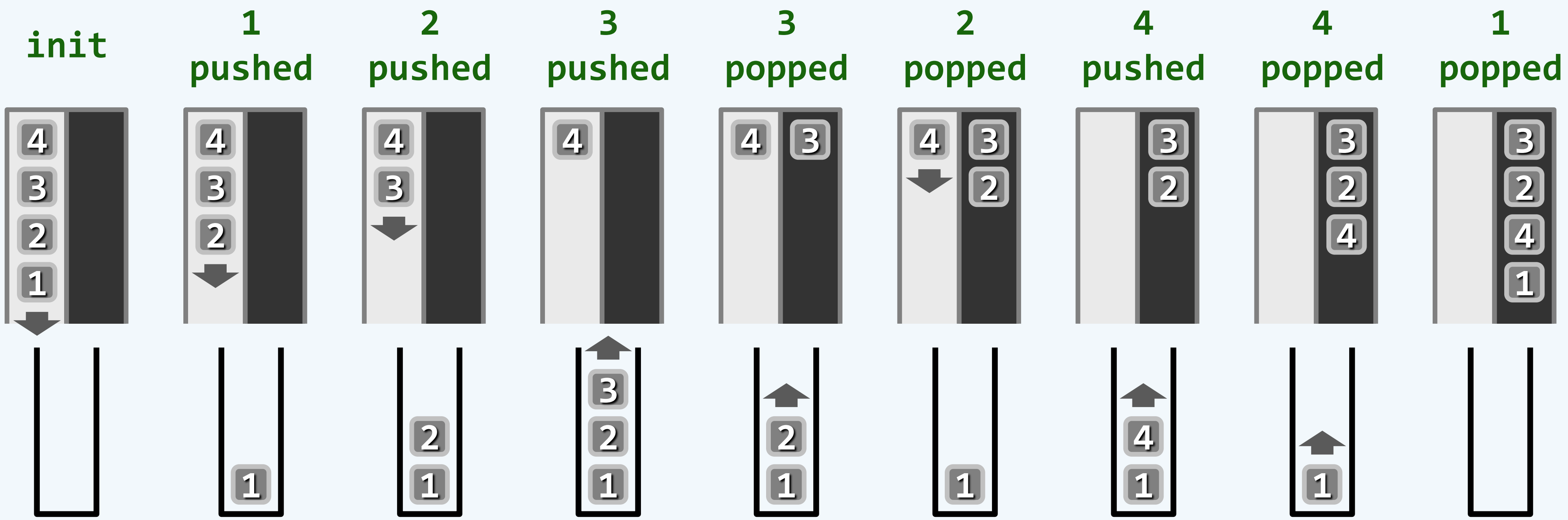


计数

❖ 同一输入序列，可有多种栈混洗

[1, 2, 3, 4 >，[4, 3, 2, 1 >，[3, 2, 4, 1 >，...

❖ 长度为n的序列，可能的混洗总数SP(n) = ?　　//显然，SP(n) <= n!



计数

❖ SP(1) = 1

❖ 设栈S在第 $\boxed{k}$ 次pop()之后重新变空，则 $\boxed{k}$ 无非 $\boxed{n}$ 种情况：

SP(n) = $\boxed{\sum_{k=1}^{n} SP(k-1) \cdot SP(n-k)}$

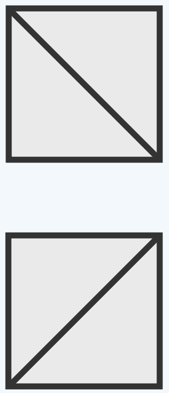
= Catalan(n) = (2n)! / (n+1)! / n!

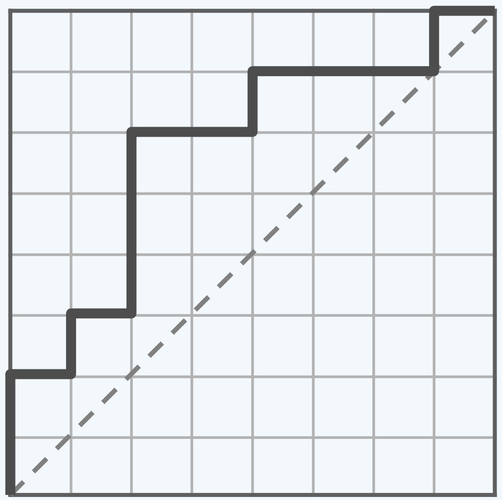
❖ SP(2) = 4! / 3! / 2! = 2

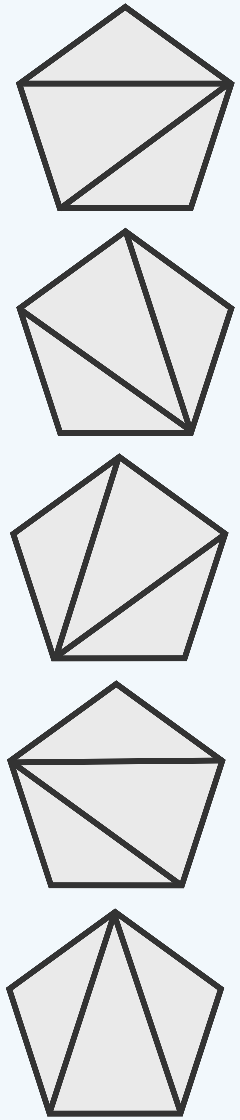
SP(3) = 6! / 4! / 3! = 5

... ...

SP(6) = 12! / 7! / 6! = 132







甄别

❖ 输入序列< 1, 2, 3, ..., n]的

任一排列[p_1, p_2, p_3, ..., p_n >

是否为栈混洗？

❖ 简单情况：< 1, 2, 3]，n = 3

栈混洗共 6! / 4! / 3! = 5 种

全排列共 3! = 6 种 //少了一种...

❖ [3, 1, 2 > //为什么是它？

❖ 观察：任意三个元素能否按某相对次序出现于混洗中，与其它元素无关 //故可推而广之...

❖ 对于任何1 ≤ i < j < k ≤ n，[..., $\boxed{k}$, ..., $\boxed{i}$, ..., $\boxed{j}$, ... > 必非栈混洗

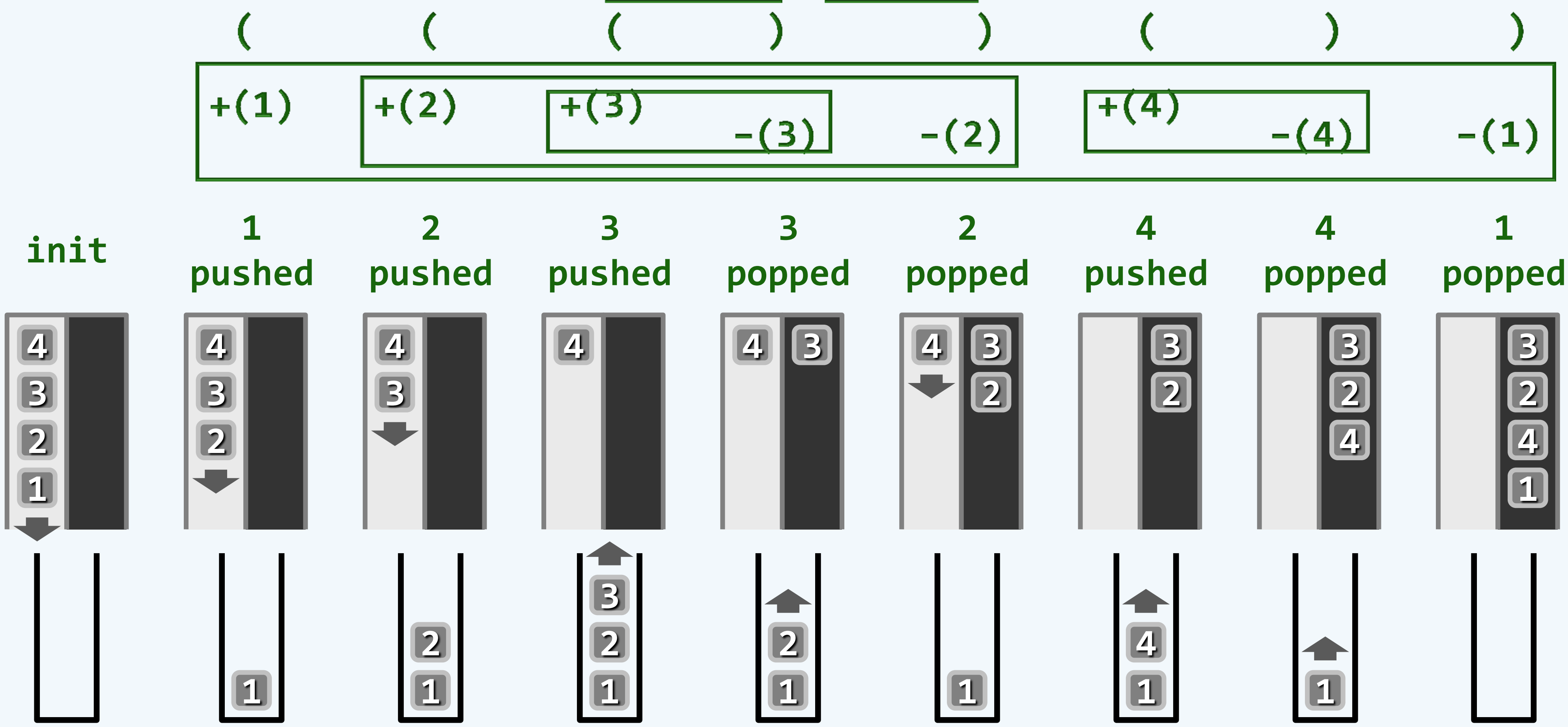
❖ 反过来，不存在 "$\boxed{312}$"模式的序列，一定是栈混洗吗？

甄别

- 充要性：　A permutation is a stack permutation $\boxed{\text{iff}}$
  （Knuth, 1968）　it does NOT involve the permutation $\boxed{312}$　　//习题[4-3]
- 如此，可得一个$\boxed{O(n^3)}$的甄别算法　　//进一步地...
- [p_1, p_2, p_3, ..., p_n >是< 1, 2, 3, ..., n]的栈混洗，$\boxed{\boxed{当}且仅当}$
  对于任意i < j，不含模式[..., $\boxed{j + 1}$, ..., $\boxed{i}$, ..., $\boxed{j}$, ... >
- 如此，可得一个$\boxed{O(n^2)}$的甄别算法　　//再进一步地...
- $\boxed{O(n)}$算法：直接借助栈A、B和S，模拟混洗过程　　//为何可行？
  每次S.pop()之前，检测S是否已空；或需弹出的元素在S中，却非顶元素

括号匹配

❖ 观察：每一栈混洗，都对应于栈S的n次push与n次pop操作构成的序列



❖ n个元素的栈混洗，等价于n对括号的匹配