COL106 A-2 BONUS ATTEMPT

Amortized analysis is a method in which we calculate the average cost of operations over a sequence of operations, instead of the individual cost of each operation.

Claim: The average cost (amortized cost) of push using my stack implementation (due to hysteresis) is O(1).

Proof:

I have implemented a stack that doubles its capacity when it becomes full and halves its capacity when it is filled only to 1/4 of its capacity. This implementation is using **Hysteresis,** to prevent frequent resizing which balances the overall operation costs.

Push Operation:

The push operation is **O(1)** in an average case because it involves a single insertion and, in the case of resizing, copying of existing elements. The **worst-case** cost of a push operation is **O(n).**

Pop Operation:

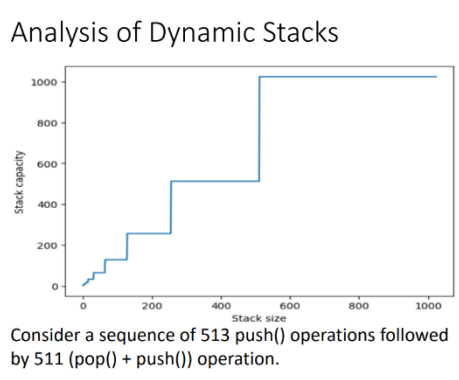
Without hysteresis, the stack would resize down immediately when the number of elements becomes less than half the capacity. Pop operations can lead to frequent resizing, which results in an average cost proportional to the cost of resizing **O(n),** where n is the current size of the stack.

Analysis of Dynamic Stack:

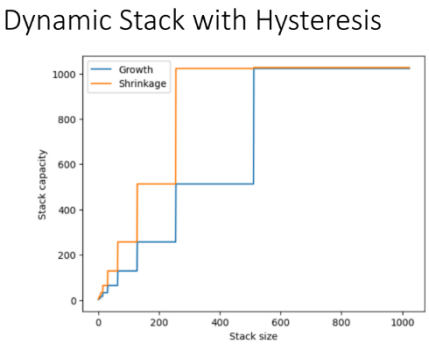
Assume N (= 2^k) push followed by N pop()

T(2N) = Time for N push + Time for N pop = O(N) + O(N) = O(N)

T(N) = O(N) (worst case scenario)



Consider a sequence of 513 push() operations followed by 511 (pop() + push()) operation.



With Hysteresis: Using hysteresis, we set the **threshold** for resizing as **size=capacity/4** for resizing down. This means that the stack will resize down to **capacity/2** only when the number of elements becomes less than capacity/4.

Consider a series of pop operations that frequently reduce the stack size to capacity/2. With hysteresis, the stack will not resize down immediately; instead, it waits until the stack size reaches threshold(capacity/4) before resizing down. This causes a delay in resizing down, maintaining the cost of operation to be O(1) only.

Amortized Analysis: In the worst case, a pop operation costs **O(1)** and resizing costs **O(n).** But with hysteresis, the resizing cost is delayed and distributed over a sequence of pop operations , leading to an amortized cost of **O(1)** per pop operation.

Explanation: Although resizing and copying the array is an expensive operation, the **number of times resizing takes place is much smaller than the total number of pop operations**. **While individual insertions may be costly occasionally, the average cost over the entire sequence is much lower.**

Using Amortized analysis, we can show the **average cost per insertion is not as high as the worst-case scenario might suggest**. In this case, the amortized cost of an pop and push is O(1), even though individual insertions may sometimes take O(n) due to array resizing.