Problems: Amortized Analysis

Abhranil Chatterjee and Palash Dey
Indian Institute of Technology, Kharagpur

July 31, 2026

1. [CLRS book] Recall the dynamic table data structure discussed in the class. Suppose that instead of contracting a table by halving its size when its load factor drops below $1/4$, we contract it by multiplying its size by $2/3$ when its load factor drops below $1/3$. Show that insert and delete operations take $\mathcal{O}(1)$ amortized time.

2. [CLRS book] Design a data structure to support the following two operations for a dynamic multiset S of integers, which allows duplicate values:

   INSERT(S,x) inserts x into S.

   DELETE-LARGER-HALF(S) deletes the largest $\lceil |S|/2 \rceil$ elements from S.

   Explain how to implement this data structure so that the amortized time complexity of INSERT and DELETE-LARGER-HALF is $\mathcal{O}(1)$.

3. [CLRS book] Consider an ordinary binary min-heap data structure with n elements supporting the instructions INSERT and EXTRACT-MIN in $\mathcal{O}(\log n)$ worst-case time. Show that the amortized cost of INSERT is $\mathcal{O}(\log n)$ and EXTRACT-MIN in $\mathcal{O}(1)$.

4. [CLRS book] What is the total cost of executing n of the stack operations PUSH, POP, and MULTIPOP, assuming that the stack begins with s_0 objects and finishes with s_n objects?

5. [CLRS book] Show that if a DECREMENT operation is included in the k-bit counter example, n operations can cost as much as $\Theta(nk)$ time.

6. [CLRS book] If the set of stack operations includes a MULTIPUSH operation, which pushes k items onto the stack, does the $\mathcal{O}(1)$ bound on the amortized cost of stack operations continue to hold?

7. [CLRS book] You perform a sequence of PUSH and POP operations on a stack whose size never exceeds k. After every k operations, a copy of the entire stack is made automatically, for backup purposes. Show that the cost of n stack operations, including copying the stack, is $\mathcal{O}(n)$ by assigning suitable amortized costs to the various stack operations.

8. [CLRS book] You wish not only to increment a counter but also to reset it to 0 (i.e., make all bits in it 0). Counting the time to examine or modify a bit as $\Theta(1)$, show how to implement a counter as an array of bits so that any sequence of n INCREMENT and RESET operations takes $\mathcal{O}(1)$ time on an initially zero counter.

9. [Queue Using Two Stacks] A queue is implemented using two stacks:

   - ▷ Stack S_1 stores newly enqueued elements.
   - ▷ Stack S_2 is used for deletions.
   - ▷ If S_2 is empty during dequeue, all elements of S_1 are moved to S_2.

   Assume every push or pop costs one unit.

   (a) What is the worst-case cost of one dequeue? Answer in Θ notation and justify your answer.

   (b) Prove that both enqueue and dequeue have an amortized cost of $\mathcal{O}(1)$.

10. [Dynamic Table] Consider the dynamic table data structure discussed in the class. Suppose we double the table size when the table becomes $(3/4)$-th full and halve the table when the table becomes $(1/4)$-th full. Prove that the amortized time complexity of both insert and delete is $\mathcal{O}(1)$.