CS 430 Lecture 20 Activities

Opening Questions

1. How do you think the allocated size growth of a dynamic array like Java's ArrayList is implemented? How much bigger does it grow when needed? What is the runtime for a sequence of n insertions starting from a default size of 10 considering the worst individual insert?

Amortized (to pay off gradually) Analysis

So far, we have analyzed best and wort case running times for an operation without considering its context. With amortized analysis, we study a sequence of operations rather than individual operations. An amortized analysis is any strategy for analyzing a sequence of operations to show that the average cost per operation is small, even though a single operation within the sequence might be expensive.

Aggregate Method of Amortized Analysis

1. Can we do a better analysis by amortizing the cost over all inserts? Starting with a table size one and doubling the size when necessary, make a table shoring the first 10 inserts and determine a formula for $\text{COST}(i)$ for the cost of the ith insert. Then aggregate "add up" all the costs and divide by n (aggregate analysis).

Accounting Method of Amortized Analysis

Figure out a specific amortized cost to be allocated to each operation to ensure you have enough "balance" to handle the bad operations.

Charge ith operation a fictitious amortized cost $\hat{c}_i$, where \$1 pays for 1 unit of work (i.e., time).

- This fee is consumed to perform the operation.
- Any amount not immediately consumed is stored in the bank for use by subsequence operations.
- The bank balance must not go negative! We must ensure that for all n

$$\sum_{i=1}^{n} c_i \leq \sum_{i=1}^{n} \hat{c}_i$$

Thus, the amortized costs provide an upper bound on the total true costs.

2. For the previous ArrayList example, determine the amortized cost $\hat{c}_i$ necessary.

3. Consider, as a second example, a binary counter that is being implemented in hardware. Assume that the machine on which it is being run can flip a bit as its basic operation. We now want to analyze the cost of counting up from 0 to n (using k bits).

   What is the naive worst-case analysis for how many bits we need to flip?

Decimal	Binary
1	000001
2	000010
3	000011
4	000100
5	000101
...	...
n	100110

4. Use the aggregate method to perform a more careful analysis for n increments of a binary counter.

5. Use the accounting method to perform a more careful analysis for n increments of a binary counter.