Algorithms 2018/2019 (CC4010) – November 6th, 2018

Test #1 - Auxiliary Material

Loop Invariants and Correctness

To prove the correctness of a loop, find a suitable loop **invariant condition** and then show the following things:

- **Initialization:** It is true prior to the first iteration of the loop.
- **Maintenance:** If it is true before an iteration of the loop, it remains true before the next iteration.
- **Termination:** When the loop terminates, the invariant gives us a useful property that helps show that the algorithm is correct.

We also need to show that the loop terminates:

- **Progress:** Each iteration gets us closer to the end until eventually we finish

Asymptotic Notation

- $\mathbf{f(n) = O(g(n))}$ if there are positive constants n_0 and c such that $f(n) \leq cg(n)$ for all $n \geq n_0$.
- $\mathbf{f(n) = \Omega(g(n))}$ if there are positive constants n_0 and c such that $f(n) \geq cg(n)$ for all $n \geq n_0$.
- $\mathbf{f(n) = \Theta(g(n))}$ if there are positive constants n_0, c_1 and c_2 such that $c_1 g(n) \leq f(n) \leq c_2 g(n)$ for all $n \geq n_0$.
- $\mathbf{f(n) = o(g(n))}$ if for any positive constant c there exists n_0 such that $f(n) < cg(n)$ for all $n \geq n_0$.
- $\mathbf{f(n)} = \omega(\mathbf{g(n)})$ if for any positive constant c there exists n_0 such that $f(n) > cg(n)$ for all $n \geq n_0$.

Solving Recurrences

- **Unrolling:** unroll the recurrence to obtain an expression (ex: summation) you can work with
- **Substitution:** guess the answer and prove by induction
- **Recursion Tree:** draw a tree representing the recursion and sum all the work done in the nodes
- **Master Theorem:** If the recurrence is of the form $\mathbf{aT(n/b) + cn^k}$ (*this is one version of the theorem*):

(1) $T(n) = \Theta(n^k)$ if $a < b^k$
(2) $T(n) = \Theta(n^k \log n)$ if $a = b^k$
(3) $T(n) = \Theta(n^{\log_b a})$ if $a > b^k$

Amortized Analysis

- **Aggregate** method: examine/bound total cost and calculate the average
- **Accounting** method: impose extra charge on inexpensive operations, saving for future expensive operations
- **Potential** method: define a potential function on the data structure state and use it to bound the cost

Probabilistic Analysis

- **Expectation**: For a discrete random variable X over sample space S, $\mathbf{E}[X] = \sum_{e \in S} Pr(e)X(e)$
- **Linearity of Expectation**: For any two random variables X and Y: $\mathbf{E}[X+Y] = \mathbf{E}[X] + \mathbf{E}[Y]$
- **Indicator Random Variable**: The indicator random variable $\mathbf{I\{A\}}$ associated with event A is defined as: $I\{A\} = 1$ if A occurs, 0 if A does not occur.
- **Las Vegas algorithm:** always outputs the correct answer, but runtime is a random variable.
- **Monte Carlo algorithm:** always terminates in given time bound, and outputs the correct answer with at least some (high) probability.

Lower Bounds

- **Information Theory**: answers to the queries must give enough information to specify any possible output
- **Adversarial Strategy:** answering the queries with the goal of delaying as much as possible the final answer