Practice Set: Decrease & Conquer vs Divide & Conquer

October 15, 2025

Q1. Insertion Sort trace and analysis (decrease by 1).

(a) Trace insertion sort on: $A = [6, 2, 9, 1, 5, 3]$. Show the array after each insertion step.

(b) Prove the loop invariant for the inner while-loop and use it to argue correctness.

(c) Give tight $\Theta(\cdot)$ bounds for best, average, and worst cases; state an input family achieving each bound.

Q2. Binary Search vs Interpolation Search (decrease by factor vs variable-size).

(a) For a sorted array $A[0 \mathinner{..} n-1]$ whose keys are uniformly distributed over $[0, 1]$, explain why interpolation search has expected $O(\log \log n)$ probes (high-level argument is fine).

(b) Construct a pathological input where interpolation search degrades to $\Theta(n)$, while binary search remains $\Theta(\log n)$.

Q3. Fake coin on a balance scale (decrease by constant factor).

(a) Design the 2-pile algorithm (split into $\lfloor n/2 \rfloor$ and $\lceil n/2 \rceil$) and derive the recurrence for the number of weighings $W_2(n)$; solve it.

(b) Design the 3-pile algorithm and derive $W_3(n)$; compare asymptotically to $W_2(n)$.

(c) For $n = 26$, give an explicit weighing plan using the 3-pile method.

Q4. Euclid's GCD (variable-size decrease).

(a) Prove correctness of the recurrence $\gcd(m, n) = \gcd(n, m \bmod n)$ using divisibility properties.

(b) Show the running time is $O(\log \min\{m, n\})$ by relating the remainder sequence to Fibonacci numbers.

Q5. Quickselect (k-th order statistic).

(a) Write pseudocode for randomized Quickselect. State its base cases.

(b) Give the recurrence for expected time and solve to $O(n)$.

(c) Run your algorithm by hand to find the median of $[12, 7, 3, 9, 14, 1, 10]$; show one possible pivot sequence and the subarray considered at each step.

Q6. Generating the powerset (decrease by 1) and Gray code.

(a) Give a recursive algorithm that generates $\mathcal{P}(S)$ for $S = \{a, b, c, d\}$ by adding one element at a time. List the first 10 subsets produced.

(b) Describe how to enumerate the powerset via Binary Reflected Gray Code so consecutive subsets differ by exactly one element; prove the one-bit flip property.

Q7. Johnson–Trotter permutations (decrease by 1 with minimal change).

(a) Define "mobile element." Generate the first six permutations of $\{1, 2, 3, 4\}$ using Johnson–Trotter, showing arrow directions after each step.

(b) Give the time complexity to output all permutations and argue optimality w.r.t. output size.

Q8. Merge Sort vs Quick Sort (divide-and-conquer).

(a) Derive the Merge Sort recurrence and solve it. State space usage and stability.

(b) For Quick Sort with "median-of-three" pivot and a cutoff to insertion sort for subarrays of size $\leq t$, give pseudocode and explain why this hybrid often wins in practice.

(c) On a RAM model with constant-time comparisons/moves, argue how to choose a practical t (no experiments needed—reason about constants and branch overhead).

Q9. Closest pair of points in the plane (divide-and-conquer).

(a) Outline the split–recur–merge ("straddle") algorithm and state its overall time bound.

(b) Prove the packing lemma that in the strip you only need to check a constant number (e.g. at most 5–7) of subsequent points per point when sorted by y.

(c) Apply the algorithm to the set $\{(0, 0), (1, 2), (2, 1), (3, 3), (4, 0), (5, 2)\}$ and show the comparisons in the strip.

Q10. Strassen's Matrix Multiplication (divide-and-conquer with algebraic trick).

(a) Write the recurrence for the number of multiplications and additions; solve the multiplication count to $\Theta(n^{\log_2 7})$.

(b) For $n = 8$, compute (i) the number of scalar multiplications in classical $\Theta(n^3)$ multiplication and (ii) in Strassen (assume exact powers of two and ignore padding).

(c) Discuss when Strassen becomes beneficial in practice; list two implementation concerns that affect the crossover point.