

# Algorithms Crash Review

Laura Toma, csci2200, Bowdoin College

Here is a list of topics studied in algorithms this semester:

1. Worst case, best case running times.
  - e.g. Binary search runs in logarithmic time worst case and  $O(1)$  best case.
2. Asymptotic growth of functions ( $O, \Omega, \Theta$ ). Used to express efficiency (running time, space).
  - This algorithm runs in  $O(n^2)$  time and uses  $\Theta(n)$  space.
  - This algorithm runs in  $\Theta(n \lg n)$  worst case.
3. Recurrences. Used to analyze complexity of recursive algorithms.
  - Example: mergesort recurrence:  $T(n) = 2T(n/2) + \Theta(n)$  solves to  $\Theta(n \lg n)$ .
4. Comparison-based sorting
  - Quadratic sorts: insertion sort, bubble sort, selection sort
  - Mergesort, Heapsort, Quicksort, Randomized quicksort
  - Comparison-based sorting lower bound: Any sorting algorithm that uses only comparisons to order the elements must take  $\Omega(n \lg n)$  in the worst case.
5. Sorting without comparisons
  - Not a general purpose sort; makes additional assumptions, usually that the keys to be sorted are integers in a small range.
  - Example: sort  $n$  keys that are all integers in the range  $\{-10, \dots, 10\}$ .
  - Counting sort, Radix sort, Bucket sort
  - Runs in  $O(n + k)$  where  $k$  is the range
6. Selection: given a set of  $n$  elements, find the  $i$ th smallest.
  - Idea: partition and recurse. Quick-Select in expected  $O(n)$  time ;  $O(n)$  Smart-Select in worst-case  $O(n)$  time. Theoretically better but in practice Quick-Select is preferred.
7. The binary heap
  - Implements the priority queue ADT, and supports FindMin (in  $O(1)$  time), DeleteMin, Insert, ChangeKey all in  $O(\lg n)$  time
  - A max-heap is defined symmetrically

8. Binary search trees (BST) and red-black trees

- A BST supports insert, delete, search, min, max, pred, succ, all in  $O(h)$  time; tree walks (in-order, post-order, pre-order) in  $O(n)$  time.
- A red-black tree is a BST plus additional invariants that ensure that  $h = \Theta(\lg n)$ .
- On red-black-trees all above operations run in  $O(\lg n)$  time which includes the additional time to maintain  $h = \Theta(\lg n)$  via rotations

9. Technique: Divide-and-conquer

- Technique that solves a problem by decomposing it into smaller subproblems and solving each one recursively, and then “merging” their solutions
- E.g. mergesort, maximum partial sum, counting inversions.
- Also: binary search, finding missing number

10. Technique Dynamic programming

- Technique used for solving optimization problems that have optimal substructure and overlapping subproblems. Store (cache) solutions to subproblems in a table to avoid recomputation.
- Examples: rod cutting, house robber, 0-1 knapsack, longest common subsequence, weighted interval scheduling, subset sum, weighted subset sum (0-1 knapsack), longest common subsequence; skis and skiers, also shortest paths (Bellman Ford)

11. Technique: Greedy algorithms

- Instead of dynamic programming (which does an exhaustive search), make a choice “locally” and recurse on what’s left. The choice is determined by examining only local information, without going into recursion to see how good the choice is “globally”.
- Examples: fractional knapsack, activity selection, art gallery guarding. Also Dijkstra SSSP, Prim and Kruskal’s algorithm for MST are all greedy.
- Greedy heuristics are used a lot in AI but not so much in algorithms (they rarely guarantee the optimal solution)

12. Graph algorithms

- Graph representation (adjacency list, adjacency matrix)
- Graph traversal: BFS and DFS
  - $G$  can be directed or undirected
  - run in linear time  $O(V + E)$
  - BFS used to: find connected components, reachability, check bipartiteness ( $G$  undirected), compute shortest paths ( $G$  un-weighted, all edges have weight 1)
  - DFS used to: find connected components, reachability, find directed cycles, topological sort ( $G$  DAG)
- Directed acyclic graphs (DAGs)

- Topological order: Order of the vertices such that for any edge  $(x, y)$ , vertex  $x$  comes before vertex  $y$  in topological order (all edges are “forward”)
- Any DAG can be topologically ordered in linear  $O(|V| + |E|)$  time either directly or via DFS
- Many problems have easier/faster solutions on DAGs. E.g. SSSP on DAGs can be computed in  $O(|V| + |E|)$  time; Longest paths on DAGs in  $O(|V| + |E|)$  (note: on general graphs longest path is an NPC problem).
- Shortest paths (SSSP): Find shortest paths from a vertex to all other vertices
  - On DAGs:  $O(V + E)$  time
  - On general graphs with non-negative weights: Dijkstra’s runs in  $O(E \lg V)$  time
  - On general digraphs without negative cycles: Bellman-Ford’s algorithm runs in  $O(V \cdot E)$  and can also detect negative-weight cycles
- Minimum spanning tree (MST)
  - $G$ : connected, undirected, weighted
  - MST can be computed in  $O(E \lg V)$  time with Prim’s or Kruskal’s algorithms
  - Union-find data structure: supports Find and Join/Union operations.

### 13. Complexity:

- $P$ : problems that can be solved in polynomial time (on a deterministic Turing machine)
- $NP$ : problems that can be verified in polynomial time
- $NPC$ : a problem is in  $NPC$  if it is in  $NP$  and all problems in  $NP$  reduce to it in polynomial time. Intuitively, NPC is the core of hard problems in NP.
- Some NPC problems: SAT (satisfiability: is there an assignment that makes a given formula true), traveling salesman (TSP: find minimum-cost tour), longest path (find the longest simple path in a graph); find the largest clique subgraph (a clique is a complete graph).
- All NPC problems reduce to each other therefore if any of them is shown to be in P, it follows they all are and  $P = NP$
- The \$1M questions: Is  $P = NP$ ? Not known. Most people believe the answer is NO.