Fundamental techniques

- The greedy method
  - philosophy (greedy choice, substructure property)
  - problems
    - Fractional knapsack
      - algorithm
      - run-time
    - Task scheduling
      - algorithm
      - run-time
- Divide & Conquer
  - philosophy (divide, recur, conquer)
  - problems
    - Merge Sort
      - algorithm
      - run-time
    - Integer Multiplication
      - algorithm
      - runtime
  - recurrence equations and master theorem
- Dynamic programming
  - philosophy (subproblem optimality) (bottom-up) (subproblem overlap) (table)
    - define subproblems
    - show subproblem optimality
    - express solution to a larger problem through solutions to smaller problems (recurrence formula)
    - implementation
  - problems
    - 0/1 knapsack problem
      - solution and algorithm
      - complexity
    - matrix chain multiplication
      - solution and algorithm
      - complexity

Graphs

- Definitions
  - graph, vertex, edge, directed, weighted, vertex degree [redacted], adjacent, incident, path, simple path, cycle, simple cycle, subgraph, spanning subgraph, connected, connected components, spanning trees, forest, biconnected graph (components), separation vertex and edge
- Properties
- Presentations
  - edge list, Adjacency list, adjacency matrix
  - performances
- DFS
  - algorithm (time bound)
  - properties
    - connected component of v by DFS(G, v)
    - spanning tree by red edges (discovery edges, back edges = black)
  - applications
    - path finding
    - cycle finding
    - connectedness
    - connected components
    - spanning tree (forest)
    - biconnected components
      - be able to find (any method)
      - separation vertices
      - separation edges
      - biconnected components
- BFS
  - algorithm (time bound)
  - properties
    - connected component of v by BFS(G, v)
    - spanning tree by discovery edges (cross edges)
    - layering the vertices of G $L_0, L_1, L_2, \ldots$
  - applications
    - connected components (connectedness)
    - spanning tree (forest)
    - cycle finding
    - path with min. number of edges

— Comparison of DFS and BFS

Directed graphs

- Definitions
  - in-degree, out-degree, directed path, reachability, directed cycle, DAGs, strong connectivity
- Representation
  - $v \rightarrow$ (incoming edges)
  - $\rightarrow$ (outgoing edges)
- Directed DFS (complexity)
  - strong connectivity algorithm (complexity)
- Transitive closure
  - definition
  - algorithm (Floyd-Warshall)
    - running time
- DAGs and topological sorting
  - any topological sorting algorithm (one by one, DFS)
  - running time

Weighted graphs

- Shortest path problem formulation
- Shortest path tree and Dijkstra's algorithm
  - algorithm
  - complexity
  - applicability (no neg. edges)
- Bellman-Ford algorithm
  - algorithm
  - complexity
  - applicability (neg. edges-OK, neg cycles-no)
- Shortest path in DAGs and linear time algorithm
  - algorithm (uses topological sorting)
  - applicability (neg. edges - OK)
- no all pairs sh. paths (Floyd-Warshall)

- Minimum Spanning trees
  - definitions
  - Prim-Jarnik's algorithm
    - algorithm
    - complexity
    - properties behind the correctness (partition property) (cycle property)
  - Kruskal's Algorithm
    - algorithm (diff. from P.-J. approach)
    - data structure and implementation (find, union)
    - complexity
  - no Boruvka's algorithm

Maximum Flows

- Definitions (edge capacity, flow network, source, sink, flow, cut, flow over cut, cap. of a cut)
- Maximum Flow problem formulation.
- Flow augmentation
  - augmenting path
- Ford-Fulkerson's algorithm
  - be able to apply/use
  - complexity
- Max-Flow and Min-Cut Theorem.

String Matching

- def. (string, substring, prefix, suffix)
- Problem formulation
- Brute-Force alg.
  - algorithm
  - complexity
- Boyer-Moore algorithm
  - algorithm
  - Last occurrence function (run time to build)
  - run-time of the BM algorithm
- Knuth-Morris-Pratt's alg
  - algorithm (be able to use)
  - Failure function (no details of computation)
  - complexity