CU Boulder 5454 Review Topics
Fall 2019, midterm

This list of topics covered so far is not guaranteed to be comprehensive.

1 Background, Prereqs, Intro

Topics:

- word RAM model
- big-O, big-Omega, little-o, little-omega
- example: stable matching problem and deferred acceptance algorithm

2 Module 1

2.1 Graph Search

Definitions:

- graph, weighted and unweighted, directed and undirected
- Path, cycle, neighbor, out-degree, in-degree
- s-t reachability problem on unweighted graphs
- directed acyclic graph (DAG)
- topological sort (also called linearization)
- Tree, parent, child, root, subtree
- s-t shortest-path problem on unweighted and weighted graphs
- Using graph search for tasks like: check if e is part of any cycle; check if the graph is bipartite

Algorithms (or algorithmic frameworks):

- depth-first search (DFS)
- breadth-first search (BFS)
- how to use DFS and BFS; know when each is appropriate
- Dijkstra's
- Bellman-Ford

2.2 Dynamic Programming

Definitions/topics:

- Dynamic programming paradigm; general outline of DP
- defining a subproblem
- defining the "recurrence", how to solve a subproblem using solutions to earlier ones
- defining base cases
- solving subproblems in order (what order?)
- returning only the value of the optimal solution (e.g. length of longest increasing subsequence) versus reconstructing the solution (e.g. returning the subsequence itself)
- Memo-ization versus DP

Problems we considered:

- Shortest path in DAGs
- Longest increasing subsequence
- Edit distance
- Knapsack problem
- All-pairs shortest paths in graphs (Floyd-Warshall algorithm)
- Longest common subsequence of two strings
- Change-making problem

2.3 Flows, Cuts, and Matchings

Definitions and theorems:

- max s to t flow problem (hint: first define the input, then define a valid *flow* including the two constraints, then define the *amount* of the flow)
- min s to t cut problem (hint: first define a *cut*)
- max-flow min-cut theorem (need to recall and understand statement, do not need to prove from scratch)
- Residual graph G_f for flow f
- definition of bipartite graph
- definition of a matching

Algorithms and problems:

- Ford-Fulkerson framework (why is Ford-Fulkerson not a well-defined algorithm?)
- Edmonds-Karp algorithm
- algorithms for finding a min cut in a directed, weighted graph
- graph reductions: reducing one problem to another by constructing related graphs
- e.g. vertex capacity constraints, multiple sources/sinks
- the Integrality theorem
- edge-disjoint paths
- s-t connectivity (number of edges that must be removed so there is no path)
- maximum matching in bipartite graphs

3 Module 2

3.1 Basics of approximation algs

Definitions and concepts:

- recognize if a problem is maximization or minimization
- α approximation ratio and C-approximation
- maximum and *maximal* matchings in bipartite and general graphs
- max-weighted matchings in bipartite and general graphs
- Load balancing problem
- Min vertex cover
- Knapsack, revisited

Algorithms:

- Greedy algorithm for finding a *maximal* matching in bipartite and general graphs; 0.5 approximation ratio to *maximum* matching
- Greedy algorithm for load balancing, 2-approx
- Using matching alg for min vertex cover, 2-approx
- CarefulGreedy algorithm for Knapsack, 0.5 approx ratio

3.2 Basics of online algs

Definitions and concepts:

- general concept of online algorithm
- *instance* of the problem
- competitive analysis: compare to offline OPT that knows the instance in advance
- α competitive ratio, C-competitive

Problems and algorithms:

- Ski rental problem, 2-competitive algorithm
- Online unweighted bipartite matching, 0.5-competitive
- Online vertex cover, 2-competitive
- Online weighted bipartite matching (naive model), impossibility
- Online bin packing problem, 2-competitive

3.3 Basics of randomized algs

Definitions and concepts:

- Review of probability
- Randomness in word RAM model
- Definition of approximation and competitive ratio for randomized algorithms
- Key facts: linearity of expectation, law of total probability

Problems and algorithms:

- Max-3SAT 7/8
- Max Cut 0.5
- Min-weighted vertex cover (at least know the algorithm, do not need full analysis)