Some practice problems

CS218, Winter 2020

Greedy
A 1-2 graph is an undirected weighted graph whose edges have weights either 1 or 2. Give an efficient algorithm to compute the shortest paths in a 1-2 graph from a given vertex $s$ to any other vertex. Remember to analyze the complexity of your solution.

You are given an implementation of Dijkstra that uses an unsorted array for the priority queue. What is the time complexity of Dijkstra as a function of $n$ and $m$? Under which conditions on $n$ and $m$ is the array-based implementation faster than the binary-heap-based implementation?
Greedy (proof of optimality)

Problem 10. Assume that you are given two unsorted arrays $A = \{a_1, a_2, \ldots, a_n\}$ and $B = \{b_1, b_2, \ldots, b_n\}$ of $n$ distinct positive integers. Give a $O(n \log n)$-time greedy algorithm that determines an ordering of the elements of $A$ and $B$ such that $W = \sum_{i=1}^{n} a_i b_i$ is maximized. Explain why your algorithm runs in $O(n \log n)$-time, and prove the greedy choice property for your algorithm.

Dynamic Programming
**Dynamic programming (design)**

- Give an efficient algorithm for the following problem
- Given a rod of length $n$, and a set of prices $p_i$, $i=1,2,\ldots,n$ for a rod piece of length $i$, determine the maximum revenue obtainable by cutting up the rod and selling the individual pieces
### Dynamic programming (design)

We say that a path in a graph is *even-length* if the number of edges in the path is even.

You are given a directed acyclic graph $G = (V, E)$ and two vertices $s, t \in V$. Design a $O(n + m)$-time algorithm that computes the number of even-length paths from $s$ to $t$. 

<table>
<thead>
<tr>
<th>Length $i$</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
<th>10</th>
</tr>
</thead>
<tbody>
<tr>
<td>Price $p_i$</td>
<td>1</td>
<td>5</td>
<td>8</td>
<td>9</td>
<td>10</td>
<td>17</td>
<td>17</td>
<td>20</td>
<td>24</td>
<td>30</td>
</tr>
</tbody>
</table>

![Diagram](image)
Dynamic programming (2D design)

We are given a list of $n$ items with sizes $s_1, s_2, \ldots, s_n$. A sequential bin packing of these items is an assignment of items to bins, such that in each bins the items are consecutive. (That is, each bin has items $s_i, s_{i+1}, \ldots, s_j$ for some indices $i < j$.) Bins have unbounded capacities. The load of a bin is the sum of the elements in it. Give an algorithm that determines a sequential packing of $n$ items into $k$ bins for which the maximum load of a bin is minimized. Analyze the time-complexity and space-complexity.