PSO #6 Solutions Sketch (Week 8)

Week of 2021-10-11

1 Optional: Cover a DP problem from last week of your choice if you didn't cover them all (Highly Recommended if they need)

2 Warm-up: Greedy

1. Given 3 arrays of positive numbers, you can remove one element at a time from any array, but it must be from the end of that array. Determine a greedy algorithm to remove the minimum number of elements from the end of the arrays such that the sum of the elements of the three arrays are all equal.

   Solution: Just remove the element from the array with maximum sum each time until the sums are equal.

2. Consider m hard drives $D_1, D_2, \ldots, D_m$, each with capacity C_m. You receive a stream of n requests for memory, one at a time, each requesting for memory of size $k_1, k_2, \ldots, k_n$. Memory must be allocated if possible, and freeing is not considered.

   Develop a greedy algorithm to handle the stream of requests, discuss its runtime, and determine if it is optimal.

   Solution:

   We use a best-fit strategy, greedily allocating requests to a drive which has enough space to handle the request **and** the minimal space remaining after allocation.

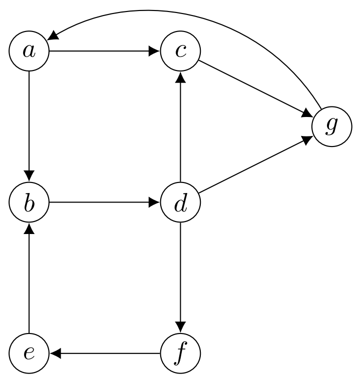
   Store the amount of remaining space for each drive in a sorted array of size m. The initial sort takes $O(m \log m)$. For a given request, binary search the array for the best-fit disk, allocate data, and re-insert the updated drive. Since we have to handle n total requests, overall runtime is $O(m \log m + n \log m)$

   Best-fit is not always optimal. Consider two disks such that $T_1 = 50$, $T_2 = 70$. Let the requests be $k_1 = 40$, $k_2 = 50$, $k_3 = 30$. The best-fit algorithm will assign request 1 to drive 1, request 2 to drive 2, and not be able to accommodate request 3. The optimal solution assigns request 1 and 3 to drive 2, and request 2 to drive 1.

   Note that it is impossible to achieve an optimal solution with an input stream.

3 Graphs

1. Run DFS and BFS on the following graph (start from vertex a)



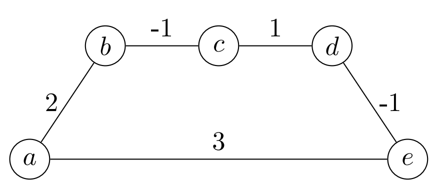
2. Let G be a weighted directed graph with negative weights (but no negative cycles). We want to find the length of the shortest path from s to every vertex. Alice proposes the following in order to apply Dijkstra's:

   - Find the minimum weight M in G
   - Add $|M|$ to the weight of every edge in G
   - All edges are now positive, so apply Dijkstra's

   Does this work?

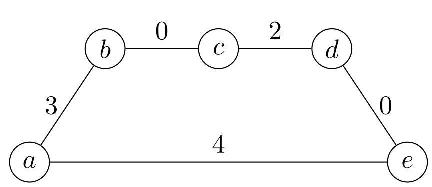
   Solution:

   No. Consider:



   where the shortest path from $a \rightsquigarrow e$ is $a \to b \to c \to d \to e = 1$.

   After adding $|M| = 1$, we have the graph



   but now, $a \to e = 4$ is the shortest path, not $a \to b \to c \to d \to e = 5$

3. Given an undirected, connected graph G (positive edge weights only), determine a collection of edges to form a connected graph G' where the product of the edge weights is minimized.

   Design a greedy algorithm to produce such a collection of edges.

   Solution: Two options:

   Option 1: Apply the log function to all edge weights and find the minimum spanning tree.
   Option 2: Find the minimum spanning tree. They're the same.