6.0002 Assignment 1

Pureumae Lee

February 14, 2024

Problem A.5

1. What were your results from compare_cow_transport_algorithms? Which algorithm runs faster? Why?

The greedy algorithm was 1,000 times faster than the brute force algorithm. Brute force considers every permutation, which gives $O(2^N)$ of time complexity. Whereas greedy has $O(NlogN)$ at most.

2. Does the greedy algorithm return the optimal solution? Why/why not?

It doesn't. An easy counterexample is [6, 5, 3, 2, 2, 2]. We can easily find that minimum round needed is 2, [6, 2, 2] and [5, 3, 2]. But with greedy algorithm, it needs 3, [6, 3], [5, 2, 2], [2].

3. Does the brute force algorithm return the optimal solution? Why/why not?

It does. Brute force checks power set of given set, so basically it checks every state it could have and find the best optimal solution.

Problem B.2

1. Explain why it would be difficult to use a brute force algorithm to solve this problem if there were 30 different egg weights. You do not need to implement a brute force algorithm in order to answer this.

Similar to A.5. Considering a power set of given problem, it requires $O(2^{30})$ of time complexity, which take quite a while.

2. If you were to implement a greedy algorithm for finding the minimum number of eggs needed, what would the objective function be? What would the constraints be? What strategy would your greedy algorithm follow to pick which coins to take? You do not need to implement a greedy algorithm in order to answer this.

It would be easy to implement a greedy algorithm on this problem. The function should take as many as possible eggs in order of weight. For example, take maximum eggs weight of 25, then weight of 10, etc. Then it would be the minimum number of eggs and also satisfies the constraints.

3. Will a greedy algorithm always return the optimal solution to this problem? Explain why it is optimal or give an example of when it will not return the optimal solution. Again, you do not need to implement a greedy algorithm in order to answer this.

In this case, it does. The greedy algorithm will act as same as dynamic programming since it will pick maximum eggs of the most heaviest egg and so on. For our problem, both algorithms will give 9 eggs.