Problem Set-1

Problem A.5: Writeup

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

Greedy Algorithm for cow_data1:

Transportation of cows : [['Betsy'], ['Henrietta'], ['Herman', 'Oreo'], ['Millie', 'Maggie', 'Moo Moo'], ['Milkshake', 'Lola', 'Florence']]

Length of Transportation : 5

Computational Time : 0.00024509429931640625 seconds

Brute Force Algorithm for cow_data1:

Transportation of cows : [['Betsy'], ['Milkshake', 'Oreo', 'Lola'], ['Millie', 'Moo Moo', 'Florence'], ['Henrietta'], ['Maggie', 'Herman']]

Length of Transportation : 5

Computational Time :0.311614990234375 seconds

Greedy Algorithm for cow_data2:

Transportation of cows : [['Lotus'], ['Horns'], ['Dottie', 'Betsy'], ['Milkshake', 'Miss Moo-dy', 'Rose'], ['Miss Bella']]

Length of Transportation : 5

Computational Time :0.0001761913299560547 seconds

Brute Force Algorithm for cow_data2:

Transportation of cows : [['Horns'], ['Miss Bella', 'Dottie'], ['Lotus'], ['Miss Moo-dy', 'Rose', 'Milkshake'], ['Betsy']]

Length of Transportation : 5

Computational Time : 0.01152491569519043 seconds

From the above results, it is clear that a greedy algorithm runs faster than brute force because it always picks the heaviest cows first and provides the solution quickly (complexity is m log n), whereas brute force algorithm look at every possible combination of trips and pick the best one so the computational time is high.

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

The greedy algorithm does not always yield optimal solutions, because it doesn't even know how good the approximation is.

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

The brute force algorithm provides the optimal solution, because it enumerates all possible combinations of items and removes all of the combinations whose total units exceed the allowed weight. From the remaining combinations choose any one whose value is largest.

Problem B.2: Writeup

1.Explain why it would be difficult to use a brute force algorithm to solve this problem, if there were 30 different egg weights?

Brute force algorithm would take a lot of time to solve this problem, as it considers all possible combinations to return the solution

2. 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 be?

In this case, the objective function would be eggs with largest weight, the constraints would be the overlapping subproblems(i.e. minimum number of eggs)

3. Will greedy algorithm always return optimal solution to this problem?

No, greedy algorithm often provide adequate (though not necessarily optimal) solutions. Finding an optimal solution using greedy algorithm is usually exponentially hard.