CMPS102 HW2
Jinxuan Jiang
jjiang17@ucsc.edu

Question 1 (Star Wars):

Match satellites to planets to maximize the Empire's benefit in O(n log n) time using a greedy algorithm.

Solution :

i.) Algorithm:

1. Sort the order of planets by their benefits. **(O(n log n))**
2. Match the planet with maximum benefit with the smallest fi which can break the planet's shield by using Binary Search Tree. **O(n)**
3. If there is such a satellite, remove the satellite;If there is no satellite exists, continue to next planet.

   Total complexity : O(n log n) + O(n) = O(n log n)

ii.) Proof of Correctness:

Claim:

The algorithm will always give the maximum benefit pairs.

Proof by induction:

Base case:

When n = 1, the number of death satellites and the number of planets are both 1.

Because there is only one pair of death satellites and planets, the benefit must be the maximum benefit.

Induction step:

For any n ≥ 2, pick n randomly. Assume there are n satellites and n planets, and the paired satellites and planets gives the maximum profits. We need to prove that for n+1 pairs of Satellites and planets, the assumption is still correct.

By induction hypothesis, we can conclude that the first n iterations will produce the maximum profits. For the last iteration. Whether we can find a f(j) that is bigger than si, the last

choice for f(j) is still the most efficient which means that it would have the maximum profit for the last iteration. Therefore, the overall iterations will produce the global maximum.

So, the algorithm will always give the maximum benefit pairs.

Question 2 (Super Slam):

Solution:

i.) Algorithm:

1. Sort array A and B is decreasing order. **(O(n log n))**
2. Pair A[i] with B[i] for all the elements in A and B and sum all the |A[i] - B[i]| **(O(n))**

Total time complexity: O(n log n)+O(n) = O(n log n)

ii.) Proof of correctness:

Claim 1:

Each pair has the smallest difference.

Proof:

Let i, j be two integer such that i<j.

According the the algorithm above, we have A[i] > A[j], B[i] > B[j].

Then, we have:

| A [i] - B [i] | + | A [j] - B [j] | ≤ | A [i] - B [j] | + | A [j] - B [i] |

Therefore, each pair always has the smallest difference.

Claim 2:

The algorithm will get the list of pair with minimum sum of difference.

Proof:

From claim 1, we know that each pair has the smallest difference, so the sum of each pair will have the minimum sum.

iii.) Bonus:

If the notion of imbalance is modified from $|w(a_i) - w(b_j)|$ to $(w(a_i) - w(b_j))^2$, the same algorithm will still give the optimum solution.

Question 3:

Solutions:

i.) Algorithms:

Question 4

Solutions:

i.) Algorithms:

1. Sort the ships according to their unloading capacity in decreasing order.
2. For every ship on the port, pick the ship with largest containers and unload them
3. Make sure the departure time ti ≥ currentTime +1 Hr.

ii.) Proof of correctness:

Claim :

The Algorithm will generate the maximum total number of containers.

Proof by contradiction:

Assume that there exists an optimal algorithm that is at least better than the algorithm above. Because the array above is sorted, for each time, the biggest possible will always exists. Another thing is if the ship's leaving time t(i) < currentTime + 1hr, the only way that one ship is picked and has more containers than the algorithm above is by not satisfying the leaving time if statement above. In this situation, the ship should not be picked by the optimal algorithm because the unloading process cannot be completed before

leaving, which is a contradiction. So, my algorithm above is at least as sufficient as the optimal one. As said above, each choice is optimal, then, the global choice should be optimal too.. Therefore, the algorithm will generate the maximum total number of containers.