TIN093 Xinyan Liu

Problem 3

(a)
If the payment is 1, 99, 100, 99

Greedy Approach: Pick 100, as it is the highest one, and we should deleted two 99 (near 100). As a result, what we choose is 1 and 100, and the sum is 100+1=101

Optimal Approach: choose two 99, and the sum is 99+99=198 (As we do not allow working in consecutive two days)

198 > 101, so the greedy solution is not the optimal one

(b)
If the payment is 10, 1, 10, 1, 10, 100

Greedy Approach:
For odd (1st, 3rd, 5th): We choose 10 10 10, and 10+10+10=30
For even (2nd, 4th, 6th): We choose 1 1 100, and 1+1+100=102
102 > 30, so we choose 1 1 100

Optimal Solution: We can choose 10(1st) 10(3rd) 100(6th), and the sum is 10+10+100=120, which is greater than 102 (greedy solution)

Therefore, the greedy method is not optimal one

Problem 4

(a)

The fastest way is to use the sorting Algorithms: O(nlogn)

As we give some example, for [50 40 30 20 10], we group them as (50 40 30), (20,10), therefore 30 is for free, and we can not make 50 or 40 for free because there are no other two elements that are higher than 50 and 40, therefore we save the cost.

Furthermore, if the costs are [60,50,40,30,20,10], we group them as (60, 50,40),(30,20,10), we can let 40 and 10 for free, totally saving $40 + 10 = 50$, for other combinations who might save the money greater than 50:

60+50=110 (not applicable)
60+40=100 (not applicable)
60+30=90 (not applicable)

60+20=80 (not applicable)
60+10=70 (not applicable)
Since no two elements are greater than 60

50+40=90 (not applicable)
50+30=80 (not applicable)
50+20=70 (not applicable)
50+10=60 (not applicable)
Since two elements are greater than 50

40+30=70 (not applicable)
40+20=60 (not applicable)
Since if we let 40 for free, one group should be (60,50,40), if group in this way, the other group should be (30,20,10), it is impossible to let 20 or 30 for free.

So actually 40+10=50 is the optimal solution

If it is right, what we do is to let the (3k)th most expensive items (for $k = 1, 2, 3, \ldots$) for free, we need to prove that sorting is the quickest way to find these items.

1. **Optimal Grouping Requires Identifying the (3k)th Most Expensive Items**:
   - To maximize savings, we need to ensure that the (3k)th most expensive items (for $k = 1, 2, 3, \ldots$) are the ones given for free. This is achieved by pairing each such item with two more expensive items in a group of three.
   - Formally, for each group of three items, the two most expensive items are paid for, and the (3k)th most expensive item is free.
2. **Sorting Enables Efficient Retrieval of the (3k)th Most Expensive Items**:
   - If the items are sorted in descending order, the (3k)th most expensive item can be directly retrieved from the sorted list using its index. For example:
     - The 3rd most expensive item is at index 2 (0-based indexing).
     - The 6th most expensive item is at index 5, and so on.
   - Retrieving an item by index in a sorted list takes $O(1)$ time.
3. **Without Sorting, the Time Complexity Increases Significantly**:
   - If sorting is not used, we must repeatedly compare each element to all other elements in the list to determine the (3k)th most expensive items.

- For each of the $\left\lfloor \frac{n}{3} \right\rfloor$ free items, we need to perform a linear search or selection algorithm to find the (3k)th most expensive item.
- Each selection operation takes $O(n)$ time, leading to a total time complexity of:

$$O(n \cdot n) = O(n^2).$$

4. **Sorting Provides the Fastest Solution**:
   - Sorting the list of n items takes $O(n \log n)$ time using efficient algorithms like quicksort or mergesort.
   - After sorting, retrieving the (3k)th most expensive items takes $O(1)$ time per item, and grouping the items into triples takes $O(n/3) = O(n)$ time.
   - Thus, the overall time complexity of the sorting-based approach is:

$$O(n \log n) + O(n) = O(n \log n).$$

5. **Conclusion**:
   - Sorting is the most efficient way to solve this problem because it reduces the time complexity from $O(n^2)$ (without sorting) to $O(n \log n)$
   - By sorting the items in descending order, we can directly retrieve the (3k)th most expensive items and group them optimally, ensuring maximum savings.

(b - Prove by Exchange)

Greedy Solution

Definitions

- Let F denote the greedy solution, where the free items are chosen greedily.
- Let F' denote an alternative solution that is claimed to be better than F.
- Let $f_1, f_2, \ldots, f_n$ be the free items in F.
- Let $f'_1, f'_2, \ldots, f'_n$ be the free items in F'.

Exchange Argument

To show that F is optimal, we perform the following exchange:

1. Swap f'_1 (a free item in F') with f_1 (a free item from the greedy solution F).
2. After the swap, the free items in F' are modified to produce a new list of free items, where the new $F' = \{f_1, f'_2, \ldots, f'_j, \ldots, f'_n\}$, with f_1 now free instead of f'_1.

3. In this case, because f_1 is chosen by the greedy solution, it must be the third largest item in its group. For f'_1, if it is not chosen by the greedy solution, it must be cheaper than or equal to f_1. Therefore, after the exchange, the total cost of F' will either remain the same or decrease. But If the total cost of F' increase, it will contradict the assumption that F' is optimal. Therefore, $f_1 = f'_1$ must hold.

4. Following the same logic, we can exchange f_2 with f'_2, f_3 with f'_3, and so on. Since f_i is always chosen by the greedy solution, it is the smallest item in its group. Except for the two items that are greater than it. All other items in the list are smaller than or equal to f_i. Thus, we can exchange f_i with f'_i for all i from 1 to n, and the optimal solution F' will eventually become identical to the greedy solution F as $f_n = f'_n$ must hold to ensure that F' is the optimal solution.

5. We can now conclude that the greedy solution F is optimal.

(b - Prove by Induction)

Define $f_1, \ldots, f_n$ as the free items chosen by the greedy solution, and $f'_1, \ldots, f'_n$ as the free items chosen by the optimal solution. Here, f_i represents the price of the book chosen as free in the i-th group of three books.

Claim:

If for the first $n-1$ groups,

$$\sum_{i=1}^{n-1} f_i \geq \sum_{i=1}^{n-1} f'_i,$$

then for the n-th group,

$$\sum_{i=1}^{n-1} f_i + f_n \geq \sum_{i=1}^{n-1} f'_i + f'_n.$$

Proof:

- **Base case:**
  When $n = 1$, only one book can be chosen as free. This means there are only three books in the first group. Using the greedy method, the free book must be the third most expensive one in the group. For any other method, the price of the chosen free book cannot exceed the price of the third most expensive book. Therefore,

$$f_1 \geq f'_1.$$

- **Inductive step:**
  Assume that for the first $n-1$ groups,

$$\sum_{i=1}^{n-1} f_i \geq \sum_{i=1}^{n-1} f'_i.$$

  For the n-th group, the greedy solution chooses f_n, which is the third most expensive item from the remaining books. For any other solution, the free item f'_n must either be the same as f_n or have a smaller price. Thus, it follows that:

$$f_n \geq f'_n.$$

  Adding f_n and f'_n to the sums of the first $n-1$ groups, we have:

$$\sum_{i=1}^{n-1} f_i + f_n \geq \sum_{i=1}^{n-1} f'_i + f'_n.$$

  By the principle of mathematical induction, the inequality holds for all n.

$$\sum_{i=1}^{n} f_i \geq \sum_{i=1}^{n} f'_i.$$

As a result, we can conclude that for all n, the total sum of the free items, $\sum_{i=1}^{n} f_i$, is maximized when using the greedy method.