CS 577: HW 5

Daniel Ko

Summer 2020

§1 Joe is a project manager in a software company called SuperDuo. Joe found out that programmers usually produces the highest quality codes when they are working in pair. However, the productivity of each pair of programmers is the speed of the slower programmer. So Joe needs to figure out the best strategy for pairing the programmers to generate the maximum sum of productivity for his project. Suppose that the number of programmers in Joe's project is even.

§1.1 Give a greedy strategy that maximizes the sum of the productivity of all pairs. You should describe the greedy strategy first and then write it in the form of an algorithm. Also you should analyze the computing complexity of the greedy algorithm.

The greedy strategy is to first sort the programmers in ascending order of productivity. Then, we pair the first two programmers, then the next two, etc. We sum up all the slower programmer in each pair, which will be our maximum sum of productivity.

```
Data: A list gamer of even size n, containing the productivity level of each programmer
Result: The maximum sum of productivity for pair programming
def maxProd(gamer):
    sortedGamer ← Sort gamer in ascending order of productivity
    totalProd ← 0
    for int i = 0; i < n; i+=2:
        totalProd += sortedGamer[i]
    return totalProd
```

The time complexity of maxProd is $O(n \log n)$

Proof. We first sort the list gamer of size n. Using an efficient sorting algorithm such as merge sort, this will take $O(n \log n)$ time. Then, we sum up every other element in the list, which takes $O(n)$ time. The rest of the computations take constant time. Hence, the total time complexity is $O(n \log n + n) = O(n \log n)$ □

§1.2 Prove that this greedy strategy indeed generates the maximum sum of productivity.

It does bro