[CSN212] Assignment 3
Greedy Algorithms and Amortized Analysis
Maximum Marks 100

1 Coin change problem [20 Marks]

Given a value V, *describe* and *analyze* an algorithm to find the minimum number of coins required to produce V where each denomination has an infinite supply, where the denominations of coins are:

1. Fibonacci numbers $1, 2, 3, 5, 8, 13, ...$
2. Powers of a constant positive integer $c > 1$, eg. $1, 2, 4, 8, 16, ...$ for $c = 2$.

2 Huffman Coding [20 Marks]

Huffman codes allow unequal bit allocation to characters in order to optimize space. The huffman coding algorithm iteratively chooses the least frequent characters, combines them together to form a new character whose frequency is the sum of these characters. The old characters become children of the combined character. This process is repeated until all characters are combined to form a single tree, and then the binary code is assigned to each character (leaves of the tree) as follows. On the path from the root to the leaf (character), append the code for the character by 0 whenever the path goes to the left child, and append 1 whenever the path goes to the right child. *Prove* that this greedy choice of the huffman coding produces the optimal size for a given text (set of characters and their frequencies).

3 Activity Selection and Scheduling variants [20 Marks]

Design and analyze an algorithm for the following, also prove its correctness:

1. *Activity Selection Variant.* Given a set of n activities with a start time s_i and finish time f_i ($s_i < f_i$). Two activities i, j *conflict* with each other if if the intervals $[s_i, f_i)$ and $[s_j, f_j)$ overlap. If multiple venues are available, we can always schedule all the activities. The aim is to schedule all the activities using minimum number of venues.
2. *Activity Scheduling Variant.* Given a set of n activities requiring p_i units of time to complete. All activities can be scheduled at the same venue but cannot overlap each other. The aim is to schedule all activities such that their *average* completion time is minimum.

4 MultiQuack [20 Marks]

Consider a data structure combining properties of both stacks and queues (recall Quacks) along with multiple operations simultaneously (recall MultiPop Stacks). It can be viewed as a list of elements written left to right such that three operations are possible:

- *MultiPush*(x, k) adds k copies of item x to the beginning of the sequence.
- *MultiEPush*(x, k) adds k copies of item x to the end of the sequence.
- *MultiPop*(k) removes k items from the beginning of the sequence (if exists).
- *MultiPull*(k) removes k items from the end of the sequence (if exists).
- *Lookup*(k) returns the kth item in the sequence (if exists).

Describe and analyze a simple data structure that supports these operations using $O(n)$ space, where n is the current number of items in the sequence, where lookup should require $O(1)$ worst-case time and all other operations require $O(1)$ amortized time.

5 Amortized Analysis of Union Find [20 Marks]

In the class, we saw $O(\log^* n)$ bound using the Aggregate method. Describe its amortized analysis

1. Using Accounting method.
2. Using Potential method.

Note: Solution should be inspired by classroom Aggregate analysis for $O(\log^* n)$ bound.