SOLUTIONS FOR HOMEWORK 3

Question 1

Claim 0.1. *No, this greedy strategy can not always give a optimal solution.*

Proof. It suffices to show a counterexample. Consider the room is available between 1. pm to 12. pm, and we have three activities $[1, 6), [6, 12), [2, 10)$. Clearly the greedy strategy in Question 1 will give $[2, 10)$, but the optimal solution is $[1, 6), [6, 12)$. □

Question 2

The greedy algorithm mentioned in the this question chooses compatible activity with earliest finishing time from the remaining activities in each step. Let $a_1, a_2, \ldots, a_k$ be the set of activities that we picked using this algorithm (sorted increasingly according to their times), let $b_1, b_2, \ldots, b_k$ be any other optimal solution to the same problem (sortied similarly).

Claim 0.2. *The finishing time of a_i is no later than that of b_i for all $1 \leq i \leq k$.*

Proof. Prove by induction on i. Let $s(a_i)$, $f(a_i)$ be the starting and fishing time of a_i respectively. $s(b_i)$ and $f(b_i)$ are defined similarly.
Base case. a_1 is picked as the one with earliest finishing time over all activities, hence $f(a_1) \leq f(b_1)$.
Inductive Hypothesis. $f(a_i) \leq f(b_i)$.
Inductive step. We want to prove $f(a_{i+1}) \leq f(b_{i+1})$ in this step.

By the Inductive Hypothesis, we have $f(a_i) \leq f(b_i)$, then $s(b_{i+1}) \geq f(b_i) \geq f(a_i)$, which means b_{i+1} is also compatible with a_i and is not one of $a_1, a_2, \ldots, a_i$.

If $f(b_{i+1}) > f(a_{i+1})$, a_{i+1} will not be chosen in our greedy algorithm because b_{i+1} is also a compatible activity in the remaining activities and has earlier finishing time. Therefore we must have $f(a_{i+1}) \leq f(b_{i+1})$.
Conclusion. By the principle of mathematical induction, we conclude that $f(a_i) \leq f(b_i)$ for $1 \leq i \leq k$.

□

Question 3

Solution. Huffman tree is presented by Figure 1. We have

$$5 \times (0.02 + 0.03 + 0.04 + 0.04) + 3 \times (0.01 + 0.17 + 0.2) + 0.4 = 2.46.$$

So on average we use 2.46 bits per symbol via Huffman code. As we have 8 different symbols, and $2^3 = 8$, we use 3 bits to represent each symbol in the Fixed-Length-Code. The improvement will be $\frac{3-2.46}{3} = 0.18 = 18\%$.

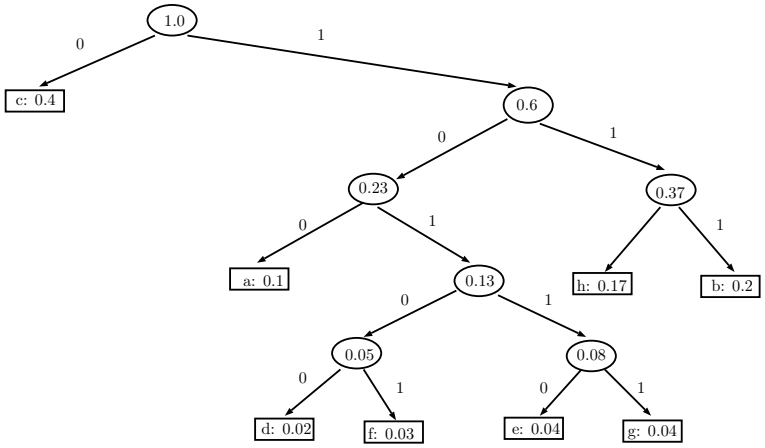

FIGURE 1. Huffman Tree for Question 3

QUESTION 4

16.1-2. Solution. The algorithm described in the question is a greedy algorithm, because it makes a greedy choice at each step and makes the choice before solving the subproblems. When we have a choice to make, make the one that looks best right now. Make a locally optimal choice in hope of getting a globally optimal solution.

Notations. Let s_i and f_i denote the start and finish times of activity a_i respectively. Define $S_k = \{a_i | f_i \leq a_k\}$ as the set of activities that finish before a_k starts.

Now we prove the optimality. We show two things,

- This problem exhibits optimal substructure.
- each greedy choice is included in an optimal solution for current subproblem.

Claim 0.3. *The given problem exhibits optimal substructure.*

Proof. Clearly for problem S_k, if a_j is the last activity to start in S_k, then a_j together with a solution of S_j (denote as $A_j \subset S_j$) will be a solution of S_k. If $\{a_j\} \cup A_j$ is an optimal solution of S_k, then A_j has to be a optimal solution of S_j otherwise it will contrast with the optimality of $\{a_j\} \cup A_j$. □

Claim 0.4. *Let S_k be a subproblem. Let $a_m \in S_k$ be the last activity to start in S_k, then a_m is included in an optimal solution of S_k.*

Proof. Let A_k be an optimal solution of S_k, let a_j be the last activity to start in A_k, then a_m is compatible with any activities in $A_k - \{a_j\}$ because $s_m \geq s_j$. Therefore $A'_k = (A_k - \{a_j\}) \cup \{a_m\}$ is another optimal solution of S_k which includes a_m. □

Claim 0.3 and claim 0.4 together yield what we need to prove.

16.3-8. (Adapted from Hani T.Dawoud's solution)

Claim 0.5. *Let C be an alphabet from which the maximum frequency is less than twice the minimum frequency. If $|C| = 2^k$, then Huffman algorithm will generate a complete binary tree, hence each symbol in C has codeword of length exactly k.*

Proof. Let's prove the claim by induction on k.
Base case. When $k = 1$, it is a trivial case.
Inductive hypothesis. Assume the claim is true for $|C| = 2^{l-1}$.
Inductive step. Now we consider the case $|C| = 2^{l-1}$. Let $f_1, f_2, \ldots, f_{2^l}$ be the frequencies of all 2^l symbols in C and they were listed in non-decreasing order. We say a node is *internal* if it is combined by other nodes (possibly internal nodes), for example, we combine f_1, f_2 to $(f_1 + f_2)$ as an internal node, but each of f_i is not internal. Note that for any internal node, it has strictly higher frequency than f_{2^l}, this means after 2^{l-1} steps of merge, we have $f_1+f_2, f_3+f_4, f_5+f_6, \ldots, f_{2^l-1}+f_{2^l}$, which can be considered as a alphabet $|C'| = 2^{l-1}$.

Note that $f_1 + f_2 \leq f_3 + f_4 \leq f_5 + f_6 \leq \ldots \leq f_{2^l-1} + f_{2^l}$, and $f_{2^l-1} + f_{2^l} < 2(f_1 + f_2)$, hence for C' we also have the property that the maximum frequency is less than twice the minimum frequency. Inductive hypothesis then gives a complete binary tree for C', it is then clear that Huffman algorithm will generate a complete binary tree for $|C|$. □