CSE100 Algorithm Design and Analysis — Homework set #3

Fall semester 2020 — Miguel Á. Carreira-Perpiñán

Total possible points: 100. This set covers chapters 15–16 of the textbook *Introduction to Algorithms*, 3rd. ed., by Cormen et al.

Exercise 1: dynamic programming and greedy algorithms (60 points). Consider the following equation:

$$\alpha_1 n_1 + \alpha_2 n_2 + \dots + \alpha_K n_K = N \tag{1}$$

where $\boldsymbol{\alpha} = (\alpha_1, \dots, \alpha_K)$ and N are known integer values satisfying $1 \leq \alpha_1 < \cdots < \alpha_K < N$, and $\mathbf{n} = (n_1, \dots, n_K)$ are unknown integer values $n_1, \dots, n_K \geq 0$. Sometimes this equation has no solution, for example for $\boldsymbol{\alpha} = (3, 4)$ and $N = 5$, but we will assume $\boldsymbol{\alpha}$ and N are such that at least one solution $\mathbf{n}$ exists. We want to find a solution $\mathbf{n} = (n_1, \dots, n_K)$ having smallest sum $n_1 + \cdots + n_K$. For example, if $K = 3$, $\boldsymbol{\alpha} = (2, 3, 7)$ and $N = 14$, then $\mathbf{n} = (0, 0, 2)$, $(7, 0, 0)$ and $(2, 1, 1)$ are all solutions, but $(0, 0, 2)$ is the optimal one.

1. (5 points) Come up with a brute-force approach that examines all possible solutions and show that its run time is at least exponential as a function of K.
2. (1 point) Prove that if $\alpha_1 = 1$ then there is always a solution.
3. Consider a dynamic programming algorithm to find an optimal solution.
   - (10 points) Write the value of an optimal solution as a recursive expression using the value of optimally solved subproblems. *Hint*: define a subproblem where we pick one "item" out of the $\alpha_1 + \cdots + \alpha_K$ "items".
   - (8 points) Write pseudocode that implements this solution top-down using memoization.
   - (13 points) Write pseudocode that implements this solution bottom-up by filling in a table in order. Give its run time.
   - (5 points) Write pseudocode for an algorithm to print the optimal solution found by the bottom-up algorithm (the values $\mathbf{n} = (n_1, \dots, n_K)$). Give its runtime.
4. Consider a greedy algorithm to find an optimal solution.
   - (10 points) First, consider the special case where $K = 3$ and $\boldsymbol{\alpha} = (1, 5, 10)$. Come up with a greedy algorithm to solve this case and prove it always finds an optimal solution for any N value (i.e., prove the greedy choice property).
   - (5 points) Now, consider the general case for arbitrary $\boldsymbol{\alpha}$, N as above. State a generalized version of the previous greedy algorithm (no need to write its pseudocode) and give its runtime.
   - (3 points) Does that greedy algorithm always work? If yes, prove it; if not, give a counterexample.

Exercise 2: Huffman codes (25 points). The following table gives the frequencies of each character in a file with 1 000 characters.

α	β	γ	δ	ϵ	ζ	η	θ
10	400	80	50	20	300	100	40

1. (2 points) How many bits do we need to store the file if using a fixed-length code?
2. (8 points) Show the code built by the Huffman algorithm, both as a tree and as a list (character, codeword).
3. (2 points) How many bits do we need to store the file with this Huffman code?

Assume now that all the the characters have the same frequency (= 125).

4. (8 points) Show the code built by the Huffman algorithm, both as a tree and as a list (character,codeword).
5. (2 points) How many bits do we need to store the file with this Huffman code?
6. (3 points) Generalizing from this example, what can you say about the code built by the Huffman algorithm for an alphabet with $n = 2^b$ characters each of which occurs with equal frequency f? Explain your answer.

Explain 3: longest common subsequence (15 points). Construct the matrices c and b as in fig. 15.8 in the textbook for the words SABATONS and ABBREVIATE, and give a longest common subsequence for them.