Homework Assignment #1

Due: Sep 25th, 2018

CMPUT 304
Department of Computing Science
University of Alberta

Note: All logarithms are in base 2 unless specified otherwise.
Exercises are for you and you alone. You need not submit them and they will not be graded.
You may submit as many answers to as many problems as you like.

Exercise I. True or False? Provide a *short* explanation.

(i) $19^{n^{75}} \in O(75^{n^{19}})$

(ii) $1.975^n + 1.025^n \in O(3^n)$.

(iii) $(\log\log n)^{\log n} \in \Omega(1.05^n)$

(iv) $n^{\frac{\log\log n}{\log n}} \in O(n^{0.1975})$.

(v) $n! \in O(2^{n\log n})$

(vi) $1^c + 2^c + \ldots + n^c \in \Theta(n^{c+1})$ for every constant $c \geq 0$.

(vii) For any functions $f, g : \mathbb{N} \to \mathbb{R}^+$, $f \in O(g)$ if and only if $g \in \Omega(f)$.

(viii) For any functions $f, g : \mathbb{N} \to \mathbb{R}_{\geq 1}$, $\log(f(n)) \in \Theta(\log(g(n))$ if and only if $f(n) \in \Theta(g(n))$.

Exercise II. Order the following list of functions by increasing big-O growth rate. Group together (for example, by underlining) those functions that are big-Θ of one another.

$1975n$	19^{7^5}	$2^n + 2^{n-1} + 2^{n-2} + \ldots + 2^0$	$(\log(n))^{1975}$	$n\log(1975n)$	4^n	$n^{1.975}$

Exercise III. In class we have shown that for any alphabet Σ and *any* optimal prefix encoding represented in a tree T, any leaf must have a sibling.
Use a similar approach to argue the following. For any alphabet Σ and *any* optimal prefix encoding represented in a tree T, the can be no node (a leaf or an internal node) without a sibling. In other words, T is a full tree (each node has either 0 or 2 children).

Exercise IV. Let $U = \{\bar{v}_1, ..., \bar{v}_n\}$ be an arbitrary set of n non-zero vectors in the Euclidean space $\mathbb{R}^d$. Denote $\mathcal{I} = \{A \subset U :\ A \text{ is linearly independent}\}$. Show that $\mathcal{M} = (U, \mathcal{I})$ is a matroid.

Recall, a set of k vectors $\bar{u}_1, \bar{u}_2, ..., \bar{u}_k$ is called *linearly independent* if the <u>only</u> solution to the equation $\lambda_1\bar{u}_1 + \lambda_2\bar{u}_2 + ... + \lambda_k\bar{u}_k = \bar{0}$ (where $\lambda_1, ..., \lambda_k$ are the scalar variables in this equation) is the trivial solution $\lambda_1 = \lambda_2 = ... = \lambda_k = 0$.

Problem 1. (1 pts) What is the optimal prefix-code for the alphabet $\Sigma = \{a, b, c, d, e, f, g, h\}$ with the following frequencies

a	b	c	d	e	f	g	h
0.09	0.11	0.08	0.02	0.1	0.38	0.21	0.01

Problem 2. (1 pts) Prove the following claim.
Let T be an optimal binary tree for alphabet Σ and let x, y be the two characters associated with two sibling leafs of T. Let Σ' be the alphabet we get by removing x and y from Σ and adding a single character $x|y$ (which means "x or y") with frequency $= x.freq + y.freq$. Let T' be the tree we get by removing the sibling leafs associated with x and y and labeling their former parent (and current leaf) with the new character $x|y$.
Then T' is an optimal tree for Σ'.

Problem 3. (1 pts) Given an integer $m \geq 2$, give an alphabet of size m – with character frequencies – for which the Huffman algorithm creates the most balanced tree. Explain.

Problem 4. (2 pts) Given an integer $m \geq 2$, give an alphabet of size m – with character frequencies – for which the Huffman code creates the most imbalanced tree. Explain.

Problem 5. (1 pts) Show that if $\mathcal{M} = (U, \mathcal{I})$ is a matroid and $x \in U$ then the restriction of $\mathcal{M}|_x = (U', \mathcal{I}')$ defined by $U' = U \setminus \{x\}$ and $\mathcal{I} = \{\emptyset\} \cup \{A \subset U' : \ A \cup \{x\} \in \mathcal{I}\}$ is also a matroid.

Problem 6. Let $\mathcal{M} = (U, \mathcal{I})$ be an arbitrary pair of a ground set U and a collection of subsets $\mathcal{I}$ which *not* a matroid — because, while $\mathcal{I}$ does satisfy downward closure, $\mathcal{I}$ doesn't satisfy the augmentation property.

(i) (1 pts) Show that there exists a weight function $w_1 : U \to \Re$, which is *not* the all-zero function, for which the greedy algorithm does succeed in finding a maximum-weight subset.

(ii) (3 pts) Show that there exists a weight function $w_2 : U \to \Re$ for which the greedy algorithm does *not* succeed in finding a maximum-weight subset.

Problem 7. (3 pts) **Chapter 16-5**:
In the problem of **unit length task scheduling with deadlines and penalties** the input consists of

- a set $S = \{a_1, a_2, ..., a_n\}$ of n unit-time tasks (each takes one unit of time).
- a set of n positive integer deadlines $d_1, .., d_n$ such that task a_i must conclude by time d_i. (each $d_i \leq n$)
- a set of n penalties $w_1, ..., w_n$ such that if task a_i is not finished by deadline d_i we incur a penalty of w_i (other we incur no penalties).

and the output is a schedule for S that minimizes the total penalty incurred by all tasks for which we've missed the deadline. Since all tasks take a unit of time, such a scheduling is given in the form of a permutation π of the elements of S: we first run task $a_{\pi(1)}$ for one unit of time until it concludes; then run task $a_{\pi(2)}$ for one unit of time till it concludes, and so on and so forth. Under this scheduling, task $a_{\pi(1)}$ concludes at time one, task $a_{\pi(2)}$ concludes at time 2, and in general, $a_{\pi(i)}$ concludes at time i.

Prove that the algorithm that greedily finds the maximum-cost set of tasks we can schedule without incurring any penalty (and then builds a suitable schedule) indeed solves this problem, by identifying the underlying matroid structure of the problem. (Prove this is indeed a matroid.)