Design and Analysis of Algorithms (CS F364) Comprehensive Exam (2018)

There are 8 questions in all and total marks is 10+10+10+10+10+(1+8+1)+10+10 = 80. Please show all steps in computations or proofs. This is a **closed book exam**. Calculators are not allowed. Time: 180 minutes.

1. In the *Modular FFT Algorithm*, instead of performing an $n-$element FFT over complex numbers (where n is even), we use the complete residue system modulo m (Z_m), where $m = 2^{\frac{tn}{2}} + 1$ and t is an arbitrary positive integer. Instead of addition of complex numbers, we use addition modulo m. Instead of multiplication of complex numbers, we use multiplication modulo m. We use $\omega = 2^t$ instead of ω_n as a principal nth root of unity, modulo m. Instead of ω_n^{-1}, we use the inverse ω^{-1} of ω computed in Z_m using efficient algorithms. Find $(2x+1) \times (3x+2)$ using the *Modular FFT Algorithm* (by taking $t = 8$) showing its divide and conquer graphs.

2. Using the *Matroid Algorithm*, solve the following instance of the scheduling problem (*scheduling unit-time tasks with deadlines and penalties for a single processor that minimizes the total penalty incurred for missed deadlines*):
Set of tasks: $(a_i)_{i=1}^{7} = (1, 2, 3, 4, 5, 6, 7)$
Deadline of tasks: $(d_i)_{i=1}^{7} = (4, 2, 4, 3, 1, 4, 6)$
Penalty of tasks (if a_i is not completed within time d_i): $(w_i)_{i=1}^{7} = (10, 20, 30, 40, 50, 60, 70)$

3. Determine a *Longest Common Subsequence* of $< 1, 0, 0, 1, 0, 1, 0, 1, 1, 0 >$ and $< 0, 1, 0, 1, 1, 0, 1, 1, 0, 1, 0 >$ using the *Dynamic Programming Algorithm*.

4. Using the *Ford-Fulkerson Algorithm*, solve the *Bipartite Matching Problem* for the instance of the graph given in problem 6 (only the adjacency matrix, not the weights).

5. Under the assumption of $\mathtt{P} = \mathtt{NP}$ (all complexity classes are defined for binary alphabet $\Sigma = \{0, 1\}$), prove that $\mathtt{P} - \mathtt{NPComplete} = \{\{\}, \{0 \cup 1\}^*\}$

6. Consider the following instance of the *Weighted Vertex Cover Problem* with vertices $\{v_1, v_2, v_3, v_4, v_5, v_6\}$, having weight of vertices defined as $w(v_1) = 1, w(v_2) = 3, w(v_3) = 5, w(v_4) = 2, w(v_5) = 4, w(v_6) = 4$, for an undirected graph having the following adjacency matrix: (edge between v_i and v_j is lebelled as e_{ij})

$$\begin{bmatrix} 0 & 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 & 1 \\ 1 & 1 & 1 & 0 & 0 & 0 \\ 1 & 1 & 1 & 0 & 0 & 0 \\ 1 & 1 & 1 & 0 & 0 & 0 \end{bmatrix}$$

   (a) Formulate the above problem as a *Set Cover Problem*.

   (b) Using the *Primal Dual Approximation Algorithm* for the *Set Cover Problem*, solve the above instance of the *Set Cover Problem*.

   (c) Now, from the above solution, get back your original solution of the *Weighted Vertex Cover Problem*.

7. Consider a very simple online auction system that works as follows. There are n bidding agents; agent i has a bid b_i, which is a positive natural number. We will assume that all bids b_i are distinct from one another. The bidding agents appear in an order chosen uniformly at random, each proposes its bid b_i in turn, and at all times the system maintains a variable b^* equal to the highest bid seen so far. (Initially b^* is set to 0.) What is the expected number of times that b^* is updated when this process is executed, as a function of the parameters in the problem? *Example.* Suppose $b_1 = 20, b_2 = 25$, and $b_3 = 10$, and the bidders arrive in the order $1, 3, 2$. Then b^* is updated for 1 and 2, but not for 3.

8. By making use of efficient algorithms, find the last three digits in the decimal expansion of $3^{20182018201820182018}$