Indian Institute of Information Technology Allahabad
Discrete Mathematical Structures (DMS)
Assignment

Program: B.Tech. 2$^{\text{nd}}$ Semester (IT)
Deadline: **April 25, 2024** Full Marks: 15

Let us define $\mathbf{N}$ = 700 - last three digits of your enrolment no., **Num** = number of simple graphs with $\mathbf{N}$ vertices, and let $\mathbf{T}$ be a number such that $\mathbf{Num} \equiv \mathbf{T} \pmod{26}$.
If $\mathbf{N}$ is even then **Question-I** is compulsory, and if $\mathbf{N}$ is odd then **Question-II** is compulsory. In addition to that, you have to choose any two **topics** from $\{\mathbf{T}, \mathbf{T}+1, \mathbf{T}+2\}$ (mod 26). You have to explain the chosen topics with at least one example wherever applicable. We would prefer to receive your final assignment having at least 2 pages and at most 5 pages.

Question-I [4+3+3]

1. Show that:
   If f is an increasing function that satisfies the recurrence relation $f(n) = af(n/b) + c$ whenever n is divisible by b, where $a \geq 1,\ b$ is an integer greater than 1, and c is a positive real number. Then

$$f(n) = \begin{cases} O(n^{\log_b a}), & \text{if } \ a > 1, \\ O(\log n), & \text{if } \ a = 1. \end{cases}$$

   Furthermore, when $n = b^k$ and $a \neq 1$, where k is a positive integer, $f(n) = C_1 n^{\log_b a} + C_2$, where $C_1 = f(1) + c/(a-1)$ and $C_2 = -c/(a-1)$.

2. Write algorithm based on the Kuratowski theorem.

3. Write code in C/C++/Python and compute time complexity in big-O estimate for Laplace expansion for determinants.

Question-II [4+3+3]

1. Show that:
   If f is an increasing function that satisfies the recurrence relation $f(n) = af(n/b) + cn^d$ whenever $n = b^k$, where k is a positive integer, $a \geq 1,\ b$ is an integer greater than 1, and c and d are real numbers with c positive and d nonnegative. Then

$$f(n) = \begin{cases} O(n^d), & \text{if } \ a < b^d, \\ O(n^d \log n), & \text{if } \ a = b^d, \\ O(n^{\log_b a}), & \text{if } \ a > b^d. \end{cases}$$

2. Write algorithm of Ore's Theorem on Hamiltonian Cycles.

3. Write code in C/C++/Python and compute time complexity in big-O estimate for Leibniz formula for determinants.

Topics are as follows: [5]

0. Big-O estimate for quicksort algorithm.
1. Ramsey numbers.
2. Big-O estimate for bubble sort algorithm.
3. Elliptic-curve cryptography.
4. Graph isomorphism problem.
5. Koch snowflake curve.
6. Big-O estimate for insertion sort algorithm.
7. Complexity of Matrix Multiplication algorithm.
8. Constant complexity and Linear complexity.
9. Logarithmic complexity.
10. P versus NP problem.
11. Linearithmic complexity.
12. Travelling salesman problem.
13. Exponential complexity.
14. Factorial complexity.
15. Sierpinski Gasket.
16. Complexity of the bubble sort.
17. Complexity of the insertion sort.
18. Possible positions in a $3 \times 3 \times 3$ Rubik's cube.
19. Big-O estimate for the number of comparisons used by a binary search.
20. RSA cryptosystem.
21. Big-O estimate for the number of multiplications and additions required to multiply two $n \times n$ matrices using the Fast Matrix Multiplication algorithm.
22. Prisoner's dilemma.
23. Big-O estimate for finding the determinant of $n \times n$ matrix with Laplace expansion.
24. Polynomial complexity.
25. Big-O estimate for tree sort algorithm.