Advanced Algorithms (3501)

Spring Semester, 2023

Reichman University

Problem Set 2

Published on: 16/05/2023
Submit by: 30/05/2023, 23:59

1. Let G be an undirected graph with no parallel edges or self loops. A *triangle* in G is a set of 3 distinct vertices v_1, v_2, v_3 such that each pair of them is connected by an edge. Design an algorithm that, given a graph G with n vertices, counts the number of triangles in G. The running time of your algorithm should be *asymptotically smaller* than n^3.

   (Hint: you may want to use fast matrix multiplication.)

2. An $n \times n$ matrix A is called a *Toeplitz matrix* if it is constant on its diagonals. That is, there exist $2n-1$ numbers $a_0, a_1 \ldots, a_{2n-2}$ such that

$$A = \begin{pmatrix} a_{n-1} & a_{n-2} & a_{n-3} & \cdots & \cdots & a_0 \\ a_n & a_{n-1} & a_{n-2} & \ddots & \ddots & a_1 \\ a_{n+1} & a_n & a_{n-1} & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & a_{n-2} & a_{n-3} \\ \vdots & \ddots & \ddots & a_n & a_{n-1} & a_{n-2} \\ a_{2n-2} & \cdots & \cdots & a_{n+1} & a_n & a_{n-1} \end{pmatrix}.$$

   Show an algorithm that, given a Teoplitz matrix $A \in \mathbb{C}^{n\times n}$ and a vector $v \in \mathbb{C}^n$ computes Av using $O(n \log n)$ arithmetic operations.

   (Hint: associate with A the polynomial $\sum_{i=0}^{2n-2} a_i x^i$ and with v the polynomial $\sum_{i=0}^{n-1} v_i x^i$. Try to read off the entries of Av from the product of these polynomials)

3. You're given a dice with n sides. In every roll of the dice, the probability that it falls on the i-th side is p_i, for every $1 \leq i \leq n$ (so in particular we must have $p_i \geq 0$ and $\sum_{i=1}^{n} p_i = 1$).

   We'll now play a game in which we roll the dice twice, and sum the two results that we get. We'd like to calculate the distribution of the outcomes of this game. That is, for every $2 \leq s \leq 2n$, we want to compute the probability that the sum of the results of both rolls is s.

   (a) Design an algorithm that computes all of these probabilities using $O(n \log n)$ arithmetic operations.

(b) We'll play the same game except we now roll the dice 2^k times instead of just twice (further assume $2^k \leq n$), and sum the results. Design an algorithm that computes the probabilities of all the outcomes of this game. Use as few arithmetic operations as you can.

4. In this question we'll design a parallel algorithm that computes the rank of an $n \times m$ matrix A over $\mathbb{R}$. Suppose without loss of generality that $m < n$.

(a) Show that $\text{rank}(A) = \text{rank}(A^T A)$.

(b) Denote $B = A^T A \in \mathbb{R}^{m \times m}$. Let $\lambda_1, \ldots, \lambda_m$ denote the eigenvalues of B. Let k be the number of λ_i's that equal zero. Show that $\text{rank}(B) = m - k$.

(Hint: you may find it convenient to use the fact that B is symmetric and hence it is diagonalizable, so the algebraic multiplicity of every eigenvalue equals the geometric multiplicity.)

(c) Let $p(x) = \det(xI - B) = (x - \lambda_1) \cdots (x - \lambda_m)$ be the characteristic polynomial of B. Show that x^k is the smallest power of x with non-zero coefficient of $p(x)$ (where k is as defined in the previous item). Use this fact to design an algorithm that computes $\text{rank}(A)$ in parallel time $O(\log^2 n)$.