BU CLA CS 530: Analysis of Algorithms

Spring 1996

Problem Set 10

Due Thursday 96.04.04, at 11:00 am, in class. Late submissions not accepted.

---

1. Exercise 30.2-3, page 709, in [CLR]. (*Hint*: Although you are asked to design an n-processor CRCW algorithm, you will get full credit for a constant time n^2-processor CRCW algorithm.)

2. Let A and B be two arbitrary $n \times n$ matrices (of real numbers or boolean values).

   (a) Design a CREW algorithm to multiply A and B in $\mathcal{O}(\log n)$ time using n^3 processors. (*Hint*: Make minor modifications in the standard sequential algorithm to multiply two matrices, given at the bottom of page 554 in [CLR]. Your CREW algorithm should work in two phases, the first of which computes $a_{ij} \cdot b_{jk}$ for every $i, j, k \in \{1, \ldots, n\}$ in constant time.)[1]

   (b) Describe a CREW algorithm to compute the powers A^1, A^2, A^3, …, A^n, in $\mathcal{O}(\log^2 n)$ time using $\mathcal{O}(n^4)$ processors. (*Hint*: Multiplication of square matrices is an associative operation.)

3. Let $G = (V, E)$ be a directed graph, with a non-negative weight function. Describe a CREW algorithm to compute all-pairs shortest paths in G in $\mathcal{O}(\log^2 n)$ time using $\mathcal{O}(n^4)$ processors. Assume G is represented by an adjacency-weight matrix $W = (w_{ij})$. (*Hint*: Consult Section 26.1 in [CLR], and solve Problem 2 first.) Can you reduce the number of processors to $\mathcal{O}(n^3)$?

4. Let $G = (V, E)$ be a connected undirected graph, and v a distinguished vertex in G. Describe a CREW algorithm to construct a BFS tree of G rooted at v, in $\mathcal{O}(\log^2 n)$ time using $\mathcal{O}(n^3)$ processors. (*Hint*: Solve Problem 3 first.)

5. Let $G = (V, E)$ be a connected undirected graph. Describe a CREW algorithm to determine whether G is bipartite, in $\mathcal{O}(\log^2 n)$ time using $\mathcal{O}(n^3)$ processors. (*Hint*: G is bipartite iff G does not have an odd cycle. Solve Problem 3 first.)

Created: 96.03.27

[1] Exercise 30.2-5 in [CLR], page 709, asks for an $\mathcal{O}(\log n)$-time EREW algorithm for matrix multiplication, which is a bit more complicated to design.