Algorithms – handout 4

Matrix Operations

November 25, 1999

Let $A = \{a_{ij}\}$ and $B = \{b_{ij}\}$ be two square matrices of order n. Their product $AB = C = \{c_{ij}\}$ is defined by $c_{ij} = \sum_k a_{ik}b_{kj}$. Note that the product operation is not commutative. Matrix products can be computed in a straight forward way in $O(n^3)$ time (where time is measured in number of scalar additions and multiplications). Strassen (1969) designed an $O(n^{\log 7}) \simeq O(n^{2.81})$ algorithm for this problem. His algorithm also uses subtractions. There are several version of Strassen's algorithm. We shall show one of them in class. A different one appears in the book by Cormen, Leiserson and Rivest.

The exponent for matrix multiplication, denoted by ω, has been further improved. The current best bound (by Coppersmith and Winograd) is $\omega \simeq 2.376$. The practical value of these algorithms is questionable, because of the complexity of implementing them, and the large constants hidden by the O notation.

Checking whether two matrices were multiplied correctly can be done in $O(n^2)$ time with high probability.

Fast matrix multiplication can be extended (with small overhead) to Boolean matrices, even though Boolean addition (logical *or*) does not have an inverse operation of Boolean subtraction. We shall show two different ways of doing this.

Adjacency matrices of graphs are square Boolean matrices. Matrix multiplication for these matrices has a natural interpretation in terms of reachability information for graphs. We shall show how this can be used to find triangles in a graph, and how it can be used so as to compute the *transitive closure*.

Remark: On November 29, 14:30, Uri Zwick will talk about "All pairs shortest paths using fast matrix multiplication" in the CS seminar.

Homework:

1. Let a and b be two n-digit numbers. Naive multiplication takes $O(n^2)$ digit multiplications, and a comparable number of digit additions (the exact number depends on *carry* effects). Show a recursive algorithm, based on breaking the numbers in two (high order digits and low order digits) that uses roughly $O(n^{\log 3}) \simeq O(n^{1.585})$ digit multiplications, and a comparable number of digit additions.

2. The naive algorithm for checking whether a graph has a clique of size 9 takes time proportional to $\binom{n}{9} = \Theta(n^9)$. Show how fast matrix multiplication can be used to obtain a substantially improved running time. What running time do you get as a function of ω (where ω is the exponent for fast matrix multiplication)?