Algorithms – handout 5

Permanents and determinants

June 16, 2014

Given an order n matrix A, its *permanent* is $per(A) = \sum_\sigma \prod_{i=1}^n a_{i\sigma(i)}$ where σ ranges over all permutations on n elements. Recall that the determinant of a matrix is $det(A) = \sum_\sigma (-1)^\sigma \prod_{i=1}^n a_{i\sigma(i)}$ where $(-1)^\sigma$ is $+1$ for even permutations and -1 for odd permutations.

The determinant can be computed in polynomial time by gaussian elimination, and in time n^ω by fast matrix multiplication (a topic that will be reviewed in class). On the other hand, there is no polynomial time algorithm known for computing the permanent. In fact, Valiant showed that the permanent is complete for the complexity class $\#P$, which makes computing it as difficult as computing the number of solutions of NP-complete problems (such as SAT, Valiant's reduction was from Hamiltonicity).

For 0/1 matrices, the matrix A can be thought of as the adjacency matrix of a bipartite graph (technically, A is an of-diagonal block of the adjacency matrix), and then the permanent counts the number of perfect matchings. Computing the permanent of integer matrices can be reduced in polynomial time to that of computing the permanent of 0/1 matrices.

We shall see Ryser's formula for computing the permanent, Lovasz's approach to determining if a bipartite graph has a perfect matching, and Kirchoff's tree-matrix theorem for counting spanning trees. There is a very good exposition of this theorem by Mark Jerrum in Chapter 1 in the lecture notes in http://homepages.inf.ed.ac.uk/mrj/pubs.html. The same exposition shows Kasteleyn's polynomial time algorithm for counting the number of perfect matchings in planar graphs.

Homework. Hand in by June 30.

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 running time of roughly $O(n^{3\omega})$ (where ω is the exponent for fast matrix multiplication). It is an open question whether any faster algorithm exists.