Selected Topics in Algorithms
K. Mehlhorn
Exercise 5

Summer 2009
We will discuss this exercise sheet on June 12th and June 19th.

Horton's Algorithm Horton suggested the following algorithm.

$B := \emptyset$
for an edge $e = uv$ and a node z, let $C_{z,e} = p_{zu} + e + p_{vz}$.
sort the nm candidate cycles $C_{z,e}$ in order of increasing weight
for all candidate cycles C (in order of increasing weight) **do**
  if C is independent of B **then**
    add C to B
  end if
end for

The crucial step is the test for independence. Assume $B = \{C_1, \ldots, C_k\}$. The *span* of B is the set of linear combinations of the cycles in B, i.e.,

$$span(B) = \left\{ C; C = \sum_{1 \le i \le k} x_i C_i \text{ for some } x_i \right\} = Ax \ ,$$

where A is the $m \times k$ cycle matrix corresponding to B and x is a k-vector.

1. A *column operation* is the operation of subtracting a multiple of some column of A from another column of A. Show that a column operation does not change the span of A.

2. An $m \times k$ matrix is in *canonical form* if it contains a $k \times k$ identity matrix. Show that A can be transformed by a sequence of column operations into a canonical matrix A'.

3. Assume that such a canonical A' is available. What is the cost of testing whether $C \in span(A')$?

4. Assume that $C \notin span(A')$. Let A'' be the $m \times (k+1)$ matrix obtained from A' by adding C as an additional column. What is the cost of bringing A'' into canonical form?

Remark: we obtain different kinds of cycle bases depending on the field k over which the independence test is carried out.

Fast Matrix Multiplication

1. The natural method for multiplying two $n \times n$ matrices multiplies each row of the first matrix with each column of the second matrix. Each such multiplication requires n multiplications and $n-1$ additions in the base field. How many multiplications and additions are needed altogether.

2. Strassen showed that two 2×2 matrices can be multiplied with 7 multiplications and 18 additions of field elements. Believe this for the moment and derive a recursive algorithm for multiplying $n \times n$ matrices. How many field operations does the method require? Hint: The correct answer is $O(n^{\log 7})$.

3. Find out how Strassen did it. Either look it up in Wikipedia or a text book or try to discover it yourself. If you try to discover it yourself, recall Karatsuba's method for multiplication of long integers.

4. Assume $q \leq \min(p,q)$. How fast can you multiply a $p \times q$ by a $q \times r$ matrix?

Verifying that a Matrix is Nonsingular Let A be a square matrix with integral entries and determinant $D = \det A$.

1. Let p be a prime that does not divide D. What can you say about the determinant of A, when you compute it modulo p?

2. Show that there are at most $\log D$ distinct primes that divide D.

3. Let P be a set of at least $2 \log D$ distinct primes. Consider the following algorithm.

   choose $p \in P$ at random.
   compute the determinant of A in $\mathbb{Z}_p$, where $\mathbb{Z}_p$ is the field of integers modulo p.
   declare A nonsingular if the determinant is nonzero.

   Show: If A is singular, the algorithm will never declare A non-singular. If A is nonsingular, the algorithm will declare A nonsingular with probability at least $1/2$.

4. Assume now that A has entries in $\{0, +1, -1\}$. Give an upper bound U for D.

5. Gaussian elimination determines the determinant of a $n \times n$ matrix with $O(n^3)$ arithmetic operations. How many bits may be required for representing D in the worst-case? Numbers with L bits can certainly be multiplied and added in time $O(L^2)$. Can you derive from this a statement about the bit-complexity of Gaussian elimination, i.e., its complexity when bit-operations instead of arithmetic operations are counted.

6. Use your upper bound from item 4 and let P be the set of the $2 \log U$ smallest primes. Give an upper bound on the largest prime in P. You may want to search for "prime number theorem" in the web.

7. Derive from the previous item a bound on the bit-complexity of computing the determinant of A module p for a prime $p \in P$.