6.890 HOMEWORK 1 **Due: 9/28**
Submit by email to `6890mit@gmail.com` Handed out: 9/14

Instructions: Everyone needs to submit their own write-up. If you work together with other students, indicate their names on your write-up.

Below use the following definitions:

A matrix A is *lower triangular* if $A[i,j] = 0$ whenever $j > i$. A matrix A is *upper triangular* if $A[i,j] = 0$ whenever $j < i$.

A LU-decomposition of an $n \times n$ matrix A are two $n \times n$ matrices L and U such that L is lower triangular, U is upper triangular, and $A = LU$.

A LUP decomposition of an $m \times p$ matrix A consists of three matrices L, U, P where $A = LUP$ and L is $m \times m$ lower triangular, U is $m \times p$ upper triangular, and P is a $p \times p$ permutation matrix.

Problem 0 [4pts]

Show that if one can detect a triangle in an n node graph in $T(n)$ time such that there is some $\varepsilon > 0$ for which $T(N) \geq (1+\varepsilon)T(N/2)$ for all N, then one can also find a triangle in an n node graph in $O(T(n))$ time.

Problem 1 [4pts]

Let $\ell, \kappa \geq 2$ be such that $\kappa \leq \ell$ and let $f(\ell, \kappa) = \sum_{i=1}^{\kappa} \binom{\ell}{i}$. Suppose that $\kappa \log \ell \leq O(\log n)$.

Show that given an $n \times n$ Boolean matrix A, one can preprocess A in $O\left(n^2/(\ell \log n) f(\ell, \kappa)\right)$ time, so that one can then multiply A by any n length Boolean vector v on t nonzeros in $O\left((n/\log n)(n/\ell + t/\kappa)\right)$ time.

Use the assumption from class that if we have a look-up table T indexed by $O(\log n)$ bit integers, and if each slot $T[i]$ stores an $O(\log n)$ bit integer, then $T[i]$ can be looked up in $O(1)$ time.

Problem 2 [4pts]

For each of the following problems, show that an $O(n^c)$ time algorithm for it for any $c \geq 2$ would imply an $O(n^c)$ time algorithm for multiplying two $n \times n$ matrices.

(a) Given an $n \times n$ matrix A, compute $A \cdot A$.

(b) Given a lower triangular matrix A and an upper triangular matrix B, compute $A \cdot B$.

Problem 3 [4pts]

Suppose that there is an $O(n^c)$ time algorithm (for some $c \geq 2$) that given two $n \times n$ *binary* matrices A and B, can compute their product AB (over the integers) in $O(n^c)$ time. Show that then there is an $O(b^2 n^c)$ time algorithm that can compute the product of two $n \times n$ matrices with entries in $\{0, \ldots, 2^b - 1\}$.

Problem 4 [2pts]

Does every invertible matrix have a LU-decomposition? If so, provide a proof. If not, give an example of such a matrix and prove that no LU-decomposition exists for it.

Problem 5 [6pts]

Suppose that one can compute the LUP decomposition of an *invertible* $n \times n$ matrix in $O(n^c)$ time for $c \in [2, \omega]$, and in particular that every invertible matrix has a LUP decomposition. Then show the following:

(a) Given an invertible $n \times n$ matrix A and an $n \times 1$ vector b, one can solve the linear system $Ax = b$ in $O(n^c)$ time.

(b) Given a (possibly non-invertible) $n \times n$ matrix A, one can compute its determinant Δ in $O(n^\omega \log(|\Delta| + 2))$ time. For this problem assume that the entries of the matrices L, U and P given by the LUP algorithm are integers, and that multiplying two b-bit integers takes $O(b)$ time. A randomized algorithm that succeeds with high probability is perfectly acceptable.

Problem 6: BONUS [4pts]

(you do not have to solve this problem but it is worth some bonus points)

Show that if one can multiply $n \times n$ matrices in $O(n^\omega)$ time, then one can compute the LUP decomposition of an $n \times n$ invertible matrix in $O(n^\omega)$ time.

Comment: I'll give full points if you can show this for computing the LUP decomposition of an s.p.d. matrix.