Note: Your TA may not get to all the problems. This is totally fine, the discussion worksheets are not designed to be finished in an hour. The discussion worksheet is also a resource you can use to practice, reinforce, and build upon concepts discussed in lecture, readings, and the homework.

1 Coffee Shops

A rectangular city is divided into a grid of $m \times n$ blocks. You would like to set up coffee shops so that for every block in the city, either there is a coffee shop within the block or there is one in a neighboring block. (There are up to 4 neighboring blocks for every block). It costs r_{ij} to rent space for a coffee shop in block ij.

Write an integer linear program to determine which blocks to set up the coffee shops at, so as to minimize the total rental costs.

(a) What are your variables, and what do they mean?

(b) What is the objective function?

(c) What are the constraints?

(d) Solving the non-integer version of the linear program gets you a real-valued solution. How would you round the LP solution to obtain an integer solution to the problem? Describe the algorithm in at most two sentences.

(e) What is the approximation ratio obtained by your algorithm?

(f) Briefly justify the approximation ratio.

2 Local Search for Max Cut

Sometimes, local search algorithms can give good approximations to NP-hard problems. In the Max-Cut problem, we have an unweighted graph $G(V, E)$ and we want to find a cut (S, T) with as many edges "crossing" the cut (i.e. with one endpoint in each of S, T) as possible. One local search algorithm is as follows: Start with any cut, and while there is some vertex $v \in S$ such that more edges cross $(S - v, T + v)$ than (S, T) (or some $v \in T$ such that more edges cross $(S + v, T - v)$ than (S, T)), move v to the other side of the cut. Note that when we move v from S to T, v must have more neighbors in S than T.

(a) Give an upper bound on the number of iterations this algorithm can run for (i.e. the total number of times we move a vertex).

(b) Show that when this algorithm terminates, it finds a cut where at least half the edges in the graph cross the cut.

3 Modular Arithmetic

(a) Show that if $a_1 \equiv b_1 \pmod{n}$ and $a_2 \equiv b_2 \pmod{n}$, then $a_1 + a_2 \equiv b_1 + b_2 \pmod{n}$.

(b) Show that for integers a_1, b_1, a_2, b_2, and n, if $a_1 \equiv b_1 \pmod{n}$ and $a_2 \equiv b_2 \pmod{n}$, then $a_1 \cdot a_2 \equiv b_1 \cdot b_2 \pmod{n}$.

(c) What is the last digit (i.e., the least significant digit) of 3^{4001}?

4 Random Prime Generation

Lagrange's prime number theorem states that as N increases, the number of primes less than N is $\Theta(N/\log(N))$.

An important primitive in cryptography is the ability to sample a prime number uniformly at random. Assume we can verify that an n-bit number is a prime in $O(n^2)$ time. Briefly describe a randomized algorithm that samples a prime uniformly at random from all primes in $\{2, 3, \ldots, 2^n - 1\}$ with expected runtime polynomial in n. What is the expected runtime of your algorithm?

(Recall that if we have a coin that lands heads with probability p, the expected number of coin flips we make before we see the first heads is $1/p$.)