MATH 482, Spring 2013 - Homework 5

Assigned Monday 11/04. Due Monday 11/11.

For this homework, solve four of the following five problems, but definitely complete problems 4 and 5. Each is worth 5 points. Problem 1 has a point breakdown for the parts, should you choose to do that problem.

Complete AT MOST four problems. If you complete all five, then your top score will be dropped!

1. Determine which of the matrices below are (i) unimodular, (ii) totally unimodular, or (iii) neither. Be sure to explain your answer.

$$\begin{bmatrix} 1 & -1 & -1 & 0 \\ -1 & 0 & 0 & 1 \\ 0 & 1 & 0 & -1 \\ 0 & 0 & 1 & 0 \end{bmatrix} \qquad \begin{bmatrix} 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 \\ 1 & 1 & 0 & 1 \end{bmatrix} \qquad \begin{bmatrix} 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 1 & 1 \\ 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 1 & 0 \end{bmatrix}$$

a. (*1.5pts*) **b.** (*1.5pts*) **c.** (*2pts*)

2. An (n, k, λ, μ) *strongly regular graph* is an undirected graph with vertex set $\{v_1, \dots, v_n\}$ where every vertex is incident to k edges and for every pair v_i, v_j:

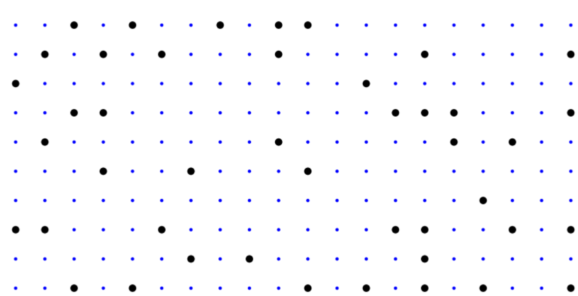
- If $v_i v_j$ is an edge, then v_i and v_j have exactly λ common neighbors.
- If $v_i v_j$ is not an edge, then v_i and v_j have exactly μ common neighbors.

For arbitrary n, k, λ, μ, construct an integer program encoding the constraints of an (n, k, λ, μ) strongly regular graph. Prove that the feasible integer solutions are in bijection with the (n, k, λ, μ) strongly regular graphs on vertex set $v_1, \dots, v_n$. (For this last part, we consider the graphs to be *labeled* and so do not worry about isomorphism.)

3. Solve the following integer linear program using branch-and-bound. Use the graphical method to solve each linear program relaxation. Plot your branch-and-bound cuts in the x_1, x_2-plane.

$$\begin{array}{rrcrcl} \text{min} & x_1 & + & x_2 & = & z \\ \text{subject to} & x_1 & + & 2x_2 & \geq & 2 \\ & -x_1 & + & 4x_2 & \leq & 3 \\ & x_1, & & x_2 & \geq & 0, \text{ integer} \end{array}$$

4. (*Assigned!*) Using the following set of points on the integer grid, compute a heuristic TSP using the Nearest Neighbor heuristic, then locally improve it using 2-switches until it is locally optimal.



A larger version is on the next page. Feel free to print multiple copies and work directly on those copies.

5. (*Assigned!*) Using the above set of points on the integer grid, compute a heuristic TSP using the Farthest Insertion heuristic, then locally improve it using 2-switches until it is locally optimal.