Caleb Logemann
MATH 566 Discrete Optimization
Homework 9

1. Implement Directed Minimum Mean Cycle in Sage. Before implementing the algorithm, show that it is possible to slightly modify the algorithm. Instead of adding an extra vertex s and edges from s to all other vertices, it is possible to simply assign $F_0(v) = 0$ for all $v \in V$ at the beginning. This avoids the hassle with adding an extra vertex. But it requires an argument that the algorithm is still correct.

   The following function performs the minimum mean cost cycle algorithm as described in the notes with the modification described above.

```
def minimumMeanCostCycle(graph):
    n = graph.order()
    p = {v:[None]*(n+1) for v in graph.vertices()}
    F = {v:[0]+[oo]*n for v in graph.vertices()}
    for k in range(1,n+1):
        for e in graph.edges():
            u = e[0]
            v = e[1]
            c = e[2]
            if F[v][k] > F[u][k-1] + c:
                F[v][k] = F[u][k-1] + c
                p[v][k] = u

    # If all are infinite then min will be infinite
    if min([F[v][n] for v in graph.vertices()]) == oo:
        print 'This graph is acyclic'
        return (None, None)

    m = {v:oo for v in graph.vertices()}
    for v in graph.vertices():
        if F[v][n] != oo:
            m[v] = max([(F[v][n] - F[v][k])/(n-k) for k in range(n-1)])

    x = min(m, key=m.get)
    mu = m[x]

    # create cycle
    cycle = DiGraph()
    cycle.add_vertex(x)
    k = n
    v = x
    while p[v][k] != x:
        u = v
        v = p[v][k]
        cycle.add_vertex(v)
        cycle.add_edge(v, u, graph.edge_label(v, u))
        k-= 1
    cycle.add_edge(x, v, graph.edge_label(x, v))
    return (cycle, mu)
```

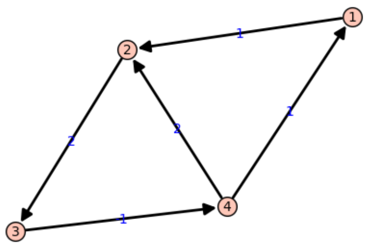
This script runs this algorithm on 3 test graphs.

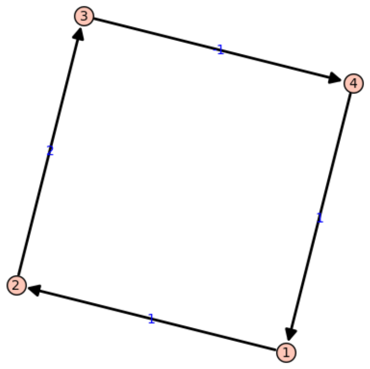
```
load('minimumMeanCostCycle.sage')
g = DiGraph([(1,2,1), (2,3,2), (3,4,-1), (4,1,1), (4,2,2) ])
g.show(edge_labels=True)
(cycle,mu) = minimumMeanCostCycle(g)
if cycle != None:
    print "Mu=",mu
    cycle.show(edge_labels=True)


g = DiGraph([(1,2,3), (1,3,2), (2,4,-1), (3,4,1), (4,5,0), (5,6,1),
    ↪ (5,7,2), (6,8,1), (7,8,3), (8,1,2) ])
g.show(edge_labels=True)
(cycle,mu) = minimumMeanCostCycle(g)
if cycle != None:
    print "Mu=",mu
    cycle.show(edge_labels=True)


g = DiGraph([(1,2,1), (2,3,1), (3,4,1), (1,4,1), (4,5,2) ])
g.show(edge_labels=True)
(cycle,mu) = minimumMeanCostCycle(g)
if cycle != None:
    print "Mu=",mu
    cycle.show(edge_labels=True)
```

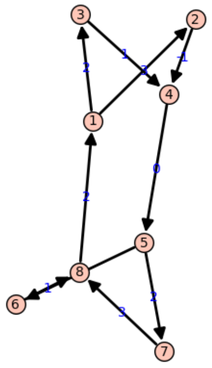
The output of this script is shown below.

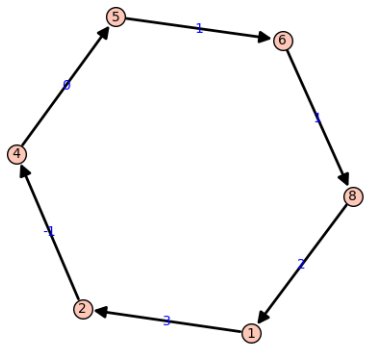
```
Launched png viewer for Graphics object consisting of 15 graphics primitives
Mu= 3/4
Launched png viewer for Graphics object consisting of 13 graphics primitives
Launched png viewer for Graphics object consisting of 29 graphics primitives
Mu= 1
Launched png viewer for Graphics object consisting of 19 graphics primitives
Launched png viewer for Graphics object consisting of 16 graphics primitives
This graph is acyclic
```

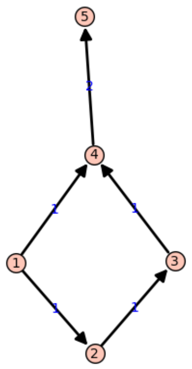
The scipt also produces the following images.











2. Show that in integer program, it is possible to express the following constraint:

$$x \in [100, 200] \cup [300, 400]$$

in other words

$$100 \leq x \leq 200 \text{ or } 300 \leq x \leq 400$$

How to express the constraint *without* using *or*?

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

$$\underset{\textbf{a.}}{\begin{pmatrix} 1 & -1 & -1 & 0 \\ -1 & 0 & 0 & 1 \\ 0 & 1 & 0 & -1 \\ 0 & 0 & 1 & 0 \end{pmatrix}} \qquad \underset{\textbf{b.}}{\begin{pmatrix} 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 \\ 1 & 1 & 0 & 1 \end{pmatrix}} \qquad \underset{\textbf{c.}}{\begin{pmatrix} 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 1 & 1 \\ 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 1 & 0 \end{pmatrix}}$$

(i) Since these are all square matrices, they will be unimodular if their determinant is ± 1. So I compute the determinants of these 3 matrices. First I will compute the determinant of (a).

$$\det(a) = \begin{vmatrix} 1 & -1 & -1 & 0 \\ -1 & 0 & 0 & 1 \\ 0 & 1 & 0 & -1 \\ 0 & 0 & 1 & 0 \end{vmatrix}$$

Expand along first column

$$\det(a) = 1 \times \begin{vmatrix} 0 & 0 & 1 \\ 1 & 0 & -1 \\ 0 & 1 & 0 \end{vmatrix} - (-1) \times \begin{vmatrix} -1 & -1 & 0 \\ 1 & 0 & -1 \\ 0 & 1 & 0 \end{vmatrix}$$

Expand along the first column for the first determinant and along the last column for the second determinant

$$\begin{aligned} &= 1 \times \left(-1 \times \begin{vmatrix} 0 & 1 \\ 1 & 0 \end{vmatrix}\right) - (-1) \times \left(1 \times \begin{vmatrix} -1 & -1 \\ 0 & 1 \end{vmatrix}\right) \\ &= 1 \times (-1 \times -1) - (-1) \times (1 \times -1) \\ &= 1 - 1 \\ &= 0 \end{aligned}$$

Since the determinant of (a) is 0 this matrix is not unimodular.
Second I will compute the determinant of (b).

$$\det(b) = \begin{vmatrix} 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 \\ 1 & 1 & 0 & 1 \end{vmatrix}$$

I will first expand along the first row.

$$\det(b) = 1 \times \begin{vmatrix} 1 & 0 & 0 \\ 0 & 1 & 1 \\ 1 & 0 & 1 \end{vmatrix} + 1 \times \begin{vmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 1 & 1 & 1 \end{vmatrix}$$

I will expand the first determinant along the first row and the second determinant along the first column

$$\begin{aligned} \det(b) &= 1 \times \left(1 \times \begin{vmatrix} 1 & 1 \\ 0 & 1 \end{vmatrix}\right) + 1 \times \left(1 \times \begin{vmatrix} 1 & 0 \\ 0 & 1 \end{vmatrix}\right) \\ &= 1 \times (1 \times 1) + 1 \times (1 \times 1) \\ &= 1 + 1 \\ &= 2 \end{aligned}$$

Since the determinant of (b) is 2 this matrix is not unimodular.
Lastly I will compute the determinant of (c).

$$\det(c) = \begin{vmatrix} 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 1 & 1 \\ 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 1 & 0 \end{vmatrix}$$

First I will expand along the first column

$$\det(c) = -1 \times \begin{vmatrix} 1 & 0 & 0 & 0 \\ 1 & 0 & 1 & 1 \\ 0 & 1 & 0 & 1 \\ 0 & 1 & 1 & 0 \end{vmatrix}$$

Next I will expand along the first row

$$\det(c) = -1 \times \left(1 \times \begin{vmatrix} 0 & 1 & 1 \\ 1 & 0 & 1 \\ 1 & 1 & 0 \end{vmatrix} \right)$$

Expanding along the first column gives

$$\begin{aligned} \det(c) &= -1 \times \left(1 \times \left(-1 \times \begin{vmatrix} 1 & 1 \\ 1 & 0 \end{vmatrix} + 1 \times \begin{vmatrix} 1 & 1 \\ 0 & 1 \end{vmatrix} \right) \right) \\ &= -1 \times (1 \times (-1 \times -1 + 1 \times 1)) \\ &= -1 \times (1 \times (1 + 1)) \\ &= -1 \times (1 \times 2) \\ &= -1 \times 2 \\ &= -2 \end{aligned}$$

Since the determinant of (c) is -2 this matrix is not unimodular.

In summary none of the matrices are unimodular.

(ii) In order for a matrix to be totally unimodular every square submatrix must have determinant -1, 0, or 1. Note that since matrices (b) and (c) don't have a determinant -1, 0, or 1 when considered as a whole matrix they cannot be totally unimodular. Matrix (a) which has determinant 0 can potentially be totally unimodular. In fact we see that each column has exactly one 1 and one -1, so by a theorem in the notes (a) is totally unimodular.

(iii) We have shown that (a) is totally unimodular but not unimodular. However (b) and (c) are neither unimodular nor totally unimodular.

4. Show that $A \in \mathbb{Z}^{m \times n}$ is totally unimodular iff $[A\ I]$ is unimodular (where I is $m \times m$ unit matrix).

5. Find a unimodular matrix A, that is not totally unimodular.

   Consider the matrix

$$A = \begin{matrix} 9 & 7 \\ 5 & 4 \end{matrix}$$

   The matrix A is unimodular because $A \in \mathbb{Z}^{2\times 2}$ and $\det(A) = 9 \times 4 - 5 \times 7 = 36 - 35 = 1$. However A is not totally unimodular because not all of the entries of A are -1, 0, 1.