L4.py

```
from mlib import *

from numpy import arange
from pylab import plot, axis, xlabel, ylabel, grid, show, savefig, cla, rc,
legend, figure


REPEAT = 100
STEP = 5


def sort_matrix(matrix):
    for shift in range(len(matrix)):
        wm = matrix[shift:, shift:]
        min_id = wm.sum(axis=1).argmin() + shift

        buf = matrix.copy()
        matrix[shift] = matrix[min_id]
        matrix[min_id] = buf[shift]
        # print ("1:", min_id)
        # print (matrix)

        wm = matrix[shift:, shift:]
        max_col = [(id, s.sum()) for id, s in
                   filter(lambda m: m[1][:, 0] == 1, enumerate(wm.transpose()))]
        # print ("checking ", max_col)
        if len(max_col) > 0:
            max_id = max(max_col, key=lambda x: x[1])[0] + shift
            buf = matrix.copy()
            matrix[:, shift] = matrix[:, max_id]
            matrix[:, max_id] = buf[:, shift]
            # print ("2:", max_id)
            # print (matrix)

    return matrix


def find_conflict(matrix):
    matrix = sort_matrix(matrix)
    N = len(matrix)

    for i in range(1, N):
        k = i
        l = N-k
        # print (matrix[:k, N-l:])
        if not matrix[:k, N-l:].any():
            return True
    return False


def xplot(y, title_t, index, cl=True, ax=None, sv=True):
    color = 'bgrcmykw'
    rc('font', **{'family': 'serif'})
    rc('text', usetex=True)
    rc('text.latex', unicode=True)
    rc('text.latex', preamble='\\usepackage[utf8]{inputenc}')
    rc('text.latex', preamble='\\usepackage[russian]{babel}')
    x = arange(0.0, 1.0+STEP/100, STEP/100)
    print (x, y)
    print (len(x), len(y))
```

```
    if ax:
        ax.plot(x, y, '%s.-' % color[index], label=title_t)
    grid()
    ylabel("вероятность")
    xlabel("связность")
    if sv:
        savefig('%s.png' % title_t.replace(" ", "_"))
    if cl:
        cla()


def test():
    data = []
    for index, N in enumerate(range(10, 31, 5)):
        lst = []
        for pc in range(0, 101, STEP):
            print("N=%d pc=%d%%" % (N, pc))
            matrixes = generate(N, pc/100., REPEAT)
            # print (list((mpermutations(matrix))))
            conf = list(map(find_conflict, matrixes))
            # print (conf)
            lst.append(len(list(filter(None, conf))) / len(conf))
        print (lst)
        data.append(lst)

    fig = figure()
    ax = fig.add_subplot(111)
    for index, l in enumerate(data):
        xplot(l, str(index*5+10), index, True, ax)

    fig = figure()
    ax = fig.add_subplot(111)
    for index, l in enumerate(data):
        xplot(l, str(index*5+10), index, False, ax, False)
    ax.legend()
    savefig('all.png')


if __name__ == '__main__':
    test()
```

mlib.pyx

```
import random
import numpy


def generate(N, pc, count):
    for x in range(count):
        matrix = [[0 for x in range(N)] for x in range(N)]
        count = int(N*N*pc)
        for i in random.sample(range(N*N), count):
            matrix[i % N][i//N] = 1
        yield numpy.matrix(matrix)
```