import networkx as n

import matplotlib.pyplot as plt

import random as r

def get\_c(g):

color=list()

for var in g.nodes():

if g.node[var]['type']==1:

color.append('red')

elif g.node[var]['type']==2:

color.append('blue')

else :

color.append('white')

return color

def disp(g):

col=get\_c(g)

n.draw(g,dt,labels=labels,node\_color=col)

plt.show()

def internal\_neigh(u,v):

return [(u,v-1),(u,v+1),(u-1,v),(u+1,v+1),(u-1,v-1),(u+1,v-1),(u-1,v+1)]

def boundary\_neigh(u,v):

if(u==0 and v==0):

return [(u+1,v),(u,v+1),(u+1,v+1)]

if(u==size-1 and v==0):

return [(u,v+1),(u-1,v),(u-1,v+1)]

if(u==size-1 and v==size-1):

return [(u,v-1),(u-1,v),(u-1,v-1)]

if(u==0 and v==size-1):

return [(u,v-1),(u+1,v),(u+1,v-1)]

if(v==0):

return [(u+1,0),(u-1,0),(u,v+1),(u+1,v+1),(u-1,v+1)]

if(v==size-1):

return [(u+1,v),(u-1,v),(u+1,v-1),(u-1,v-1),(u,v-1)]

if(u==0):

return [(u,v+1),(u,v-1),(u+1,v),(u+1,v-1),(u+1,v+1)]

if(u==size-1):

return [(u,v+1),(u,v-1),(u-1,v),(u-1,v+1),(u-1,v+1)]

def nodetype(g):

bn=list()

intl=list()

for n,d in g.nodes():

if (n==0 or n==size-1) or (d==0 or d==size-1):

bn.append([n,d])

else :

intl.append([n,d])

return bn,intl

def pr(g):

t=list()

for n in g.nodes():

clr=g.node[n]['type']

if clr==0:

t.append(n)

return t

def fetch(g):

unstable=list()

t=4

for u,v in g.nodes():

if g.node[(u,v)]['type']==0:

continue

else :

count=0

get\_type=g.node[(u,v)]['type']

if [u,v] in inter:

data=internal\_neigh(u,v)

else:

data=boundary\_neigh(u,v)

for ele in data:

if g.node[ele]['type']==get\_type:

count=count+1

if count<t:

unstable.append((u,v))

return unstable

def satisfy(g):

node\_move=r.choice(unsatisfied\_nodes)

node\_pos=r.choice(empty)

g.node[node\_pos]['type']=g.node[node\_move]['type']

g.node[node\_move]['type']=0

labels[node\_move],labels[node\_pos]=labels[node\_pos],labels[node\_move]

size=30

g=n.grid\_2d\_graph(size,size)

dt=dict((n,n) for n in g.nodes())

labels=dict(((i,j),(i\*10+j+1)) for (i,j) in dt.values())

for i,j in g.nodes():

if (i<=size-2 and j<=size-2):

g.add\_edge((i,j),(i+1,j+1))

for i,j in g.nodes():

if (i<=size-2 and j>=1):

g.add\_edge((i,j),(i+1,j-1))

for i in g.nodes():

g.node[i]['type']=r.randint(0,2)

bound,inter=nodetype(g)

print 'Initial figure'

disp(g)

ele =0

for ele in range(1000):

empty=pr(g)

unsatisfied\_nodes=fetch(g)

if len(unsatisfied\_nodes)==0:

break

satisfy(g)

print 'Final figure'

disp(g)

print 'No. of iterations involved: ',ele