Tarea 2

March 5, 2018

Objetivo

Realizar un programa que pueda generar Grafos No Dirigidos, Grafos Dirigidos y Grafos Ponderados

Codigo

Contenido del archivo `main.py`:

```
from Grafo import Grafo

nodos = 15

GrafoNDir = Grafo()
GrafoNDir.generar(nodos)
GrafoNDir.imprimir("nodosNoDirigido.txt")
GrafoNDir.conectar(0.25)
GrafoNDir.graficar(0,"NoDirigido.tex")
print("unset arrow")

GrafoDir = Grafo()
GrafoDir.generar(nodos)
GrafoDir.imprimir("nodosDirigido.txt")
GrafoDir.conectar(0.25)
GrafoDir.graficar(1,"Dirigido.tex")
print("unset arrow")


GrafoPond = Grafo()
GrafoPond.generar(nodos)
GrafoPond.imprimir("nodosPonderado.txt")
GrafoPond.conectar(0.25)
GrafoPond.graficar(2,"Ponderado.tex")
print("unset arrow")
```

Contenido del archivo `Grafo.py`:

```
from random import random, choice
import math

def cabecera():
    print("set terminal epslatex size 3.5,2.62 color colortext")

    print('set xrange [-0.1:1.1]')
    print('set yrange [-0.1:1.1]')
    print('set size square')
    print('set key off')
    print('unset colorbox')

def inf(destino, tipo):
    if tipo == 0:
        print("plot '{:s}' using 1:2:0 with points pt 7 lc 'black'".format(destino))
    else:
        print("plot '{:s}' using 1:2:(rand(0)) with points pt 7 lc palette".format(dest
class Grafo:
    def __init__(self):
        self.n = None
        self.x = dict()
        self.y = dict()
        self.E = []
        self.destino = None

    def generar(self, orden):
        self.n = orden
        for nodo in range(self.n):
            self.x[nodo] = random()
            self.y[nodo] = random()

    def imprimir(self, direccion):
        self.destino = direccion
        with open(self.destino, "w") as archivo:
            for nodo in range(self.n):
                print(self.x[nodo], self.y[nodo], file=archivo)

    def conectar(self, dist):
        for nodo in range(self.n - 1):
            for nodo2 in range(nodo + 1, self.n):
                distancia = math.sqrt(pow(self.x[nodo2]-self.x[nodo],2)+pow(self.y[nodo
                if distancia < dist:
                    self.E.append((nodo,nodo2))
                if distancia > (1-(dist/10)):
                    self.E.append((nodo2,nodo))
```

```
def graficar (self, tipo,nombreArchivo):
    cabecera()
    print("set output '"+nombreArchivo+"'")
    numArrow = 1
    nodoidx = 0
    for(v, w) in self.E:
        x1 = self.x[v]
        x2 = self.x[w]
        y1 = self.y[v]
        y2 = self.y[w]
        peso = int(random()*5 + 1)
        if tipo == 0:
            print("set arrow {:d} from {:f}, {:f} to {:f}, {:f} nohead lw 1 lc rgb
        if tipo == 1:
            print("set arrow {:d} from {:f}, {:f} to {:f}, {:f} head filled lw 1 lc
        if tipo == 2:
            print("set arrow {:d} from {:f}, {:f} to {:f}, {:f} head filled lw {:d}
        numArrow+=1
        nodoidx+=1
    inf(self.destino,tipo)
```

Resultados

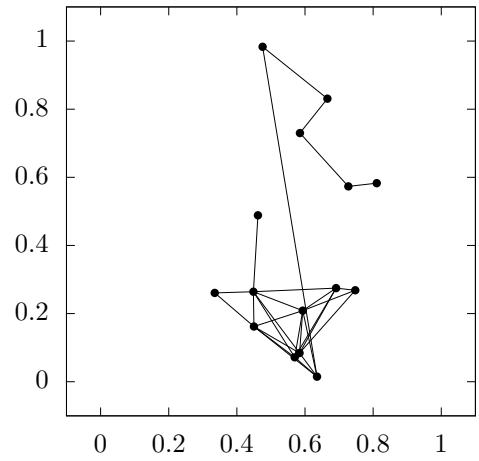

Figure 1: Grafo no dirigido

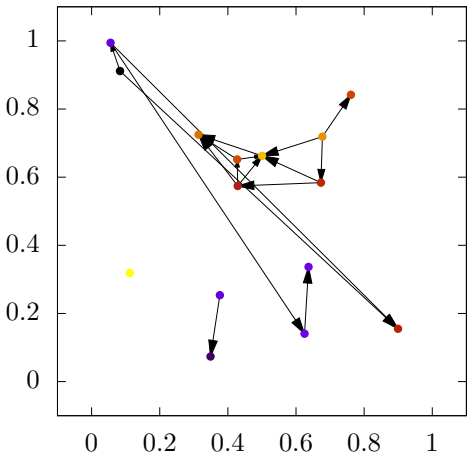

Figure 2: Grafo dirigido

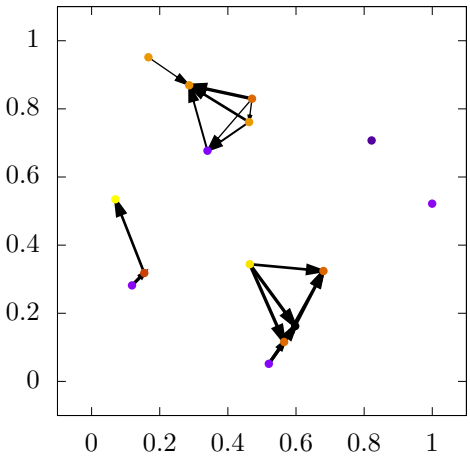

Figure 3: Grafo ponderado