Tutorial Igraph with Python

Technical Report · October 2017

DOI: 10.13140/RG.2.2.15335.14248

CITATIONS
0

READS
11,081

1 author:

Josimar Chire
University of São Paulo
46 PUBLICATIONS **47** CITATIONS

SEE PROFILE

Tutorial Igraph with Python

Josimar Edinson Chire Saire
jecs89@gmail.com

October 13, 2017

Abstract

Complex Networks is a field with good growing during the last two decades, the main idea is one graph with many thousands of vertexes and edges with some special properties like free-scale, small worlds and others. Actually, there are many tools available for many languages, during the search about them Igraph was one of the most popular for Python. The review of Igraph tutorial was not enough and the search in code communities started after some results were obtained. The idea of this tutorial is to share the date recompiled.

Contents

1 Introduction

Complex Networks is a topic with recent researching from physics, computer science field but they well-know in other fields like sociology, biology. Python is a language with fast growing during the last decade for it is advantages of performance and the intuitive syntax. Igraph is one of the most popular libraries for Python to work with Complex Network. Linux is a wide-known in the world for it is free philosophy and Ubuntu is one the most famous distributions with GUI friendly to the final user. This tutorial is performed on Ubuntu 16 with python 2.7.

2 Installation

If you are using at least Ubuntu 14 then you have python 2.7 installed or just can install python for your own distribution go to Python official website. If you want to check your version just run:

```
python --version
```

I suggest to install **virtualenv** to have your own environment of python to test everything with no risks. Before of this installation we need to install **python-pip**.

```
sudo apt-get install python-pip
```

Virtualenv installation

```
pip install virtualenv
```

if you are running on server, you could use:

```
pip install --user virtualenv
```

Now, you need to create your virtual environment:

```
virtualenv project
```

You could choose your version if you are using other python version:

```
virtualenv -p /usr/bin/python2.7 project
```

We need to activate our virtual environment:

```
source project-/bin/activate (check the name of your folder :D)
```

Then, your terminal should show something similar to Fig.1.

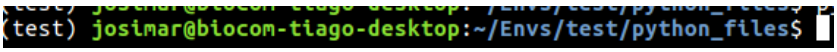

Figure 1: Virtual Environment with virtualenv

Now we need install **python-numpy**, **python-igraph**, **python-cairo** be careful with names **igraph** is other different thing.

3 Igraph

Never forget the import line:

```
from igraph import *
```

3.1 Graph creation from text file

Let's create how graph using a text file and save with the name **net.txt**, you can download:

```
https://github.com/jecs89/Tutorials/tree/master/Python/Igraph
```

```
# load data into a graph
g = igraph.Graph.Read_Ncol('net.txt')

# plot graph
igraph.plot(g)
```

You should have a similar result to Fig.2.

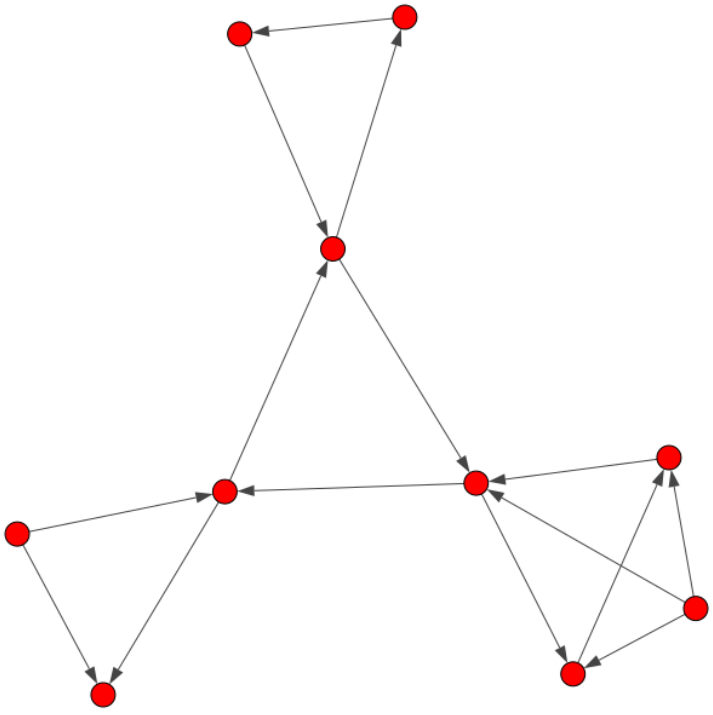

Figure 2: Example 1

3.2 Creation of Graph using functions

Graph with tree structure in Fig.3.

```
g = Graph.Tree(100,10) #first parameter=number of vertexes,
second=number of edges
plot(g)
```

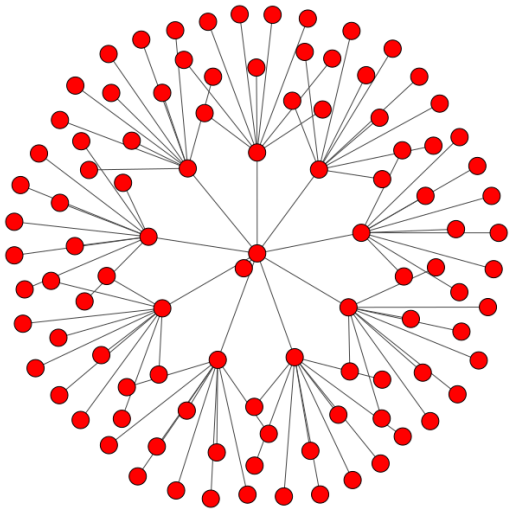

Figure 3: Example 2

Let's change this 100 for 50 then one figure like Fig.4.

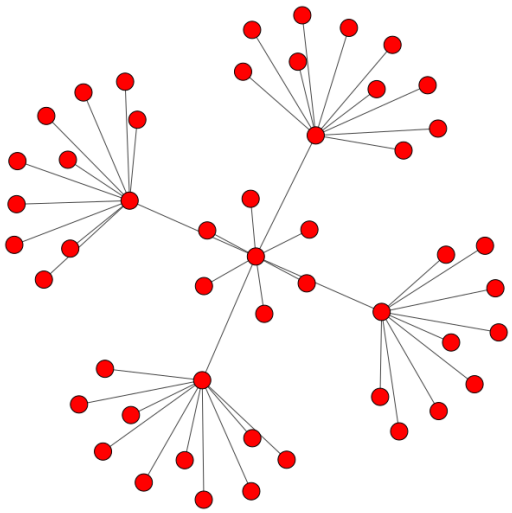

Figure 4: Modification of Example 2

Let's create other with a stochastic generator:

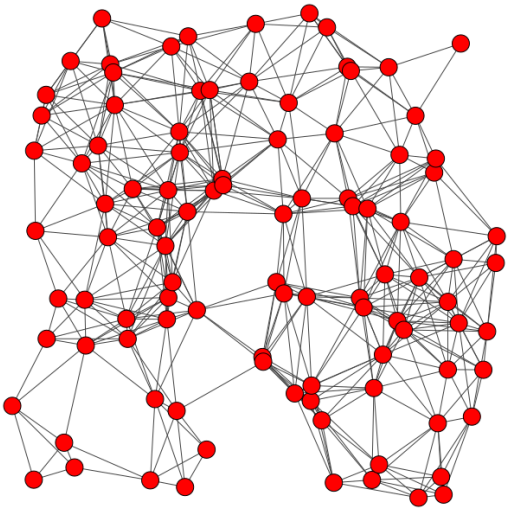

Figure 5: Example 3

If you want to plot more complex graph using gml format or you are wondering about networkx just check:
https://plot.ly/python/igraph-networkx-comparison/
https://networkdata.ics.uci.edu/data.php?id=11

Then you can get something like Fig. 6

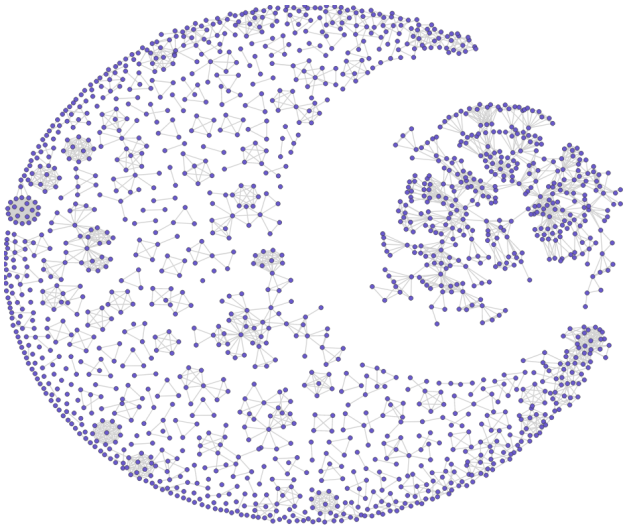

Figure 6: Example 4

3.3 Saving Graphs

After the creation of our graph, it will be necessary to store them, we can use write svg function. Add this line to your previous example to check it.

```
g.write_svg('first.svg', vertex_size=20)
```

Let's try other example:

```
from igraph import *
g = Graph.Tree(7, 2)
layout = g.layout_reingold_tilford(mode="in", root=[0])
g.vs['size'] = ['60']
g.vs['color'] = ['green', 'red', 'blue', 'yellow']
g.vs['label'] = ['Matthew', 'John', 'Luke', 'Mark',
'Paul', 'James', 'Eliah', 'Emanuel']
g.write_svg('first.svg', layout=layout, vertex_size=20)
plot(g, layout=layout,bbox = (500, 300), margin = 50)
```

You should get a picture like Fig.7.

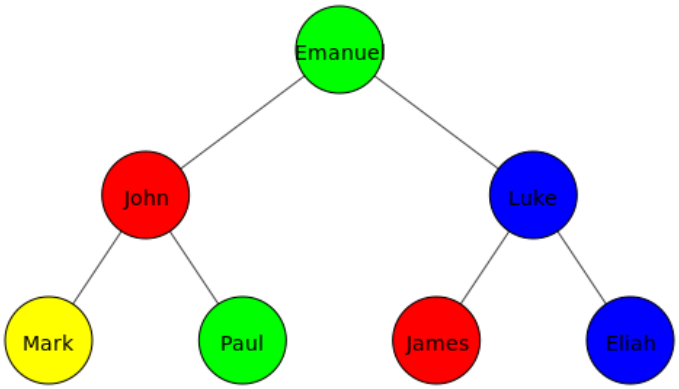

Figure 7: Example 5

Now, we are going to plot very famous artificial complex network and get some features of the graph .

```
from igraph import *
g = Graph.Erdos_Renyi(1000,0.0008,directed=False, loops=False)
array = g.degree(vertices=None, mode=ALL, loops=True)
print 'Numeros de vertices isolados (Grau 0)  = %d '% array.count(0)
print 'Numeros de vertices com grau 1  =  %d' % array.count(1)
print 'Numeros de vertices com grau 2  =  %d' % array.count(2)
print 'Maior grau  =  %d ' % g.maxdegree(vertices=None, mode=ALL, loops=False)
```

```
components = g.components()
print 'O numero de componentes  =  %d' % len(components)
print 'Tamanho da maior componente  =  %d' % max(components.sizes())
print 'Tamanho da menor component  =  %d'  % min(components.sizes())
g.write_svg('teste.svg', width=1000, height=1000, vertex_size=5, font_size=10)
```

You should get a picture like Fig.8.

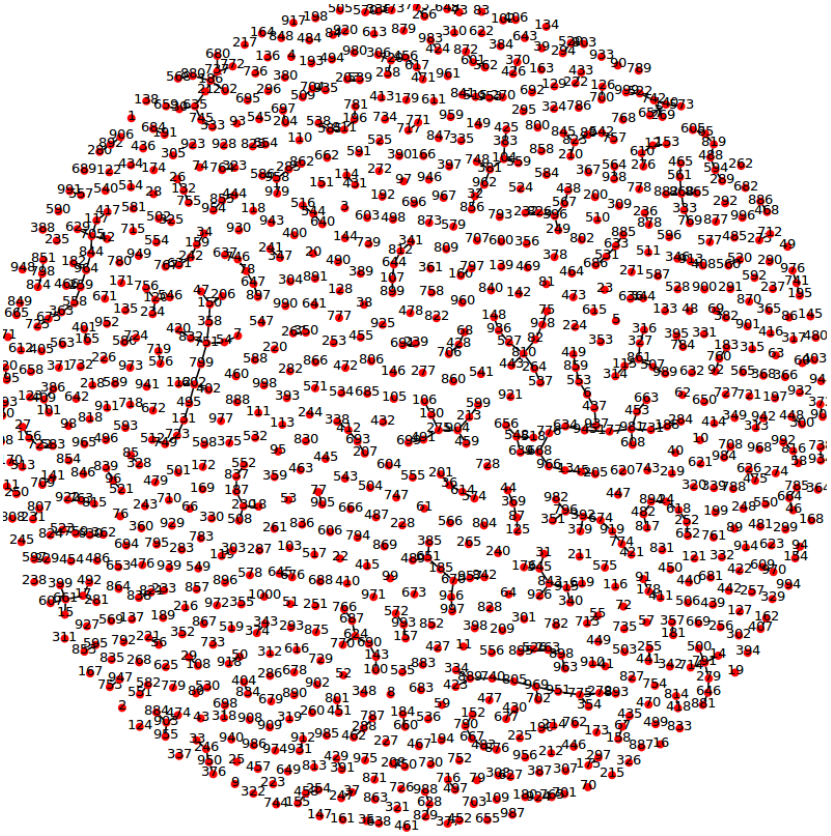

Figure 8: Example 6

3.4 Plot features

After the creation and save of the graphs, we can change color, visualization and other features. If you notice, the preliminar examples, there was a line:

```
g.vs['color'] = ['green', 'red', 'blue', 'yellow']
```

Then when you plot, the color is assigned. We can try this way to set color:

```
igraph.plot(g,vertex_color='blue')
```

You should get a picture like Fig.9.

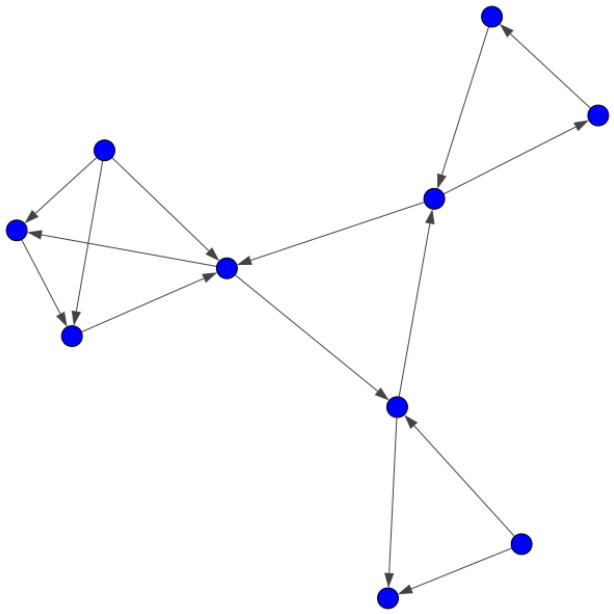

Figure 9: Example 7

We can try using layout for different kind of visualization:

```
from igraph import *
g = Graph.Erdos_Renyi(100,0.008,directed=False, loops=False)
layout = 'tree'
g.write_svg('teste.svg', layout=layout, width=1000,
height=1000, vertex_size=5, font_size=10)
```

You should get a picture like Fig.10.

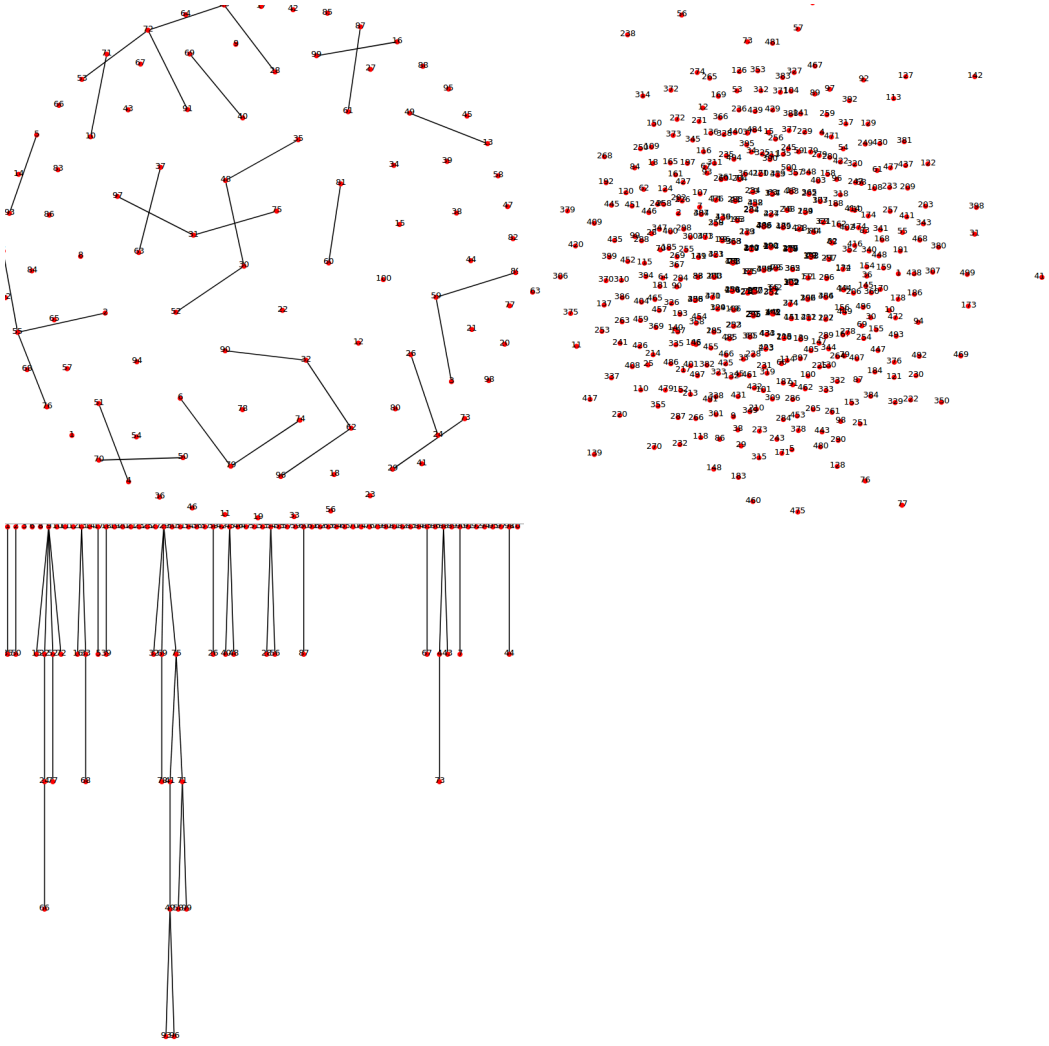

Figure 10: Example 8 with circular, fr, tree layout

4 Conclusion

Igraph is a useful tool to create, manage and visualize graphs. Igraph is available for python, R, C and you can get it for different platforms. If you have more doubts about igraph, you can check Igraph site.

5 Repository

If you want to get the source code or learn other things, just check:

`https://github.com/jecs89/Tutorials/tree/master/Python/Igraph`