GitHub Gist

Sign up for a GitHub account

Sign in

Instantly share code, notes, and snippets.

Create a gist now



rbnvrw / community_detection.py

Last active 2 months ago

python-igraph example

```
    community_detection.py
```

```
from igraph import *
     import numpy as np
     # Create the graph
     vertices = [i for i in range(7)]
     edges = [(0,2),(0,1),(0,3),(1,0),(1,2),(1,3),(2,0),(2,1),(2,3),(3,0),(3,1),(3,2),(2,4),(4,5),(4,6),(5,4),(5,6),(6,4),(6,5)]
     g = Graph(vertex attrs={"label":vertices}, edges=edges, directed=True)
8
9
     visual style = {}
10
11
     # Scale vertices based on degree
12
     outdegree = g.outdegree()
     visual_style["vertex_size"] = [x/max(outdegree)*25+50 for x in outdegree]
14
15
     # Set bbox and margin
16
17
     visual style["bbox"] = (800,800)
     visual_style["margin"] = 100
18
19
     # Define colors used for outdegree visualization
20
```

```
21
     colours = ['#fecc5c', '#a31a1c']
     # Order vertices in bins based on outdegree
23
     bins = np.linspace(0, max(outdegree), len(colours))
24
     digitized_degrees = np.digitize(outdegree, bins)
25
26
27
     # Set colors according to bins
     g.vs["color"] = [colours[x-1] for x in digitized degrees]
28
29
     # Also color the edges
30
31
     for ind, color in enumerate(g.vs["color"]):
32
             edges = g.es.select( source=ind)
             edges["color"] = [color]
33
34
     # Don't curve the edges
     visual style["edge curved"] = False
36
37
     # Community detection
38
     communities = g.community edge betweenness(directed=True)
39
     clusters = communities.as clustering()
40
41
     # Set edge weights based on communities
42
     weights = {v: len(c) for c in clusters for v in c}
43
     g.es["weight"] = [weights[e.tuple[0]] + weights[e.tuple[1]] for e in g.es]
45
     # Choose the layout
46
     N = len(vertices)
47
     visual style["layout"] = g.layout fruchterman reingold(weights=g.es["weight"], maxiter=1000, area=N**3, repulserad=N**3)
48
49
     # Plot the graph
50
     plot(g, **visual style)
51
```

```
simple-graph.py

from igraph import *
2
```

```
vertices = ["one", "two", "three"]

deges = [(0,2),(2,1),(0,1)]

g = Graph(vertex_attrs={"label": vertices}, edges=edges, directed=True)

plot(g)
```

styling_graph.py

```
from igraph import *
     import numpy as np
     # Create the graph
     vertices = ["one", "two", "three"]
     edges = [(0,2),(2,1),(0,1)]
6
8
     g = Graph(vertex attrs={"label": vertices}, edges=edges, directed=True)
9
     visual style = {}
10
11
     # Scale vertices based on degree
12
     outdegree = g.outdegree()
13
     visual style["vertex size"] = [x/max(outdegree)*50+110 for x in outdegree]
14
15
     # Set bbox and margin
16
     visual_style["bbox"] = (800,800)
17
     visual style["margin"] = 100
18
19
     # Define colors used for outdegree visualization
20
     colours = ['#fecc5c', '#a31a1c']
21
22
     # Order vertices in bins based on outdegree
23
     bins = np.linspace(0, max(outdegree), len(colours))
24
     digitized_degrees = np.digitize(outdegree, bins)
25
26
```

```
# Set colors according to bins
27
     g.vs["color"] = [colours[x-1] for x in digitized_degrees]
28
29
     # Also color the edges
30
     for ind, color in enumerate(g.vs["color"]):
31
             edges = g.es.select(_source=ind)
32
             edges["color"] = [color]
33
34
     # Don't curve the edges
     visual style["edge curved"] = False
37
     # Plot the graph
38
     plot(g, **visual_style)
39
```



farhankhwaja commented on Oct 6

I am getting a "plot not available error". I tried installing cairo/pycairo but I am getting a conflict i.e. python3.5 pycairo are conflicting. Is there a way to plot the graph anyother way? Or how can I res0lve the issue.

Sign up for free

to join this conversation on GitHub. Already have an account? Sign in to comment