Name : Arul Kumar ARK

Roll No. : 225229103

```
                                Lab : 06 : Detecting communities in large networks using
networkx package
```

In [1]:

```
import random
from numpy import random as nprand
random.seed(123)
nprand.seed(123)
```

In [2]:

```
from matplotlib import pyplot as plt
%matplotlib inline
plt.rcParams.update(plt.rcParamsDefault)
plt.rcParams.update({'figure.figsize': (15, 10)})
```

In [3]:

```
import networkx as nx
import networkx.algorithms.community as nxcom
```

Exercise : 1

In [4]:

```
G_karate = nx.karate_club_graph()
```

In [5]:

```
G_karate
```

Out[5]:

```
<networkx.classes.graph.Graph at 0x203f59fae10>
```

In [6]:

```
communities = sorted(nxcom.greedy_modularity_communities(G_karate), key=len, reverse=Tru
```

In [7]:

```
communities
```

Out[7]:

```
[frozenset({8,
            14,
            15,
            18,
            20,
            22,
            23,
            24,
            25,
            26,
            27,
            28,
            29,
            30,
            31,
            32,
            33}),
 frozenset({1, 2, 3, 7, 9, 12, 13, 17, 21}),
 frozenset({0, 4, 5, 6, 10, 11, 16, 19})]
```

In [8]:

```
len(communities)
```

Out[8]:

```
3
```

In [9]:

```
def set_node_community(G, communities):
    for c, v_c in enumerate(communities):
        for v in v_c:
            G.nodes[v]['community'] = c + 1
```

In [10]:

```
def set_edge_community(G):
    for v, w, in G.edges:
        if G.nodes[v]['community'] == G.nodes[w]['community']:
            G.edges[v, w]['community'] = G.nodes[v]['community']
        else:
            G.edges[v, w]['community'] = 0
```

In [11]:

```
def get_color(i, r_off=1, g_off=1, b_off=1):
    r0, g0, b0 = 0, 0, 0
    n = 16
    low, high = 0.1, 0.9
    span = high - low
    r = low + span * (((i + r_off) * 3) % n) / (n - 1)
    g = low + span * (((i + g_off) * 5) % n) / (n - 1)
    b = low + span * (((i + b_off) * 7) % n) / (n - 1)
    return (r, g, b)
```

In [12]:

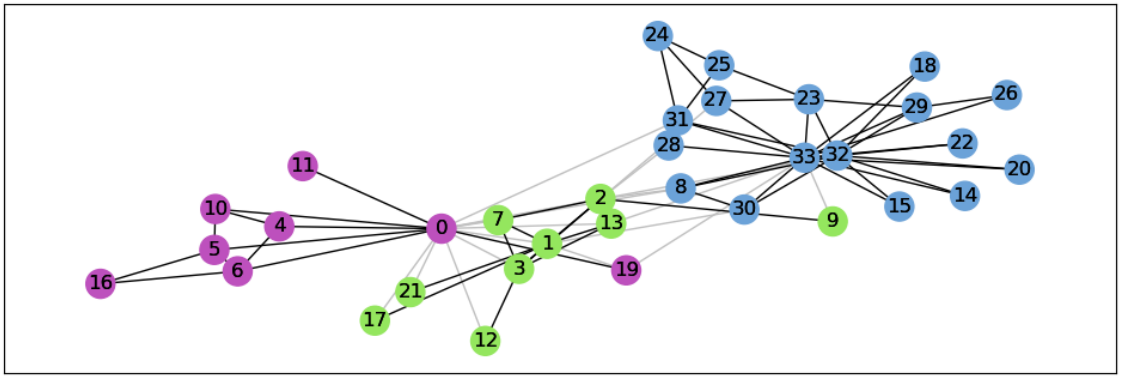
```
set_node_community(G_karate, communities)
set_edge_community(G_karate)
node_color = [get_color(G_karate.nodes[v]['community']) for v in G_karate.nodes]
```

In [13]:

```
external = [(v, w) for v, w in G_karate.edges if G_karate.edges[v, w]['community'] == 0]
internal = [(v, w) for v, w in G_karate.edges if G_karate.edges[v, w]['community'] > 0]
internal_color = ['black' for e in internal]
```

In [14]:

```
karate_pos = nx.spring_layout(G_karate)
plt.rcParams.update({'figure.figsize': (12, 4)})
nx.draw_networkx(G_karate,pos=karate_pos,node_size=0,edgelist=external,edge_color="silve
nx.draw_networkx(G_karate,pos=karate_pos,node_color=node_color,edgelist=internal,edge_cc
plt.show()
```



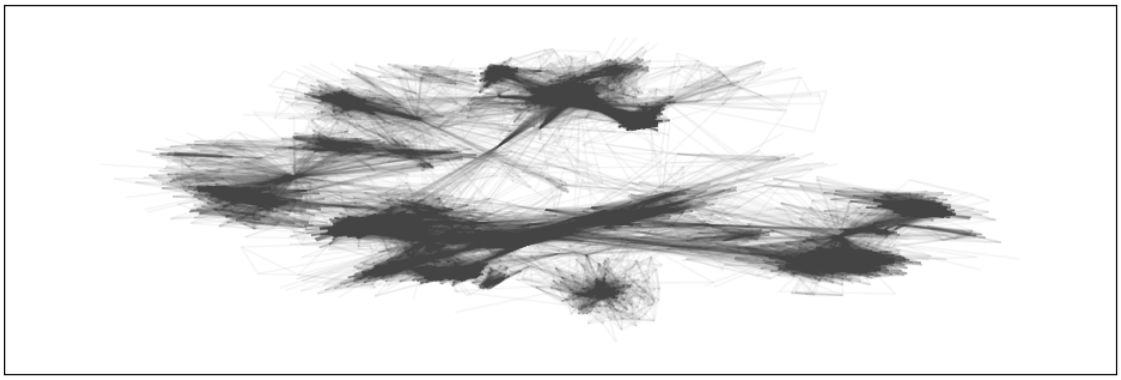
Exercise : 2

In [15]:

```
G_social = nx.read_edgelist('facebook_combined (1).txt')
```

In [16]:

```
pos = nx.spring_layout(G_social, k=0.1)
plt.rcParams.update({'figure.figsize': (12, 4)})
nx.draw_networkx(G_social,pos=pos,node_size=0,edge_color="#444444",alpha=0.05,with_label
plt.show()
```



In [17]:

```
communities = sorted(nxcom.greedy_modularity_communities(G_social), key=len, reverse=Tru
```

In [18]:

```
len(communities)
```

Out[18]:

```
16
```

In [19]:

```
plt.rcParams.update(plt.rcParamsDefault)
plt.rcParams.update({'figure.figsize': (12, 4)})
plt.style.use('dark_background')
```

In [20]:

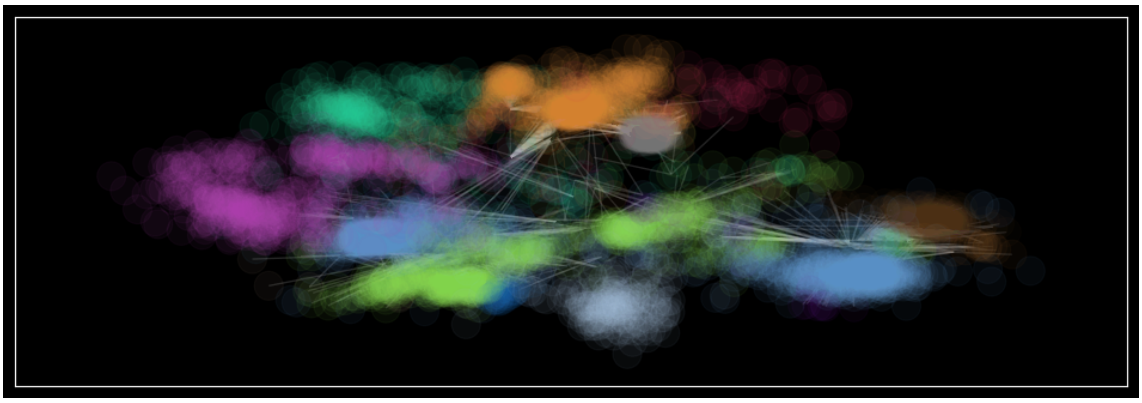
```
set_node_community(G_social, communities)
set_edge_community(G_social)
```

In [21]:

```
external = [(v, w) for v, w in G_social.edges if G_social.edges[v, w]['community'] == 0]
internal = [(v, w) for v, w in G_social.edges if G_social.edges[v, w]['community'] > 0]
internal_color = ["black" for e in internal]
node_color = [get_color(G_social.nodes[v]['community']) for v in G_social.nodes]
```

In [22]:

```
nx.draw_networkx(G_social,pos=pos,node_size=0,edgelist=external,edge_color="silver",node
nx.draw_networkx(G_social, pos=pos,edgelist=internal,edge_color=internal_color,node_colo
plt.show()
```



GIRVAN-NEWMAN COMMUNITY DETECTION

In [23]:

```
result = nxcom.girvan_newman(G_karate)
communities = next(result)
```

In [24]:

```
communities
```

Out[24]:

```
({0, 1, 3, 4, 5, 6, 7, 10, 11, 12, 13, 16, 17, 19, 21},
 {2, 8, 9, 14, 15, 18, 20, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31, 32, 3
3})
```

In [25]:

```
len(communities)
```

Out[25]:

```
2
```

In [33]:

```
plt.rcParams.update(plt.rcParamsDefault)
plt.rcParams.update({'figure.figsize': (12, 4)})
```

In [34]:

```
set_node_community(G_karate, communities)
set_edge_community(G_karate)
```

In [35]:

```
node_color = [get_color(G_karate.nodes[v]['community']) for v in G_karate.nodes]
```

In [36]:

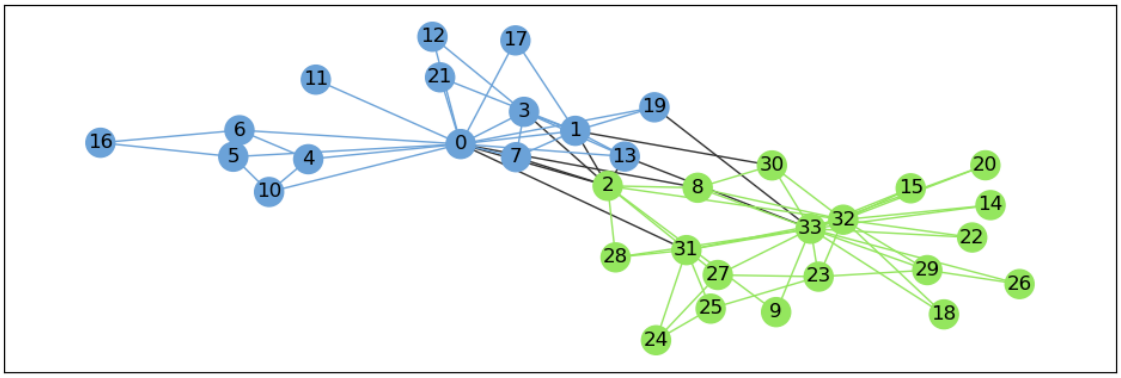
```
external = [(v, w) for v, w in G_karate.edges if G_karate.edges[v, w]['community'] == 0]
internal = [(v, w) for v, w in G_karate.edges if G_karate.edges[v, w]['community'] > 0]
internal_color = [get_color(G_karate.edges[e]['community']) for e in internal]
karate_pos = nx.spring_layout(G_karate)
```

In [37]:

```
nx.draw_networkx(G_karate, pos=karate_pos, node_size=0,edgelist=external, edge_color="#3
nx.draw_networkx(G_karate, pos=karate_pos, node_color=node_color,edgelist=internal, edge
```

In [38]:

```
plt.show()
```



In [41]:

```
G_facebook = nx.read_edgelist('facebook_combined (1).txt')
```

In [43]:

```
result_facebook = nxcom.girvan_newman(G_facebook)
communities_facebook = next(result_facebook)
```

```
y", line 2, in <module>
    communities_facebook = next(result_facebook)
                           ^^^^^^^^^^^^^^^^^^^^^
  File "C:\Users\arulk\anaconda3\Lib\site-packages\networkx\algorithms
\community\centrality.py", line 147, in girvan_newman
    yield _without_most_central_edges(g, most_valuable_edge)
          ^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^
  File "C:\Users\arulk\anaconda3\Lib\site-packages\networkx\algorithms
\community\centrality.py", line 166, in _without_most_central_edges
    edge = most_valuable_edge(G)
           ^^^^^^^^^^^^^^^^^^^^^
  File "C:\Users\arulk\anaconda3\Lib\site-packages\networkx\algorithms
\community\centrality.py", line 138, in most_valuable_edge
    betweenness = nx.edge_betweenness_centrality(G)
                  ^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^
  File "<class 'networkx.utils.decorators.argmap'> compilation 29", li
ne 4, in argmap_edge_betweenness_centrality_26
    import inspect
           ^^^^^^^
  File "C:\Users\arulk\anaconda3\Lib\site-packages\networkx\algorithms
```

In [ ]:

```
set_node_community(G_facebook, communities)
set_edge_community(G_facebook
```

In [ ]:

```
node_color = [get_color(G_facebook.nodes[v]['community']) for v in G_facebook.nodes]
```

In [ ]:

```
external = [(v, w) for v, w in G_facebook.edges if G_facebook.edges[v, w]['community'] =
internal = [(v, w) for v, w in G_facebook.edges if G_facebook.edges[v, w]['community'] >
internal_color = [get_color(G_facebook.edges[e]['community']) for e in internal]
```

In [ ]:

```
karate_pos = nx.spring_layout(G_facebook)
```

In [ ]:

```
plt.rcParams.update(plt.rcParamsDefault)
plt.rcParams.update({'figure.figsize': (15, 10)})
```

In [ ]:

```
nx.draw_networkx(G_facebook, pos=karate_pos, node_size=0,edgelist=external, edge_color='
nx.draw_networkx(G_facebook, pos=karate_pos, node_color=node_color,edgelist=internal, ec
plt.show()
```

K-CORES

In [47]:

```
G_core_30 = nx.k_core(G_social, 30)
G_core_60 = nx.k_core(G_social, 60)
```

In [48]:

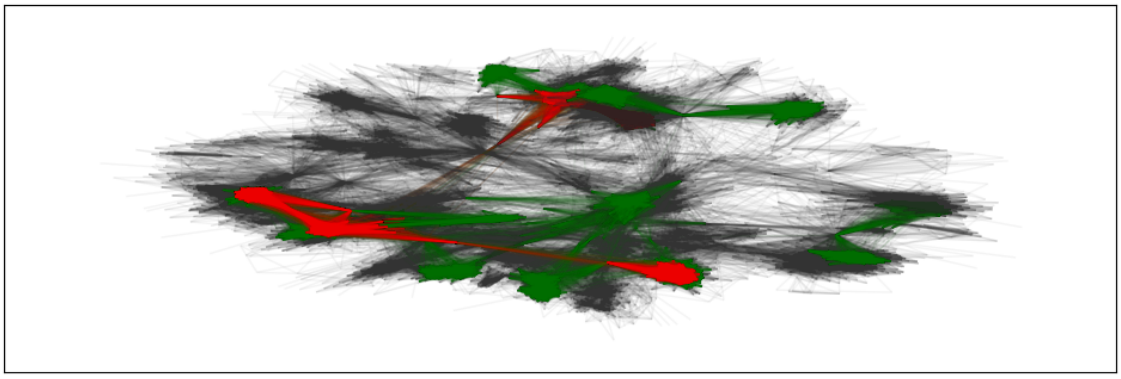
```
plt.rcParams.update(plt.rcParamsDefault)
plt.rcParams.update({'figure.figsize': (15, 10)})
plt.style.use('dark_background')
pos = nx.spring_layout(G_social, k=0.1)
```

In [49]:

```
nx.draw_networkx(G_social, pos=pos, node_size=0, edge_color="#333333", alpha=0.05, with_
nx.draw_networkx(G_core_30, pos=pos, node_size=0, edge_color="green", alpha=0.05, with_l
nx.draw_networkx(G_core_60, pos=pos, node_size=0, edge_color="red", alpha=0.05, with_lab
```

In [50]:

```
plt.show()
```



CLIQUES

In [52]:

```
plt.rcParams.update(plt.rcParamsDefault)
plt.rcParams.update({'figure.figsize': (15, 10)})
cliques = list(nx.find_cliques(G_karate))
max_clique = max(cliques, key=len)
node_color = [(0.5, 0.5, 0.5) for v in G_karate.nodes()]
for i, v in enumerate(G_karate.nodes()):
    if v in max_clique:
        node_color[i] = (0.5, 0.5, 0.9)
nx.draw_networkx(G_karate, node_color=node_color, pos=karate_pos)
```

In [53]:

```
plt.show()
```