NATIONAL INSTITUTE OF TECHNOLOGY SILCHAR

Cachar, Assam

B.Tech. VIth Sem

Subject Code: CS-321

Subject Name: Social Network Analysis Lab

Submitted By:

Name	: Subhojit Ghimire
Sch. Id.	: 1912160
Branch	: CSE – B

AIM: TO GENERATE OVERLAPPING COMMUNITIES WITH WLC ALGORITHM METHOD AND VISUALIZE THE GENERATED COMMUNITIES USING NETWORKX LIBRARY OR GEPHI. (ALTERNATIVELY, CAN ALSO USE ANY OF THE FOLLOWING TOOLS: MATPLOTLIB, PLOTLY, GGPLOT, SEABORN, BOKEH).

THEORY:

1. **Overlapping Community:** Overlapping community detection is one node having multiple community memberships in the networks.

2. **WLC Algorithm:** It is a local algorithm for overlapping community detection based on clustering coefficient and common neighbour similarity**.**

REAL WORLD NETWORK DATASETS: Zachary's Karate Club, American College Football and Dolphin Social Network.

CODE:

```
import networkx as nx
import time

def modu1(G,N,res):
    m=0
    for  U in res:
        n=len(U);

        S=G.subgraph(U)

        rr=[]
        for kk in res:
            if not kk==U:
                rr.extend(kk)

        ov=list(set(U).intersection(set(rr)))

        sum1= 0
        i=0
        while i<len(U):
            j=i+1
            while j<len(U):
                if U[i] in ov :


                    o=S.degree(U[i])


                    o1=0
                    for ll in res:
                        if U[i] in ll:
                            S1=G.subgraph(ll)
                            o1=o1+S1.degree(U[i])

                    al1=o/o1
                else :
                    al1=1

                if  U[j] in ov  :


                    oo=S.degree(U[j])


                    oo1=0
                    for ll in res:

                        if U[j]in ll:
                            S1=G.subgraph(ll)
                            oo1=oo1+S1.degree(U[j])

                    al2=oo/oo1

                else :

                    al2=1




                #tt=2*cpt


                if G.has_edge(U[j],U[i]) :
                    x=((1-((G.degree(U[i])*G.degree(U[j]))/(2*N)))*al1*al2)
                    sum1= sum1+2*x

                else :


                    sum1= sum1+2*((0-
((G.degree(U[i])*G.degree(U[j]))/(2*N)))*al1*al2)
                j=j+1
            i=i+1
        m=m+sum1
```

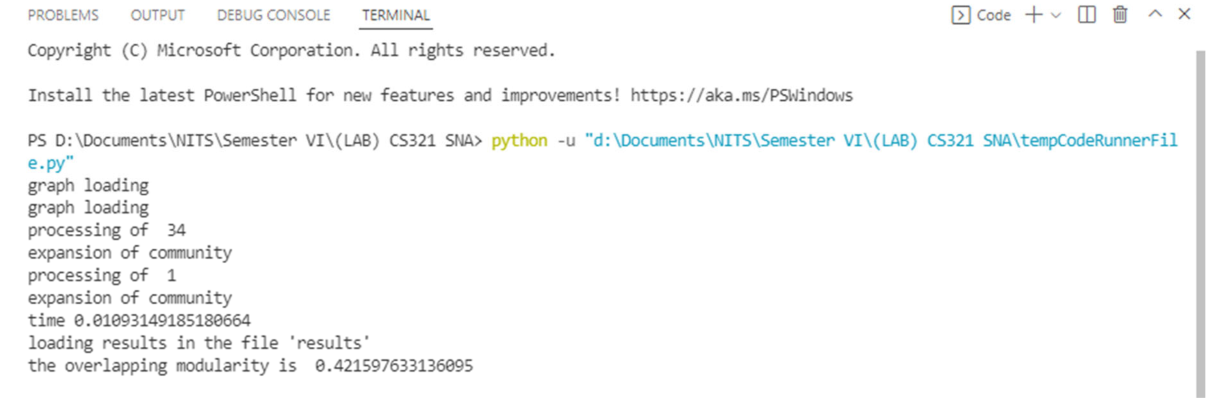
```
    m=m/(2*N)

    return(m)

def WLC(path,sep):

    t=[]
    tri=[]

    print('graph loading')
    G=nx.read_edgelist(path, comments='#', delimiter=sep,
nodetype=int,encoding='utf-8')#txt file

    print('graph loading')


    ns=len(G.nodes())
    N=G.number_of_edges()

    t=[]
    den=nx.density(G)

    re=[]
    res=[]
    res1=[]
    res2=[]
    rr=[]

    w1=[]
    tps1= time.time()
    T11=list(G.nodes())
    i=0
    while i<len(T11):
        cpt1=0
        xx=list (G.neighbors(T11[i]))
        a=len(xx)
        j=0
        while j < a-1:
            j1=j+1
            while j1<a:
                if  G.has_edge(xx[j],xx[j1]):
                   cpt1=cpt1+1
                j1=j1+1
            j=j+1
        if a>1:
            w1.append(2*cpt1/(a*(a-1)))
        else:
            w1.append(0)
        i=i+1

    T=G.nodes()
    while len(T)>0:
        nst=[]
        S=G.subgraph(T)
        for k in T:
            nst.append([S.degree(k),k])

        nst.sort(reverse=True)
        l=nst[0][1]
        print('processing of ',l)
        ini=list(set(S.neighbors(l)))
        ini.append(l)
        n=len(ini)
        n1=len(ini)
        b=True

        while b==True:


            m1=[]
            temp=-1
            for r in ini:

                a=w1[T11.index (r)]
                x=list(S.neighbors(r))
                ww1=0
                ww2=0
                if len(x)>0:
                    for rr1 in x:
                        d1=w1[T11.index (rr1)]
                        d=(d1+len(sorted(nx.common_neighbors(G, r, rr1))))
                        ww1=ww1+d
                        if rr1 in ini:
                            ww2=ww2+d
                    if ww1>0:
                        bl=ww2/ww1
                        if bl<0.5:

                            ini.remove(r)

            n1=len(ini)
            if  n1<n:
                n=n1
                b=True
            else:
                b=False
```

```
b=1
print('expansion of community')
while b==1:
    x=[]
    for k in ini:
        x.extend(G.neighbors(k))
        x=list(set(x)-set(ini))

    n=len(ini)
    m1=[]
    for r in x:

        x1=list(G.neighbors(r))
        ww1=0
        ww2=0
        if len(x1)>0:
            for rr1 in x1:
                d1=w1[T11.index(rr1)]
                d=(d1+len(sorted(nx.common_neighbors(G, r, rr1))))#
                ww1=ww1+d
                if rr1 in ini:
                    ww2=ww2+d
            if ww1>0:
                bl=ww2/ww1

                if bl>=0.4:

                    m1.append(r)



    ini.extend(m1)
    n1=len(ini)
    if n1>n:
        b=1

    else:

            b=0

            break


res.append(ini)
```

```
        rr.extend(ini)
        T=list(set(T)-set(ini))


        if (len(ini)==0):
            T.remove(l)

    tps2= time.time()
    print('time',tps2-tps1)
    m=0
    print("loading results in the file \'results\'")
    fichier = open("results.txt", "w")
    for res1 in res:
        for k in res1:
            fichier.write(str(k))
            fichier.write(' ')
        fichier.write('\n')
    fichier.close()

    m=modu1(G,N,res)
    print("the overlapping modularity is ",m, '\n\n')

graph = nx.read_gml ('karate.gml', label = 'id') # karate club dataset
nx.write_edgelist (graph, 'karateedge.txt', delimiter = ',')
f = 'karateedge.txt'
WLC(f,',')

graph = nx.read_gml ('football.gml', label = 'id') # football club dataset
nx.write_edgelist (graph, 'footballedge.txt', delimiter = ',')
f = 'footballedge.txt'
WLC(f,',')

graph = nx.read_gml ('dolphins.gml', label = 'id') # dolphin social network
dataset
nx.write_edgelist (graph, 'dolphinsedge.txt', delimiter = ',')
f = 'dolphinsedge.txt'
WLC(f,',')
```

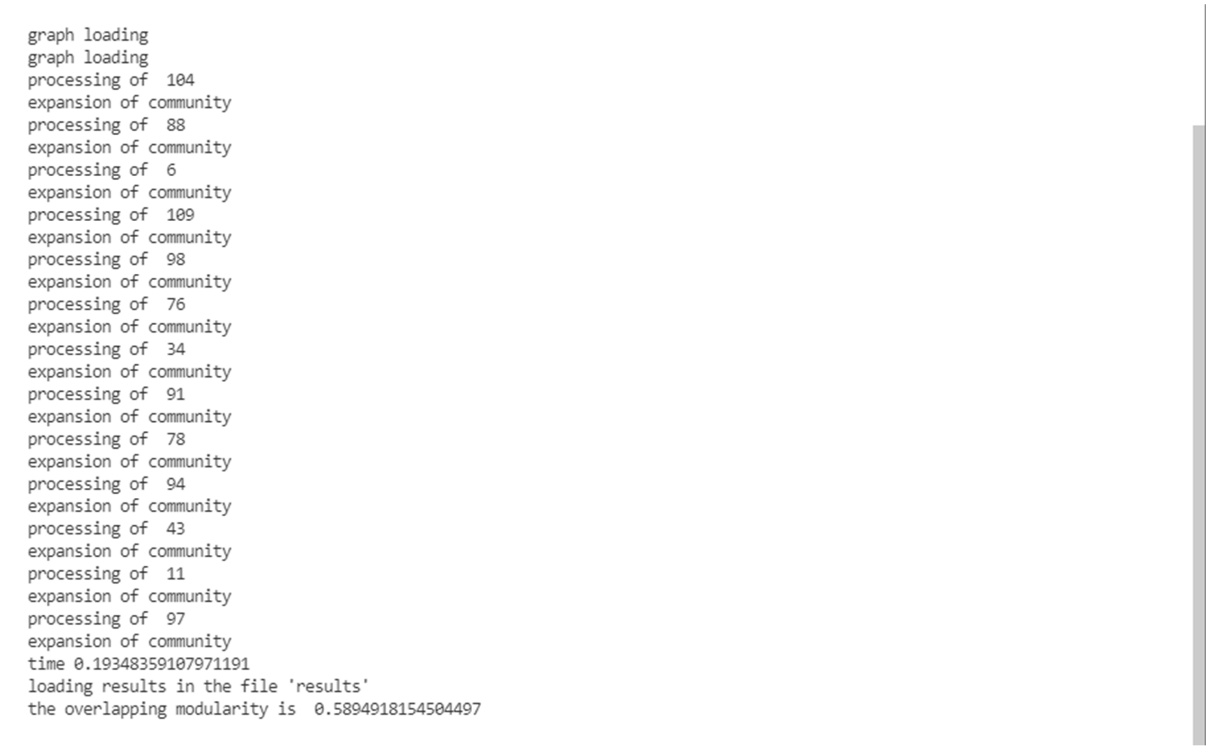
OUTPUT AND OBSERVATIONS (NETWORKX LIBRARY):

// KARATE CLUB



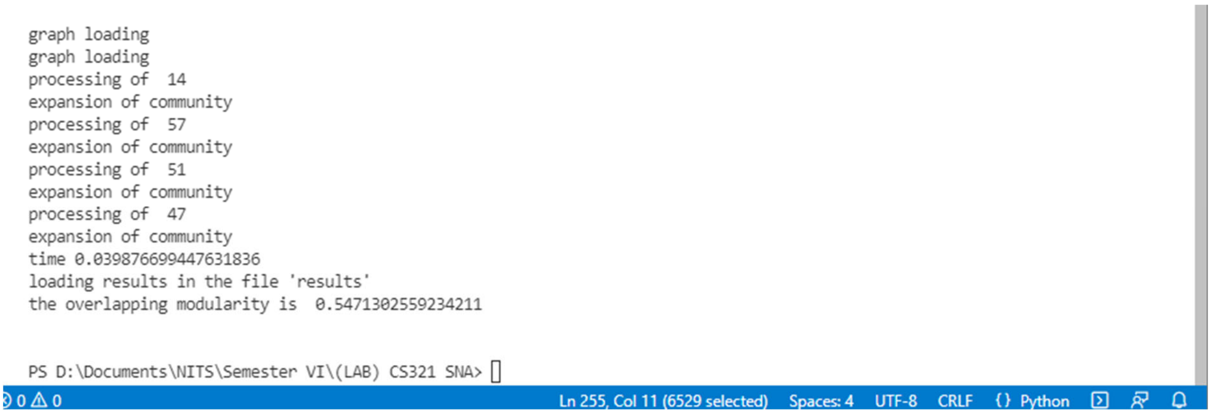
```
Copyright (C) Microsoft Corporation. All rights reserved.

Install the latest PowerShell for new features and improvements! https://aka.ms/PSWindows

PS D:\Documents\NITS\Semester VI\(LAB) CS321 SNA> python -u "d:\Documents\NITS\Semester VI\(LAB) CS321 SNA\tempCodeRunnerFil
e.py"
graph loading
graph loading
processing of  34
expansion of community
processing of  1
expansion of community
time 0.01093149185180664
loading results in the file 'results'
the overlapping modularity is  0.421597633136095
```

// FOOTBALL CLUB



```
graph loading
graph loading
processing of  104
expansion of community
processing of  88
expansion of community
processing of  6
expansion of community
processing of  109
expansion of community
processing of  98
expansion of community
processing of  76
expansion of community
processing of  34
expansion of community
processing of  91
expansion of community
processing of  78
expansion of community
processing of  94
expansion of community
processing of  43
expansion of community
processing of  11
expansion of community
processing of  97
expansion of community
time 0.19348359107971191
loading results in the file 'results'
the overlapping modularity is  0.5894918154504497
```

// DOLPHINS NETWORK



```
graph loading
graph loading
processing of  14
expansion of community
processing of  57
expansion of community
processing of  51
expansion of community
processing of  47
expansion of community
time 0.039876699447631836
loading results in the file 'results'
the overlapping modularity is  0.5471302559234211


PS D:\Documents\NITS\Semester VI\(LAB) CS321 SNA> 
```