BT 3051 — Data Structures and Algorithms for Biology

Jul–Nov 2017

Assignment 4: Graph Theory / Essay on Algorithms

31st October 2017

Due date: 9th November, 2017 @ 17:00 **Maximum marks: 40**

Instructions: Write Python codes to solve the problems mentioned below. If you need any assistance, feel free to write to me or the TAs via Piazza (private note).
Academic Integrity: You are allowed to discuss the problems verbally with your friends, but copying or looking at codes (either from your friend or the Web) is not permitted. Transgressions are easy to find, and will be reported to the "Sub-committee for the Discipline and Welfare of Students" and will be dealt with very strictly. Mention any collaboration (discussions only!) in your solutions.
Late submission penalties: 1 second – 24 h: 20%; 24–48 h: 40%; $>$ 48h: 60%
Early submission bonuses: $>$ 24h: 5%, $>$ 48h: 10%, $>$ 72h: 20%
Evaluation: Assignments will be evaluated by the TAs within two weeks of the due date. You can check out your marks and contest them, if needed, for at most one more week post-evaluation, i.e. three weeks from the due date of the assignment.

Problem Statement

1. (15 marks) Write a Python program to create a Barabaśi–Albert network. Such a network, also known as a power-law network (because it has a power law distribution for node degrees), can be constructed by a method known as *preferential attachment.* You are allowed to use the NetworkX package, but only the basic functions — you cannot use any graph generation routines that are built in.

   How to construct the network?

   - Start with the graph G[1]; you can assume that the nodes are of `class 'int'`, and have been numbered sequentially beginning from 1.
   - In each iteration, a node *arrives*
   - Add m links to the existing nodes, at random, with probability proportional to the degree of each node in the current graph (*preferential attachment* to nodes with higher degree)
   - Stop, when n nodes have been added to the original graph G

   Plot a picture of a graph created using your code: with a starting graph with two nodes and a solitary edge, followed by the addition of 98 nodes, with one link being added each time ($n = 98$, $m = 1$). **Include this picture in your submission.**

[1] G is actually of type `<class 'networkx.classes.graph.Graph'>`

Your method should read something like this:

```
import networkx as nx

def barabasi_albert_graph (G, nNodes, mLinks):
    # G is an initial graph, which should be an nx.Graph type of object
    # nNodes is the number of nodes to add to G
    # mLinks is the number of links that get added from a new node to existing
                                                 nodes in each iteration
    G_new = G.copy()

    ### CODE FOR graph creation

    return G_new

if __name__ == '__main__':
    G = nx.Graph()
    G.add_edge(1,2)
    G_new = barabasi_albert_graph (G,98,1)

    #plot G_new
```

2. (25 marks) **Essay on Algorithms for Computational Biology**. Write a short essay (800 - 1200 words) illustrating the use of computational methods / algorithms to solve biological problems. Use examples and algorithms that were not covered in class. Make sure you clearly explain the problem being solved (motivation), a brief outline of the algorithm / methodology and finally, how the algorithm has been useful in solving the problem. For the essay, upload a single file (including figures if any, and references) `hw4.pdf`. The main idea of this assignment is to give you some technical writing practice.

How to Submit your Homework

Submission: Submit your assignment ONLY via the submission link: http://tinyurl.com/bt3051-submit. Save your solution file as `hw4.py`. Do not use a different name! Also do not send the files by e-mail — they will not be evaluated. Your submission file `hw4.py` should begin with the following information — the number of the assignment, your roll number, your collaborators' roll number(s), and approximately how much time you took to solve the problems in that part of the assignment.

```
#BT3051 Assignment 4
#Roll number: BE13B001
#Collaborators: CH12B001, EE13B001
#Time: 1:15
import networkx as nx

def barabasi_albert_graph (G, nNodes, mLinks):
    ...
```