

Bioinformatics III

Prof. Dr. Volkhard Helms

Ruslan Akulenko, Ahmad Barghash, Duy Nguyen, Christian Spaniol
Winter Semester 2013/2014

Saarland University
Chair for Computational Biology

Exercise Sheet 1

Due: October 25, 2013 13:15

1 Introduction to Python and some Network Properties

In order to keep the complexity down for this assignment, we focus on the random network and provide several hints and code stubs to ease your the work in respect of the upcoming exercises. Therefore this assignment is split into separate modules which can (and have to) be reused in later assignments.

General remarks

Since first couple of assignments are based on each other, it is recommended you visit the tutorials if there are any problems early on. Otherwise you are likely to encounter difficulties when working on the next assignment sheets.

Python

For the programming tasks of the assignments the “*Python*” scripting language is used. The tutorial will use version 3. The aim of this assignment is to learn basic Python concepts and to get familiar with network properties.

You can find documentation on Python online in the “documentation” section of www.python.org. The most valuable documents are the classic tutorial by Guido van Rossum and the “Python Library Reference”.

Submission

- (a) You are advised to work in groups of *two* people. If necessary, we will suggest teammates after the first assignment has been submitted. You may submit your solutions in english or german language.
- (b) Submit your solutions on paper, hand-written or printed at the *beginning* of the lecture in the lecture hall or in Room 3.01, both E2 1. Or alternatively, you may send an email with a **single** PDF attachment to christian.spaniol@bioinformatik.uni-saarland.de. Also attach source code, if appropriate. Late submissions will not be considered.
- (c) Include source code listings into the submitted document, we will **not** merge and layout your source code. If relevant sources are missing in the exercise sheet, they will not be graded.
- (d) Feel free to use the L^AT_EX-template provided in the supplementary material.
- (e) Do not forget to mention your names/matriculation numbers. :)

Discussion of this exercise will be on Monday, October 28th at 12:00 c.t. in the lecture room (E2 1 007).

Exercise 1.1: Constructing The Network (50 Points)

We start building a simple data structure that represents a network in general. Therefore we (a) construct a “Node”-class that represents a single node in the network and stores its links to other nodes, (b) define an abstract superclass, which can be used to derive our different network types from, and (c) implement a subclass that actually specifies how to build a random network.

We provide class stubs that you may use as guideline to help those getting started who are not familiar with the Python syntax at all. Most method implementations are supposed to not exceed 1-2 lines. Feel free to extend your interface as you need.

- (a) Implement the missing methods of the Node-class that is provided in the supplementary material.
- (b) We need a network class that stores all nodes for a network. Use a Python-Dictionary to store all nodes in a $\text{key} \rightarrow \text{node}$ fashion within a class variable called `self.nodes`.

Your task is to write an abstract network class using the API given in the supplementary material.

Hint:

- An example how to use python dictionaries:

```
# init dictionary
nodes = {}

# create node with id 0
node = Node(0)

# add entry to dictionary
nodes[node.id] = node
```

- (c) Implement the algorithm given in the lecture to set up a random graph. Start from a given number of nodes and add unique links stepwise. Extend the AbstractNetwork from (b) and save it to “RandomNetwork.py”:

```
0 from AbstractNetwork import AbstractNetwork
  from Node import Node

  class RandomNetwork(AbstractNetwork):
    """
5    This is the specific random network implementation,
    derived from the AbstractNetwork
    """
    def __createNetwork__(self, amount_nodes, amount_links):
    """
10   Create a random network
       1. ...
       2. ...
    """
    ...
```

Hints:

- When setting a link for two nodes i and k , do not forget the entry for $k \rightarrow i$.

Exercise 1.2: Degree Distribution (25 Points)

- (a) Write a class that determines and prints the degree distribution for a network created above. Again, a stub is given in the supplement.

Hints:

- First determine the biggest number of links that occurs in your network and initialize a list of that size with zero values. An example to do this with 10 entries is:

```
histogram = [0] * 10
```

This array then holds the degree distribution.

- Now loop over the network list and increment the cells indexed with the corresponding degree.
 - Normalization?
- (b) Write a program to verify that the degree distribution obeys the Poisson distribution $P(k)$ with a mean value of λ :

$$P(k) = \frac{\lambda^k}{k!} e^{-\lambda}$$

Calculate λ from the numbers of nodes and links and determine the Poisson distribution for a sufficiently large range of k .

Hint: In case you encounter numerical problems calculating the factorial, consider an iterative solution, e.g.

$$P(0) = e^{-\lambda}, P(1) = \frac{\lambda}{1} \cdot P(0), \dots, P(n) = \frac{\lambda}{n} \cdot P(n-1)$$

- (c) Use the provided script (`createNetworks.py`) that sets up the following networks (each given in the form nodes/links) and stores the degree distribution for each network in a separate file.

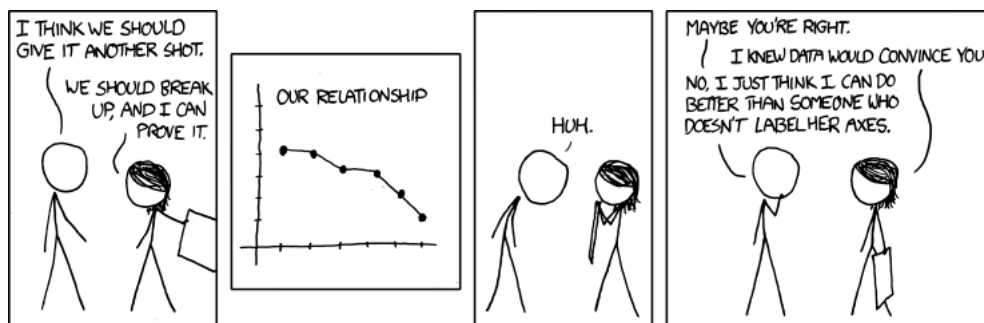
Plot 1:	50/100	500/1000	5000/10000	50000/100000
Plot 2:	20000/5000	20000/17000	20000/40000	20000/70000

Plot $P(k)$ together with the degree distributions of the random networks.

Describe both plots and *explain* the difference (or the trend) between the different parameter sets.

Hints:

- You can use “*Gnuplot*” to plot the graphs and adapt the script provided in the supplementaries.
- Do not forget to label the axes! (This hint applies to the entire lecture.)



Exercise 1.3: Classify Real-World Network Examples (25 Points)

Characterize (with a short explanation) the following examples of networks into the following categories: random, scale-free, hierarchic, and clustered. Are they directed, undirected? Some of the examples might fit into more than one category. If so, explain your choice.

- (a) File sharing such as ...
 - ... Rapidshare, Sockshare, etc.
 - ... Bittorrent
- (b) Street networks in ...
 - ... Germany
 - ... USA
- (c) your own social network ...
 - ... when you were in school (living at home)
 - ... now as a student
- (d) online social network services such as ...
 - ... Twitter
 - ... Facebook
- (e) Broadcasting networks like ...
 - ... Cable television
 - ... Satellite television

Have fun!