CS-E4600 — Programming project

November 13, 2017

Due: Friday, Dec 29, 2017, midnight

Instructions

The deliverable for this project assignment is: (1) a writeup; (2) all the source code and scripts that you will develop. More detailed instructions on the format of this deliverable is given below.

The deliverable should be returned via `mycourses.aalto.fi`

You can work on the project individually or in two-person teams. You can find your own teammate or ask us to find one for you. In all cases, send an email to Antonis (`antonis.matakos@aalto.fi`) by Monday, Nov 20, clearly indicating your intention: (*i*) you prefer to work on the project alone; (*ii*) you want to work in a team; also mention who are the team members; one email per team is sufficient; (*iii*) you want to work in a team and you need some help to find a teammate.

The objective of the project is to obtain an in-depth understanding of graph-mining algorithms. You will get the chance to experiment with algorithms that compute distances on large graphs using randomized algorithms.

The amount of code required for the project is not a lot, however, you may have to think carefully about design choices and you will have to perform experiments with different settings. Do not let the project for the last moment. Start early!

It is also possible to work on a different programming project than the one proposed here. In this case, you should propose your own project. Your project needs to be related to the topic of the course, *algorithmic methods for data analysis*, it needs to be of equivalent difficulty to the one described here, it should not be already implemented, and it should not be used in another course.

If you wish to propose an alternative project, you should write a project description, send it to the course instructors, discuss it with them, and have it approved. The deadline to propose your own project is Monday, Nov 20.

Remember that there is a budget of 5 late days, which you can use in any way you want for the three homeworks and the project. Weekend days count as late days.

Project description

We will use networks from the Stanford Network Analysis Project (SNAP)

`http://snap.stanford.edu/data/index.html`

in particular, we will work with the following social networks, listed in increasing size

`wiki-Vote` : 7 115 nodes
`soc-Epinions1` : 75 879 nodes
`ego-Gplus` : 107 614 nodes
`soc-Pokec` : 1 632 803 nodes

All networks are directed, but they can also be treated as undirected by ignoring the edge directions.

For our analysis we are focusing on the largest connected component; all other connected compenents are dropped. When working with the directed version of a network we consider the largest *strongly* connected compenent, while when working with the undirected version we consider the largest *weakly* connected compenent.

Q0.1: Download the networks listed above from SNAP and process them to obtain the largest strongly connected component (LSCC) and the largest weakly connected component (LWCC). For each network report the size of the LSCC and LWCC in terms of number of nodes and number of edges.

For all the networks listed above we are interested in computing the following statistics

1. *median distance*
2. *mean distance*
3. *diameter*
4. *effective diameter*

Recall that all these network statistics are computed by considering the shortest-path distances between all pairs of nodes in the network (either directed or undirected), and then taking the median, mean, max, and 0.9-quantile,[1] respectively, of the distribution.

Task 1 [Exact computation]

Design and implement algorithms that compute *exact* network statistics for all listed networks.

Consider both the directed and undirected version of the networks; so in total you have to report numbers for 2 versions $\times$ 4 networks $\times$ 4 statistics = 32 cases.

Q1.1: Report the network statistics computed by your algorithm for each different case.

Q1.2: Report the running time of your algorithm for each different case.

Hint: A convenient way to report your results for questions **Q1.1** and **Q1.2** is by using two (for the directed and the undirected case) 4×4 tables.

Start with the smallest network and proceed to the largest. If computation is too slow to finish in a reasonable time, it is OK to abandon and report N/A. However, you should try to optimize your code and try to compute statistics for as many cases as possible.

[1]The ϕ-quantile of set of n items, with $0 \leq \phi \leq 1$, is obtained by sorting the items in increasing order and returning the $\lfloor \phi n \rfloor$-th item in that order.

Task 2 [approximate computation]

You should now implement approximation algorithms for computing network statistics.

You should consider the following three approximation schemes:

1. Sample random pairs.
2. Sample random sources and perform a complete breadth-first search (BFS) for each source
3. Use the Flajolet-Martin algorithm discussed in class [1].

There are many papers that apply the Flajolet-Martin algorithm, variants, or other similar probabilistic schemes. A classic paper is the one by Palmer et al. [2]. You are free to pick and implement any of the existing schemes, however, you should provide a reference to the precise scheme that you are implementing.

All three approximation schemes have a parameter (number of samples or repetitions) that controls the approximation accuracy.

Q2.1: For each of the approximation schemes fix the accuracy parameter (number of samples or repetitions) and report the computed network statistics. Compare with the exact network statistics for the cases that it has been possible to compute those.

Q2.2: Consider the largest network for which you have computed exact network statistics. For that network compute the approximate network statistics for different values of the accuracy parameter. Plot the approximate network statistics versus the accuracy parameter and check whether your estimates converge to the ground truth, and if yes, how fast.

Q2.3: Discuss your findings and observations. (*i*) How good is the approximation? (*ii*) Which of the three approximation methods is better and why? (*iii*) What is the recommended value of the accuracy parameter, according to your experiments?

Deliverable

You should provide the following deliverable in a zipped package, using the following naming convention:

```
LastNameOfTeammate1_LastNameOfTeammate2/report.pdf
LastNameOfTeammate1_LastNameOfTeammate2/source/
```

Report. You should describe the methods you implemented, your design choices, your experimental methodology, and your findings.

Source code. Provide all the source code and scripts, and include a README.txt file explaining how to use it. The program should be able to run locally using standard packages.

References

[1] P. Flajolet and G. N. Martin. Probabilistic counting algorithms for data base applications. *Journal of computer and system sciences*, 31(2):182–209, 1985.

[2] C. R. Palmer, P. B. Gibbons, and C. Faloutsos. ANF: A fast and scalable tool for data mining in massive graphs. In *Proceedings of the eighth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 81–90. ACM, 2002.