

M2 – NSD (Practical Work 6 - Sessions 10,11)

Efficient measurements using link prediction

Maximilien Danisch, Marwan Ghanem, Lionel Tabourier
(adapted from Fabien Tarissan)

The purpose of this practical work is to simulate different measurement strategies (as seen in the course on link queries) and compare their efficiency, whether it be in terms of their ability to extract rapidly a graph sample reasonably large or to extract representative samples. To do so, we will test the strategies on a real-world dataset extracted from the Flickr social network (August 2006).

1 Preliminaries

Exercise 1 — *Properties of the original networks*

Download the files `FLICKR` and `FLICKR-TEST` from the usual webpage : http://lioneltabourier.fr/teaching_en.html. The file `FLICKR-TEST` is a sub-sample of the graph `FLICKR` and contains only 500 nodes. It will be useful to test your programs before running them on the large dataset.

Using the programs written for previous practical works, compute the characteristics discussed in the course on link queries of those graphs : m , δ , degree distribution, average degree, maximum degree, cc and tr . Do they exhibit usual properties of real networks? (justify your answer)

2 Preparing simulations

Exercise 2 — *Clear the input*

Write a program `simul` that takes as input a graph and duplicates the structure while removing all information related to the links.

From now on, we will distinguish in `simul` the structure `g-original` standing for the original graph from the structure `sample` denoting the sample extracted after simulation.

Exercise 3 — *Measurement primitive*

Add a function that, given two nodes of the graph, tests if a link exists between those two nodes in `g-original` and modify the state of the `sample` structure accordingly.

Exercise 4 — *Output of the simulations*

Add to the program a variable to track how many tests (or tries) have been made and add a function that displays the detected links along with the number of tested links until then (i.e. each line will have the format (t, n_1, n_2) where t stands for the number of tries made so far and n_1 and n_2 are the two nodes of the link detected).

In the rest of the PW, we will refer to a file in such a format as `FILE-RES`.

Exercise 5 — *Worst, best and random strategies*

Write a program `analyze` which compute the efficiency of the best and the worst strategy, given the number n of nodes, m of links and t of tries. Complete the program by adding the efficiency of a purely random strategy.

Exercise 6 — *Efficiency*

Write a program that compute the absolute, the relative and the normalised efficiency from a simulation. The program will take as input a file FILE-RES and two integers n and m standing respectively for the number of nodes and links of the original graph.

Add to your program the elementary metrics assessing the quality of the prediction strategy : *precision*, *recall* and *F1-score*.

3 Random strategy

Exercise 7 — *Implementation*

Implement the `Random` strategy in your program `simul` and simulate it (test on FLICKR-TEST using 50 000 tries for instance).

Exercise 8 — *Evolution*

Plot the evolution of the number of detected links as a function of the number of tested links. Comment the plot.

Exercise 9 — *Efficiency*

Using your program `analyze`, compute the different efficiency values for the `Random` strategy. Does it sustain your analysis above ?

4 Simple strategies

Exercise 10 — *Ordering the nodes*

Add to `simul` a function that order nodes according to their current degree (in decreasing order).

Exercise 11 — *Complete strategy*

Add a function that, given a node, tests all untested links with any other node of the graph. Use this function to implement the `Complete` strategy and simulate it¹.

Exercise 12 — *Ordering the links*

Applying a similar principle, add a structure that allows to represent a link and add a function that orders the links according to the sum of the degree of the involved nodes (in decreasing order).

Exercise 13 — *TBF strategy*

Implement the TBF strategy (taking only nodes whose degree is higher or equal to 1).

5 Evaluation

Exercise 14 — *Comparison*

Using the result of the simulations of the `Complete` and `TBF` strategies, study the evolution of the number of detected links as a function of the number of tested links and compare it to the `Random` strategy. Compare also the efficiency metrics proposed in Exercise 9 for the three strategies. What is your conclusion ?

Exercise 15 — *Random phase*

Use different values for the random phase in order to assess the impact on the random phase on the efficiency of the different strategies. What do you conclude ?

1. For FLICKR, use a random phase until you find 0.1% of the existing links for instance.

6 Refined strategies

Exercise 16 — *Mixing strategies*

Test different associations of simple and random strategies. Compare the obtained results with the previous strategies. What is your conclusion ?

Exercise 17 — *Representative sample*

Use the programs written for previous PW in order to test the representativeness of the extracted samples using the different strategies. Does it change your previous conclusions ?

7 Submission of the project

You have until January 7th of 2019 to submit the results of your project. The form of the submission is an archive that contains :

- a report (in pdf format) describing the results and your analysis
- the source of your program(s) that can be executed in order to reproduce the results.

Please name the archive name.zip (or .tar.gz) and send it to lionel.tabourier@lip6.fr.

The report. The report describes the results that you have obtained (plots, statistics) and the interpretation that you have made. You should address the questions asked explicitly in the exercises. In particular, you are expected to :

1. comment on the evolution of the different strategies,
2. compare the efficiency of the different strategies,
3. study the impact of the random phase in the simple strategies,
4. evaluate the quality of the samples extracted from the different strategies.

In addition to the exercises proposed in this document, you are encouraged to propose your own strategy, integrate it in the simulation framework developed in this project and add a comparison with the other strategies.

The program. Your program can be written in the programming language of your choice, but you have to provide everything required to execute it. It means in particular :

- a Readme file indicating all requirement and how to compile/execute the code,
- a command line that work on a small toy example you provide.