Case Study 1 : Graph Structure & Modelling

Broder et al, Power-Law & Price's Model

Author : *Paula Dwan*
Email : *paula.dwan@gmail.com*

Student ID : *13208660*

Course : *MSc Advanced Software Engineering*
Module : *COMP-47270 Computational Network Analysis and Modelling*

Lecturer : *Dr. Neil Hurley*
Email : *neil.hurley@ucd.ie*

Due Date : *11 May 2015*

TABLE OF CONTENTS

1 CASE STUDY REQUIREMENTS

Determine the qualitative nature of the networks you are studying and write up a report.

Do some of the following:

1.	For the directed networks, sketch the **Broder et al** picture of the network. Number of nodes in *(SCC)* strongest connected components, IN, OUT, and other sections of the network.
2.	Fit a line to a log-log plot of the degree distribution. Compute the slope of this line to determine the parameter β of the power-law degree distribution model.
3.	Simulate the Price model to generate a network of n nodes for a particular power-law parameter α where $\alpha = \beta - 1$. Compute all of the above parameters for the **Price model.** Which (real-life) network/s does the Price's Model best represent?

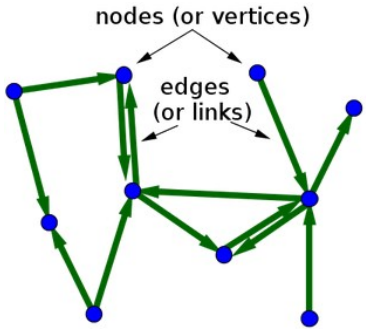
1.1 UNDERSTANDING OF REQUIREMENTS FROM CLASS DISCUSSIONS

1.	Choose three directed networks from http://snap.stanford.edu/data/.
2.	Generate Broder et al : not directly possible using NetworkX and python, thus need to calculate number of nodes exist as SCC, IN, OUT, Tendrils, Tubes and Disconnected.
3.	Plot the results for each [$x - y$] → [*log(rank) – log (deg)*]. Compute the slope of the line plotted to get parameter $\boldsymbol{\beta}$ of the power-law degree distribution model. We know that the slope $\boldsymbol{m = -\beta}$, thus $\boldsymbol{\beta = -m}$ (slope x -1) *Aside: For website analysis, say we have* $\boldsymbol{n}$ *= number of web-sites visited by* $\boldsymbol{u}$ *users. Then applying power-law, we have* $\boldsymbol{n = C\ u^{-\alpha} \rightarrow \log(n) = \log(C) - \alpha \log(u)}$. *Thus a power-law with exponent* $\boldsymbol{\alpha}$ *is represented as a straight line having a slope* $\boldsymbol{-\alpha}$. Calculate power-law (ultimately $\boldsymbol{\beta}$) for ***rank -v- in-degree*** and for ***rank -v- out-degree*** for each dataset.
4.	Simulate the Price's Model to generate a network of n nodes for power-law parameter $\boldsymbol{\alpha}$ where $\boldsymbol{\alpha = \beta - 1}$, and calculate degree distribution for Price's Model also.
5.	Which real-life networks best matches the Price's Model?

2 INTRODUCTION

2.1 WHAT IS A DIRECTED NETWORK?

Source : http://mathinsight.org/directed_graph_definition



A directed graph is graph, consisting of a set of nodes (aka. vertices) that are connected together. The connections are directed from one node to another. A directed graph is sometimes called a digraph or a directed network.

In mathematical terms, this is represented by ***G = (V,A),*** where :

- **V** → set, whose elements are called vertices or nodes,
- **A** → set of ordered pairs of vertices, called arcs, directed edges, or arrows (and sometimes simply edges with the corresponding set named E instead of A).

2.2 DIRECTED GRAPHS USED

Each of the directed graphs chosen deal with Social media (Facebook, Twitter, Google+) or were included to vary the number of nodes present (High-energy physics citation network,). For more information on each, please see the section : References – Datasets.

3 BRODER ET AL

3.1 OVERVIEW OF BRODER ET AL

In 2000, Broder and various colleagues conducted a large-scale analysis of the web. They concluded that as a directed graph there was a recognisable structure in place that of a bow-tie (see following figure.) In short, if a page x is connected to page y via a directed edge then a hyper-link connects page x to page y.

3.1.1 COMPONENTS : BOW-TIE MODEL

Component	Explanation
SCC	Nodes which are part of the 'strongly connected component' (SCC). Each nodes has *IN→edges* from IN, SCC and *OUT←edges* to OUT or SCC. Each node in SCC has one or more paths to every other node in SCC.
IN (→)	Nodes that are ***up***-stream of the SCC (i.e.: these nodes have *IN→edges* from other IN only and *OUT←edges* to other nodes in IN, SCC, Tendrils or Tubes).
OUT (←)	Nodes that are ***down***-stream of the SCC (i.e.: these nodes have *OUT←edges* to other OUT only and *IN→edges* from other nodes in OUT, SCC, Tendrils or Tubes).
Tubes	Nodes that have *IN→edges* from IN or other pages in Tubes and *OUT←edges* to nodes in Tubes or OUT.
Disconnected	Nodes that are basically outside of the bow-tie structure. Nodes have no *IN→edges* from and no *OUT←edges* to any components. These pages may be linked to each other.
Tendrils	Nodes that are can accessible from IN only or only have *OUT-edges* to OUT. These nodes do not pass through SCC.

3.1.2 How Calculated Using Python

There is no one function in NetworkX or in python to evaluate a directed graph and confirm what node is part of what component, therefore the nodes and edges must be examined to see what connections exist and in what direction.

Component	Explanation
SCC	Use NetworkX function : `strongly_connected_components(G)` which generates the nodes in strongly connected component of the graph provided as a list of nodes. This sub-graph ***SCC*** **[1]** is then used as the source for ***IN*** and for ***OUT.***
IN	Use NetworkX function : *`all_pairs_shortest_path_length(G)`* which computes the shortest path lengths between all nodes in G. ***IN*** nodes will have the longest path of all nodes. **[2]**
OUT	Once IN is excluded from SCC sub-graph, nodes with shortest path are the ***OUT*** nodes. The NetworkX function : *`shortest_path(G)`* is used to calculate this. **[3]** SCC now excludes IN and OUT nodes. **[3]**
	Remaining nodes are used to calculate the remaining sets :
Tubes	IN Nodes which have a path to OUT nodes but which do not intersect SCC nodes. The NetworkX function : *`shortest_path(G)`* is used to calculate this, combined with all nodes in OUT. **[4]**
IN-Tendrils	IN Nodes which do not intersect with OUT nodes not SCC nodes. Thus if nodes are part of IN nodes but are not part of SCC (already excluded) and not part of Tubes (just calculated) then the nodes is IN-Tendril. **[5]**
OUT-Tendrils	Nodes which which flow into OUT but do not connect to SCC directly. **[6]**
Disconnected	All remaining nodes. ***[7]*** Alternatively, could use density = 0, i.e.: no connections.

The following is how Broder is calculated using python, calculation of each component is referenced from ***[1]*** to ***[7]*** :

```
def calc_broder_values(g):
    logging.info(cs_ref, 'calculate values for Broder bow-tie')
    func_intro = "\nCALCULATE BRODER  BOW-TIE VALUES ... \n"
    print func_intro
    with open(dest_file, "a") as dat_file:
        dat_file.write(func_intro)

    bt_dict = {}

    scc = nx.strongly_connected_components(g)           [1]

    shortest_path = nx.all_pairs_shortest_path_length(g)
    inc = {n for n in g.nodes() if scc in shortest_path[n]}
    inc -= scc                                          [2]

    outc = set()
    for n in scc:
        outc |= set(shortest_path[n].keys())
    outc -= scc                                         [3]

    tubes = set()
    in_tendrils = set()
    out_tendrils = set()
    disconnected = set()
    remainder = set(g.nodes()) - scc - inc — outc

    inc_out = set()

    for n in scc:
        inc_out |= set(shortest_path[n].keys())
```

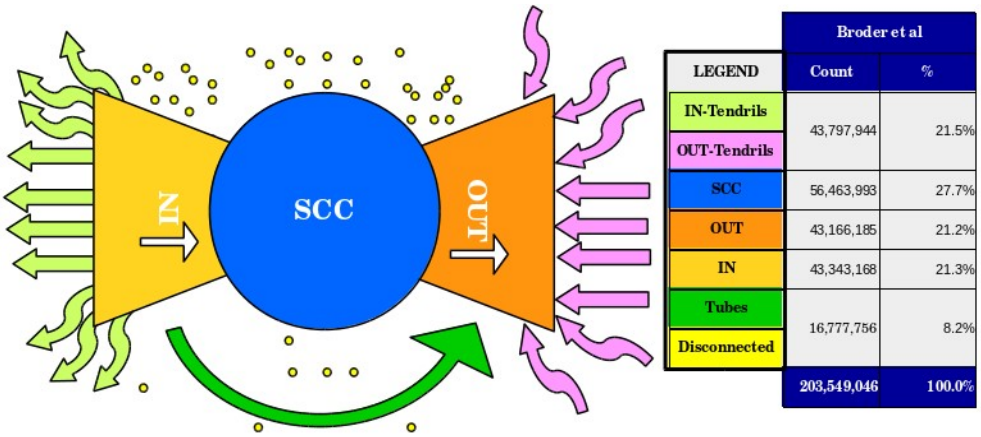
```
    inc_out = inc_out - scc - inc - outc
    for n in remainder:
        if n in inc_out:
            if set(shortest_path[n].keys()) && outc:
                tubes.add(n)                          [4]
            else:
                in_tendrils.add(n)                    [5]
        elif set(shortest_path[n].keys()) & outc:
            out_tendrils.add(n)                       [6]
        else:
            disconnected.add(n)                       [7]

bt_dict.update({'IN-tendrils': len(in_tendrils) }) # return number of each type present
bt_dict.update({'IN': len(inc) })
bt_dict.update({'SCC': len(scc) })
bt_dict.update({'OUT': len(outc) })
bt_dict.update({'OUT-tendrils': len(out_tendrils) })
bt_dict.update({'Tubes': len(tubes) })
bt_dict.update({'Disconnected': len(disconnected) })

    return bt_dict
```

3.2 Results of Broder et al

3.2.1 Results → Broder et al, (2000)



LEGEND	Broder et al Count	%
IN-Tendrils	43,797,944	21.5%
OUT-Tendrils		
SCC	56,463,993	27.7%
OUT	43,166,185	21.2%
IN	43,343,168	21.3%
Tubes	16,777,756	8.2%
Disconnected		
	203,549,046	100.0%

Figure 1 : Broder et al bow-tie diagram including values from 2000 study

3.2.2 Results → Expected

I would expect that even when the number of nodes increases significantly, that the ratio would remain somewhat constant and in line with what Broder et al noted in 2000, especially in medium to large sized datasets. They examined a network with over 200 million nodes, the largest I look at is ***Google+*** at 107,614 nodes. For all datasets examined, SCC % is as follows :

Details	Facebook	cit-HepTh	cit-HepPh	Twitter	Google +
Total Nodes	4,039	27,770	34,546	81,306	107,614
Nodes in largest SCC	4,039	7,464	12,711	68,413	69,501
SCC %	**100%**	**27%**	**37%**	**84%**	**65%**

The significant high SCC % is particularly obvious in social networks where there is a high degree of communication amongst users (nodes). The two citation datasets are more similar to the results obtained by Broder et Al of Q2 percentage (25% – 50%). In fact, ***citHepTh*** matches the results obtained by Broder et al. Further, examination of these datasets would be needed to ascertain why the significant difference.

I would also expect that even with a exponential increase in the number of nodes that the diameter would remain relatively small at less 8 to 15. This is confirmed by the data provided by snap (see section : Statistics – Datasets). In the case of the chosen datasets, ***citHepTh*** has the largest diameter of 13 (academic citation dataset) and Google+ has smallest diameter of 6 (social network). This is not unexpected, a large social network with many users would have a smaller diameter than would and a much smaller (relatively) academic citation network. Diameter for each dataset is :

Details	Facebook	cit-HepTh	cit-HepPh	Twitter	Google +
Diameter	8	13	12	7	6

Again, the citation datasets have the largest diameter and the social network datasets have the smallest.

3.2.3 RESULTS → ACTUAL

Unfortunately, I did not succeed in getting *cs1_broder.py* to work. I believe that the methodology behind the code is sound but I was prevented from obtaining any results due to lack of familiarity on my part in python. This is something I will work on in the next while.

4 PLOT THE DEGREE DISTRIBUTION

4.1 OVERVIEW

A power-law is a direct relationship between the rank and degree or the number of edges and the number of nodes in the directed graphs chosen. Here, we evaluate the degree-distribution for each graph.

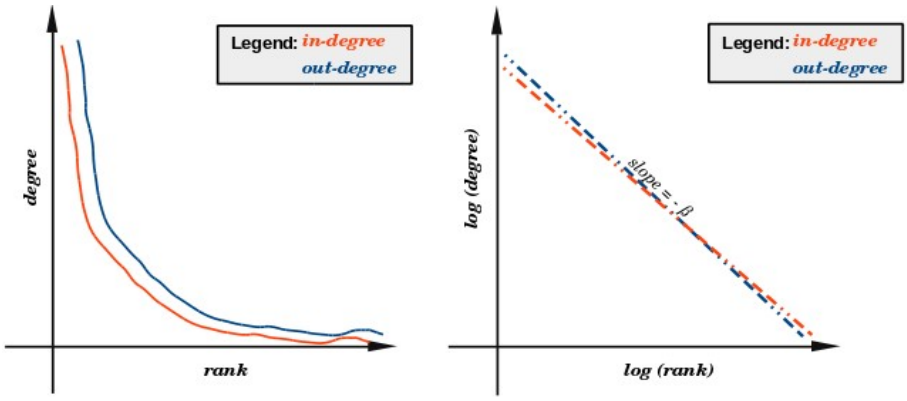


Figure 2: Power-Law Distribution for in-degree and out-degree.

4.1.1 COMPONENTS : LOG-LOG PLOT

Component	Explanation
in-degree	The number of edges ending at a node is the in-degree of that node.
out-degree	The number of edges originating at a node is the out-degree of that node.
parameter β	We know that the slope $m = -\beta$, thus $\beta = -m$ (slope x -1)

4.1.2 HOW CALCULATED USING PYTHON

Component	Explanation

in-degree	First calculate number of sorted in-degrees and the count of each in our graph G:
	`in_degrees = graph.in_degree()`
	`in_values = sorted(set(in_degrees.values()))`
	`in_rank = [in_degrees.values().count(x) for x in in_values]`

Component	Explanation

out-degree	And also, calculate number of sorted out-degrees and the count of each in our graph G:
	`out_degrees = graph.out_degree()`
	`out_values = sorted(set(out_degrees.values()))`
	`out_rank = [out_degrees.values().count(x) for x in out_values]`

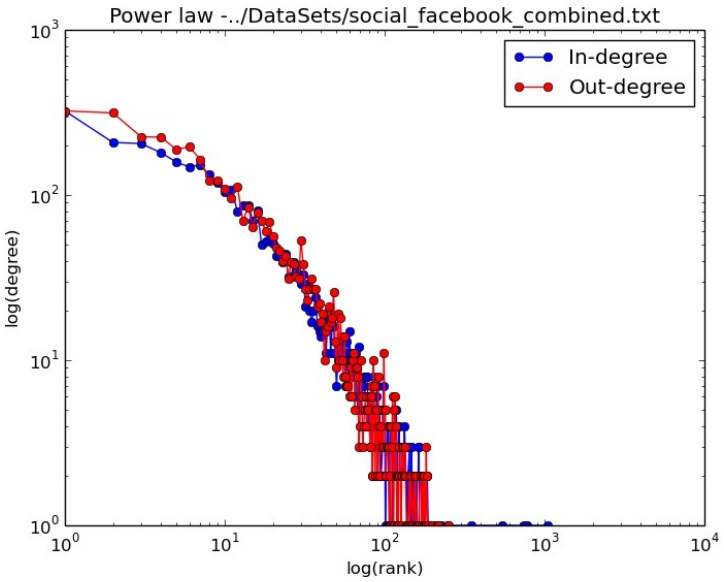
Slope	Then calculate the slope ***m,*** I initially tried to use ***numpy.polyfit*** for all (x,y) for in_degree and out_degree but kept getting errors related to division by zero, so I removed (x, y) at position zero:
	`xi = in_values`
	`xi.pop(0)`
	`yi = in_rank`
	`yi.pop(0)`
	`slope, intercept = np.polyfit(np.log(xi), np.log(yi), 1)`
	I decided not to plot the line for the slope as it cause the detailed graph to become even more cluttered and perhaps illegible.

Plot results	Then for in-degree and out-degree, plot the results :
	`plt.figure()`
	`plt.loglog(out_values, out_rank, 'bv-')`
	`plt.loglog(in_values,in_rank,'ro-')`
	`plt.legend(['In-degree', 'Out-degree'])`
	`plt.title('Power law' + src_file)`
	`plt.xlabel('log(degree)')`
	`plt.ylabel('log(rank)')`
	`plt.savefig('plots/power_law.png')`
	`plt.close()`

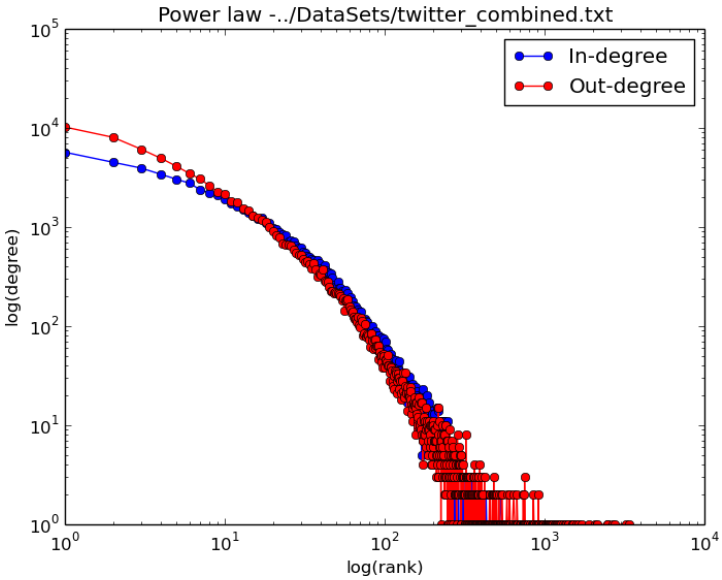
parameter β	Ultimately β was calculated using : `beta = slope * -1`

4.2 RESULTS

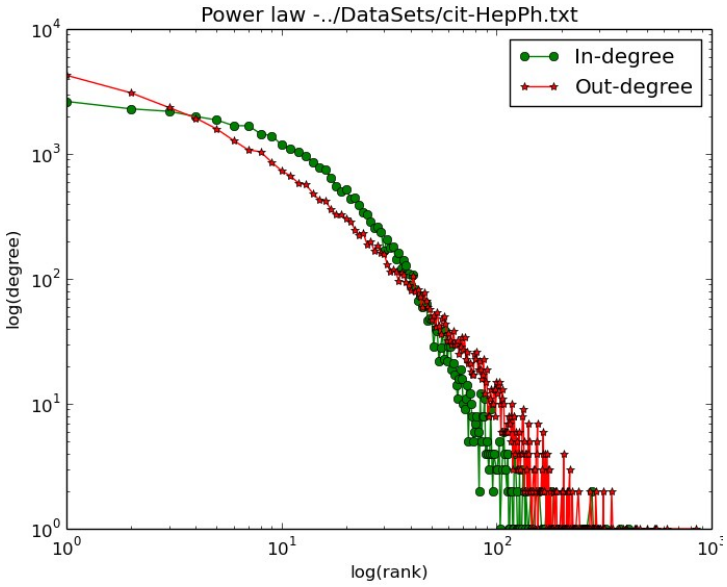
4.2.1 POWER LAW DISTRIBUTION : SMALL GRAPH → ≤ 10K NODES (FACEBOOK)



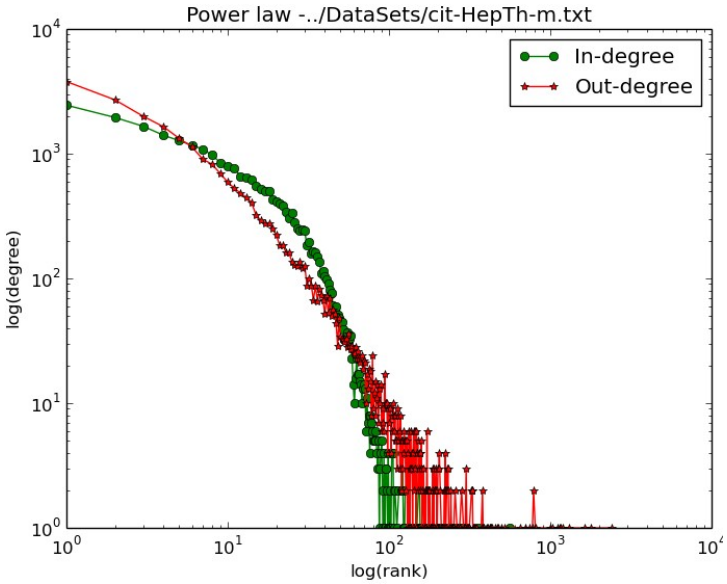
4.2.2 Power Law Distribution : Medium graph→ 10k to 100k nodes (Twitter)



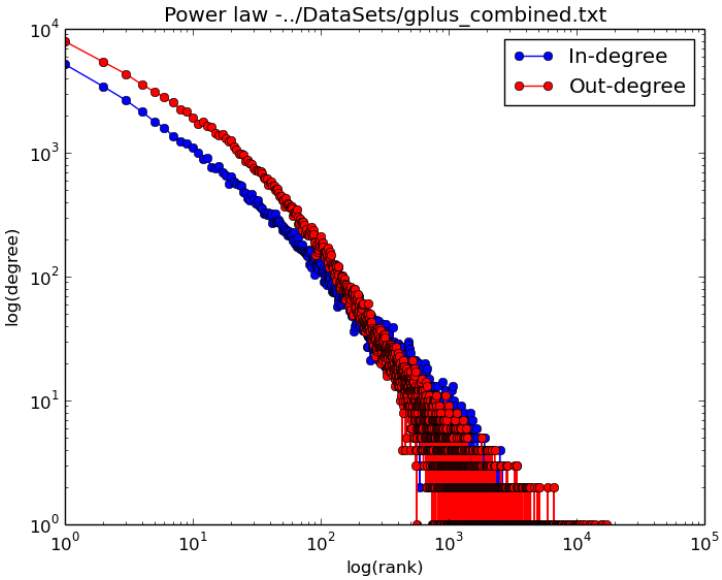
4.2.3 Power Law Distribution : Medium graph→ 10k to 100k nodes (cit-HepPh)



4.2.4 Power Law Distribution : Medium graph→ 10k to 100k nodes (cit-HepTh)



4.2.5 Power Law Distribution : Large graph → > 100k nodes (Google +)



4.2.6 Parameters Alpha (a) & Beta (b) for Power-Law Degree Distribution Model

Datasets		Slope	Beta (β)	Alpha (α)
Small (≤ 10k) : Facebook	in-degrees	-1.47695159255	1.47695159255	0.47695159255
	out-degrees	-1.32682677798	1.32682677798	0.32682677798
Medium (10k to 100k) : cit-HepPh	in-degrees	-1.92957821097	1.92957821097	0.92957821097
	out-degrees	-2.22457143373	2.22457143373	1.22457143373
Medium (10k to 100k) : cit-HepTh	in-degrees	-1.58351423392	1.58351423392	0.58351423392
	out-degrees	-2.19225822089	2.19225822089	1.19225822089
Medium (10k to 100k) : Twitter	in-degrees	-1.74631831806	1.74631831806	0.74631831806
	out-degrees	-2.05677900841	2.05677900841	1.05677900841
Large (> 100k) : Google +	in-degrees	-1.3051390885	1.3051390885	0.3051390885
	out-degrees	-1.37524044036	1.37524044036	0.37524044036

5 Price's Model

5.1 Overview

While the Broder et al Model evaluates the physical properties of the three directed graphs chosen, Price's Model (1976) evaluates how social networks grow. Derek de Solla Price used published academic papers as his nodes. He proposed that the increase in citations by other authors should be proportional to the number of citations the paper already has. He did add a constant factor for a newly published paper which initially will have zero citations. This is basically preferential advantage → *'the rich get richer'*. This is also the first known example of a scale-free network, where we expect that regardless of the number of nodes in the graph the power-law remains constant.

5.1.1 COMPONENTS : PRICE'S MODEL

Component	Explanation
citations	How many times a paper was cited (referred to) by authors of other papers.
Age of paper	A new paper has 0 citations, however that new paper will have some references to existing papers already. An existing paper will also have some citations / references to existing papers and will be cited by existing and also new papers thus increasing number of citations over time.

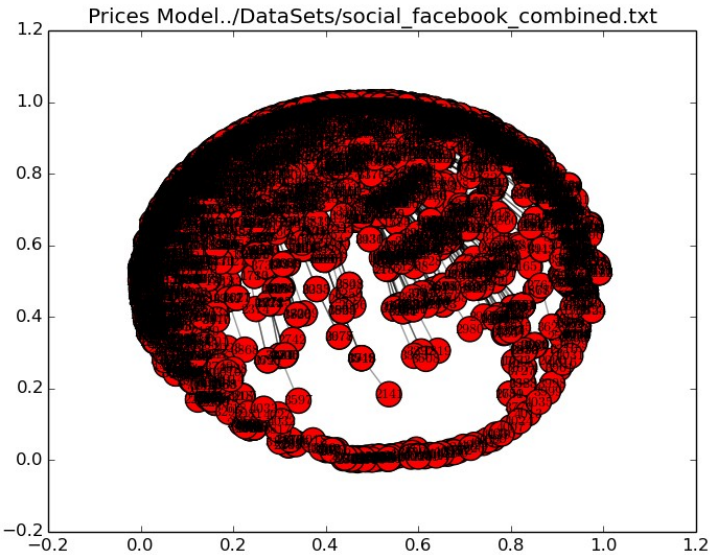
5.1.2 HOW CALCULATED USING PYTHON

Component	Explanation
nodes	Add the same number of nodes as per the datasets used for Broder and Degree distribution.
edges	in-degrees → how many times this node(paper) was 'cited' out-degrees → how many times this node (paper) 'cited' other notes (papers) Both in-degrees and out-degrees were emulated. The resulting graphs were mapped using Matlab.

I used the same number of nodes as the actual datasets but used Price's Model to generate edges and thus had difference in-degrees and out-degrees.

5.2 *RESULTS*

5.2.1 PRICE'S MODEL : SMALL GRAPH → ≤ 10K NODES (*N_MAX = 4,039)*



5.2.2 PRICE'S MODEL : MEDIUM GRAPH→ 10K TO 100K NODES (*N_MAX = 27,770)*

This took over three hours to complete. It seems that creating a network using Prices Model as coded for a network having ≥ 7,000 nodes is too much for my laptop to contend with in a reasonable time-frame. It would be worthwhile in completing these calculations on a more suitable machine to see if the same "trend" as noted for real -v- Price's Model. Real networks as examined in this exercise tend to have an exponent Alpha **(α)** < 1 for small or large networks, whereas medium sized networks tend to be > 1 but still < 2; and so not within the ideal range for Alpha **(α).** While the nodes and edges were created successfully, slope for in-degree and slope for out-degree were not calculated, neither were the results plotted before the laptop hanged / crashed.

5.2.3 PRICE'S MODEL : MEDIUM GRAPH→ 10K TO 100K NODES (*N_MAX = 34,546)*

Memory issues prevented this dataset from completing.

5.2.4 PRICE'S MODEL : MEDIUM GRAPH→ 10K TO 100K NODES (*N_MAX = 81,306)*

Memory issues prevented this dataset from completing.

5.2.5 LARGE GRAPH → > 100K NODES : PRICE'S MODEL (*N_MAX = 107,614*)

Memory issues prevented this dataset from completing.

5.2.6 ALPHA (A) : PRICE'S MODEL

Alpha **(α)** as calculated earlier using the degree distribution for the actual directed graphs was < 2 in all instances, thus a value of 3 was used for Price's Model computations. This produced the following Slope and Beta **(β)** results :

Datasets		Slope	Beta (β)	Alpha (α)
Small (≤ 10k) : Facebook	in-degrees	-1.63345556109	1.63345556109	0.63345556109
	out-degrees	-1.63345556109	1.63345556109	0.63345556109
Medium (10k to 100k) : cit-HepTh	in-degrees			
	out-degrees			
Medium (10k to 100k) : cit-HepPh	in-degrees	*N/A*	*N/A*	*N/A*
	out-degrees	*N/A*	*N/A*	*N/A*
Medium (10k to 100k) : Twitter	in-degrees	*N/A*	*N/A*	*N/A*
	out-degrees	*N/A*	*N/A*	*N/A*
Large (> 100k) : Google +	in-degrees	*N/A*	*N/A*	*N/A*
	out-degrees	*N/A*	*N/A*	*N/A*

5.2.7 NETWORKS BEST REPRESENTED BY PRICE'S MODEL

Price's Model best illustrates networks whose degree distribution follows a power-law. Basically, the number of connections a node increases exponentially provided that node has a large number of connections (edges) initially compared to other nodes. A typical *"rich get richer"* scenario. Internet and social networks behave in a similar manner. If a web-page has a high number of hits (is deemed popular) then the number of hits will continue to grow. Conversely, if a web-site has a low number of hits (is unpopular) it will remain so.

A straight-line or a good facsimile of one strongly suggests that a power-law applies to the dataset. The slope of line equates to the power-law exponent (alpha → α). This value is ideally in the range : $2 \leq \alpha \leq 3$, although sometimes α lies slightly outside this range. This was observed on all datasets when calculated using actual graph information and also during price's model simulations as all points "converged" on a straight-line observed in the data plots. However, it should be noted that the value obtained for alpha was less than 2 and even less than 1 for some, which falls outside the ideal range of the power-law exponent, Alpha **(α).**

The three datasets chosen (facebook, twitter and Google +) are all social networks and are all public. Price's Model usually. A social network consists of nodes; companies or individuals (commercial or personal) and some connection between them. *cs1_price.py* does not work on larger datasets (having > 50,000 nodes). This could be insufficient time being allowed but after two hours I ceased execution on the larger datasets and used an additional two small-medium citation networks to obtain a better range of results for smaller datasets. I also separately plotted the degree distribution and calculated Alpha **(α)** for these two datasets ***cit-HepTh*** and ***cit-HepPh.***

6 CONCLUSIONS

Broder et al and Price's Model, combined with Degree Distribution, all review components of social networks and the interaction between those components. Border et al evaluates the structure of those components. In 2000, when Broder and associates reviewed 200 million nodes, they concluded that there were at most 16 edges ("clicks") between one node and any other. Price's Model exhibit heavy-tailed distribution where popularity of a web-page increases exponentially. The Degree Distribution calculates the exponent Alpha ***(α)*** of the model using the slope of the model, where :

Alpha ***(α)*** = Beta ***(β)*** – 1 where Beta ***(β)*** = Slope x – 1.

The datasets used were decided on due to commonalities in nature : social networks or academic citation datasets. I would expect a larger number of connections (edges) in the social networks and this was proven when the SCC % of total nodes was calculated for each.

7 REFERENCES / ACKNOWLEDGEMENTS

7.1 *REFERENCES – NETWORKS DATASETS*

Datasets : http://snap.stanford.edu/data/index.html
The information in this section is taken directly from http://snap.stanford.edu/ and is included for informational purposes only.

7.1.1 ACKNOWLEDGEMENTS – DATASETS

Name	Size	Link
Social circles: Facebook	Small → ≤ 10k nodes	http://snap.stanford.edu/data/egonets-Facebook.html
Social circles: Twitter	Medium → 10k to 100k nodes	http://snap.stanford.edu/data/egonets-Twitter.html
High-energy physics citation network	Medium → 10k to 100k nodes	http://snap.stanford.edu/data/cit-HepPh.html
High-energy physics theory citation network	Medium → 10k to 100k nodes	http://snap.stanford.edu/data/cit-HepTh.html
Social circles: Google+	Large → > 100k nodes	http://snap.stanford.edu/data/egonets-Gplus.html

7.1.2 STATISTICS – DATASETS

Details	≤ 10k : Facebook	10k → 100k : cit-HepTh	10k → 100k cit-HepPh	10k → 100k : Twitter	> 100k : Google +
Nodes	4,039	27,770	34,546	81,306	107,614
Edges	88,234	352,807	421,578	1,768,149	13,673,453
Nodes in largest WCC	4,039 (1.000)	27,400 (0.987)	34,401 (0.996)	81,306 (1.000)	107,614 (1.000)
Edges in largest WCC	88,234 (1.000)	352,542 (0.999)	421,485 (1.000)	1,768,149 (1.000)	13,673,453 (1.000)
Nodes in largest SCC	4,039 (1.000)	7,464 (0.269)	12,711 (0.368)	68,413 (0.841)	69,501 (0.646)
Edges in largest SCC	88,234 (1.000)	116268 (0.330)	139,981 (0.332)	1,685,163 (0.953)	9,168,660 (0.671)
Average clustering coefficient (ACE)	0.6055	0.3120	0.2848	0.5653	0.4901
Number of triangles	1,612,010	1,478,735	1,276,868	13,082,506	1,073,677,742
Fraction of closed triangles	0.2647	0.04331	0.05377	0.06415	0.6552
Diameter (longest shortest path)	8	13	12	7	6
90-percentile effective diameter	4.7	5.3	5	4.5	3

7.2 CITATIONS / ACKNOWLEDGEMENTS

Reference	Acknowledgement
SNAP → datasets	Sourced from : http://snap.stanford.edu/data Please see each dataset page link provided for specific link.
Price's Model	Influenced by Dr. Neil Hurley
Broder et al	Influenced by https://github.com/lamda/bowtie
Degree Distribution	NetworkX : http://networkx.github.io/ Various howTo's : e.g.: http://stackoverflow.com/